An Adaptive Two-Stage Evolutionary Algorithm for Large-Scale Continuous Multi-Objective Optimization

Qiuzhen Lin, Jun Li, Songbai Liu*, Lijia Ma, Jianqiang Li, and Jianyong Chen

College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, PR. China

Abstract:

This paper proposes an adaptive two-stage large-scale multi-objective evolutionary algorithm, in which a neural network-based accelerating optimizer is designed in the first stage to speed up the population's convergence and a layer-based competitive swarm optimizer is used in the second stage to maintain the population's diversity by spreading the solutions obtained in the first stage. To properly train the neural network in the first stage, the whole population, i.e., the training data, is evenly divided into two subsets with different qualities based on the dominant relationship between solutions. Then, the paired low-quality solutions and high-quality solutions, respectively, act as the input and the expected output when training the neural network. In this way, the potentially directional improvement information of the evolutionary population can be learned by this neural network, which is used to guide the adopted differential evolution in promising search directions. Once the population is detected to be evolutionarily stagnated in the first stage, the second stage will be activated for remedying the population's diversity. Specifically, the promising solutions gained in the first stage are assigned into four layers with different qualities by sequentially implementing reference vectors-guided sorting and shift-based density estimation. After that, the solutions in low-quality layers can learn from that in high-quality layers in the proposed competitive swarm optimizer, which allows the population to evolve further in appreciable directions while increasing its diversity. Experimental studies validate the performance of the proposed evolutionary large-scale optimizer when compared with eight state-of-the-art algorithms in solving two widely tested benchmark suites of large-scale multi-objective optimization problems with decision variables ranging from 100 to 1000 under a limited computational resource.

Keywords: Evolutionary Algorithms, Large-Scale Continuous Multi-Objective Optimization, Neural Network, Competitive Swarm Optimization

1. Introduction

Many real-world applications, such as the automatic design of deep learning models [1], vehicle routing in complex scenarios [2], community detection in complex networks [3], etc., involve multiple often mutually conflicting objectives to be optimized simultaneously. This kind of optimization problems is termed multi-objective optimization problems (MOPs), and the conflicting nature of MOPs results in the fact that not a unique solution but a set of equally optimal (or near-optimal) solutions will be obtained

* Corresponding author.

E-mail addresses: songbai@szu.edu.cn (S. Liu).

when optimizing all the objectives at the same time.

During the recent decades, various multi-objective evolutionary algorithms (MOEAs) have been proposed to tackle MOPs, mainly including Pareto-based MOEAs [4], [5], clustering-driven MOEAs [6]-[7], reference vector-guided (or decomposition-based) MOEAs [8], [9], and indicator-oriented MOEAs [10]-[11]. Nevertheless, most of them only focus on improving their environmental selection ability in the objective space and rarely consider enhancing their search capability in the variable space. Due to the curse of dimensionality, these MOEAs encounter more challenges for solving MOPs with the increasing number of decision variables. In this paper, an MOP is called large-scale MOP (LMOP) when the dimensionality of its search space scales up to more than one hundred [12], [13], [14]. Specifically, the traditional evolutionary search strategies used in these MOEAs become inefficient and ineffective when exploring and exploiting the exponentially expanded search space of LMOPs. The reason behind this is that the feature of LMOPs becomes more complex (e.g., the numbers of local optimum will be increased) with the increasing number of variables and the efficiency of evolutionary search methods with some randomness will be greatly reduced in the large-scale search space [15]. Therefore, the search for the optimal solution set becomes more challenging as the evolutionary population may easily get trapped into local optima or converge to the global optima at a particularly slow speed. Typical representatives of such traditional search methods include polynomial mutation (PM), simulated binary crossover (SBX) [16], differential evolution (DE) [17], particle swarm optimization [18], etc. In practice, the LMOPs involving large-scale search space are getting increasing attention. With this regard, a variety of large-scale MOEAs (LMOEAs) have been developed to solve LMOPs, which can be roughly divided into the following three categories.

The first category of LMOEAs is designed following the cooperative coevolution (CC) framework inspired by the idea of "divide-and-conquer". In existing CC-based LMOEAs, the target LMOP is first divided into several subproblems via a variable grouping strategy, followed by solving each subproblem using traditional MOEAs with the corresponding subpopulation. For example, Luis et al. [19] first implemented the CC framework based on the third evolution step of generalized differential evolution (GDE3), called CC-GDE3, for solving LMOPs. In CC-GDE3, the decision variables of the target LMOP are divided into several even groups via a differential variable grouping method, and then the variables in each group construct the subproblem, which is optimized by GDE3. After that, Luis et al. [20] designed a decomposition-based LMOEA, called MOEA/D2, in which the target MOP is divided by a reference vector-guided environmental selection in the objective space and by a random variable grouping method in the search space, respectively. Considering the interactions between variables, Frederick et al. [21] proposed a variable grouping strategy that assigns two decision variables into the same group when they are interacted on at least one objective. Obviously, the performance of LMOEAs based on CC framework is susceptible to the variable grouping results, as the assignment of relevant variables to different groups may mislead the direction of evolutionary search [15]. However, the grouping methods introduced above are basically designed in the case of solving single-objective optimization problems, and the direct extension of them to the variable grouping for LMOPs may be inappropriate without considering the conflict between multiple objectives. To address this issue, a graph-based variable grouping method is proposed in [22] and an importance-based variable grouping approach is developed in [23], respectively. Nonetheless, the CC-based LMOEAs are only suitable for solving separable LMOPs, while their performance will be greatly reduced for solving non-separable or partially separable LMOPs [24].

The second category of LMOEAs is designed based on the decision variable analysis (DVA) to detect the contribution type of each variable, including convergence-related, diversity-related, or mixed variables. The representative algorithm of this kind is MOEA/DVA [25], in which the interdependent decision variables are classified into the above three contribution types by checking the non-dominated relationship between the obtained solutions when only perturbing their corresponding decision variables. Similar to the CC-based framework, different types of variables are individually optimized with targeted strategies (e.g., convergence-preferred and diversity-preferred selection strategies are respectively used on the populations correspondingly for the subproblems constructed with convergence-related variables and diversity-related variables). That's why some literatures also classify this kind of LMOEAs as CC-based optimizers [26]. One of the common issues with this kind of LMOEAs is that majority of variables may be classified into mixed variables and thus cannot be optimized specifically. To solve this issue, Zhang et al. [27] proposed a more suitable decision variable division method based on *k*-means clustering [28], where the contribution of variables in optimizing LMOPs only involves two types, i.e., convergence-related and diversity-related variables. To balance the convergence and diversity when separately processing these two types of decision variables, Xu et al. [29] proposed a new metric called optimization degree to determine the contributions of different variables. Nevertheless, dividing all variables into two groups based on their contributions will still result in two relatively large-scale search spaces, and traditional evolutionary search operators will still be inefficient. To alleviate this issue, Liu et al. [30] further applied the principal component analysis to get a lower representation space of convergence-related variables, followed by using the interdependence analysis to divide it into multiple subspaces. This kind of LMOEAs has achieved considerable performance in solving LMOPs when allocating a relatively sufficient function evaluation. However, when the assigned computational resources are limited, their performance will deteriorate. The main reason is that most of the function evaluations are used to analyze the contribution of variables, while only a few or almost no computational resources are available to optimize the target LMOP [26].

The third category of LMOEAs is designed to simplify the large-scale search space by using specially designed techniques, including two main categories: problem transformation and dimensionality reduction. The main idea of problem transformation is to map the original LMOP into a relatively small-scale search space, in which a relatively simple representation MOP is formulated. In this way, the optimization of the original LMOP is transformed to the solving of this representation MOP by searching in its small-scale space, which can be effectively explored using traditional evolutionary strategies. For instance, Zille et al. [31] proposed a framework called weighted optimization framework (WOF) for solving LMOPs based on problem transformation, where the selected evolutionary operator is run in the reformulated weight space instead of in the original search space. Specifically, the variables in WOF are firstly divided into multiple groups, followed by associating each variable group with a weight variable. Therefore, the performance of WOF is highly dependent on the used variable grouping strategy and the transformation relationship between the variables of a group and their associated weights. To alleviate this issue, He et al. [32] further proposed a generic large-scale multiobjective optimization framework, termed LSMOF, in which the problem transformation is achieved without using any grouping method. Instead, a bi-directional reference vector-guided strategy is designed in LSMOF to reformulate the original LMOP as a low-dimensional representative MOP. Similarly, Qin et al. [33] extended the problem transformation strategy proposed in LSMOF to develop a new LMOEA (called LMOEA-DS) by directed sampling offspring with the guidance of directional references in the decision space. Regarding dimensionality reduction, the large-scale LMOP is simplified by training a machine learning model to extract its Pareto subspace (or compact representation), followed by effectively searching in this dimensionality-reduced Pareto subspace to produce offspring. For example, Tian et al. [34] proposed a new LMOEA based on Pareto subspace learning, termed MOEA/PSL, in which two unsupervised neural networks, i.e., Boltzmann machine and denoise autoencoder, are utilized to learn the Pareto subspace of discontinued and continued LMOPs, respectively. Besides, Tian et al. [35] tried to reduce the decision space of LMOPs by developing an evolutionary pattern mining approach, which can excavate the Pareto subspace of sparse LMOPs. Although this kind of LMOEAs can accelerate the convergence speed of evolutionary population by simplifying the target LMOP, some effective information in the original large-scale search space is often lost in the search of the simplified representation space, which may lead to the fact that the population easily gets trapped in the local optimum.

Moreover, some LMOEAs that may not belong to the above three categories also have shown very promising performance. For example, a preselection strategy is run to select a balanced parent population in DGEA [36], which is then used to construct direction vectors in the decision spaces for guiding the reproduction of promising offspring solutions. A scalable hypervolume-based LMOEA is proposed in [37] by designing a dual local search mechanism. An objective space-based population generation strategy is proposed in [38], where the offspring are generated in the objective space at first, followed by mapping them back to the search space, aiming to accelerate the evolutionary search. A fuzzing decision variables (FDV) strategy is proposed in [39] to narrow the range of the search space instead of reducing the number of decision variables. In FDV, two evolution stages are run, in which the fuzzy evolution will blur the decision variables in order to speed up the convergence and the precise evolution will increase the population's diversity to evenly approximate the true Pareto-optimal front. Furthermore, a novel LMOEA inspired by competitive swarm optimizer (CSO) is suggested [40], where the resultant optimizer is termed LMOCSO. For CSO, all particles are competitively arranged into the loser-winner pairs. Then, the search direction of a loser particle is always guided by its corresponding winner partner. Recently, various CSO variants, e.g., comprehensive CSO [41] and self-exploratory CSO [42], have been proposed in the literatures to solve LMOPs, which show the promising performance by searching directly in the original large-scale decision space of LMOPs.

Among the LMOEAs introduced above can solve LMOPs effectively, most of them involve two different but complementary evolutionary stages, in which the one stage aims to speed up the convergence while the other expects to remedy the population' diversity, such as WOF [31], LSMOF [32], FDV [39], MOEA/PSL [34], DGEA [36], etc. However, they always allocate fixed proportion of computational resources for these two stages and ignore the diverse difficulties when solving different LMOPs. Thus, to better cope with the challenges brought by different LMOPs, this paper proposes an adaptive two-stage LMOEA for solving continues LMOPs, called ATLMOEA. The main contributions of this paper are summarized as follows:

1) An adaptive two-stage strategy is proposed in this paper, where the first and second stages focus more on speeding up convergence and enhancing diversity of the evolutionary population, respectively. Different from the existing two-stage LMOEAs (e.g., WOF, LSMOF, and FDV), in which a fixed scheme is predetermined to switch one stage to another, ATLMOEA can adaptively switch two evolutionary states and the computational resources can be allocated more rationally. Once the population is detected to be evolutionarily stagnated in the first stage, the second stage will be activated for remedying the population's diversity. In this way, ATLMOEA can fully realize the promising performance complementarity between accelerating evolutionary search and maintaining population diversity when solving LMOPs with different difficulties.

2) An accelerating optimizer is designed in the first stage of ATLMOEA to speed up the population's convergence. To be specific, an artificial neural network (ANN) is trained by using paired low-quality solutions and high-quality solutions, respectively, acting as the input and the expected output in the prepared training data. In this way, the potentially directional improvement information of evolutionary population can be learned by this ANN model, which is used to guide the evolutionary operator search in promising directions.

3) A layer-based CSO is designed in the second stage of ATLMOEA, aiming to remedy the population's diversity obtained in the first stage. Specifically, the population is firstly divided into four layers with different qualities by sequentially implementing the reference vector-guided sorting [43] and shift-based density estimation [44]. Then, the solutions in low-quality layers can learn from that in high-quality layers, which allows the population to evolve further in appreciable directions while increasing its diversity.

To study the effectiveness of ATLMOEA for solving various LMOPs, ATLMOEA is compared with five competitive LMOEAs on the two benchmark suites, i.e., LSMOP [45] and IMF [46]. Here, the number of decision variable ranges from 100 to 1000. Finally, the experimental results validate the performance of ATLMOEA by comparing with its competitors (i.e., MOEA/DVA [25], LMOCSO [40], DGEA [36], WOF [31], LMEA [27], LMOEA-DS [33], FDV [39], and MOEA/PSL [34]).

The rest of this paper is organized as follows. In Section 2, the related work of evolutionary large-scale multiobjective optimization and the motivation to develop ATLMOEA are provided respectively. Then, the details of ATLMOEA are given in Section 3 and the experimental studies of ATLMOEA with the selected competitive LMOEAs are provided in Section 4. Finally, the conclusions and our future work are given in Section 5.

2. Background and Motivation

In this section, the basic definitions related to LMOPs are first given. Since the modified ANN [47] and CSO [48] are used to design the optimizers in the different two stages of the proposed ATLMOEA, the details of them are introduced in Section 2.2 and Section 2.3, respectively. Finally, the motivation of designing ATLMOEA is elaborated in Section 2.4.

2.1. Large-Scale Multi-objective Optimization Problems

Without loss of generality, the mathematical formulation of an LMOP is given as follows [49]:

$$\begin{aligned} &\min \mathrm{F}(\mathrm{x}) = (f_1(\mathrm{x}), f_2(\mathrm{x}), \ldots, f_m(\mathrm{x})), \\ &\text{subject to } \mathrm{x} \in \Omega, \end{aligned} \tag{1}$$

where $\mathrm{x}=\left(x_1, x_2, \ldots, x_n\right)$ is an n-dimensional variable vector in the search space Ω. Here, the value of n for LMOPs is often larger than one hundred, i.e., $n \geq 100$ in this paper. Besides, F(x) is an objective function vector with m conflicting objective functions and $m \geq 2$. A set of Pareto-optimal solutions can be found when solving LMOPs. Let $\mathrm{x}_1, \mathrm{x}_1 \in \Omega$ and x_1 is said to dominate x_2, if and only if $f_i(\mathrm{x}_1) \leq f_i(\mathrm{x}_2)$ for each $i \in \{1, \ldots, m\}$, meanwhile, $f_j(\mathrm{x}_1) < f_j(\mathrm{x}_2)$ for at least one index $j \in \{1, \ldots, m\}$. x_1 is called a Pareto-optimal solution if no solutions from Ω can dominate it. All Pareto-optimal solutions of the target LMOP construct its Pareto-optimal set (PS), and the mapping of all Pareto-optimal solutions into the objective space will construct the Pareto-optimal front (PF). When solving an LMOP, the goal of an LMOEA is to find a final solution set that can approximate the whole PF with a proper tradeoff between convergence and diversity.

2.2. Artificial Neural Network

Structurally, ANN includes three indispensable layers (i.e., input layer, hidden layer, and output layer). Each layer is composed of an indefinite number of neurons. In particular, a neuron can be simulated as shown in Fig. 1, which includes a number of components (e.g., input, weights, bias, summing junction, activation function, and output). In this paper, the mean square error (MSE) [50] and the sigmoid function are used as the loss function and activation function of ANN. Similar to training a regular feedforward neural network, the parameters (i.e., the weights and biases) of ANN can be updated based on the backpropagation with gradient descent [51]. Specifically, based on the present parameters, the steepest descent direction of the loss function can be calculated, followed by iteratively modifying the parameters along this gradient descent direction to reduce the loss. For instance, a weight w_i of the neuron in Fig. 1 can be updated as

$$w_i^+ = w_i - \eta \frac{\partial L}{\partial w_i}, \tag{2}$$

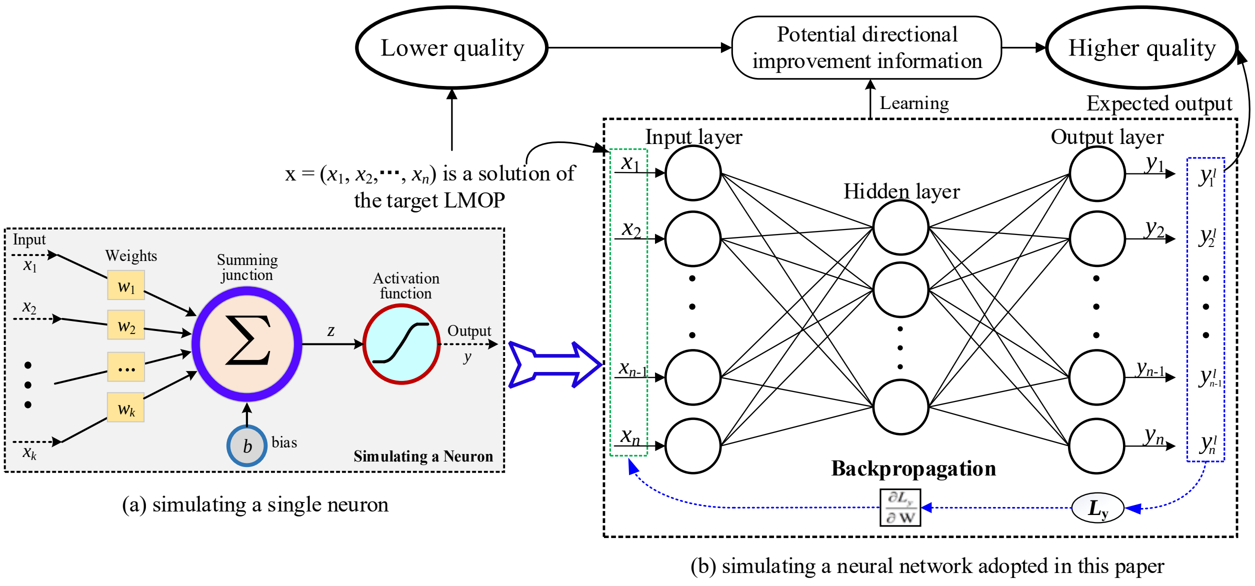
where $\eta > 0$ is the learning rate and L indicates the loss of output based on the MSE. Here, the partial derivative $\partial L / \partial w_i$ can be calculated by using the chain rule twice as follows:

$$\begin{aligned}\frac{\partial L}{\partial w_i} &= \frac{\partial L}{\partial \delta}\frac{\partial \delta}{\partial w_i} = \frac{\partial L}{\partial \delta}\frac{\partial \delta}{\partial z}\frac{\partial z}{\partial w_i} = \frac{\partial L}{\partial y}\frac{\partial \delta}{\partial z}\frac{\partial}{\partial w_i}\left(\sum_{k=1}^{n}(w_k x_k)\right), \\ &= (y - y^t)[z(1-z)]x_i \end{aligned} \tag{3}$$

where y and y^t respectively indicate the output and expected output values of the neuron with the input x, while z can be computed by the summing junction of this neuron as follows:

$$z = \sum_{i=1}^{n}(w_i x_i) + b, \tag{4}$$

where b and z represent the bias and the output of a neuron, respectively.



Lower quality
Potential directional improvement information
Higher quality
Learning
Expected output
x = (x1, x2,···, xn) is a solution of the target LMOP
Input layer
Hidden layer
Output layer
Input
Weights
Summing junction
Activation function
Output
bias
Simulating a Neuron
Backpropagation
(a) simulating a single neuron
(b) simulating a neural network adopted in this paper

Figure 1 Illustration of the general components in a neuron and the constructed neural network.

The general process of training a neural network is presented in **Algorithm 1**, which requires two inputs: the input data **I** and the expected output data **E**. Particularly, both **I** and **E** consist of the solutions to the target LMOP, where the solutions in **E** perform better than that in **I**. As the sigmoid function is adopted in the proposed ANN, all the solutions are first normalized in line 1. Here, the ith variable of a solution x is normalized as follows:

$$x_i' = \frac{x_i - lb_i}{ub_i - lb_i}, i = 1, 2, \ldots, n, \tag{5}$$

where **lb** = (lb_1, …, lb_n) and **ub** = (ub_1, …, ub_n) are the corresponding lower bound and upper bound of the n-dimensional search space for the target LMOP. After preprocessing the training data, the ANN is iteratively learned in line 2 to line 6. Concretely, by inputting each x (belong to **I**), the corresponding loss ***L*** of the ANN is computed according to the actual output y and the expected output y^l using the MSE loss function, as shown in line 4. Here, y^l is randomly selected from **E** in line 3. Then, the parameters of the model are updated based on the backpropagation with gradient descent (represented by $\partial L / \partial w$) in line 5. Finally, the trained ANN is outputted in line 7, which can be used in the first stage of ATLMOEA to speed up the search for the adopted evolutionary operator.

```
Algorithm 1 Training (I, E)
Input: the input data I, the expected output data E
Output: the trained ANN
 1. normalize the data in I and E by (5)
 2. for each solution x ∈ I do
 3.     randomly select an expected output $y^l$ from E
 4.     calculate the loss $L$ of ANN based on x and $y^l$ using MSE
 5.     renew the weights w of ANN based on $\partial L / \partial w$
 6. end for
 7. return the trained ANN
```

2.3. Competitive Swarm Optimizer

Particle swarm optimization (PSO) [52] imitates the predation behavior of birds to complete the search task, in which each particle has two attributes: velocity (v) and position (x). Specifically, each particle finds the personal optimal position individually by exploring in the search space, which is recorded by *pbest*. Then, the global best position is found by sharing all *pbest* among the particle swarm, which is recorded by *gbest*. Each particle in the swarm updates its velocity and position according to its inertia, meanwhile approximating to the *pbest* and *gbest*. Thus, once *pbest* or *gbest* falls into the local optimum, the whole swarm will also easily fall into the local optimum, especially when exploring in the large-scale search space. To solve large-scale optimization problems effectively, an improved PSO variant, called competitive swarm optimizer (CSO), is proposed in [48] to enhance the diversity of the search. In CSO, each particle can be the potential *pbest* or *gbest* based on a randomly pairwise competition. Specifically, two different particles are randomly selected from the swarm. Then, the two particles mutually compete according to their fitness values, and the loser learns from the winner to update its velocity and position, while the winner does not need to be updated or only need slight mutation. Recently, an improved CSO variant for LMOPs is proposed, called LMOCSO [40], in which a shift-based density estimation (SDE) [44] is utilized to calculate the fitness value of each particle in the swarm. In this wary, the competition results between two different particles are determined based on their SDE fitness. After that, the particle who loses the competition can update its velocity ($\overrightarrow{v_l}$) and position ($\overrightarrow{x_l}$) by learning from the winner's position ($\overrightarrow{x_w}$) as follows:

$$\begin{aligned}
&\overrightarrow{v_l}(t+1) = r_0\overrightarrow{v_l}(t) + r_1(\overrightarrow{x_w}(t) - \overrightarrow{x_l}(t)) \\
&\overrightarrow{x_l}(t+1) = \overrightarrow{x_l}(t) + \overrightarrow{v_l}(t+1) + r_0(\overrightarrow{v_l}(t+1) - \overrightarrow{v_l}(t))
\end{aligned} \tag{6}$$

where $\overrightarrow{v_l}(t)$ and $\overrightarrow{x_l}(t)$ indicate the velocity and position of loser particle in the *t*-th generation, respectively; $\overrightarrow{x_w}$(t) represents the position of winner particle in the *t*-th generation; r_0 and r_1 are two random values sampled within [0,1]. It's noted that the loser particle in the current generation is updated with the assistance of the loser and winner particles in the previous generation. Besides, polynomial mutation is executed on all particles to further remedy the swarm's diversity. Under this competitive mechanism, the loser particle can sufficiently learn from its random winner partner, which can ensure the diversity of the swarm. However, the convergence speed of swarm is still slow by driving the search in CSO manner. Thus, LMOCSO performs relatively poor for solving LMOPs when the computational resources are limited [53].

2.4. Motivations

With the number of decision variables increased, the search space is enlarged exponentially, which will lead to the fact that the LMOPs become relatively more challenging to be solved by traditional MOEAs. It is mainly attributed to the poor search capacity of the existing evolutionary search operators (e.g., PM, SBX, DE, and PSO), so that they are only efficient in dealing with low-dimensional MOPs. The search operator used in LMOCSO is specifically designed for LMOPs, but it always needs to assign sufficient function evaluations as the CSO-based search manners [40]-[42] cannot make the evolutionary population converge fast enough to reach the optimum within the relatively limited computational resources. Moreover, most CSO following the pairwise random competition mechanism can simply classify the solutions into two categories (i.e., the loser or the winner). However, as particles have distinct exploration and exploitation capabilities in different evolutionary stages, CSO-based LMOEAs cannot take full use of their potential driving force and ensure the balance between diversity and convergence of population [54].

As introduced in Section 1, most LMOEAs (e.g., WOF [31], LSMOF [32], DGEA [36], MOEA/PSL [34], and FDV [39]) designed for LMOPs try to simplify the large-scale search space at first, and thus they always miss some significant information of the target LMOP if only search in the simplified space. To remedy this issue, they often involve two evolutionary stages, where one stage aims to search in the simplified space for accelerating the population's convergence and the other state focuses on maintaining the population's diversity by searching in the original large-scale decision space of the target LMOP. It has been experimentally validated that the use of two evolutionary stages can significantly improve their performance in solving LMOPs. However, the above LMOEAs may waste some computational resources, as they cannot adaptively judge the switch of two evolutionary stages. Moreover, their used search methods in different evolutionary stages are still not promising as validated in our experiments. Finally, it is also intuitive to improve the performance of the overall optimization by addressing different challenges at different stages, such as in two-stage robust optimization [55] and expensive optimization [56].

Based on the above discussions, this paper proposes an adaptive two-stage LMOEAs (termed ATLMOEA) for handling LMOPs. In our method, the first evolutionary stage aims to speed up the convergence by quickly finding the quasi-optimal solutions. In this stage, a simple ANN, as shown in Fig. 1(b), is designed to learn the potential promising improvement directions, which can drive the evolutionary search, i.e., the DE operator used in this paper, to search the quasi-optimal solutions faster. The second evolutionary stage focuses more on the population's diversity by spreading solutions over the true Pareto front. In this stage, a layered CSO is designed to provide more diversified evolutionary directions around the quasi-optimal solutions obtained in the first stage. In addition, in order to make reasonable allocation of computational resources, the two stages of ATLMOEA are adaptively switched according to the evolutionary state of the population.

3. The proposed algorithm

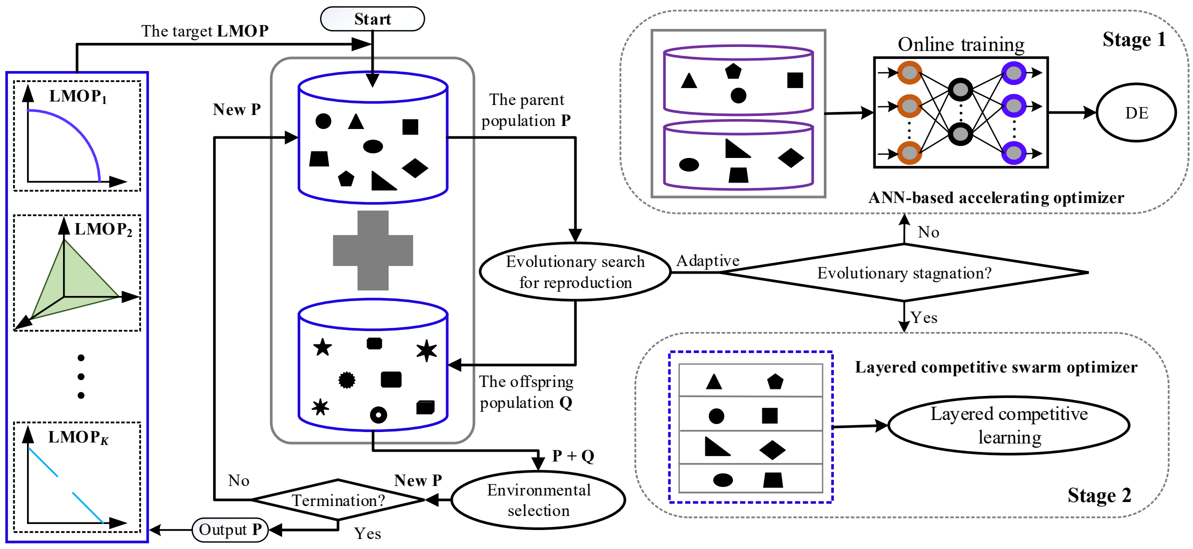


Figure 2 Illustration of the overall flow for the proposed ATLMOEA.

3.1. Main procedures of ATLMOEA

The main procedures of ATLMOEA are presented in **Algorithm 2** with an adaptive two-stage strategy. In the first stage, a neural network model is learned to speed up the population's convergence by quickly searching quasi-optimal solutions close to the PF with the evolutionary operator. In the second stage, a layered competitive swarm optimizer is developed to enhance the population's diversity by spreading the solutions over the approximate PF. In detail, an initial population **P** with *N* solutions is randomly generated in the search space of the target LMOP, where *N* is the population size. Besides, the set of weight vectors, represented by **V**, is generated using the uniform method [57] in the objective space, as shown in line 1 of **Algorithm 2**. Then, the evaluation function (*FE*) counter is initialized to 0, the preset threshold ε used to adaptively control the switch from the first stage to the second stage is set to 0.1 (the values of ε will be experimentally discussed in Section 4), and the symbol *flag* is set to **false** in line 2. Particularly, the first stage is executed when *flag* is **false** and the second stage is activated when *flag* is **true**. Then, the main loop of the evolutionary process is repeatedly run from line 3 to line 11, until the termination condition is satisfied (i.e., $FE \leq FE_{\max}$ in line 3), where $FE_{\max}$ represents the maximum function evaluations preset by the users.

The first evolutionary stage is run from line 4 to line 7. Concretely, the current population **P** is updated in line 5 by an accelerating optimizer (Algorithm 3). This optimizer will also return the parameter *eNum*, which indicates the number of solutions in **P** that are dominated by the newly generated offspring in the accelerating optimizer. Then, the current evolutionary status is discriminated based on the value of *eNum* in line 6. If the evolutionary status is determined to be stagnated (i.e., $eNum/N < \varepsilon$), the symbol *flag* is changed to **true**, which represents the following evolutionary process will switch to the second stage in line 8. The condition $eNum/N < \varepsilon$ indicates that the population is detected to be evolutionarily stagnated or evolved with a very slow speed. Thus, it will waste computational resources if the evolutionary process is still kept in the first stage. To avoid this issue, the population **P** obtained in the first stage is further updated in the second stage by a layered competitive swarm optimizer in line 8. The details of these two optimizers are provided in Section 3.2 and Section 3.3, respectively. To give a more intuitive representation of the proposed ATLMOEA, its overall flow with two stages is illustrated in Figure 2.

```
Algorithm 2 Main Framework of the Proposed ATLMOEA
Input: the maximum function evaluations $FE_{\max}$, the population size $N$
Output: the final population $\mathbf{P}$
1. initialize the weight vectors $\mathbf{V}$ and the current population $\mathbf{P}$
2. initialize $flag$ = false, $\varepsilon = 0.1$ and $FE = 0$
3. while $FE \le FE_{\max}$ do
4.     if $flag$ == false                    //Optimization in the first stage
5.         ($\mathbf{P}$, $eNum$) = Accelerating-Optimizer ($\mathbf{P}$)
6.         set $flag$ = true if $eNum/N < \varepsilon$
7.     else                                  //Optimization in the second stage
8.         $\mathbf{P}$ = Layered-Competitive-Swarm-Optimizer ($\mathbf{P}, \mathbf{V}$)
9.     end if
10.    $FE = FE + N$
11. end while
12. return $\mathbf{P}$
```

3.2. Neural Network-based Accelerating Optimizer

With the increase of decision variables, LMOEAs need more computational resources to tackle the LMOPs, which is caused by the poor efficiency of their adopted search operators (e.g., DE, SBX and PSO). Therefore, an accelerating optimizer is proposed in the first stage of ATLMOEA with three main procedures: 1) training a neural network model $\mathbf{M}_{NN}$ with three layers as shown in Figure 1; 2) designing an improved DE operator based on the learned $\mathbf{M}_{NN}$ to reproduce an offspring population by searching along more promising directions, and 3) running the Pareto-based environmental selection to get an updated population for the next generation. As described in Section 2, the neural network has an excellent capability of learning from the training data. In this paper, the used $\mathbf{M}_{NN}$ tries to learn the potentially directional improvement information of the population, which can be employed to drive the population converge faster. Here, an improved DE operator driven by the learned $\mathbf{M}_{NN}$ is proposed to generate a child solution x^c starting from the parent solution x, as follows:

$$x^C = x + t(x^{P_1} - x^{P_2}) + (1-t)(x^{hq} - x) \quad (7)$$

where x^{hq} is obtained by inputting x into the $\mathbf{M}_{NN}$, and x^{P_1}, x^{P_2} are two randomly selected solutions from the current population **P** ($x^{P_1} \neq x^{P_2} \neq x$). This formula mainly consists of two parts: the distribution part (i.e., $x^{P_1} - x^{P2}$) and the acceleration part (i.e., $x^{hq} - x$). As the evolution progresses, the proportions of the two parts will change, and the acceleration part accounts for less. This is because more high-quality solutions should be saved in the population to converge quickly in the early evolution.

Algorithm 3 presents the whole process of the neural network-based accelerating optimizer, which requires to input the current population **P**. In particular, the offspring population **Q** is initialized as an empty set and *eNum* that records the number of solutions in **P** dominated by that in **Q** is initialized to 0 in line 1. Then, the current population is equally divided into two subsets (i.e., $\mathbf{S}_l$ represents the low-quality subset and $\mathbf{S}_h$ represents the high-quality subset) by the nondominated sorting [17] in line 2. To properly train the initialized $\mathbf{M}_{NN}$ with random parameters in line 3, the solutions in $\mathbf{S}_l$ are iteratively inputted into this $\mathbf{M}_{NN}$, correspondingly, the solutions in $\mathbf{S}_h$ are acted as the expected output in training $\mathbf{M}_{NN}$. Thus, both the input layer and the output layer have *n* nodes, which equal to the number of decision variables for the target LMOP. Besides, for each input solution in $\mathbf{S}_l$, a random solution in $\mathbf{S}_h$ is selected as its expected output. By learning from the progress of low-quality solutions moving toward high-quality solutions, the potentially directional improvement information of the evolutionary population can be extracted in this learned $\mathbf{M}_{NN}$. Starting from the position of a solution x, the x^{hq} obtained by inputting x into the $\mathbf{M}_{NN}$ in line 5 can be used to guide the DE operator searching along more promising directions. In addition, two different solutions are randomly selected as parents in line 6. By this way, a new child solution x^c with higher quality can be produced in line 7 by the improved DE operator, and the x^c is saved in the offspring population **Q** in line 8. After that, all solutions in the parent population **P** are checked in lines 10-14 to identify whether they are dominated by newly generated offspring. Correspondingly, *eNum* records the number of dominated parent solutions. Therefore, the value of *eNum* can indicate the number of offspring solutions better than their parents, so as to show whether the evolutionary population is stagnant in terms of convergence. Finally, the accelerating optimizer selects more promising solutions to survive to the next generation by a Pareto-based environmental selection, which runs a non-dominated sorting first, followed by selecting solutions with better dominant ranks and crowding distances, as suggested in [4].

Algorithm 3 Accelerating-Optimizer (**P**)

```
1.  initialize Q as an empty set and eNum = 0
2.  divide P into two layers $\mathbf{S}_l$ and $\mathbf{S}_h$ via non-dominated sorting
3.  $\mathbf{M}_{NN}$ = Training ($\mathbf{S}_l$, $\mathbf{S}_h$)
4.  for each $x \in \mathbf{P}$ do
5.      obtain a new solution $x^{hq}$ by inputting x to the $\mathbf{M}_{NN}$
6.      select two different solutions $(x^{P_1}, x^{P_2})$ as parents
7.      reproduce a child $x^c$ with solutions $(x^{hq}, x^{P_1}, x^{P_2})$ by (7)
8.      add $x^c$ into Q;
9.  end for
10. for each $x \in \mathbf{P}$ do
11.     if x is dominated by a solution in Q
12.         eNum = eNum + 1
13.     end if
14. end for
15. P = Pareto-Based-Environmental-Selection (P, Q)
16. return (P, eNum)
```

3.3. Layered Competitive Swarm Optimizer

The particles (or solutions) in swarm (or population) have different abilities of exploration and exploitation during different evolutionary stages. If all particles are simply divided into two categories (i.e., loser and winner), they cannot fully exploit their potential and ensure the swarm's diversity. Therefore, the layered learning mechanism is developed in this paper for solving LMOPs. Specifically, the swarm is firstly divided into four categories, and the particles in low layers are considered to perform poorly than those in higher layers. In the CSO-based search, each particle in higher layers can be the potential leader for that in lower layers. The detailed implementation is depicted in Figure 3, where the swarm is divided into four fixed layers from low-quality to high-quality (i.e., L_1, L_2, L_3 and L_4), and then the particles in L_i can learn from that of L_j ($i < j$).

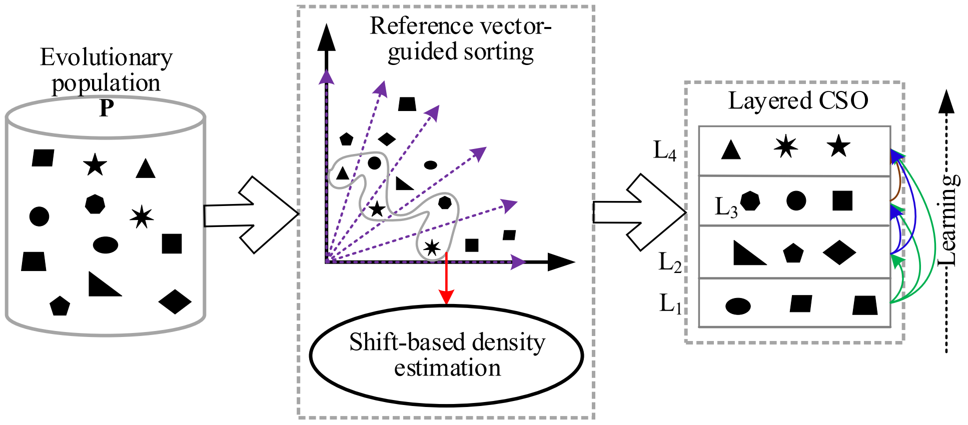

Evolutionary population
P
Reference vector-guided sorting
Shift-based density estimation
Layered CSO
L4
L3
L2
L1
Learning

Figure 3 Illustration of the framework for layer competitive learning.

To clarify this process, its pseudocode is given in **Algorithm 4** with two inputs: the current swarm **P** and the weight vectors **V**. Concretely, this process is divided into two parts, i.e., the swarm partition and layered learning. First, the swarm is normalized in line 1 by the following formula:

$$f_i'(x)=\frac{f_i(x)-z_i^*}{z_i^{nadir}-z_i^*}, \tag{8}$$

where z^* and z^{nadir} are respectively the ideal and nadir points in the objective space, m is the number of objectives and $i=1,2,...,m$. Here, z^* is determined by setting z_i^* as the minimal value of $f_i(x)$ from all particles, while the estimation of z^{nadir} is determined using the method proposed in [58]. Then, the SDE fitness of each particle in swarm **P** is calculated to reflect its diversity attribute, as shown in line 2. After that, all particles are classified into k layers (k = 4 in this paper) based on the reference vector-guided sorting [44] and their SDE fitness in line 3. Besides, the performance level of each particle x, termed *level*(x), is recorded in line 4. To be specific, the particles in **P** are first divided into several fronts based on the reference vector-guided sorting, followed by sorting the particles in each front according to their SDE values. In this way, all particles are equally divided into four layers with different levels. Thereafter, the layered learning is displayed in line 5 to line 11. In this process, the particle pair (x_l, x_h) are obtained in line 7 by comparing the level between two randomly selected particles (x^p and x^q) in line 6, where x_l represents a low-level particle and x_h represents a high-level particle. Subsequently, the position of x_l is updated by learning from x_h using (7) in line 8. In addition, both x_l and x_h are further mutated in line 9 by polynomial mutation, and then are preserved in the swarm **Q**. Here, **Q** is initialized as a clone swarm of **P** in line 1. Finally, the reference vector-guided environmental selection proposed in [59] is adopted in the layered CSO to filter the unpromising solutions in line 12.

```
Algorithm 4 Layered-Competitive-Swarm-Optimizer (P, V)
1. normalize the current swarm P and get a clone swarm Q of P
2. calculate the SDE fitness of each particle in swarm P
3. divide P into $k$ layers based on reference vector-guided sorting and SDE
4. record the $level$(x) of each particle x in P
5. while |P| >1 do
6.     randomly select two particles $x^p$ and $x^q$ from P and delete them from P
7.     $(x_l, x_h) = \begin{cases} (x^p, x^q) & \text{if } level(x^p) > level(x^q) \\ (x^q, x^p) & \text{otherwise} \end{cases}$
8.     update $x_l$ by learning from $x_h$ in (6)
9.     mutate $x_l$ and $x_h$ by polynomial mutation
10.    add $x_l$ and $x_h$ into Q
11. end while
12. P = Reference-Vector-Guided-Environmental-Selection (Q, V)
13. return P
```

4. Experimental Studies

To study the performance of ATLMOEA, it is compared with eight competitive LMOEAs (i.e., MOEA/DVA [25], LMOCSO [40], DGEA [36], WOF [31], LMEA [27], LMOEA-DS [33], FDV [39], and MOEA/PSL [34]) on two benchmark suits (LSMOP [45] and IMF [46]). Then, the parameter sensitivity analysis of ε in ATLMOEA is conducted when dealing with various LMOPs and ATLMOEA is further compared with its variants to further validate the effectiveness of the adaptive two-stage strategy. Finally, to more comprehensively study the performance of ATLMOEA, the time complexity of ATLMOEA is discussed and the average running times of ATLMOEA and other algorithms are collected for evidence.

4.1. Benchmark Problems and Performance Metrics

In this paper, two benchmark suites (LSMOP and IMF) are adopted to evaluate the performance of ATLMOEA. For each test problem in LSMOP and IMF, the number of decision variables is set as n = {100, 300, 500, 1000}. Moreover, the number of objectives for each problem in LSMOP is set to m = {2, 3}. The number of objectives in IMF-IMF3, IMF5-IMF7 and IMF9 is set to m = 2, while that in IMF4 and IMF8 is set to m = 3. Please note that other parameters in the above two benchmark suites are set the same as suggested in their original references. Here, the well-known inverted generational distance (IGD) [60] is adopted as the performance indicator, since IGD can simultaneously evaluate whether the convergence and diversity of the final solution set are good or not. To calculate IGD, a large number of solutions are evenly sampled from the real PF of the target LSMOP. In our experiments, 5000 and 10000 reference points are evenly sampled from the true PF in solving bi-objective and tri-objective test instances, respectively. Let us suppose that P' is a set of reference points uniformly distributed on the real

PF and R is a set of non-dominated solutions selected from the final population. Hence, IGD can be calculated as

$$IGD(P',R) = \frac{\sum_{p \in P'} dis(p,R)}{|P'|} , \tag{9}$$

where $dis(p,R)$ denotes the minimum Euclidean distance between the reference point p and solutions in R, and $|P'|$ denotes the size of P'. A smaller IGD value indicates that the final population with better convergence and diversity is obtained. In addition, in the following experimental studies, the Wilcoxon rank-sum test with a 0.05 significance level is applied to show statistically significant differences on the IGD results. Symbols "+," "−," and "=" indicate that the result of the compared algorithm is, respectively, better than, worse than, and similar to that obtained by ATLMOEA in the following tables.

4.2. The Compared Algorithms and Parameter Settings

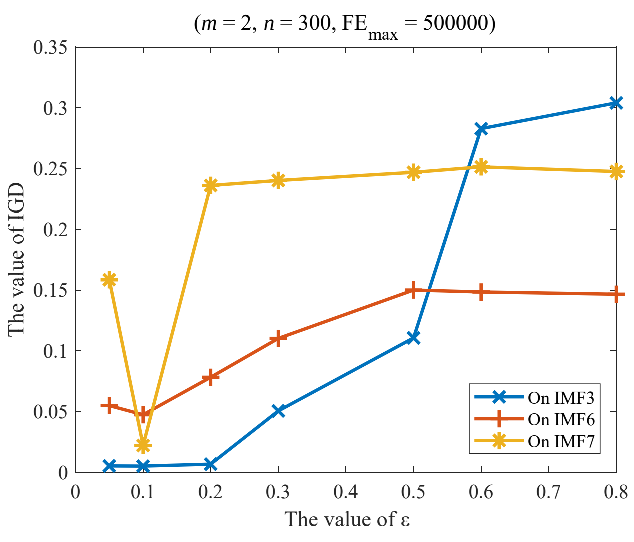
In this paper, ATLMOEA is compared with eight competitive LMOEAs (i.e., MOEA/DVA [25], LMOCSO [40], DGEA [36], WOF [31], LMEA [27], LMOEA-DS [33], FDV [39], and MOEA/PSL [34]) in the experimental studies. The above LMOEAs are run by Matlab codes on PlatEMO [61]. To be fair, the corresponding parameters for these compared algorithms are all set as suggested in their original references. For ATLMOEA, to construct an appropriate $\mathbf{M}_{NN}$, the number of neurons of the input and output layers is equal to the number of decision variables and only one hidden layer with 10 neurons is considered in this $\mathbf{M}_{NN}$. Besides, the learning rate is set to 0.1. For all the compared algorithms, the population size N is set to 100 for bi-objective test problems and 200 for tri-objective test problems. All the compared algorithms are run 20 times independently on each test problem and terminated until a preset maximum number of generations is reached. The maximum number of function evaluations ($FE_{\max}$) is all set to10^6 for solving LSMOP problems. Correspondingly, the number of generations is 10000 and 5000 in handling bi-objective and tri-objective problems, respectively. For solving IMF problems, $FE_{\max}$ is set to 500000 for n = 100 and 300. Correspondingly, the number of generations is 5000 and 2500 in handling bi-objective and tri-objective problems, respectively. $FE_{\max}$ is set to 10^6 for n = 500 and 1000. Correspondingly, the number of generations is 10000 and 5000 in handling bi-objective and tri-objective problems, respectively.

4.3. Parameter Sensitivity Analysis

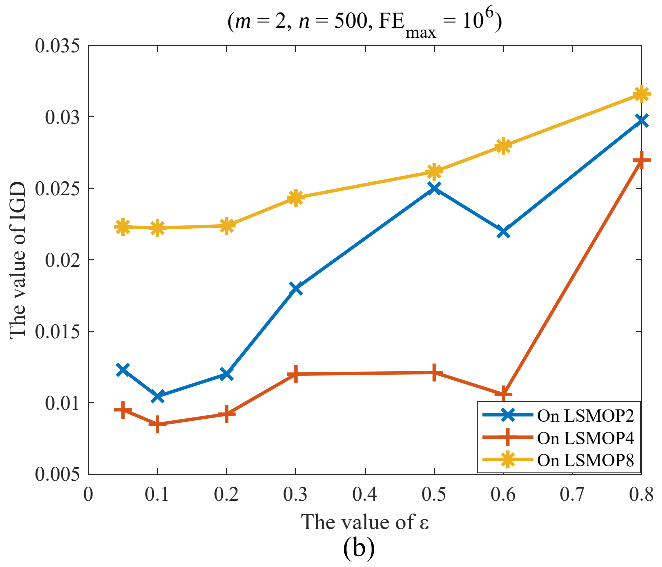
In this paper, with a simple neural network, we utilize an improved DE operator to generate superior offspring to help the whole population accelerate convergence in the first stage. It should be noted that the learning ability of neural network directly affects the performance of the accelerating optimizer in the first stage, and a very important parameter is the number of neurons (i.e., K) in hidden layer. In addition, an adaptive two-stage strategy is used to adjust the evolutionary stage, and an important parameter ε is used in this strategy to determine when to switch the optimizer. Therefore, a parameter sensitivity analysis of ε and K is conducted on the performance of ATLMOEA.

(a) Parameter Sensitivity Analysis of *ε*

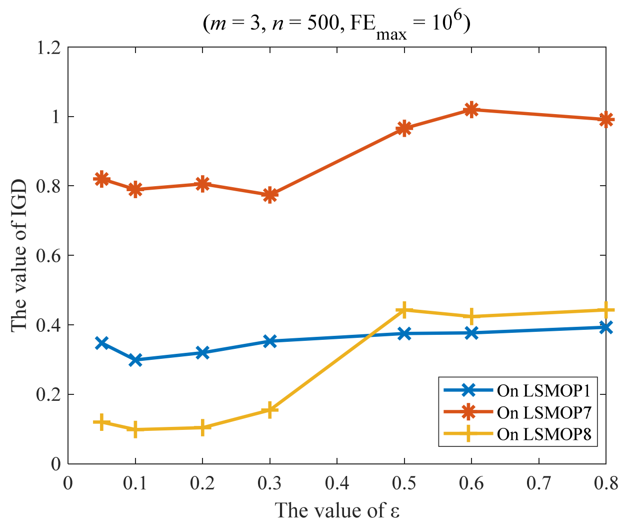
Specifically, different values of ε, ranging from 0.05 to 0.8, are considered in ATLMOEA for dealing with bi-objective LSMOP2, LSMOP4, and LSMOP8 with 500 variables, bi-objective IMF3, IMF6, IMF7 with 300 variables and tri-objective LSMOP1, LSMOP7, and LSMOP8 with 500 variables. As shown in Figure 4, with the increase of ε, the performance of ATLMOEA in dealing with the above problems is deteriorating as a whole. When ε is preset relatively large, the potential risk is that the population may be early switched to the second stage before the solutions have stopped evolving in the first stage, which will directly affect the overall performance of ATLMOEA. Therefore, a large value of ε is not necessarily required in ATLMOEA. Moreover, it can be seen from the curve in Figure 4 that when ε is approximately equal to 0.1, ATLMOEA gets the minimum value, so $\varepsilon = 0.1$ is suggested for ATLMOEA in this paper.



(a)



(b)



(c)

Figure 4 Parameter sensitivity analysis of ε on (a) bi-objective IMF problems (b) bi-objective LSMOP problems and (c) tri-objective LSMOP problems.

(b) Parameter Sensitivity Analysis of *K*

In the process of network construction, the number of neurons in hidden layer greatly affects the learning ability of the neural network. Here, a parameter sensitivity analysis of K is conducted for showing its impact on the performance of ATLMOEA. To be specific, different values of *K*, ranging from 2 to 100, are considered in ATLMOEA for dealing with bi-objective IMF1, IMF5, and IMF6 with 300 decision variables. As shown in Figure 5, the performance of ATLMOEA is worst in handling these three IMFs when $K = 2$. It is attributed to the fact that the neural network does not have sufficient learning ability to assist the improved DE operator when the number of neurons in hidden layer is relatively small. Furthermore, ATLMOEA attains the similar performance in handling these three IMFs when $K \geq 10$. This shows that when $K \geq 10$, the neural network has enough learning ability and helps to generate high-quality solutions to guide the evolution of the population. However, the excessive neurons will not only greatly increase the complexity of the network structure, but also make the learning speed of the network become very slow. It is not advisable to set *K* too small or too large, so $K = 10$ is suggested for ATLMOEA in this paper.

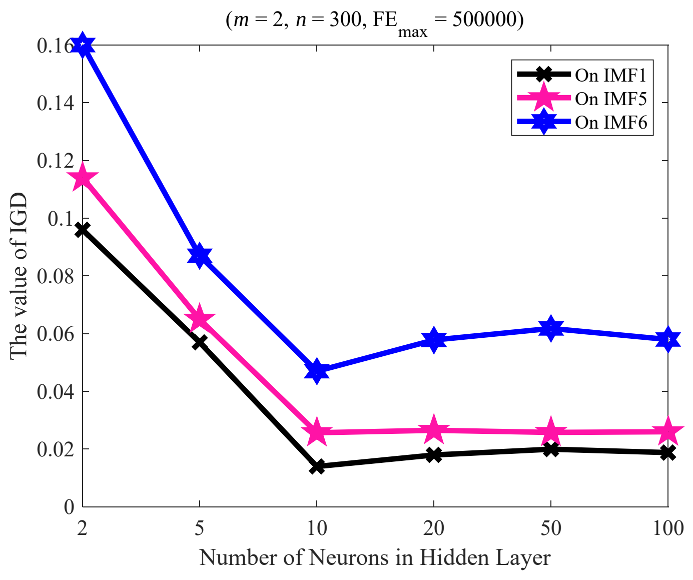


Figure 5 Parameter sensitivity analysis of *K* in ATLMOEA

4.4. Comparison Results on LSMOP and IMF Problems

In this section, to validate the performance of ATLMOEA, it is compared with eight competitive LMOEAs, which cover three different types of LMOEAs introduced in Section 1, for solving LSMOP and IMF problems.

4.4.1. Comparison Results on LSMOP Problems

Here, ATLMOEA is compared with eight competitive LMOEAs (i.e., MOEA/DVA, LMOCSO, DGEA, WOF, LMEA, LMOEA-DS, FDV, and MOEA/PSL) in handling LSMOP1 to LSMOP9 in the case of $m = \{2, 3\}$ and $n = \{100, 300, 500, 1000\}$. The experimental results obtained by ATLMOEA and its five competitors on these problems are presented in Tables I-IV, in which the best result compared with the others is highlighted in bold. As observed from the last row of Tables I-IV, the comparsion summary of ATLMOEA with other compared LMOEAs is listed. To be specific, ATLMOEA shows superiority over LMOCSO, DGEA, FDV, MOEA/PSL, MOEA/DVA, WOF, LMEA, LMOEA-DA on 57, 52, 58, 58, 52, 50, 65 and 48 cases out of 72 test instances, respectively, where ATLMOEA only underperforms them on 4, 12, 5, 4, 13, 13, 1 and 19 cases. In particular, our proposed ATLMOEA shows evident superiority over the above compared LMOEAs on most LSMOP problems, except LSMOP3, LSMOP6 and LSMOP9. Therefore, we can reasonably conclude that ATLMOEA performs better than the above eight competitors in solving most of these LSMOP problems.

Table I The IGD results obtained by ATLMOEA and its four competitors on 2-objective LSMOP benchmarks

Problem	*n*	LMOCSO	DGEA	FDV	MOEA/PSL	ATLMOEA
LSMOP1	100	**3.998e-03(1.04e-04)=**	4.639e-03(1.64e-04)-	1.681e-01(1.15e-01)-	3.346e-01(1.36e-01)-	4.097e-03(2.34e-04)
	300	1.261e-01(1.35e-01)=	7.051e-02(1.20e-02)-	1.791e-01(1.05e-01)-	3.019e-01(5.09e-02)-	**5.285e-02(2.07e-02)**
	500	2.868e-01(5.00e-02)-	1.657e-01(4.47e-03)-	1.484e-01(2.21e-02)=	2.826e-01(7.19e-02)-	**1.337e-01(6.98e-03)**
	1000	5.786e-01(1.06e-01)-	2.516e-01(5.56e-03)-	2.829e-01(7.96e-02)-	4.283e-01(2.25e-01)-	**1.712e-01(7.24e-03)**
LSMOP 2	100	3.806e-02(3.55e-03)-	1.742e-02(2.94e-03)-	3.820e-02(3.33e-03)-	8.492e-02(8.82e-03)-	**1.099e-02(1.59e-03)**
	300	5.885e-02(1.13e-03)-	1.553e-02(7.95e-04)-	5.950e-02(1.81e-03)-	5.052e-02(1.33e-03)-	**1.422e-02(3.21e-03)**
	500	4.087e-02(5.46e-04)-	1.181e-02(7.02e-04)-	4.994e-02(1.03e-03)-	3.103e-02(7.48e-04)-	**1.018e-02(1.02e-03)**
	1000	2.329e-02(2.93e-04)-	8.217e-03(2.09e-04)-	3.259e-02(5.74e-04)-	1.707e-02(4.25e-04)-	**7.836e-03(1.48e-03)**
LSMOP 3	100	1.707e+00(2.23e+00)-	7.237e-01(2.86e-02)-	**5.677e-01(3.98e-02)=**	8.322e-01(3.64e-01)-	5.732e-01(7.41e-02)
	300	1.318e+00(1.19e+00)=	9.938e-01(3.63e-01)=	1.546e+00(1.44e-01)-	**7.687e-01(2.82e-01)+**	1.039e+00(1.16e-01)
	500	1.625e+00(1.38e+00)=	**1.018e+00(3.00e-01)+**	3.725e+00(3.50e-01)-	1.452e+00(7.61e-01)=	1.275e+00(1.52e-01)
	1000	1.721e+00(5.99e-01)=	**8.853e-01(3.07e-01)+**	9.458e+00(1.21e+00)-	2.534e+00(1.97e+00)-	1.454e+00(1.13e-01)
LSMOP 4	100	2.377e-02(1.12e-03)-	1.631e-02(2.56e-03)-	7.008e-02(3.79e-03)-	8.225e-02(5.78e-03)-	**6.972e-03(9.50e-04)**
	300	3.848e-02(4.22e-03)-	1.351e-02(1.11e-03)-	7.030e-02(2.43e-03)-	8.579e-02(5.72e-03)-	**8.712e-03(2.27e-04)**
	500	5.612e-02(3.49e-03)-	2.427e-02(4.88e-03)-	6.946e-02(2.75e-03)-	8.834e-02(1.75e-03)-	**8.757e-03(1.31e-03)**
	1000	4.586e-02(1.28e-03)-	2.310e-02(1.36e-03)-	5.418e-02(1.12e-03)-	4.915e-02(1.08e-03)-	**1.250e-02(8.82e-04)**
LSMOP 5	100	4.329e-03(9.84e-05)=	5.159e-03(4.07e-04)-	2.949e-01(1.28e-01)-	3.824e-01(1.26e-01)-	**4.303e-03(6.21e-05)**
	300	2.652e-01(2.20e-01)-	4.890e-02(3.67e-02)-	3.412e-01(1.21e-01)-	5.023e-01(2.06e-01)-	**1.455e-02(1.44e-03)**
	500	4.060e-01(1.12e-02)-	1.289e-01(5.04e-02)-	3.372e-01(1.92e-01)-	4.673e-01(1.86e-01)-	**8.289e-02(1.44e-02)**
	1000	1.049e+00(6.32e-02)-	4.803e-01(6.39e-02)-	4.809e-01(1.99e-01)=	4.958e-01(1.81e-01)=	**4.274e-01(6.64e-02)**
LSMOP 6	100	7.661e-01(1.33e-02)-	3.862e-01(2.78e-02)-	5.548e-01(1.26e-01)-	9.597e-01(4.07e-01)-	**3.367e-01(2.25e-02)**
	300	7.656e-01(1.50e-02)-	**2.002e-01(1.17e-01)+**	4.246e-01(2.74e-02)=	6.660e-01(4.29e-01)=	4.444e-01(1.11e-02)
	500	7.645e-01(4.36e-03)-	**2.450e-01(2.07e-01)+**	5.305e-01(9.08e-02)-	6.560e-01(4.43e-01)=	4.089e-01(5.58e-02)
	1000	7.652e-01(4.10e-03)-	3.990e-01(2.31e-01)=	6.031e-01(1.26e-01)-	5.136e-01(3.50e-01)=	**3.866e-01(6.55e-02)**
LSMOP 7	100	1.385e+00(4.27e-01)-	1.072e+00(6.08e-01)-	1.138e+00(3.25e-01)-	1.032e+00(2.76e-01)=	**8.554e-01(4.51e-02)**
	300	2.564e+00(3.66e-01)-	1.885e+00(3.39e-01)-	1.757e+00(3.18e-01)=	**1.277e+00(1.59e-01)+**	1.483e+00(3.41e-02)
	500	2.763e+00(4.01e-01)-	1.591e+00(2.11e-01)-	2.063e+00(3.24e-01)-	1.367e+00(1.75e-01)=	**1.476e+00(9.41e-02)**
	1000	2.345e+00(5.49e-01)-	1.678e+00(3.52e-01)=	2.760e+00(1.49e-01)-	1.518e+00(1.22e-01)-	**1.514e+00(1.71e-03)**
LSMOP 8	100	2.309e-02(7.00e-03)-	2.734e-02(1.54e-02)-	2.442e-01(2.69e-02)-	3.424e-01(4.42e-08)-	**4.753e-03(5.44e-04)**
	300	3.836e-02(1.51e-03)-	2.557e-02(2.84e-03)-	2.292e-01(4.07e-02)-	3.826e-01(1.26e-01)-	**1.897e-02(1.62e-03)**
	500	3.334e-02(9.93e-04)-	4.046e-02(2.00e-03)-	2.383e-01(6.35e-03)-	4.273e-01(1.59e-01)-	**1.893e-02(6.65e-04)**
	1000	4.485e-01(4.96e-02)-	1.828e-01(1.77e-02)-	3.206e-01(2.67e-02)-	6.267e-01(1.85e-01)-	**3.768e-02(4.39e-03)**
LSMOP 9	100	**1.288e-02(9.63e-04)+**	1.297e-02(9.46e-04)+	5.671e-01(8.68e-02)+	8.101e-01(1.17e-16)+	8.107e-01(1.86e-03)
	300	**2.663e-01(8.66e-03)+**	3.681e-01(1.21e-01)+	5.199e-01(6.12e-03)+	7.834e-01(8.45e-02)-	7.828e-01(8.57e-02)
	500	**1.925e-01(3.80e-03)+**	2.232e-01(3.92e-03)+	5.155e-01(2.08e-03)+	8.101e-01(1.17e-16)-	8.089e-01(5.23e-04)
	1000	**3.530e-01(4.81e-03)+**	3.666e-01(1.02e-02)+	5.294e-01(5.04e-03)+	5.872e-01(2.89e-01)=	7.233e-01(1.16e-01)
+/-/=		4/26/6	8/25/3	4/27/5	3/25/8	——

Table II The IGD results obtained by ATLMOEA and its four competitors on 2-objective LSMOP benchmarks

Problem	*n*	MOEA/DVA	WOF	LMEA	LMOEA-DS	ATLMOEA
LSMOP1	100	7.243e-03(8.79e-04)-	3.597e-01(1.26e-02)-	8.151e-02(2.01e-01)-	7.491e-02(1.74e-02)-	**4.097e-03(2.34e-04)**
	300	**4.343e-02(3.10e-03)=**	4.377e-01(1.26e-01)-	5.633e-01(4.70e-01)-	1.651e-01(3.15e-03)-	5.285e-02(2.07e-02)
	500	**6.816e-02(1.61e-03)+**	4.710e-01(7.52e-02)-	1.082e+01(3.41e-01)-	1.840e-01(1.91e-03)-	1.337e-01(6.98e-03)
	1000	**2.115e-02(3.79e-04)+**	4.515e-01(1.17e-01)-	8.960e+00(3.62e+00)-	2.274e-01(5.15e-03)-	1.712e-01(7.24e-03)
LSMOP 2	100	1.911e-01(1.90e-03)-	3.081e-02(8.12e-04)-	9.782e-02(6.27e-02)-	3.242e-02(1.24e-03)-	**1.099e-02(1.59e-03)**
	300	9.795e-02(4.83e-04)-	2.831e-02(2.50e-03)-	9.338e-02(1.06e-02)-	1.849e-02(3.04e-04)-	**1.422e-02(3.21e-03)**
	500	6.177e-02(3.34e-04)-	2.519e-02(1.47e-03)-	6.643e-02(5.18e-03)-	1.274e-02(2.34e-04)-	**1.018e-02(1.02e-03)**
	1000	3.348e-02(4.06e-04)-	1.891e-02(2.16e-04)-	4.076e-02(4.43e-04)-	8.141e-03(1.57e-04)=	**7.836e-03(1.48e-03)**
LSMOP 3	100	6.422e-01(5.37e-02)-	6.449e-01(1.02e-03)=	2.942e+00(4.57e+00)=	1.161e+00(1.68e-01)-	**5.732e-01(7.41e-02)**
	300	1.812e+00(1.74e-01)-	**6.635e-01(2.78e-04)+**	1.053e+01(6.00e+00)-	1.524e+00(2.81e-02)-	1.039e+00(1.16e-01)
	500	2.264e+00(1.22e-01)-	**6.723e-01(8.12e-03)+**	2.594e+03(1.87e+03)-	1.555e+00(3.45e-03)-	1.275e+00(1.52e-01)
	1000	1.305e+00(4.70e-02)+	**7.409e-01(1.26e-02)+**	9.915e+02(9.12e+02)-	1.571e+00(5.69e-04)-	1.454e+00(1.13e-01)
LSMOP 4	100	4.262e-02(1.01e-02)-	5.440e-02(3.70e-03)-	9.258e-02(1.12e-02)-	2.295e-02(1.00e-03)-	**6.972e-03(9.50e-04)**
	300	4.672e-02(1.97e-03)-	6.797e-02(1.99e-03)-	8.550e-02(1.80e-03)-	4.362e-02(4.41e-03)-	**8.712e-03(2.27e-04)**
	500	3.318e-02(5.85e-04)-	5.815e-02(2.28e-03)-	1.321e-01(1.51e-02)-	3.604e-02(8.02e-04)-	**8.757e-03(1.31e-03)**
	1000	1.320e-02(5.32e-04)=	4.382e-02(3.02e-03)-	7.493e-02(7.11e-03)-	2.161e-02(2.99e-04)-	**1.250e-02(8.82e-04)**
LSMOP 5	100	8.580e-03(7.01e-04)-	1.143e-01(4.91e-02)-	4.668e-01(2.59e-01)-	1.641e-01(4.60e-03)-	**4.303e-03(6.21e-05)**
	300	1.016e-01(4.93e-03)-	1.309e-01(4.91e-02)-	5.683e-01(2.00e-01)-	1.803e-01(3.08e-02)-	**1.455e-02(1.44e-03)**
	500	1.922e-01(4.18e-03)-	2.061e-01(1.01e-01)-	2.364e+01(6.59e-01)-	3.109e-01(4.73e-02)-	**8.289e-02(1.44e-02)**
	1000	**5.727e-02(7.93e-04)+**	3.214e-01(6.60e-02)+	2.364e+01(3.52e-01)-	6.358e-01(1.30e-01)-	4.274e-01(6.64e-02)
LSMOP 6	100	8.645e-01(3.63e-02)-	5.426e-01(6.53e-02)-	7.371e-01(3.73e-01)-	**2.438e-01(3.84e-02)+**	3.367e-01(2.25e-02)
	300	9.819e-01(5.52e-01)-	5.942e-01(1.16e-01)-	5.365e+00(1.33e+01)-	**2.886e-01(4.45e-02)+**	4.444e-01(1.11e-02)
	500	5.259e+00(3.33e+00)-	4.400e-01(1.16e-01)=	1.618e+03(2.89e+03)-	**3.225e-01(1.06e-02)+**	4.089e-01(5.58e-02)
	1000	1.995e+00(1.87e+00)-	4.166e-01(8.87e-02)-	3.130e+03(2.92e+03)-	**3.128e-01(3.42e-03)+**	3.866e-01(6.55e-02)
LSMOP 7	100	5.036e+00(1.89e+00)-	**7.743e-01(5.68e-02)+**	1.241e+00(1.15e-01)-	1.052e+00(2.77e-01)=	8.554e-01(4.51e-02)
	300	1.071e+01(4.66e+00)-	**9.565e-01(9.68e-02)+**	6.374e+00(1.26e+01)-	1.467e+00(5.04e-03)+	1.483e+00(3.41e-02)
	500	3.603e+01(3.14e+00)-	**1.067e+00(1.12e-01)+**	8.112e+04(3.81e+03)-	1.490e+00(3.45e-03)-	1.476e+00(9.41e-02)
	1000	1.386e+01(4.43e-01)-	**1.506e+00(1.13e-02)+**	8.553e+04(3.99e+03)-	1.507e+00(1.22e-03)+	1.514e+00(1.71e-03)
LSMOP 8	100	3.608e-02(1.43e-03)-	1.412e-01(9.90e-02)-	1.008e-01(2.88e-02)-	6.332e-02(1.88e-02)-	**4.753e-03(5.44e-04)**
	300	6.738e-02(2.85e-03)-	1.614e-01(8.21e-02)-	1.428e-01(8.93e-02)-	1.106e-01(1.32e-02)-	**1.897e-02(1.62e-03)**
	500	1.231e-01(1.28e-03)-	2.056e-01(1.07e-01)-	1.968e+01(4.23e-01)-	2.325e-01(1.74e-02)-	**1.893e-02(6.65e-04)**
	1000	3.605e-02(7.29e-04)=	3.312e-01(1.96e-01)-	2.025e+01(2.86e-01)-	3.612e-01(7.21e-02)-	**3.768e-02(4.39e-03)**
LSMOP 9	100	**1.339e-01(2.30e-02)+**	8.101e-01(1.17e-02)=	8.333e-01(4.60e-01)=	8.101e-01(1.17e-16)+	8.107e-01(1.86e-03)
	300	**1.086e-01(5.19e-03)+**	8.100e-01(1.20e-04)=	9.316e-01(3.31e-01)=	6.677e-01(7.61e-02)+	7.828e-01(8.57e-02)
	500	**2.161e-01(1.55e-02)+**	8.098e-01(2.97e-04)-	4.970e+01(1.71e+01)-	4.630e-01(1.33e-01)+	8.089e-01(5.23e-04)
	1000	**6.278e-02(3.04e-03)+**	8.078e-01(1.02e-03)=	5.753e+01(2.27e+00)-	4.376e-01(1.17e-01)+	7.233e-01(1.16e-01)
+/-/=		8/24/4	8/20/5	0/33/3	10/24/2	——

Table III The IGD results obtained by ATLMOEA and its four competitors on 3-objective LSMOP benchmarks

Problem	*n*	LMOCSO	DGEA	FDV	MOEA/PSL	ATLMOEA
LSMOP1	100	8.063e-02(1.29e-02)-	2.014e-01(4.93e-02)-	1.763e-01(1.69e-02)-	1.845e-01(2.20e-02)-	**4.612e-02(1.34e-02)**
	300	2.353e-01(1.31e-02)=	3.387e-01(1.28e-02)-	**2.124e-01(3.39e-02)=**	2.421e-01(1.78e-02)=	2.392e-01(2.88e-02)
	500	4.652e-01(3.01e-02)-	3.544e-01(1.40e-02)-	4.154e-01(1.94e-02)-	6.016e-01(2.51e-01)-	**2.991e-01(2.04e-02)**
	1000	8.550e-01(3.65e-02)-	3.994e-01(2.15e-02)-	1.256e+00(1.41e-01)-	9.022e-01(1.32e-01)-	**3.547e-01(1.36e-02)**
LSMOP 2	100	7.332e-02(1.58e-03)-	9.264e-02(2.11e-02)-	1.407e-01(2.46e-03)-	1.844e-01(2.73e-03)-	**6.357e-02(2.91e-03)**
	300	5.662e-02(8.39e-04)=	7.460e-02(1.90e-03)-	9.045e-02(1.82e-03)-	9.156e-02(1.74e-03)-	**5.613e-02(7.33e-04)**
	500	4.402e-02(4.15e-04)=	4.906e-02(6.57e-04)-	6.821e-02(1.92e-03)-	6.607e-02(1.90e-03)-	**4.390e-02(4.30e-04)**
	1000	3.496e-02(2.76e-04)-	3.606e-02(3.54e-04)-	5.372e-02(2.29e-03)-	4.930e-02(2.65e-03)-	**3.466e-02(1.94e-04)**
LSMOP 3	100	7.090e-01(4.42e-02)-	5.407e-01(4.00e-02)+	4.486e-01(8.35e-02)+	**3.949e-01(3.28e-02)+**	6.330e-01(5.24e-02)
	300	6.169e+00(1.12e+00)-	**6.536e-01(5.05e-02)+**	6.572e-01(1.65e-01)=	7.359e-01(1.15e-01)=	7.206e-01(4.17e-02)
	500	7.667e+00(8.28e-01)-	7.352e-01(5.50e-02)=	2.503e+00(4.99e-01)-	8.607e-01(1.17e-16)-	**7.335e-01(4.09e-02)**
	1000	9.688e+00(4.26e-01)-	8.090e-01(3.92e-02)-	6.454e+00(1.09e+00)-	2.829e+00(6.22e+00)-	**7.551e-01(3.06e-02)**
LSMOP 4	100	1.298e-01(1.78e-02)-	1.723e-01(1.88e-02)-	2.456e-01(3.89e-02)-	2.142e-01(6.16e-03)-	**1.071e-01(7.26e-03)**
	300	1.741e-01(4.08e-03)-	**1.355e-01(7.49e-03)=**	2.506e-01(1.12e-02)-	2.630e-01(2.60e-03)-	1.435e-01(1.30e-02)
	500	1.356e-01(1.48e-03)-	1.354e-01(2.16e-03)-	1.849e-01(4.62e-03)-	1.715e-01(2.26e-03)-	**1.224e-01(6.05e-03)**
	1000	8.525e-02(5.63e-04)-	8.725e-02(2.42e-03)-	1.145e-01(1.17e-03)-	9.849e-02(1.79e-03)-	**8.132e-02(1.49e-03)**
LSMOP 5	100	1.382e-01(8.14e-02)-	2.418e-01(1.68e-01)-	3.522e-01(3.10e-02)-	4.000e-01(1.07e-01)-	**6.003e-02(1.63e-02)**
	300	5.586e-01(2.22e-02)-	2.208e-01(5.74e-02)-	2.327e-01(1.81e-02)-	2.780e-01(2.08e-02)-	**1.494e-01(4.91e-03)**
	500	5.363e-01(2.37e-02)-	2.409e-01(1.48e-02)-	2.908e-01(2.39e-02)-	3.311e-01(2.10e-01)-	**1.987e-01(6.21e-03)**
	1000	1.139e+00(6.06e-02)-	3.458e-01(2.47e-02)-	1.245e+00(1.59e-01)-	6.788e-01(1.51e-01)-	**3.028e-01(2.71e-02)**
LSMOP 6	100	8.916e-01(1.54e-01)-	**6.507e-01(1.41e-01)=**	1.094e+00(1.27e-01)-	1.112e+00(3.10e-02)-	6.894e-01(5.66e-02)
	300	2.175e+00(4.55e-01)-	**5.722e-01(4.09e-02)+**	2.253e+00(4.54e-01)-	1.265e+00(3.93e-03)-	1.203e+00(2.55e-03)
	500	3.289e+00(7.32e-01)-	**8.799e-01(5.07e-01)=**	3.346e+00(8.02e-01)-	1.292e+00(9.15e-03)-	1.242e+00(2.03e-03)
	1000	1.554e+01(2.96e+00)-	8.399e-01(9.09e-01)+	7.131e+00(1.94e+00)-	1.347e+00(2.75e-01)-	1.267e+00(3.24e-03)
LSMOP 7	100	8.816e-01(1.30e-01)-	8.199e-01(4.86e-02)-	8.782e-01(1.84e-01)-	1.099e+00(1.41e-01)-	**5.126e-01(7.99e-03)**
	300	8.984e-01(6.44e-02)-	7.464e-01(1.11e-01)-	1.064e+00(2.78e-02)-	1.008e+00(4.73e-02)-	**6.240e-01(4.26e-02)**
	500	7.967e-01(5.33e-02)-	7.503e-01(1.05e-02)-	1.052e+00(3.03e-02)-	9.500e-01(2.41e-02)-	**6.366e-01(2.09e-02)**
	1000	8.147e-01(4.43e-02)-	6.677e-01(5.66e-02)-	1.086e+00(5.03e-03)-	9.326e-01(1.38e-01)-	**6.066e-01(2.26e-02)**
LSMOP 8	100	1.179e-01(1.02e-02)-	1.476e-01(3.49e-02)-	3.581e-01(1.08e-02)-	3.647e-01(5.39e-03)-	**4.835e-02(8.10e-03)**
	300	5.203e-01(1.06e-01)-	9.651e-02(3.46e-02)-	2.073e-01(2.08e-02)-	2.361e-01(3.20e-02)-	**6.830e-02(4.68e-03)**
	500	5.566e-01(5.34e-03)-	**8.343e-02(3.25e-03)=**	2.055e-01(8.90e-03)-	2.449e-01(2.52e-02)-	9.876e-02(2.67e-02)
	1000	5.375e-01(6.40e-03)-	1.496e-01(2.38e-02)-	3.077e-01(2.55e-02)-	2.612e-01(3.19e-02)-	**1.002e-01(2.23e-02)**
LSMOP 9	100	**2.850e-01(2.87e-01)=**	6.745e-01(1.77e-03)-	8.579e-01(2.40e-01)=	1.538e+00(1.47e-06)-	5.930e-01(3.13e-06)
	300	**4.629e-01(1.70e-01)=**	8.296e-01(1.06e-01)-	9.245e-01(4.25e-01)=	1.538e+00(4.84e-14)-	5.930e-01(3.57e-05)
	500	7.934e-01(9.43e-02)-	8.194e-01(2.17e-01)-	1.110e+00(2.64e-01)-	1.538e+00(6.33e-11)-	**5.927e-01(3.53e-04)**
	1000	2.022e+00(6.71e-01)-	1.927e+00(7.67e-01)-	1.082e+00(3.40e-01)-	1.538e+00(3.78e-07)-	**5.932e-01(4.42e-04)**
+/-/=		0/31/5	4/27/5	1/31/4	1/33/2	——

Table IV The IGD results obtained by ATLMOEA and its four competitors on 3-objective LSMOP benchmarks

Problem	*n*	MOEA/DVA	WOF	LMEA	LMOEA-DS	ATLMOEA
LSMOP1	100	**3.836e-02(1.05e-03)=**	2.035e-01(2.75e-02)-	7.698e-02(7.24e-02)-	1.691e-01(3.43e-02)-	4.612e-02(1.34e-02)
	300	**1.687e-01(3.33e-03)+**	2.031e-01(1.60e-02)+	1.611e-01(1.50e-02)+	2.970e-01(8.61e-03)-	2.392e-01(2.88e-02)
	500	**2.268e-01(3.68e-03)+**	2.347e-01(4.29e-02)+	1.107e+01(4.85e-01)-	3.339e-01(7.30e-03)-	2.991e-01(2.04e-02)
	1000	**8.121e-02(1.18e-03)+**	3.918e-01(6.08e-02)-	1.049e+01(1.85e+00)-	3.768e-01(1.31e-02)-	3.547e-01(1.36e-02)
LSMOP 2	100	1.300e-01(1.40e-03)-	1.487e-01(5.52e-03)-	9.028e-02(6.64e-02)=	8.018e-02(1.78e-03)-	**6.357e-02(2.91e-03)**
	300	8.299e-02(1.30e-03)-	1.022e-01(1.78e-03)-	7.559e-02(6.88e-03)-	**4.739e-02(1.28e-03)+**	5.613e-02(7.33e-04)
	500	6.024e-02(1.46e-03)-	7.492e-02(3.04e-03)-	6.288e-02(4.83e-03)-	**4.052e-02(1.16e-03)+**	4.390e-02(4.30e-04)
	1000	4.461e-02(1.18e-03)-	5.456e-02(1.81e-03)-	5.047e-02(1.18e-03)-	**3.375e-02(8.31e-04)+**	3.466e-02(1.94e-04)
LSMOP 3	100	5.798e-01(6.32e-02)=	**4.836e-01(2.03e-01)+**	6.147e-01(1.46e-01)=	6.407e-01(8.07e-02)=	6.330e-01(5.24e-02)
	300	2.947e+00(4.51e-01)-	**5.378e-01(1.76e-01)+**	4.736e+00(3.10e+00)-	8.027e-01(4.57e-02)-	7.206e-01(4.17e-02)
	500	3.053e+00(3.50e-01)-	**5.630e-01(1.58e-01)+**	1.316e+02(1.19e+02)-	8.320e-01(4.40e-02)-	7.335e-01(4.09e-02)
	1000	1.634e+00(7.73e-02)-	**7.548e-01(7.80e-02)=**	2.175e+02(2.51e+02)-	8.425e-01(3.74e-02)-	7.551e-01(3.06e-02)
LSMOP 4	100	1.693e-01(1.10e-02)-	1.825e-01(2.57e-02)-	1.102e-01(8.26e-03)=	2.029e-01(2.35e-02)-	**1.071e-01(7.26e-03)**
	300	1.550e-01(2.86e-03)-	2.130e-01(4.72e-03)-	1.662e-01(1.14e-01)-	1.518e-01(2.57e-02)-	**1.435e-01(1.30e-02)**
	500	1.377e-01(1.45e-03)-	1.948e-01(6.70e-03)-	2.066e-01(1.50e-03)-	**9.587e-02(1.84e-03)+**	1.224e-01(6.05e-03)
	1000	8.877e-02(2.22e-03)-	1.282e-01(2.22e-03)-	1.209e-01(4.89e-03)-	9.231e-02(1.14e-03)-	**8.132e-02(1.49e-03)**
LSMOP 5	100	**5.266e-02(3.98e-03)=**	3.614e-01(2.64e-03)-	1.294e+00(1.52e+00)-	2.790e-01(1.51e-02)-	6.003e-02(1.63e-02)
	300	4.550e-01(1.41e-02)-	3.906e-01(3.88e-02)-	4.282e+00(4.87e+00)-	2.948e-01(1.69e-02)-	**1.494e-01(4.91e-03)**
	500	5.883e-01(5.03e-03)-	4.502e-01(5.87e-02)-	1.367e+01(2.84e+00)-	3.398e-01(1.01e-02)-	**1.987e-01(6.21e-03)**
	1000	3.229e-01(2.32e-02)-	5.251e-01(7.82e-03)-	1.996e+01(6.39e-01)-	4.294e-01(2.01e-02)-	**3.028e-01(2.71e-02)**
LSMOP 6	100	1.930e+00(6.40e-01)-	9.011e-01(3.28e-02)-	1.441e+02(3.15e+02)-	6.964e-01(5.37e-02)-	**6.894e-01(5.66e-02)**
	300	2.983e+01(5.77e+00)-	1.215e+00(1.24e-03)-	1.070e+03(3.37e+03)-	**7.646e-01(2.98e-02)+**	1.203e+00(2.55e-03)
	500	5.545e+01(6.95e+00)-	1.268e+00(1.76e-03)-	1.910e+04(1.74e+04)-	**8.010e-01(6.08e-02)+**	1.242e+00(2.03e-03)
	1000	1.895e+01(1.76e+00)-	1.304e+00(1.72e-03)-	3.310e+04(1.24e+04)-	**8.378e-01(1.68e-01)+**	1.267e+00(3.24e-03)
LSMOP 7	100	8.950e-01(8.66e-02)-	7.972e-01(3.29e-02)-	2.099e+00(8.36e-01)-	7.194e-01(4.15e-02)-	**5.126e-01(7.99e-03)**
	300	1.932e+00(1.44e+00)-	8.707e-01(1.24e-02)-	1.520e+00(1.50e-01)-	8.858e-01(1.89e-02)-	**6.240e-01(4.26e-02)**
	500	1.670e+00(1.30e+00)-	8.640e-01(2.97e-03)-	2.619e+02(6.94e+02)-	8.694e-01(9.42e-03)-	**6.366e-01(2.09e-02)**
	1000	7.580e-01(7.16e-02)-	8.499e-01(5.96e-04)-	6.210e+02(7.08e+02)-	8.526e-01(1.07e-03)-	**6.066e-01(2.26e-02)**
LSMOP 8	100	9.727e-02(2.05e-03)-	3.379e-01(5.49e-02)-	1.932e-01(6.13e-02)-	3.083e-01(2.90e-02)-	**4.835e-02(8.10e-03)**
	300	2.039e-01(8.98e-03)-	2.608e-01(4.30e-02)-	2.065e-01(4.82e-02)-	1.565e-01(6.45e-02)-	**6.830e-02(4.68e-03)**
	500	3.194e-01(4.19e-03)-	2.527e-01(2.29e-02)-	3.548e-01(9.84e-02)-	1.373e-01(4.11e-02)-	**9.876e-02(2.67e-02)**
	1000	1.307e-01(2.50e-03)-	2.774e-01(4.09e-03)-	6.421e-01(4.17e-02)-	1.210e-01(6.88e-03)-	**1.002e-01(2.23e-02)**
LSMOP 9	100	**2.277e-01(5.02e-02)+**	1.145e+00(1.22e-05)-	6.706e-01(5.20e-01)-	5.969e-01(6.12e-04)=	5.930e-01(3.13e-06)
	300	7.602e-01(7.20e-02)-	1.145e+00(1.12e-05)-	4.928e+00(2.17e+00)-	**5.900e-01(1.80e-03)=**	5.930e-01(3.57e-05)
	500	1.332e+00(1.24e-01)-	1.145e+00(1.44e-05)-	9.617e+01(6.43e+01)-	**5.771e-01(1.61e-03)+**	5.927e-01(3.53e-04)
	1000	**3.756e-01(2.43e-02)+**	1.145e+00(2.18e-05)-	1.150e+02(5.00e+01)-	5.761e-01(1.31e-03)+	5.932e-01(4.42e-04)
+/-/=		5/28/3	5/30/1	1/32/3	9/24/3	——

After comparing the performance of ATLMOEA with its competitors, the final solutions of the above algorithms on tri-objective LSMOP8 with 300 decision variables are depicted in Figure 6. As observed from Figure 6, only ATLMOEA can obtain the final population to evenly cover the true PF. For LMEA and MOEA/DVA, the accuracy of decision variable grouping affects its efficiency significantly. However, there is no effective way to group decision variables correctly. Thus, LMEA and MOEA/DVA preform worse than ATLMOEA on LSMOP problems. For LMOCSO and DGEA, their search operators are specifically designed for LMOPs, but the final solutions obtained by LMOCSO and DGEA are inferior to ATLMOEA. It is attributed to the fact that their search operators are not efficient enough. In addition, MOEA/PSL utilizes two unsupervised neural networks to reduce the dimensionality of decision variables, which will result in the loss of potentially valid information in the discarded search space. For WOF and FDV, they may waste some computational resources, as they cannot adaptively judge the switch of two evolutionary stages. For LMOEA-DS, it selects a set of solutions in each generation to guide the evolutionary search, but when this solution is not properly selected, the entire evolutionary process will go astray. In contrast, our proposed algorithm ATLMOEA can avoid the above shortcomings by using an adaptive two-stage strategy and designing two improved optimizers for different stages.

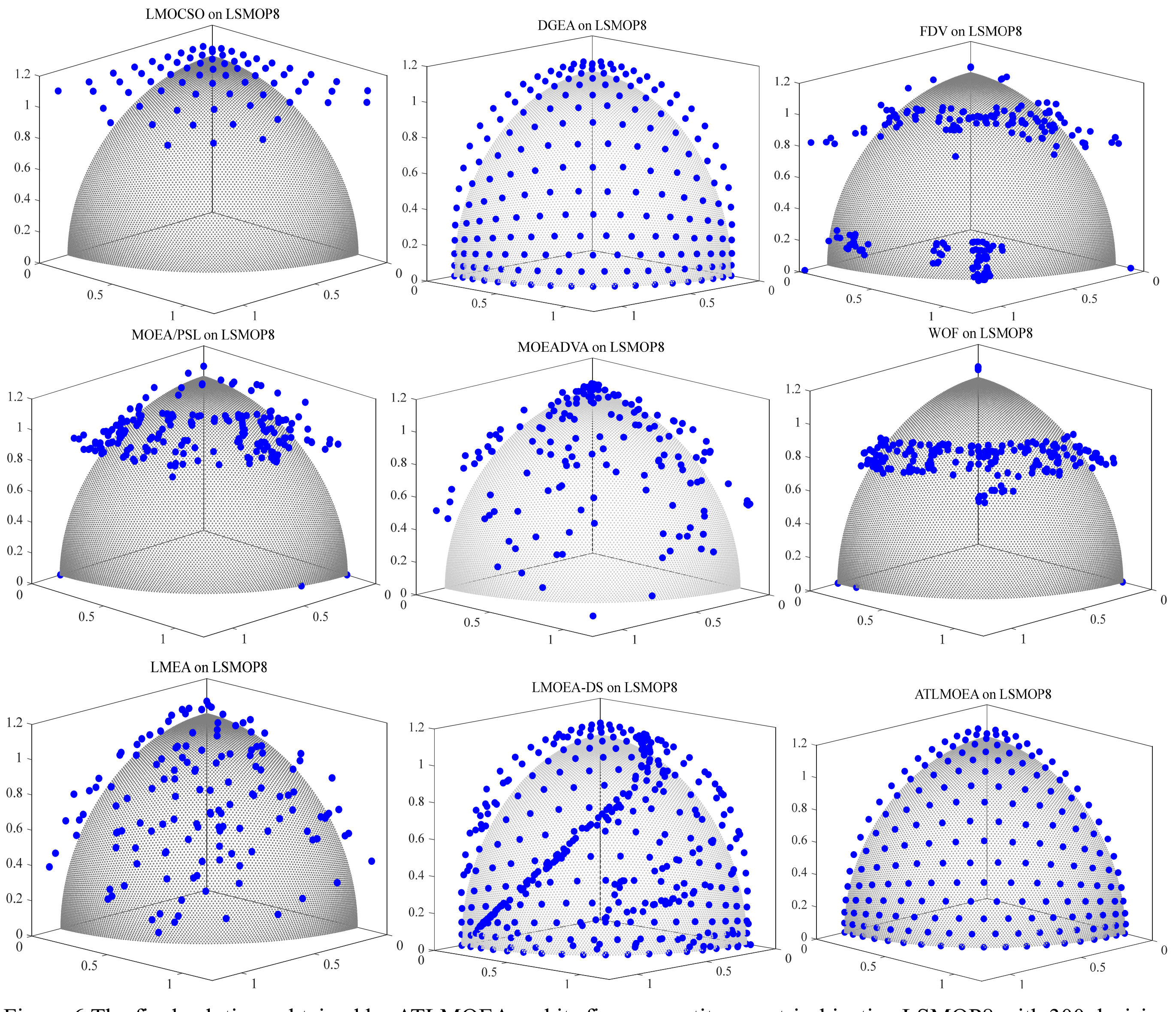


Figure 6 The final solutions obtained by ATLMOEA and its five competitors on tri-objective LSMOP8 with 300 decision variables, with the true PF indicated by the shaded area.

Table V The IGD results obtained by ATLMOEA and its four competitors on IMF benchmarks

Problem	*n*	LMOCSO	DGEA	FDV	MOEA/PSL	ATLMOEA
IMF1 (m=2)	100	**4.422E-3(1.93E-4)=**	6.263e-03(4.25e-04)-	1.160E-1(1.06E-2)-	7.623E-1(1.26E-1)-	4.490E-3(3.53E-4)
	300	1.646E-1(2.31E-2)-	2.438e-01(2.67e-02)-	1.308E-1(1.48E-2)-	7.403E-1(1.29E-1)-	**1.448E-2(1.68E-3)**
	500	1.733E-1(1.99E-2)-	2.757e-01(2.30e-02)-	1.193E-1(1.04E-2)-	6.414E-1(1.05E-1)-	**1.071E-2(1.15E-3)**
	1000	1.606E+0(7.57E-2)-	1.063e+00(6.98e-02)-	1.291E-1(6.29E-3)-	7.158E-1(1.31E-1)-	**7.146E-2(5.24E-3)**
IMF2 (m=2)	100	**5.829E-3(7.03E-4)+**	6.680e-03(4.42e-04)+	2.047E-1(8.09E-3)+	6.095E-1(0.00E+0)-	6.095E-1(4.57E-6)
	300	**1.870E-1(4.76E-2)+**	3.540e-01(3.32e-02)+	2.305E-1(9.13E-3)+	6.095E-1(0.00E+0)-	6.061E-1(1.06E-2)
	500	**2.099E-1(3.63E-2)+**	3.692e-01(3.20e-02)+	2.279E-1(8.51E-3)+	6.095E-1(0.00E+0)-	6.063E-1(4.98E-3)
	1000	2.378E+0(1.39E-1)-	1.597e+00(1.23e-01)-	**2.364E-1(9.49E-3)+**	6.095E-1(0.00E+0)-	5.980E-1(1.10E-2)
IMF3 (m=2)	100	3.495E-3(1.81E-4)=	3.838e-03(7.60e-04)=	5.633E-3(5.78E-4)-	2.320E-2(2.31E-3)-	**3.463E-3(7.72E-5)**
	300	3.458E-1(1.71E-2)-	6.649e-01(8.69e-02)-	2.714E-2(5.41E-3)-	2.535E-2(4.85E-4)-	**5.053E-3(2.28E-4)**
	500	3.510E-1(1.51E-2)-	6.034e-01(6.31e-02)-	1.803E-2(4.59E-3)-	2.537E-2(4.01E-4)-	**4.461E-3(7.08E-4)**
	1000	1.860E+0(1.28E-1)-	1.880e+00(1.06e-01)-	8.741E-2(5.83E-3)-	2.564E-2(1.67E-4)-	**6.040E-3(3.98E-4)**
IMF4 (m=3)	100	5.115E-2(2.63E-3)-	4.094e-01(3.33e-02)-	4.897E-1(6.30E-3)-	5.408E-1(4.13E-5)-	**4.721E-2(1.63E-3)**
	300	4.366E+1(5.42E+0)-	3.757e+01(3.49e+01)-	5.347E-1(1.52E-2)-	5.408E-1(9.15E-6)-	**4.324E-1(1.10E-2)**
	500	7.211E+1(7.35E+0)-	2.338e+01(4.36e+01)-	5.408E-1(1.49E-4)-	5.544E-1(4.32E-2)-	**4.527E-1(1.76E-2)**
	1000	4.804E+2(2.30E+1)-	1.210e+02(1.34e+02)=	4.845E+0(1.28E+0)-	**5.473E-1(2.07E-2)+**	2.880E+0(3.65E+0)
IMF5 (m=2)	100	1.262E-2(2.43E-3)-	1.456e-02(9.02e-04)-	3.358E-2(7.93E-3)-	3.810E-1(2.57E-1)-	**6.489E-3(1.20E-3)**
	300	1.605E-1(1.28E-3)-	3.796e-02(1.50e-03)-	5.231E-2(9.53E-3)-	2.174E-1(1.96E-1)-	**2.573E-2(3.25E-3)**
	500	1.597E-1(2.54E-3)-	3.939e-02(1.27e-03)-	4.872E-2(5.68E-3)-	4.561E-1(2.98E-1)-	**2.525E-2(2.32E-3)**
	1000	1.864E-1(1.36E-3)-	4.494e-02(4.13e-04)-	8.721E-2(4.62E-3)-	4.802E-1(2.49E-1)-	**3.997E-2(3.09E-3)**
IMF6 (m=2)	100	1.639E-2(2.55E-3)-	3.446e-02(3.18e-03)-	7.110E-2(1.67E-2)-	4.554E-1(2.48E-1)-	**1.197E-2(1.01E-3)**
	300	2.295E-1(5.55E-3)-	7.503e-02(7.67e-03)-	1.182E-1(1.23E-2)-	4.839E-1(2.05E-1)-	**4.183E-2(6.34E-3)**
	500	2.283E-1(8.32E-3)-	7.667e-02(3.56e-03)-	1.120E-1(1.01E-2)-	4.997E-1(1.83E-1)-	**4.548E-2(1.21E-2)**
	1000	2.707E-1(2.75E-3)-	8.725e-02(7.56e-03)-	1.703E-1(8.79E-3)-	4.337E-1(1.95E-1)-	**7.907E-2(3.35E-2)**
IMF7 (m=2)	100	3.250E-2(2.38E-2)-	6.197e-03(1.39e-03)-	5.502E-3(5.75E-4)-	2.387E-2(1.81E-3)-	**4.451E-3(4.39E-4)**
	300	2.529E-1(4.45E-3)-	**9.784e-03(1.37e-03)=**	3.467E-2(2.47E-2)=	2.529E-2(4.50E-4)=	2.204E-2(2.23E-2)
	500	2.430E-1(4.22E-3)-	**1.027e-02(1.57e-03)+**	5.161E-2(2.75E-2)+	2.560E-2(2.38E-4)+	1.896E-1(3.25E-2)
	1000	2.796E-1(2.84E-3)-	**1.739e-02(6.66e-03)+**	2.120E-1(1.21E-2)+	2.602E-2(1.35E-4)+	2.409E-1(2.56E-2)
IMF8 (m=3)	100	1.203E-1(3.11E-2)-	3.414e-01(2.56e-03)-	3.484E-1(1.81E-2)-	3.609E-1(6.45E-2)-	**6.502E-2(4.73E-3)**
	300	1.013E+0(4.25E-2)-	4.156e-01(1.84e-01)+	**3.195E-1(1.83E-3)+**	6.444E-1(2.40E-1)-	4.515E-1(2.53E-1)
	500	1.063E+0(4.41E-2)-	3.588e-01(1.15e-07)=	**3.218E-1(1.25E-3)+**	6.123E-1(2.25E-1)=	5.930E-1(3.56E-1)
	1000	4.241E+0(3.46E-1)=	**4.760e-01(2.47e-01)+**	2.451E+0(1.08E+0)=	7.563E-1(2.64E-1)=	2.928E+0(2.01E+0)
IMF9 (m=2)	100	6.101E-2(6.59E-2)-	4.714e-02(1.87e-02)-	1.541E-2(1.28E-3)-	1.665E-1(2.92E-2)-	**4.926E-3(2.84E-4)**
	300	3.153E-1(2.04E-2)-	3.504e-01(1.75e-02)-	3.391E-2(1.63E-3)-	2.995E-1(1.92E-2)-	**1.002E-2(5.84E-4)**
	500	3.462E-1(2.91E-2)-	3.595e-01(2.14e-02)-	3.433E-2(2.56E-3)-	3.589E-1(4.78E-2)-	**8.688E-3(7.49E-4)**
	1000	6.797E-1(6.78E-3)-	6.546e-01(2.37e-02)-	6.300E-2(1.75E-3)-	4.687E-1(5.86E-3)-	**1.817E-2(1.38E-3)**
+/−/=		3/30/3	7/25/4	8/26/2	3/30/3	——

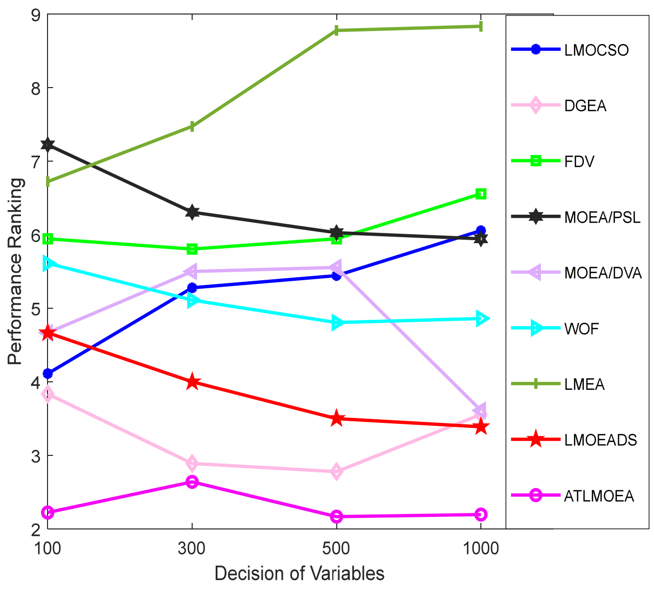
Table VI The IGD results obtained by ATLMOEA and its four competitors on IMF benchmarks

Problem	*n*	MOEA/DVA	WOF	LMEA	LMOEA-DS	ATLMOEA
IMF1 (m=2)	100	5.635e-03(4.87e-04)-	3.397E-1(1.41E-2)-	1.041E-2(1.63E-3)-	8.976e-02(1.41e-02)-	**4.490E-3(3.53E-4)**
	300	8.691e-02(1.89e-03)-	4.014E-1(2.23E-2)-	1.438E+1(9.44E-1)-	2.492e-01(1.25e-02)-	**1.448E-2(1.68E-3)**
	500	3.566e-02(4.55e-04)-	3.750E-1(2.28E-2)-	1.541E+1(5.76E-1)-	2.659e-01(9.48e-03)-	**1.071E-2(1.15E-3)**
	1000	**1.006e-02(4.63e-05)+**	4.293E-1(2.15E-2)-	1.575E+1(7.53E-1)-	2.905e-01(6.77e-03)-	7.146E-2(5.24E-3)
IMF2 (m=2)	100	**6.032e-03(4.79e-04)+**	6.095E-1(0.00E+0)-	9.442E-3(1.16E-3)+	6.095e-01(1.49e-08)=	6.095E-1(4.57E-10)
	300	**1.572e-01(5.19e-03)+**	6.095E-1(0.00E+0)-	1.892E+1(9.55E-1)-	6.095e-01(8.47e-07)=	6.061E-1(1.06E-2)
	500	**6.609e-02(1.03e-03)+**	6.095E-1(0.00E+0)-	1.928E+1(6.59E-1)-	6.095e-01(6.05e-08)-	6.063E-1(4.98E-3)
	1000	**1.811e-02(2.14e-04)+**	6.095E-1(0.00E+0)-	2.005E+1(5.61E-1)-	6.095e-01(1.07e-07)-	5.980E-1(1.10E-2)
IMF3 (m=2)	100	3.087e-02(7.07e-03)-	4.772E-3(1.06E-4)-	1.292E-2(6.68E-3)-	5.219e-03(1.62e-04)-	**3.463E-3(7.72E-5)**
	300	1.926e-01(9.74e-03)-	5.323E-3(1.23E-4)-	1.889E+1(9.57E-1)-	6.492e-03(3.68e-04)-	**5.053E-3(2.28E-4)**
	500	9.247e-02(3.66e-03)-	5.374E-3(1.84E-4)-	1.940E+1(5.74E-1)-	6.852e-03(1.83e-04)-	**4.461E-3(7.08E-4)**
	1000	3.595e-02(5.82e-04)-	6.575E-3(2.49E-4)-	1.998E+1(7.11E-1)-	7.403e-03(1.59e-04)-	**6.040E-3(3.98E-4)**
IMF4 (m=3)	100	3.468e-01(6.99e-02)-	5.407E-1(1.01E-4)-	8.884E-2(7.22E-3)-	2.613e-01(5.33e-02)-	**4.721E-2(1.63E-3)**
	300	1.455e+01(1.29e+00)-	5.408E-1(6.98E-6)-	5.847E+2(3.92E+1)-	5.407e-01(9.51e-05)-	**4.324E-1(1.10E-2)**
	500	1.066e+01(3.50e-01)-	5.408E-1(3.81E-6)-	1.036E+3(4.35E+1)-	5.407e-01(8.38e-05)-	**4.527E-1(1.76E-2)**
	1000	6.304e+00(1.55e-01)-	**5.408E-1(3.74E-6)+**	2.178E+3(2.25E+1)-	5.408e-01(2.32e-05)+	2.880E+0(3.65E+0)
IMF5 (m=2)	100	**4.414e-03(1.76e-04)+**	1.612E-2(5.88E-4)-	9.334E-3(1.52E-3)-	3.465e-02(2.25e-03)-	6.489E-3(1.20E-3)
	300	3.317e-02(3.62e-04)-	2.983E-2(1.15E-3)-	3.343E-1(7.39E-3)-	4.898e-02(1.14e-03)-	**2.573E-2(3.25E-3)**
	500	**2.210e-02(2.16e-04)+**	3.207E-2(2.00E-3)-	3.376E-1(6.36E-3)-	5.008e-02(1.28e-03)-	2.525E-2(2.32E-3)
	1000	**1.321e-02(8.17e-05)+**	4.435E-2(3.44E-3)-	3.433E-1(3.90E-3)-	5.109e-02(1.36e-03)-	3.997E-2(3.09E-3)
IMF6 (m=2)	100	**4.756e-03(1.92e-04)+**	2.538E-2(3.36E-3)-	1.360E-2(7.60E-3)=	5.455e-02(3.04e-03)-	1.197E-2(1.01E-3)
	300	5.480e-02(7.97e-04)-	5.261E-2(3.11E-3)-	5.842E-1(2.28E-2)-	7.381e-02(2.42e-03)-	**4.183E-2(6.34E-3)**
	500	**3.876e-02(4.86e-04)=**	6.971E-2(4.55E-3)-	5.972E-1(1.23E-2)-	7.425e-02(1.03e-03)-	4.548E-2(1.21E-2)
	1000	**2.464e-02(1.69e-04)+**	7.206E-2(4.19E-3)=	6.130E-1(5.25E-3)-	7.572e-02(8.29e-04)=	7.907E-2(3.35E-2)
IMF7 (m=2)	100	2.710e-02(1.84e-03)-	4.816E-3(1.46E-4)=	3.528E-2(1.23E-2)-	1.453e-02(9.23e-04)-	**4.451E-3(4.39E-4)**
	300	1.290e-01(1.86e-03)-	**5.294E-3(1.96E-4)+**	5.015E-1(1.77E-2)-	1.832e-02(1.58e-03)=	2.204E-2(2.23E-2)
	500	1.102e-01(2.21e-03)+	**5.333E-3(1.90E-4)+**	5.194E-1(9.34E-3)-	1.808e-02(1.20e-03)+	1.896E-1(3.25E-2)
	1000	7.793e-02(1.56e-03)+	**6.146E-3(1.51E-4)+**	5.237E-1(1.16E-2)-	2.036e-02(2.01e-03)+	2.409E-1(2.56E-2)
IMF8 (m=3)	100	2.047e-01(1.38e-02)-	3.570E-1(6.97E-5)-	2.900E-1(5.91E-2)-	3.564e-01(3.76e-04)-	**6.502E-2(4.73E-3)**
	300	9.424e-01(2.25e-02)-	3.570E-1(5.77E-6)+	1.434E+1(6.83E-1)-	**3.569e-01(1.68e-04)+**	4.515E-1(2.53E-1)
	500	8.320e-01(1.11e-02)=	3.571E-1(5.97E-6)=	2.547E+1(6.78E-1)-	**3.569e-01(1.90e-04)=**	5.930E-1(3.56E-1)
	1000	6.689e-01(4.98e-03)=	**3.571E-1(5.45E-6)+**	5.145E+1(1.35E+0)-	3.570e-01(2.18e-05)+	2.928E+0(2.01E+0)
IMF9 (m=2)	100	8.884e-02(4.57e-02)-	1.396E-2(7.73E-4)-	9.149E-3(1.37E-3)-	1.230e-02(6.02e-04)-	**4.926E-3(2.84E-4)**
	300	1.281e-01(4.93e-03)-	4.272E-2(3.24E-2)-	6.429E-1(2.86E-3)-	1.777e-02(1.34e-03)-	**1.002E-2(5.84E-4)**
	500	7.432e-02(1.64e-03)-	3.811E-2(1.55E-2)-	7.198E-1(5.76E-3)-	1.850e-02(1.16e-03)-	**8.688E-3(7.49E-4)**
	1000	3.436e-02(5.31e-04)-	8.045E-2(3.05E-2)-	9.116E-1(1.02E-2)-	2.294e-02(1.28e-03)-	**1.817E-2(1.38E-3)**
+/−/=		12/21/3	6/27/3	1/34/1	5/26/5	——

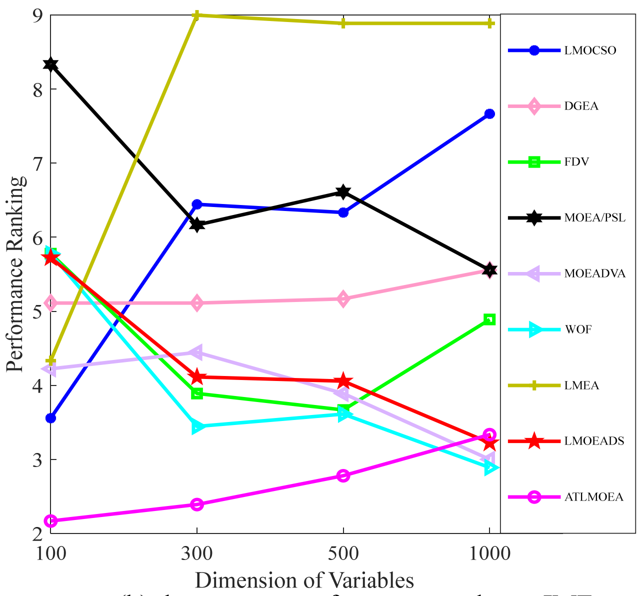
4.4.2. Comparison Results on IMF Problems

To further validate the performance of ATLMOEA, ATLMOEA is compared with eight competitive

LMOEAs (i.e., MOEA/DVA, LMOCSO, DGEA, WOF, LMEA, LMOEA-DS, FDV, and MOEA/PSL) for handling IMF1 to IMF9 in the cases of *n* = {100, 300, 500, 1000}. The experimental results obtained by ATLMOEA and the above competitors on these problems are presented in Table V and Table VI, in which the best result for each problem is highlighted in bold. As observed from Table V and Table VI, ATLMOEA gets 18 out of 36 best results, LMOCSO obtains 1 best result, FDV gains 2 best results, MOEA/DVA gets 10 best results, WOF gets 5 best results, while DGEA, MOEA/PSL, LMEA and LMOEADS doesn't obtain any best result. To be specific, due to the problem characteristics, such as irregular Pareto front, nonlinear variable linkage, multimodal landscape and mixed function formulation, the IMF test suite has a great challenge to the existing LMOEAs. It is worth noting that the IGD results achieved by ATLMOEA on IMF7 and IMF8 are relatively worse. This is attributed to the fact that both IMF7 and IMF8 are too tough to be solved, which have concave Pareto front and nonlinear variable linkage. Observing the statistic results presented in the last row of Table V and Table VI, ATLMOEA outperforms MOEA/DVA, LMOCSO, DGEA, WOF, LMEA, LMOEA-DS, FDV, and MOEA/PSL in 21, 30, 25, 27, 34, 26, 26 and 30 out of 36 IMF cases, respectively. Therefore, we can reasonably conclude that ATLMOEA performs better than the above competitors in solving most of these IMF problems.



(a) the average performance ranks on LSMOP



(b) the average performance ranks on IMF

Figure 7 Illustration of the average performance ranks over all test problems for different dimensions of variables.

4.4.3. Average Performance Ranks

To quantify how well each optimizer performs, Friedman's test with the KEEL is used to rank ATLMOEA and its compared algorithms on the LSMOP and IMF problems with different numbers of variables (*n*). The average performance ranks for all cases with *n* ranging from 100 to 1000 are presented in Figure 7, where a lower value indicates that the corresponding optimizer performs better. For ease of observation, the rank values of ATLMOEA for the LSMOP and IMF problems are connected by a pink line in Figure 7, respectively. Obviously, ATLMOEA shows overwhelming advantages over other compared algorithms overall. It is noted that ATLMOEA gains lower rank values than that obtained by any of its competitors except when n=1000 on IMF problems. This may be because ATLMOEA is mainly driven by the layer-based CSO in the second stage, but the evolutionary search of CSO needs more function evaluations with the increase of n. As observed from Figure 7(b), it can be found that the ranking gap between MOEA/DVA and LMEA is large though they both take advantage of the decision variable analysis. The reason behind this may be the difference between their grouping strategies. The grouping strategy directly determines the performance of LMEA and MOEA/DVA. In addition, WOF and LMOEA-DS show better performance as the number of variables increases. This is attributed to the fact that both WOF and LMOEA-DS are designed based on the problem transformation, in which the original LMOP is transformed and searched in its small-scale space. Thus, the performance of WOF and LMOEA-DS will be significantly improved with the increase of decision variables. In general, although ATLMOEA cannot completely suppress other algorithms on LSMOP and IMF problems, ATLMOEA has remedied the defects of the above algorithms to a large extent with an adaptive two-stage strategy and two improved optimizers.

Table VII The IGD results obtained by ATLMOEA and three versions on 2-objective LSMOP benchmarks

Problem	*n*	ATLMOEA1	ATLMOEA2	ATLMOEA3	ATLMOEA
LSMOP1	100	2.369E-2(6.01E-2)-	1.224E-2(6.09E-3)-	2.113E-1(2.98E-2)-	**4.097E-3(2.34E-4)**
	300	1.207E-1(1.81E-2)-	1.542E-1(1.23E-2)-	2.118E-1(2.62E-2)-	**5.285E-2(2.07E-2)**
	500	1.495E-1(7.71E-3)-	1.679E-1(8.70E-3)-	2.401E-1(2.53E-2)-	**1.337E-1(6.98E-3)**
	1000	1.825E-1(1.03E-2)-	2.414E-1(1.10E-1)-	3.283E-1(2.76E-2)-	**1.712E-1(7.24E-3)**
LSMOP2	100	1.499E-2(3.40E-3)-	2.370E-2(4.67E-3)-	5.540E-2(1.22E-2)-	**1.099E-2(1.59E-3)**
	300	1.466E-2(3.16E-3)=	1.335E-2(4.67E-4)=	1.614E-2(6.32E-4)-	**1.422E-2(3.21E-3)**
	500	1.133E-2(2.63E-3)=	9.629E-3(1.93E-4)=	1.176E-2(4.40E-4)-	**1.018E-2(1.02E-3)**
	1000	8.379E-3(1.37E-3)=	7.786E-3(1.72E-3)=	8.792E-3(1.77E-3)=	**7.836E-3(1.48E-3)**
LSMOP3	100	6.799E-1(7.72E-2)-	7.110E-1(5.46E-2)-	8.118E-1(6.77E-2)-	**5.732E-1(7.41E-2)**
	300	1.060E+0(1.37E-1)=	1.069E+0(9.62E-2)=	1.239E+0(2.21E-1)=	**1.039E+0(1.16E-1)**
	500	1.265E+0(8.73E-2)=	**1.231E+0(1.89E-1)=**	1.399E+0(1.77E-1)=	1.275E+0(1.52E-1)
	1000	**1.436E+0(1.34E-1)=**	1.486E+0(6.31E-2)=	1.568E+0(1.97E-2)-	1.454E+0(1.13E-1)
LSMOP4	100	7.609E-3(1.02E-3)=	1.290E-2(1.08E-3)-	2.271E-2(1.54E-3)-	**6.972E-3(9.50E-4)**
	300	9.138E-3(1.92E-4)-	1.114E-2(1.11E-3)-	2.660E-2(2.94E-3)-	**8.712E-3(2.27E-4)**
	500	8.779E-3(3.53E-4)-	1.209E-2(2.08E-3)-	2.552E-2(1.63E-3)-	**8.757E-3(1.31E-3)**
	1000	1.262E-2(1.04E-3)=	1.366E-2(6.71E-4)-	1.858E-2(4.58E-4)-	**1.250E-2(8.82E-4)**
LSMOP5	100	4.331E-3(6.90E-5)=	4.802E-3(2.13E-4)-	3.586E-2(1.13E-2)-	**4.303E-3(6.21E-5)**
	300	1.647E-2(2.08E-3)-	3.494E-2(1.19E-2)-	3.486E-1(5.99E-2)-	**1.455E-2(1.44E-3)**
	500	9.748E-2(7.99E-3)-	1.416E-1(1.50E-2)-	6.457E-1(1.77E-2)-	**8.289E-2(1.44E-2)**
	1000	4.959E-1(5.07E-2)-	5.756E-1(5.10E-2)-	6.614E-1(1.77E-2)-	**4.274E-1(6.64E-2)**
LSMOP6	100	3.604E-1(3.92E-3)-	3.689E-1(6.75E-3)-	4.095E-1(3.10E-2)-	**3.367E-1(2.25E-2)**
	300	**4.311E-1(3.81E-3)+**	4.347E-1(1.16E-2)+	4.420E-1(1.20E-2)=	4.444E-1(1.11E-2)
	500	4.271E-1(5.74E-3)=	4.229E-1(3.97E-3)=	4.311E-1(1.41E-2)=	**4.089E-1(5.58E-2)**
	1000	4.080E-1(3.96E-3)=	4.042E-1(2.13E-3)=	4.057E-1(1.40E-3)=	**3.866E-1(6.55E-2)**
LSMOP7	100	8.964E-1(2.60E-2)-	9.683E-1(8.50E-2)-	1.360E+0(1.62E-1)-	**8.554E-1(4.51E-2)**
	300	1.493E+0(8.56E-3)=	**1.483E+0(4.19E-2)=**	1.498E+0(2.38E-3)=	1.483E+0(3.41E-2)
	500	**1.433E+0(1.55E-1)=**	1.507E+0(1.84E-3)=	1.510E+0(1.09E-3)-	1.476E+0(9.41E-2)
	1000	**1.514E+0(1.52E-3)=**	1.516E+0(2.77E-3)=	1.516E+0(3.02E-3)=	1.514E+0(1.71E-3)
LSMOP8	100	**4.747E-3(4.49E-4)=**	8.135E-3(2.87E-3)-	5.179E-2(7.02E-3)-	4.753E-3(5.44E-4)
	300	2.147E-2(1.79E-3)-	2.271E-2(9.79E-4)-	4.186E-2(4.38E-3)-	**1.897E-2(1.62E-3)**
	500	1.953E-2(9.44E-4)=	2.195E-2(9.05E-4)-	7.673E-2(1.58E-2)-	**1.893E-2(6.65E-4)**
	1000	4.389E-2(9.21E-3)-	5.728E-2(1.18E-2)-	3.857E-1(7.83E-2)-	**3.768E-2(4.39E-3)**
LSMOP9	100	8.101E-1(1.17E-16)=	7.797E-1(9.60E-2)=	**7.445E-1(1.38E-1)=**	8.107E-1(1.86E-3)
	300	8.101E-1(5.48E-16)=	8.100E-1(1.19E-4)=	7.982E-1(4.18E-2)-	**7.828E-1(8.57E-2)**
	500	8.092E-1(6.41E-4)=	**8.084E-1(1.01E-3)=**	8.136E-1(1.35E-2)=	8.089E-1(5.23E-4)
	1000	8.154E-1(1.07E-2)=	8.051E-1(2.58E-2)=	8.271E-1(2.93E-2)-	**7.233E-1(1.16E-1)**
+/−/=		1/15/20	1/20/15	0/26/10	——

4.5. The effectiveness of our adaptive two-stage strategy

As discussed above, the general performance of ATLMOEA has been verified when compared with several competitive LMOEAs. In this section, the effectiveness of the adaptive two-stage strategy, as one of the contributions of this paper, will be validated by allocating different computational resources to the first stage, on the premise that FE_{max} is set to 10^6. To be specific, three variants, named as ATLMOEA1, ATLMOEA2 and ATLMOEA3 are compared with ATLMOEA, which will allocate 50000, 150000 and 500000 evaluation times for the first stage, respectively. The detailed average IGD results obtained by ATLMOEA and its compared variants on solving bi-objective LSMOP problems with n = {100, 300, 500, 1000} are presented in Table VII. As observed from Table VII, ATLMOEA gets 27 out of 36 best results, ATLMOEA1 obtains 5 best results, ATLMOEA2 gains 3 best results and ATLMOEA3 gets only 1 best result. Obviously, when comparing with the variants using fixed computational resources in the first stage, ATLMOEA that can adaptively adjust the evolutionary stage shows absolute advantages over other compared variants. For solving different test problems, the number of function evaluations to be allocated in the first stage is also different. If too many computational resources are wasted in the first stage, less computation resources will be allocated to the second stage, which will then affect the diversity of the whole population.

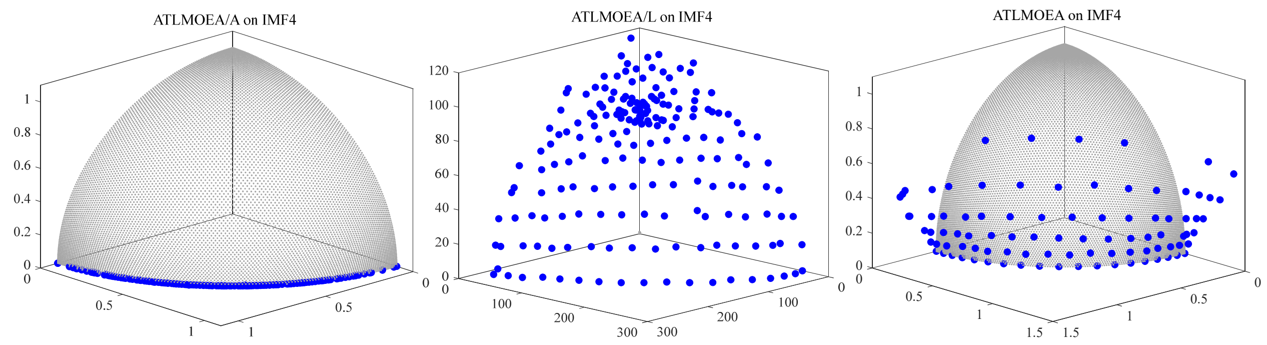


Figure 8 The final solutions obtained by ATLMOEA and its two competitors on tri-objective IMF4 with 500 decision variables, with the true PF indicated by the shaded area.

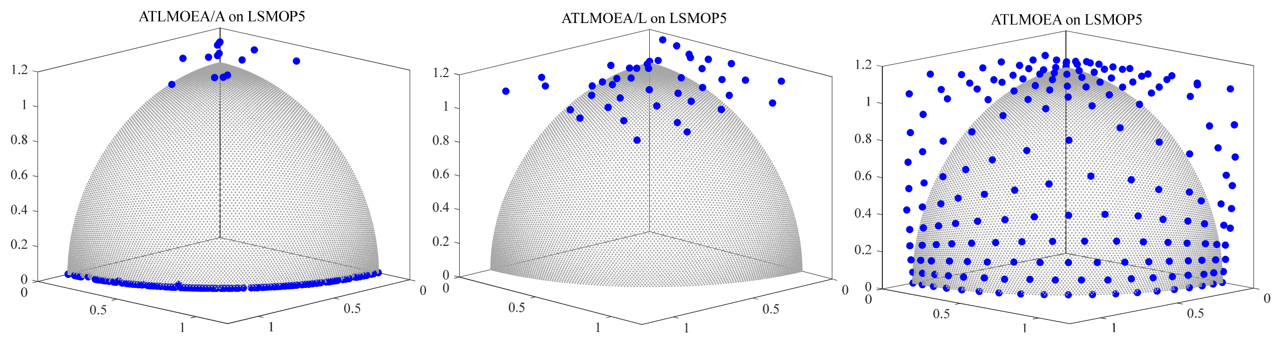


Figure 9 The final solutions obtained by ATLMOEA and its two competitors on tri-objective LSMOP5 with 300 decision variables, with the true PF indicated by the shaded area.

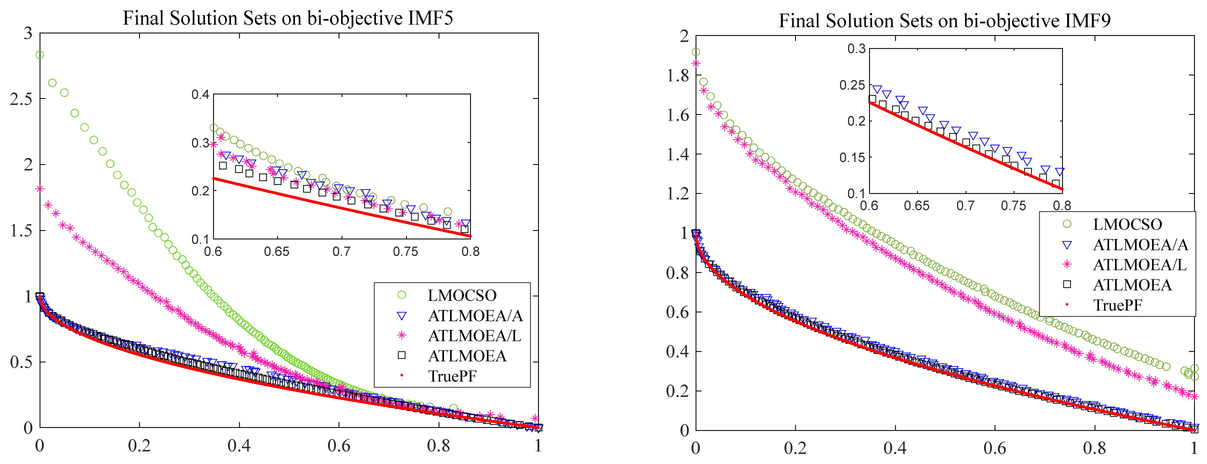
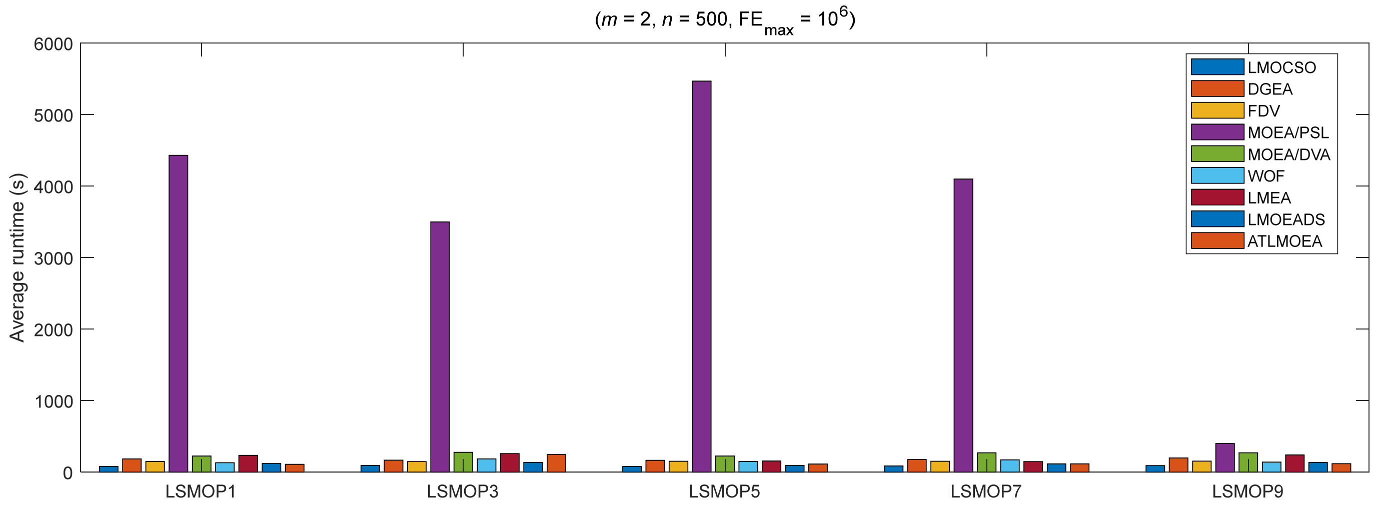
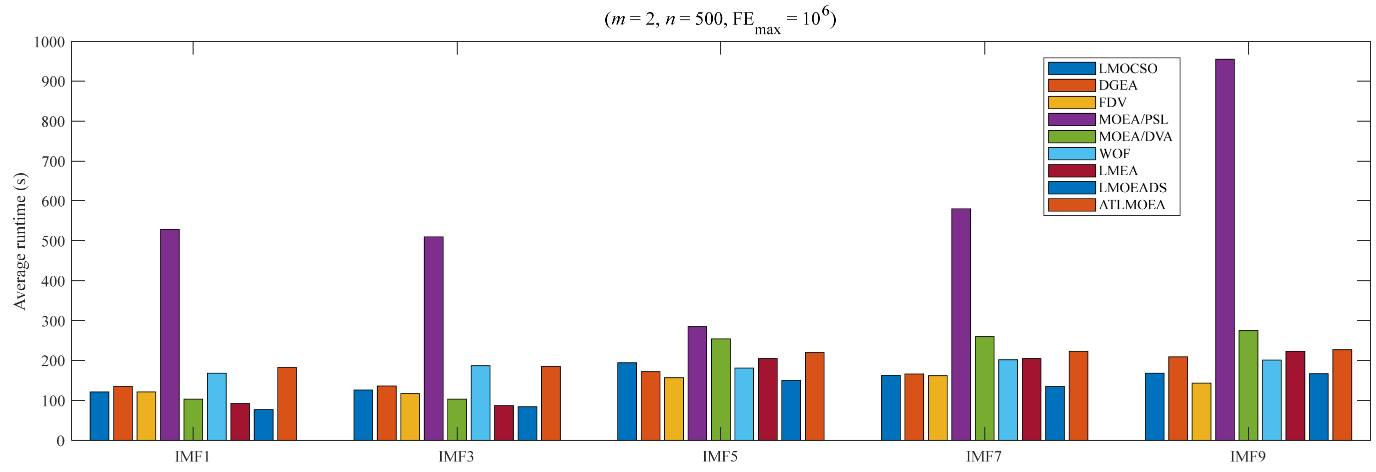


Figure 10 The final solutions obtained by TLMOEA and its three competitors on bi-objective IMF5 and bi-objective IMF9 with 300 decision variables.

Moreover, to investigate the performance of the two optimizers (i.e., accelerating-optimizer and layered competitive-swarm-optimizer), ATLMOEA is further compared with two variants by only running one optimizer. Specifically, one variant that utilizes the accelerating-optimizer is named as ATLMOEA/A, while the other one that runs the layered-competitive-swarm-optimizer is named as ATLMOEA/L. Their final solutions obtained for solving tri-objective IMF4 with n = 500, tri-objective LSMOP5 with n = 300, bi-objective IMF5 with n = 500 and bi-objective IMF9 with n = 500 are depicted in Figures 8-10, respectively. As observed from Figures 8-10, the performance of ATLMOEA is much better than ATLMOEA/A and ATLMOEA/L. For the accelerating-optimizer, its advantage is to speed up the convergence by quickly finding the quasi-optimal solutions, which is realized by training a simple ANN to learn the potential promising improvement directions. For the layered competitive swarm optimizer, it focuses more on the population's diversity by spreading solutions over the true Pareto front, which can gain more diversified evolutionary directions. Therefore, two new optimizers are essential for ATLMOEA. Based on the characteristic of the target LMOPs, the adaptive two-stage strategy adaptively adjusts the proportion of the first stage, which can reasonably allocate the computation resources to obtain the better overall optimization performance.



(a)



(b)

Figure 11 The average actual runtime of ATLMOEA and its competitors on (a) LSMOP and (b) IMF problems.

4.6. Computational Complexity Analysis of ATLMOEA

In our proposed ATLMOEA, two optimizers (i.e., accelerating optimizer and layered competitive swarm optimizer) are designed for ensuring the convergence and diversity of population. In the first stage, the computation complexity of the accelerating optimizer in one generation is mainly dominated by three major procedures (i.e., the training of $\mathbf{M}_{NN}$, the reproduction and the environmental selection). In the second stage, the computation complexity of the layered competitive swarm optimizer in one generation is mainly dominated by two major procedures (i.e., the reproduction and the environmental selection).

To be specific, to gain a trained $\mathbf{M}_{NN}$ in **Algorithm 1** requires a time complexity of $O(mN^2 + SnN)$ where m represents the number of objectives, N is the population size, n is number of decision variables and S represents the number of nodes in the hidden layer of MNN. To generate N new offspring, accelerating optimizer and layered competitive swarm optimizer require a time complexity of $O(nN)$ and $O(nN/2)$, respectively. The Pareto-based environmental selection in accelerating optimizer and the Reference-Vector-Guided environmental selection in layered competitive swarm optimizer require the same time complexity of $O(mN^2)$. Thus, the overall time complexity of ATLMOEA in one generation is $O(t(2mN^2 + (S+1)nN)+(1- t)(mN^2 + 1.5nN))$. where t represents the computational resources occupied by each optimizer in the whole stage.

To further evaluate the actual runtime of ATLMOEA and its competitors, their average runtimes of each algorithm in 20 times are shown in Figure 11 on bi-objective LSMOP1, LSMOP3, LSMOP5, LSMOP 7 and LSMOP9 problems and bi-objective IMF1, IMF3, IMF5, IMF7 and IMF9 problems with 500 variables. As observed from Figure 11, the actual runtime of ATLMOEA is similar to most comparison algorithms on LSMOP problems and more than that of LMOCSO, DGEA, FDV, LMEA and LMOEA-DS on IMF problems. Therefore, in terms of running time, ATLMOEA has not clear superiority over the existing LMOEAs. It is attributed to the fact that we trained a network in the first stage. Although it does not consume function evaluations, it wastes computation time and affect the efficiency of ATLMOEA. If ATLMOEA deals with more decision variables (e.g., 10000 decision variables), ATLMOEA may obviously require more than 10^6 evaluation function to get good results. Therefore, the improved operation to improve the efficiency of the algorithm should be considered in the second stage.

5. Conclusions and future work

In this paper, an evolutionary algorithm with an adaptive two-stage search strategy has been proposed for large-scale multi-objective optimization, called ATLMOEA. In the proposed optimizer, the two evolutionary stages are adaptively adjusted according to the characteristic of LMOPs. Specifically, the neural network-based accelerating optimizer is used in the first stage to accelerate the convergence speed, while the layer-based CSO optimizer is run in the second stage to diversify the quasi-optimal solutions obtained in the first stage to cover the PF. To validate the performance of ATLMOEA in solving various kinds of LMOPs, it is compared to eight competitive LMOEAs (i.e., MOEA/DVA [25], LMOCSO [40], DGEA [36], WOF [31], LMEA [27], LMOEA-DS [33], FDV [39], and MOEA/PSL [34]) on two well-known benchmark suites of LSMOP and IMF problems. Moreover, the parameter sensitivity analysis of ε (controlling the switch of two evolutionary stages) and K (setting the number of neurons) in ATLMOEA, the effectiveness of adaptive two-stage strategy, and the running time of ATLMOEA have been experimentally studied in this paper to clarify our advantages.

In our future work, to train a more suitable model, the selection of training model (e.g., machine learning or deep learning models) into LMOEA and the construction of training model will be further studied for tackling various LMOPs. In addition, it is also interesting to apply the proposed ATLMOEA for solving complete and real combinatorial problems to test the proposed method. Finally, in the follow-up research, we focus on how to improve the search speed and performance of ATLMOEA when dealing with LMOPs with more than 1000 decision variables.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61876110, in part by Shenzhen Science and Technology Program under Grants JCYJ20220531101411027, JCYJ20190808164211203 and R2020A045, in part by the Guangdong "Pearl River Talent Recruitment Program" under Grant 2019ZT08X603, in part by the Guangdong "Pearl River Talent Plan" under Grant 2019JC01X235, and in part by Shenzhen Talents Special Project - Guangdong Provincial Innovation and Entrepreneurship Team Supporting Project (Grant 2021344612).

Reference

[1] C. Chang, "Deep and Shallow Architecture of Multilayer Neural Networks," IEEE Transactions on Neural Networks and Learning Systems, 26 (10) (2015) 2477-2486.

[2] Y. Mei, X. Li, X. Yao, "Cooperative Coevolution with Route Distance Grouping for Large-Scale Capacitated Arc Routing Problems," IEEE Transactions on Evolutionary Computation,18 (3) (2012) 435-449.

[3] X. Zhang, K. Zhou, H. Pan, L. Zhang, X. Zeng, and Y. Jin, "A Network Reduction-Based Multiobjective Evolutionary Algorithm for Community Detection in Large-Scale Complex Networks," IEEE Transactions on Cybernetics, 50 (2) (2020) 703-716.

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Transactions on Evolutionary Computation, 6 (2) (2002) 182-197.

[5] Y. Xiang, Y. Zhou, X. Yang, and H. Huang, "A many-objective evolutionary algorithm with pareto-adaptive reference points," IEEE Transactions on Evolutionary Computation, 24 (1) (2019) 99–113.

[6] S. Liu, Q. Yu, Q. Lin, K. Tan, "An adaptive clustering-based evolutionary algorithm for many-objective optimization problems," Information Sciences, 537 (2020) 261-283.

[7] S. Liu, J. Zheng, Q. Lin, K. Tan, "Evolutionary multi and many-objective optimization via clustering for environmental selection," Information Sciences, 578 (2021) 930-949.

[8] Q. Zhang and H. Li, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," IEEE Transactions on Evolutionary Computation, 11 (6) (2007) 712-731.

[9] S. Liu, Q. Lin, K. Wong, L. Ma, C. A. Coello, D. Gong, "A novel multi-objective evolutionary algorithm with dynamic decomposition strategy," Swarm and Evolutionary Computation,48 (2019) 182-200.

[10] J. Li, G. Chen, M. Li, H. Chen, "An enhanced-indicator based many-objective evolutionary algorithm with adaptive reference point," Swarm and Evolutionary Computation, 55 (2020) 100669.

[11] P. Zhang, J. Li, T. Li, and H. Chen, "A new many-objective evolutionary algorithm based on determinantal point processes," IEEE Transactions Evolutionary Computation, 25 (2) (2020) 334–345.

[12] S. Qi, J. Zou, S. Yang, J. Zheng, "A level-based multi-strategy learning swarm optimizer for large-Scale multi-objective optimization," Swarm and Evolutionary Computation, 73 (2022) 101100.

[13] Z. Ding, L. Chen, D. Sun, X. Zhang, "A multi-stage knowledge-guided evolutionary algorithm for large-scale sparse multi-objective optimization problems," Swarm and Evolutionary Computation, 73 (2022) 101119.

[14] S. Qin, C. Sun, Y. Jin, Y. Tan, and J. Fieldsend, "Large-Scale Evolutionary Multiobjective Optimization Assisted

by Directed Sampling," IEEE Transactions on Evolutionary Computation, 25 (4) (2021) 724-738.

[15] W. Hong, P. Yang &K. Tang, "Evolutionary Computation for Large-scale Multi-objective Optimization: A Decade of Progresses," International Journal of Automation and Computing, 18 (2021) 155–169.

[16] K. Deb and H. Jain, "An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems with Box Constraints," IEEE Transactions on Evolutionary Computation, 18 (4) (2014) 577-601.

[17] S. Das and P. N. Suganthan, "Differential Evolution: A Survey of the State-of-the-Art," IEEE Transactions on Evolutionary Computation, 15 (1) (2011) 4-31.

[18] Q. Lin, S. Liu, Q. Zhu, C. T, "Particle Swarm Optimization With a Balanceable Fitness Estimation for Many-Objective Optimization Problems," IEEE Transactions on Evolutionary Computation, 22 (1) (2018) 32-46.

[19] L. M. Antonio and C. A. C. Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," 2013 IEEE Congress on Evolutionary Computation, (2013) 2758-2765.

[20] L. M. Antonio and C. A. C. Coello, "Decomposition-based approach for solving large-scale multi-objective problems". In Proceedings of the International Conference on Parallel Problem Solving from Nature, (2016) 525–534.

[21] F. Sander, H. Zille, and S. Mostaghim, "Transfer strategies from single- to multi-objective grouping mechanisms." In Proceedings of the 2018 Annual Conference on Genetic and Evolutionary Computation Conference. ACM, (2018) 729–736.

[22] Bin Cao, Jianwei Zhao, Yu Gu, Yingbiao Ling, Xiaoliang Ma, "Applying graph-based differential grouping for multiobjective large-scale optimization," Swarm and Evolutionary Computation, 53 (2020) 100626.

[23] S. Liu, Q. Lin, Y. Tian and K. C. Tan, "A Variable Importance-Based Differential Evolution for Large-Scale Multiobjective Optimization," IEEE Transactions on Cybernetics, in press (2021).

[24] S. Liu, Q. Lin, K. C. Wong, Q. Li, and K. C. Tan, "Evolutionary Large-Scale Multiobjective Optimization: Benchmarks and Algorithms," IEEE Transactions on Evolutionary Computation, in press (2021).

[25] X. Ma, et al., "A Multiobjective Evolutionary Algorithm Based on Decision Variable Analyses for Multiobjective Optimization Problems with Large-Scale Variables," IEEE Transactions on Evolutionary Computation, 20 (2) (2016) 275-298.

[26] S. Liu, Q. Lin, L. Feng, K. C. Wong and K. C. Tan, "Evolutionary Multitasking for Large-Scale Multiobjective Optimization," IEEE Transactions on Evolutionary Computation, in press (2022).

[27] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A Decision Variable Clustering-Based Evolutionary Algorithm for Large-Scale Many-Objective Optimization," IEEE Transactions on Evolutionary Computation, 22 (1) (2018) 97-112.

[28] Li Wenchao, Z. Yong and X. Shixiong, "A Novel Clustering Algorithm Based on Hierarchical and K-means Clustering," 2007 Chinese Control Conference, (2007) 605-609.

[29] Y. Xu et al., "A Multi-Population Multi-Objective Evolutionary Algorithm Based on the Contribution of Decision Variables to Objectives for Large-Scale Multi/Many-Objective Optimization," IEEE Transactions on Cybernetics, in press (2022).

[30] R. Liu, R. Ren, J. Liu, J. Liu, "A clustering and dimensionality reduction based evolutionary algorithm for large-scale multi-objective problems," Applied Soft Computing, 89 (2020) 106120.

[31] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "A Framework for Large-Scale Multiobjective Optimization Based on Problem Transformation," IEEE Transactions on Evolutionary Computation, 22 (2) (2018) 260-275.

[32] C. He, et al., "Accelerating Large-Scale Multiobjective Optimization via Problem Reformulation," IEEE Transactions on Evolutionary Computation, 23 (6) (2019) 949-961.

[33] S. Qin, C. Sun, Y. Jin, Y. Tan, and J. Fieldsend, "Large-Scale Evolutionary Multiobjective Optimization Assisted by Directed Sampling," IEEE Transactions on Evolutionary Computation, 25 (4) (2021) 724-738.

[34] Y. Tian, C. Lu, X. Zhang, K. C. Tan, and Y. Jin, "Solving Large-Scale Multiobjective Optimization Problems with Sparse Optimal Solutions via Unsupervised Neural Networks," IEEE Transactions on Cybernetics, 51 (6)

(2021) 3115-3128.

[35] Y. Tian, C. Lu, X. Zhang, F. Cheng and Y. Jin, “A Pattern Mining-Based Evolutionary Algorithm for Large-Scale Sparse Multiobjective Optimization Problems,” IEEE Transactions on Cybernetics, in press, 2020.

[36] C. He, R. Cheng, and D. Yazdani, “Adaptive offspring generation for evolutionary large-scale multiobjective optimization,” IEEE Transactions on Systems, Man, and Cybernetics: System, 52 (2) (2022) 786–798.

[37] W. Hong, K. Tang, A. Zhou, H. Ishibuchi and X. Yao, "A Scalable Indicator-Based Evolutionary Algorithm for Large Scale Multiobjective Optimization," IEEE Transactions on Evolutionary Computation, 23 (3) (2019) 525-537.

[38] Q. Deng, Q. Kang, L. Zhang, M. C. Zhou and J. An, "Objective Space-based Population Generation to Accelerate Evolutionary Algorithms for Large-scale Many-objective Optimization," IEEE Transactions on Evolutionary Computation, in press (2022).

[39] X. Yang, J. Zou, S. Yang, J. Zheng and Y. Liu, “A Fuzzy Decision Variables Framework for Large-scale Multiobjective Optimization,” IEEE Transactions on Evolutionary Computation, in press, 2021.

[40] Y. Tian, X. Zheng, X. Zhang, and Y. Jin, “Efficient Large-Scale Multiobjective Optimization Based on a Competitive Swarm Optimizer,” IEEE Transactions on Cybernetics, 50 (8) (2020) 3696-3708.

[41] S. Liu, Q. Lin, Q. Li and K. C. Tan, "A Comprehensive Competitive Swarm Optimizer for Large-Scale Multiobjective Optimization," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, 52 (9) (2022) 5829-5842.

[42] Sheng Qi, Juan Zou, Shengxiang Yang, Yaochu Jin, Jinhua Zheng, Xu Yang, “A self-exploratory competitive swarm optimization algorithm for large-scale multiobjective optimization,” Information Sciences,609 (2022) 1601-1620

[43] Y. Yuan, H. Xu, B. Wang and X. Yao, “A New Dominance Relation-Based Evolutionary Algorithm for Many-Objective Optimization,” IEEE Transactions on Evolutionary Computation, 20 (1) (2016) 16-37.

[44] M. Li, S. Yang and X. Liu, “Shift-Based Density Estimation for Pareto-Based Algorithms in Many-Objective Optimization,” IEEE Transactions on Evolutionary Computation, 18 (3) (2014) 348-365.

[45] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, “Test Problems for Large-Scale Multiobjective and Many-Objective Optimization,” IEEE Transactions on Cybernetics, 47 (12) (2017) 4108-4121.

[46] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff, “A multiobjective evolutionary algorithm using Gaussian process-based inverse modeling,” IEEE Transactions on Evolutionary Computation, 19 (2015) 838–856.

[47] N. Jin and D. Liu, “Wavelet Basis Function Neural Networks for Sequential Learning,” in IEEE Transactions on Neural Networks, 19 (3) (2008) 523-528.

[48] R. Cheng and Y. Jin, “A Competitive Swarm Optimizer for Large Scale Optimization,” IEEE Transactions on Cybernetics, 45 (2) (2015) 191-204.

[49] S. Liu, M. Jiang, Q. Lin and K. C. Tan, "Evolutionary Large-Scale Multiobjective Optimization via Self-guided Problem Transformation," 2022 IEEE Congress on Evolutionary Computation (CEC), (2022) 1-8.

[50] B. G. Bodmann and P. K. Singh, “Burst Erasures and the Mean-Square Error for Cyclic Parseval Frames,” in IEEE Transactions on Information Theory, 57 (7) (2011) 4622-4635.

[51] J. Wu, "MARG Attitude Estimation Using Gradient-Descent Linear Kalman Filter," IEEE Transactions on Automation Science and Engineering, 17 (6) (2020) 1777-1790.

[52] X. Wang, B. Zhang, J. Wang, K. Zhang, Y. Jin, “A Cluster-Based Competitive Particle Swarm Optimizer with a Sparse Truncation Operator for Multi-Objective Optimization,” Swarm and Evolutionary Computation, 71 (2022) 101083.

[53] Y. Tian, et al., “Evolutionary Large-Scale Multi-Objective Optimization: A Survey,” ACM Computing Surveys, in press, 2021.

[54] Q. Yang, W. -N. Chen, J. D. Deng, Y. Li, T. Gu and J. Zhang, “A Level-Based Learning Swarm Optimizer for Large-Scale Optimization,” IEEE Transactions on Evolutionary Computation, 22 (4) (2018) 578-594.

[55] Y. Zhang, et al., “Two-Stage Robust Optimization Under Decision Dependent Uncertainty,” IEEE/CAA Journal

of Automatica Sinica, in press, 2022.

[56] H. K. Singh, M. M. Islam, T. Ray, M. R., “Nested evolutionary algorithms for computationally expensive bilevel optimization problems: Variants and their systematic analysis,” Swarm and Evolutionary Computation, 48 (2019) 329-344.

[57] Y. Tian, X. Xiang, X. Zhang, R. Cheng and Y. Jin, “Sampling Reference Points on the Pareto Fronts of Benchmark Multi-Objective Optimization Problems,” 2018 IEEE Congress on Evolutionary Computation (CEC), (2018) 1-6.

[58] S. Liu, Q. Lin, K. C. Tan, M. Gong and C. A. Coello Coello, “A Fuzzy Decomposition-Based Multi/Many-Objective Evolutionary Algorithm,” IEEE Transactions on Cybernetics, 52 (5) (2022) 3495-3509.

[59] R. Cheng, Y. Jin, M. Olhofer and B. Sendhoff, “A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization,” IEEE Transactions on Evolutionary Computation, 20 (5) (2016) 773-791.

[60] P. A. N. Bosman and D. Thierens, “The balance between proximity and diversity in multiobjective evolutionary algorithms,” IEEE Transactions on Evolutionary Computation, 7 (2) (2003) 174-188.

[61] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, “PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum],” IEEE Computational Intelligence Magazine, 12 (2017) 73–87.