Applied Soft Computing 52 (2017) 812–833

Contents lists available at ScienceDirect

Applied Soft Computing

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

Micro-differential evolution: Diversity enhancement and a comparative study

Hojjat Salehinejad [a,*], Shahryar Rahnamayan [a], Hamid R. Tizhoosh [b]

[a] Department of Electrical, Computer, and Software Engineering, University of Ontario Institute of Technology, 2000 Simcoe Street North, Oshawa, ON L1H 7K4, Canada

[b] Department of Systems Design Engineering, University of Waterloo, 200 University Avenue West, Waterloo, ON N2L 3G1, Canada

ARTICLE INFO

Article history:
Received 24 November 2015
Received in revised form 26 July 2016
Accepted 24 September 2016
Available online 17 October 2016

Keywords:
Diversification
Micro-differential evolution
Mutation factor
Stagnation
Premature convergence

ABSTRACT

Differential evolution (DE) algorithm suffers from high computational time due to slow nature of evaluation. Micro-DE (MDE) algorithms utilize a very small population size, which can converge faster to a reasonable solution. Such algorithms are vulnerable to premature convergence and high risk of stagnation. This paper proposes a MDE algorithm with vectorized random mutation factor (MDEVM), which utilizes the small size population benefit while empowers the exploration ability of mutation factor through randomizing it in the decision variable level. The idea is supported by analyzing mutation factor using Monte-Carlo based simulations. To facilitate the usage of MDE algorithms with very-small population sizes, a new mutation scheme for population sizes less than four is also proposed. Furthermore, comprehensive comparative simulations and analysis on performance of the MDE algorithms over various mutation schemes, population sizes, problem types (i.e. uni-modal, multi-modal, and composite), problem dimensionalities, and mutation factor ranges are conducted by considering population diversity analysis for stagnation and pre-mature convergence. The MDEVM is implemented using a population-based parallel model and studies are conducted on 28 benchmark functions provided for the IEEE CEC-2013 competition. Experimental results demonstrate high performance in convergence speed of the proposed MDEVM algorithm.

1. Introduction

Accuracy enhancement as well as increase of convergence speed toward finding global solution(s) in optimization algorithms have motivated many researchers to develop more efficient evolutionary approaches. Such methods have solved many problems successfully such as smart grid design [1], vehicle navigation using ant colony optimization (ACO) [2], and communication system design using harmony search algorithm [3]. Differential evolution (DE) algorithm is one of the state-of-the-art global optimization algorithms, which is popular due to its simplicity and effectiveness. This algorithm works based on a set of individuals, called population, where an optimal size setting is imperative for a good performance [4].

Different variants of DE algorithm with large population size often grant more reasonable results than their small population size versions. A large population size supports a higher diversity for the population, which recombination of its diverse members offers a higher exploration ability to the optimizer to find global solution(s) [5–7]. The proposed diversity enhancement technique in this paper offers a better exploration of problem landscape. Most of the research works during past years were focused on developing complex approaches with a large populations size [49]. Utilizing a large population size intrinsically encompasses more function evaluations [5]. Therefore, using algorithms with a large population size may not be satisfactory for real-time or online applications [48,50].

Using a population size much smaller than the number of decision variables is sometimes more efficient than some of the state-of-the-art DE algorithms with a large population. The term micro-algorithm, denoted by μ-algorithm, refers to population-based algorithms with a small population size [7]. The micro-algorithms have been used in diverse applications, exceptionally due to their lighter hardware requirements and opportunity to operate in embedded systems with a memory saving approach [4]. Employing small population sizes decreases the number of function calls, but unfortunately due to lack of diversity, it also increases the risk of premature convergence as well as stagnation.

* Corresponding author.
E-mail address: hojjat.salehinejad@gmail.com (H. Salehinejad).

http://dx.doi.org/10.1016/j.asoc.2016.09.042
1568-4946/

The premature convergence problem refers to the situation where the population has converged to a sub-optimal solution of a multi-modal objective function [5]. This situation mostly occurs when the population has lost its diversity and cannot jump out of local optima. In this case, the algorithm progresses slower than usual and may stop any further improvement of the evolved candidate solutions [5,50].

The stagnation refers to either of the following scenarios occur: the population has not converged to any point and is still diverse; the algorithm cannot find a better solution as it proceeds, even if new individuals are entered to the population [5,50].

Based on the stagnation and premature convergence characteristics, it seems reducing the population size while raising the diversity of the population is a key point to achieve a faster convergence speed while maintaining a low risk of premature convergence or stagnation [5,50]. The DE algorithm is consisted of several manually tunable control parameters, where different adaptive proposals have been devised to avoid manual adjustments [80]. One way to increase the diversity of population while keeping its convergence toward global solution(s) is using parameter adaptive techniques.

One of the main criticisms for population-based algorithms is their computational cost. The core idea of this paper is to decrease the computational cost by reducing the population size and compensate the lack of diversity by randomizing the mutation scale factor. This idea is not only simple, but also introduces a typical interval for randomization of mutation scale factor (e.g. [0.1,1.5]) which relaxes the user from strict parameter setting of mutation scale factor to a constant value [37] (even the self-adaptive versions introduce a new hyper-parameter, the decay rate). The authors have recently proposed the idea of vectorized random mutation factor in DE algorithm for each decision variable of the problem, called MDEVM [50]. The fact that using randomized mutation scale factor acts randomly to select the mutation scale factor is right (from a hyperbox), which is the reason for added diversity while having many less individuals. But, it does not question the essential idea of DE, since still computes the differences between individuals in generating mutation vectors. The MDEVM has been applied on 3D sensor localization problem successfully [57].

The proposed transition from a scalar constant F to a scalar random F to a vectorized random $\mathbf{F}$ in DE has an interesting inverse in PSO [50]. In plain PSO [89], "velocities were adjusted according to their difference, per dimension, from best locations". This design guideline is further solidified in the 2007 standard for PSO [90]. As an extreme similar to scalar constant mutation factor in plain DE, there is also "Canonical Deterministic PSO" in which the update equation "system does not contain stochastic factors" [91]. It is noticeable that the role and effect of update weights in both DE and PSO is an interesting area for continued and coordinated study.

We have made the following contributions in this paper: (1) Conducted a comprehensive survey on micro-evolutionary algorithms; (2) Studied the proposed enhanced version of the MDE algorithm in [50], i.e. MDEVM, with population distributed implementation; (3) Mutation factor diversity analysis by Monte-Carlo simulations; (4) Presented population-based parallel model of MDEVM; (5) Conducted comparative study and analysis of MDE algorithms in terms of: variant mutation schemes and very-small population sizes (i.e. $N_P = \{2, 3, 4\}$), various problems types (uni-modal, multi-modal, composite), variant problem dimensions (i.e. $D = \{10, 30, 50, 100\}$) and mutation schemes, variant ranges for mutation factor, population diversity of the MDE algorithms in stagnation and trapping in local optimums, and variant stopping conditions for the MDE algorithm.

In the next section, the micro-population based methods are briefly surveyed. Then, a review of the DE algorithm is presented in Section 3. In Section 4, the proposed method is presented, and diversity enhancement in MDE using different structures of mutation scale factor is studied in detail. The simulation results and corresponding analysis are provided in Section 5. Finally, the paper is concluded in Section 6.

Table 1
Summary of related works in micro-population-based algorithms.

Population-based algorithm	Related research works
Genetic algorithm (GA)	[8,13,25,39–47]
Particle swarm optimization (PSO)	[9,10,51–68,77]
Differential evolution (DE)	[4,6,11,12,14–22,38,36,50,79]
Artificial bee colony (ABC)	[69,70]
Bacterial foraging optimization (BFO)	[71,72]
Artificial immune system (AIS)	[75]
Elitistic evolution (EEv)	[76]
Bat algorithm	[73]

2. Related works

Many research works have attempted to introduce efficient micro-algorithms. We can categorize such research works into four main groups which are micro-genetic algorithms (micro-GAs), micro-particle swarm optimization (micro-PSO), MDE, and other population-based approaches, Table 1. Another classification is according to different methods in helping realizing the effect of 'micro', such as population initialization and re-initialization, preserving individuals for the next generations, adaptive population size, adaptive local search, cooperative sub-population/sub-swarm, hybridization, and parameter adaptation techniques.

2.1. Population initialization and re-initialization

The idea of population reinitialization for micro-GA is one of the early works in the field [42]. In this approach, the best individual of each converged population, after a predefined number of generations, is replaced with a randomly selected individual in the population of the next iteration.

The micro-algorithms also have been employed in multi-objective optimization (MOO). The improved version of non-dominated sorting genetic algorithm (NSGA-II) with a specific population initialization strategy is embedded into the standard micro-GA to solve MOO problems [13]. A micro-GA with a population size of four and a reinitialization strategy is proposed in [39] which can produce a major part of the Pareto-front at a very low computational cost. Three forms of elitism and a memory are used to generate the initial population [39]. PSO is one of the well-known swarm intelligence algorithms, which its small population size versions have been developed recently [9,67,10]. A five-particle micro-PSO is used in [53] to deal with constrained optimization problems. This method preserves population diversity by using a reinitialization process and incorporates a mutation operator to improve the exploratory ability of the algorithm. The reported results present competitive performance versus the simple multi-member evolution strategy (SMES) and stochastic ranking (SR) method [53]. The micro-ODE is proposed and evaluated for an image thresholding case study [16]. Its performance is compared with the Kittler algorithm and MDE. The micro-ODE method has outperformed these algorithms on 16 challenging test images and has demonstrated faster convergence speed due to embedding the opposition-based population initialization scheme [16]. A 'micro' version of bacterial foraging optimization algorithms (BFOAs), called (μ-BFOA), is proposed in [71]. This method keeps the best bacterium unaltered, whereas the other population members are reinitialized [71]. This approach has outperformed the standard BFOA with a larger population size [71]. A micro-artificial immune system (Micro-AIS) with five individuals (antibodies), from which

only 15 clones are obtained is proposed in [75]. In this approach, the diversity is preserved by considering two simple but fast mutation operators in a nominal convergence manner, which work together in a reinitialization process [75]. Centroid-based DE is proposed in [60,61]. This approach uses centroid of population to compute the next generation of population. It has been developed for population initialization, which has superior performance versus uniform random population generation.

2.2. Preserving individuals

A general technique to increase performance of population-based algorithms, and particularly, their micro-version is preserving one or more individuals of the current generation for the next generation. One of the earlier research works in this direction is a GA with five chromosomes [8]. The strategy is to copy the best found chromosome from the current population to the next generation. It is tested on low-dimensional problems, which has resulted in a faster convergence speed compared to the classical GA. An improved version of micro-GA for constrained MOO is proposed in [44], called archive-based micro-GA (AMGA2). This algorithm is based on a steady-state GA that preserves an external archive of best and divert candidate solutions. This small population-based approach facilitates the decoupling of the working population, the external archive, and the number of required solutions as the outcome of the optimization procedure.

The micro-PSO is employed for MOO in [54]. It produces reasonably good approximations of the Pareto front of moderate dimensional problems with a small number of objective function evaluations (only 3000 calls per run), comparing to PSO approach. This method uses a reinitialization method and two external archives to preserve diversity [54]. One of the archives stores the found solution during the search. The other archive stores the final obtained solutions [54].

2.3. Adaptive population size

Some approaches toward reducing computational cost of DE-based algorithms by reducing the population size are proposed [11,12,14–16]. A gradually reducing population size method is proposed in [11]. This method is examined on 13 benchmark functions, where the results have demonstrated a higher robustness as well as efficiency compared to the parent DE [11]. The idea of self-adaptive population size is utilized to test absolute and relative encoding methods for DE [15]. The reported simulation results on 20 benchmark problems denote that in terms of the average performance and stability, the self-adaptive population size using relative encoding outperforms the absolute encoding method and the standard DE algorithm [15]. The smallest population size used in [38] is $N_P = 10$. This method tries to decrease the population size by using three different rules to select candidates for replacing the trial vector [38].

2.4. Adaptive local search

The micro-GA is used for local fine tuning in an adaptive local search intensity manner for training recurrent artificial neural networks (ANN) [46]. It is reported that this approach is useful for system identification tasks. A local search procedure is hybridized with the MDE algorithm in [6] to overcome high dimensional problems. However, the reported performance results are comparable with some other methods. In order to increase the exploration ability of MDE algorithm and to prevent stagnation, an extra search move is incorporated into the MDE algorithm in [4] by perturbing it along the axes.

2.5. Cooperative sub-population/sub-swarm

A cooperative PSO approach is proposed in [63] which uses a company of low-dimensional and low-cardinality sub-swarms to deal with complex high-dimensional problems. Promising results are reported using these methods. A parallel master–slave model of cooperative micro-PSO is introduced in [10]. The original search space is decomposed into subspaces with smaller dimensions. Then, five individuals are considered in each subspace to identify suboptimal partial solution components. Its performance is assessed on a set of five widely used test problems with significant improvements in solution quality, compared to the standard PSO algorithm [10]. In a micro-DE approach [12], small size cooperative sub-populations are employed to find sub-components of the original problem concurrently. During cooperation of sub-populations, the sub-components are combined to construct the complete solution of the problem. Performance evaluation of this method has been done on high-dimensional instances of five sample test problems with encouraging results reported in [12]. An efficient scheduler for heterogeneous computing (HC) and grid environments, based on parallel micro-cross generational elitist selection, heterogeneous recombination, and cataclysmic mutation, called pμ-CHC is proposed in [78]. This method combines a parallel sub-population model with a focused evolutionary search using a micro population and a randomized local search method. Performance comparisons of algorithms such as ant colony algorithm (ACO) and GA have demonstrated good scheduling in reduced execution times [78]. The parallel version of micro-GA, called parallel micro-genetic algorithm (PMGA), is reported in [43]. This method solves the ramp rate constrained economic dispatch (ED) problems for generating units with non-monotonically and monotonically increasing cost functions. The PMGA is implemented on a Beowulf cluster with 32 processors. The reported results demonstrate feasibility of this approach for online applications.

2.6. Hybridization

A clonal selection algorithm (CSA), which belongs to the family of AIS, in conjunction with a micro-PSO (CS^2P^2SO) is introduced in [62] as a hybrid scheme. In this hybridization, the strength of standard PSO algorithm is enhanced, where the micro-PSO helps to find the optimum solution with less memory requirement and the CSA increases the exploration capability while reducing the chance of convergence to a local minima. Simulations are conducted on only four benchmark functions, where competitive performance is reported. The micro-PSO has been developed for many applications such as motion estimation [51], power system stabilizers design [56,59], optimal design of static var compensator (SVC) damping controllers [58], reactive power optimization [64], short-term hydrothermal scheduling [66], reconfiguration of shipboard power system [68], and transient stability constrained optimal power flow [65]. A mixed-integer-binary small-population PSO is proposed in [77] for solving a problem of optimal power flow. The constraint handling technique used in this algorithm is based on a strategy to generate and keep its four decision variables in feasible space through heuristic operators. In this way, the algorithm focuses its search procedure on the feasible solution space to obtain a better objective value. This technique improves the final solution quality as well as the convergence speed [77]. A cooperative micro-artificial bee colony (CMABC) approach for large-scale optimization is presented in [69]. This approach has combined the divide-and-conquer property of cooperative algorithms and low computational cost of micro-artificial bee colony (MABC) method. As an application of MDE, a hybrid differential evolution (HDE) with population size of five is used for finding a global solution [79]. MDE is also employed

Table 2
Parameter setting for all conducted experiments.

Parameter	Description	Value
Cr	Crossover probability constant	0.9
NFC_{Max}	Maximum number of function calls	$1000 \times D$
$EVTR$	Objective function error value to reach	1e−8
N_{Run}	Number of runs	30
F	Mutation factor	0.9

for evolving an indirect representation of the bin packing problem with acceptable performance [14].

A model of MOO for hierarchical GA (MOHGA) based on the micro-GA approach for modular neural networks (MNNs) optimization is proposed in [45]. This approach is used for iris recognition application. The MOHGA divides the input data into granules and sub-modules and then decides to split the data for training and testing phases. It is reported that this technique can obtain good results based on using less data [45]. In [25] a multi-objective micro genetic extreme learning machine is proposed, which provides the appropriate number of hidden nodes in the machine for solving the problem, which minimizes the mean square error (MSE) of the training phase. The micro-GA is applied successfully for many applications such as designing waveguide slot antenna with dielectric lenses [47], detection of flaws in composites [41], and scheduling of a real-world pipeline network [40], where better performances compared to the standard GA are reported.

2.7. Parameter adaptation

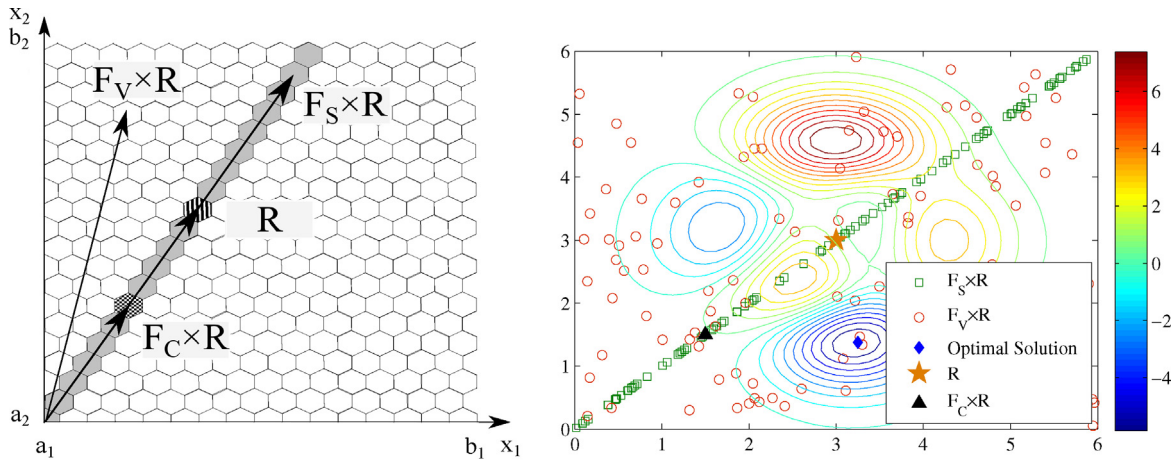
Adaptation of micro-population-based algorithms is one of the promising approaches to increase performance of such algorithms. Many methods have been proposed in the literature to increase robustness and reliability of DE algorithm through adaptive or self-adaptive approaches [26,27,32]. This is particularly important for the hyper-parameters adjustment. The mutation factor is one of those parameters which generally is set to a constant value [37]. However, it has been shown that randomization of mutation factor can offer a potential new search moves and compensate the excessively deterministic search structure of a standard DE algorithm [28,50]. Studies have used various distribution such as Gaussian, Log-normal, and Cauchy to generate random mutation factor. However, none of them is superior over the others [37].

The methods proposed in [28,29] use random mutation factor at each generation to increase diversity of the population, which reportedly is effective for both noise and stationary problems [30]. These methods use a standard population size whereas the mutation factor F is randomly selected from the range (0.5, 1) such that its mean value is controlled to remain at 0.75. In [31], four different mutation scale factor schemes are proposed. The population size is set to $N_P = 200$. The study shows that none of the methods can show promising results for all problems, since the performance is dependent on the employed type of the distribution [31]. In [34], a self-adapted DE algorithm for the mutation factor and crossover rate parameters is presented. The smallest population size used in the experiments is $N_P = 25$. A self-adaptive control mechanism is used in [35] to change the mutation factor F and crossover rate C_r during the generations. In this method, only the "rand/1/bin" mutation vector is used for a multi-population method with aging mechanism. The jDE method is one of the promising methods, as F is generated with a specific ratio for each individual of a standard population size [33]. The idea of generating random mutation factor at the lowest level (for each individual of population and dimension of problem per generation) is proposed in [50]. This technique is used to increase search performance of the standard MDE algorithms. This method is evaluated on a set of 28 benchmark functions for CEC-2013 competition, where the results show superior exploration performance. This algorithm, called MDEVM, is used as a measure to compare the performance of a new mutation factor, called current-by-rand-to-pbest, proposed in the μJADE algorithm [36]. This approach is a DE algorithm for unconstrained optimization problems, the smallest considered population size is $N_P = 8$ [36]. The mutation factor F and crossover rate C_r are randomly generated at the beginning of each generation, where the mean of distribution is updated in each generation. The proposed mutation factor in [36], called current-by-rand-to-pbest, is tested for both large and small population sizes on a set of 13 classical benchmark functions. The comparative results in [50] show competitive performance between MDEVM and μJADE algorithms. The cDE methods uses a statistical representation of population, with similar memory requirement to the populations with four individuals, regardless of the problem's dimension to solve problems [32,36]. Since in this work we are focused on discussing small non-virtual populations, this class of DE algorithms is left for further investigation in other works.

For the environmental economic dispatch case study, a chaotic micro bacterial foraging algorithm (CMBFA) with a time-varying chemotactic step size is proposed in [74]. It is reported that the convergence characteristic, speed, and solution quality of this method are better than the classical BFOA for a 3-unit system and the standard IEEE 30-bus test system. An other type of EAs, called elitistic evolution (EEv), is proposed for optimizing high-dimensional problems in [76], which works without using complex mechanisms such as Hessian or covariance matrix. This approach utilizes adaptive and elitism behaviour, in which a single adaptive parameter controls the evolutionary operators to provide reasonable local and global search abilities [76]. The Coulomb's law is used in micro-PSO method for high dimensional problems [9]. First achievement of this approach is removal of the burden for determining the suitable size of space needed to enclose the blacklisted solutions and the amount of repulsion needed to repel the particles, as these parameters are extremely difficult to determine for high dimensional problems. The other achievement is the flexibility of controlling the repulsion on particles through the use of a parameter which controls the amount of repulsion experienced by the particles at a particular position. The simulation results on five high-dimensional benchmark functions demonstrate superior performance of micro-PSO versus the standard PSO with a large populations size.

3. Differential evolution

The DE algorithm works based on the scaled difference between two individuals of a population set, where the scaling factor is called the mutation factor. Due to reliability and simplicity of the DE algorithm, it has been employed in many science and engineering areas such as solving large capacitor placement problem [21] and synthesis of spaced antenna arrays [22]. Many works have put new schemes forward to enhance the DE algorithm such as opposition-based differential evolution (ODE) [18], Type-II ODE [55], enhanced differential evolution using center-based sampling [19], and opposition-based adaptive differential evolution [20]. Generally speaking, while solving a black-box problem to find optimal decision variables, an optimizer has no knowledge about the structure of the problem landscape to minimize/maximize an objective function. The DE algorithm, similar to other algorithms in its category, starts its search procedure with some uniform random initial vectors and tries to improve them in each generation toward an optimal solution. The population $\mathbf{P} = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_P}\}$ consists of N_P vectors in generation g, where $\mathbf{X}_i$ is a D-dimensional vector defined as $\mathbf{X}_i = (x_{i,a}, \ldots, x_{i,D})$. Generally a simple DE



(a) Mutation vector distribution of vector R **(b)** Monte-Carlo simulation

Fig. 1. Diversity of vector for a 2-D individual vector R on a 2-D map for constant (F_C), scalar random (F_S), and vectorized random (F_V) mutation scale factors.

algorithm consists of the following three major operations: mutation, crossover, and selection.

Mutation: This step selects three vectors randomly from the population such that $i_1 \neq i_2 \neq i_3 \neq i$ where $i \in \{1, \ldots, N_P\}$ and $N_P \geq 4$, for each vector $\mathbf{X}_i$, the mutant vector scheme "DE/rand/1" is calculated as

$$\mathbf{V}_i = \mathbf{X}_{i_1} + F(\mathbf{X}_{i_2} - \mathbf{X}_{i_3}), \tag{1}$$

where the factor $F \in (0, 2]$ is a real constant number, which controls the amplification of the added differential vector of $(\mathbf{X}_{i_2} - \mathbf{X}_{i_3})$. The exploration ability of DE increases by selecting higher values for F. So far, four main mutation schemes are introduced [80,81], summarized as

- DE/best/1:

$$\mathbf{V}_i = \mathbf{X}_{best} + F(\mathbf{X}_{i_1} - \mathbf{X}_{i_2}) \tag{2}$$

- DE/t2b/1:

$$\mathbf{V}_i = \mathbf{X}_i + F(\mathbf{X}_{best} - \mathbf{X}_i) + F(\mathbf{X}_{i_1} - \mathbf{X}_{i_2}) \tag{3}$$

- DE/rand/2:

$$\mathbf{V}_i = \mathbf{X}_{i_1} + F(\mathbf{X}_{i_2} - \mathbf{X}_{i_3}) + F(\mathbf{X}_{i_4} - \mathbf{X}_{i_5}) \tag{4}$$

- DE/best/2:

$$\mathbf{V}_i = \mathbf{X}_{best} + F(\mathbf{X}_{i_1} - \mathbf{X}_{i_2}) + F(\mathbf{X}_{i_3} - \mathbf{X}_{i_4}), \tag{5}$$

where $\mathbf{X}_{best}$ is the corresponding vector of the best objective value in the population.

Crossover: The crossover operation increases diversity of the population by shuffling the mutant and parent vector as follows:

$$U_{i,d} = \begin{cases} V_{i,d}, & \text{if } rand_d(0,1) \leq C_r \text{ or } d_{rand} = d \\ x_{i,d}, & \text{otherwise} \end{cases}, \tag{6}$$

where $d = 1, \ldots, D$, is the dimension and $C_r \in [0, 1]$ is the crossover rate parameter, and $rand(a, b)$ generates a real random uniform number in the interval $[a, b]$. Therefore, the trial vector $\mathbf{U}_i \; \forall i \in \{1, .., N_P\}$ can be generated as

$$\mathbf{U}_i = (U_{i,1}, \ldots, U_{i,D}). \tag{7}$$

Selection: The $\mathbf{U}_i$ and $\mathbf{X}_i$ vectors are evaluated and compared with respect to their fitness values; the one with better fitness value is selected for the next generation.

4. Proposed diversity enhancement via vectorized random mutation

In this section, first we discuss limitations of constant mutation scale factor and provide motivation for a vectorized random mutation scale factor. Then, we formally propose our vectorized random mutation scale factor algorithm followed with a population-based parallel model for parallel implementation. At the end, we discuss behaviour of MDE in different problem dimensionalities to support our proposed algorithm.

4.1. Exploration ability of mutation scale factor

The mutation scale factor has serious affect on the exploration ability of the DE. We consider three type of mutation scale factors: constant mutation scale factor (CMF), scalar random mutation scale factor (SRMF), and vectorized random mutation scale factor(VRMF). In order to visualize these factors, a variable space is presented in Fig. 1a. For better presentation, it is made of small hexagons, where each hexagon represents a point on the variable space. The landscape for variables x_1 and x_2 is limited to boundaries $[a_1, b_1]$ and $[a_2, b_2]$. A sample vector $\mathbf{R}$ is denoted with a dashed hexagon. The vector $F_C \times \mathbf{R}$ denotes multiplication of an arbitrary CMF to the vector $\mathbf{R}$, shown with a dotted dark hexagon. Therefore, diversity of the generated mutation vector $F_C \times \mathbf{R}$ is limited to one hexagon (i.e. the dotted dark hexagon) on the direction of vector $\mathbf{R}$. In the case of having an identical uniform random F for all variables of an individual (i.e., the SRMF scheme), the diversity of mutation vector $F_S \times \mathbf{R}$ is not just limited to one hexagon (i.e. the dotted dark hexagon), yet is along the vector $\mathbf{R}$, denoted by grey hexagons. Conversely, by randomizing F for each variable of each individual using a uniform random vector $\mathbf{F}$, i.e. $F_V \times \mathbf{R}$, the VRMF diversity covers the whole plane containing all the hexagons, which presents the highest exploration power.

The diversities of CMF, SRMF, and VRMF are investigated by employing Monte-Carlo simulation on an arbitrary landscape in Fig. 1b. In this simulation for arbitrary vector $\mathbf{R} = [1, 1]$, 100 sample mutation vectors for each CMF, SRMF, and VRMF schemes with $F_C = 0.5$ and F_S, $F_V \in [0, 2]$ are generated, where the variables are limited as $x_1, x_2 \in [0, 3]$. The simulation illustrates that the VRMF scheme supports a higher diversity than the SRMF, where its diversity is limited to the points on a line. Strictly speaking, if all variables in the individual vector $\mathbf{R}$ are multiplied by a random scalar number, other points are generated on the same direction of the line which is indicated by vector $F_S \times \mathbf{R}$. In fact, the SRMF generates

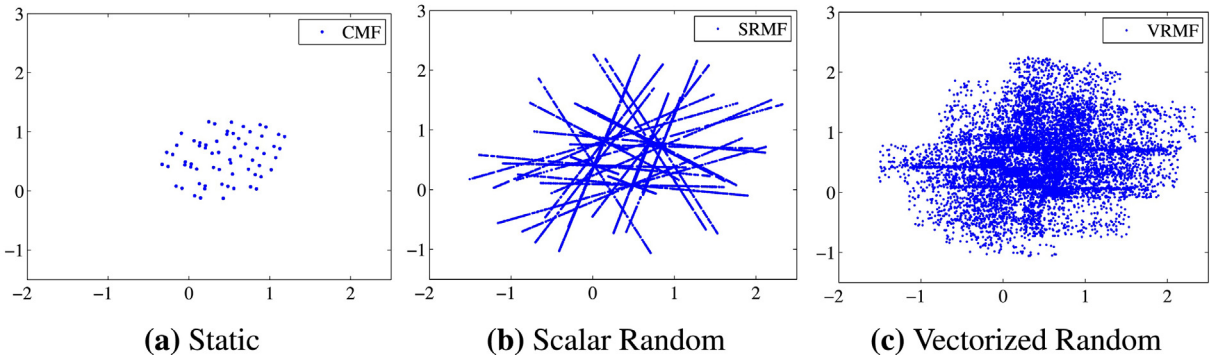


Fig. 2. Monte-Carlo simulation of population diversity for $D = 2$ and $N_P = 5$ after 10,000 random generation by considering the crossover operator.

points on the same direction as vector **R**. If the relationship among the variables (variables' interaction) are linear, the mutation vector is doing fine (which is a very exceptional case, especially during solving real-world problems). However, when the VRMF scheme is utilized, the mutation vector has no restriction to explore any point on the search space with no linearity restriction, which was the case for SRMF. This discussion is valid for higher dimensions, where the line needs to be replaced with a plane or hyperplane.

By taking into account the crossover component of MDE algorithm, another Monte-Carlo simulation is conducted for CMF, SRMF, and VRMF schemes as presented in Fig. 2. This simulations are conducted using the "DE/rand/1" mutation scheme for a population size of $N_P = 5$, and 10,000 sample individuals are generated from an identical uniform random population, in a 2D variables space, where each variable is uniform randomly selected as $x_i \in [0, 1]$. The crossover plays a decisive role in taking diversity into the populations, as presented for the CMF scheme in Fig. 2a. However as presented in Fig. 2b and c, the crossover also expands the diversity of SRMF and VRMF schemes dramatically such that almost the whole variable space is explored by the VRMF scheme.

4.2. Proposed micro-differential evolution with vectorized random mutation factor

In this subsection we formally introduce our proposed algorithm. The population size of MDE is very small compared to the standard DE. Reducing the population size means faster convergence rate with a higher risk of stagnation. This is mostly due to the lack of diversity in the population. Therefore, we need to increase the diversity in population through other techniques.

In order to foster diversity, the mutation factor F, as one of the most significant control parameters for the DE algorithm, can play a major role. The mutation factor F in the DE algorithm is a constant mutation factor (CMF) generally set to $F = 0.5$ [5,18]. This factor can also be selected randomly for each individual [6]. Different versions of this scalar random mutation factor (SRMF) is proposed in literature for standard DE algorithm. We call its micro version as the micro-differential evolution with scalar random mutation factor (MDESM) where the population size is very small as well. In order to increase the population diversity of MDE algorithm, we propose the idea of vectorized random mutation vector (VRMF) for each dimension of each individual, where its application for MDE is called the MDEVM. Therefore, the mutation factor can be defined for each individual i as

$$\mathbf{F}_i = \{F_{i,1}, \ldots, F_{i,D}\}, \quad \forall i \in \{1, \ldots, N_P\}, \tag{8}$$

where $F_{i,j} = rand[0.1,\ 1.5],\ \forall j \in \{1,\ \ldots,\ D\}$, [6]. This interval is selected based on the experimental results presented in the next section.

Algorithm 1. Micro-Differential Evolution with Vectorized Mutation (MDEVM)

Procedure MDEVM
$g = 0$
//Initial Population Generation
for $i = 1 \rightarrow N_P$ **do**
  for $d = 1 \rightarrow D$ **do**
    $\mathbf{X}_{i,d} = x_d^{\min} + rand(0, 1) \times (x_d^{\max} - x_d^{\min})$
  end for
  $\mathbf{P}_i^g = \mathbf{X}_i$
end for
//End of Initial Population Generation
while $(|BFV - VTR| > EVTR\ \&\ NFC < NFC_{Max})$ **do**
  for $i = 1 \rightarrow N_P$ **do**
//Mutation
    Select three random population vectors from $\mathbf{P}^g$ *where* $(i_1 \neq i_2 \neq i_3 \neq i)$
    for $d = 1 \rightarrow D$ **do**
      $F = rand(0.1, 1.5)$
      $\mathbf{V}_{i,d} = \mathbf{X}_{i_1,d} + F(\mathbf{X}_{i_2,d} - \mathbf{X}_{i_3,d})$
    end for
  //End of Mutation
  //Crossover
    for $d = 1 \rightarrow D$ **do**
      if $rand(0, 1) < C_r$ or $d_{rand} = d$ **then**
        $U_{i,d} = V_{i,d}$
      else
        $U_{i,d} = x_{i,d}$
      end if
    end for
  //End of Crossover
  //Selection
    if $f(\mathbf{U}_i) \leq f(\mathbf{X}_i)$ **then**
      $\mathbf{X'}_i = \mathbf{U}_i$
    else
      $\mathbf{X'}_i = \mathbf{X}_i$
    end if
  //End of Selection
  end for
  $\mathbf{X}_i = \mathbf{X'}_i, \forall i \in \{1, \ldots, N_P\}$
  $g = g + 1$
  $\mathbf{P}^g = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_P}\}$
end while

The pseudocode of the proposed MDEVM approach is in Algorithm 1. After generation of initial population, the mutation vector is computed by using the proposed mutation factor, Eq. (8). Then, the *crossover* and *mutation* procedures are conducted similar to the DE algorithm to generate the next population. The termination criterion is met when the difference between best fitness value (*BFV*) and fitness value-to-reach (*VTR*) is less than fitness error-value-to-reach (*EVTR*), or the searching procedure exceeds the maximum number of function calls NFC_{Max}, i.e., $NFC \geq NFC_{Max}$.

4.3. Population-based parallel model

Generally speaking, the evolutionary algorithms are categorized into population distributed and dimension distributed models. The

Table 3
Ratio of number of generated vectors by the plain DE over proposed DE after binary crossover stage for population sizes $N_P = \{3, 4, 5, 6, 10, 30\}$ and dimensions $D = \{10, 100, 1000\}$. $M \to \infty$ represents possible number of unique generated mutation vectors using vectorized random mutation scale factor.

N_P	D		
	10	100	1000
3	$\frac{6\times 2^{10}}{6\times M\times 2^{10}}$	$\frac{6\times 2^{100}}{6\times M\times 2^{100}}$	$\frac{6\times 2^{1000}}{6\times M\times 2^{1000}}$
4	$\frac{24\times 2^{10}}{24\times M\times 2^{10}}$	$\frac{24\times 2^{100}}{24\times M\times 2^{100}}$	$\frac{24\times 2^{1000}}{24\times M\times 2^{1000}}$
5	$\frac{60\times 2^{10}}{60\times M\times 2^{10}}$	$\frac{60\times 2^{100}}{60\times M\times 2^{100}}$	$\frac{60\times 2^{1000}}{60\times M\times 2^{1000}}$
6	$\frac{120\times 2^{10}}{120\times M\times 2^{10}}$	$\frac{120\times 2^{100}}{120\times M\times 2^{100}}$	$\frac{120\times 2^{1000}}{120\times M\times 2^{1000}}$
10	$\frac{720\times 2^{10}}{720\times M\times 2^{10}}$	$\frac{720\times 2^{100}}{720\times M\times 2^{100}}$	$\frac{720\times 2^{1000}}{720\times M\times 2^{1000}}$
30	$\frac{24{,}360\times 2^{10}}{24{,}360\times M\times 2^{10}}$	$\frac{24{,}360\times 2^{100}}{24{,}360\times M\times 2^{100}}$	$\frac{24{,}360\times 2^{1000}}{24{,}360\times M\times 2^{1000}}$

population-based parallel models are parallelizable at the evolution task in population, individual, or operation levels [17]. Such models are generally divided into master-slave, island, cellular, hierarchical, and pool architectures. The dimension distributed models are focused on parallelization of the dimensions. Some of the famous models are co-evolution and multi-agent models. More details are provided in [17].

In this paper, we have implemented the MDEVM algorithm based on the master–slave population-based parallel model, Fig. 3. This model is simple and individuals as well as mutation vectors are mutually independent, therefore no message exchanging is necessary between the slaves. In this model, the master performs several tasks including initial population generation, mutation factor scale generation, mutation vector generation, and parent selection, to name some. It controls the crossover, mutation, and selection operation. The master sends the population objects to the pool of slaves, which is consisted of N_P slaves. The slaves receive each allocated individual from the master to a slaves to perform and return fitness evaluation. The fitness evaluation is allocated to the slaves because this operation is the most expensive computation in a typical DE algorithm.

4.4. Micro-population behaviour in different problem dimensionalities

By considering N_P as the population size and the mutation scheme in Eq. (1) with three parents, the DE algorithm can generate $(N_P)_3 = N_P!/(N_P-3)!$ mutant vectors. In case of having a binary crossover, each parent can generate 2^D possible vectors. Therefore, we totally have $2^D \times N_P!/(N_P-3)!$ possible vectors to help the algorithm to explore the problems landscape.

Lets assume a finite discrete sample space $\mathrm{IF}^D \subset \mathrm{IR}^D_{\geq 0} = \{f \in \mathrm{IR}^D : f \geq 0\}$, with cardinality M such that $inf\{f \in \{\mathrm{IF}^D\} : f > a\} = a$ and $sup\{f \in \{\mathrm{IF}^D\} : f < b\} = b$. The mutation scale factor set $\mathbf{F} : \{f_1, f_1, .., f_D\}$ for CMF, SRMF, and VRMF schemes is defined as $f_d = cte$, $f_d = rand([a, b])$, and $f_d = rand_d([a, b])$, respectively, where $rand([a, b])$ is a randomly selected element (with uniform distribution) from IF^D. In this case, $\mathrm{IF}^D = \mathrm{IR}^D_{\geq 0}$ if $M \to \infty$.

The fraction of number of unique generated points of the plain DE over the proposed DE after the crossover stage for $N_P = \{2, 3, 4, 5, 6, 10, 30\}$ and dimensions $D = \{10, 100, 1000\}$ is reported in Table 3. We see that since the CMF is tailored to a constant value, it generates many less unique mutation vectors compared to the VRMF. For example, for $D = 10$ and $N_P = 6$, $\frac{6!\times 2^D}{3!\times M\times 2^D} \to 0$ as $M \to \infty$. Similar behaviour is observable for other dimensionalities and population sizes. From a population size view point, as it increases, more unique mutation vectors are generated; however, this value is far less than the number of unique mutation vectors generated by MDEVM. The total number of generated vectors are many more for high-dimensional data, comparing to low-dimensional ones. For $D = 10$, the number of possible vectors for $N_p = 5$ is 0.24% of the possible vectors for $N_p = 30$, which means the power of exploration is 0.24%.

The relationship among population size, dimensionality of problem, and number of generated vectors is visualized for $5 \leq D \leq 10$ and $5 \leq D \leq 1000$ separately in Fig. 4. As it is demonstrated in Fig. 4a and b, when the population size is small (i.e. $N_P \leq 10$), the number of generated vectors is far fewer than the standard population size (i.e. $N_P \approx 30$). The problems dimensionality also has the same effect on the algorithm performance, Fig. 4c. The very important point is that for the small population sizes, dimensionality of the problem does not affect performance of the algorithm. Fig. 4a clearly shows that for the small population sizes and different problem dimensionalities much smaller number of vectors are generated, comparing to the large population sizes. The micro-population region of interest (ROI) means when micro-population algorithms are in use, exploration ability of algorithm is limited to small number of generated vectors after crossover, regardless of problems' dimensionality. Therefore, the micro-population algorithms have almost similar performance for high-dimensional problems as for low-dimensional problems. A wider view for higher dimensionalities is presented in Fig. 4d–f.

5. Simulation results

In this section, performance of the proposed MDEVM algorithm is compared with the MDE, MDESM, μJADE [36], self-adaptive DE (SaDE) [87], and composite DE (CoDE) [88] algorithms. The parameter setting and employed benchmark functions (i.e. CEC-2013 testbed [23]) are described in the next subsection. Then, the comprehensive experimental series are presented in details. The algorithm is implemented in parallel using the multiprocessing library of Python programming language, conducted on sharcnet.ca clusters [86].

5.1. Benchmark functions and parameters setting

All the experiments are conducted on the CEC-2013 problems [23]. It is comprised of 28 benchmark functions and an improved version of CEC-2005 [24] counterpart with additional test functions and modified formula in order to create the composite functions, oscillations, and symmetric-breaking transforms. This testbed is divided into three categories which are uni-modal functions (f_1–f_5), multi-modal functions (f_6–f_{20}), and composite functions (f_{21}–f_{28}) [23]. Parameters setting for all the experiments is presented in Table 2 adapted from the literature [6,18,23], unless a change is mentioned. The reported values are averaged for $N_{Run} = 30$ independent runs per function per algorithm to minimize the effect of the stochastic nature of the algorithms on the reported results.

The mutation schemes presented by Eqs.(1)–(5) are the five main schemes, which are used for $N_P \geq 5$ in experiments [80,81]. For the small size and very small size populations, we use the mutation schemes based on their structure for different sizes of population as demonstrated in Table 4.

5.2. Experimental series 1: mutation schemes and population size analysis

Performance of the MDE, MDESM, and MDEVM schemes are evaluated for mutation schemes in Table 4, population sizes $N_P \in \{2, 3, 4, 5, 6, 50\}$, and dimension $D = 50$. The Wilcoxon test results are reported in terms of pair-wise comparison in Table 5. The symbols "+", "=", and "−" indicate a statistically better, equivalent, and worse performance, respectively, compared with the MDEVM algorithm [82].

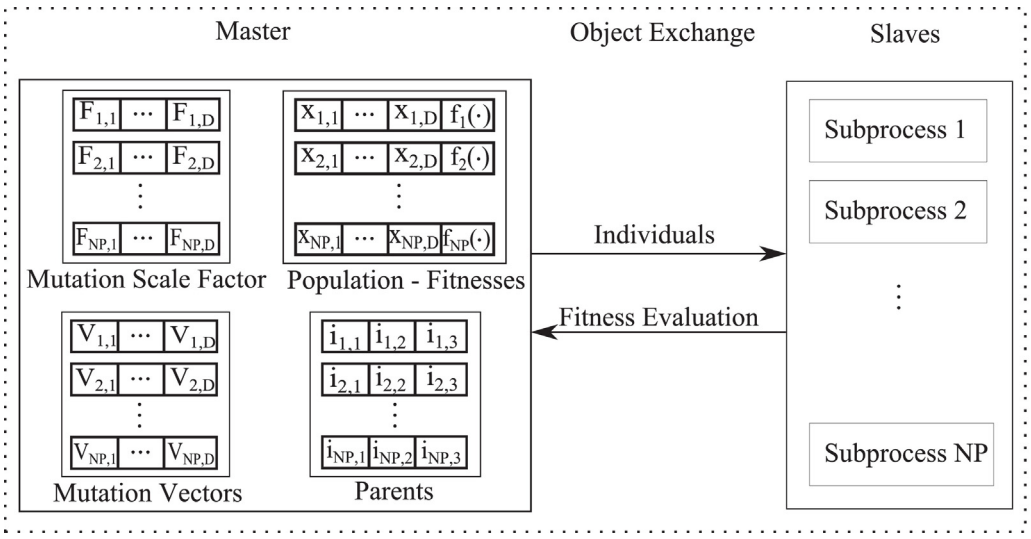
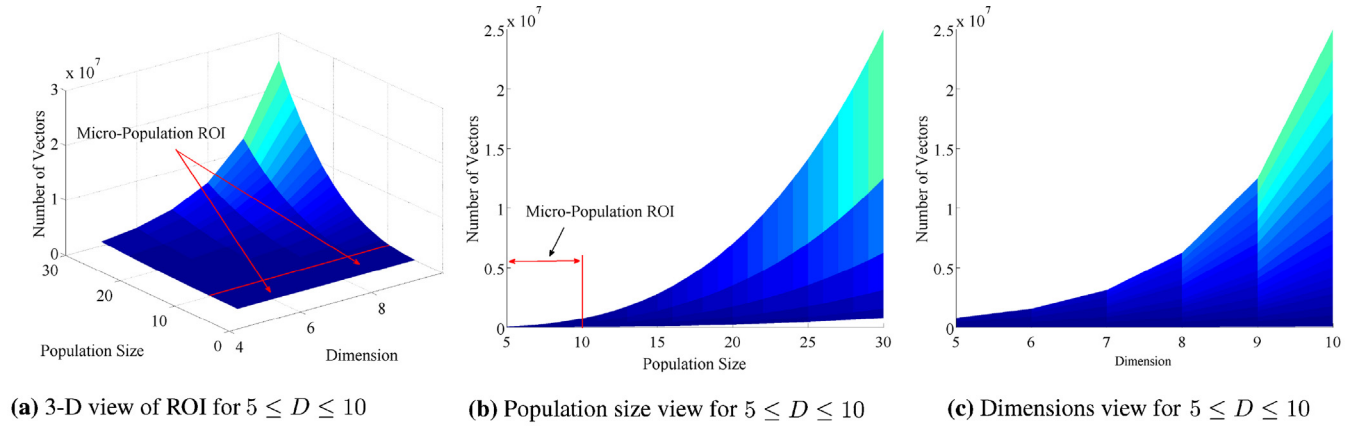


Fig. 3. Master, slave, and exchanging objects between processes for the population-based parallel model.



(a) 3-D view of ROI for $5 \leq D \leq 10$ **(b)** Population size view for $5 \leq D \leq 10$ **(c)** Dimensions view for $5 \leq D \leq 10$

(d) 3-D view of ROI for $5 \leq D \leq 1000$ **(e)** Population size view for $5 \leq D \leq 1000$ **(f)** Dimensions logarithmic view for $5 \leq D \leq 1000$

Fig. 4. Number of generated vectors after mutation and crossover stages for different population sizes N_P and problem dimensions D. The micro-population has far fewer number of possible vectors compared to the larger population size, which encourages lack of proper exploration. This causes stagnation and pre-mature convergence. The micro-population region of interest (ROI) in figures (a) and (d) indicate the exploration power of micro-population algorithms is very limited.

The results in Table 5 demonstrate that for small population size (i.e. $N_P \leq 5$), the MDEVM method has competitive or better performance than other methods. This is particularly obvious for the "DE/best/1" and "DE/t2b/1" schemes with $N_P = 2$ and "DE/best/1" scheme for $N_P = 3$. Regarding $N_P \in 4, 5$, we see that the MDEVM method has more successful results for the "DE/rand/1", "DE/best/1", and "DE/t2b/1" schemes. However, as the diversity of population is increasing by adding more number of individuals to the population, the MDEVM method achieves less successful results. This situation is obvious for $N_P = 50$ where a standard size

Table 4
Mutant vector (MV) schemes for population sizes $N_P \in \{2, 3, 4\}$ and $N_P \geq 5$.

N_P	MV	$\mathbf{V}_i$
2	DE/rand/1	$\mathbf{X}_1 + F(\mathbf{X}_1 - \mathbf{X}_2)$
	DE/best/1	$\mathbf{X}_{best} + F(\mathbf{X}_1 - \mathbf{X}_2)$
	DE/t2b/1	$\mathbf{X}_i + F(\mathbf{X}_{best} - \mathbf{X}_i) + F(\mathbf{X}_1 - \mathbf{X}_2)$
3	DE/rand/1	$\mathbf{X}_1 + F(\mathbf{X}_2 - \mathbf{X}_3)$
	DE/best/1	$\mathbf{X}_{best} + F(\mathbf{X}_1 - \mathbf{X}_2)$
	DE/t2b/1	$\mathbf{X}_i + F(\mathbf{X}_{best} - \mathbf{X}i) + F(\mathbf{X}_1 - \mathbf{X}_2)$
4	DE/rand/1	$\mathbf{X}_1 + F(\mathbf{X}_2 - \mathbf{X}_3)$
	DE/best/1	$\mathbf{X}_{best} + F(\mathbf{X}_1 - \mathbf{X}_2)$
	DE/t2b/1	$\mathbf{X}_i + F(\mathbf{X}_{best} - \mathbf{X}i) + F(\mathbf{X}_1 - \mathbf{X}_2)$
	DE/best/2	$\mathbf{X}_{best} + F(\mathbf{X}_1 - \mathbf{X}_2) + F(\mathbf{X}_3 - \mathbf{X}_4)$
5≤	DE/rand/1	$\mathbf{X}_1 + F(\mathbf{X}_2 - \mathbf{X}_3)$
	DE/best/1	$\mathbf{X}_{best} + F(\mathbf{X}_1 - \mathbf{X}_2)$
	DE/t2b/1	$\mathbf{X}_i + F(\mathbf{X}_{best} - \mathbf{X}i) + F(\mathbf{X}_1 - \mathbf{X}_2)$
	DE/best/2	$\mathbf{X}_{best} + F(\mathbf{X}_1 - \mathbf{X}_2) + F(\mathbf{X}_3 - \mathbf{X}_4)$
	DE/rand/2	$\mathbf{X}_1 + F(\mathbf{X}_2 - \mathbf{X}_3) + F(\mathbf{X}_4 - \mathbf{X}_5)$

of population is used and we see better performance of the MDE and MDESM methods. The results clearly show that for the $N_P \geq 5$, the VRMF technique in MDEVM can add a good diversity to the population which results in better performance than other methods. However, this diversity enhancement method has extra affect on large population sizes, in a way that the population has more than enough diversity and cannot converge to an optima, which is the stagnation situation. The "DE/best/2" and "DE/rand/2" schemes have more exploration capability due to incorporating more population individuals. Therefore, the VRMF technique adds extra diversity into the population. This is another additive diversity which stops the MDEVM method to converge to optimal solution(s). The difference between DE and MDE algorithms is in population size which delivers diversity into the population. Combining the VRMF technique with the DE algorithm consequences in extra diversity which results in a poor performance of the algorithm. The standalone DE-algorithm may result in a better performance, but by the cost of more number of function calls. Therefore, utilizing the MDE algorithm with small population sizes can deliver higher diversity and performance into the algorithm. In overall, the "DE/best/1", "DE/rand/1", and "DE/t2b/1" schemes have the best performance among the various mutation schemes for MDEVM.

Table 5
Number of Wilcoxon rank-sum test comparisons for MDEVM against MDM and MDESM with population sizes $N_P \in \{2, 3, 4, 5, 6, 50\}$ for dimension $D = 50$. If the bold value is under "+" column, the MDEVM method has the highest overall performance, otherwise, the corresponding method under the column header has the best overall performance.

N_P	MV	MDE			MDESM		
		+	=	–	+	=	–
2	DE/rand/1	0	23	**5**	2	19	**7**
	DE/best/1	**14**	11	3	**15**	9	4
	DE/t2b/1	**25**	3	0	**17**	9	2
3	DE/rand/1	**11**	10	7	7	11	**10**
	DE/best/1	**24**	4	0	**20**	5	3
	DE/t2b/1	**17**	9	2	**12**	10	6
4	DE/rand/1	**20**	4	4	**12**	5	11
	DE/best/1	**21**	7	0	**19**	5	4
	DE/t2b/1	**20**	4	4	**17**	1	10
	DE/best/2	2	3	**23**	0	4	**24**
5	DE/rand/1	**19**	2	7	**12**	8	8
	DE/best/1	**24**	2	2	**16**	7	5
	DE/t2b/1	**19**	2	7	**15**	4	9
	DE/best/2	**12**	6	10	6	7	**15**
	DE/rand/2	0	1	**27**	1	1	**26**
6	DE/rand/1	**13**	5	10	13	7	**8**
	DE/best/1	**21**	3	4	**18**	6	4
	DE/t2b/1	**19**	5	4	**15**	2	11
	DE/best/2	11	5	**12**	7	4	**17**
	DE/rand/2	1	1	**26**	1	2	**25**
50	DE/rand/1	1	2	**25**	1	3	**24**
	DE/best/1	10	5	**13**	2	6	**20**
	DE/t2b/1	0	3	**25**	0	4	**24**
	DE/best/2	0	4	**24**	0	3	**25**
	DE/rand/2	0	2	**26**	1	2	**25**

In Fig. 5, a summary of better performance counting of all schemes is presented, where $N_P = 5$ has the highest number of success for all mutation schemes on average. In order to have a closer look, average of better, equal, and worse performance counting for the MDEVM vs. MDE and MDEVM vs. MDESM comparisons are presented in Fig. 6.

Regarding the average of better and equivalent performances results as shown in Fig. 6a and b, it is clear that the "DE/best/1" scheme has the most number of successes. In terms of worse performance comparison, it is interesting that as the population size increases, the number of worse performance counts, particularly for the "DE/t2b/1", "DE/best/2", and "DE/rand/2" mutation schemes, increase dramatically.

The best error value for each benchmark function family is illustrated in Fig. 7. The dash line separates the uni-modal, multi-modal, and composite, benchmark functions types. For the uni-modal and multi-modal functions, the VRMF method with the "DE/t2b/1" mutation scheme and $N_P = 6$ has the best performance. For the composite functions, the SRMF method with the "DE/t2b/1" mutation scheme and $N_P = 6$ has the best performance. In overall, the "DE/best/1" mutation scheme with population size of $N_P = 5$ is recommended as the well-performance scheme among the all. Further analysis are conducted on "DE/best/1" scheme in deep, including the popular scheme "DE/rand/1".

The best value so far of the MDE, MDESM, MDEVM, μJADE, SaDE, and CoDE algorithms for $N_P = 5$ and $D = 50$ are presented in Fig. 8. The method-r and method-b represents DE/rand/1 and DE/best/1 mutation schemes, respectively. As an example, the SaDE and MDEVM-r method have similar best so far values for f_5. The SaDE has converged earlier, while the MDEVM for both mutation schemes converge in further generations. This shows how the MDEVM method can gradually use its diversity enhancement method to achieve better solutions. We can see the same situation for f_6. The MDEVM-r has the most best so far value at the early generations, but gradually converges to a similar best so far value as SaDE, CoDE and MDEVM-b. Similar behaviour is observable for f_{11} and f_{12}. The SaDE and μJADE have better best value so far values than the MDEVM, but it limited to some functions such as f_{14} and f_{23}. The important is behaviour of MDEVM in exploring the landscape without using additive parameter or storage, just by randomizing the mutation scale factor. Functions f_{19} and f_{20} are good examples of MDEVM attempt to reach better solutions, when almost all other methods are converged or stagnated.

5.3. Experimental series 2: dimensionality effect

In this subsection, performance of the proposed MDEVM is compared with the MDE, MDESM, μJADE, SaDE, and CoDE algorithms with "DE/rand/1" and "DE/best/1" mutation vector schemes for dimension $D \in \{10, 30, 50, 100\}$. Comprehensive results are in the appendix (Tables 11–14). Summary of the Wilcoxon test results in terms of pair-wise comparisons with respect to MDEVM is reported in Table 6.

The results for micro-population size $N_P = 5$ clearly demonstrate that the proposed MDEVM has outperformed the other methods for different dimensions. The MDESM shows a better performance than the MDE, which is due to the SRMF diversity

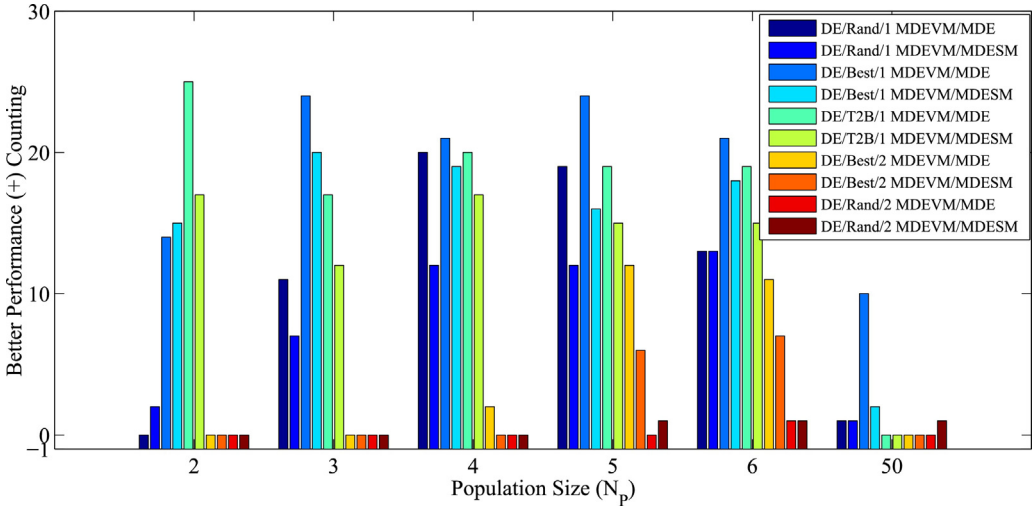
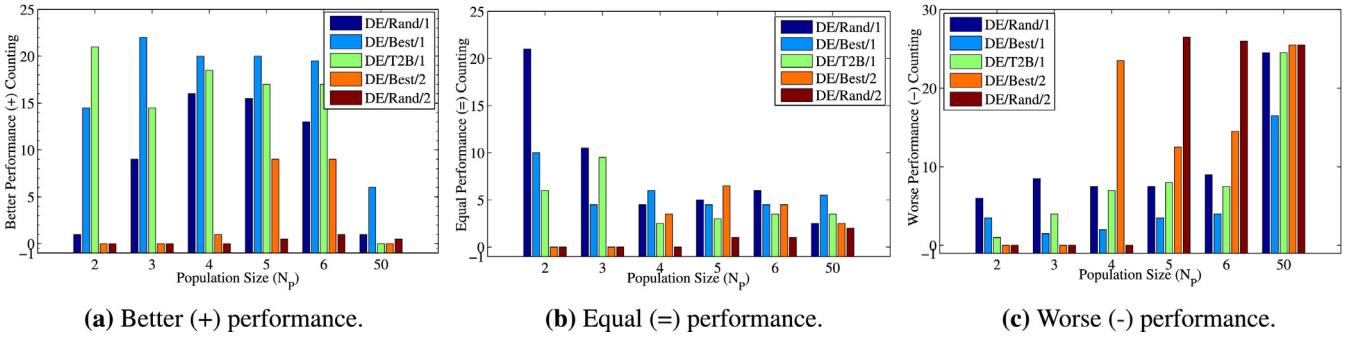


Fig. 5. The better (+) performance counting for the MDEVM vs. MDE and MDEVM vs. MDESM comparison for different mutation schemes and populations sizes.



(a) Better (+) performance. **(b)** Equal (=) performance. **(c)** Worse (-) performance.

Fig. 6. Average performance of MDEVM vs. MDE and MDEVM vs. MDESM methods for different mutation schemes and populations sizes.

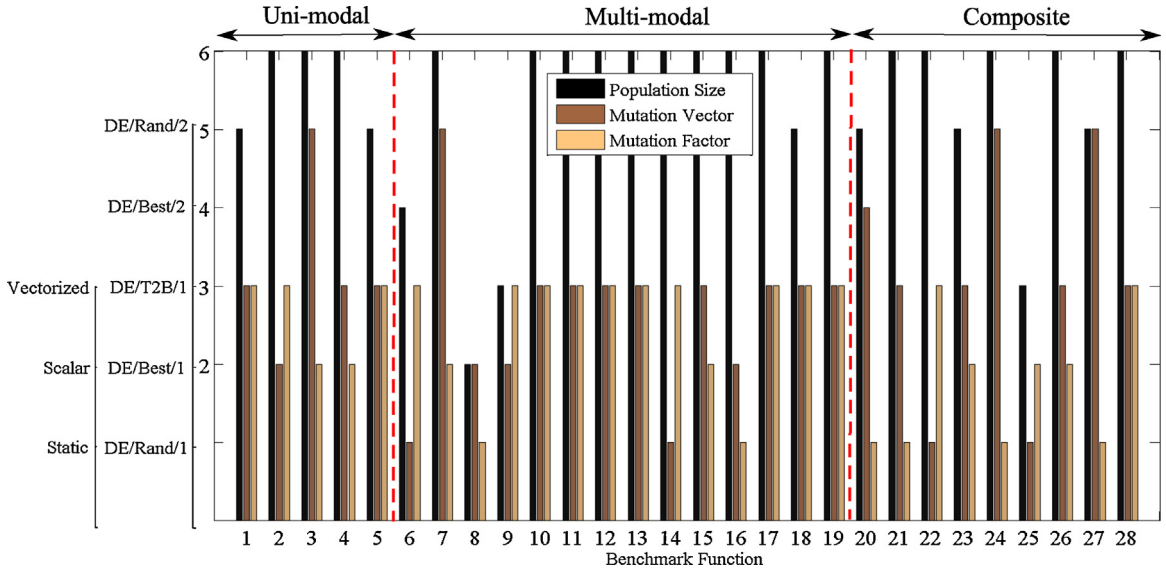


Fig. 7. Components of highest performance algorithm with respect to the best error for each benchmark function, and function families uni-modal (f_1–f_5), multi-modal (f_6–f_{19}), and composite (f_{20}–f_{28}).

enhancement technique used in this scheme. Both "DE/rand/1" and "DE/best/1" mutation schemes have competitive performances over all dimensions and MDE schemes. As the dimensionality of problems increases, the μJADE, CoDE, and SaDE methods provide competitive performance versus the MDEVM. Larger number of equal performance than the number of success/failure show similar performance over most of the functions. This shows that for high-dimensional problems, the adaptive method along with a small size of population can provide a good diversity. We can see the same situation with the MDEVM, where the diversification of

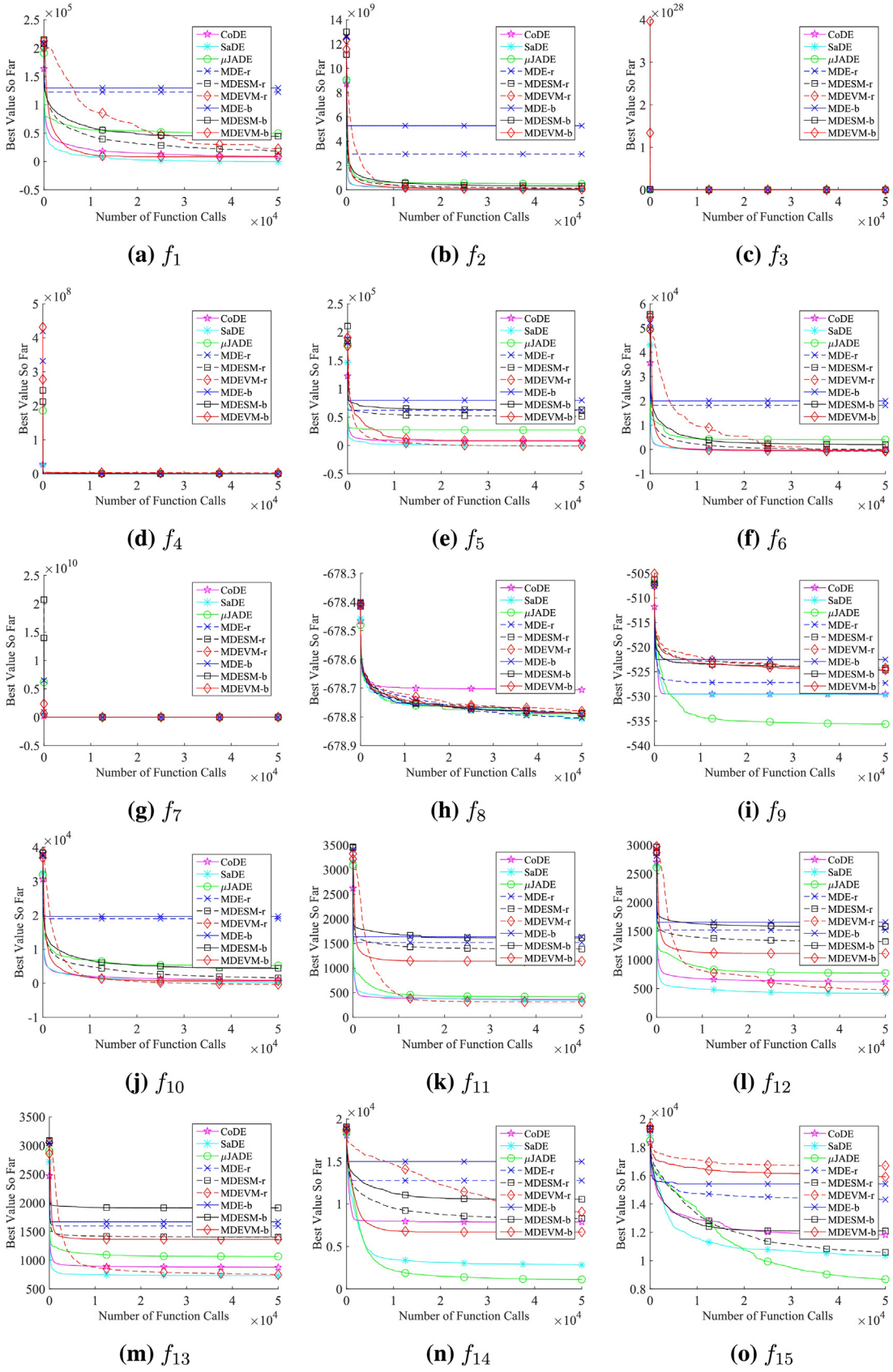


Fig. 8. Best value so far of the MDE, MDESM, MDEVM, μJADE, SaDE, and CoDE algorithms for $N_P = 5$ and $D = 50$. The method-r and method-b represents DE/rand/1 and DE/best/1 mutation schemes, respectively.

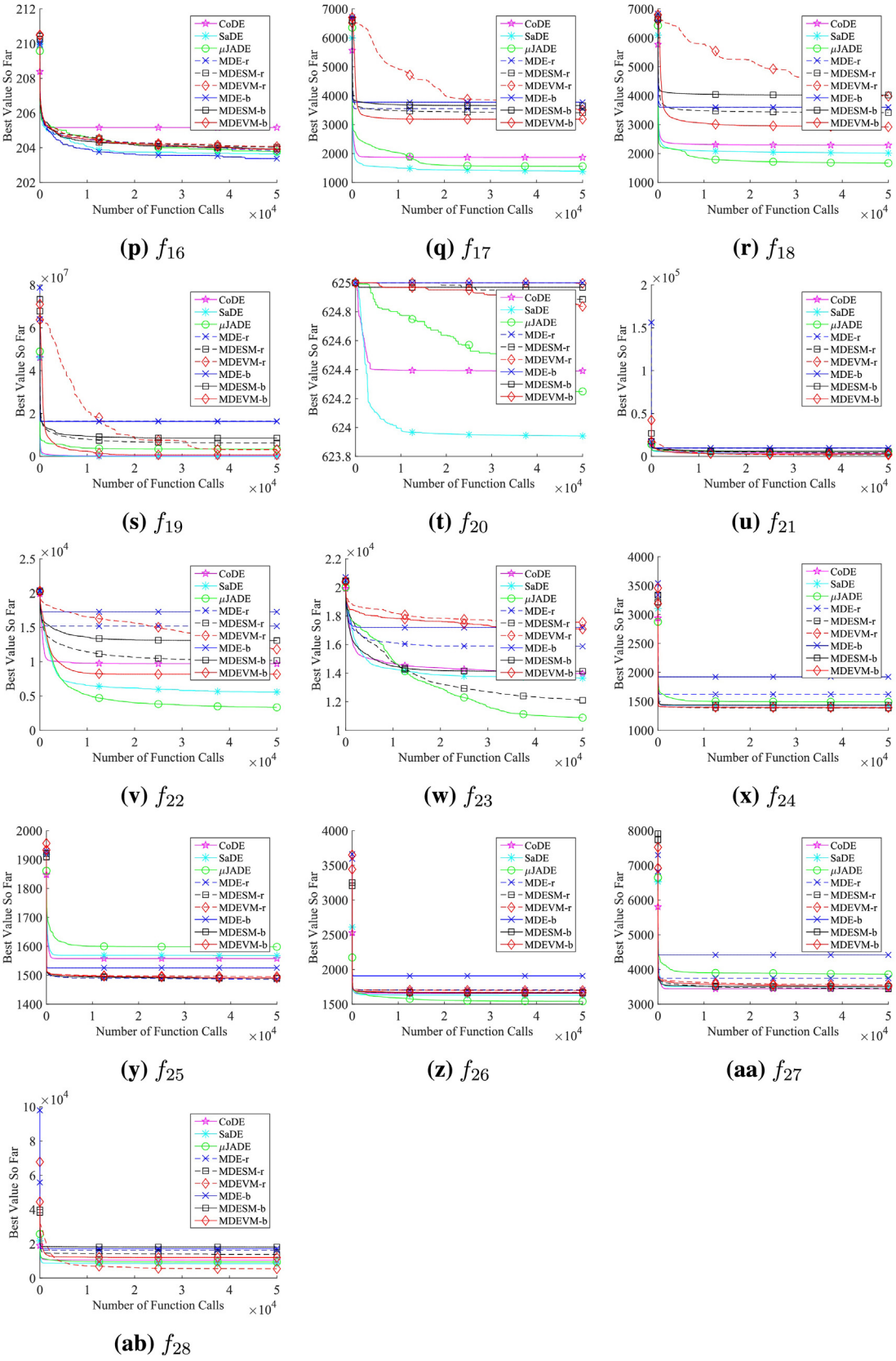


Fig. 8. (*Continued*)

Table 6
Number of Wilcoxon rank-sum test comparisons for MDEVM ($N_P = 5$) vs. MDE ($N_P = 5$), MDESM ($N_P = 5$), μJADE ($N_P = \{5, 8\}$), SaDE ($N_P = \{5, 50\}$), and CoDE ($N_P = \{5, 30\}$) for dimension $D \in \{10, 30, 50, 100\}$. The bold value represents the dominant performance result.

D	MV	MDE			MDESM			μJADE			SaDE			CoDE			μJADE			SaDE			CoDE		
								$N_P = 5$									$N_P = 8$			$N_P = 50$			$N_P = 30$		
		+	=	–	+	=	–	+	=	–	+	=	–	+	=	–	+	=	–	+	=	–	+	=	–
10	DE/Rand/1	**23**	3	2	**12**	13	3	**20**	8	0	**16**	12	0	**19**	8	1	**23**	3	2	**15**	3	10	**15**	4	9
	DE/Best/1	**26**	0	2	**24**	3	1	**18**	10	0	**13**	15	0	**17**	10	1	**23**	3	2	12	**4**	12	11	4	**13**
30	DE/Rand/1	**22**	4	2	**17**	5	6	**18**	8	2	10	**17**	1	**18**	9	1	**19**	4	5	6	4	**18**	2	2	**24**
	DE/Best/1	**21**	5	2	**20**	6	2	**16**	11	1	**10**	16	2	**15**	13	0	**20**	2	6	6	3	**19**	5	1	**22**
50	DE/Rand/1	**19**	2	7	**12**	8	8	**16**	10	2	**14**	13	1	**16**	12	0	**17**	4	7	7	3	**18**	4	1	**23**
	DE/Best/1	**24**	2	2	**16**	7	5	**12**	15	1	**8**	17	3	**13**	14	1	**13**	7	8	4	6	**18**	3	2	**23**
100	DE/Rand/1	**18**	3	7	**16**	5	7	**11**	15	2	**4**	11	3	**8**	19	1	**15**	3	10	3	4	**21**	1	2	**25**
	DE/Best/1	**20**	5	3	**15**	9	4	**10**	16	2	3	**12**	3	**7**	21	0	**13**	4	11	3	4	**11**	1	2	**25**

the mutation factor for decision variables can help the small size population to increase its diversity, suitable for high-dimensional problems.

The MDEVM is compared with the default versions of μJADE ($N_P = 8$), SaDE ($N_P = 50$), and CoDE ($N_P = 30$) in Table 6. The MDEVM with $N_P = 5$ has more significant success number than the μJADE with $N_P = 8$. However, as the problem dimension increases, μJADE shows a growing significant success, from two in $D = 10$ to 10 in $D = 100$. The SaDE with $N_P = 50$ has many more significant success than MDEVM with $N_P = 5$. Similar trend is obvious for CaDE with $N_P = 30$, such that as the problem dimension increases (need for more diversity), the CaDE with $N_P = 30$ achieves more significant success. Comparing to the SaDE with $N_P = 5$ and CaDE with $N_P = 30$, such performance is due to added diversity to the population by adding more individuals and of course, forcing more computational cost.

5.4. Experimental series 3: range of mutation factor analysis

The most common mutation factor in the literature is $F = 0.5$, selected from the recommended range $F \in [0, 2]$ [81]. Recently, different values for F and its range have been proposed, such as $F = 0.7$ in [4] and $F \in [0.1, 1.5]$ in [6]. Some experiments are conducted in this subsection to analyse affect of mutation factor range on the performance of the MDESM and MDEVM. The best error, standard deviation, and Wilcoxon rank-sum test results by considering $N_P = 5$, $D = 30$, mutation vector schemes "DE/rand/1" and "DE/best/1", and the MDESM and MDEVM as reference methods are presented in Table 7. The mutation factor ranges are considered as $F \in [0, 2]$ and $\mathbf{F} \in [0, 2]$ for MDESM and MDEVM, respectively. The $\text{MDESM}_{[0,2]}$ has competitive performance with $\text{MDESM}_{[0.1,1.5]}$. However, the $\text{MDEVM}_{[0.1,1.5]}$ has outperformed the $\text{MDESM}_{[0,2]}$ due to the delivered diversity by the VRMF into the $\text{MDEVM}_{[0.1,1.5]}$. The results of comparing $\text{MDEVM}_{[0,2]}$ with the $\text{MDESM}_{[0.1,1.5]}$ and $\text{MDEVM}_{[0.1,1.5]}$ demonstrate that selecting $\mathbf{F}$ in the interval [0, 2] has a better performance than the limited interval [0.1, 1.5]. The comparison between the $\text{MDEVM}_{[0,2]}$ and $\text{MDEVM}_{[0.1,1.5]}$ also shows almost equal performance. Overall, better performance of the $\text{MDEVM}_{[0,2]}$ method is obvious, since the it has diversity served from both VRMF and wider mutation scale factor interval [0, 2].

Performance of MDEVM for two different mutation scale factor intervals is studied. We have considered different problem types and dimensionalities and have reported the results based on Wilcoxon rank-sum test and considering MDEVM with $\mathbf{F} \in [0.1, 1.5]$ as reference method. The results in Table 8 show that the range $\mathbf{F} \in [0.1, 1.5]$ is more suitable for composite functions. This is while for multi-modal and uni-modal problems the $\mathbf{F} \in [0, 2]$ performs better. From the problem dimension view point, as the problem dimensionality increases, the $\mathbf{F} \in [0, 2]$ approach has better performance, which is due to the provided diversity in mutation vector generation.

5.5. Experimental series 4: population's diversity analysis

The VRMF technique can empower MDE to escape trapping in local optima and decrease the stagnation risk. In order to analyze the effect of randomization of mutation factor on the population diversity, by considering the centroid diversity measure and performance of the MDE algorithm, the best-value-so-far and population diversity plots of the MDE, MDESM, and MDEVM for composite functions f_{20} to f_{22} are presented in Fig. 9. The simulations are conducted for dimension $D = 100$, population size $N_P = 5$, and "DE/rand/1" and "DE/best/1". Conductive to have a better sense of analysis, the maximum number of function calls is considered $NFC_{Max} = 5000D$.

The MDEVM method for the mutation scheme "DE/rand/1" has the best performance for the function f_{20} as shown in Fig. 9a, denoted by "B". The population diversities in Fig. 9d and for the "DE/best/1" mutation scheme in Fig. 9j, clearly show that while the MDE and MDESM are stagnated, due to almost static large value of centroid diversity value, the MDEVM for the "DE/rand/1" has escaped the stagnation, denoted by region "A", while trying to converge in generations denoted by region "B". When the diversity is high, and the performance of algorithm in finding the solution is almost static with respect to the best-value-so-far measure, the population is considered stagnated. For situation of trapping in a local minimum, the population is not divert and the diversity is low, while having a poor best-value-so-far performance.

For the f_{21} case, the MDEVM method using the "DE/rand/1" and "DE/best/1" schemes has the best performance, as shown in Fig. 9b and h. The MDE algorithm is trapped in local minimum for both mutation schemes, while the MDESM method has better capability to escape from both stagnation and local optimum trapping, denoted by region "C" in Fig. 9e and k. The MDEVM has the best best-value-so-far for both mutation schemes. For the "DE/rand/1" mutation scheme, the population's diversity shows a similar convergence trend to the MDESM method, but has achieved a much better best-value-so-far at the beginning generations (i.e., exploration phase) and then trapped in the local minimum, as denoted in region "C" of Fig. 9b. The same performance is obvious for the "DE/best/1" mutation scheme as shown in Fig. 9h, where in region "A" it is converged to a solution. The corresponding diversity measure is well-illustrated in Fig. 9k. In region "A", which is the exploration phase, the population's diversity is decreased and it is converged, as shown in region "B". In "D", it has trapped but recovered fast to the same level as region "B".

The exploring ability of VRMF is well illustrated for the benchmark function f_{22} in Fig. 9c and i. In Fig. 9c, it is clear that the VRMF

Table 7
Performance results of the MDESM and MDEVM for mutation scale factor interval of [0, 2] and [0.1, 1.5] with $N_P = 5$ and $D = 30$.

Method	MV	$MDESM_{[0.1,1.5]}$			$MDEVM_{[0.1,1.5]}$		
		+	=	–	+	=	–
$MDESM_{[0,2]}$	DE/rand/1	14	12	2	6	7	15
	DE/best/1	14	14	0	7	9	12
$MDEVM_{[0,2]}$	DE/rand/1	19	5	4	6	9	13
	DE/best/1	19	6	3	19	3	6

Table 8
Performance comparison between MDEVM with $\mathbf{F} \in [0.1, 1.5]$ and MDEVM with $\mathbf{F} \in [0, 2]$ for uni-modal(f_1–f_5), multi-modal(f_6–f_{20}), and composite (f_{21}–f_{28}) functions for $D \in \{10, 30, 50, 100\}$.

F	D											
	10			30			50			100		
	+	=	–	+	=	–	+	=	–	+	=	–
f_1–f_5	3	0	2	1	1	3	1	1	3	1	0	4
f_6–f_{20}	7	1	8	2	5	9	1	4	11	3	4	9
f_{21}–f_{28}	4	1	2	3	3	1	3	2	2	2	3	2
Total	14	2	12	6	9	13	5	7	16	6	7	15

technique has escaped the DE-algorithm from stagnation (denoted by "A") approximately at $NFC = 3 \times 10^5$ and with a sudden movement, as denoted by region "B", it has reached a better performance than the other methods in region "C". This is clearly shown in Fig. 9f, that the MDEVM algorithm is rescued from stagnation (region "A") and gradually converging as shown in regions "B" and "C". This is while the MDE algorithm is completely trapped in a local minimum, since its best-value-so-far remains constant for all *NFCs* and the population diversity is extremely low for all generations, i.e. almost $1e{-}28$ in Fig. 9.f. The MDESM has tried to converge (part "D" of Fig. 9f) to the solution as presented in part "E" of Fig. 9c. However, its exploration is stopped as shown in parts "E" and "E" of the Fig. 9c and f, respectively, and no further improvements are achieved. For the "DE/best/1" mutation scheme, the MDE is trapped in a local minimum similar to the "DE/best/1" mutation scheme, as shown in Fig. 9i and l. The MDESM has achieved better performance by converging its population toward a solution as denoted by regions "C" and "A" in Fig. 9i and l, respectively. In further generations, although it has spent some time in generations denoted by region "B" in Fig. 9l to find a better solution, but it has been trapped finally in a local minimum as illustrated in part "C" of the Fig. 9l. The MDEVM has experienced the similar trend as the MDESM (regions "A", "D", and "E" for centroid diversity in Fig. 9l), but with better performance from region "A" toward region "B" of Fig. 9i.

The centroid-based diversity measure along the best-so-far-value analysis clearly have demonstrated performance of the MDE, MDESM, and MDEVM algorithms in stagnation and local optimum trapping scenarios. The results clearly indicate a successful performance of the VRM approach in delivering diversity into the population. Particularly that after some generations where the algorithm is trapped in local optimum or stagnated, it is rescued and moved toward better solutions, while the other algorithms could not survive.

5.6. Experimental series 5: comparison of solution accuracy with higher number of function calls

In this subsection we compare performance of the MDE, MDESM, μJADE, SaDE, CoDE, and the MDEVM for a higher number of function calls, i. e. $NFC_{Max} = 5000D$, than the previous experiments. The algorithms are compared to solve the set of benchmark functions for $D = 30$ and $N_P = 5$. Summary of the-best-so-far value and Wilcoxon rank-sum results by considering the MDEVM as the reference algorithm is presented in Table 9. The comparison of number of "better", "equal", and "worse" performance results for $NFC_{Max} = 5000D$ and $NFC_{Max} = 1000D$ presented in Table 6 are shown using the arrows. The upward arrow demonstrates the difference between count of problems that are solved using the $NFC_{Max} = 5000D$ compared to the $NFC_{Max} = 1000D$. The results demonstrate that by allowing more number of function calls, the MDEVM algorithm could obtain better performance comparing to MDE and MDESM. As an example for "DE/best/1", the MDEVM could solve four more problems comparing to the MDE, i. e. an increase from 21 to 25, Table 9. The MDE could also increase its count by one versus MDEVM. In case of MDEVM versus the MDESM, the MDEVM has achieved two more "better" records, from 20 to 22, and two less failure from 2 to zero. For the "DE/rand/1" mutation scheme, the MDEVM has one more success than both the MDE and MDESM algorithms.

Best value so far of the MDE, MDESM, μJADE, SaDE, CoDE, and MDEVM methods for the "DE/rand/1" and "DE/best/1" mutation schemes and the composite functions f_{20} to f_{27} with $D = 30$ and $NFC_{Max} = 5000D$ are visualized in Fig. 10. The performance results show that by providing more number of function calls to the algorithm, we have more success in exploring the f_{20} and f_{23} functions. For f_{20}, the SaDE is already closer to the global solution, however, approximately after the $NFC = 7000$, it has a fall to a better solution. The same situation is observable for the MDESM with "DE/best/1" mutation scheme, but at $NFC = 6,000$. For f_{23}, we see a mild converges progress in all algorithms. The MDEVM method with "DE/rand/1" and "DE/best/1" mutation schemes show more alternations in the best value so far, due to its exploration capability. We can conclude that more number of function calls provide more opportunity to explore the problem landscape; however, the performance increase is not significant.

Table 9
Summary of Wilcoxon rank-sum test comparisons for MDEVM against MDE and MDESM with $N_P = 5$, $D = 30$, and $NFC_{Max} = 5000D$. The number next to the upward arrows present additional number of success comparing with the $NFC_{Max} = 1000D$ and the downward arrow represents the vice-versa. The ↔ represents no change.

MV	MDE			MDESM		
	+	=	–	+	=	–
DE/rand/1	**23** ↑1	3 ↓1	2 ↔0	**18** ↑1	4 ↓1	6 ↔0
DE/best/1	**25** ↑4	2 ↓3	1 ↓1	**22** ↑2	6 ↔0	0 ↓2

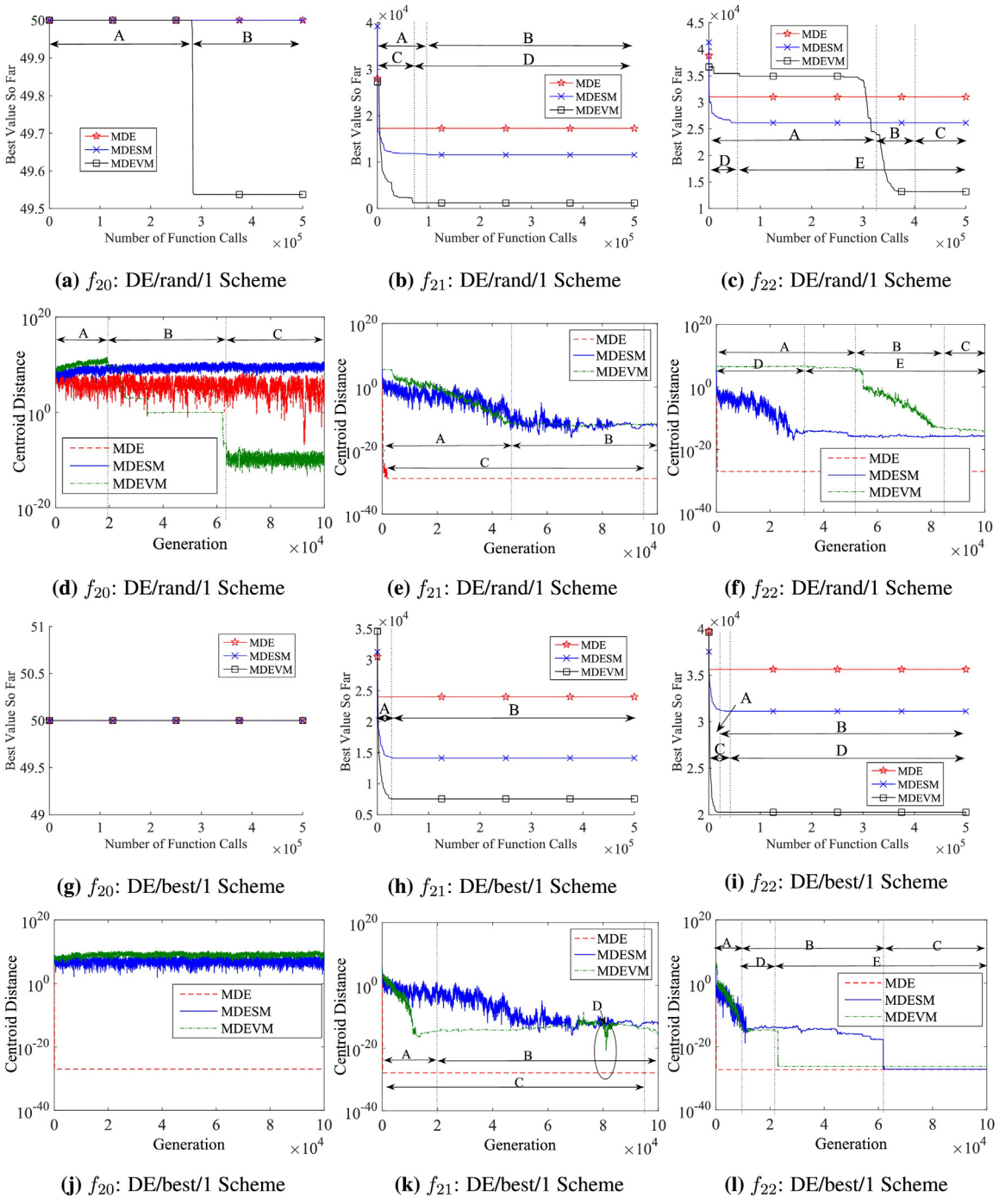


Fig. 9. Performance comparison and population centroid-based distance diversity analysis among the MDE, MDESM, and MDEVM for the maximum number of function calls $NFC_{Max} = 5000D$, dimension $D = 100$, population size $N_P = 5$, and DE/rand/1 and DE/best/1 mutation schemes.

5.7. Experimental Series 6: population-based parallel model analysis

The average CPU time for the population-based parallel model in Fig. 3 is presented in Table 10. The population size is $N_P = 5$ and $D = \{10, 100, 1000\}$. The results are average of 30 independent experiments on Intel Core i7-5930K Haswell-E 6-Core 3.5 GHz CPUs with 64 GB of RAM. The results show that the CPU times are totally competitive. The non-parallel implementation has a record of 7.03E−03 s, while the parallel implementation record is 1.41E−02 seconds for $D = 10$. As the dimensionality increases, we observe the gap between timing of both methods shrinks, such that

Table 10
Average CPU time for objective function evaluation. The results are averaged over 30 independent runs for $D = \{10, 100, 1000\}$, $N_P = 5$, and over all under study objective functions.

Mode	D		
	10	100	1000
Parallel	1.41E−02	1.16E−01	6.91E+00
Non-Parallel	7.03E−03	1.13E−01	6.98E+00

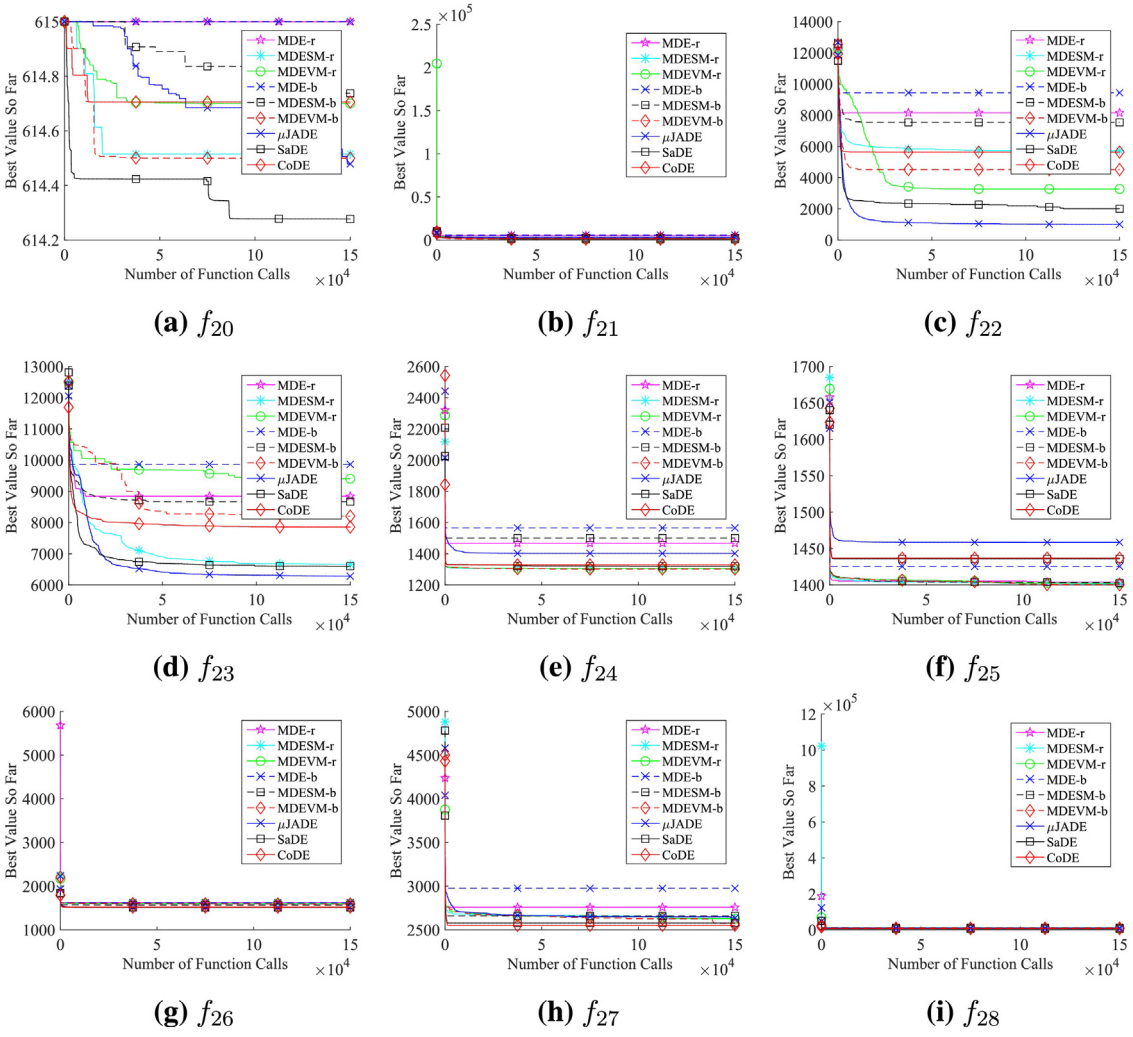


Fig. 10. Performance comparison among the MDE, MDESM, MDEVM, μJADE, SaDE, and CoDE for maximum number of function calls $NFC_{Max} = 5000D$, $D = 30$, $N_P = 5$, and DE/rand/1 (denoted as method-r) and DE/best/1 (denoted as method-b) mutation schemes.

for $D = 1000$, the parallel and non-parallel timing is 6.91E+00 s and 6.98E+00 s, respectively. It is expected that as the population size and dimensionality increases, the parallel mode runs faster than the non-parallel mode.

6. Conclusion and future work

Our proposed method is an enhanced version of the micro-differential evolution (MDE) based on the important capability of the mutation factor to increase diversity of the population, i.e. the MDE using vectorized random mutation factor (MDEVM) algorithm. In contrast to the standard MDE, the mutation factor is selected randomly for each decision variable of each individual in the population. In this case, the population can provide much higher diversity during the search process. We have conducted experiments for different schemes of the mutation factor. The results demonstrate capability of the proposed MDEVM method in solving complex optimization problems with very small population size. The MDEVM has competitive performance compared with state of the art such as μJADE, SaDE, and CoDE. Experiments demonstrate the exploration ability of MDEVM through generation, while other methods are stagnated or trapped in a local solution. The population distributed version of the model is also proposed, which the parallelization can occur at either objective function evaluation, mutation vector generation or both. The results show that for small size population, parallelization at either of the steps accelerates the CPU time; however, parallelization at both steps takes more CPU time than serial execution. Our study about allowable maximum number of function calls show that it provides more opportunity to explore the problem landscape; however, it does not increase the performance significantly.

The results show beneficial diversity enhancement for DE algorithm with small population size for different type of problems such as Quadratic ill-conditioned, Asymmetrical, and Non-continuous. It is interesting to further investigate diversity enhancement for other type of problems, such as ill-conditioned non-quadratic functions, in further works. The results also show some DE operations such as DE/rand/1 and DE/Best/1 mutation schemes are beneficial, while some other DE operators show detrimental behaviour. Further investigation on this topic is necessary. The MDEVM can be implemented on embedded system for remote and mobile applications.

Appendix A.

Table 11
Best error (Best), standard deviation (Std), and Wilcoxon rank-sum (W) test for MDEVM ($N_P = 5$) vs. MDE ($N_P = 5$), MDESM ($N_P = 5$), μJADE ($N_P = \{5, 8\}$), SaDE ($N_P = \{5, 50\}$), and CoDE ($N_P = \{5, 30\}$) for dimension $D = 10$.

f	MV	MDEVM		MDE			MDESM			μJADE			SaDE			CoDE			μJADE			SaDE			CoDE		
		$N_P = 5$																	$N_P = 8$			$N_P = 50$			$N_P = 30$		
		Best	Std	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W
1	DE/rand/1	4.78E−01	1.83E+03	2.02E+03	8.44E+03	+	4.95E−02	3.05E+03	=	3.15E+03	1.15E+03	+	2.48E+01	6.79E+01	+	1.54E+01	4.05E+01	+	4.82E+03	1.51E+03	+	9.17E−08	3.72E−08	–	9.10E−04	6.39E−04	–
	DE/best/1	5.01E−09	2.90E−07	2.86E+03	6.03E+03	+	2.28E−03	2.45E+03	+			+			+			+			+			+			+
2	DE/rand/1	5.76E+05	6.48E+06	2.50E+06	6.23E+07	+	2.21E+05	3.79E+06	=	8.89E+06	7.84E+06	+	5.11E+06	3.55E+06	+	2.47E+06	1.99E+06	+	4.04E+06	9.81E+05	+	1.12E+05	2.21E+05	–	2.52E+04	3.13E+04	–
	DE/best/1	1.13E+05	1.43E+06	7.76E+06	1.37E+08	+	1.73E+06	1.16E+07	+			+			+			+			+			=			–
3	DE/rand/1	8.70E+07	6.56E+09	6.12E+09	3.18E+13	+	4.49E+07	3.34E+10	=	3.97E+09	7.14E+09	+	3.47E+09	3.89E+09	+	2.18E+09	3.47E+09	+	2.53E+09	3.83E+09	+	7.87E+05	6.76E+05	–	1.31E+07	1.33E+07	–
	DE/best/1	9.93E+06	5.34E+09	9.41E+09	3.75E+17	+	3.30E+08	1.84E+16	+			+			+			+			+			–			+
4	DE/rand/1	2.09E+04	1.29E+05	1.59E+04	4.32E+04	–	4.34E+03	5.68E+04	–	2.15E+04	9.85E+03	=	2.17E+04	9.74E+03	=	1.92E+04	1.03E+04	=	2.72E+04	1.23E+04	+	1.22E+03	9.33E+02	–	1.77E+02	1.75E+02	–
	DE/best/1	9.06E+03	1.75E+05	1.18E+04	4.15E+04	+	1.00E+04	4.15E+04	+			+			+			+			+			–			–
5	DE/rand/1	1.34E+00	7.28E+03	5.14E+02	1.00E+04	+	1.58E+01	2.17E+04	=	8.25E+02	8.74E+02	+	5.17E+00	8.23E+00	+	2.02E+02	4.83E+02	+	5.08E+02	5.84E+02	+	3.17E−05	2.49E−05	–	9.85E−03	5.99E−03	–
	DE/best/1	8.35E−09	3.48E−01	5.45E+02	2.37E+04	+	5.93E+01	1.58E+04	+			+			+			+			+			+			+
6	DE/rand/1	9.11E−04	5.89E+01	2.68E+02	6.73E+02	+	4.33E+00	8.13E+01	=	2.36E+02	1.53E+02	+	5.47E+01	3.53E+01	+	5.88E+01	5.36E+01	+	2.26E+02	1.17E+02	+	3.62E+00	4.12E+00	+	5.64E+00	4.49E+00	+
	DE/best/1	5.14E−02	3.37E+01	2.20E+02	1.40E+03	+	2.41E+00	2.73E+02	+			+			+			+			+			+			+
7	DE/rand/1	1.70E+01	5.30E+01	5.44E+01	6.34E+05	+	6.21E+01	1.63E+02	+	6.28E+01	4.40E+01	+	5.72E+01	3.07E+01	+	9.74E+01	5.52E+01	+	4.21E+01	1.28E+01	+	8.48E+00	5.19E+00	–	1.43E+01	6.49E+00	–
	DE/best/1	7.92E+01	1.34E+02	6.91E+01	1.67E+06	–	1.10E+02	1.51E+04	+			=			=			–			–			–			–
8	DE/rand/1	2.02E+01	1.20E−01	2.03E+01	2.02E−01	+	2.03E+01	8.27E−02	=	2.05E+01	6.74E−02	=	2.05E+01	1.09E−01	=	2.08E+01	1.47E−01	=	2.05E+01	9.01E−02	=	2.05E+01	9.66E−02	=	2.05E+01	1.05E−01	=
	DE/best/1	2.02E+01	1.21E−01	2.04E+01	1.98E−01	+	2.03E+01	9.40E−02	=			=			=			=			=			=			=
9	DE/rand/1	3.55E+00	1.81E+00	5.66E+00	1.74E+00	=	7.20E+00	1.03E+00	=	6.08E+00	1.30E+00	+	8.79E+00	9.88E−01	+	8.79E+00	1.72E+00	+	6.50E+00	8.95E−01	+	7.38E+00	8.73E−01	+	5.92E+00	1.44E+00	+
	DE/best/1	5.15E+00	1.87E+00	8.35E+00	1.61E+00	+	8.39E+00	1.45E+00	+			=			+			+			+			+			+
10	DE/rand/1	1.15E+01	1.01E+02	1.35E+02	5.93E+02	+	1.03E+01	1.56E+02	=	2.95E+02	1.72E+02	+	5.58E+01	5.39E+01	+	3.85E+01	3.56E+01	+	2.72E+02	6.92E+01	+	4.86E−01	1.79E−01	–	8.81E−01	1.31E−01	–
	DE/best/1	2.14E+00	3.14E+01	2.14E+02	1.10E+03	+	2.15E+01	1.36E+02	+			+			+			+			+			–			–
11	DE/rand/1	1.19E+01	4.51E+01	5.55E+01	8.78E+01	+	3.78E+01	8.18E+01	+	8.49E+01	9.32E+01	+	3.60E+01	3.83E+01	+	2.44E+01	1.36E+01	+	6.50E+01	2.29E+01	+	9.32E+00	1.85E+00	–	2.78E+00	1.40E+00	–
	DE/best/1	2.19E+01	6.22E+01	7.23E+01	1.04E+02	+	4.55E+01	1.04E+02	+			+			=			+			+			–			–
12	DE/rand/1	1.18E+01	5.57E+01	7.48E+01	8.44E+01	+	3.92E+01	5.32E+01	+	8.13E+01	1.20E+01	+	3.92E+01	1.56E+01	=	6.04E+01	3.27E+01	+	7.15E+01	2.82E+01	+	3.15E+01	5.53E+00	+	3.00E+01	8.29E+00	+
	DE/best/1	1.39E+01	7.02E+01	6.62E+01	1.07E+02	+	5.40E+01	8.90E+01	+			+			=			+			+			+			+
13	DE/rand/1	1.71E+01	4.58E+01	8.66E+01	7.03E+01	+	6.09E+01	6.01E+01	+	9.58E+01	2.58E+01	+	7.27E+01	1.89E+01	+	9.46E+01	5.76E+01	+	7.56E+01	4.78E+00	+	3.18E+01	6.52E+00	+	3.53E+01	9.80E+00	+
	DE/best/1	5.92E+01	6.68E+01	8.76E+01	1.14E+02	+	7.90E+01	9.00E+01	+			+			=			+			–			–			–
14	DE/rand/1	1.05E+02	3.08E+02	7.73E+02	5.11E+02	+	3.58E+02	3.67E+02	+	1.79E+02	1.42E+02	+	2.83E+02	2.93E+02	+	7.02E+02	2.36E+02	+	1.96E+02	1.36E+02	+	4.56E+02	1.38E+02	+	1.40E+02	6.30E+01	+
	DE/best/1	1.43E+02	3.72E+02	1.19E+03	3.78E+02	+	6.75E+02	3.09E+02	+			=			=			+			+			+			=
15	DE/rand/1	7.99E+02	3.85E+02	8.24E+02	4.77E+02	+	5.20E+02	3.68E+02	–	7.76E+02	2.85E+02	=	1.01E+03	2.99E+02	+	1.83E+03	6.07E+02	+	7.69E+02	2.98E+02	=	1.57E+03	1.57E+02	+	1.47E+03	2.63E+02	+
	DE/best/1	5.01E+02	5.72E+02	1.62E+03	3.85E+02	+	8.12E+02	3.84E+02	+			+			+			+			+			+			+
16	DE/rand/1	6.64E−01	3.85E−01	1.50E+00	6.36E−01	+	3.26E−01	5.08E−01	=	1.47E+00	2.65E−01	+	1.43E+00	4.87E−01	+	3.24E+00	9.13E−01	+	1.56E+00	3.17E−01	+	1.64E+00	3.83E−01	+	1.57E+00	3.10E−01	+
	DE/best/1	1.07E+00	2.75E−01	8.47E−01	1.52E+00	–	5.90E−01	4.26E−01	–			=			=			+			+			+			+
17	DE/rand/1	2.13E+01	5.78E+01	4.83E+01	1.20E+02	+	4.21E+01	1.00E+02	+	5.80E+01	1.72E+01	+	3.42E+01	2.93E+01	=	6.55E+01	4.00E+01	+	4.62E+01	1.07E+01	+	2.39E+01	3.00E+00	+	1.62E+01	2.09E+00	–
	DE/best/1	2.78E+01	6.53E+01	7.32E+01	1.38E+02	+	4.93E+01	1.31E+02	+			+			=			+			+			–			–
18	DE/rand/1	4.61E+01	9.11E+01	1.14E+02	8.28E+01	+	4.81E+01	7.64E+01	+	1.22E+02	5.11E+01	+	5.55E+01	2.49E+01	=	9.34E+01	3.19E+01	–	7.65E+01	3.33E+01	+	4.76E+01	5.07E+00	=	5.53E+01	5.65E+00	+
	DE/best/1	3.93E+01	9.19E+01	1.69E+02	1.07E+02	+	9.87E+01	1.18E+02	+			+			=			–			+			+			+
19	DE/rand/1	1.10E+00	1.02E+04	8.07E+02	5.02E+05	+	1.92E+01	1.16E+04	+	7.18E+03	9.61E+03	+	5.20E+00	3.74E+00	+	6.58E+02	1.84E+03	+	1.58E+03	1.03E+03	+	1.87E+00	3.10E−01	+	1.81E+00	3.80E−01	+
	DE/best/1	6.10E−01	1.98E+02	7.88E+02	5.00E+05	+	6.36E+01	3.13E+05	+			+			=			+			+			+			+
20	DE/rand/1	4.05E+00	2.62E−01	3.80E+00	3.92E−01	=	4.06E+00	2.13E−01	+	3.71E+00	6.52E−01	=	3.59E+00	4.20E−01	=	4.03E+00	4.62E−01	=	3.71E+00	6.92E−01	–	3.49E+00	1.90E−01	–	3.64E+00	1.95E−01	–
	DE/best/1	3.51E+00	3.52E−01	3.80E+00	2.53E−01	+	3.70E+00	3.14E−01	+			=			=			+			=			=			=
21	DE/rand/1	3.04E+02	4.74E+01	5.31E+02	2.69E+02	+	2.22E+02	1.38E+02	=	4.83E+02	3.94E+01	=	4.02E+02	2.58E+00	=	3.56E+02	7.77E+01	=	4.76E+02	2.90E+01	+	4.00E+02	3.58E−08	+	3.11E+02	1.16E+02	=
	DE/best/1	1.00E+02	6.27E+01	5.42E+02	2.67E+02	+	1.13E+02	9.08E+01	+			+			=			+			+			+			+
22	DE/rand/1	2.23E+02	4.99E+02	9.11E+02	4.42E+02	+	6.71E+02	3.59E+02	+	3.40E+02	2.73E+02	=	3.53E+02	1.49E+02	=	1.02E+03	4.77E+02	=	2.40E+02	1.86E+02	+	6.81E+02	1.34E+02	+	3.85E+02	1.37E+02	+
	DE/best/1	4.20E+02	2.70E+02	1.57E+03	4.03E+02	+	7.38E+02	4.02E+02	+			=			=			–			+			+			–
23	DE/rand/1	8.23E+02	4.63E+02	9.22E+02	5.33E+02	+	9.04E+02	4.41E+02	+	1.43E+03	2.44E+02	+	1.45E+03	5.09E+02	+	1.62E+03	8.54E+02	+	1.34E+03	1.65E+02	+	1.80E+03	2.34E+02	+	1.58E+03	2.52E+02	+
	DE/best/1	7.65E+02	5.20E+02	2.06E+03	3.39E+02	+	1.22E+03	4.87E+02	=			+			+			+			+			+			+
24	DE/rand/1	2.10E+02	4.78E+00	2.19E+02	6.04E+00	+	2.17E+02	4.45E+00	=	2.17E+02	2.49E+01	=	2.12E+02	3.12E+01	=	2.23E+02	9.36E+00	=	2.21E+02	4.23E+00	+	2.01E+02	1.46E+01	=	2.07E+02	1.47E+01	=
	DE/best/1	1.58E+02	1.28E+01	2.21E+02	4.32E+01	+	2.21E+02	4.18E+00	+			+			+			+			+			+			+
25	DE/rand/1	2.20E+02	2.22E+00	2.18E+02	4.38E+00	–	2.17E+02	4.02E+00	=	2.24E+02	4.48E+00	=	2.12E+02	2.62E+01	=	2.22E+02	6.07E+00	=	2.18E+02	4.57E+00	=	2.04E+02	5.19E+00	–	2.12E+02	4.95E+00	=
	DE/best/1	2.17E+02	2.43E+00	2.23E+02	4.17E+00	+	2.20E+02	4.76E+00	=			=			=			=			=			–			=
26	DE/rand/1	2.00E+02	5.76E+01	1.73E+02	6.43E+01	=	1.78E+02	5.05E+01	–	2.08E+02	4.80E+01	=	1.83E+02	5.69E+01	=	1.93E+02	7.45E+01	=	1.53E+02	2.79E+01	–	1.40E+02	1.66E+01	–	1.63E+02	3.28E+01	–
	DE/best/1	1.54E+02	7.71E+01	2.04E+02	1.04E+02	+	2.00E+02	4.49E+01	+			=			=			=			–			+			+
27	DE/rand/1	4.29E+02	4.48E+01	5.41E+02	2.30E+02	+	4.09E+02	2.15E+02	=	5.31E+02	9.93E+01	+	4.95E+02	1.10E+02	=	5.89E+02	1.04E+02	+	4.59E+02	8.34E+01	+	3.09E+02	5.73E+00	–	3.71E+02	6.10E+01	–
	DE/best/1	4.00E+02	4.48E+01	5.75E+02	2.31E+02	+	4.91E+02	8.40E+01	+			+			=			+			+			–			–
28	DE/rand/1	3.10E+02	2.28E+02	7.02E+02	4.30E+02	+	4.12E+02	3.03E+02	+	8.52E+02	9.79E+01	+	7.22E+02	1.76E+02	+	7.74E+02	2.02E+02	+	7.96E+02	7.44E+01	+	3.00E+02	2.86E−02	=	2.92E+02	4.50E+01	–
	DE/best/1	1.00E+02	4.63E+02	9.18E+02	3.68E+02	+	7.97E+02	4.04E+02	+			+			+			+			+			+			+

Table 12
Best error (Best), standard deviation (Std), and Wilcoxon rank-sum (W) test for MDEVM ($N_P = 5$) vs. MDE ($N_P = 5$), MDESM ($N_P = 5$), μJADE ($N_P = \{5, 8\}$), SaDE ($N_P = \{5, 50\}$), and CoDE ($N_P = \{5, 30\}$) for dimension $D = 30$.

f	MV	MDEVM		MDE			MDESM			μJADE			SaDE			CoDE			μJADE			SaDE			CoDE		
		$N_P = 5$																	$N_P = 8$			$N_P = 50$			$N_P = 30$		
		Best	Std	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W
1	DE/rand/1	4.51E+00	4.02E+03	4.33E+04	1.62E+04	+	6.64E+02	5.48E+03	+	3.30E+04	1.07E+04	+	7.21E+02	9.62E+02	+	3.47E+03	5.01E+03	+	2.72E+04	6.36E+03	+	9.54E−09	7.78E−10	−	1.03E−04	9.31E−05	−
	DE/best/1	6.29E−04	2.13E+03	5.91E+04	1.88E+04	+	2.15E+03	1.04E+04	+			+			+			+			+			−			−
2	DE/rand/1	1.29E+07	2.46E+07	2.69E+08	7.63E+08	+	2.17E+07	4.76E+07	+	1.44E+08	1.66E+08	+	3.97E+07	1.49E+07	+	4.45E+07	3.66E+07	+	3.58E+07	6.60E+06	+	6.64E+06	3.04E+06	−	4.09E+06	2.20E+06	−
	DE/best/1	6.33E+06	2.84E+07	5.93E+08	1.07E+09	+	2.33E+07	1.22E+08	+			+			+			+			+			=			−
3	DE/rand/1	2.06E+10	6.46E+10	1.42E+11	2.35E+21	+	1.37E+10	5.27E+10	=	5.17E+16	1.42E+17	+	3.89E+10	2.05E+10	+	3.40E+10	1.30E+10	+	3.00E+10	1.44E+10	+	1.80E+08	2.14E+08	−	8.47E+07	7.24E+07	−
	DE/best/1	4.42E+09	1.42E+12	3.70E+12	3.61E+21	+	2.78E+10	1.51E+12	+			+			+			+			+			−			−
4	DE/rand/1	6.94E+04	2.73E+05	7.42E+04	7.24E+04	=	2.45E+04	4.31E+04	−	7.96E+04	4.90E+04	+	9.19E+04	2.68E+04	+	8.49E+04	2.09E+04	+	5.87E+04	9.61E+03	−	2.92E+04	5.43E+03	−	1.16E+04	3.70E+03	−
	DE/best/1	3.61E+04	3.92E+05	6.03E+04	8.00E+04	=	3.76E+04	1.72E+05	+			+			+			+			+			−			−
5	DE/rand/1	2.81E+02	2.60E+04	1.31E+04	3.96E+04	+	1.07E+03	2.99E+04	=	2.46E+04	2.90E+04	+	2.74E+02	3.57E+02	=	2.27E+03	2.86E+03	+	7.03E+03	4.48E+03	+	7.72E−04	1.41E−03	−	2.81E−03	1.25E−03	−
	DE/best/1	1.17E−01	1.09E+03	2.12E+04	5.30E+04	+	1.40E+03	4.66E+04	+			+			+			+			+			−			−
6	DE/rand/1	5.91E+01	1.31E+02	6.56E+03	5.66E+03	+	7.45E+01	3.64E+02	+	6.02E+03	3.34E+03	+	1.66E+02	4.25E+01	+	2.31E+02	1.13E+02	+	3.10E+03	3.34E+03	+	6.17E+01	2.31E+01	=	3.31E+01	1.81E+01	−
	DE/best/1	2.38E+01	3.94E+02	7.66E+03	6.57E+03	+	3.12E+02	1.76E+03	+			+			+			+			+			+			+
7	DE/rand/1	2.00E+02	2.16E+03	1.20E+03	7.09E+07	+	2.30E+02	5.01E+05	+	2.76E+04	5.50E+04	+	3.42E+03	5.80E+03	+	8.67E+03	1.43E+04	+	1.62E+03	2.09E+03	+	5.17E+01	2.03E+01	−	4.30E+01	1.52E+01	−
	DE/best/1	1.90E+02	1.27E+05	6.76E+04	9.31E+07	+	3.75E+02	1.40E+07	+			+			+			+			+			−			−
8	DE/rand/1	2.09E+01	4.30E−02	2.09E+01	6.22E−02	=	2.08E+01	6.63E−02	−	2.10E+01	5.76E−02	=	2.11E+01	5.76E−02	=	2.12E+01	1.06E−01	=	2.10E+01	9.02E−02	=	2.11E+01	4.24E−02	=	2.11E+01	4.07E−02	=
	DE/best/1	2.09E+01	4.46E−02	2.09E+01	7.99E−02	=	2.10E+01	4.24E−02	=			=			=			=			=			=			=
9	DE/rand/1	3.21E+01	2.08E+00	3.30E+01	2.89E+00	+	3.49E+01	2.00E+00	=	3.59E+01	1.82E+00	=	3.68E+01	2.22E+00	=	3.61E+01	2.72E+00	=	3.20E+01	3.53E+00	=	3.56E+01	1.36E+00	+	2.40E+01	6.42E+00	−
	DE/best/1	3.88E+01	1.21E+00	3.69E+01	2.73E+00	−	3.66E+01	2.36E+00	=			=			=			−			−			−			−
10	DE/rand/1	6.80E+01	2.47E+02	5.02E+03	3.26E+03	+	2.05E+02	5.62E+02	+	3.88E+03	2.80E+03	+	1.78E+02	1.12E+02	+	4.62E+02	2.66E+02	+	2.36E+03	1.13E+03	+	5.60E+00	3.16E+00	−	1.43E+00	2.75E−01	−
	DE/best/1	1.49E+01	1.47E+02	5.26E+03	4.42E+03	+	4.90E+02	8.66E+02	+			+			+			+			+			−			−
11	DE/rand/1	1.98E+02	2.19E+02	5.79E+02	3.58E+02	+	2.95E+02	2.42E+02	+	5.65E+02	2.83E+02	+	2.20E+02	7.64E+01	=	4.91E+02	1.73E+02	+	3.88E+02	2.67E+02	+	2.98E+01	5.00E+00	−	3.28E+01	5.61E+00	−
	DE/best/1	2.67E+02	2.85E+02	9.06E+02	3.12E+02	+	4.10E+02	4.37E+02	+			+			=			+			+			−			−
12	DE/rand/1	1.97E+02	1.61E+02	4.23E+02	2.90E+02	+	3.68E+02	2.78E+02	+	6.55E+02	2.48E+02	+	3.85E+02	1.24E+02	+	6.03E+02	2.38E+02	+	5.28E+02	5.64E+01	+	1.64E+02	1.91E+01	−	1.07E+02	5.43E+01	−
	DE/best/1	3.81E+02	2.29E+02	6.39E+02	3.59E+02	+	5.65E+02	2.57E+02	+			+			=			+			+			−			−
13	DE/rand/1	2.87E+02	1.52E+02	6.27E+02	3.03E+02	+	5.65E+02	2.99E+02	+	8.43E+02	2.80E+02	+	4.67E+02	1.38E+02	+	5.73E+02	1.32E+02	+	5.66E+02	9.80E+01	+	1.90E+02	1.68E+01	−	1.72E+02	3.71E+01	−
	DE/best/1	4.62E+02	1.73E+02	8.42E+02	3.20E+02	+	8.22E+02	2.90E+02	+			+			=			=			+			−			−
14	DE/rand/1	1.73E+03	1.25E+03	5.62E+03	6.94E+02	+	3.03E+03	8.82E+02	+	1.18E+03	4.99E+02	=	1.09E+03	3.29E+02	=	3.82E+03	5.43E+02	+	3.62E+02	3.16E+02	−	2.45E+03	3.19E+02	+	9.27E+02	3.34E+02	−
	DE/best/1	2.75E+03	6.82E+02	6.43E+03	7.14E+02	+	4.32E+03	6.56E+02	+			=			−			=			−			−			−
15	DE/rand/1	7.56E+03	3.90E+02	5.57E+03	8.75E+02	−	3.76E+03	7.84E+02	−	4.18E+03	8.69E+02	−	4.96E+03	7.69E+02	−	5.73E+03	1.27E+03	−	4.52E+03	2.83E+02	−	7.39E+03	2.50E+02	=	7.06E+03	8.82E+02	−
	DE/best/1	5.45E+03	8.65E+02	5.97E+03	8.23E+02	=	4.91E+03	7.68E+02	−			=			=			−			+			+			+
16	DE/rand/1	2.36E+00	3.06E−01	1.73E+00	5.80E−01	=	2.59E+00	3.34E−01	=	2.80E+00	5.18E−01	+	2.70E+00	5.48E−01	=	3.62E+00	9.14E−01	+	2.69E+00	3.78E−01	+	3.03E+00	2.47E−01	+	3.12E+00	2.83E−01	+
	DE/best/1	2.45E+00	2.91E−01	1.63E+00	6.92E−01	=	1.23E+00	6.39E−01	=			+			=			+			+			+			+
17	DE/rand/1	2.23E+02	3.70E+02	6.99E+02	4.43E+02	+	7.18E+02	4.77E+02	+	6.09E+02	3.66E+02	+	3.69E+02	9.45E+01	=	5.21E+02	1.73E+02	+	4.07E+02	1.30E+02	+	9.06E+01	7.72E+00	−	7.53E+01	6.77E+00	−
	DE/best/1	5.33E+02	3.94E+02	1.22E+03	3.55E+02	+	1.23E+03	4.90E+02	+			=			−			=			−			−			−
18	DE/rand/1	4.56E+02	2.95E+02	1.22E+03	3.81E+02	+	9.77E+02	3.03E+02	+	8.23E+02	4.84E+02	+	5.65E+02	1.36E+02	=	6.61E+02	1.21E+02	+	6.60E+02	8.96E+01	+	2.21E+02	1.46E+01	−	2.42E+02	1.96E+01	−
	DE/best/1	5.28E+02	4.51E+02	1.16E+03	4.49E+02	+	1.07E+03	5.27E+02	+			+			=			+			+			−			−
19	DE/rand/1	4.36E+02	1.01E+06	4.93E+05	7.07E+06	+	3.37E+04	3.34E+06	+	4.79E+05	6.47E+05	+	4.84E+02	6.54E+02	=	6.36E+04	6.35E+04	+	7.59E+05	7.39E+05	+	9.68E+00	1.56E+00	−	1.03E+01	1.29E+00	−
	DE/best/1	7.73E+01	4.89E+05	3.11E+05	1.16E+07	+	4.96E+04	4.91E+06	+			+			+			+			+			−			−
20	DE/rand/1	1.45E+01	2.17E−01	1.50E+01	3.98E−07	+	1.45E+01	1.25E−01	−	1.48E+01	2.10E−01	=	1.36E+01	1.06E+00	=	1.43E+01	7.08E−01	=	1.49E+01	1.83E−01	=	1.26E+01	2.54E−01	−	1.30E+01	2.59E−01	−
	DE/best/1	1.45E+01	1.71E−01	1.50E+01	0.00E+00	+	1.45E+01	8.95E−02	+			=			=			=			=			−			−
21	DE/rand/1	2.19E+02	4.20E+02	3.22E+03	1.07E+03	+	4.56E+02	5.79E+02	+	2.40E+03	3.77E+02	+	5.09E+02	9.11E+01	+	1.01E+03	6.02E+02	+	2.87E+03	3.62E+02	+	3.07E+02	6.92E+01	+	2.92E+02	7.95E+01	+
	DE/best/1	2.10E+02	3.38E+02	3.31E+03	1.20E+03	+	6.08E+02	7.53E+02	+			+			+			+			+			+			+
22	DE/rand/1	1.88E+03	1.53E+03	6.00E+03	8.71E+02	+	3.38E+03	8.44E+02	+	8.90E+02	9.34E+02	−	1.28E+03	6.97E+02	=	4.14E+03	6.12E+02	+	7.35E+02	8.37E+02	−	3.03E+03	4.64E+02	+	1.30E+03	5.21E+02	−
	DE/best/1	2.30E+03	6.90E+02	7.43E+03	6.57E+02	+	4.23E+03	7.75E+02	+			−			=			+			−			+			−
23	DE/rand/1	7.86E+03	4.43E+02	6.82E+03	8.80E+02	−	4.59E+03	7.76E+02	−	5.38E+03	6.35E+02	=	5.99E+03	1.30E+03	=	7.06E+03	1.55E+03	=	4.87E+03	1.16E+03	−	7.72E+03	3.59E+02	=	7.26E+03	1.07E+03	−
	DE/best/1	3.87E+03	1.64E+03	7.60E+03	6.71E+02	=	4.97E+03	9.57E+02	+			=			=			=			+			+			+
24	DE/rand/1	2.70E+02	9.58E+00	3.02E+02	1.20E+02	+	2.79E+02	1.69E+01	+	3.69E+02	3.59E+01	+	3.19E+02	1.30E+01	=	3.14E+02	9.69E+00	=	3.42E+02	2.88E+01	+	2.26E+02	7.35E+00	−	2.41E+02	1.03E+01	−
	DE/best/1	2.99E+02	2.85E+00	3.21E+02	2.07E+02	+	2.86E+02	1.93E+02	−			=			=			=			+			−			−
25	DE/rand/1	2.97E+02	3.09E+00	2.83E+02	9.27E+00	=	2.95E+02	4.42E+00	=	3.61E+02	2.21E+01	=	3.36E+02	1.48E+01	=	3.32E+02	1.62E+01	=	3.56E+02	1.69E+01	+	2.82E+02	2.19E+01	−	2.72E+02	9.57E+00	−
	DE/best/1	2.95E+02	4.34E+00	3.03E+02	1.80E+01	+	2.92E+02	5.40E+00	=			=			=			=			+			−			−
26	DE/rand/1	2.01E+02	3.79E+01	2.33E+02	5.78E+01	+	2.04E+02	6.00E+01	+	2.80E+02	9.67E+01	=	3.71E+02	6.03E+01	=	3.37E+02	9.27E+01	=	2.40E+02	7.77E+01	+	2.20E+02	4.85E+01	+	2.00E+02	8.71E−02	=
	DE/best/1	3.78E+02	9.85E+00	2.61E+02	1.24E+02	−	2.04E+02	7.92E+01	=			=			=			−			−			−			−
27	DE/rand/1	1.25E+03	3.96E+01	1.28E+03	1.09E+02	+	1.18E+03	6.33E+01	−	1.32E+03	1.07E+02	=	1.29E+03	1.33E+02	=	1.33E+03	5.11E+01	=	1.28E+03	6.51E+01	=	6.13E+02	8.11E+01	−	8.02E+02	1.19E+02	−
	DE/best/1	1.18E+03	4.34E+01	1.43E+03	6.97E+02	+	1.17E+03	9.37E+01	=			=			=			+			−			−			−
28	DE/rand/1	2.64E+03	1.50E+03	4.43E+03	2.97E+03	+	4.26E+03	3.39E+03	+	5.23E+03	1.28E+03	+	3.76E+03	7.61E+02	=	4.24E+03	1.40E+03	=	4.55E+03	8.14E+02	+	3.00E+02	1.89E−02	−	3.01E+02	3.64E−01	−
	DE/best/1	2.74E+03	2.54E+03	7.42E+03	9.88E+03	+	3.81E+03	4.38E+03	+			+			=			=			+			−			−

Table 13
Best error (Best), standard deviation (Std), and Wilcoxon rank-sum (W) test for MDEVM ($N_P = 5$) vs. MDE ($N_P = 5$), MDESM ($N_P = 5$), μJADE ($N_P = \{5, 8\}$), SaDE ($N_P = \{5, 50\}$), and CoDE ($N_P = \{5, 30\}$) for dimension $D = 50$.

f	MV	MDEVM		MDE			MDESM			μJADE			SaDE			CoDE			μJADE			SaDE			CoDE		
		$N_P = 5$																	$N_P = 8$			$N_P = 50$			$N_P = 30$		
		Best	Std	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W
1	DE/rand/1	1.03E−02	6.74E+04	8.34E+04	1.90E+04	+	1.90E+03	1.01E+04	+	5.12E+04	1.12E+04	+	1.00E+03	2.16E+03	+	1.05E+04	9.78E+03	+	3.62E+04	1.20E+04	+	3.16E−08	6.52E−08	–	7.69E−06	4.29E−06	–
	DE/best/1	2.91E+01	8.95E+03	7.79E+04	2.41E+04	+	1.06E+04	1.65E+04	+			+			+			+			+			–			–
2	DE/rand/1	2.15E+07	2.94E+07	7.54E+08	1.56E+09	+	3.87E+07	4.74E+07	+	4.81E+08	3.87E+08	+	6.93E+07	3.19E+07	+	7.82E+07	2.11E+07	+	1.07E+08	5.69E+07	+	1.45E+07	4.42E+06	–	5.48E+06	1.52E+06	–
	DE/best/1	2.04E+07	3.07E+07	1.86E+09	2.57E+09	+	1.09E+08	1.70E+08	+			+			+			+			+			–			–
3	DE/rand/1	2.42E+10	2.77E+11	4.00E+12	3.05E+18	+	4.86E+10	1.07E+11	=	9.04E+11	1.82E+12	+	6.33E+10	1.58E+10	+	8.36E+10	4.69E+10	+	2.53E+10	7.38E+09	=	1.92E+09	1.72E+09	–	1.14E+09	1.34E+09	–
	DE/best/1	2.45E+10	1.37E+11	2.43E+13	4.45E+20	+	7.63E+10	2.59E+11	+			+			+			=			–			–			–
4	DE/rand/1	2.81E+05	2.00E+06	1.31E+05	1.36E+05	–	5.63E+04	2.68E+05	–	9.84E+04	4.81E+04	–	1.16E+05	2.44E+04	=	1.01E+05	2.60E+04	=	9.12E+04	1.84E+04	–	4.71E+04	6.15E+03	–	1.62E+04	4.33E+03	–
	DE/best/1	1.16E+05	1.94E+06	1.27E+05	1.59E+05	+	7.04E+04	3.08E+05	–			=			=			=			–			–			–
5	DE/rand/1	1.25E−01	1.32E+02	1.90E+04	3.21E+04	+	1.09E+04	3.77E+04	+	2.80E+04	3.58E+04	+	2.92E+02	3.12E+02	+	8.55E+03	6.15E+03	+	7.97E+03	4.78E+03	+	3.42E−03	2.15E−03	–	1.31E−03	5.71E−04	–
	DE/best/1	2.77E+02	1.26E+04	3.48E+04	3.55E+04	+	1.59E+04	3.34E+04	+			=			+			+			+			–			–
6	DE/rand/1	4.34E+01	4.85E+01	9.58E+03	6.31E+03	+	4.09E+02	3.58E+02	+	4.95E+03	1.84E+03	+	2.92E+02	6.35E+01	+	5.85E+02	2.61E+02	+	2.65E+03	1.92E+03	+	6.91E+01	3.05E+01	+	4.69E+01	1.81E+00	+
	DE/best/1	1.25E+02	2.79E+02	1.03E+04	7.21E+03	+	9.12E+02	1.57E+03	+			=			+			+			+			–			–
7	DE/rand/1	1.92E+02	2.04E+02	3.86E+02	2.25E+05	+	6.23E+02	2.14E+05	+	8.57E+03	1.77E+04	+	4.78E+02	3.36E+02	+	5.59E+03	6.44E+03	+	6.58E+02	1.02E+03	+	7.15E+01	1.46E+01	–	8.65E+01	1.57E+01	–
	DE/best/1	2.77E+02	2.10E+04	2.53E+03	6.97E+06	+	3.02E+03	2.10E+07	+			+			+			+			+			–			–
8	DE/rand/1	2.11E+01	5.80E−02	2.11E+01	3.49E−02	+	2.11E+01	3.67E−02	=	2.12E+01	2.91E−02	=	2.12E+01	3.55E−02	=	2.13E+01	8.23E−02	=	2.12E+01	3.26E−02	=	2.12E+01	4.10E−02	=	2.12E+01	4.74E−02	=
	DE/best/1	2.11E+01	3.59E−02	2.11E+01	3.16E−02	=	2.11E+01	3.21E−02	=			=			=			=			=			=			=
9	DE/rand/1	7.23E+01	1.38E+00	6.55E+01	3.90E+00	–	6.82E+01	2.36E+00	=	6.44E+01	3.59E+00	=	7.03E+01	5.63E+00	=	7.05E+01	5.76E+00	=	6.39E+01	1.95E+00	–	6.39E+01	5.62E+00	–	4.21E+01	7.45E+00	–
	DE/best/1	6.52E+01	3.01E+00	7.11E+01	3.73E+00	+	6.97E+01	3.40E+00	=			=			=			=			=			=			–
10	DE/rand/1	1.09E+01	1.55E+02	1.34E+04	4.21E+03	+	8.51E+02	8.03E+02	+	5.72E+03	2.06E+03	+	6.82E+02	2.43E+02	+	1.39E+03	9.62E+02	+	3.23E+03	2.03E+03	+	2.29E+01	5.49E+00	+	3.25E+00	9.13E−01	–
	DE/best/1	2.34E+02	7.58E+02	1.08E+04	5.94E+03	+	2.03E+03	2.09E+03	+			+			+			+			+			–			–
11	DE/rand/1	2.84E+02	2.88E+02	1.40E+03	3.77E+02	+	1.03E+03	4.02E+02	+	8.17E+02	2.45E+02	+	7.46E+02	1.25E+02	+	7.61E+02	2.40E+02	+	3.87E+02	2.18E+02	+	6.42E+01	8.14E+00	–	6.23E+01	1.19E+01	–
	DE/best/1	9.72E+02	3.63E+02	1.22E+03	4.44E+02	+	9.15E+02	5.27E+02	–			=			=			–			–			–			–
12	DE/rand/1	2.75E+02	2.27E+02	1.32E+03	3.82E+02	+	9.93E+02	3.41E+02	+	1.07E+03	3.98E+02	+	7.17E+02	1.21E+02	=	9.17E+02	2.52E+02	+	8.85E+02	2.71E+02	+	2.19E+02	9.68E+01	–	1.09E+02	2.92E+01	–
	DE/best/1	1.01E+03	3.24E+02	1.21E+03	4.17E+02	+	1.27E+03	3.85E+02	+			=			–			–			–			–			–
13	DE/rand/1	5.78E+02	1.88E+02	9.64E+02	2.97E+02	+	1.21E+03	2.42E+02	+	1.27E+03	2.58E+02	+	9.33E+02	1.96E+02	+	1.08E+03	1.96E+02	+	9.60E+02	2.29E+02	+	3.93E+02	2.37E+01	–	3.15E+02	7.74E+01	–
	DE/best/1	9.30E+02	2.94E+02	1.28E+03	3.45E+02	+	1.38E+03	3.95E+02	+			=			=			=			=			–			–
14	DE/rand/1	4.20E+03	4.05E+03	1.08E+04	9.62E+02	+	6.50E+03	1.14E+03	=	1.21E+03	9.46E+02	–	2.92E+03	9.29E+02	=	7.97E+03	1.97E+03	=	1.23E+03	1.62E+03	–	5.27E+03	4.71E+02	+	1.66E+03	4.50E+02	–
	DE/best/1	5.11E+03	9.32E+02	1.31E+04	8.07E+02	+	8.96E+03	1.01E+03	+			–			–			=			–			=			–
15	DE/rand/1	1.54E+04	4.76E+02	1.24E+04	8.31E+02	–	7.41E+03	1.19E+03	–	8.58E+03	1.27E+03	=	1.02E+04	1.37E+03	=	1.17E+04	1.80E+03	=	7.83E+03	3.03E+02	–	1.41E+04	4.56E+02	–	1.34E+04	1.47E+03	–
	DE/best/1	1.47E+04	4.58E+02	1.40E+04	7.25E+02	–	1.01E+04	1.31E+03	–			=			=			–			–			=			–
16	DE/rand/1	3.56E+00	2.41E−01	3.01E+00	3.97E−01	–	3.28E+00	3.36E−01	=	3.78E+00	2.86E−01	=	3.64E+00	5.80E−01	=	5.16E+00	1.00E+00	=	3.56E+00	3.33E−01	=	3.89E+00	4.49E−01	+	3.86E+00	3.97E−01	+
	DE/best/1	2.99E+00	3.46E−01	1.94E+00	7.73E−01	–	2.89E+00	3.91E−01	=			=			=			+			+			+			+
17	DE/rand/1	4.68E+02	2.59E+03	2.00E+03	6.52E+02	=	1.56E+03	6.70E+02	=	1.26E+03	6.23E+02	+	1.08E+03	2.12E+02	+	1.56E+03	3.85E+02	+	8.30E+02	1.97E+02	+	1.81E+02	9.76E+00	–	1.44E+02	1.25E+01	–
	DE/best/1	1.71E+03	5.92E+02	2.10E+03	6.76E+02	+	2.21E+03	6.89E+02	+			=			=			–			–			–			–
18	DE/rand/1	1.03E+03	2.55E+03	2.41E+03	4.68E+02	=	2.15E+03	6.00E+02	=	1.27E+03	2.22E+02	=	1.62E+03	3.12E+02	=	1.89E+03	3.66E+02	=	1.70E+03	9.08E+02	+	4.44E+02	1.71E+01	–	4.47E+02	2.59E+01	–
	DE/best/1	1.71E+03	5.50E+02	2.24E+03	5.22E+02	+	1.57E+03	7.99E+02	–			=			=			=			=			–			–
19	DE/rand/1	1.25E+02	1.19E+07	1.94E+06	1.49E+07	+	9.26E+04	5.81E+06	+	3.36E+06	6.13E+06	+	2.59E+03	4.75E+03	+	1.65E+05	2.27E+05	+	7.06E+04	7.22E+04	+	2.46E+01	2.94E+00	–	2.00E+01	2.72E+00	–
	DE/best/1	1.56E+03	2.04E+06	2.30E+06	1.31E+07	+	7.44E+05	1.24E+07	+			=			+			+			+			–			–
20	DE/rand/1	2.49E+01	2.31E−02	2.50E+01	0.00E+00	+	2.45E+01	2.12E−01	–	2.42E+01	4.48E−01	=	2.39E+01	1.10E+00	–	2.44E+01	4.10E−01	=	2.42E+01	5.41E−01	=	2.20E+01	3.16E−01	–	2.26E+01	3.89E−01	–
	DE/best/1	2.45E+01	2.36E−01	2.50E+01	0.00E+00	+	2.45E+01	1.21E−01	+			=			–			=			=			–			–
21	DE/rand/1	2.02E+02	4.82E+02	6.54E+03	1.39E+03	+	2.02E+03	7.16E+02	+	5.04E+03	8.30E+02	+	1.47E+03	8.82E+02	+	2.95E+03	8.89E+02	+	4.02E+03	2.31E+02	+	8.66E+02	3.63E+02	+	9.09E+02	3.74E+02	+
	DE/best/1	5.55E+02	7.76E+02	6.52E+03	1.97E+03	+	3.13E+03	9.86E+02	+			+			+			+			+			+			+
22	DE/rand/1	4.82E+03	4.44E+03	1.23E+04	1.12E+03	+	7.32E+03	1.16E+03	=	2.55E+03	1.73E+03	=	4.79E+03	1.43E+03	=	8.89E+03	1.68E+03	+	1.57E+03	2.00E+03	–	6.54E+03	6.68E+02	+	2.24E+03	8.86E+02	–
	DE/best/1	4.76E+03	1.06E+03	1.45E+04	8.94E+02	+	1.05E+04	1.03E+03	+			=			=			+			–			+			–
23	DE/rand/1	1.52E+04	6.18E+02	1.27E+04	8.51E+02	–	8.37E+03	1.84E+03	–	9.98E+03	1.35E+03	=	1.27E+04	1.06E+03	=	1.31E+04	1.81E+03	=	1.00E+04	5.35E+02	–	1.50E+04	5.47E+02	=	1.42E+04	6.86E+02	–
	DE/best/1	1.48E+04	5.27E+02	1.48E+04	7.96E+02	=	9.77E+03	1.88E+03	–			=			=			–			=			=			=
24	DE/rand/1	3.86E+02	4.36E+00	3.90E+02	2.24E+02	+	3.65E+02	1.01E+01	–	4.92E+02	5.02E+01	+	4.30E+02	1.76E+01	=	4.26E+02	1.56E+01	=	4.40E+02	4.28E+01	+	2.75E+02	8.26E+00	–	2.75E+02	1.13E+01	–
	DE/best/1	3.64E+02	6.23E+00	5.03E+02	4.32E+02	+	3.73E+02	2.12E+02	=			+			=			+			+			–			–
25	DE/rand/1	3.85E+02	3.88E+00	3.63E+02	1.23E+01	–	3.73E+02	4.82E+00	–	4.98E+02	2.98E+01	+	4.68E+02	2.30E+01	+	4.58E+02	2.01E+01	+	4.62E+02	3.66E+01	+	3.69E+02	3.26E+01	=	3.31E+02	1.63E+01	–
	DE/best/1	3.83E+02	3.51E+00	3.92E+02	3.02E+01	+	3.72E+02	7.53E+00	=			+			+			+			+			=			–
26	DE/rand/1	4.90E+02	4.28E+00	4.59E+02	5.63E+01	–	2.10E+02	5.02E+01	–	3.40E+02	1.40E+02	=	4.27E+02	1.18E+02	=	4.54E+02	8.73E+01	=	2.06E+02	6.65E+00	–	2.67E+02	9.24E+01	–	3.24E+02	9.37E+01	–
	DE/best/1	2.04E+02	7.28E+01	4.87E+02	7.23E+02	+	2.09E+02	9.15E+01	=			=			+			=			+			+			+
27	DE/rand/1	2.04E+03	5.54E+01	2.16E+03	2.26E+02	+	1.98E+03	8.64E+01	–	2.56E+03	2.41E+02	=	2.20E+03	1.19E+02	=	2.14E+03	1.77E+02	=	2.29E+03	6.58E+01	+	1.17E+03	1.40E+02	–	1.19E+03	1.23E+02	–
	DE/best/1	2.11E+03	3.33E+01	2.33E+03	1.06E+03	+	2.02E+03	8.13E+01	=			=			=			+			–			–			–
28	DE/rand/1	4.35E+02	3.05E+03	1.00E+04	9.63E+03	+	7.42E+03	2.26E+03	+	8.01E+03	1.17E+03	+	7.06E+03	1.20E+03	+	9.08E+03	1.48E+03	+	8.16E+03	1.01E+03	+	5.66E+02	7.43E+02	+	1.02E+03	1.28E+03	+
	DE/best/1	2.86E+03	3.61E+03	1.05E+04	3.36E+03	+	1.05E+04	1.04E+04	+			+			+			+			+			–			–

Table 14
Best error (Best), standard deviation (Std), and Wilcoxon rank-sum (W) test for MDEVM ($N_P = 5$) vs. MDE ($N_P = 5$), MDESM ($N_P = 5$), μJADE ($N_P = \{5, 8\}$), SaDE ($N_P = \{5, 50\}$), and CoDE ($N_P = \{5, 30\}$) for dimension $D = 100$.

f	MV	MDEVM		MDE			MDESM			μJADE			SaDE			CoDE			μJADE			SaDE			CoDE		
										$N_P = 5$									$N_P = 8$			$N_P = 50$			$N_P = 30$		
		Best	Std	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W	Best	Std	W
1	DE/rand/1	1.92E+04	1.99E+05	2.25E+05	3.74E+04	+	6.00E+04	2.16E+04	=	1.41E+05	2.95E+04	+	6.45E+03	3.52E+03	–	5.24E+04	1.67E+04	+	1.10E+05	1.50E+04	+	1.32E−05	1.87E−05	–	2.50E−06	1.67E−06	–
	DE/best/1	2.16E+04	2.93E+04	1.87E+05	4.12E+04	+	1.12E+05	3.26E+04	+			+			–			+			+			–			–
2	DE/rand/1	9.68E+07	1.02E+08	7.70E+09	3.69E+09	+	3.37E+08	2.22E+08	+	1.18E+09	1.15E+09	+	2.19E+08	7.25E+07	+	3.33E+08	1.11E+08	+	4.20E+08	3.88E+08	+	2.93E+07	5.04E+06	–	1.19E+07	3.54E+06	–
	DE/best/1	1.75E+08	2.00E+08	1.18E+10	5.07E+09	+	8.77E+08	1.01E+09	+			+			=			=			+			–			–
3	DE/rand/1	3.00E+11	9.77E+11	5.51E+19	2.58E+25	+	1.40E+12	5.23E+15	+	4.56E+18	9.59E+18	+	2.01E+14	6.31E+14	+	3.52E+14	9.78E+14	+	6.52E+17	1.44E+18	+	2.22E+10	5.50E+09	–	8.64E+09	3.35E+09	–
	DE/best/1	3.46E+11	2.71E+12	3.08E+20	4.69E+28	+	4.27E+14	1.29E+20	+			+			+			+			+			–			–
4	DE/rand/1	4.38E+05	9.53E+05	3.19E+05	4.06E+05	–	1.23E+05	1.11E+06	–	2.16E+05	5.70E+04	=	2.27E+05	2.49E+04	=	1.94E+05	5.10E+04	–	2.19E+05	4.28E+04	–	1.10E+05	1.34E+04	–	3.80E+04	7.63E+03	–
	DE/best/1	2.69E+05	2.28E+06	3.09E+05	2.52E+05	+	1.94E+05	7.36E+05	–			=			=			=			–			–			–
5	DE/rand/1	2.40E+04	4.64E+04	1.08E+05	5.63E+04	+	4.18E+04	8.46E+04	+	7.33E+04	3.18E+04	+	4.47E+03	1.01E+04	–	3.05E+04	6.95E+03	=	2.70E+04	1.60E+04	+	6.81E−02	3.47E−02	–	2.86E−03	1.31E−03	–
	DE/best/1	9.10E+03	5.73E+04	1.12E+05	5.45E+04	+	7.49E+04	7.17E+04	+			+			–			+			+			–			–
6	DE/rand/1	1.92E+03	5.56E+04	5.03E+04	2.26E+04	+	5.41E+03	4.00E+03	+	2.42E+04	5.80E+03	+	2.05E+03	9.79E+02	=	5.22E+03	1.38E+03	+	2.07E+04	6.07E+03	+	3.43E+02	7.27E+01	–	2.40E+02	4.80E+01	–
	DE/best/1	2.93E+03	3.68E+03	6.42E+04	2.06E+04	+	1.73E+04	9.37E+03	+			+			=			+			+			–			–
7	DE/rand/1	2.87E+04	1.34E+07	4.99E+06	2.05E+09	+	1.14E+06	9.27E+07	+	1.53E+07	2.79E+07	+	7.95E+05	1.29E+06	+	4.09E+06	5.66E+06	+	5.95E+05	9.51E+05	+	1.39E+02	3.34E+01	–	1.04E+02	2.72E+01	–
	DE/best/1	4.01E+04	2.57E+08	2.64E+07	1.19E+10	+	6.69E+06	6.67E+09	+			+			+			+			+			–			–
8	DE/rand/1	2.13E+01	2.87E−02	2.12E+01	5.11E−02	–	2.13E+01	2.38E−02	=	2.14E+01	2.72E−02	=	2.14E+01	1.53E−02	=	2.14E+01	4.50E−02	=	2.14E+01	1.34E−02	=	2.14E+01	3.04E−02	=	2.14E+01	1.80E−02	=
	DE/best/1	2.13E+01	2.86E−02	2.12E+01	5.01E−02	–	2.12E+01	4.19E−02	=			=			=			=			=			=			=
9	DE/rand/1	1.61E+02	1.84E+00	1.43E+02	7.41E+00	–	1.48E+02	4.58E+00	–	1.52E+02	4.84E+00	=	1.59E+02	4.66E+00	=	1.59E+02	5.55E+00	=	1.48E+02	5.30E+00	–	1.44E+02	1.49E+01	–	9.31E+01	7.40E+00	–
	DE/best/1	1.39E+02	5.13E+00	1.51E+02	5.18E+00	+	1.50E+02	6.21E+00	=			=			=			=			+			=			–
10	DE/rand/1	2.51E+03	1.56E+03	3.56E+04	9.79E+03	+	5.56E+03	2.76E+03	+	1.66E+04	7.19E+03	+	2.35E+03	1.33E+03	=	4.79E+03	1.99E+03	=	8.52E+03	5.26E+03	+	8.79E+01	2.36E+01	–	2.16E+01	7.35E+00	–
	DE/best/1	5.26E+03	2.57E+03	3.98E+04	9.77E+03	+	1.19E+04	4.27E+03	+			+			=			=			+			–			–
11	DE/rand/1	2.44E+03	8.08E+02	3.78E+03	9.22E+02	+	2.94E+03	1.00E+03	+	2.23E+03	7.07E+02	=	2.07E+03	3.54E+02	=	2.68E+03	6.03E+02	=	1.93E+03	7.44E+02	-	1.76E+02	1.69E+01	-	1.41E+02	2.15E+01	-
	DE/best/1	2.49E+03	9.43E+02	4.29E+03	9.79E+02	+	3.68E+03	1.01E+03	+			=			=			=			-			-			-
12	DE/rand/1	2.12E+03	5.59E+02	3.81E+03	1.15E+03	+	3.03E+03	6.86E+02	+	2.66E+03	7.98E+02	=	2.35E+03	4.00E+02	=	3.08E+03	6.07E+02	=	2.51E+03	8.06E+02	+	4.61E+02	6.33E+01	-	3.37E+02	4.82E+01	-
	DE/best/1	3.56E+03	5.57E+02	4.07E+03	7.65E+02	+	3.44E+03	9.49E+02	-			=			=			=			-			-			-
13	DE/rand/1	2.59E+03	5.68E+02	3.46E+03	8.68E+02	+	3.27E+03	8.76E+02	+	3.18E+03	1.18E+03	=	2.68E+03	4.82E+02	=	3.17E+03	3.73E+02	=	2.94E+03	7.43E+02	+	1.00E+03	9.06E+01	-	6.40E+02	7.87E+01	-
	DE/best/1	2.93E+03	7.07E+02	4.06E+03	9.65E+02	+	4.10E+03	8.06E+02	+			=			=			=			=			-			-
14 –	DE/rand/1	1.26E+04	1.43E+03	2.62E+04	1.54E+03	+	1.76E+04	1.67E+03	+	3.53E+03	2.76E+03	–	1.08E+04	1.19E+03	=	2.20E+04	2.68E+03	=	1.97E+03	1.39E+03	–	1.39E+04	6.62E+02	+	3.67E+03	7.29E+02	–
	DE/best/1	1.48E+04	1.59E+03	3.07E+04	1.04E+03	+	2.25E+04	1.83E+03	+			–			=			=			–			–			–
15	DE/rand/1	3.18E+04	7.00E+02	2.73E+04	1.46E+03	–	1.99E+04	3.47E+03	–	1.59E+04	1.91E+03	=	2.28E+04	2.50E+03	=	2.38E+04	2.77E+03	=	1.70E+04	2.00E+03	–	2.95E+04	6.58E+02	–	1.99E+04	5.52E+03	–
	DE/best/1	3.23E+04	8.49E+02	2.93E+04	1.20E+03	–	2.46E+04	1.82E+03	–			=			=			=			–			–			–
16	DE/rand/1	3.57E+00	3.12E−01	3.87E+00	2.40E−01	=	3.90E+00	2.61E−01	=	3.86E+00	5.12E−01	=	3.84E+00	7.54E−01	=	4.76E+00	1.15E+00	=	3.96E+00	3.56E−01	+	4.40E+00	2.68E−01	+	4.32E+00	5.19E−01	+
	DE/best/1	3.79E+00	2.95E−01	2.87E+00	5.05E−01	–	3.30E+00	3.18E−01	=			=			=			=			+			=			+
17	DE/rand/1	4.97E+03	2.99E+03	6.43E+03	9.35E+02	+	5.16E+03	1.42E+03	+	3.31E+03	2.02E+03	=	3.08E+03	4.07E+02	=	4.33E+03	4.92E+02	=	2.09E+03	1.47E+03	–	4.94E+02	2.27E+01	–	3.02E+02	2.12E+01	–
	DE/best/1	5.52E+03	1.26E+03	6.11E+03	8.72E+02	=	6.81E+03	1.47E+03	=			=			=			=			–			–			–
18	DE/rand/1	5.06E+03	3.64E+03	5.91E+03	1.33E+03	+	5.87E+03	1.23E+03	+	5.10E+03	3.17E+03	=	4.20E+03	6.23E+02	=	5.32E+03	1.11E+03	=	3.75E+03	1.69E+03	–	1.17E+03	7.47E+01	–	8.81E+02	2.57E+02	–
	DE/best/1	5.93E+03	1.22E+03	5.91E+03	1.22E+03	=	6.20E+03	1.03E+03	=			=			=			=			–			–			–
19	DE/rand/1	3.98E+05	9.63E+07	2.17E+07	2.32E+07	+	5.74E+06	2.03E+07	+	2.10E+06	1.98E+06	+	7.64E+04	1.12E+05	–	2.47E+06	2.07E+06	+	8.57E+05	4.86E+05	+	8.93E+01	1.46E+01	–	5.20E+01	1.09E+01	–
	DE/best/1	1.15E+06	3.52E+07	1.64E+07	2.38E+07	+	2.61E+06	4.37E+07	+			=			–			=			–			–			–
20	DE/rand/1	5.00E+01	0.00E+00	5.00E+01	0.00E+00	=	5.00E+01	0.00E+00	=	5.00E+01	3.16E−11	=	5.00E+01	0.00E+00	=	5.00E+01	0.00E+00	=	5.00E+01	0.00E+00	=	5.00E+01	0.00E+00	=	5.00E+01	0.00E+00	=
	DE/best/1	5.00E+01	0.00E+00	5.00E+01	0.00E+00	=	5.00E+01	0.00E+00	=			=			=			=			=			=			=
21	DE/rand/1	2.85E+03	2.62E+03	1.24E+04	3.06E+03	+	6.28E+03	1.66E+03	+	8.35E+03	9.76E+02	+	1.95E+03	1.00E+03	=	6.28E+03	1.47E+03	+	7.27E+03	2.97E+02	+	4.83E+02	1.01E+02	–	4.37E+02	3.62E+01	–
	DE/best/1	2.97E+03	3.13E+03	1.27E+04	4.92E+03	+	9.86E+03	1.66E+03	+			+			=			+			+			–			–
22	DE/rand/1	1.20E+04	1.92E+03	2.84E+04	1.45E+03	+	2.04E+04	1.68E+03	+	4.71E+03	3.53E+03	–	1.32E+04	1.88E+03	=	2.59E+04	2.69E+03	=	2.56E+03	9.21E+02	–	1.70E+04	1.10E+03	+	4.06E+03	8.31E+02	–
	DE/best/1	1.48E+04	1.66E+03	3.24E+04	1.29E+03	+	2.41E+04	2.20E+03	+			–			=			=			–			+			–
23	DE/rand/1	3.23E+04	6.97E+02	2.89E+04	1.25E+03	–	2.13E+04	3.52E+03	–	2.37E+04	1.62E+03	=	2.81E+04	2.37E+03	=	3.07E+04	4.21E+03	=	2.29E+04	2.50E+03	–	3.15E+04	6.75E+02	=	2.21E+04	5.39E+03	–
	DE/best/1	2.52E+04	1.91E+03	3.24E+04	9.58E+02	=	2.58E+04	2.63E+03	+			=			=			=			–			+			–
24	DE/rand/1	5.99E+02	5.83E+00	6.20E+02	1.67E+03	+	5.70E+02	9.21E+02	–	1.68E+03	8.55E+02	+	8.15E+02	8.69E+01	=	7.60E+02	8.89E+01	=	8.46E+02	1.38E+02	+	4.28E+02	1.19E+01	–	3.90E+02	2.08E+01	–
	DE/best/1	5.98E+02	6.37E+00	9.28E+02	1.44E+03	+	5.85E+02	1.87E+03	–			+			=			=			+			–			–
25	DE/rand/1	5.98E+02	5.98E+00	5.80E+02	2.31E+01	–	6.00E+02	5.75E+00	=	9.42E+02	1.31E+02	+	8.65E+02	7.32E+01	+	8.15E+02	7.60E+01	+	8.79E+02	3.01E+01	+	5.78E+02	1.87E+01	=	4.90E+02	1.86E+01	–
	DE/best/1	5.99E+02	5.77E+00	6.18E+02	9.77E+01	+	5.88E+02	8.38E+00	=			+			+			+			+			=			–
26	DE/rand/1	7.04E+02	6.14E+00	6.85E+02	1.44E+03	–	3.02E+02	7.42E+02	–	6.96E+02	1.31E+01	=	7.01E+02	2.56E+01	=	6.88E+02	1.29E+01	=	7.02E+02	3.33E+01	=	5.25E+02	1.79E+01	–	5.20E+02	2.77E+01	–
	DE/best/1	6.42E+02	1.76E+01	7.24E+02	1.98E+03	+	6.93E+02	8.05E+02	+			=			=			=			+			–			–
27	DE/rand/1	4.31E+03	6.93E+01	4.31E+03	1.98E+03	+	4.05E+03	1.03E+02	–	5.76E+03	5.87E+02	=	4.94E+03	3.07E+02	=	4.85E+03	2.51E+02	=	5.18E+03	3.32E+02	+	2.72E+03	1.93E+02	–	2.54E+03	2.33E+02	–
	DE/best/1	4.29E+03	6.10E+01	5.00E+03	3.09E+03	+	4.21E+03	7.03E+02	=			=			=			=			+			–			–
28	DE/rand/1	2.23E+04	1.09E+04	2.78E+04	2.64E+04	=	2.99E+04	3.98E+04	+	2.08E+04	6.70E+03	=	2.14E+04	1.80E+03	=	2.49E+04	2.56E+03	=	2.02E+04	1.53E+03	–	7.10E+03	2.46E+03	–	3.73E+03	1.12E+03	–
	DE/best/1	2.71E+04	2.78E+04	2.66E+04	1.77E+04	=	2.98E+04	8.56E+04	=			=			=			=			–			–			–

References

[1] F. Pouladi, H. Salehinejad, H. Sanatnama, S. Talebi, Optimum communication infrastructure design for power grids synchronisation in smart grids, Int. J. Bio-Inspired Computat. 6 (2014) 262–274.
[2] H. Salehinejad, S. Talebi, Dynamic fuzzy logic-ant colony system-based route selection system, Appl. Computat. Intell. Soft Comput. (2010).
[3] H. Salehinejad, S. Talebi, PAPR reduction of OFDM signals by novel global harmony search in PTS scheme, Int. J. Digital Multim. Broadcast. (2012).
[4] F. Caraffini, F. Neri, I. Poikolainen, Micro-differential evolution with extra moves along the axes, in: Proc. IEEE Symposium on Differential Evolution, 2013, pp. 46–53.
[5] J. Lampinen, I. Zelinka, On stagnation of the differential evolution algorithm, in: Proc. of 6th International Mendel Conference on Soft Computing, 2000, pp. 76–83.
[6] M. Olguin-Carbajal, E. Alba, J. Arellano-verdejo, Micro-differential evolution with local search for high dimensional problems, in: Proc. IEEE Congress on Evolutionary Computation, 2013, pp. 48–54.
[7] F. Viveros-Jimnez1, E. Mezura-Montes, A. Gelbukh, Empirical analysis of a micro-evolutionary algorithm for numerical optimization, Int. J. Phys. Sci. 7 (8) (2012) 1235–1258.
[8] K. Krishnakumar, micro-genetic algorithms for stationary and non-stationary function optimization, Intell. Control Adapt. Syst. 1196 (1989) 289–296.
[9] T. Huang, A.S. Mohan, Micro-particle swarm optimizer for solving high dimensional optimization problems (PSO for high dimensional optimization problems), Appl. Math. Comput. 181 (2) (2006) 1148–1154.
[10] K.E. Parsopoulos, Parallel cooperative micro-particle swarm optimization: a master–slave model, Appl. Soft Comput. 12 (11) (2012) 3552–3579.
[11] J. Brest, M. Sepesy Mauec, Population size reduction for the differential evolution algorithm, Appl. Intell. 29 (3) (2007) 228–247.
[12] K.E. Parsopoulos, Cooperative micro-differential evolution for high-dimensional problems, in: Proc. 11th Annual Conference on Genetic and Evolutionary Computation, 2009.
[13] Choo Jun Tan, Chee Peng Lim, Yu-N. Cheah, A modified micro genetic algorithm for undertaking multi-objective optimization problems, J. Intell. Fuzzy Syst. 24 (3) (2013) 483–495.
[14] M.A. Sotelo-figueroa, H. Jos, P. Soberanes, J.M. Carpio, H.J.F. Huacuja, L.C. Reyes, J. Alberto, S. Alcaraz, Evolving bin packing heuristic using micro-differential evolution with indirect, Recent Adv. Hybrid Intell. Syst. (2013) 349–359.
[15] N.S. Teng, J. Teo, M.H.a. Hijazi, Self-adaptive population sizing for a tune-free differential evolution, Soft Comput. 13 (7) (2009) 709–724.
[16] S. Rahnamayan, H.R. Tizhoosh, Image thresholding using micro opposition-based Differential Evolution (Micro-ODE), in: Proc. IEEE Congress on Evolutionary Computation, 2008, pp. 1409–1416.
[17] Y. Gong, et al., Distributed evolutionary algorithms and their models: a survey of the state-of-the-art, Appl. Soft Comput. 34 (2015) 286–300.
[18] S. Rahnamayan, H.R. Tizhoosh, M.M.A. Salama, Opposition-based differential evolution, IEEE Trans. Evol. Comput. 12 (1) (2008) 64–79.
[19] A. Esmailzadeh, S. Rahnamayan, Enhanced differential evolution using center-based sampling, in: Proc. IEEE Congress on Evolutionary Computation, 2011, pp. 2641–2648.
[20] X. Zhang, S.Y. Yuen, Opposition-based adaptive differential evolution, in: Proc. IEEE Congress on Evolutionary Computation, 2012, pp. 1–8.
[21] J.-P. Chiou, C.-F. Chang, C.-T. Su, Ant direction hybrid differential evolution for solving large capacitor placement problems, IEEE Trans. Power Syst. 19 (4) (2004) 1794–1800.
[22] D.G. Kurup, M. Himdi, A. Rydberg, Synthesis of uniform amplitude unequally spaced antenna arrays using the differential evolution algorithm, IEEE Trans. Antennas Propag. 51 (9) (2003) 2210–2217.
[23] J.J. Liang, B.-Y. Qu, P.N. Suganthan, G. Alfredo, Hernndez-Daz, Problem definitions and evaluation criteria for the CEC 2013 special session and competition on real-parameter optimization," Technical Report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, 2013.
[24] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.P. Chen, A. Auger, S. Tiwari, Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization, Nanyang Tech. Univ., Singapore and KanGAL, Kanpur Genetic Algorithms Lab., IIT, Kanpur, India, Tech. Rep., No.2005005, May 2005.
[25] D. Lahoz, B. Lacruz, P.M. Mateo, A multi-objective micro genetic ELM algorithm, Neurocomputing 111 (2013) 90–103.
[26] F. Neri, V. Tirronen, Recent advances in differential evolution: a review and experimental analysis, Artif. Intell. Rev. 33 (1) (2010) 61–106.
[27] F. Neri, E. Mininno, Memetic compact differential evolution for Cartesian robot control, IEEE Computat. Intell. Mag. 5 (2) (2010) 54–65.
[28] S. Das, A. Konar, An improved differential evolution scheme for noisy optimization problems Pattern Recognition and Machine Intelligence, Lecture notes in computer science, vol. 3776, Springer, Berlin, 2005, pp. 417–421.
[29] S. Das, A. Konar, U. Chakraborty, Improved differential evolution algorithms for handling noisy optimization problems, in: Proc. IEEE Congress on Evolutionary Computation, vol. 2, 2005, pp. 1691–1698.
[30] K.V. Price, R. Storn, J. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer, Berlin, 2005.
[31] M. Weber, F. Neri, V. Tirronen, A study on scale factor in distributed differential evolution, Inf. Sci. 181 (12) (2011) 2488–2511.
[32] E. Mininno, F. Neri, F. Cupertino, D. Naso, Compact differential evolution, IEEE Trans. Evol. Computat. 15 (1) (2011) 32–54.
[33] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems, IEEE Trans. Evol. Comput. 10 (6) (2006) 646–657.
[34] J. Brest, A. Zamuda, B. Boskovic, V. Zumer, High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction, in: IEEE Congress on Evolutionary Computation, 2008, pp. 1–6.
[35] J. Brest, A. Zamuda, B. Boskovic, M.S. Maucec, V. Zumer, Dynamic optimization using self-adaptive differential evolution, IEEE Congress Evol. Computat. (2009) 415–422.
[36] C. Brown, Y. Jin, M. Leach, M. Hodgson, μ JADE: adaptive differential evolution with a small population, Soft Comput. (2015) 1–10.
[37] C. Segura, C.A.C. Coello, A.G. Hernandez-Diaz, Improving the vector generation strategy of differential evolution for large-scale optimization, Inf. Sci. 323 (2015) 106–129.
[38] I. Fajfar, T. Tuma, J. Puhan, J. Olensek, A. Burmen, Towards smaller populations in differential evolution, Electron. Comp. Mater. 42 (3) (2012) 152–163.
[39] C.A.C. Coello, G.T. Pulido, Multiobjective optimization using a micro-genetic algorithm, in: Proc. Genetic Evolutionary Computation Conference, 2001, pp. 274–282.
[40] P.C. Ribas, L. Yamamoto, H.L. Polli, L.V.R. Arruda, F. Neves-Jr, A micro-genetic algorithm for multi-objective scheduling of a real world pipeline network, Eng. Appl. Artif. Intell. 26 (1) (2013) 302–313.
[41] Y.G. Xu, G.R. Liu, Detection of flaws in composites from scattered elastic-wave field using an improved μGA and a local optimizer, Comput. Methods Appl. Mech. Eng. 191 (36) (2002) 3929–3946.
[42] D.E. Goldberg, Sizing populations for serial and parallel genetic algorithms, in: Proc. 3rd International Conference on Genetic Algorithms, 1989, pp. 70–79.
[43] J. Tippayachai, W. Ongsakul, I. Ngamroo, Parallel micro genetic algorithm for constrained economic dispatch, IEEE Trans. Power Syst. 17 (3) (Aug 2002) 790–797.
[44] S. Tiwari, G. Fadel, K. Deb, AMGA2: improving the performance of the archive-based micro-genetic algorithm for multi-objective optimization, Eng. Optim. 43 (4) (2011) 377–401.
[45] D. Snchez, P. Melin, O. Castillo, F. Valdez, Modular granular neural networks optimization with multi-objective hierarchical genetic algorithm for human recognition based on iris biometric, in: Proc. IEEE Congress on Evolutionary Computation, 2013, pp. 772–778.
[46] J.H. Ang, C.K. Goh, E.J. Teoh, A.A. Mamun, Multi-objective evolutionary Recurrent Neural Networks for system identification, in: Proc. IEEE Congress on Evolutionary Computation, 2007, pp. 1586–1592.
[47] K. Itoh, K. Miyata, H. Igarashi, Evolutional design of waveguide slot antenna with dielectric lenses, IEEE Trans. Magn. 48 (2) (2012) 779–782.
[48] F. Neri, G. Iacca, E. Mininno, Compact optimization, in: Handbook of Optimization, Springer, Berlin, Heidelberg, 2013, pp. 337–364.
[49] A. Prugel-Bennett, Benefits of a population: five mechanisms that advantage population-based algorithms, IEEE Trans. Evol. 14 (4) (2010) 500–517.
[50] H. Salehinejad, S. Rahnamayan, H.R. Tizhoosh, S.Y. Chen, Micro-differential evolution with vectorized random mutation factor, in: Proc. IEEE Congress on Evolutionary Computation, 2014, pp. 2055–2062.
[51] K.M. Bakwad, S.S. Pattnaik, B.S. Sohi, S. Devi, S.V.R.S. Gollapudi, C.V. Sagar, P.K. Patra, Fast motion estimation using small population-based modified parallel particle swarm optimisation, Int. J. Parallel, Emerg. Distrib. Syst. 26 (6) (2011) 457–476.
[52] H. Salehinejad, S. Rahnamayan, H.R. Tizhoosh, Exploration enhancement in ensemble micro-differential evolution, in: Proc. IEEE Congress on Evolutionary Computation, 2016.
[53] J.C.F. Cabrera, C.A.C. Coello, Handling constraints in particle swarm optimization using a small population size, in: MICAI 2007: Advances in Artificial Intelligence, 2007, pp. 41–51.
[54] J. Carlos, F. Cabrera, C.A.C. Coello, "Micro-MOPSO: A Multi-Objective Particle Swarm Optimizer that uses a very small population size, in: Multi-Objective Swarm Intelligent Systems, Springer, Berlin, Heidelberg, 2010, pp. 83–104.
[55] H. Salehinejad, S. Rahnamayan, H.R. Tizhoosh, Type-II opposition-based differential evolution, in: Proc. IEEE Congress on Evolutionary Computation, 2014, pp. 1768–1775.
[56] T.K. Das, G.K. Venayagamoorthy, Optimal design of power system stabilizers using a small population based PSO, in: Proc. IEEE Power Engineering Society General Meeting, 2006, pp. 1–7.
[57] H. Salehinejad, R. Zadeh, R. Liscano, S. Rahnamayan, 3D localization in large-scale Wireless Sensor Networks: a micro-differential evolution approach, in: Proc. IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC), 2014, pp. 1824–1828.
[58] T.K. Das, S.R. Jetti, G.K. Venayagamoorthy, Optimal design of SVC damping controllers with wide area measurements using small population based PSO, in: Proc. International Joint Conference on Neural Networks, 2006, pp. 2255–2260.
[59] T.K. Das, G.K. Venayagamoorthy, U.O. Aliyu, Bio-inspired algorithms for the design of multiple optimal power system stabilizers: SPPSO and BFA, IEEE Trans. Ind. Appl. 44 (5) (2008) 1445–1457.
[60] S. Rahnamayan, J. Jesuthasan, F. Bourennani, G.F. Naterer, H. Salehinejad, Centroid opposition-based differential evolution, Int. J. Appl. Metaheur. Comput. (IJAMC) 5 (4) (2014) 1–25.

[61] S. Rahnamayan, J. Jesuthasan, F. Bourennani, H. Salehinejad, G.F. Naterer, Computing opposition by involving entire population, in: Proc. IEEE Congress on Evolutionary Computation, China, 2014, pp. 1800–1807.
[62] P. Mitra, G. Venayagamoorthy, Empirical study of a hybrid algorithm based on clonal selection and small population based PSO, in: Proc. IEEE Swarm Intelligence Symposium, 2008, pp. 1–7.
[63] K.E. Parsopoulos, Cooperative micro-particle swarm optimization, in: Proc. First ACM/SIGEVO Summit on Genetic and Evolutionary Computation, 2009, pp. 467–474.
[64] H.A.N. Wen-hua, Improved MICROPSO algorithm and its application on reactive power optimization, in: Proc. Asia-Pacific Power and Energy Engineering Conference, 2012, pp. 1–4.
[65] D. Wu, D. Gan, J.N. Jiang, An improved micro-particle swarm optimization algorithm and its application in transient stability constrained optimal power flow, Int. Trans. Electr. Energy Syst. 24 (3) (2014) 395–411.
[66] J. Zhang, J. Wang, C. Yue, Small population-based particle swarm optimization for short-term hydrothermal scheduling, IEEE Trans. Power Syst. 27 (1) (2012) 142–152.
[67] T. Huang, A.S. Mohan, A novel micro-particle swarm optimizer for solving high dimensional optimization problems, in: Proc. IEEE Antennas and Propagation Society International Symposium, no. 1, 2006, pp. 3535–3538.
[68] C. Wang, Y. Liu, Y. Zhao, Application of dynamic neighborhood small population particle swarm optimization for reconfiguration of shipboard power system, Eng. Appl. Artif. Intell. 26 (4) (2013) 1255–1262.
[69] A. Rajasekhar, S. Das, Cooperative micro artificial bee colony algorithm for large scale global optimization problems, in: Swarm, Evolutionary, and Memetic Computing, Springer, Berlin, Heidelberg, 2013, pp. 469–480.
[70] A. Rajasekhar, S. Das, S. Das, μABC: a micro artificial bee colony algorithm for large scale global optimization, in: 14th Annual Conference Companion on Genetic and Evolutionary Computation, 2012, pp. 1399–1400.
[71] S. Dasgupta, A. Biswas, S. Das, B.K. Panigrahi, A. Abraham, A micro-bacterial foraging algorithm for high-dimensional optimization, in: Proc. IEEE Congress on Evolutionary Computation, 2009, pp. 785–792.
[72] Y.E. Yildiz, O. Altun, A.O. Topal, Computational chemotaxis in micro bacterial foraging optimization for high dimensional problems: a comparative study on numerical benchmark, Int. J. Comput. Appl. 124 (4) (2015).
[73] A.O. Topal, O. Altun, Y.E. Yildiz, Micro bat algorithm for high dimensional optimization problems, Int. J. Comput. Appl. 122 (12) (2015).
[74] N. Pandit, A. Tripathi, S. Tapaswi, M. Pandit, Static/dynamic environmental economic dispatch employing chaotic micro bacterial foraging algorithm, in: Swarm, Evolutionary, and Memetic Computing, Springer, Berlin, Heidelberg, 2011, pp. 585–592.
[75] J.C. Herrera-lozada, H. Calvo, H. Taud, A micro artificial immune system, Polibits 43 (2011) 107–111.
[76] F. Viveros-Jimanez, E. Mezura-Montes, A. Gelbukh, Elitistic evolution: a novel micro-population approach for global optimization problems, in: Proc. 8th Mexican International Conference on Artificial Intelligence, 2009, pp. 15–20.
[77] V.H. Hinojosa, R. Araya, Modeling a mixed-integer-binary small-population evolutionary particle swarm algorithm for solving the optimal power flow problem in electric power systems, Appl. Soft Comput. 13 (9) (2013) 3839–3852.
[78] S. Nesmachnow, H. Cancela, E. Alba, A parallel micro evolutionary algorithm for heterogeneous computing and grid scheduling, Appl. Soft Comput. 12 (2) (2012) 626–639.
[79] K.Y. Tsai, F.S. Wang, Evolutionary optimization with data collocation for reverse engineering of biological networks, Bioinformatics 21 (7) (2005) 1180–1188.
[80] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2011) 4–31.
[81] K. Price, R. Storn, J. Lampinen, Differential Evolution – A Practical Approach to Global Optimization, Springer, Berlin, Germany, 2005.
[82] F. Wilcoxon, Individual comparisons by ranking methods, Biomet. Bull. 1 (6) (1945) 80–83.
[86] SHARCNET (www.sharcnet.ca) is a consortium of colleges, universities and research institutes operating a network of high-performance computer clusters across south western, central and northern Ontario.
[87] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417.
[88] Y. Wang, Z. Cai, Q. Zhang, Differential evolution with composite trial vector generation strategies and control parameters, IEEE Trans. Evol. Comput. 15 (1) (2011) 55–66.
[89] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proc. IEEE International Conference on Neural Networks, 1995, pp. 1942–1948.
[90] D. Bratton, J. Kennedy, Defining a standard for particle swarm optimization, in: Proc. IEEE Swarm Intelligence Symposium, 2007, pp. 120–127.
[91] K. Jinno, T. Shindo, Analysis of dynamical characteristic of canonical deterministic PSO, in: Proc. IEEE World Congress on Computational Intelligence, 2010, pp. 1105–1110.