# Automatic Niching Differential Evolution With Contour Prediction Approach for Multimodal Optimization Problems

Zi-Jia Wang, *Student Member, IEEE*, Zhi-Hui Zhan, *Senior Member, IEEE*, Ying Lin, *Member, IEEE*, Wei-Jie Yu, *Member, IEEE*, Hua Wang, Sam Kwong, *Fellow, IEEE*, and Jun Zhang, *Fellow, IEEE*

***Abstract*—Niching techniques have been widely incorporated into evolutionary algorithms (EAs) for solving multimodal optimization problems (MMOPs). However, most of the existing niching techniques are either sensitive to the niching parameters or require extra fitness evaluations (FEs) to maintain the niche detection accuracy. In this paper, we propose a new automatic niching technique based on the affinity propagation clustering (APC) and design a novel niching differential evolution (DE) algorithm, termed as automatic niching DE (ANDE), for solving MMOPs. In the proposed ANDE algorithm, APC acts as a parameter-free automatic niching method that does not need to predefine the number of clusters or the cluster size. Also, it can facilitate locating multiple peaks without extra FEs. Furthermore, the ANDE algorithm is enhanced by a contour prediction approach (CPA) and a two-level local search (TLLS) strategy. First, the CPA is a predictive search strategy. It exploits the individual distribution information in each niche to estimate the contour landscape, and then predicts the rough position of the potential peak to help accelerate the convergence speed. Second, the TLLS is a solution refine strategy to further increase the solution accuracy after the CPA roughly predicting the peaks. Compared with the other state-of-the-art DE and non-DE multimodal algorithms, even the winner of competition on multimodal optimization, the experimental results on 20 widely used benchmark functions illustrate the superiority of the proposed ANDE algorithm.**

***Index Terms*—Affinity propagation clustering (APC), contour prediction approach (CPA), differential evolution (DE), multimodal optimization problems (MMOPs), niching techniques.**

Manuscript received January 23, 2018; revised July 2, 2018, August 28, 2018, and January 29, 2019; accepted April 5, 2019. Date of publication April 11, 2019; date of current version January 29, 2020. This work was supported in part by the Outstanding Youth Science Foundation under Grant 61822602, in part by the National Natural Science Foundations of China under Grant 61772207 and Grant 61873097, in part by the Natural Science Foundation of Guangdong Province (NSFGD) for Distinguished Young Scholars under Grant 2014A030306038, in part by the Project for Pearl River New Star in Science and Technology under Grant 201506010047, in part by GDUPS (2016), in part by NSFGD under Grant 2014B050504005, and in part by the Hong Kong GRF-RGC General Research Fund (9042489) under Grant CityU 11206317. *(Corresponding authors: Zhi-Hui Zhan; Jun Zhang.)*

Z.-J. Wang is with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China.

Z.-H. Zhan, Y. Lin, and W.-J. Yu are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China, and also with the Guangdong Provincial Key Laboratory of Computational Intelligence and Cyberspace Information, South China University of Technology, Guangzhou 510006, China (e-mail: zhanapollo@163.com).

H. Wang and J. Zhang are with the Institute for Sustainable Industries and Liveable Cities, College of Engineering and Science, Victoria University, Melbourne, VIC 8001, Australia (e-mail: junzhang@ieee.org).

S. Kwong is with the Department of Computer Science, City University of Hong Kong, Hong Kong.

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2019.2910721

## I. INTRODUCTION

MANY real-world problems own multiple global optima, such as protein structure prediction [1], electromagnetic design [2], and pedestrian detection [3], which are known as multimodal optimization problems (MMOPs). For example, pedestrian detection often requires to extract multiple pedestrian from a given image [3]. Locating all the global optima of an MMOP has significant benefits. If the optimizer is able to find multiple promising solutions simultaneously, we will have several choices to keep the satisfactory performance [4], [5]. Therefore, it is desirable to locate multiple optima of practical MMOPs.

Evolutionary algorithms (EAs), such as genetic algorithm (GA) [6]–[9], ant colony optimization (ACO) [10]–[13], estimation of distribution algorithm (EDA) [14]–[16], particle swarm optimization (PSO) [17]–[23], and differential evolution (DE) [24]–[33], have the potential advantages for solving MMOPs since their population-based search manner maintains multiple candidate solutions. However, most of the traditional EAs only focus on locating a single optimal solution. To tackle MMOPs, techniques known as "niching" have been proposed to partition the whole population into several niches [34]–[52]. Following this idea, different niching methods have been proposed, such as the crowding [34], speciation [35], clustering [36], hill-valley [46], fitness sharing [46], recursive middling [47], and topological species conservation [52]. Based on these niching techniques, various EAs have been extended for solving MMOPs, including GA [48]–[51], ACO [53], EDA [54], PSO [55], [56], and DE [34]–[45]. Among these existing multimodal algorithms, DE variants have shown their effectiveness and superiority in the reported results [34]–[45]. Therefore, this paper focuses on DE for tackling MMOPs.

However, when applying DE or other EAs variants in MMOPs, there is still much room for improvement. One of the most significant issues is that the current niching methods are very sensitive to the niching parameters, such as the crowding size $C$ in crowding [34], the species radius $r$ in speciation [35], or the cluster size $M$ in clustering [36]. If the niching parameters are not properly set, the performance of the algorithms will deteriorate severely. There are also some parameter-free niching methods using fitness evaluations (FEs) to discover hill and valley for better niching [46]. However, how to design an efficient method to disperse the search of population to locate different peaks is still a challenging issue and a significant research topic in the MMOP community.

To overcome the above drawbacks, this paper proposes a novel automatic niching method based on the affinity propagation clustering (APC). APC is a famous clustering approach which was published in *Science* [57] and has also been applied in EAs [58]. It does not require the number of clusters and the initial selection of exemplars for clustering. By using the APC, the niching method can not only form niches automatically to locate multiple peaks but also can avoid predefining the sensitive parameters, such as the cluster size or the number of clusters compared with other clustering niching methods [36], [37]. Meanwhile, compared with other parameter-free niching methods [46], the use of APC niching does not introduce any extra FEs.

As we focus on DE for tackling MMOPs in this paper, we termed our proposed APC-based DE algorithm as automatic niching DE (ANDE). After using APC to partition the population into suitable clusters/niches automatically to locate different peak regions, the DE evolutionary operators are performed within each niche. Then, after the evolution of each generation, a contour prediction approach (CPA) is further developed to estimate the contour landscape of each niche. Specifically, the CPA utilizes the distribution information of some individuals in the niche to predict the rough position of the potential optima, so as to accelerate the convergence speed. However, as the potential optimum predicted by CPA is a rough position, it may still not be accurate enough. In order to enhance the exploitation ability and improve the solution accuracy, a two-level local search (TLLS) strategy is further performed after the CPA.

Therefore, the performance of ANDE is guaranteed by not only the APC but also the CPA and TLLS. Noted that these three components act different roles in ANDE and compensate with each other. Specifically, the APC is used for automatically forming niches and effectively locating different optimal areas in solving MMOPs. Based on the results of APC niching, the CPA and TLLS are further performed within each cluster/niche for approaching the peaks. When dealing with the high-dimensional MMOPs, the principal component analysis (PCA) technique is incorporated into ANDE to achieve dimensionality reduction for better niching. Note that the PCA does not affect the functional landscape because it performs on an additional population to reduce the dimension so as to help the APC cluster individuals more easily and thus to obtain better niching results. All the other operators in ANDE (such as the evolutionary operators and FEs) are still executed on the original space, which are not affected by the PCA. Moreover, the locations of the optima and the topology of the functional landscape are also not affected by the PCA. The performance of ANDE is evaluated on 20 widely used benchmark multimodal functions from the CEC2015 multimodal competition. The experimental results fully show the superiority and feasibility of ANDE compared with many state-of-the-art multimodal optimization algorithms and the winner of the CEC2015 competition on multimodal optimization.

The rest of this paper is organized as follows. Section II reviews the DE algorithm and its current developments on MMOPs. Section III describes the proposed ANDE algorithm in detail. The experimental results and discussions are shown in Section IV. Finally, the conclusions are given in Section V.

## II. Related Work

### A. DE Algorithm

DE is a population-based stochastic search algorithm, which evolves according to the difference between the individuals and by a loop of operators, including mutation, crossover, and selection. Recently, ensemble methods receive an increasing attention in designing high-quality DE algorithms [59]–[61]. The operations of DE in each generation are described below.

*Mutation:* In each generation $g$, the mutation operation is performed on each individual $\boldsymbol{x}_{i,g}$ to create its corresponding mutant vector $\boldsymbol{v}_{i,g}$. Three mutation strategies frequently used in the literatures are listed below.

1) The DE/rand/1 mutation strategy is as

$$\boldsymbol{v}_{i,g} = \boldsymbol{x}_{r1,g} + F \times \left(\boldsymbol{x}_{r2,g} - \boldsymbol{x}_{r3,g}\right). \tag{1}$$

2) The DE/best/1 mutation strategy is as

$$\boldsymbol{v}_{i,g} = \boldsymbol{x}_{\text{best},g} + F \times \left(\boldsymbol{x}_{r1,g} - \boldsymbol{x}_{r2,g}\right). \tag{2}$$

3) The DE/current-to-best/1 mutation strategy is as

$$\begin{aligned}\boldsymbol{v}_{i,g} = \boldsymbol{x}_{i,g} &+ F \times \left(\boldsymbol{x}_{\text{best},g} - \boldsymbol{x}_{i,g}\right)\\ &+ F \times \left(\boldsymbol{x}_{r1,g} - \boldsymbol{x}_{r2,g}\right)\end{aligned} \tag{3}$$

where $r1$, $r2$, and $r3$ are different random integers selected from $\{1, 2, \ldots, N\}$, which are all different from $i$. The amplification factor $F$ is a positive control parameter, which amplifies the differential vectors. $\boldsymbol{x}_{\text{best},g}$ is the individual with the best fitness value in generation $g$.

*Crossover:* Generally, after the mutation, DE performs a binomial crossover operation on $\boldsymbol{x}_{i,g}$ and $\boldsymbol{v}_{i,g}$ to generate a trial vector $\boldsymbol{u}_{i,g}$ by

$$u_{i,j,g} = \begin{cases} v_{i,j,g} & \text{if rand}(0,1) \le \text{CR or } j = j_{\text{rand}} \\ x_{i,j,g} & \text{otherwise} \end{cases} \tag{4}$$

where $j_{\text{rand}}$ is an integer randomly selected from $\{1, 2, \ldots, D\}$ to ensure that the trial vector has at least one dimension different from $\boldsymbol{x}_{i,g}$. The crossover rate CR is another parameter, which determines the fraction of vector components inherited from the mutant vector.

*Selection:* To determine whether the trial vector $\boldsymbol{u}_{i,g}$ will survive into the next generation, the $\boldsymbol{u}_{i,g}$ is compared with the $\boldsymbol{x}_{i,g}$. The one with the better fitness value enters the next generation. For instance, for a maximization problem, the individual with a larger fitness value survives into the next generation, as

$$\boldsymbol{x}_{i,g+1} = \begin{cases} \boldsymbol{u}_{i,g}, & \text{if } f(\mathrm{u}_{i,g}) \geq f(\mathrm{x}_{i,g}) \\ \mathrm{x}_{i,g}, & \text{otherwise} \end{cases} \tag{5}$$

where $f(\boldsymbol{x})$ is the FE function.

### B. DE for MMOPs

Various algorithms have been proposed to solve MMOPs in recent years. Among these existing multimodal algorithms, DE variants have shown promising performance [34]–[45]. To have a better view of these multimodal algorithms based on DE, which is also the focus of this paper, we briefly describe them as follows.

*1) Niching Methods for DE:* Niching techniques have been widely used to help EAs solve MMOPs [34]–[52]. The two most representative niching methods are crowding [34] and speciation [35]. When applied into DE, the algorithms are called crowding DE (CDE) and speciation DE (SDE). In CDE, each newly generated offspring is compared with its most similar parent $\boldsymbol{X}$ from a crowd, which is formed by randomly selecting $C$ individuals. The offspring will replace $\boldsymbol{X}$ if it is better. In SDE, the population is partitioned into several species according to the individuals' fitness and a species radius $r$, and the DE operators are executed within each species. However, these two niching methods are very sensitive to their parameters, such as the crowding size $C$ in crowding and the species radius $r$ in speciation.

To improve population partition, the self-adaptive clustering-based DE (Self-CCDE and Self-CSDE) proposed by Gao *et al.* [36] and the neighborhood mutation-based DE (NCDE and NSDE) proposed by Qu *et al.* [37] applied the clustering techniques into crowding and speciation methods. These clustering methods introduced a parameter, cluster size $M$, which is less sensitive compared with the crowding size $C$ and the species radius $r$. However, the $M$ also influences the algorithm performance directly [36], [37].

*2) Improved Mutation Strategies for DE:* Many efforts have been paid to improve the mutation strategy in DE-based multimodal algorithms. Biswas *et al.* [38] improved the niching DE by developing an information sharing mechanism, and the proposed algorithms were termed as LoINDE (LoICDE and LoISDE). Meanwhile, they presented a parent-centric normalized mutation with proximity-based CDE (PNPCDE) which can fully utilize the neighborhood information [39]. Dual-strategy DE with APC (DSDE), proposed by Wang *et al.* [40], which used two mutation strategies and adaptively chooses one of them for each individual, so as to balance the convergence and diversity. Furthermore, they also proposed a new selection operator based on APC to select the more suitable individuals. Hui and Suganthan [41] proposed ensemble and arithmetic recombination-based SDE, termed as EARSDE, which applied arithmetic recombination in the speciation method and used ensemble strategies in the neighborhood mutation to balance the exploration and exploitation.

*3) Multiobjective Techniques for DE:* Different from the methods mentioned above, some researchers use multiobjective techniques to transfer an MMOP into a multiobjective optimization problem (MOP), more specifically, a bi-objective optimization problem. Generally, the first objective is the multimodal function itself for fast convergence, whereas the second objective is self-designed for improving diversity. For example, the MOBiDE [44] used the mean Euclidean distance of one individual to all the other individuals as the second objective, which should be maximized to prevent from converging to only one peak. Apart from this, Wang *et al.* [45] designed the MOMMOP algorithm using a quite different transformation, which designed two conflict objectives for each dimension.

Although these techniques mentioned above have shown their effectiveness in solving MMOPs, their performance is still not satisfactory especially in the problems with high dimensions or complexities.

## III. ANDE

This section describes the proposed ANDE algorithm. First, the APC for efficiently partitioning the population to automatically form niches and to locate different peaks is described. Second, the CPA for predicting the rough position of the potential optimum in each niche is designed. Moreover, the TLLS strategy to improve the solution accuracy and enhance the exploitation ability is introduced later. At last, ANDE is extended for solving high-dimensional MMOPs and the whole ANDE algorithm is given.

### A. APC to Locate Different Peaks

Different from other clustering methods, APC does not require the number of clusters or cluster size and initial selection of exemplars or clustering centers, which can avoid the sensitive parameters. The motivation and rationality of APC are that all the individuals are regarded as potential exemplars for any other individual, and then the clusters/niches are automatically formed according to the message-passing process [57], [58]. The message-passing process is a loop process to calculate how suitable for an individual being the exemplar for another individual, and how appropriate for an individual to choose another individual as its exemplar. In order to calculate such information, two kinds of messages are defined in APC for exchanging information among individuals: "responsibility" and "availability." The responsibility between individual $\boldsymbol{x}_i$ and its candidate exemplar individual $\boldsymbol{x}_k$ is denoted as $r(i, k)$, which is sent from $\boldsymbol{x}_i$ to $\boldsymbol{x}_k$, as illustrated in Fig. 1(a). The availability between individual $\boldsymbol{x}_i$ and its candidate exemplar individual $\boldsymbol{x}_k$ is denoted as $a(i, k)$, which is sent from $\boldsymbol{x}_k$ to $\boldsymbol{x}_i$, as shown in Fig. 1(b).

Particularly, the responsibility $r(i, k)$ shows how suitable for the individual $\boldsymbol{x}_k$ being the exemplar for individual $\boldsymbol{x}_i$. It is set to the similarity between individual $\boldsymbol{x}_i$ and individual $\boldsymbol{x}_k$, minus the largest of the availabilities and similarities between individual $\boldsymbol{x}_i$ and other competing candidate exemplars. Inversely, the availability $a(i, k)$ reflects how appropriate for individual $\boldsymbol{x}_i$ to choose individual $\boldsymbol{x}_k$ as its exemplar. It is set to the self-responsibility $r(k, k)$ plus the sum of the positive responsibilities the individual $\boldsymbol{x}_k$ receives from other supporting individuals. To limit the influence of strong incoming positive responsibilities, the availability $a(i, k)$ is no larger than zero.

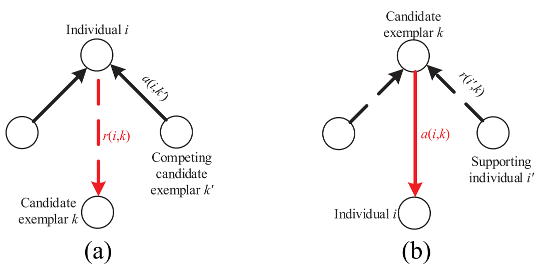


Fig. 1. Message passing in APC. (a) Responsibility. (b) Availability.

To begin with, the availabilities $a(i, k)$ are initialized as 0. Then, in each iteration, the responsibilities and availabilities are computed using the rules

$$r(i,k) = s(i,k) - \max\{a(i,k') + s(i,k')\}$$
$$k' = 1, 2, 3, \ldots, N, k' \neq k \tag{6}$$

$$a(i,k) = \min\left\{0, r(k,k) + \sum_{i'=1, i'\neq i,k}^{N} \max\{0, r(i',k)\}\right\} \tag{7}$$

where $s(i, k)$ is the similarity between individuals $\boldsymbol{x}_i$ and $\boldsymbol{x}_k$, which is set as the negative squared error (Euclidean distance): $s(i, k) = -\|\boldsymbol{x}_i - \boldsymbol{x}_k\|^2$.

During the message-passing process, each message is set as $\lambda$ times its value from the last iteration plus $1-\lambda$ times its current value, shown as

$$r(i,k) = \lambda \times r(i,k)_{\text{last}} + (1-\lambda) \times r(i,k) \tag{8}$$

$$a(i,k) = \lambda \times a(i,k)_{\text{last}} + (1-\lambda) \times a(i,k). \tag{9}$$

For each individual $\boldsymbol{x}_i$, the individual $\boldsymbol{x}_k$ that maximizes $a(i, k) + r(i, k)$ is identified as the exemplar for individual $\boldsymbol{x}_i$. The message-passing process will terminate after the maximum number of iterations *Mits*, or the estimated exemplars stay stagnation for a certain number of iterations *Cits*. In that way, the clusters are automatically formed. We have investigated the influences of the parameters $\lambda$, *Mits*, and *Cits* and presented the results in the Tables S.I–S.III in the supplementary material. Considering the aspects of promising results and light computational burden, we use parameters of $\lambda = 0.9$, $Mits = 100$, and $Cits = 30$. Herein, the relatively larger $\lambda$ will maintain more message from the last iteration, which will avoid numerical oscillations effectively and make clustering results more stable. Besides, the *Mits* and *Cits* with relatively smaller values can relieve the computational burden of APC.

The complete procedure of APC can be shown as the following five steps.

*Step 1:* Initialize $a(i, k) = 0$.

*Step 2:* Calculate the temporary values of $r(i, k)$ and $a(i, k)$ by using (6) and (7).

*Step 3:* Keep the message from the last iteration and calculate the final values of $r(i, k)$ and $a(i, k)$ by using (8) and (9).

*Step 4:* For each individual $\boldsymbol{x}_i$, the individual $\boldsymbol{x}_k$ that maximizes $a(i, k) + r(i, k)$ is identified as the exemplar for individual $\boldsymbol{x}_i$.

*Step 5:* Repeat steps 2 to 4 until the clustering termination criterion is satisfied.

After the clustering procedure, the clusters, so-called niches are automatically formed. As we can see, the niching strategy based on APC avoids using the sensitive parameters, such as the number of clusters or the cluster size $M$. Besides, comparing with other parameter-free niching strategies [46], [47], [52], the whole message-passing process does not cost any extra FEs.

## *B. CPA to Predict Optimal Region*

After forming niches automatically by APC, the basic DE operators are executed within the niche to generate new individuals. However, for the individuals in each niche, many current multimodal algorithms do not make use of the distribution information of the individuals to help the search. In fact, if we can narrow the search space or speculate the distribution of the potential optima by using the distribution information of individuals, the search process will be quicker and more effective. To this aim, a novel CPA is proposed for predicting the peak by using the distribution information of some individuals in each niche.

The contour method is first used by Lin *et al.* [62] in GA to solve unimodal optimization problems. Here, we modified and applied this method into DE to tackle the MMOPs.

A contour in topography is a smooth line which connects the points with the same elevation. Fig. 2 displays an example of contours. Obviously, the potential summit is also enclosed in the contour.

Inspired by the contours in topography, the CPA is proposed for effectively solving MMOPs by estimating the contour landscape of the problem and predicting the rough position of the potential optima. In CPA, the solutions can be regarded as the positions, whereas the fitness values of the solutions can be perceived as the elevations of the positions. Taking a 2-D problem for instance, each solution $\boldsymbol{x}_i$ can be represented by $(x_{i,1}, x_{i,2})$. When several niches are formed, each niche $S_j$ will have a niche seed which is with the best fitness value. For each niche $S_j$, we found some individuals nearest to the niche seed in $S_j$ to form a network, as shown in Fig. 3(a). In this figure, the solid cycles denote the niche seeds, while the hollow cycles denote the individuals nearest to the niche seed. Each position (solution/individual) is denoted by $(a, b)$ $c$ in the figure, where $(a, b)$ is the coordinate of the current individual and $c$ is the elevation (fitness value) of the current individual.

Then, we calculate each dimension of each interpolated point $\boldsymbol{x}'_i$ as

$$\begin{cases} x'_{i,1} = x_{\text{lbest},1} + \frac{f - f_{\text{lbest}}}{f_i - f_{\text{lbest}}}\left(x_{i,1} - x_{\text{lbest},1}\right) \\ x'_{i,2} = x_{\text{lbest},2} + \frac{f - f_{\text{lbest}}}{f_i - f_{\text{lbest}}}\left(x_{i,2} - x_{\text{lbest},2}\right) \end{cases} \tag{10}$$

where the $\boldsymbol{x}_{\text{lbest}}$ and $\boldsymbol{x}_i$ are the niche seed and its $i$th adjacency individual in the niche, $f_{\text{lbest}}$ and $f_i$ are the fitness values of $\boldsymbol{x}_{\text{lbest}}$ and $\boldsymbol{x}_i$, respectively, whereas $f$ is the contour value. In order to accelerate the approaching speed to the summit or the best solution, $f$ is set larger than $f_{\text{lbest}}$ for a maximization problem or smaller than $f_{\text{lbest}}$ for a minimization problem, shown as

$$f - f_{\text{lbest}} = \begin{cases} 0.2 \times |f_{\text{lbest}}| + 0.1, & \text{maximization problem} \\ -0.2 \times |f_{\text{lbest}}| - 0.1, & \text{minimization problem} \end{cases} \tag{11}$$

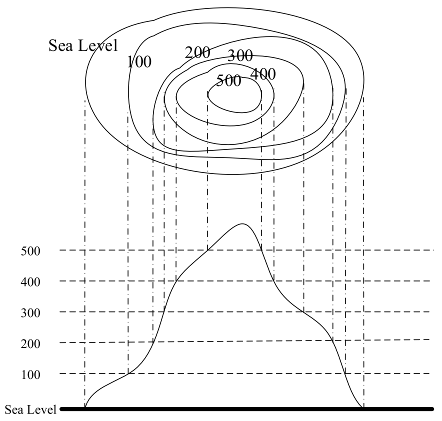


Fig. 2. Example of contours.

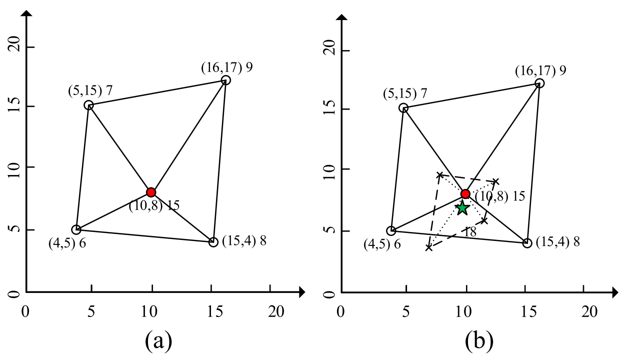


Fig. 3. CPA in a 2-D example. (a) Form the networks. (b) Draw contours and predict peaks.

where 0.2 is the coefficient of $|f_{\text{lbest}}|$ to control the approaching speed to the summit and 0.1 is the disturbance to deal with the condition that the function is with zero fitness value. We have investigated these two values in Tables S.IV and S.V in the supplementary material. The results show that setting the coefficient as 0.2 ($-0.2$ for minimization problems) can achieve the balance between fast convergence and avoiding distortion. Moreover, the disturbance is not sensitive and is simply set as 0.1 herein.

In Fig. 3(b), the contour value is set to 18, and the symbol $x$ denotes the interpolated points. After obtaining the interpolated points, we draw the contours by connecting all these interpolated points.

As we mentioned before, the potential summits are enclosed in a contour. Therefore, after getting several interpolated points and drawing the contours, the potential summits within the contours will be determined. In this paper, we use the centroid $\boldsymbol{x}'$ of the contour to approximately estimate the optima, which can be expressed as

$$\boldsymbol{x}' = \frac{\sum_{i=1}^{K} \boldsymbol{x}_i'}{K} \tag{12}$$

where the $K$ is the number of interpolated points and no more than 5, $\boldsymbol{x}_i'$ is the $i$th interpolated point. Note that if the legal interpolated points are fewer than 3, the contour cannot be drawn. In other words, if the niche $S_j$ has fewer than four individuals, CPA is not used. In Fig. 3(b), the stars represent the potential optima we estimated using (12). For each niche, if the

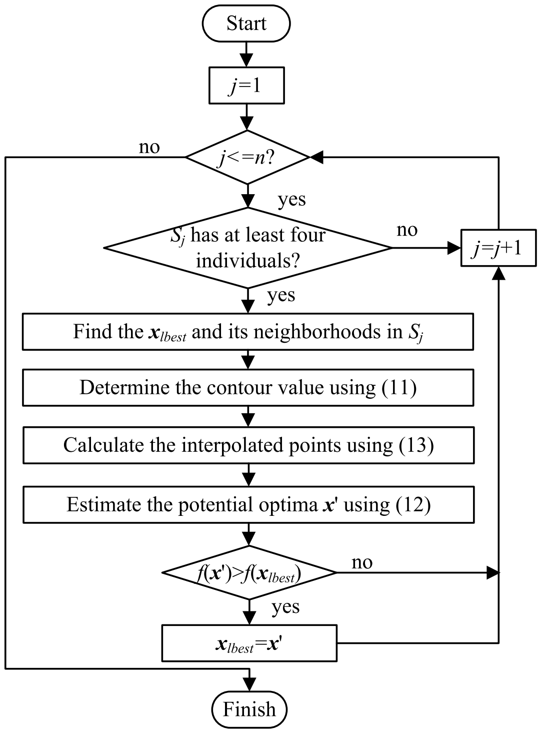

Start
j=1
j<=n?
no
yes
S_j has at least four individuals?
no
j=j+1
yes
Find the x_lbest and its neighborhoods in S_j
Determine the contour value using (11)
Calculate the interpolated points using (13)
Estimate the potential optima x' using (12)
f(x')>f(x_lbest)
no
yes
x_lbest=x'
Finish

Fig. 4. Flowchart of CPA.

fitness of the estimated optimum is better than the niche seed, the estimated optimum will replace the niche seed, otherwise the estimated optimum is ignored.

Now, we extend CPA into any dimensional problems. In a $D$-dimensional problem, each individual $\boldsymbol{x}_i$ can be expressed as $(x_{i,1}, x_{i,2}, \ldots, x_{i,D})$. Similarly, we use the formula like (10) to determine each dimension of the interpolated point $\boldsymbol{x}_i'$ as

$$\begin{cases} x_{i,1}' = x_{\text{lbest},1} + \frac{f-f_{\text{lbest}}}{f_i-f_{\text{lbest}}}\left(x_{i,1} - x_{\text{lbest},1}\right) \\ x_{i,2}' = x_{\text{lbest},2} + \frac{f-f_{\text{lbest}}}{f_i-f_{\text{lbest}}}\left(x_{i,2} - x_{\text{lbest},2}\right) \\ \cdots \\ x_{i,D}' = x_{\text{lbest},D} + \frac{f-f_{\text{lbest}}}{f_i-f_{\text{lbest}}}\left(x_{i,D} - x_{\text{lbest},D}\right). \end{cases} \tag{13}$$

After that we also use (12) to predict the potential optima. The whole procedure of CPA can be seen in Fig. 4 as a flowchart, where $n$ is the number of niches identified in APC.

Based on CPA, ANDE can locate the ranges of the potential summits effectively, which will give a proper guidance of evolution. Meanwhile, forecasting the optima directly will simplify the search process and accelerate the convergence speed. The effectiveness and feasibility of CPA will be fully investigated in Section IV-D2.

## C. TLLS to Refine Solution Accuracy

Although CPA can predict the potential optima and accelerate convergence to some degree, it may still not be accurate enough. Motivated by this, a TLLS is proposed in this paper to enhance the exploitation ability of algorithm and increase the accuracy of solutions. Gaussian distribution is utilized here due to its promising local search performance by sampling small areas, shown in

$$\boldsymbol{x}_{\text{new}} = \text{Gaussian}(\boldsymbol{x}, \sigma) \tag{14a}$$

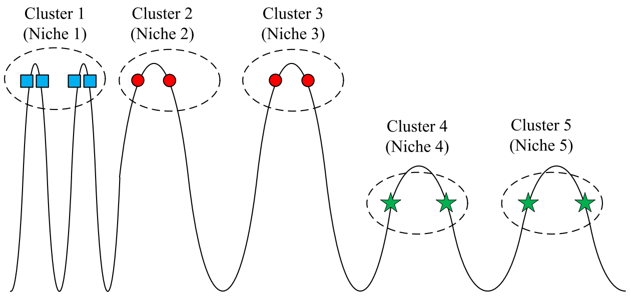


Fig. 5. Illustration of TLLS.

where $\boldsymbol{x}_{\text{new}}$ is the new generated individual by the Gaussian distribution with the individual $\boldsymbol{x}$ as the mean and a standard deviation $\sigma$.

During the evolutionary process, the standard deviation $\sigma$ in Gaussian distribution is decreased using an exponential model, shown in (14b), to sample relatively wide areas in the early stage and sample narrow areas at late. Besides, to keep population diversity and avoid getting trapped in the local optima, $\sigma$ is set larger for the problems with higher dimensions to sample wide areas while smaller for problems with lower dimensions. Herein, the exponent 3 is used to control the lower bound of the $\sigma$. We have investigated the exponent used in (14b) in Table S.VI in the supplementary material. The results show that setting the exponent as 3 can achieve the balance between exploration and exploitation

$$\sigma = 10^{-1-\frac{(10/D+3)\times\text{FEs}}{\text{MaxFEs}}}. \tag{14b}$$

As indicated by its name, the TLLS has two-level local search, including a niching-level local search and an individual-level local search. As locating more optima is the main objective of MMOPs, we first execute the local search operator on niching-level because different niches focus on different peaks. On niching-level, the local search is executed on the niche seed for finding more promising solutions. As there are many niche seeds (each niche has one), it is intuitive that the better niche seed is, the greater chance it is close to the global optima and should do local search. This indicates the opportunity of executing local search should be related to the fitness values of niche seeds. Therefore, we first sort the niche seeds in ascending order according to their fitness values (from worse to better). Then, set the probability of the $i$th niche to do local search as

$$P_i = r_i/n \tag{15}$$

where $r_i$ is the rank of the $i$th niche seed in the sort of fitness, and $n$ is the number of niches.

However, without any heuristic information or prior knowledge about the distribution of peaks in MMOPs, several peaks may be covered by the same niche if the peaks are close to each other or the number of peaks is too large. Suppose the current population distribution is shown in Fig. 5. The niche 1 covers two peaks because they are very close to each other. Niches 2 and 3 cover one peak, respectively. While the niches 4 and 5 cover the local optima. Our proposed niching-level local search based on $P_i$ can avoid wasting the local search FEs on local optima to some degree. For example, the chance of performing local search in the niche 4 and niche 5 is small

**Algorithm 1** TLLS

```
Begin
 1.  Generate the sample standard deviation $\sigma$ using (14b).
 2.  Calculate the niche-level local search probability $P_i$ using (15).
 3.  For each niche $S_i$
 4.    If $rand<P_i$
 5.      For each individual $\boldsymbol{x}_{ik}$ in $S_i$;
 6.        Calculate the individual-level local search probability
             $P_{ik}$ using (16);
 7.        If $rand<P_{ik}$
 8.          Sample 2 points around individual $\boldsymbol{x}_{ik}$ based on
               Gaussian distribution in (14a);
 9.          Evaluate these 2 points and denote the better one as $\boldsymbol{x}'_{ik}$;
10.          If $\boldsymbol{x}'_{ik}$ is better than $\boldsymbol{x}_{ik}$
11.            Replace $\boldsymbol{x}_{ik}$ by $\boldsymbol{x}'_{ik}$;
12.          End If
13.        End If
14.      End For
15.    End If
16.  End For
End
```

because their $P_i$ values are small. However, if we only do local search on niching-level, the accuracy of some optima may not be improved, e.g., only one peak in the niche 1 can be refined. Consequently, after executing local search operator at niching-level, an individual-level local search is also executed. Specifically, if the current niche $i$ satisfies the probability $P_i$ to do local search, some individuals with better fitness value in this niche should also do local search. The method is similar to that in niching-level, which is first sorting the individuals in ascending order according to their fitness values in the current niche (from worse to better). Then set the probability of the $k$th individual to do local search in the $i$th niche as

$$P_{ik} = r_k/n_i \tag{16}$$

where $r_k$ is the rank of the $k$th individual in the sort of fitness, and $n_i$ is size of $i$th niche.

The number of points sampled by local search is set as 2. The whole framework of the TLLS is shown in Algorithm 1.

Using this TLLS scheme, we can enhance the exploitation ability of algorithm and increase the accuracy of all the global optima. Moreover, the fitness rank-based probabilistic scheme [i.e., (15) and (16)] can avoid wasting FEs on local optima and inferior individuals.

## *D. ANDE for High Dimensional Problems*

However, when dealing with the high-dimensional problems, the niching strategy based on APC may also face the difficulty of "curse of dimensionality" like other clustering methods. In order to relieve this difficulty, we do not apply the APC directly on the original population whose individuals (solutions) are in high dimension, but first construct an additional population that is with dimensionality reduction from the original population. This way, the APC can be performed on the individuals of the additional population that are with low dimension, so as to cluster the individuals more easily and more efficiently.

PCA is an effective method to achieve dimensionality reduction using mathematical projection. Therefore, we apply PCA into our ANDE algorithm to construct the additional

**Algorithm 2** ANDE

```
Begin
 1. Randomly generate $N$ individuals as the population $\boldsymbol{P}$.
 2. While $FEs \le MaxFEs$
 3.   (For problems with dimensions higher than 3, using PCA
        method to achieve dimensionality reduction.)
 4.   Using APC for niching.
 5.   For each niche $S_i$
 6.     If $n_i \le 4$ // $n_i$ is the size of $S_i$.
 7.       Generate offspring using standard DE;
 8.       For each offspring in $S_i$
 9.         Compare its fitness with the fitness of its most similar
              parent $X$ in $S_i$ and replace $X$ if the offspring is better;
10.       End For
11.     End If
12.     Predict the potential optima using CPA in Fig. 4.
13.   End For
14.   Execute the TLLS using Algorithm 1.
15. End While
End
```

population by reducing the $D$ dimensions to $k$ dimensions. The specific implementation is shown as the following six steps.

1) Construct a $D \times N$ matrix $X$ using the current population, where $D$ is the dimension of problem and $N$ is the population size.
2) Transform the average value of each row in $X$ to zero.
3) Generate the covariance matrix $C = (1/D)XX^T$.
4) Calculate the eigenvalues and eigenvectors of $C$.
5) Sorting the eigenvectors according to their eigenvalues, and we choose the first $k$ eigenvectors as a new matrix $P$, where $k$ is the dimension we wish to obtain.
6) Obtain $X' = PX$, where $X'$ is the matrix after dimensionality reduction.

In our algorithm, for problems with higher than three dimensions ($D > 3$), we set $k = 3$. In other words, we additionally construct a $3 \times N$ matrix $X'$ using PCA. Accordingly, we can obtain the additional population with $N$ individuals and three dimensions. This way, the additional population is a dimension reduced version of the original population. The APC can be carried out on this additional population to help cluster the individuals more easily and thus obtain better niching results. However, all the other operators in ANDE (such as the evolutionary operators and FEs) are still executed on the original space and using the original population. Therefore, the topology of the functional landscape of the solving problems, the locations of the optima, and the search information of the evolutionary population are not affected by PCA.

Based on PCA technique, we can achieve better niching using APC when dealing with the problems with high dimensions.

### E. Complete Algorithm ANDE

Overall, based on all the components mentioned above, the complete algorithm ANDE is shown in Algorithm 2 as the pseudocode, together with the following advantages.

1) Niching strategy based on APC forms the niches automatically without sensitive parameters or any extra FEs, which is efficient to locate different peaks of MMOPs.
2) After the population partition and evolutionary operations, CPA can effectively speculate the rough position of the potential optimum in each niche, which can give a proper guidance of evolution and accelerate convergence.
3) The TLLS refines the solutions and enhances the exploitation ability of algorithm, which attempts to increase the accuracy of all the global optima. Moreover, the two-level fitness rank-based probabilistic scheme can avoid wasting FEs on local optima and inferior individuals.

## IV. Experimental Results

### A. Benchmark Functions and Performance Measures

All the 20 frequently used multimodal benchmark functions from CEC2015 competition are used to test the performance of ANDE and compared state-of-the-art multimodal optimization algorithms (CEC2015 competition contains the same problems as the CEC2013 test suite) [63]. The main characteristics of these functions are summarized in Table S.VII in the supplementary material due to the page limitation.

Besides, three performance measures, including peak ratio (PR), success rate (SR), and convergence speed (AveFEs) are utilized to evaluate the performance of all the multimodal algorithms. Given a fixed maximum FEs (MaxFEs) and a fixed accuracy level $\varepsilon$, the PR reflects the mean percentage of all global optima found over multiple runs. SR is the percentage of successful runs, where a successful run means all global optima are found in a single run. The AveFEs is the average FEs required to find all the global optima. The mathematical formulas can be expressed

$$\mathrm{PR} = \frac{\sum_{i=1}^{R} \mathrm{NFP}_i}{NP \times R}, \quad \mathrm{SR} = \frac{\mathrm{NSR}}{R}, \quad \mathrm{AveFEs} = \frac{\sum_{i=1}^{R} \mathrm{FEs}_i}{R} \tag{17}$$

where $\mathrm{NFP}_i$ is the number of optima found in $i$th run, $NP$ is the number of peaks, NSR is the number of successful runs, $\mathrm{FEs}_i$ is the number of FEs required in $i$th run, and $R$ is the number of runs. Note that if an algorithm cannot find all global optima in the $i$th run, the $\mathrm{FEs}_i$ is set as MaxFEs.

Five accuracy levels, $\varepsilon$ = 1.0E-01, $\varepsilon$ = 1.0E-02, $\varepsilon$ = 1.0E-03, $\varepsilon$ = 1.0E-04, and $\varepsilon$ = 1.0E-05, are frequently used in the literatures. However, in this paper, only the last three accuracy levels $\varepsilon$ = 1.0E-03, $\varepsilon$ = 1.0E-04, and $\varepsilon$ = 1.0E-05 are adopted because the accuracy levels $\varepsilon$ = 1.0E-01 and $\varepsilon$ = 1.0E-02 are not accurate enough. Moreover, unless otherwise stated, we mainly discuss the results with accuracy level $\varepsilon$ = 1.0E-04, which is common in [34]–[45] and [52]–[56]. Readers can refer to [63] for more details about the performance measures and the approach for calculating the number of global optima found.

Moreover, the population size is set as in Table I, where the MaxFEs is adopted directly from CEC2015 competition [63]. DE/rand/1 mutation strategy is used in ANDE, and the amplification factor $F$ and crossover rate CR in ANDE are set as 0.9 and 0.1, respectively. The experiments are conducted on a PC with 4 Intel Core i5-7400 3.00-GHz CPUs, 8-GB memory, the Windows 10 64-bit system and MATLAB 2015b edition. All algorithms run 51 times on each function independently and the mean results are reported.

TABLE I
PARAMETER SETTINGS

| Test Function | $N$ | MaxFEs |
|---|---|---|
| $F_1$-$F_5$ | 80 | 5.00E+04 |
| $F_6$ | 100 | 2.00E+05 |
| $F_7$ | 300 | 2.00E+05 |
| $F_8$-$F_9$ | 300 | 4.00E+05 |
| $F_{10}$ | 100 | 2.00E+05 |
| $F_{11}$-$F_{13}$ | 200 | 2.00E+05 |
| $F_{14}$-$F_{20}$ | 200 | 4.00E+05 |

## *B. Comparisons With State-of-the-Art Multimodal Algorithms*

To examine the performance of ANDE, we compare ANDE with the following state-of-the-art 9 DE multimodal algorithms, including CDE [34], SDE [35], Self-CCDE, Self-CSDE [36], NCDE, NSDE [37], LoICDE, LoISDE [38], and PNPCDE [39]. Moreover, we also compare ANDE with other 5 EA multimodal algorithms, including PSO with ring topology (R2PSO and R3PSO) [55], locally informed PSO (LIPS) [56], multimodal EDA (LMCEDA and LMSEDA) [54] and a multiobjective technique MOMMOP [45]. All these competing multimodal algorithms use the same population size $N$ and MaxFEs in Table I, the same as ANDE, to solve CEC2015 problems well and to make the comparisons fair.

Table II summarizes the comparison results between ANDE and other multimodal algorithms in PR and SR at all the three accuracy levels, while the detailed comparison results are shown in Tables S.VIII–S.XIII in the supplementary material for saving space. The best PRs are highlighted by **boldface** for clarity. Besides, Wilcoxon's rank sum test [64] at $\alpha = 0.05$ in PR is used to evaluate the statistical significance between ANDE and the compared multimodal algorithms. The symbols "+," "−," or "≈" indicate that ANDE is significantly better than, worse than, or similar to the compared multimodal algorithms. According to the results in Table II and Tables S.VIII–S.XIII in the supplementary material, we find that with the accuracy level increases, the performance of many multimodal algorithms deteriorates rapidly, except ANDE.

1) *For the First 15 Functions $F_1$–$F_{15}$ With No More Than Three Dimensions ($D = 3$):* ANDE performs significantly better than other algorithms on most functions, no matter on which accuracy level. Take accuracy level $\varepsilon = 1.0\text{E-}04$ as an instance. As we can see from Table S.X in the supplementary material, ANDE dominates SDE, R2PSO, R3PSO, NSDE, Self-CSDE, and LoISDE on at least ten functions; dominates CDE, LIPS, NCDE, PNPCDE, Self-CCDE, LoICDE, LMCEDA, and LMSEDA on at least seven functions. Significantly, the results show that all the competitors cannot outperform ANDE more than one function, except MOMMOP. Note that ANDE performs similarly to MOMMOP on these 15 functions. However, ANDE shows an obvious advantage on the last five functions, which will be discussed next.
2) *For the Last Five Functions $F_{16}$–$F_{20}$ With More Than Three Dimensions ($D > 3$):* ANDE outperforms all the competitors on almost all the five functions. Take accuracy level $\varepsilon = 1.0\text{E-}04$ in Table S.XI in the supplementary material for example again, ANDE dominates CDE, SDE, LIPS, R2PSO, R3PSO, NSDE, PNPCDE, Self-CCDE, Self-CSDE, LoICDE, and LoISDE on all the five functions. It is noteworthy that ANDE dominates MOMMOP and NCDE on four and three out of the five functions, respectively, while is dominated by MOMMOP on only one function. Such observation fully illustrates that ANDE can tackle MMOPs effectively when the dimensions and complexities increase. Even though ANDE performs a little worse than LMCEDA and LMSEDA on these five functions, ANDE still outperforms and shows a great dominance than these two algorithms on $F_1$–$F_{15}$, especially on $F_6$–$F_9$, where exist numerous peaks as shown in Table S.X in the supplementary material and discussed above.

Therefore, ANDE is much more promising and suitable than these state-of-the-art multimodal algorithms. The superiority and dominance of ANDE are increasingly obvious with the required accuracy level increases. Besides, ANDE can maintain its dominance when dealing with the MMOPs with a larger number of peaks and with higher dimensions or complexities.

For the comparison in AveFEs, since it is no sense to evaluate this performance metric on functions that no algorithm has a successful run, therefore, we only select the first five functions $F_1$–$F_5$ for investigations. Meanwhile, we only compare ANDE with CDE, NCDE, Self-CCDE, LoICDE, PNPCDE, MOMMOP, LMCEDA, and LMSEDA because they can achieve the comparable performance. Table S.XIV in the supplementary material shows the comparison results in AveFEs at all these three accuracy levels, and the best results are emphasized by **boldface**. As we can see, ANDE generally achieves a faster convergence speed than other multimodal algorithms at accuracy level $\varepsilon = 1.0\text{E-}03$, except NCDE. That may be due to the CPA in ANDE can speculate the approximate position of the potential optima and accelerate convergence speed. However, when the accuracy level increases from $\varepsilon = 1.0\text{E-}03$ to $\varepsilon = 1.0\text{E-}05$, ANDE performs a little worse than both NCDE and LMSEDA. That may be explained by the TLLS in ANDE, which will consume more FEs to get more accurate results. However, the more FEs brought by TLLS can be compensated by the high solution accuracy and better PR and SR results, shown above. Even so, ANDE still maintains a faster convergence speed in locating all the global optima at all the three accuracy levels than other algorithms.

From all the above comparison results (including PR, SR, and AveFEs), we can observe that ANDE can locate more global optima within fewer FEs, especially when solving the problems with high complexities and numerous global optima, which fully proves the superiority of ANDE. The advantages of ANDE may be due to its novel techniques in ANDE: 1) APC; 2) CPA; 3) TLLS; and 4) PCA. The first APC niching technique can generate suitable niches automatically to match the landscape of the problem, without sensitive parameter and extra FEs. The second CPA technique can effectively predict the rough position of the potential optimum in each niche, which can provide a proper guidance of evolution and lead to

TABLE II
SUMMARIZED RESULTS IN PR AND SR ON PROBLEMS AT ALL THE THREE ACCURACY LEVELS

| **ANDE** V.S. | | CDE | SDE | LIPS | R2PSO | R3PSO | NCDE | NSDE | PNPCDE |
|---|---|---|---|---|---|---|---|---|---|
| $\varepsilon$=1.0E-03 | + | 13 | 18 | 14 | 16 | 16 | 11 | 18 | 14 |
| | - | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | ≈ | 7 | 2 | 6 | 4 | 4 | 9 | 2 | 6 |
| $\varepsilon$=1.0E-04 | + | 13 | 19 | 14 | 16 | 16 | 11 | 18 | 14 |
| | - | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | ≈ | 7 | 1 | 5 | 4 | 4 | 9 | 2 | 6 |
| $\varepsilon$=1.0E-05 | + | 14 | 19 | 14 | 16 | 17 | 12 | 18 | 15 |
| | - | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | ≈ | 6 | 1 | 5 | 4 | 3 | 8 | 2 | 5 |
| **ANDE** V.S. | | Self-CCDE | Self-CSDE | LoICDE | LoISDE | LMCEDA | LMSEDA | MOMMOP | |
| $\varepsilon$=1.0E-03 | + | 11 | 15 | 12 | 18 | 7 | 7 | 8 | |
| | - | 1 | 0 | 0 | 0 | 3 | 3 | 4 | |
| | ≈ | 8 | 5 | 8 | 2 | 10 | 10 | 8 | |
| $\varepsilon$=1.0E-04 | + | 13 | 17 | 13 | 18 | 8 | 7 | 7 | |
| | - | 1 | 0 | 0 | 0 | 3 | 3 | 4 | |
| | ≈ | 6 | 3 | 7 | 2 | 9 | 10 | 9 | |
| $\varepsilon$=1.0E-05 | + | 13 | 18 | 14 | 18 | 8 | 7 | 7 | |
| | - | 1 | 0 | 0 | 0 | 2 | 3 | 4 | |
| | ≈ | 6 | 2 | 6 | 2 | 10 | 10 | 9 | |

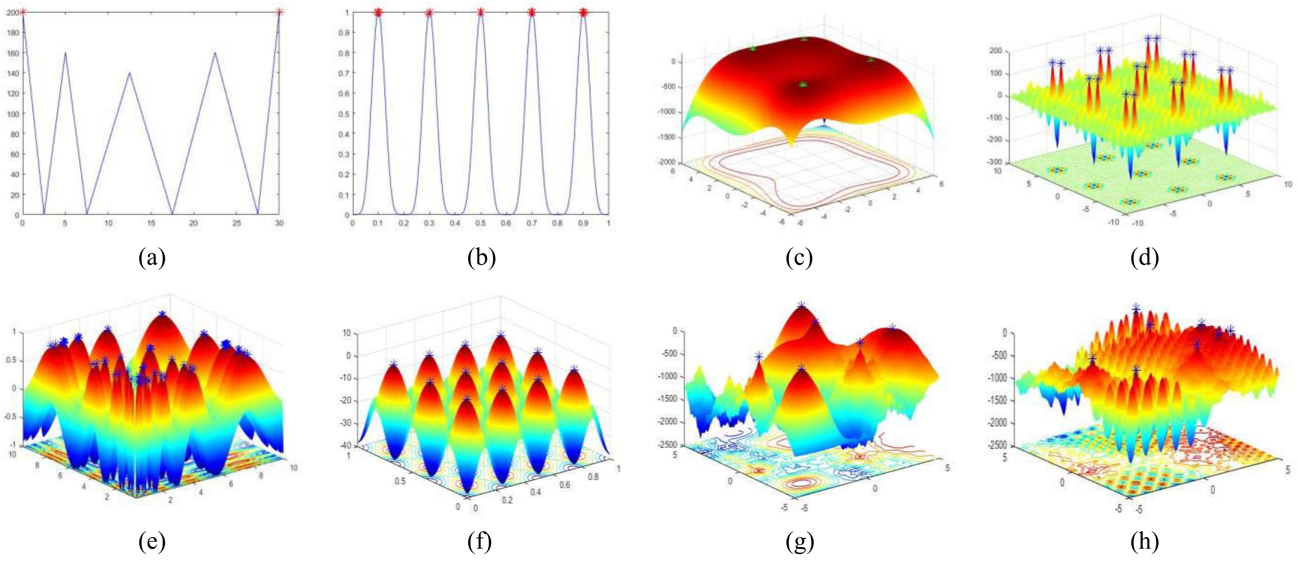


Fig. 6. Final population distribution on eight selected functions. (a) $F_1$. (b) $F_2$. (c) $F_4$. (d) $F_6$. (e) $F_7$. (f) $F_{10}$. (g) $F_{11}$. (h) $F_{12}$.

fast convergence. The third TLLS strategy aims to refine the solution, which enhances the exploitation ability of ANDE and increases the accuracy of all the global optima. In addition, the dimensionality reduction technique using PCA can achieve better niching, which is more suitable for dealing with the complicated and high dimensional MMOPs. The effects of these techniques will be further discussed in Section IV-D. Therefore, equipped with the novel techniques mentioned above, ANDE outperforms other state-of-the-art algorithms.

To have a more intuitive view of the ANDE, we display the final population distribution on eight selected functions in Fig. 6.

As we can see from Fig. 6, ANDE can locate all the global optima, even there exist many global optima, such as the $F_6$ and $F_7$ in Fig. 6(d) and (e). For the more complex functions which contain massive local optima, such as the $F_{11}$ and $F_{12}$ in Fig. 6(g) and (h), ANDE maintains the exploration ability and population diversity, also avoids local optima and locates all the global optima.

## C. Comparison With Winner of CEC2015 Competition

To further demonstrate the superiority of ANDE, in this section, we compare ANDE with the winner of the CEC2015 competition on multimodal optimization, NMMSO [64]. For fair comparison, we directly cite the mean results of NMMSO from the CEC2015 competition (https://github.com/mikeagn/CEC2013/tree/master/NichingCompetition2015FinalData).

The detailed comparison results with respect to PR and SR between ANDE and NMMSO on all the three accuracy levels are listed in Table III. The best PR results are highlighted in **boldface**. Due to the lack of the detailed results of NMMSO in each run, we cannot conduct the Wilcoxon's rank sum test to evaluate the statistical significance between ANDE and NMMSO. Therefore, whether ANDE is better than (+), worse than (−), or similar to (≈) NMMSO is just measured by the values of PR.

From Table III, we find that ANDE still keeps its promising performance compared with NMMSO. First, at the accuracy

TABLE III
EXPERIMENTAL RESULTS IN PR AND SR BETWEEN ANDE AND NMMSO ON PROBLEMS $f_1$–$f_{20}$ AT ALL THE THREE ACCURACY LEVELS

| $\varepsilon$=1.0E-03 | | | | | $\varepsilon$=1.0E-04 | | | | | $\varepsilon$=1.0E-05 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Func | ANDE | | NMMSO | | Func | ANDE | | NMMSO | | Func | ANDE | | NMMSO | |
| | PR | SR | PR | SR | | PR | SR | PR | SR | | PR | SR | PR | SR |
| $F_1$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_1$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_1$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 |
| $F_2$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_2$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_2$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 |
| $F_3$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_3$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_3$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 |
| $F_4$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_4$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_4$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 |
| $F_5$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_5$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_5$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 |
| $F_6$ | **1.000** | 1.000 | 0.992(+) | 0.880 | $F_6$ | **1.000** | 1.000 | 0.992(+) | 0.880 | $F_6$ | **1.000** | 1.000 | 0.000(+) | 0.000 |
| $F_7$ | 0.936 | 0.176 | **1.000(-)** | 1.000 | $F_7$ | 0.933 | 0.176 | **1.000(-)** | 1.000 | $F_7$ | 0.941 | 0.196 | **1.000(-)** | 1.000 |
| $F_8$ | **0.947** | 0.078 | 0.922(+) | 0.020 | $F_8$ | **0.944** | 0.078 | 0.899(+) | 0.020 | $F_8$ | **0.948** | 0.039 | 0.870(+) | 0.000 |
| $F_9$ | 0.516 | 0.000 | **0.978(-)** | 0.120 | $F_9$ | 0.512 | 0.000 | **0.978(-)** | 0.120 | $F_9$ | 0.506 | 0.000 | **0.978(-)** | 0.120 |
| $F_{10}$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_{10}$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 | $F_{10}$ | **1.000** | 1.000 | **1.000(≈)** | 1.000 |
| $F_{11}$ | **1.000** | 1.000 | 0.990(+) | 0.940 | $F_{11}$ | **1.000** | 1.000 | 0.990(+) | 0.940 | $F_{11}$ | **1.000** | 1.000 | 0.990(+) | 0.940 |
| $F_{12}$ | **1.000** | 1.000 | 0.995(+) | 0.960 | $F_{12}$ | **1.000** | 1.000 | 0.993(+) | 0.940 | $F_{12}$ | **1.000** | 1.000 | 0.990(+) | 0.920 |
| $F_{13}$ | 0.771 | 0.078 | **0.983(-)** | 0.900 | $F_{13}$ | 0.686 | 0.000 | **0.983(-)** | 0.900 | $F_{13}$ | 0.686 | 0.000 | **0.983(-)** | 0.900 |
| $F_{14}$ | 0.667 | 0.000 | **0.723(-)** | 0.020 | $F_{14}$ | 0.667 | 0.000 | **0.720(-)** | 0.000 | $F_{14}$ | 0.667 | 0.000 | **0.720(-)** | 0.000 |
| $F_{15}$ | **0.645** | 0.000 | 0.642(+) | 0.000 | $F_{15}$ | **0.632** | 0.000 | **0.632(≈)** | 0.000 | $F_{15}$ | **0.632** | 0.000 | **0.632(≈)** | 0.000 |
| $F_{16}$ | **0.667** | 0.000 | 0.660(+) | 0.000 | $F_{16}$ | **0.667** | 0.000 | 0.660(+) | 0.000 | $F_{16}$ | **0.667** | 0.000 | 0.660(+) | 0.000 |
| $F_{17}$ | 0.397 | 0.000 | **0.470(-)** | 0.000 | $F_{17}$ | 0.397 | 0.000 | **0.468(-)** | 0.000 | $F_{17}$ | 0.397 | 0.000 | **0.460(-)** | 0.000 |
| $F_{18}$ | **0.654** | 0.000 | 0.650(+) | 0.000 | $F_{18}$ | **0.654** | 0.000 | 0.650(+) | 0.000 | $F_{18}$ | **0.650** | 0.000 | **0.650(≈)** | 0.000 |
| $F_{19}$ | 0.363 | 0.000 | **0.457(-)** | 0.000 | $F_{19}$ | 0.363 | 0.000 | **0.450(-)** | 0.000 | $F_{19}$ | 0.363 | 0.000 | **0.437(-)** | 0.000 |
| $F_{20}$ | **0.250** | 0.000 | 0.172(+) | 0.000 | $F_{20}$ | **0.248** | 0.000 | 0.172(+) | 0.000 | $F_{20}$ | **0.248** | 0.000 | 0.172(+) | 0.000 |
| +(ANDE is better) | | | 8 | | +(ANDE is better) | | | 7 | | +(ANDE is better) | | | 6 | |
| -(ANDE is worse) | | | 6 | | -(ANDE is worse) | | | 6 | | -(ANDE is worse) | | | 6 | |
| ≈ | | | 6 | | ≈ | | | 7 | | ≈ | | | 8 | |

level $\varepsilon$ = 1.0E-03 and $\varepsilon$ = 1.0E-04, ANDE dominates NMMSO on eight and seven functions, respectively, while only dominated by NMMSO on six functions. Second, at the last accuracy level $\varepsilon$ = 1.0E-05, ANDE achieves the equivalent performance compared to NMMSO, where the number of functions that ANDE dominates NMMSO is the same to the number of functions that ANDE is dominated by NMMSO. However, we can see that ANDE performs much better than NMMSO on $F_6$ and $F_8$ at the accuracy level $\varepsilon$ = 1.0E-05. In particularly, on $F_6$, ANDE can locate all the global optima in all the runs with 1.000 for PR and 1.000 for SR, while NMMSO cannot locate any global optima in each run with 0.000 for PR and 0.000 for SR. Even on the functions where ANDE is dominated by NMMSO, ANDE still achieves the comparable performance to NMMSO. For example, at the accuracy level $\varepsilon$ = 1.0E-05, on $F_{14}$, $F_{17}$, and $F_{19}$, ANDE achieves the PR with 0.667, 0.397, and 0.363, respectively, which is very close to the PR in NMMSO with 0.720, 0.460, and 0.437, respectively. Third, when dealing with the complicated problems with higher dimensions $F_{16}$–$F_{20}$, especially with 20$D$ in $F_{20}$, ANDE performs better than NMMSO, no matter on which accuracy level, further showing the superiority of ANDE for dealing with the high complexities or high dimensional problems.

Overall, we can see that ANDE is competitive or even better than the winner of the CEC2015 competition.

## D. Influence of Each Component in ANDE

The main components in ANDE are: 1) APC; 2) CPA; 3) TLLS; and 4) PCA. Herein, we will discuss the influence of each component in ANDE.

*1) APC:* Clustering techniques have been applied in crowding and speciation niching in [36] and [37]. However, these two clustering niching methods both have a parameter, cluster size $M$, which will directly affect the performance of algorithm. Herein, to investigate the effectiveness of the new proposed niching method, the ANDE variants, where the APC-based niching is replaced by the crowding or speciation clustering niching is compared with ANDE on $F_1$–$F_{20}$. The cluster size $M$ is set as 5 or 10, which are also frequently used in [36] and [37]. The ANDE with clustering niching of crowding or speciation and with a fixed cluster size $M = \text{a}$ is termed as ANDE-C(a) or ANDE-S(a), respectively. For example, ANDE with crowding clustering and with cluster size $M = 10$ is denoted as ANDE-C(10). The comparison results between ANDE and its variants in PR and SR at accuracy level $\varepsilon$ = 1.0E-04 are shown in Table S.XV in the supplementary material.

As we can see, on functions $F_1$–$F_6$ and $F_{10}$, all the five competitors can locate all the global optima. The ANDE-C(10) performs the best on the functions $F_7$–$F_9$. However, on $F_{11}$–$F_{15}$, with a huge number of local optima, ANDE gradually shows its tremendous advantage. ANDE obtains the best results on both PR and SR performance metrics on all these complex functions, while the other algorithms can only obtain similar results on $F_{12}$ and $F_{14}$. ANDE-C(10) also obtains equivalent performance on $F_{11}$, but is significantly worse than ANDE on $F_{13}$ and $F_{15}$. Particularly, when the dimensions and complexities increase, especially on $F_{18}$–$F_{20}$ with 10 and 20 dimensions, the superiority of ANDE is more obvious.

Besides, different niching strategy and different cluster size $M$ are suitable for different problems. For instance, the crowding cluster niching strategy may be suitable for the problems with numerous global optima, such as on $F_7$–$F_9$. While the speciation cluster niching strategy is probably suitable for the problems with high complexities, such as on $F_{17}$–$F_{19}$. In addition, the large cluster size contains a wide range of information, which may be appropriate for diversity maintaining, performs well on $F_7$–$F_9$. While the small cluster size

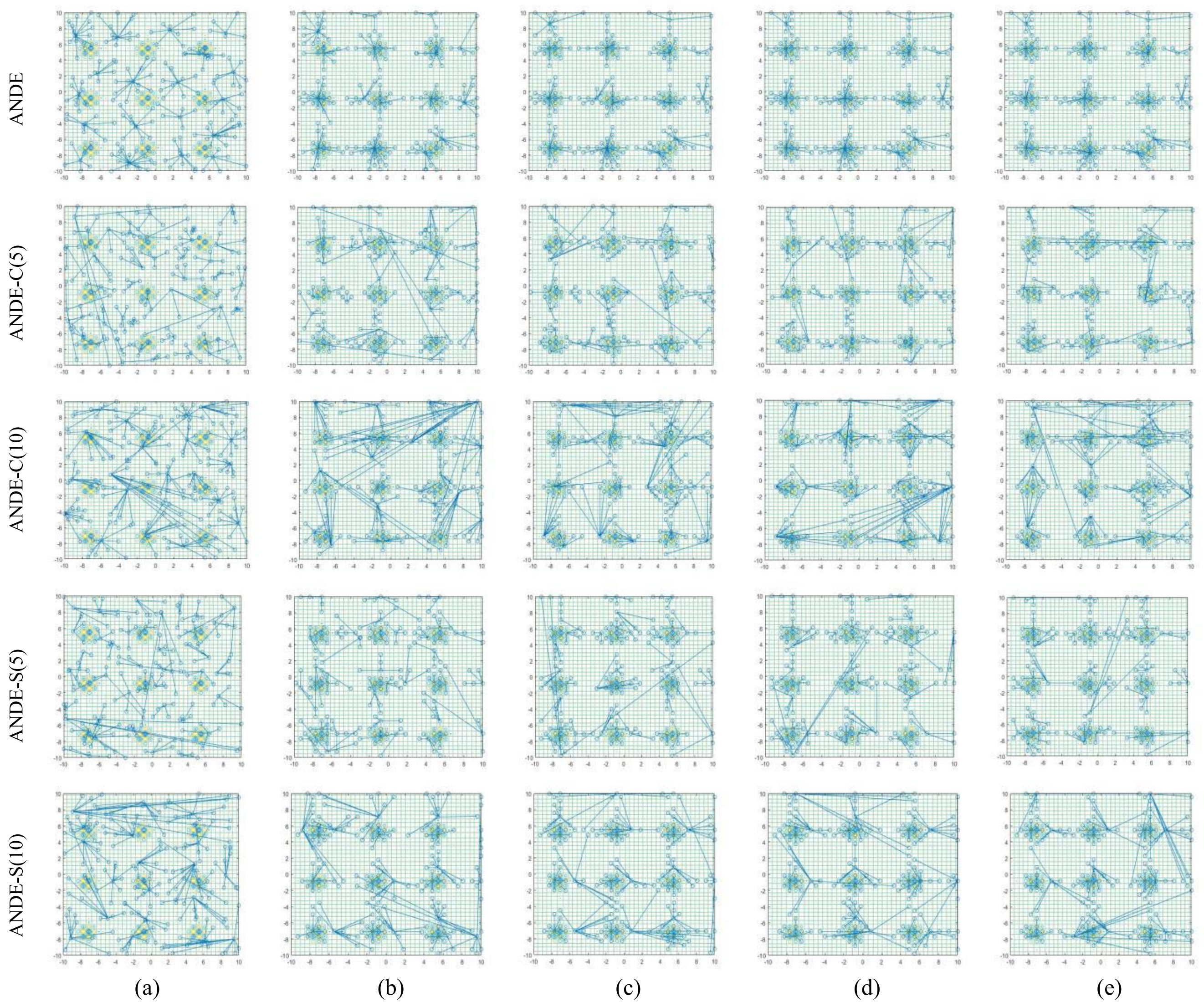


Fig. 7. Population distribution on $F_6$ using different niching strategies after a certain number of FEs. (a) FEs = 0. (b) FEs = 20 000. (c) FEs = 40 000. (d) FEs = 60 000. (e) FEs = 80 000.

covers narrow areas, which may be suitable for exploitation, performs well on $F_{13}$.

Overall, neither a small nor a large cluster size and neither crowding clustering nor speciation clustering is attractive, and these two clustering niching methods are both sensitive to the cluster size and lose their feasibilities on some sophisticated functions. However, without any heuristic information and any sensitive parameters, ANDE still generally outperforms ANDE-S(5), ANDE-S(10), ANDE-C(5), and ANDE-C(10) on 8, 7, 7, and 5 functions, while is dominated by these four variants on only 1, 2, 3, and 3 functions, respectively.

Moreover, to further present the cluster behaviors of ANDE and ANDE variants, we draw the population distribution during the evolutionary process on the contour landscape of $F_6$, shown in Fig. 7. The line connected between two individuals means they belong to the same cluster/niche. As we can see, only ANDE can produce the stable niches to match the landscape of the problem, while the niching results in other variants are in a mess, which may mislead the evolution.

From Table S.XV in the supplementary material and Fig. 7, we find that the APC-based niching strategy is almost not affected by the random initialized solutions in the search space in different runs and is more suitable for solving MMOPs than crowding clustering and speciation clustering. On the one hand, it can form stable niches automatically for better evolution, on the other hand, it does not use the sensitive parameter such as the number of clusters or the cluster size $M$ and does not take up any extra FEs.

As for the computational time shown in Table S.XV in the supplementary material, we can see that ANDE generally consumes more computational time than its variants with other clustering niching techniques. That is due to the APC-based niching actually involves the iterative process. Although the APC induces some extra computational time to ANDE, it also helps ANDE form stable niches automatically for better evolution and locate all the global optima more accurately. As a result, the improvements in performance are much worth since the increased computational time can be compensated by the stable niching result and high solution accuracy.

*2) CPA:* In this part, to investigate the influence of CPA, the ANDE variant without CPA, termed as ANDE-noCPA is compared with ANDE. Since CPA is used to estimate the rough position of the potential peak to accelerate convergence and save FEs, we only use the performance metric AveFEs herein to show the effectiveness of CPA. Similarly, since it is no sense to evaluate this performance metric on complicated functions where ANDE cannot achieve a successful run, only the first five functions $F_1$–$F_5$ are used for investigations. The comparison results at all the three accuracy levels are listed in Table S.XVI in the supplementary material. From Table S.XVI, in the supplementary material, we find that ANDE can achieve faster convergence speed and save FEs effectively than ANDE-noCPA on three functions on accuracy levels $\varepsilon = 1.0\text{E-}03$ and $\varepsilon = 1.0\text{E-}04$, which fully illustrates the advantage of the peak prediction. When accuracy level increases to $\varepsilon = 1.0\text{E-}05$, the superiority of ANDE is not significantly obvious. That may be due to the fact that both ANDE and ANDE-noCPA use the TLLS method to refine the solution accuracy, which will also consume some FEs. Even so, ANDE still achieves the fast search process than ANDE-noCPA on $F_4$ and $F_5$, while ANDE-noCPA cannot surpass ANDE on any functions. As a result, we may reasonably come to the conclusion that CPA can effectively speculate the appropriate position of potential optima and save FEs.

Table S.XVI, in the supplementary material, also shows the time required to find all the global optima to test the time influence of CPA. As we can see, ANDE generally consumes less computational time than its variant without CPA component. That is due to the CPA in ANDE can speculate the approximate position of the potential optima and accelerate convergence speed. With the help of CPA, we can locate all the global optima more quickly by using fewer FEs, which will save the computational time.

*3) TLLS:* The local search method is mainly to increase the accuracy of solution and enhance the exploitation ability of algorithm. Herein, we take a close observation at the influence of the local search and the two-level scheme. We denote the ANDE without local search and with only niche-level local search as ANDE-noLS and ANDE-onlyN, respectively. The ANDE with only niche-level local search is to sample the individuals only around the niche seed $\boldsymbol{x}_{si}$ of the $i$th niche $S_i$. The complete niche-level local search is shown in Algorithm S1 in the supplementary material. Table S.XVII, in the supplementary material, presents the comparison results with respect to PR and SR at accuracy level $\varepsilon = 1.0\text{E-}04$. We first illustrate the effectiveness of local search. According to the comparison results, we find both ANDE and ANDE-onlyN can surpass than ANDE-noLS on many functions, such as $F_6$–$F_9$, $F_{11}$–$F_{13}$, and $F_{17}$–$F_{20}$. As a whole, the local search is extremely useful for ANDE, which can increase the accuracy of solutions. Next we illustrate the advantage of the two-level scheme. From the comparison results between ANDE and ANDE-onlyN, ANDE still outperforms ANDE-onlyN on many functions, such as $F_6$–$F_9$ and $F_{11}$–$F_{12}$. That may be due to the fact that some peaks are covered by the same niche, so that some peaks cannot improve their accuracy if performing local search only on niche-level. However, as we all known, to get more accurate results, the local search scheme will take up some extra FEs. On $F_{17}$ and $F_{19}$, ANDE-onlyN performs slightly better than ANDE, which may be due to the TLLS has to consume more extra FEs on both the niche and individual levels than the local search on only niche level. Even so, ANDE still shows its superiority on other functions.

Besides, ANDE generally consumes less computational time than its variants without local search and with only niche-level local search, shown in Table S.XVII in the supplementary material. That is due to the TLLS has to be allocated FEs to further enhance the exploitation ability and improve the solution accuracy. As a result, the number of generations in evolutionary process will decrease due to some FEs are allocated for TLLS. Therefore, fewer APC-based niching and evolutionary operators are needed, which can save the computational time.

In short, we can conclude that the TLLS is beneficial for ANDE in locating more global optima and increasing accuracy.

*4) PCA:* ANDE shows its prominent advantages when dealing with MMOPs, which can be seen from Sections IV-B and IV-C. However, when dealing with the high dimensional MMOPs, the niching method based on APC is somehow affected by the dimensions of problems. Therefore, PCA is used here to achieve dimensionality reduction for better niching. In that way, in order to study the usefulness of PCA, ANDE is compared with the ANDE variant without PCA, termed as ANDE-noPCA. We only choose the last five functions $F_{16}$–$F_{20}$ to compare because PCA is only used for problems with more than three dimensions ($D > 3$). The detailed experimental results in PR and SR at accuracy level $\varepsilon = 1.0\text{E-}04$ are shown in Table S.XVIII in the supplementary material. Obviously, on $F_{16}$ and $F_{17}$, there is no significant difference between ANDE and ANDE-noPCA, which illustrates the property of APC is not severely affected when the dimension is less than or equal to 5. However, ANDE-noPCA deteriorates rapidly on $F_{18}$–$F_{20}$, where the dimension increases to 10 or 20. While ANDE maintains a stable performance and dominates ANDE-noPCA on these three functions. Such an observation directly shows the effectiveness of PCA, which can achieve the dimensionality reduction for better niching.

Moreover, ANDE generally consumes less computational time than its variant without PCA component. That is due to the PCA in ANDE can achieve the dimensional reduction, which helps clustering faster and save computational time.

## *E. FEs Consumed of Each Component in ANDE*

We further discuss the FEs consumed of each component in ANDE. The detailed experimental results are listed in Table S.XIX and Fig. S1 in the supplementary material.

The first component of ANDE is APC for population partition. The APC is an automatic niching technique, which can form clusters/niches automatically without any extra FEs. So the APC does not consume any extra FEs.

After using APC to partition the population into suitable clusters/niches automatically to locate different peak regions, the DE evolutionary operators are performed within each niche. Each individual will evolve and consume 1 FE. However, for the niches with fewer than four individuals, the DE operators are not executed because DE must have at least four individuals. Therefore, the FEs consumed in DE is about $N$ or a little fewer than $N$ in each generation.

Then, CPA is further developed to estimate the contour landscape of each niche. Each niche will consume 1 FE to estimate the optimum. However, if the legal interpolated points are fewer than three, the contour cannot be drawn. In other words, if the current niche has fewer than four individuals, CPA is not used. Therefore, the FEs consumed in CPA is about $n$ or a little fewer than $n$ ($n$ is the number of niches) in each generation.

At last, in order to enhance the exploitation ability and improve the solution accuracy, TLLS strategy is further performed after the CPA. About 50% niches will execute the niche-level local search according to the probability $P_i$. If the current niche $i$ satisfies the probability $P_i$ to do local search, about 50% individuals will execute the individual-level local search in the current niche according to the probability $P_{ik}$. As a result, there are about 25% (50% $\times$ 50%) individuals will execute TLLS and each individual will consume 2 FEs to sample 2 individuals. Therefore, the FEs consumed in TLLS is about $N/2$ (25% $\times$ 2) in each generation.

Additionally, we also use a PCA component for dimension reduction to assistant the APC-based niching in high dimensional problems. However, this component does not consume any extra FEs.

To sum, the number of FEs consumed in each generation of ANDE is approximately $1.5N + n$. However, when comparing ANDE with other algorithms, their termination criteria are set the same by the same MaxFEs. The comparison results show that ANDE can achieve better performance using the same computational budget, suggesting that the extra components (i.e., APC, CPA, and TLLS) can promote the efficiency of ANDE despite of certain computational load.

## V. Conclusion

In this paper, the DE with APC, CPA, and TLLS, termed as ANDE is proposed for solving MMOPs. First, we proposed a new automatic niching strategy using APC for population partition, which can relieve the algorithm from sensitivity of parameter such as the cluster size or the number of clusters and form stable niches automatically to match the landscape of the problems without any extra FEs. Second, CPA can predict the rough position of the potential peak in each niche, and then provide a proper guidance for evolution, which can accelerate the convergence speed. Third, the TLLS is embed for enhancing the exploitation ability of algorithm and improving the accuracy of solutions. In addition, for MMOPs with high dimensions, PCA is utilized to achieve dimension reduction for better niching.

Based on these techniques, ANDE can find a balance between diversity and convergence, leading to a competitive feasibility and effectiveness when tackling with MMOPs. The experimental results fully show the superiority of ANDE when compared with other 15 state-of-the-art multimodal algorithms and the winner of CEC2015 competition, which can find more global optima using fewer FEs, and the dominance of ANDE becomes increasingly obvious with the increasing accuracy level.

However, APC-based niching technique in ANDE will generally consume more computational time compared with other clustering niching techniques. Even so, APC also helps ANDE form stable niches automatically for better evolution and locate all the global optima more accurately. Therefore, the improvements in performance are much worth since the increased computational time can be compensated by the stable niching result and high solution accuracy.

Even though the performance of ANDE shows its apparent advantage when dealing with MMOPs, with the complexity and dimension increases, ANDE still cannot locate all the global optima. For future work, we wish to further improve the performance of ANDE on more complex MMOPs with higher dimensions and large number of global or local peaks. Moreover, we wish to apply ANDE in dynamic multimodal environments [66]–[68], and to explore the information sharing mechanism in algorithm design [69]–[71].

## References

[1] K.-C. Wong, K.-S. Leung, and M.-H. Wong, "Protein structure prediction on a lattice model via multimodal optimization techniques," in *Proc. Genet. Evol. Comput. Conf.*, Portland, OR, USA, 2010, pp. 155–162.

[2] D.-K. Woo, J.-H. Choi, M. Ali, and H.-K. Jung, "A novel multimodal optimization algorithm applied to electromagnetic optimization," *IEEE Trans. Magn.*, vol. 47, no. 6, pp. 1667–1673, Jun. 2011.

[3] D.-Z. Tan, W.-N. Chen, J. Zhang, and W.-J. Yu, "Fast pedestrian detection using multimodal estimation of distribution algorithms," in *Proc. Genet. Evol. Comput. Conf.*, Berlin, Germany, 2017, pp. 1248–1255.

[4] E. Cuevas, M. González, D. Zaldívar, and M. Pérez-Cisneros, "Multi-ellipses detection on images inspired by collective animal behavior," *Neural Comput. Appl.*, vol. 24, no. 5, pp. 1019–1033, 2014.

[5] K. Deb and A. Srinivasan, "Innovization: Innovating design principles through optimization," in *Proc. Genet. Evol. Comput. Conf.*, Seattle, WA, USA, 2006, pp. 1629–1636.

[6] X. Y. Zhang *et al.*, "Kuhn–Munkres parallel genetic algorithm for the set cover problem and its application to large-scale wireless sensor networks," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 695–710, Oct. 2016.

[7] Z.-J. Wang, Z.-H. Zhan, and J. Zhang, "Solving the energy efficient coverage problem in wireless sensor networks: A distributed genetic algorithm approach with hierarchical fitness evaluation," *Energies*, vol. 11, no. 12, p. 3526, 2018.

[8] T. Friedrich, T. Kötzing, M. S. Krejca, and A. M. Sutton, "The compact genetic algorithm is efficient under extreme Gaussian noise," *IEEE Trans. Evol. Comput.*, vol. 21, no. 3, pp. 477–490, Jun. 2017.

[9] M. A. Rashid, F. Khatib, M. T. Hoque, and A. Sattar, "An enhanced genetic algorithm for Ab initio protein structure prediction," *IEEE Trans. Evol. Comput.*, vol. 20, no. 4, pp. 627–644, Aug. 2016.

[10] Z. Y. Wang *et al.*, "A modified ant colony optimization algorithm for network coding resource minimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 325–342, Jun. 2016.

[11] Z.-G. Chen *et al.*, "Multiobjective cloud workflow scheduling: A multiple populations ant colony system approach," *IEEE Trans. Cybern.*, to be published. doi: 10.1109/TCYB.2018.2832640.

[12] F. F. Zheng, A. C. Zecchin, J. P. Newman, H. R. Maier, and G. C. Dandy, "An adaptive convergence-trajectory controlled ant colony optimization algorithm with application to water distribution system design problems," *IEEE Trans. Evol. Comput.*, vol. 21, no. 5, pp. 773–791, Oct. 2017.

[13] X.-F. Liu *et al.*, "An energy efficient ant colony system for virtual machine placement in cloud computing," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 113–128, Feb. 2018.
[14] J. Wang, K. Tang, J. A. Lozano, and X. Yao, "Estimation of the distribution algorithm with a stochastic local search for uncertain capacitated arc routing problems," *IEEE Trans. Evol. Comput.*, vol. 20, no. 1, pp. 96–109, Feb. 2016.
[15] A. M. Zhou, J. Y. Sun, and Q. F. Zhang, "An estimation of distribution algorithm with cheap and expensive local search methods," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 807–822, Dec. 2015.
[16] X. L. Liang, H. P. Chen, and J. A. Lozano, "A Boltzmann-based estimation of distribution algorithm for a general resource scheduling model," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 793–806, Dec. 2015.
[17] W.-N. Chen *et al.*, "Particle swarm optimization with an aging leader and challengers," *IEEE Trans. Evol. Comput.*, vol. 17, no. 2, pp. 241–258, Apr. 2013.
[18] Z.-H. Zhan *et al.*, "Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 445–463, Apr. 2013.
[19] M. R. Bonyadi and Z. Michalewicz, "Impacts of coefficients on movement patterns in the particle swarm optimization algorithm," *IEEE Trans. Evol. Comput.*, vol. 21, no. 3, pp. 378–390, Jun. 2017.
[20] S. Zhang *et al.*, "Optimal computing budget allocation for particle swarm optimization in stochastic optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 206–219, Apr. 2017.
[21] X.-F. Liu *et al.*, "Coevolutionary particle swarm optimization with bottleneck objective learning strategy for many-objective optimization," *IEEE Trans. Evol. Comput.*, to be published. doi: 10.1109/TEVC.2018.2875430.
[22] M. R. Bonyadi and Z. Michalewicz, "Stability analysis of the particle swarm optimization without stagnation assumption," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 814–819, Oct. 2016.
[23] Z.-H. Zhan, J. Zhang, Y. Li, and Y.-H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 15, no. 6, pp. 832–847, Dec. 2011.
[24] R. A. Sarker, S. M. Elsayed, and T. Ray, "Differential evolution with dynamic parameters selection for optimization problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 5, pp. 689–707, Oct. 2014.
[25] X.-F. Liu *et al.*, "Historical and heuristic-based adaptive differential evolution," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published. doi: 10.1109/TSMC.2018.2855155.
[26] Z.-H. Zhan *et al.*, "Cloudde: A heterogeneous differential evolution algorithm and its distributed cloud version," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 3, pp. 704–716, Mar. 2017.
[27] Y. Wang, B. Xu, G. Sun, and S. Yang, "A two-phase differential evolution for uniform designs in constrained experimental domains," *IEEE Trans. Evol. Comput.*, vol. 21, no. 5, pp. 665–680, Oct. 2017.
[28] N. R. Sabar, J. Abawajy, and J. Yearwood, "Heterogeneous cooperative co-evolution memetic differential evolution algorithm for big data optimization problems," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 315–327, Apr. 2017.
[29] V. Santucci, M. Baioletti, and A. Milani, "Algebraic differential evolution algorithm for the permutation flowshop scheduling problem with total flowtime criterion," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 682–694, Oct. 2016.
[30] N. M. Hamza, D. L. Essam, and R. A. Sarker, "Constraint consensus mutation-based differential evolution for constrained optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 447–459, Jun. 2016.
[31] X. Qiu, J.-X. Xu, K. C. Tan, and H. A. Abbass, "Adaptive cross-generation differential evolution operators for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 232–244, Apr. 2016.
[32] S.-M. Guo, C.-C. Yang, P.-H. Hsu, and J. S.-H. Tsai, "Improving differential evolution with a successful-parent-selecting framework," *IEEE Trans. Evol. Comput.*, vol. 19, no. 5, pp. 717–730, Oct. 2015.
[33] L. X. Tang, Y. Dong, and J. Y. Liu, "Differential evolution with an individual-dependent mechanism," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 560–574, Aug. 2015.
[34] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, Portland, OR, USA, 2004, pp. 1382–1389.
[35] X. Li, "Efficient differential evolution using speciation for multimodal function optimization," in *Proc. Genet. Evol. Comput. Conf.*, Washington, DC, USA, 2005, pp. 873–880.
[36] W. Gao, G. G. Yen, and S. Liu, "A cluster-based differential evolution with self-adaptive strategy for multimodal optimization," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1314–1327, Aug. 2014.
[37] B. Y. Qu, P. N. Suganthan, and J. J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 5, pp. 601–614, Oct. 2012.
[38] S. Biswas, S. Kundu, and S. Das, "Inducing niching behavior in differential evolution through local information sharing," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 246–263, Apr. 2015.
[39] S. Biswas, S. Kundu, and S. Das, "An improved parent-centric mutation with normalized neighborhoods for inducing niching behavior in differential evolution," *IEEE Trans. Cybern.*, vol. 44, no. 10, pp. 1726–1737, Oct. 2014.
[40] Z.-J. Wang *et al.*, "Dual-strategy differential evolution with affinity propagation clustering for multimodal optimization problems," *IEEE Trans. Evol. Comput.*, vol. 22, no. 6, pp. 894–908, Dec. 2018.
[41] S. Hui and P. N. Suganthan, "Ensemble and arithmetic recombination-based speciation differential evolution for multimodal optimization," *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 64–74, Jan. 2016.
[42] B.-Y. Qu and P. N. Suganthan, "Novel multimodal problems and differential evolution with ensemble of restricted tournament selection," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, 2010, pp. 1–7.
[43] M. G. Epitropakis, X. Li, and E. K. Burke, "A dynamic archive niching differential evolution algorithm for multimodal optimization," in *Proc. IEEE Congr. Evol. Comput.*, Cancén, Mexico, 2013, pp. 79–86.
[44] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 666–685, Oct. 2013.
[45] Y. Wang, H.-X. Li, G. G. Yen, and W. Song, "MOMMOP: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 830–843, Apr. 2015.
[46] R. K. Ursem, "Multinational evolutionary algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Washington, DC, USA, 1999, pp. 1633–1640.
[47] J. Yao, N. Kharma, and Y. Q. Zhu, "On clustering in evolutionary computation," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 1752–1759.
[48] J.-P. Li, M. E. Balazs, G. T. Parks, and P. J. Clarkson, "A species conserving genetic algorithm for multimodal function optimization," *Evol. Comput.*, vol. 10, no. 3, pp. 207–234, Sep. 2002.
[49] J. Gan and K. Warwick, "Dynamic niche clustering: A fuzzy variable radius niching technique for multimodal optimisation in GAs," in *Proc. IEEE Congr. Evol. Comput.*, Seoul, South Korea, 2001, pp. 215–222.
[50] A. Pétrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Nagoya, Japan, 1996, pp. 798–803.
[51] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proc. Int. Conf. Genet. Algorithms*, Cambridge, MA, USA, 1987, pp. 41–49.
[52] C. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Multimodal optimization by means of a topological species conservation algorithm," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 842–864, Dec. 2010.
[53] Q. Yang *et al.*, "Adaptive multimodal continuous ant colony optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 191–205, Apr. 2017.
[54] Q. Yang *et al.*, "Multimodal estimation of distribution algorithms," *IEEE Trans. Cybern*, vol. 47, no. 3, pp. 636–650, Mar. 2017.
[55] X. Li, "Niching without niching parameters: Particle swarm optimization using a ring topology," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 150–169, Feb. 2010.
[56] B. Y. Qu, P. N. Suganthan, and S. Das, "A distance-based locally informed particle swarm model for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 3, pp. 387–402, Jun. 2013.
[57] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.
[58] B. H. Chen and J. L. Hu, "An adaptive niching EDA based on clustering analysis," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, 2010, pp. 1–7.
[59] G. Wu *et al.*, "Ensemble of differential evolution variants," *Inf. Sci.*, vol. 423, pp. 172–186, Jan. 2018.
[60] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Ensemble strategies for population-based optimization algorithms—A survey," *Swarm Evol. Comput.*, vol. 44, pp. 695–711, Feb. 2019.
[61] G. Wu, R. Mallipeddi, P. N. Suganthan, R. Wang, H. Chen, "Differential evolution with multi-population based ensemble of mutation strategies," *Inf. Sci.*, vol. 329, pp. 329–345, Feb. 2016.
[62] Y. Lin, J. Zhang, and L.-K. Lan, "A contour method in population-based stochastic algorithms," in *Proc. IEEE Congr. Evol. Comput.*, 2008, pp. 2388–2395.

[63] X. Li, A. Engelbrecht, and M. G. Epitropakis, "Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization," Evol. Comput. Mach. Learn. Group, RMIT Univ., Melbourne, VIC, Australia, Rep., 2013. Accessed: Apr. 2019. [Online]. Available: http://titan.csit.rmit.edu.au/∼e46507/cec13-niching/competition/cec2013-niching-benchmark-tech-report.pdf

[64] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, 2011.

[65] J. E. Fieldsend, "Running up those hills: Multi-modal search with the niching migratory multi-swarm optimiser," in *Proc. IEEE Congr. Evol. Comput.*, Beijing, China, 2014, pp. 2593–2600.

[66] S. Biswas, S. Kundu, S. Das, and A. Vasilakos, "Information sharing in bee colony for detecting multiple niches in non-stationary environments," in *Proc. Genet. Evol. Comput. Conf.*, 2013, pp. 1–2.

[67] X.-F. Liu, Z.-H. Zhan, and J. Zhang, "Neural network for change direction prediction in dynamic optimization," *IEEE Access*, vol. 6, pp. 72649–72662, 2018.

[68] S. Kundu, S. Biswas, S. Das, and P. N. Suganthan, "Crowding-based local differential evolution with speciation-based memory archive for dynamic multimodal optimization," in *Proc. Genet. Evol. Comput. Conf.*, Amsterdam, The Netherlands, 2013, pp. 33–40.

[69] S. Biswas, S. Kundu, D. Bose, S. Das, and P. N. Suganthan, "Synchronizing differential evolution with a modified affinity-based mutation framework," in *Proc. IEEE Symp. Differ. Evol.*, Singapore, 2013, pp. 61–68.

[70] Y. H. Li, Z.-H. Zhan, S. J. Lin, J. Zhang, and X. N. Luo, "Competitive and cooperative particle swarm optimization with information sharing mechanism for global optimization problems," *Inf. Sci.*, vol. 293, no. 1, pp. 370–382, Feb. 2015.

[71] S. Biswas, M. A. Eita, S. Das, and A. Vasilakos, "Evaluating the performance of group counseling optimizer on CEC 2014 problems for computational expensive optimization," in *Proc. IEEE Congr. Evol. Comput.*, Beijing, China, 2014, pp. 1076–1083.

**Zi-Jia Wang** (S'15) received the B.S. degree in automation from Sun Yat-sen University, Guangzhou, China, in 2015, where he is currently pursuing the Ph.D. degree.

His current research interests include evolutionary computation algorithms like differential evolution, particle swarm optimization, and their applications in design and optimization such as cloud computing resources scheduling.

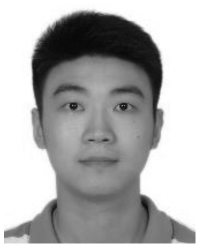
**Zhi-Hui Zhan** (M'13–SM'18) received the bachelor's and Ph.D. degrees from the Department of Computer Science, Sun Yat-sen University, Guangzhou, China, in 2007 and 2013, respectively.

He is currently the Changjiang Scholar Young Professor and the Pearl River Scholar Young Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou. He has published over 100 research papers in international journals and conference proceedings. His current research interests include evolutionary computation algorithms, swarm intelligence algorithms, and their applications in real-world problems, and in environments of cloud computing and big data.

Dr. Zhan was a recipient of the Outstanding Youth Science Foundation from National Natural Science Foundation of China in 2018, the Wu Wen Jun Artificial Intelligence Excellent Youth from the Chinese Association for Artificial Intelligence in 2017, and the China Computer Federation Outstanding Ph.D. Dissertation and the IEEE Computational Intelligence Society Outstanding Ph.D. Dissertation for his doctoral dissertation. He is listed as one of the Most Cited Chinese Researchers in Computer Science. He is currently an Associate Editor of *Neurocomputing*.

**Ying Lin** (M'13) received the B.Sc. degree in computer science and the Ph.D. degree in computer applied technology from Sun Yat-sen University, Guangzhou, China, in 2007 and 2012, respectively.

She is currently an Assistant Professor with the Department of Psychology, Sun Yat-sen University and also a Research Fellow with the South China University of Technology, Guangzhou. Her current research interest includes computational intelligence and its applications in psychology.



**Wei-Jie Yu** (S'10–M'14) received the bachelor's and Ph.D. degrees from Sun Yat-sen University, Guangzhou, China, in 2009 and 2014, respectively.

He is currently a Lecturer with the School of Information Management, Sun Yat-sen University and also a Research Fellow with the South China University of Technology, Guangzhou. His current research interests include computational intelligence and its applications on intelligent information processing, big data, and cloud computing.

**Hua Wang** received the Ph.D. degree from the University of Southern Queensland, Toowoomba, QLD, Australia.

He was a Professor with the University of Southern Queensland. He is currently a full-time Professor with Victoria University, Melbourne, VIC, Australia. He has over ten years teaching and working experience in applied informatics at both enterprise and university. He has expertise in electronic commerce, business process modeling, and enterprise architecture. As a Chief Investigator, three Australian Research Council Discovery grants have been awarded since 2006, and 200 peer reviewed scholar papers have been published. Six Ph.D. students have already graduated under his principal supervision.

**Sam Kwong** (M'93–SM'04–F'14) received the B.S. degree in electrical engineering from the State University of New York at Buffalo, Buffalo, NY, USA, in 1983, the M.S. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1985, and the Ph.D. degree from the University of Hagen, Hagen, Germany, in 1996.

From 1985 to 1987, he was a Diagnostic Engineer with Control Data Canada, Mississauga, ON, Canada, where he designed the diagnostic software to detect the manufacture faults of the very large-scale integration chips in the Cyber 430 machine. He later joined Bell-Northern Research, Ottawa, ON, Canada, as a Member of Scientific Staff. In 1990, he joined the City University of Hong Kong, Hong Kong, as a Lecturer with the Department of Electronic Engineering, where he is currently a Professor with the Department of Computer Science. His current research interests include pattern recognition, evolutionary computations, and video analytics.

Prof. Kwong has been the Vice President for IEEE Systems, Man, and Cybernetics for conferences and meetings since 2014. He was elevated to an IEEE Fellow for his contributions on optimization techniques for cybernetics and video coding in 2014. He is also appointed as an IEEE Distinguished Lecturer for IEEE SMC Society since 2017.

**Jun Zhang** (F'17) received the Ph.D. degree from the City University of Hong Kong, Hong Kong, in 2002.

He is currently a Visiting Scholar with Victoria University, Melbourne, VIC, Australia. His current research interests include computational intelligence, cloud computing, high performance computing, data mining, wireless sensor networks, operations research, and power electronic circuits. He has published over 100 technical papers in the above areas.

Dr. Zhang was a recipient of the China National Funds for Distinguished Young Scientists from the National Natural Science Foundation of China in 2011 and the First-Grade Award in Natural Science Research from the Ministry of Education, China, in 2009. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON CYBERNETICS, and the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS.