Mathematical Biosciences and Engineering

MBE, 19(6): 5968–5997.
DOI: 10.3934/mbe.2022279
Received: 19 December 2021
Revised: 22 March 2022
Accepted: 02 April 2022
Published: 11 April 2022

http://www.aimspress.com/journal/MBE

Research article

Self-organizing map based differential evolution with dynamic selection strategy for multimodal optimization problems

Shihao Yuan[1], Hong Zhao[1,2,*], Jing Liu[1] and Binjie Song[3]

[1] Xidian University, Guangzhou Institute of Technology, Guangzhou 510555, China
[2] Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China
[3] South China University of Technology, the School of Computer Science and Engineering, Guangzhou 510006, China

* **Correspondence:** Email: hongzhao@xidian.edu.cn.

Abstract: Many real-world problems can be classified as multimodal optimization problems (MMOPs), which require to locate global optima as more as possible and refine the accuracy of found optima as high as possible. When dealing with MMOPs, how to divide population and obtain effective niches is a key to balance population diversity and convergence during evolution. In this paper, a self-organizing map (SOM) based differential evolution with dynamic selection strategy (SOMDE-DS) is proposed to improve the performance of differential evolution (DE) in solving MMOPs. Firstly, a SOM based method is introduced as a niching technique to divide population reasonably by using the similarity information among different individuals. Secondly, a variable neighborhood search (VNS) strategy is proposed to locate more possible optimal regions by expanding the search space. Thirdly, a dynamic selection (DS) strategy is designed to balance exploration and exploitation of the population by taking advantages of both local search strategy and global search strategy. The proposed SOMDE-DS is compared with several widely used multimodal optimization algorithms on benchmark CEC'2013. The experimental results show that SOMDE-DS is superior or competitive with the compared algorithms.

Keywords: self-organizing map; differential evolution; multimodal optimization problem; niching; dynamic selection

1. Introduction

Multimodal optimization problems (MMOPs), which require to find all optimal solutions simultaneously, have been investigated in recent years [1]. In the real world, many engineering problems have more than one solution, such as structural damage detection [2], varied-line-spacing holographic grating design problem [3], protein structure prediction [4], and job shop scheduling problem [5]. Therefore, traditional algorithms which deal with single-solution optimization problems no longer meet the practical needs, and the new effective multi-solution optimization algorithms need to be designed to solve an increasing number of complex multi-solution problems. However, how to balance the diversity and convergence of population is still a challenge when dealing with MMOPs.

Evolutionary algorithms (EAs) have been an effective method for dealing with single-solution optimization problems for a long time, such as genetic algorithm (GA) [6], differential evolution (DE) [7–9], and particle swarm optimization (PSO) [10–13]. When EAs are used to solve single-solution optimization problems, all individuals within the population evolve towards the only one global optimum. While in MMOPs, there are multiple global optima to be found, and traditional EA methods cannot solve MMOPs effectively. Therefore, many scholars have adopted the improved EAs to solve MMOPs recently, such as the crowding clustering GA [14] that employs standard crowding strategy to eliminate genetic drift, the distance-based PSO [15] that eliminate the need to specify any niching parameter and the dual-strategy DE [16] that balance exploration and exploitation in generating offspring.

Although many efforts have been put into solving MMOPs, there are still some limitations when dealing with MMOPs. Firstly, how to divide the population to form effective niches is a challenge. Secondly, niches created by some techniques cannot cover all the possible regions of global optima, making it impossible to find out all optima. Thirdly, how to balance exploration and exploitation remains a challenge.

Therefore, this paper utilizes a self-organizing map (SOM) based niching method to deal with MMOPs. SOM has been a classic and useful tool in machine learning area for a long time [17,18], which can map high dimensional input data onto 2-dimensional plane while preserving the topology relations among input data. The potential of SOM in solving MMOPs is yet to be fully explored.

Consequently, we propose a SOM based DE with dynamic selection (DS) strategy (SOMDE-DS) to solve MMOPs more effectively. The framework of proposed SOMDE-DS and the differences between a standard DE and SOMDE-DS in solving MMOPs are shown in Figure 1. The advantages of our SOMDE-DS are listed as follows.

1) A SOM based niching method is proposed to divide the population reasonably by using similarity information among individuals. Specifically, the individuals with high similarities map to the same neuron and form a cohesive niche.

2) A variable neighborhood search (VNS) strategy is introduced to expand the search space. For some niches that are too small to find global optima, the VNS is carried out to expand the sizes and further to locate more global optima. In this way, small-sized niches are enriched and thus can find more optima that locate outside of the original niches.

3) A DS strategy based on the different evolution phases is proposed to balance the exploration and exploitation ability of population. Combining local selection and global selection strategy, DS strategy can explore more optima in the early evolution stage while maintain and refine found optima

in the later evolution stage.

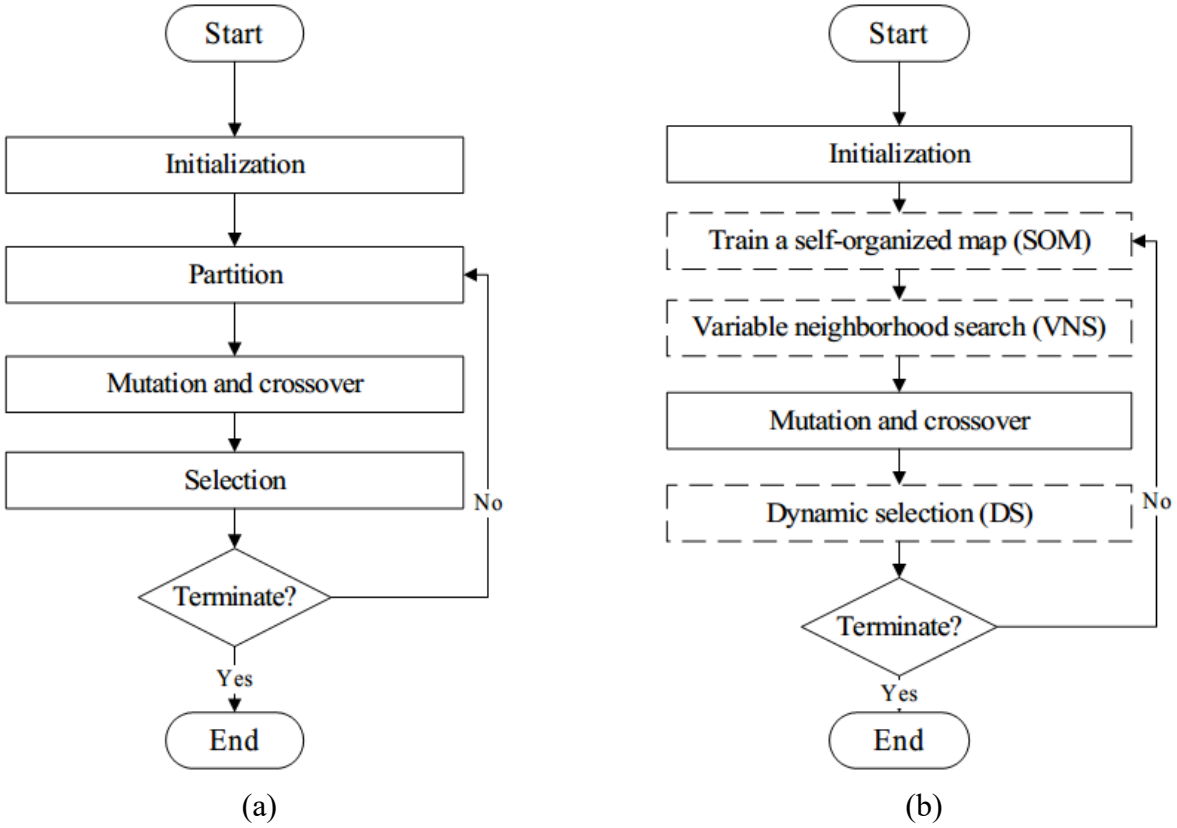


Figure 1. Differences between the framework of a traditional DE and SOMDE-DS in solving MMOPs. The three steps marked with dotted lines are improvements of SOMDE-DS on the foundation of traditional DE. (a) The framework of a common DE in solving MMOPs. (b) The framework of SOMDE-DS in solving MMOPs.

The rest of this paper is organized as follows. In Section 2, the process of DE and SOM is introduced as background knowledge for SOMDE-DS. Then SOMDE-DS is detailed in Section 3. Following are thorough experiments in Section 4 to verify the ability of SOMDE-DS to solve MMOPs effectively. Finally, the conclusions are given in Section 5.

2. Materials and methods

2.1. DE and SOM

2.1.1. DE

DE, which is Proposed by R. Storn and K. V. Price in 1995 [19], is a powerful tool for global optimization over continuous spaces. In recent years, DE has been an attractive optimization tool for lots of researchers and the reasons are obvious [7]. Comparing with other EAs, DE has the advantages of simplicity, better performance and fewer control parameters [20]. These are also the reasons why we choose DE to make improvements on solving MMOPs.

1) Simplicity. A traditional DE consists of four steps, which are initialization, mutation, crossover and selection. Comparing with other EAs (e.g., genetic programming), DE is more straightforward and easier to implement, which makes it easier for researchers in other fields to make good use of it.

2) Better performance. At the first international contest on evolutionary optimization held in Nagoya, Japan in 1996, DE ranks third among all optimization algorithms and first in all EAs [21]. In the following CEC competitions, DE still ranks high among all EAs. From CEC 2014 to CEC 2016, DE variants take the first place in a continuous three years [22–24]. In CEC 2017 and CEC 2018, the best DE variants still hold the third and the second place [25,26].

3) Fewer control parameters. In classic DE, there are only three parameters, that is, probability of crossover, scaling factor and population size. The way these parameters work and contribute to the result has been deeply studied in recent research [27]. It is easy for us to fine-tune these parameters to get better performance.

Algorithm 1: DE

Input: population size NP, probability of crossover pc, scaling factor F, and maximum number of generation G

```
Randomly initialize first generation of population;
For g = 1 to G
    For i = 1 to NP
        Perform the mutation operation according to Eq (1);
        Generate trial vector according to Eq (2);
        If f(u_{ij,G+1}) ≤ f(x_{i,G})
            x_{i,G+1} = x_{i,G};
        Else
            x_{i,G+1} = u_{ij,G+1};
        End If
    End For
End For
End
```

A classical DE algorithm can be divided into four steps: initialization, mutation, crossover, and selection. The framework of the original DE is shown in Algorithm 1.

1) Initialization. Randomly initialize individual $\boldsymbol{x}_{i,G}$, where $i = 1, \ldots, NP$. NP represents the size of population and G means the number of current generations.

2) Mutation. Mutated individual $\boldsymbol{v}_i$ is generated by

$$\boldsymbol{v}_{i,G+1} = \boldsymbol{x}_{r1,G} + F \cdot (\boldsymbol{x}_{r2,G} - \boldsymbol{x}_{r3,G}), \tag{1}$$

where F is scaling factor and three different individuals chosen to mutate are represented by $\boldsymbol{x}_{r1,G}$, $\boldsymbol{x}_{r2,G}$ and $\boldsymbol{x}_{r3,G}$.

3) Crossover. Trial vector $\boldsymbol{u}_{ij,G+1}$ is generated according to the following formula

$$\boldsymbol{u}_{ij,G+1} = \begin{cases} \boldsymbol{v}_{ij,G+1} \ if \ rand_j < pc \ or \ j = k \\ \boldsymbol{x}_{ij,G} \ otherwise \end{cases}, \tag{2}$$

where *pc* is probability of crossover.

4) Selection. To select a better individual, $\boldsymbol{u}_{ij,G+1}$ is compared with $\boldsymbol{x}_{i,G}$. If $f(\boldsymbol{u}_{ij,G+1})$ *is* better than $f(\boldsymbol{x}_{i,G})$, $\boldsymbol{u}_{ij,G+1}$ is inherited to next generation; otherwise $\boldsymbol{x}_{i,G}$ is retained.

2.1.2. SOM

Algorithm 2: SOM framework

Input: initial neighborhood radius σ_0, initial learning rate τ_0, maximum number of generation *G*, input data $\boldsymbol{X}$, the dimension of input data *D* and neighboring function $\boldsymbol{z}$

1 Randomly initialize each neuron weight vectors $\boldsymbol{w}$;
2 **For** generation $g = 1$ to G
3 Adjust neighborhood radius σ:

$$\sigma = \sigma_0 \times (1 - \frac{g}{G})$$

4 Adjust learning rate τ:

$$\tau = \tau_0 \times (1 - \frac{g}{G})$$

5 Randomly select a training point $\boldsymbol{x} \in \boldsymbol{X}$;
6 Find the winner neuron e' in all neurons e:

$$e' = \underset{1 \le e \le D}{\operatorname{argmin}} \left\| \boldsymbol{x} - \boldsymbol{w}^e \right\|_2;$$

7 Locate the set of all neighboring neurons $\boldsymbol{E}$:

$$\boldsymbol{E} = \{e \mid 1 \le e \le D \wedge \left\| \boldsymbol{z}^e - \boldsymbol{z}^{e'} \right\|_2 < \sigma\};$$

8 Update all neighboring neurons as

$$\boldsymbol{w}^u = \boldsymbol{w}^u + \tau \cdot \exp(-\left\| \boldsymbol{z}^e - \boldsymbol{z}^{e'} \right\|_2 (\boldsymbol{x} - \boldsymbol{w}^e);$$

9 **End For**
10 **End**

Output: a trained SOM model

SOM, introduced by T. Kohonen in 1982, is a data-analysis method that visualizes similarity relations in a set of data items [17,18]. As a nonlinear projection tool, SOM can map data vectors with high dimensionality onto a 2-dimensional plane, preserving the topological relationship of the original vectors. In a trained SOM model, vectors with high similarity will be mapped to the same neuron [28]. In this paper, we take the advantages of SOM and use it as a clustering technique. Comparing with other unsupervised data-analysis methods like K-means clustering [29], the trained models of SOM are capable of capturing topologic relations that are the same as the source data and therefore clusters formed by SOM are more cohesive. The working mechanism of SOM is detailed in Algorithm 2 and the layout of SOM is illustrated in Figure 2. There are two layers in a standard SOM model, that are the competition layer and the input layer. Input vectors from input layer will be

mapped to the neurons in the competition layer, according to the similarity between input vectors and the neurons, which can be measured in the form of some kind of distance (i.e., Euclidean distance). Therefore, input vectors with high similarities will be mapped to the same neuron and cohesive clusters are formed.

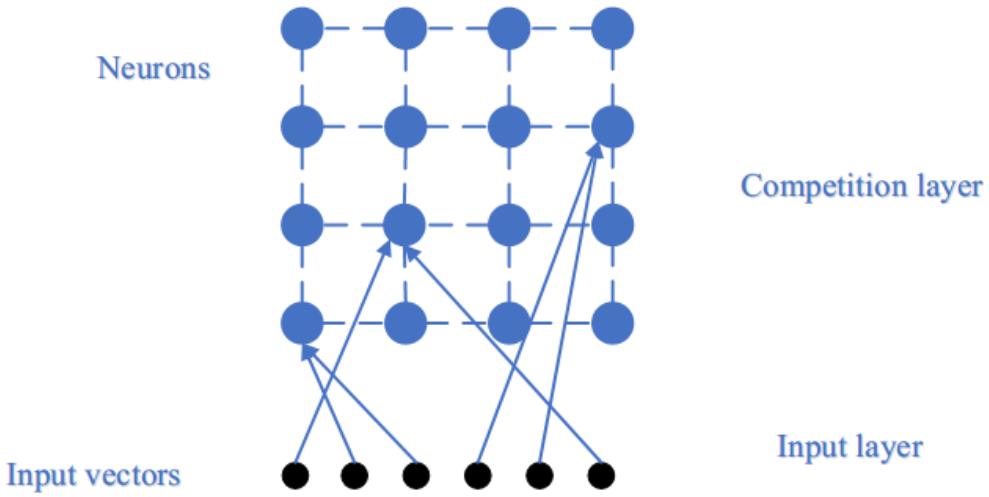


Figure 2. Layout of SOM. Input vectors with high similarity are mapped to the same neuron.

2.1.3. SOM in solving continuous optimization problems

Works have been done with SOM to solve continuous optimization problems. On the basis of PSO with elite learning strategy, Jing et al. combined it with SOM to tackle multimodal multi-objective problems. In reference [30], Qu et al. firstly employed speciation to solve multimodal multi-objective problems, where a self-organized mechanism is proposed to improve the performance of the speciation. Hu et al. [31] adopted SOM to build a good neighborhood relation for the improved pigeon-inspired optimization in order to better solve multimodal multi-objective optimization problems. Zhang et al. [32] used SOM to establish the neighborhood relationship among current solutions to control the generation of a new solution. A. Kashtiban and S. Khanmohammadi [33] use SOM based method to detect the number of niches, within which a simple GA is independently converging to the actual optima. In conclusion, SOM plays an important role in forming the neighborhood relationship. In this paper, SOM is used as the same way.

2.2. Related works on MMOP

In opposed to single-solution optimization problems [34], which have only one global optimum, MMOPs have multiple optimal solutions. In recent years, EAs have attracted a lot of attention in the field of solving MMOPs. However, in the beginning, EAs are designed for solving optimization problem with only one global optimum. Hence various modifications have been made to traditional EAs to enable them to handle MMOPs effectively. In general, there are three main methods that enable EAs to better solve MMOPs, including the niching-based methods, the novel evolution operator-based methods and the multiobjectivization-based methods.

1) Niching-based methods

The idea of niching technique is to discover and maintain multiple subpopulations at the same time to ensure population diversity. When population diversity is ensured, finding and maintaining multiple solutions at the same time is possible.

Lots of work have been done in incorporating EAs with niching techniques to better solve MMOPs. Yang et al. [35] took the difference among niches into consideration and develop an adaptive multimodal continuous ant colony optimization algorithm. A. Hackl et al. [36] used only clusters of fireflies which gather around promising local solutions to find better solutions. Two of the most well-known DE variants that incorporated niching technique are crowding DE (CDE) [37] and species-based DE (SDE) [38]. Combining the idea of neighborhood, Qu et al. [39] proposed neighborhood based CDE (NCDE) and neighborhood based SDE (NSDE). Li et al. [40] combined PSO and ring topology to propose r2PSO and r3PSO, which achieved the target of niching without niching parameters. Wei et al. [41] proposed a penalty-based DE, in which the neighboring solutions of elite solutions are penalized. Lin et al. [42] divide the population into multiple species by nearest-better clustering. Based on the affinity propagation clustering, Wang et al. [43] develop an automatic niching DE with contour prediction. Xu et al. [44] use detect-multimodal method to estimate the radius and use the estimated radius to divide the population into species. Liang et al. [45] combined a clustering-based special crowding distance method and a distance-based elite selection mechanism to enable DE to better solve multimodal multiobjective optimization problems. On the basis of affinity propagation clustering, Hu et al. [46] added a novel mutation and adaptive local search strategy to propose a niching backtracking search algorithm for solving multimodal multiobjective optimization problems.

In conclusion, the core of niching-based method is niching technique. Lots of works have been done in integrating EAs with a novel niching technique, and then novel local search strategies are added to improve the performance in solving MMOPs.

2) Novel evolution operator-based methods

The novel evolution operator-based methods usually modify evolution operators to make EAs better solve MMOPs. Qu et al. [15] proposed a local search operator to enhance the search ability and convergence of PSO. Epitropakis et al. [47] develop two new mutation strategies that incorporated spatial information of possible solution without introducing any extra parameter. Besides, a novel reinitialization mechanism was also proposed by Epitropakis et al. [48] to investigate unexplored regions of search space while maintain the best found solutions. Wang [49] employed adaptive parameter control and example-based learning to enhance the performance of PSO in solving MMOPs. Liu et al. [50] use the dynamic regulation to improve local search ability effectively.

To sum up, novel evolution operator-based methods focus on the modification on the evolutionary operations. Most of existing works are done in initialization, mutation and crossover to enhance the search ability of EAs in order to locate more optima at the same time.

3) Multiobjectivization-based methods

The idea of the multiobjectivization-based methods is to transform the MMOPs into the multiobjective optimization problems (MOPs). Then multi-objective optimization evolutionary algorithms (MOEAs) can be used to solve the transformed problem and multiple global optima of the original MMOPs can be obtained. Cheng et al. [51] transformed MMOPs into MOPs to approximate the fitness landscape of the original problem, and then a peak detection method was used to obtain

precise locations of global optima. Wang et al. [52] transformed MMOPs into MOPs with two conflicting objectives. Therefore, an MOEA is used to find the Pareto set of the MOP, and multiple optimal solutions of the MMOP can be located at the same time. Yu et al. [53] transformed MMOPs into MOPs with triple objectives, two of which conflict with each other and the extra objective can be used to improve the diversity of the population greatly. In reference [54], V. Steinhoff et al. proposed a single-objective multi-objective gradient sliding algorithm (SOMOGSA) to solve single-objective optimization problems using multi-objective approach. In reference [55], P. Aspar et al. examined the inner mechanisms of SOMOGSA to prove that single-objective optimization problems can be solved effectively by using multiobjectivization and the potential of multiobjectivization is huge. In reference [56], C. Grimme et al. studied the challenges and researched the potentials of solving continuous multimodal multi-objective optimization.

Comparing with niching-based methods and novel evolution operator-based method, multiobjectivation-based methods are relatively new. The main focus of these methods is constructing extra objectives to make good use of the advantages of multiobjective optimization. Some works have been done in the mechanism of multiobjectivization, but the inner mechanism and the potential of multiobjectivizaiton are still to be fully discovered.

2.3. SOMDE-DS

In this section, the main framework of SOMDE-DS is firstly given. Secondly, the SOM-based niching strategy is introduced. Then, the VNS strategy and DS strategy are detailed, respectively.

2.3.1. Main framework

The framework of SOMDE-DS is given in Algorithm 3. Firstly, randomly initialize the population with size NP and set the number of current generations *fe* to 0. Secondly, we use the whole population of current generation to train a SOM and use it to divide the population into several niches. Then the VNS strategy is applied to enrich the small-sized niches and niches with overlap are obtained. Following is mutation and crossover operation. Finally, the DS strategy is used to update current population.

Algorithm 3: Main framework

Input: population size *NP* and maximum number of function evaluations *MaxFEs*

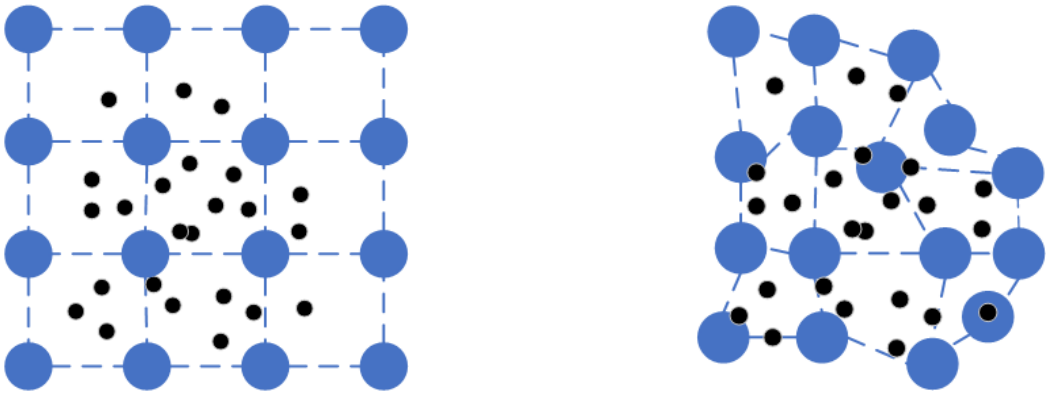
Randomly initialize the population with size *NP* and set $fe = 0$
While $fe \leq MaxFEs$
    Train a SOM with current population; (Algorithm 2)
    Perform VNS strategy on the population to get niches with overlap; (Algorithm 4)
    For each individual $\boldsymbol{x}_i$
        Randomly select three different individuals within its niche to perform mutation and crossover operation and produce offspring $\boldsymbol{v}_i$;
        Use DS strategy to select individual $\boldsymbol{q}_i$; (Algorithm 5)
        If $f(\boldsymbol{q}_i) \leq f(\boldsymbol{v}_i)$
            $\boldsymbol{y}_i = \boldsymbol{v}_i$;

```
            Else
                yi = qi;        // yi is the next generation individual of xi
            End If
        End For
        For offspring yi
            xi = yi;
        End For
        fe = NP + fe;
    End While
End
```

Output: multiple solutions for objective functions

2.3.2. SOM for niching

Due to the fact that SOM can capture the original topologic information of the source data, SOM is used for niching in SOMDE-DS. The working mechanism of SOM is detailed in Algorithm 2. In the first step, we randomly initialize the SOM with small weight vectors to minimize the influence of initial weight vectors. Then we train a SOM with all individuals in current population. Every time an individual is put into the SOM, the corresponding winning weight vector and its neighboring weight vectors are calculated. Then all neighboring weight vectors, as well as the winning weight vector itself are updated. Figure 3 shows that after training, the model of SOM matched the distribution of the current population, and therefore a better clustering result can be obtained.



(a) (b)

Figure 3. (a) SOM before training and (b) SOM after training.

2.3.3. VNS

In the original DE, mutation operation is carried out within the whole population. Offspring generated in this manner may differ greatly from its parent. In solving single-solution optimization problems, this mutation strategy can fully explore the search space. However, this global mutation strategy cannot achieve satisfactory results when dealing with MMOPs since there are multiple

optimal regions in the whole search space. By using this kind of mutation strategy, it is easily to cause the whole population to converge to only one optimal solution, which contradicts the goal of MMOPs. Thus, niching technique is integrated with DE to make it better solve MMOPs.

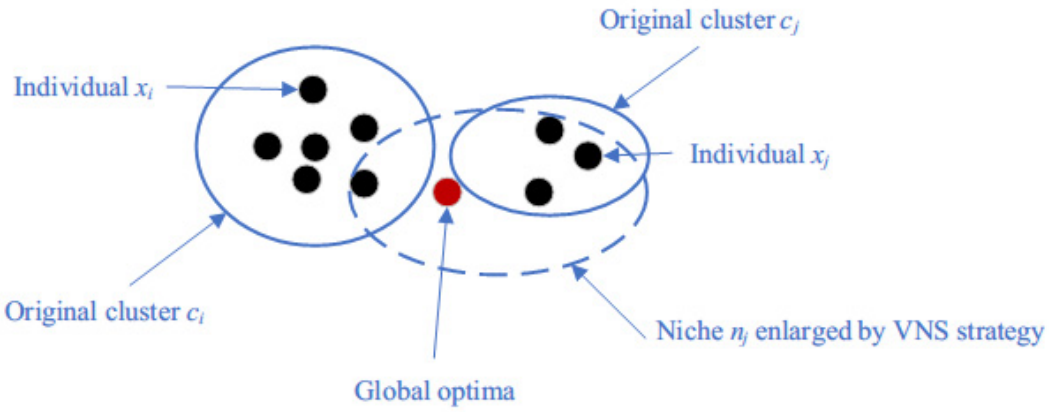


Figure 4. Niches enlarged by VNS strategy can locate more optimal regions. All individual in the same original niche map to the same neuron in SOM.

By using the trained SOM, the whole population can be divided into several niches. However, there is a situation where the number of individuals of a specific niche is too small to find global optima. To improve this situation, we designed a VNS strategy to locate more optimal regions. Figure 4 shows that with VNS strategy, small niches are enlarged and thus are enabled to locate more optimal regions. The process for enlarging niches is described as follows. Firstly, all individuals which map to the same neuron form a niche. Secondly, if the number of individuals within the niche is below a certain number M, individuals from nearest neighboring niche are merged into the niche one by one until the size of the niche reaches M. M represents the minimal number of individuals within a single niche. When the number is too small, it may cause the loss of diversity of population, while when the number is too big, it may cost more computational resources without improving the results obviously. This process of VNS is detailed in Algorithm 4

Algorithm 4: VNS

Input: a trained SOM model and current population

For each individual $\boldsymbol{x}_i$
    $\boldsymbol{x}_i$ is put into the trained SOM and its corresponding winning neuron $\boldsymbol{w}_i$ is calculated;
End for
For each neighboring neuron $\boldsymbol{e}_i$
    All individuals whose winning neuron is $\boldsymbol{e}_i$ are grouped as a cluster $\boldsymbol{c}_i$;
End for
For each cluster $\boldsymbol{c}_i$
    For each cluster $\boldsymbol{c}_i$ ($j \neq i$)
        The distance between $\boldsymbol{c}_i$ and $\boldsymbol{c}_i$ is calculated as the Euclidean distance from $\boldsymbol{e}_i$ to $\boldsymbol{e}_j$ on the SOM;
    End for
    While the number of individuals with $\boldsymbol{c}_i$ is smaller than M;

12 Add an individual from the nearest cluster to form niche $\boldsymbol{n}_i$
13 **End While**
14 **End for**
15 **End**

Output: niches for DE operations

2.3.4. DS

In the original DE, the produced offspring only competes with its parent. In this way, a better individual may be reserved but the opportunity to simultaneously eliminate a worse individual is lost. To improve this situation, a crowding based selection strategy is introduced in CDE [37], where every produced offspring competes with its nearest individual in the population. The crowding operation can be carried out within a single niche or the whole population. The former is called local selection while the latter is global selection.

Local selection strategy with niching technique benefits for exploring new optima. In comparison, global selection strategy is good at maintain and refine found optima. To make good use of these two strategies, we combine local selection and global selection to get a DS strategy, adapting to the different stages of evolution. The idea of this strategy is to use local selection to explore new optima in the early evolution stage and to use global selection to maintain and refine found optima in the later stages. In practice, we set two thresholds, one is of maximum number of function evaluations fe_t and the other is of probability p_l. When the threshold of maximum number of function evaluations fe_t is exceeded, only global selection strategy is used to maintain and refine found optima.

When the number of current function evaluations is below the threshold, a random number p_i between [0, 1] is generated to balance diversity and convergence. If p_i is below the threshold of probability p_l, local selection strategy is used to improve the convergence of population, otherwise, global selection strategy is used to maintain and refine best individuals.

Both fe_t and p_l work together to balance diversity and convergence of the population. The best setting of these two parameters for each function differs from each other. When fe_t and p_l are too big, the algorithm cannot get a stable result. When they are too small, the algorithm may not converge well. With our preliminary tests, SOMDE-DS performs well when fe_t is 0.9 and p_l is 0.6. The DS strategy is detailed in Algorithm 5.

Algorithm 5: DS

Input: current population and corresponding offspring
1 **For** each individual $\boldsymbol{x}_i$
2 p_i is randomly generated in [0, 1];
3 **If** $p_i > p_l$ and $fe < fe_t$
4 Select the nearest individual $\boldsymbol{q}_i$ of $\boldsymbol{x}_i$ from its niche;
5 **Else**
6 Select the nearest individual $\boldsymbol{q}_i$ of $\boldsymbol{x}_i$ from the whole population;
7 **End if**
8 **End for**
9 **End**

Output: new population

3. Results

This section is the experimental part, which mainly focuses on three contents. Firstly, SOMDE-DS and classic MMOP optimization algorithms are compared to verify its feasibility of solving MMOPs effectively. Secondly, we compared the situation of different mutation strategies and selection strategies in SOMDE-DS. Finally, the influence of different parameters on SOM is studied.

3.1. Benchmark functions and compared algorithms

CEC'2013 [57], a widely used benchmark function set for IEEE CEC'2013 special session and competition on niching methods for multimodal function optimization. This function set consists of 20 multimodal test functions, dimensions of which range from 1 to 20. With widely used simple functions and complex composite functions included, CEC'2013 is regarded as a suitable benchmark for evaluation in solving MMOPs.

To examine the ability of SOMDE-DS to solve MMOPs, we choose six well-known EAs with niching techniques to compete with SOMDE-DS, that are NCDE, NSDE, CDE, SDE, r2PSO and r3PSO. Among these six compared algorithms. CDE and SDE are DEs with two most well-known niching techniques. NCDE and NSDE is the development of classic CDE and SDE with the concept of neighborhood mutation. r2PSO and r3PSO are two well-known PSO algorithms with no extra niching parameters using ring topology. The performance of NCDE, NSDE, r2PSO and r3PSO in solving MMOPs are outstanding. Thus, competition with other six well-known multimodal algorithms should prove the superiority of SOM in niching and the ability of SOMDE-DS to better solve MMOPs.

3.2. Performance metrics

To measure the performance of SOMDE-DS and other multimodal optimization algorithms, two commonly used criteria peak ratio and success rate [58] are calculated by the results of 40 independent runs on each function.

3.1.1. Peak ratio

Peak ratio (*PR*) is the ratio of average peaks detected out of all peaks in the given function by an algorithm. It can be calculated as Eq (3):

$$PR = \frac{\sum peaks\_found_i}{no\_peak \times N}, \tag{3}$$

where N is the number of runs for each test function, $peaks\_found_i$ is the number of global optima found in the i^{th} run and *no_peak* is the number of global optima of the current test function.

3.1.2. Success rate

Success rate (*SR*) is the rate of successfully detecting all desired optima out of 11 runs for each

function. It can be calculated as Eq (4):

$$SR = \frac{\sum count_i}{N}, \tag{4}$$

where $count_i$ denotes whether all global optima are found in this run or not. When all global optima are found, $count_i$ is 1 otherwise 0.

PR measures the ability to find the global optima of an algorithm while *SR* represents the ability to find all global optima in a single run. To further illustrate the differences among tested algorithms, convergence is also telegraphed in several selected functions.

3.1.3. Parameter settings

The parameter settings of SOMDE-DS are as follows, probability of crossover *pc* as 0.5, scaling factor as 0.9, minimum size of niche *M* as 10, the threshold of number of function evaluations fe_t as $0.9 \times MaxFES$, the threshold of probability p_l as 0.6. The influences of these parameters are discussed in the experiment part.

The parameter settings for other tested EAs are as follows. In CDE, scaling factor is 0.5, probability of crossover is 0.9 and crowding factor is 100. In SDE, the species radius is set to 0.5 for each benchmark function. The neighborhood size of NCDE and NSDE is set to 20. In r2PSO and r3PSO, φ is 4.1 and χ is 0.7298. The settings for CEC'2013 benchmark can be referred in [57]. All tested algorithms run 40 times independently for each function on the benchmark CEC'2013.

3.2. The compared results between SOMDE-DS and other tested multimodal optimization algorithms

Table 1. The compared results between SOMDE-DS and other tested multimodal optimization algorithms.

F	SOMDE		NCDE		CDE		r2PSO		r3PSO		SDE		NSDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F2	**1.000**	1.000	0.940 (+)	0.825	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	0.780 (+)	0.725	0.840 (+)	0.800
F3	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F4	**1.000**	1.000	**1.000 (≈)**	1.000	0.469 (+)	0.000	0.875 (+)	0.700	0.925 (≈)	0.775	0.100 (+)	0.000	0.187 (+)	0.000
F5	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	0.450 (+)	0.400	0.500 (+)	0.000
F6	**0.993**	0.900	0.936 (≈)	0.500	0.909 (≈)	0.550	0.033 (+)	0.000	0.050 (+)	0.000	0.014 (+)	0.000	0.028 (+)	0.000
F7	0.800	0.000	0.830 (≈)	0.000	**0.882 (≈)**	0.000	0.360 (+)	0.000	0.250 (+)	0.000	0.019 (+)	0.000	0.028 (+)	0.000

Continued on next page

F	SOMDE		NCDE		CDE		r2PSO		r3PSO		SDE		NSDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F8	**0.960**	0.125	0.645 (+)	0.000	0.003 (+)	0.000	0.001 (+)	0.000	0.004 (+)	0.000	0.008 (+)	0.000	0.002 (+)	0.000
F9	0.326	0.000	0.375 (≈)	0.000	**0.483 (-)**	0.000	0.060 (+)	0.000	0.058 (+)	0.000	0.003 (+)	0.000	0.003 (+)	0.000
F10	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	0.687 (+)	0.100	0.646 (+)	0.000	0.058 (+)	0.000	0.033 (+)	0.000
F11	**0.792**	0.000	0.636 (≈)	0.000	0.021 (+)	0.000	0.636 (≈)	0.000	0.617 (≈)	0.000	0.110 (+)	0.000	0.083 (+)	0.000
F12	**0.625**	0.000	0.125 (+)	0.000	0.008 (+)	0.000	0.265 (+)	0.000	0.281 (+)	0.000	0.056 (+)	0.000	0.125 (+)	0.000
F13	**0.667**	0.000	0.500 (≈)	0.000	0.015 (+)	0.000	0.500 (≈)	0.000	0.646 (≈)	0.000	0.083 (+)	0.000	0.125 (+)	0.000
F14	**0.667**	0.000	**0.667 (≈)**	0.000	0.000 (+)	0.000	0.233 (+)	0.000	0.542 (≈)	0.000	0.037 (+)	0.000	0.108 (+)	0.000
F15	**0.375**	0.000	0.269 (≈)	0.000	0.000 (+)	0.000	0.031 (+)	0.000	0.219 (≈)	0.000	0.068 (+)	0.000	0.069 (+)	0.000
F16	**0.667**	0.000	**0.667** (≈)	0.000	0.000 (+)	0.000	0.000 (+)	0.000	0.269 (+)	0.000	0.025 (+)	0.000	0.033 (+)	0.000
F17	**0.250**	0.000	**0.250 (≈)**	0.000	0.000 (+)	0.000	0.000 (+)	0.000	0.150 (+)	0.000	0.000 (+)	0.000	0.009 (+)	0.000
F18	**0.375**	0.000	0.333 (≈)	0.000	0.000 (+)	0.000	0.000 (+)	0.000	0.030 (+)	0.000	0.000 (+)	0.000	0.004 (+)	0.000
F19	0.128	0.000	**0.165 (≈)**	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
F20	**0.056**	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
≈ (SOMDE-DS is similar)			17		8		7		10		3		3	
+ (SOMDE-DS is better)			3		11		13		10		17		17	
− (SOMDE-DS is worse)			0		1		0		0		0		0	

Table 1 shows the results of SOMDE-DS and other widely used multimodal algorithms in solving MMOPs, where the best PR value of each function are emphasized in boldface. Besides, Mann-Whitney U test is carried out between SOMDE-DS and every other EA with a confidence interval of 95%. There are three different results in the test represented by three different symbols, that is "≈", "+" and "–" respectively. "≈" means that the performance of SOMDE-DS on this function is similar to corresponding algorithm. "+" means that SOMDE-DS performs significantly better than corresponding algorithm, while "–" means that SOMDE-DS performs significantly to corresponding algorithm. From Table 1, it is clear that SOMDE-DS achieves the best results on most functions except for F7, F9 and F19. Among all multimodal algorithms, SOMDE-DS gets first place for 17 times in total, which is the best among all tested algorithms. Besides that, the results of the Mann-Whitney U test also show that SOMDE-DS performs better than every other algorithm in the manner of statistics.

It can be clearly seen from Figure 5 that the SOMDE-DS performs better than other methods in comparison in selected four functions (i.e., F2, F9, F11 and F18). Specifically, SOMDE-DS converges faster and finds more optima in any situation.

In addition, to better illustrate the convergence of SOMDE-DS, the fitness landscape and distribution of all individuals under three different situations are listed. Specifically, four different functions (i.e., F3, F7, F10 and F11) are shown in Figures 6–9 respectively and each of them includes the results of initialization, the results after several generations of evolution and the final results. F3 is a simple 2D function with only 1 global optimum and 4 local optima. In addition to locating the only 1 global optimum, SOMDE-DS also locates the 4 local optima, which proves its superiority in niching. Both F7 and F10 are widely used benchmark functions with multiple global optima. The number of global optima of F7 is 18, and that of F10 is 12. From Figures 7 and 8, it can be observed that in the early stage of evolutions, niches are formed around every global optimum. At the end of evolutions, every global optimum of both two functions is located. F11 is a complex composite function with 6 global optima. It can be seen from Figure 9 that, SOMDE-DS is still able to most optima of F11.

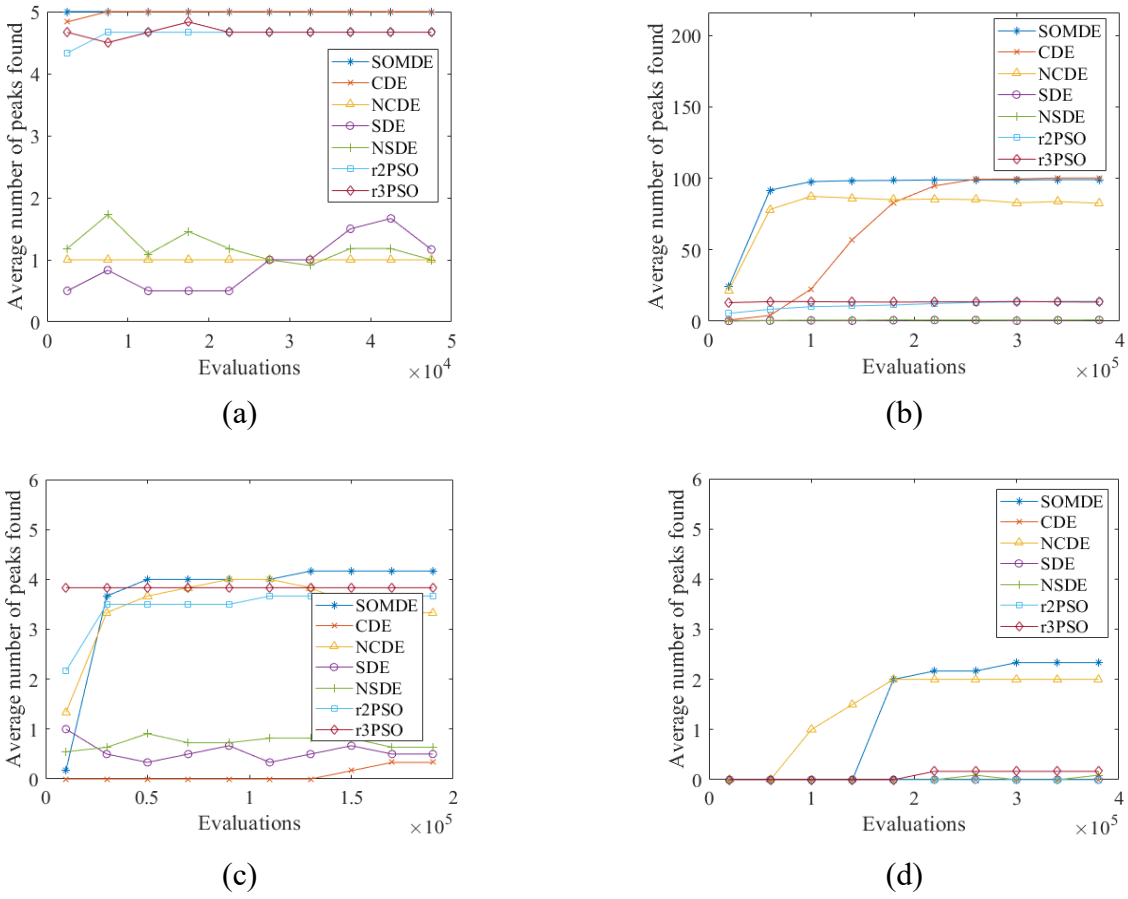


Figure 5. Average number of peaks found by tested algorithms on (a) F2, (b) F9, (c) F11, and (d) F18.

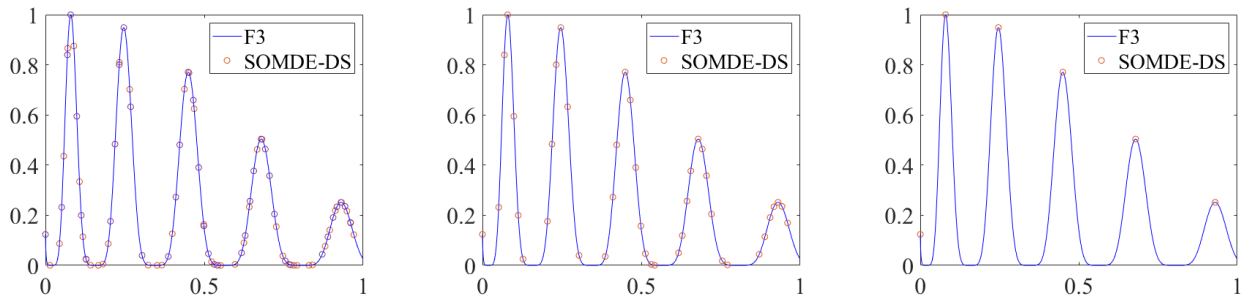


Figure 6. Fitness landscape and population distributions of F3 after (a) initialization, (b) 20 iterations and (c) final iterations.

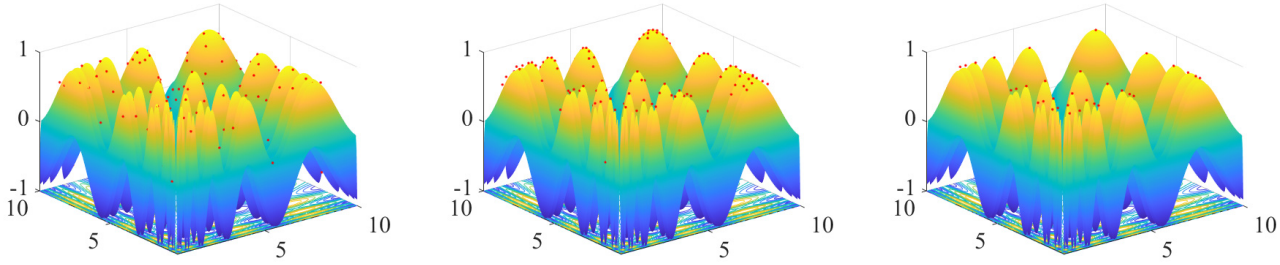


Figure 7. Fitness landscape and population distributions of F3 after (a) initialization, (b) 30 iterations and (c) final iterations.

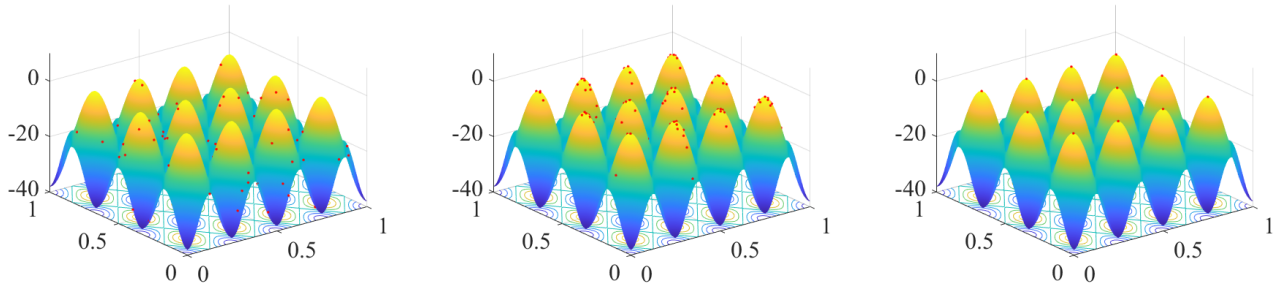


Figure 8. Fitness landscape and population distributions of F3 after (a) initialization, (b) 30 iterations and (c) final iterations.

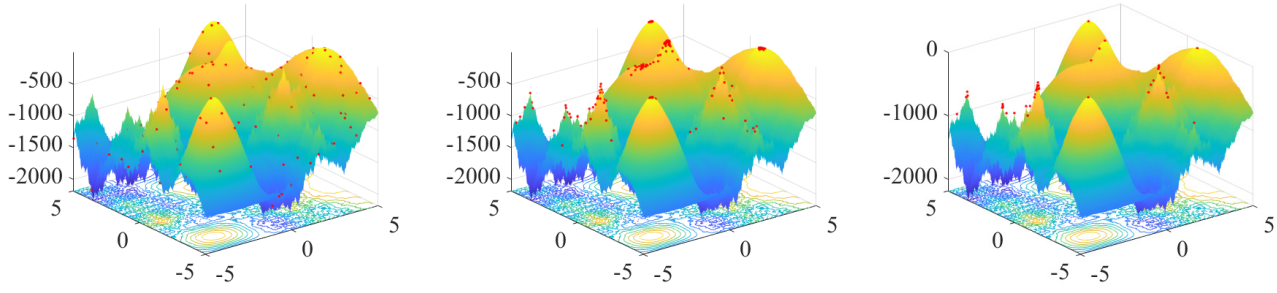


Figure 9. Fitness landscape and population distributions of F3 after (a) initialization, (b) 50 iterations and (c) final iterations.

4. Discussion

4.1. Parameter analysis

There are three key parameters in SOMDE-DS, that is M for the minimal size of niches, fe_t for the threshold of max function evaluations and p_l for the bound between global selection and local selection. In order to find out the influence of these three parameters, we conduct three comparison experiments.

4.1.1. The study of M

Table 2. The compared results among different values of M.

F	Original		$M = 5$		$M = 20$		$M = 30$	
	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	1.000 (≈)	0.000
F2	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	1.000 (≈)	1.000
F3	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	1.000 (≈)	1.000
F4	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	1.000 (≈)	1.000
F5	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	1.000 (≈)	1.000
F6	**0.995**	0.909	0.975 (≈)	0.727	0.964 (≈)	0.818	0.954 (≈)	0.636
F7	0.775	0.000	0.795 (≈)	0.000	**0.891 (-)**	0.182	0.793 (≈)	0.000
F8	**0.963**	0.091	0.942 (≈)	0.000	0.944 (≈)	0.000	0.953 (≈)	0.000
F9	0.327	0.000	0.297 (≈)	0.000	**0.355 (≈)**	0.000	0.285 (≈)	0.000
F10	**1.000**	1.000	0.992 (≈)	0.909	**1.000 (≈)**	1.000	**1.000 (≈)**	0.000
F11	**0.787**	0.000	0.682 (+)	0.000	0.727 (≈)	0.000	0.682 (+)	0.000
F12	0.636	0.000	**0.704 (≈)**	0.000	0.341 (+)	0.000	0.667 (≈)	0.000
F13	**0.667**	0.000	**0.667 (≈)**	0.000	0.651 (≈)	0.000	**0.667 (≈)**	0.000
F14	**0.667**	0.000	**0.667 (≈)**	0.000	0.651 (≈)	0.000	**0.667 (≈)**	0.000
F15	**0.375**	0.000	**0.375 (≈)**	0.000	0.193 (+)	0.000	0.329 (≈)	0.000

Continued on next page

F	Original		M = 5		M = 20		M = 30	
	PR	SR	PR	SR	PR	SR	PR	SR
F16	**0.667**	0.000	**0.667 (≈)**	0.000	0.575 (≈)	0.000	**0.667 (≈)**	0.000
F17	**0.250**	0.000	**0.250 (≈)**	0.000	**0.250 (≈)**	0.000	0.239 (≈)	0.000
F18	**0.379**	0.000	0.349 (≈)	0.000	0.182 (+)	0.000	0.348 (≈)	0.000
F19	**0.114**	0.000	0.090 (≈)	0.000	0.000 (≈)	0.000	0.091 (≈)	0.000
F20	**0.045**	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
≈ (SOMDE-DS is similar)			19		16		19	
+ (SOMDE-DS is better)			1		3		1	
− (SOMDE-DS is worse)			0		1		0	

From Table 2, when M is set to 10, SOMDE-DS gets the best performance. It can be seen that, the change of minimal size of niche M does not change the results significantly, and therefore, it is safe to say that, SOMDE-DS is not sensitive to the parameter M.

4.1.2. The study of p_l and fe_t

Table 3. The compared results among different values of p_l.

F	Original		p_l = 0.0		p_l = 0.2		p_l = 0.4		p_l = 0.8		p_l = 1.0	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F2	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F3	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F4	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F5	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F6	**0.993**	0.900	0.869 (+)	0.364	0.884 (+)	0.454	0.904 (+)	0.454	0.995 (≈)	0.909	0.768 (+)	0.091
F7	0.800	0.000	0.697 (≈)	0.000	0.639(+)	0.000	0.702 (≈)	0.000	0.826 (≈)	0.000	**0.869 (≈)**	0.000
F8	**0.960**	0.125	0.767 (+)	0.000	0.847 (+)	0.000	0.896 (≈)	0.000	0.910 (+)	0.000	0.669 (+)	0.000
F9	0.326	0.000	0.180 (+)	0.000	0.199 (+)	0.000	0.228 (+)	0.000	0.363 (≈)	0.000	**0.464 (-)**	0.000
F10	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F11	**0.792**	0.000	0.667 (+)	0.000	0.697 (+)	0.000	0.742 (≈)	0.000	0.667 (+)	0.000	0.742 (≈)	0.000
F12	0.625	0.000	0.682 (≈)	0.000	**0.716 (≈)**	0.000	0.704 (≈)	0.000	0.648 (≈)	0.000	0.182 (+)	0.000
F13	**0.667**	0.000	**0.667** (≈)	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000
F14	**0.667**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000
F15	**0.375**	0.000	0.307 (≈)	0.000	0.341 (≈)	0.000	0.364 (≈)	0.000	0.318 (≈)	0.000	0.352 (≈)	0.000
F16	**0.667**	0.000	0.636 (≈)	0.000	**0.667 (≈)**	0.000	0.651 (≈)	0.000	0.606 (≈)	0.000	0.591 (≈)	0.000
F17	0.250	0.000	0.250 (≈)	0.000	0.250 (≈)	0.000	0.250 (≈)	0.000	**0.261 (≈)**	0.000	0.250 (≈)	0.000
F18	0.375	0.000	0.348 (≈)	0.000	0.349 (≈)	0.000	0.348 (≈)	0.000	**0.394 (≈)**	0.000	0.364 (≈)	0.000
F19	**0.128**	0.000	0.023 (≈)	0.000	0.023 (≈)	0.000	0.034 (≈)	0.000	0.011 (≈)	0.000	0.034 (≈)	0.000
F20	**0.056**	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
≈ (SOMDE-DS is similar)			16		15		18		18		17	
+ (SOMDE-DS is better)			7		5		2		2		3	
− (SOMDE-DS is worse)			0		0		0		0		0	

p_l is an important parameter in DS strategy and the compared results among different values of p_l is shown in Table 3. When the value of p_l is too high, DS strategy prefers global selection and when the value is too low, local selection is preferred. The unbalance between global selection and local selection lead to worse results. The experiment results show that 0.6 may be a proper value for p_l.

fe_t is another important parameter in DS strategy and the compared results among different values of fe_t are shown in Table 4. When the value of fe_t is too low, local selection is inhibited, and thus the convergence of population is also suppressed. The results of other three values of fe_t are similar and this can be credited to the fast convergence of SOMDE-DS.

In conclusion, p_l should be a proper medium value between [0, 1] and fe_t should be a medium or higher value. The value of both p_l and fe_t are involved in the exploration and exploitation of the proposed algorithm, and different values do make a difference to the results.

Table 4. The compared results among different values of fe_t.

F	Original		$fe_t = 0.1$		$fe_t = 0.3$		$fe_t = 0.5$		$fe_t = 0.7$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F2	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F3	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F4	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F5	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F6	**0.993**	0.900	0.980 (≈)	0.818	0.984 (≈)	0.727	0.990 (≈)	0.818	0.954 (≈)	0.636
F7	0.800	0.000	**0.864 (≈)**	0.000	0.838(≈)	0.000	0.856 (≈)	0.000	0.805 (≈)	0.000
F8	0.960	0.125	0.746 (+)	0.000	0.961 (≈)	0.000	**0.985 (≈)**	0.000	**0.985 (≈)**	0.182
F9	0.326	0.000	**0.441 (+)**	0.000	0.398 (≈)	0.000	0.369 (≈)	0.000	0.355 (≈)	0.000
F10	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F11	**0.792**	0.000	0.727 (≈)	0.000	0.742 (≈)	0.000	0.712 (≈)	0.000	0.697 (≈)	0.000
F12	**0.625**	0.000	0.443 (+)	0.000	0.681 (≈)	0.000	0.682 (≈)	0.000	**0.716 (≈)**	0.000
F13	**0.667**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000
F14	**0.667**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000
F15	**0.375**	0.000	0.352 (≈)	0.000	0.341 (≈)	0.000	**0.375 (≈)**	0.000	0.352 (≈)	0.000
F16	**0.667**	0.000	0.636 (≈)	0.000	**0.667 (≈)**	0.000	0.636 (≈)	0.000	0.636 (≈)	0.000
F17	**0.250**	0.000	**0.250 (≈)**	0.000	**0.250 (≈)**	0.000	**0.250 (≈)**	0.000	**0.250 (≈)**	0.000
F18	**0.375**	0.000	0.333 (≈)	0.000	0.348 (≈)	0.000	0.378 (≈)	0.000	**0.394 (≈)**	0.000
F19	**0.128**	0.000	0.045 (≈)	0.000	0.091 (≈)	0.000	0.057 (≈)	0.000	0.028 (≈)	0.000
F20	**0.056**	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
≈ (Original is similar)			17		20		20		20	
+ (Original is better)			3		0		0		0	
− (Original is worse)			0		0		0		0	

4.2. Component analysis

To investigate the influence of DE operators, we studied 3 mutation strategies and 2 selection strategies in addition to the original one. There is only one difference between each variant and our original method, either mutation or selection.

4.2.1. Mutation

It can be seen from Table 5, that the local best individual guided mutation variant and global best individual guided mutation variant do not perform well. When the global best mutation strategy is used, the direction of mutation is dominated by the global best individual in the whole population, eliminating the role of clustering. Consequently, this variant degrades to a global best individual guided DE and it is no wonder that this variant does not perform well. The defect of the local best individual guided mutation variant is similar to that of the global one. Guided by the best individual in the niche, every other individual int the same niche is likely to converge to the same optima, leading to poor performance.

Table 5. The compared results among different mutation strategies.

F	Original		Current Mutation		Local Best Mutation		Global Best Mutation	
	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	0.500 (+)	0.000
F2	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	0.758 (+)	0.182
F3	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	1.000 (≈)	1.000
F4	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	0.250 (+)	0.000
F5	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	0.727 (+)	0.455
F6	**0.995**	0.909	0.843 (+)	0.000	0.480 (+)	0.000	0.066 (+)	0.000
F7	0.775	0.000	**0.884 (-)**	0.000	0.593 (+)	0.000	0.199 (+)	0.000
F8	**0.963**	0.091	0.017 (+)	0.000	0.397 (+)	0.000	0.012 (+)	0.000
F9	0.327	0.000	**0.423 (-)**	0.000	0.310 (≈)	0.000	0.076 (+)	0.000
F10	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	0.250 (+)	0.000
F11	**0.787**	0.000	0.667 (+)	0.000	0.667 (+)	0.000	0.167 (+)	0.000
F12	0.636	0.000	0.261 (+)	0.000	0.227 (+)	0.000	0.102 (+)	0.000
F13	**0.667**	0.000	**0.667 (≈)**	0.000	0.652 (≈)	0.000	0.167 (+)	0.000
F14	**0.667**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	0.167 (+)	0.000
F15	**0.375**	0.000	**0.375 (≈)**	0.000	0.364 (≈)	0.000	0.125 (+)	0.000
F16	**0.667**	0.000	0.636 (≈)	0.000	**0.667 (≈)**	0.000	0.167 (+)	0.000
F17	**0.250**	0.000	**0.250 (≈)**	0.000	**0.250 (≈)**	0.000	0.080 (+)	0.000
F18	0.379	0.000	0.364 (≈)	0.000	0.467 (-)	0.000	0.167 (+)	0.000
F19	**0.114**	0.000	0.091 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
F20	**0.045**	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
≈ (Original is similar)			13		12		1	
+ (Original is better)			5		7		19	
− (Original is worse)			2		1		0	

Although the performance of the current mutation variant is not as good as the original one, it does perform way better than the two best individual guided mutation variants. Comparing the results of the original method with that of the current mutation variant, it is easy to notice that the current mutation variant does extremely bad in F8. Every offspring produced using the current mutation strategy is similar to its parent, meaning that the ability to explore new space is pretty weak. Therefore, this variant performs extremely badly in a 3D function with a wide search space like F8.

From the results, the original one is better. On balance, the original is more universal.

4.2.2. Selection strategy

From Table 6, notice that the original method does not perform as well as the global selection variant in F7 and F9 but performs way better than the latter in F8. It is a tradeoff of combining local selection and global selection. Comparing the local selection variant and the global one, the former performs better than the latter in F8 while worse in F7 and F9. Although the local selection variant is good at locating multiple global optima, it's hard for it to maintain these global optima. There are up to 81 peaks in F8, meaning that there is more than one peak within a niche. Therefore, frequent selections in a small niche will lead to convergence to one peak, losing the diversity of the niche.

Table 6. The compared results among different selection strategies.

F	Original		Local Selection		Global Selection	
	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F2	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F3	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F4	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F5	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F6	**0.995**	0.909	0.919 (+)	0.727	0.934 (≈)	0.545
F7	0.775	0.000	0.747 (≈)	0.091	0.876 (-)	0.000
F8	**0.963**	0.091	0.713 (+)	0.000	0.465 (+)	0.000
F9	0.327	0.000	0.135 (+)	0.000	0.468 (-)	0.000
F10	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F11	**0.787**	0.000	0.694 (≈)	0.000	0.758 (≈)	0.000
F12	0.636	0.000	**0.659 (≈)**	0.000	0.102 (+)	0.000
F13	**0.667**	0.000	**0.667 (≈)**	0.000	0.621 (≈)	0.000
F14	**0.667**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000
F15	**0.375**	0.000	0.364 (≈)	0.000	0.330 (≈)	0.000
F16	**0.667**	0.000	0.636 (≈)	0.000	**0.667 (≈)**	0.000
F17	**0.250**	0.000	**0.250 (≈)**	0.000	**0.250 (≈)**	0.000
F18	0.379	0.000	0.409 (≈)	0.000	**0.500 (-)**	0.000
F19	**0.114**	0.000	0.034 (≈)	0.000	0.100 (≈)	0.000
F20	**0.045**	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
≈ (Original is similar)			16		14	
+ (Original is better)			4		3	
− (Original is worse)			0		3	

In conclusion, the local selection variant is good at locating multiple optima while global selection one is doing well in maintain found optima. Using DS strategy that combines local selection and global selection strategies, SOMDE-DS can explore new global optima as well as maintaining already found optima.

Through the analysis above, it can be considered that the original strategy, that is random mutation combined with DS strategy, provides the best performance.

4.3. SOM analysis

In this part, we dive into the core of our algorithm, SOM. We investigate two core components of SOM to see their influence on results of our proposed SOMDE-DS, one is neighborhood function and the other is the size of SOM.

4.3.1. Neighborhood function

Neighborhood function determines the rate of change of the neighborhood around the winner neuron [18]. A proper neighborhood function should be selected according to the dataset. We test four commonly used neighborhood functions to see how they work with our algorithm. The four functions are Gaussian, Mexican Hat, Triangle, and Bubble. The results are shown in Table 7.

Table 7. The compared results among different neighborhood functions.

F	Original		Mexican Hat		Triangle		Bubble	
	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F2	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F3	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F4	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F5	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F6	**0.995**	0.909	0.944 (+)	0.636	0.838 (+)	0.182	0.737 (+)	0.000
F7	0.775	0.000	**0.881 (−)**	0.000	0.866 (−)	0.000	0.871 (−)	0.000
F8	**0.963**	0.091	0.746 (+)	0.000	0.639 (+)	0.000	0.710 (+)	0.000
F9	0.327	0.000	0.461 (−)	0.000	0.451 (−)	0.000	**0.471 (−)**	0.000
F10	**1.000**	1.000	**1.000 (≈)**	1.000	0.992 (≈)	0.909	**1.000 (≈)**	1.000
F11	0.787	0.000	**0.833 (≈)**	0.273	0.682 (≈)	0.000	0.697 (≈)	0.000
F12	**0.636**	0.000	0.250 (+)	0.000	0.182 (+)	0.000	0.250 (+)	0.000
F13	**0.667**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000
F14	**0.667**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000
F15	**0.375**	0.000	**0.375 (≈)**	0.000	0.364 (≈)	0.000	0.352 (≈)	0.000
F16	**0.667**	0.000	0.621 (≈)	0.000	**0.667 (≈)**	0.000	**0.667 (≈)**	0.000
F17	**0.250**	0.000	**0.250 (≈)**	0.000	0.239 (≈)	0.000	**0.250 (≈)**	0.000
F18	0.379	0.000	0.333 (+)	0.000	**0.394 (≈)**	0.000	0.364 (≈)	0.000
F19	0.114	0.000	0.068 (≈)	0.000	0.034 (≈)	0.000	**0.170 (≈)**	0.000
F20	**0.045**	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
≈ (SOMDE−DS is similar)			13		14		14	
+ (SOMDE−DS is better)			5		4		4	
− (SOMDE−DS is worse)			2		2		2	

From Table 7, it can be observed that the results of the four variants are similar, meaning that

neighborhood functions do not have a visible impact on the final results. As mentioned above, the selection of the neighborhood function is dominated by the dataset. However, all of the input data used in our SOM is randomly generated and evolves on their own, meaning that there is probably no particular pattern among these data. If there is, it should be credited to the target function, which guides the evolution of data. Hence in most cases, there is no significant difference between every two variants. Besides that, there is a situation where a variant performs better than the others in some functions, for example, the Gaussian variant performs the best in F8. The reason is that the Gaussian neighborhood function is the most suitable for F8.

To draw a conclusion, there is no common pattern in the input data, hence the performance of the four variants is similar. Different neighborhood functions do not make great difference to the performance of SOMDE-DS.

4.3.2. Size of SOM

Another component that might influence the results of SOMDE-DS is the size of SOM. We set the size of SOM as $\sqrt{5\times\sqrt{NP}}$ according to the widely accepted principle. In addition to the original size, we choose 3 different sizes for comparison, one is 3 by 3, one is $\sqrt{2\times\sqrt{NP}}$ by $\sqrt{2\times\sqrt{NP}}$ and the final one is $\sqrt{NP}$ by $\sqrt{NP}$ in ascending order. Most of the test functions in the benchmark CEC' 2013 have 6 or 8 global optima, and therefore 3 by 3 may be a suitable size to see whether a SOM with a size similar to the number of global optima would provide a better result. Likewise, we suggest that $\sqrt{NP}$ by $\sqrt{NP}$ is a suitable size to test whether a SOM with a size similar to the size of the population would provide a better result. And finally, we choose one more size between these two sizes, that is $\sqrt{2\times\sqrt{NP}}$ by $\sqrt{2\times\sqrt{NP}}$ for comparison.

Table 8. The compared results among SOM of different sizes.

F	Original		Size 1		Size 2		Size 3	
	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F2	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F3	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F4	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F5	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F6	**0.995**	0.909	0.848 (+)	0.182	0.682 (+)	0.000	0.717 (+)	0.000
F7	0.775	0.000	**0.886 (≈)**	0.000	0.876 (≈)	0.000	0.869 (≈)	0.000
F8	**0.963**	0.091	0.296 (+)	0.000	0.584 (+)	0.000	0.755 (+)	0.000
F9	0.327	0.000	**0.473 (-)**	0.000	0.465 (-)	0.000	0.455 (-)	0.000
F10	**1.000**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000	**1.000 (≈)**	1.000
F11	0.787	0.000	**0.985 (-)**	0.909	0.818 (≈)	0.091	0.697 (≈)	0.000
F12	**0.636**	0.000	0.080 (+)	0.000	0.148 (+)	0.000	0.205 (+)	0.000
F13	**0.667**	0.000	0.500 (+)	0.000	0.591 (≈)	0.000	**0.667 (≈)**	0.000
F14	**0.667**	0.000	0.606 (≈)	0.000	0.652 (≈)	0.000	**0.667 (≈)**	0.000

Continued on next page

F	Original		Size 1		Size 2		Size 3	
	PR	SR	PR	SR	PR	SR	PR	SR
F15	**0.375**	0.000	**0.375 (≈)**	0.000	**0.375 (≈)**	0.000	0.352 (≈)	0.000
F16	**0.667**	0.000	0.515 (≈)	0.000	0.515 (≈)	0.000	**0.667 (≈)**	0.000
F17	**0.250**	0.000	**0.250 (≈)**	0.000	**0.250 (≈)**	0.000	**0.250 (≈)**	0.000
F18	0.379	0.000	**0.394 (≈)**	0.000	0.349 (≈)	0.000	0.379 (≈)	0.000
F19	0.114	0.000	0.000 (≈)	0.000	0.045 (≈)	0.000	**0.136 (≈)**	0.000
F20	**0.045**	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000	0.000 (≈)	0.000
≈ (SOMDE-DS is similar)			13		16		16	
+ (SOMDE-DS is better)			5		3		3	
− (SOMDE-DS is worse)			2		1		1	

As it can be observed from Table 8, there is no significant difference among all these variants in most functions. That is to say, regardless of the size, SOM is capable of capturing the similarities among the whole population in most cases.

Therefore, in most cases, neither neighborhood function nor the size of SOM does not have a significant effect on the performance. The former is caused by the randomness of the input data while the latter should be credited to the outstanding ability of SOM to capture the similarities in different sizes.

4.4. Dielectric composite design problem

The objective of this problem is to design a dielectric composite with desired effective permittivity in the direction of the field applied [59]. Usually, the desire composite consists of two different materials, and therefore the effective permittivity can be calculated as Eq (5):

$$\varepsilon_{com} = \frac{\varepsilon_1 \varepsilon_2}{\varepsilon_2 k + \varepsilon_1 (1-g)}, \tag{5}$$

where ε_1 and ε_2 are the permittivity of the first and the second material, respectively. ε_{com} is the permittivity of the composite and g is the concentration of the first material. Assume that the desired permittivity of the composite is 1.5, the permittivity of the first material ranges from 10 to 30 and the concentration of the first material varies between 0.1 and 0.9 [59]. Then the problem can be written as a multimodal optimization problem with parameters ε_1 and g as in Eq (6):

$$\begin{cases} \min g_{obj} = \left| \varepsilon_{eff} - 1.5 \right| \\ 0.1 \le g \le 0.9 \\ 10 \le \varepsilon_1 \le 30 \end{cases}, \tag{6}$$

SOMDE-DS is used to solve this problem and the evolution is shown in Figure 10.

At first, individuals are randomly generated. Then during the evolution, all individuals evolve towards their corresponding optima. It took a few iterations for almost all individuals to converge to global optima. It can be concluded that SOMDE-DS not only perform well in benchmark but also

can solve practical engineering problems.

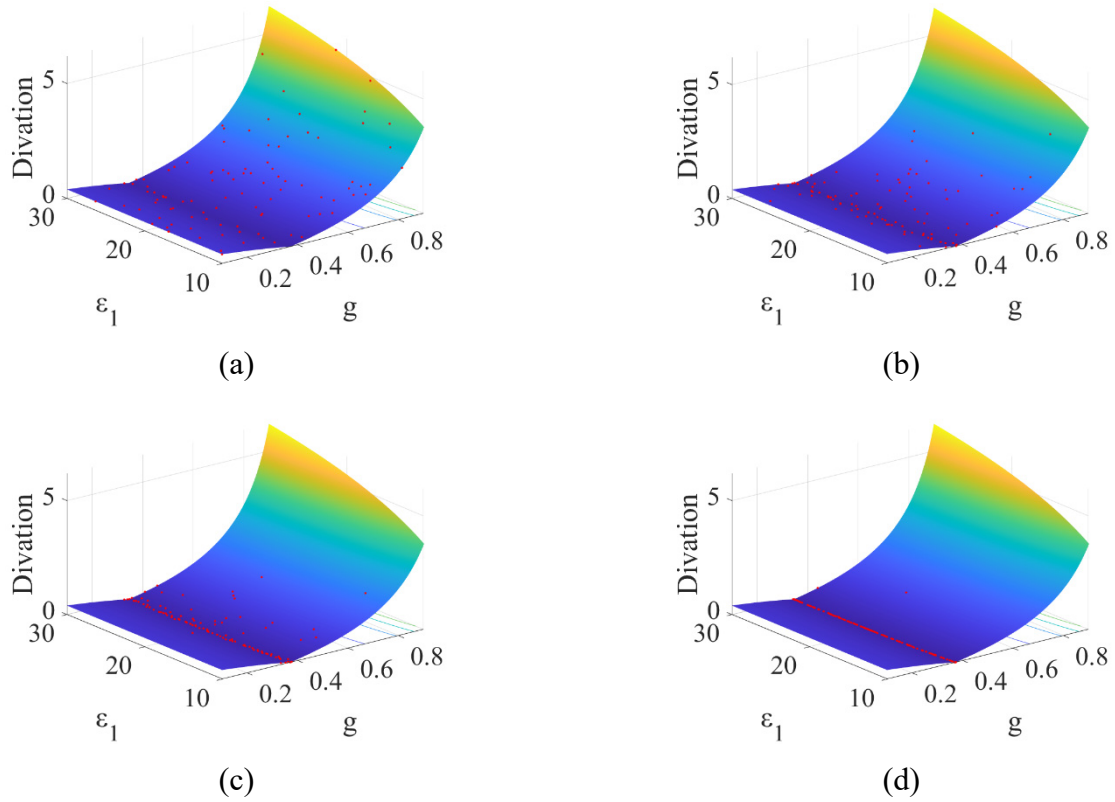


Figure 10. Fitness landscapes of dielectric composite design problem and population distribution during evolution. Population distribution after (a) initialization, (b) 5 iterations, (c) 10 iteration ands (d) 20 iterations.

5. Conclusions

In this paper, a novel SOM based MMOP algorithm SOMDE-DS is proposed. SOM is trained with the current population and then used to divide the population into several clusters. Then, these clusters are further aggregated to form more reasonable and overlapping niches by proposed VNS. After the formation of these niches, the offspring are generated by the mutation and crossover operations. Finally, the proposed DS strategy can effectively select promising individuals into next generation.

In conclusion, the SOM and VNS strategy ensure the similarity between individuals within the same niches and increase the possibility to locate more global optima. DS strategy effectively balance the ability of locating new optima and maintaining the found optima. Compared with several wildly used multimodal algorithms, SOMDE-DS can achieve satisfying results. Besides, the experimental results in solving a real-world application (i.e., dielectric composite design problem) also illustrate the effectiveness of our algorithm.

Although the proposed SOMDE-DS algorithm performs well in solving MMOPs, there are still some limitations in this study. For example, the sensitivity study is done separately without considering the interaction effect between the different algorithm parameters. Therefore, further researches should fully figure out the interactive mechanisms of the different algorithm parameters.

Acknowledgments

This work is suppored by Guangdong Basic and Applied Basic Research Foundation (2021A151511073) and suppored by the Fundametal Research Funds for the Central Universities, JLU: 93K172021K17.

Conflict of interest

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled.

References

1. R. Tanabe, H. Ishibuchi, A review of evolutionary multimodal multiobjective optimization, *IEEE Trans. Evol. Comput.*, **24** (2020), 193–200. https://doi.org/10.1109/TEVC.2019.2909744
2. D. Chen, Y. Li, A development on multimodal optimization technique and its application in structural damage detection, *Appl. Soft Comput.*, **91** (2020), 106264. https://doi.org/10.1016/j.asoc.2020.106264
3. K. Wang, J. Zheng, F. Lu, H. Gao, A. Palanisamy, S. Zhuang, Varied-line-spacing switchable holographic grating using polymer-dispersed liquid crystal, *Appl. Opt.*, **55** (2016), 4952–4957. https://doi.org/10.1364/AO.55.004952
4. A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, et al., Improved protein structure prediction using potentials from deep learning, *Nature*, **577** (2020), 706–710. https://doi.org/10.1038/s41586-019-1923-7
5. J. Zhang, G. Ding, Y. Zou, S. Qin, J. Fu, Review of job shop scheduling research and its new perspectives under Industry 4.0, *J. Intell. Manuf.*, **30** (2019), 1809–1830. https://doi.org/10.1007/s10845-017-1350-2
6. A. Lambora, K. Gupta, K. Chopra, Genetic algorithm—a literature review, in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing*, 2019. https://doi.org/10.1109/COMITCon.2019.8862255
7. Bilal, M. Pant, H. Zaheer, L. Garcia-Hernandez, A. Abraham, Differential Evolution: a review of more than two decades of research, *Eng. Appl. Artif. Intell.*, **90** (2020), 103479. https://doi.org/10.1016/j.engappai.2020.103479
8. H. Zhao, Z. H. Zhan, Y. Lin, X. F. Chen, X. N. Luo, J. Zhang, et al., Local binary pattern based adaptive differential evolution for multimodal optimization problems, *IEEE Trans. Cybern.*, **50** (2020), 3343–3357. https://doi.org/10.1109/TCYB.2019.2927780
9. H. Zhao, Z. H. Zhan, J. Zhang, Adaptive guidance–based differential evolution with archive strategy for multimodal optimization problems, in *Proceedings IEEE Congress on Evolutionary Computation*, (2020), 1–8. https://doi.org/10.1109/CEC48606.2020.9185582

10. K. E. Parsopoulos, M. N. Vrahatis, Unified particle swarm optimization for solving constrained engineering optimization problems, in *Advances in Natural Computation* (eds. L. Wang, K. Chen, and Y. S. Ong), Springer Berlin Heidelberg, (2005), 582–591. https://doi.org/10.1007/11539902_71
11. J. Liang, S. Ge, B. Y. Qu, K. Yu, F. Liu, H. Yang, et al., Classified perturbation mutation based particle swarm optimization algorithm for parameters extraction of photovoltaic models, *Energy Convers. Manag.*, **203** (2020), 112138. https://doi.org/10.1016/j.enconman.2019.112138
12. H. X. Chen, D. L. Fan, F. Lu, W. J. Huang, J. M. Huang, C. H. Cao, et al., Particle swarm optimization algorithm with mutation operator for particle filter noise reduction in mechanical fault diagnosis, *Int. J. Pattern Recognit. Artif. Intell.*, **34** (2020), 2058012. https://doi.org/10.1142/S0218001420580124
13. X. F. Song, Y. Zhang, Y. N. Guo, X. Y. Sun, Y. L. Wang, Variable-size cooperative coevolutionary particle swarm optimization for feature selection on high-dimensional data, *IEEE Trans. Evol. Comput.*, **24** (2020), 882–895. https://doi.org/10.1109/TEVC.2020.2968743
14. L. Qing, W. Gang, Y. Zaiyue, W. Qiuping, Crowding clustering genetic algorithm for multimodal function optimization, *Appl. Soft Comput.*, **8** (2008), 88–95. https://10.1016/j.asoc.2006.10.014
15. B. Y. Qu, P. N. Suganthan, S. Das, A distance-based locally informed particle swarm model for multimodal optimization, *IEEE Trans. Evol. Comput.*, **17** (2013), 387–402. https://doi.org/10.1109/TEVC.2012.2203138
16. Z. J. Wang, Z. H. Zhan, Y. Lin, W. J. Wu, H. Q. Yuan, T. L. Gu, et al., Dual-strategy differential evolution with affinity propagation clustering for multimodal optimization problems, *IEEE Trans. Evol. Comput.*, **22** (2018), 894–908. https://doi.org/10.1109/TEVC.2017.2769108
17. T. Kohonen, Self-organized formation of topologically correct feature maps, *Biol. Cybern.*, **43** (1982). https://doi.org/10.1007/BF00337288
18. T. Kohonen, Essentials of the self-organizing map, *Neural Networks*, **37** (2013), 52–65. https://doi.org/10.1016/j.neunet.2012.09.018
19. R. Storn, K. Price, Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces, *J. Glob. Optim.*, **11** (1997), 341–359, https://doi.org/10.1023/A:1008202821328
20. B. Xue, M. Zhang, W. N. Browne, X. Yao, A survey on evolutionary computation approaches to feature selection, *IEEE Trans. Evol. Comput.*, **20** (2016), 606–626. https://doi.org/10.1109/TEVC.2015.2504420
21. H. Bersini, M. Dorigo, S. Langerman, G. Seront, L. Gambardella, Results of the first international contest on evolutionary optimisation, in *Proceedings of IEEE International Conference on Evolutionary Computation*, (1996), 611–615. https://doi.org/10.1109/ICEC.1996.542670
22. S. M. Guo, J. S. H. Tsai, C. C. Yang, T. H. Hsu, A self-organizing approach for L-SHADE incorporated with eigenvector-based crossover and successful-parent-selecting framework on CEC 2015 benchmark set, in *Proceedings IEEE Congress on Evolutionary Computation*, (2015), 1003–1010. https://doi.org/10.1109/CEC.2015.7256999

23. N. Awad, M. Ali, P. Suganthan, R. Reynolds, An ensemble sinusoidal parameter adaptation incorporated with L-SHade for solving CEC2014 benchmark problems, in *Proceedings IEEE Congress on Evolutionary Computation*, (2016), 2958–2965. https://doi.org/10.1109/CEC.2016.7744163
24. R. Tanabe, A. Fukunaga, Improving the search performance of SHADE using linear population size reduction, in *Proceedings IEEE Congress on Evolutionary Computation*, (2014), 1658–1665. https://doi.org/0.1109/CEC.2014.6900380
25. N. Awad, M. Ali, P. Suganthan, Ensemble sinusoidal differential covariance matrix adaptation with Euclidean neighborhood for solving CEC2017 benchmark problems, in *Proceedings IEEE Congress on Evolutionary Computation*, (2017), 372–379. https://doi.org/10.1109/CEC.2017.7969336
26. S. Akhmedova, V. Stanovov, E. Senmenkin, LSHADE algorithm with a rank–based selective pressure strategy for the circular antenna array design problem, in *International Conference on Informatics in Control, Automation and Robotics*, (2018), 159–165. https://doi.org/10.5220/0006852501590165
27. 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** (2006), 646–657. https://doi.org/10.1109/TEVC.2006.872133
28. X. Rui, D. Wunsch, Survey of clustering algorithms, *IEEE Trans. Neural Networks*, **16** (2005), 645–678. https://doi.org/10.1109/TNN.2005.845141
29. T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, A. Y. Wu, An efficient k–means clustering algorithm: analysis and implementation, *IEEE Trans. Pattern Anal. Mach. Intell.*, **24** (2002), 881–892. https://doi.org/10.1109/TPAMI.2002.1017616
30. B. Y. Qu, C. Li, J. Liang, L. Yan, K. Yu, Y. Zhu, A self-organized speciation based multi–objective particle swarm optimizer for multimodal multi-objective problems, *Appl. Soft Comput.*, **86** (2020), 105886. https://doi.org/10.1016/j.asoc.2019.105886
31. Y. Hu, j. Wang, J. Liang, K. Yu, H. Song, Q. GUO, et al., A self-organizing multimodal multi-objective pigeon–inspired optimization algorithm, *Sci. China Inf. Sci.*, **62** (2019), 70206. https://doi.org/10.1007/s11432-018-9754-6
32. H. Zhang, A. Zhou, S. Song, Q. Zhang, X. Z. Gao, J. Zhang, A self-organizing multiobjective evolutionary algorithm, *IEEE Trans. Evol. Comput.*, **20** (2016), 792–806. https://doi.org/10.1109/TEVC.2016.2521868
33. A. M. Kashtiban, S. Khanmohammadi, A genetic algorithm with SOM neural network clustering for multimodal function optimization, *J. Intell. Fuzzy Syst.*, **35** (2018), 4543–4556. https://doi.org/10.3233/JIFS-131344
34. K. Zielinski, R. Laur, Stopping criteria for differential evolution in constrained single-objective optimization, in *Advances in Differential Evolution*, (eds. U. K. Chakraborty), Springer Berlin Heidelberg, (2008), 111–138. https://doi.org/10.1007/978–3–540–68830–3_4
35. Q. Yang, W. N. Chen, Z. Yu, T. Gu, Y. Li, H. Zhang, J.Zhang, et al., Adaptive multimodal continuous ant colony optimization, *IEEE Trans. Evol. Comput.*, **21** (2017), 191–205. https://doi.org/10.1109/TEVC.2016.2591064
36. A. Hackl, C. Magele, W. Renhart, Extended firefly algorithm for multimodal optimization, in *Proceedings 19th International Symposium on Electrical Apparatus and Technologies*, (2016), 1–4. https://doi.org/10.1109/SIELA.2016.7543010

37. R. Thomsen, Multimodal optimization using crowding–based differential evolution, in *Proceedings of the 2004 Congress on Evolutionary Computation*, (2004), 1382–1389. https://doi.org/10.1109/CEC.2004.1331058
38. A. Petrowski, A clearing procedure as a niching method for genetic algorithms, in *Proceedings of IEEE International Conference on Evolutionary Computation*, (1996), 798–803. https://doi.org/10.1109/ICEC.1996.542703
39. B. Y. Qu, P. N. Suganthan, J. J. Liang, Differential evolution with neighborhood mutation for multimodal optimization, *IEEE Trans. Evol. Comput.*, **16** (2012), 601–614. https://doi.org/10.1109/TEVC.2011.2161873
40. X. Li, Niching without niching parameters: particle swarm optimization using a ring topology, *IEEE Trans. Evol. Comput.*, **14** (2010), 150–169. https://doi: 10.1109/TEVC.2009.2026270
41. Z. Wei, W. Gao, G. Li, Q. Zhang, A penalty-based differential evolution for multimodal optimization, *IEEE Trans. Cybern.*, **2021** (2021). https://doi.org/10.1109/TCYB.2021.3117359
42. X. Lin, W. Luo, P. Xu, Differential evolution for multimodal optimization with species by nearest–better clustering, *IEEE Trans. Cybern.*, **51** (2021), 970–983. https://doi.org/10.1109/TCYB.2019.2907657
43. Z. J. Wang, Z. H. Zhan, Y. Lin, W. J. Yu, H. Wang, S. Kwong, J. Zhang, et al., Automatic niching differential evolution with contour prediction approach for multimodal optimization problems, *IEEE Trans. Evol. Comput.*, **24** (2020), 114–128. https://doi.org/10.1109/TEVC.2019.2910721
44. X. Zhuoran, M. Polojärvi, M. Yamamoto, M. Furukawa, Attraction basin estimating GA: an adaptive and efficient technique for multimodal optimization, in *Proceedings IEEE Congress on Evolutionary Computation*, (2013), 333–340. https://doi.org/10.1109/CEC.2013.6557588
45. J. Liang, K. Qiao, T. Y. Cai, K. Yu, A clustering–based differential evolution algorithm for solving multimodal multi–objective optimization problems, *Swarm Evol. Comput.*, **60** (2021), 100788. https://doi.org/10.1016/j.swevo.2020.100788
46. Z. Hu, T. Zhou, Q. Su, M. Liu, A niching backtracking search algorithm with adaptive local search for multimodal multiobjective optimization, *Swarm Evol. Comput.*, **69** (2022), 101031. https://doi.org/10.1016/j.swevo.2022.101031
47. M. G. Epitropakis, V. P. Plagianakos, M. N. Vrahatis, Finding multiple global optima exploiting differential evolution's niching capability, in *Proceedings IEEE Symposium on Differential Evolution*, (2011), 1–8. https://doi.org/10.1109/SDE.2011.5952058
48. M. G. Epitropakis, X. Li, E. K. Burke, A dynamic archive niching differential evolution algorithm for multimodal optimization, in *Proceedings IEEE Congress on Evolutionary Computation*, (2013), 79–86. https://doi.org/10.1109/CEC.2013.6557556
49. J. Wang, Enhancing particle swarm algorithm for multimodal optimization problems, in *Proceedings International Conference on Computing Intelligence and Information System*, (2017), 1–6. https://doi.org/10.1109/CIIS.2017.10
50. W. Liu, L. Liu, T. Zhang, J. Liu, Multimodal function optimization based on improved ABC algorithm, in *Proceedings 9th International Symposium on Computational Intelligence and Design*, (2016), 246–249. https://doi.org/10.1109/ISCID.2016.1063
51. R. Cheng, M. Li, K. Li, X. Yao, Evolutionary multiobjective optimization–based multimodal optimization: fitness landscape approximation and peak detection, *IEEE Trans. Evol. Comput.*, **22** (2018), 692–706. https://doi.org/10.1109/TEVC.2017.2744328

52. Y. Wang, H. Li, G. G. Yen, W. Song, MOMMOP: multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems, *IEEE Trans. Cybern.*, **45** (2015), 830–843. https://doi.org/10.1109/TCYB.2014.2337117
53. W. J. Yu, J. Y. Ji, Y. J. Gong, Q. Yang, J. Zhang, A tri–objective differential evolution approach for multimodal optimization, *Inf. Sci.*, **423** (2018), 1–23. https://doi.org/10.1016/j.ins.2017.09.044
54. V. Steinhoff, P. Kerschke, P. Aspar, H. Trautmann, C. Grimme, Multiobjectivization of local search: single–objective optimization benefits from multi–objective gradient descent, in *Proceedings IEEE Symposium Series on Computational Intelligence*, (2020), 2445–2452. https://doi.org/10.1109/SSCI47803.2020.9308259
55. P. Aspar, P. Kerschke, V. Steinhoff, H. Trautmann, C. Grimme, Multi3: optimizing multimodal single–objective continuous problems in the multi–objective space by means of multiobjectivization, in *Evolutionary Multi–Criterion Optimization* (eds. H. Ishibuchi, et al.), (2021), 12654. https://doi.org/10.1007/978-3-030-72062-9_25
56. C. Grimme, P. Kerschke, P. Aspar, H. Trautmann, M. Preuss, A. Deutz, et al., Peeking beyond peaks: challenges and research potentials of continuous multimodal multi-objective optimization, *Comput. Oper. Res.*, **136** (2021), 105489. https://doi.org/10.1016/j.cor.2021.105489
57. X. Li, A. Engelbrecht, M. G. Epitropakis, *Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization*. Available from: https://www.epitropakis.co.uk/sites/default/files/pubs/cec2013-niching-benchmark-tech-report.pdf.
58. W. Luo, Y. Qiao, X. Lin, P. Xu, M. Preuss, Hybridizing niching, particle swarm optimization, and evolution strategy for multimodal optimization, *IEEE Trans. Cybern.*, **2020** (2020). https://doi.org/10.1109/TCYB.2020.3032995
59. J. Alami, A. El-Imrani, Dielectric composite multimodal optimization using a multipopulation cultural algorithm, *Intell. Data Anal.*, **12** (2008), 359–378. https://doi.org/10.3233/IDA-2008-12404

AIMS Press