Multi-Objective Particle Swarm Optimization Based on Gaussian Sampling

GUOSEN LI, LI YAN, AND BOYANG QU
School of Electronic and Information Engineering, Zhengzhou University of Technology, Zhengzhou 450007, China
Corresponding author: Boyang Qu (legend@zzti.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61673404, Grant 61876169, and Grant 61976237, in part by the Key Scientific Research Projects in Colleges and Universities of Henan Province under Grant 19A120014, in part by the Research Award Fund for Outstanding Young Teachers in Henan Provincial Institutions of Higher Education under Grant 2016GGJS-094, and in part by the Science and Technology Innovation Team of Colleges and Universities in Henan Province under Grant 18IRTSTHN013.

ABSTRACT This paper proposes a multi-objective particle swarm optimization algorithm based on Gaussian sampling (GS-MOPSO) to locate multiple Pareto optimal solutions for solving multimodal multi-objective problems. In the proposed method, the Gaussian sampling mechanism is used to form multiple neighborhoods by learning from optimal information of particles. And particles search their own neighborhoods to obtain more optimal solutions in the decision space. Moreover, an external archive maintenance strategy is proposed which allows the algorithm to maintain an archive containing better distribution and diversity of solutions. Meanwhile, nine new multimodal multi-objective test problems are designed to evaluate the performance of algorithms. The performance of GS-MOPSO is compared with twelve state-of-the-art multi-objective optimization algorithms on forty test problems. The experimental results show that the proposed algorithm is able to handle the multimodal multi-objective problems in terms of finding more and well-distributed Pareto solutions. In addition, the effectiveness of the proposed algorithm is further demonstrated in a real-world problem.

INDEX TERMS Particle swarm optimization, evolutionary algorithm, multimodal, multi-objective optimization.

I. INTRODUCTION

Multi-objective optimization problems (MOPs) involve the optimization of multiple objective functions. In real-world applications, many optimization problems always consist of a several conflicting objectives and a number of constraints. Without loss of generality, a MOP can mathematically be described as follows:

\[
\begin{aligned}
\text{minimize } & \quad F(x) = (f_1(x), \ldots, f_m(x))^T \\
\text{subject to } & \quad g_i(x) \geq 0, \quad i = 1, \ldots, q \\
& \quad h_j(x) = 0, \quad j = 1, \ldots, p \\
& \quad x \in \mathbb{R}^n
\end{aligned}
\]

where \(x = (x_1, \ldots, x_n)\) is an \(n\)-dimensional decision vector in the decision space \(\mathbb{R}^n\), \(F(x) = (f_1(x), \ldots, f_m(x))^T\) is an \(m\)-dimensional objective vector in the objective space \(\mathbb{R}^m\), \(g_i(x) \geq 0\) is the \(i\)-th inequality constraint, \(h_j(x) = 0\) is the \(j\)-th equality constraint. Since the objectives in MOPs are often contradicting with each other, the Pareto dominance relationship is commonly utilized to compare different solutions. For any two feasible solutions \(x\) and \(x'\), it can be said that \(x\) dominates another solution \(x'\), if \(\forall i \in \{1, 2, \ldots, n\}, f_i(x) \leq f_i(x')\) and \(\exists j \in \{1, 2, \ldots, n\}, f_j(x) < f_j(x')\). If the solution \(x\) dominates any other solutions, \(x\) can be said as a non-dominated solution. The set of all non-dominated solution in the decision space is called as Pareto-optimal Set (PS) [1]. The set of the vectors in the objective space that correspond to the PS is referred to as Pareto Front (PF) [2],[3]. The crux of solving the MOPs is to identify evenly spread solutions in the objective space.

A number of multi-objective evolutionary algorithms (MOEAs) have been proposed to tackle MOPs over the past two decades [4]–[9]. Until recently, the MOPs which have multiple disjoint PSs corresponding to the same PF attract great widespread interest in the evolutionary computation research community [10]–[15]. This class of problems is referred to as multimodal multi-objective problems (MMOPs) by Liang et al. [10]. In Fig. 1, the two PSs in the...
decision space map to the same PF in the objective space, in which two solutions $A_1$ and $A_2$ in the decision space correspond to the same objective value $A'$ in the objective space. If the solutions $A_1$ and $A_2$ are obtained by the traditional multi-objective optimization algorithm concurrently, the solution $A_2$ will be deleted since the solutions $A_1$ and $A_2$ are too crowded in the objective space. However, if the solution $A_1$ becomes unavailable due to unexpected reasons, solution $A_2$ could provide an alternative option with the same quality for decision-maker. Therefore, both $A_1$ and $A_2$ should be retained simultaneously. Hence, identifying and reserving multiple PSs in decision space is an important task for addressing the multimodal multi-objective problems [16], [17].

There exist multimodal property for both single-objective and multi-objective optimization problems. For multimodal single-objective optimization, plenty of research works has been done to locate all the global and local optima instead of a single global optimum. Various niching methods have been developed at the same time, including crowding [18], fitness sharing [19], clearing [20], and speciation [21]. However, the involving research in multimodal multi-objective optimization has been little investigated [11]. Most researches focus exclusively on finding the Pareto front in the objective space but does not consider locate multiple PSs in the decision space. Even though a small number of algorithms have been proposed in the literature for multimodal multi-objective problems, further improvements are called for to alleviate observed shortcomings. First, the ability to find enough equivalent optimal solutions in the decision space is still relatively weak. Second, relatively simple test functions are adopted in the experiment. Finally, an algorithm’s capacity to tackle real-world problems is less studied. Based on the above discussion, the multimodal multi-objective optimization is worthy to be further studied. Meanwhile, an effective strategy to locate the multiple PSs in the decision space needs to be further developed.

In this paper, we propose a multi-objective particle swarm optimization algorithm based on Gaussian sampling (GS-MOPSO). In GS-MOPSO, a Gaussian sampling mechanism is adopted to establish different neighborhoods. And the particles evolve within their own neighborhoods that can locate more solutions in the decision space. Moreover, an external archive maintenance strategy is employed to maintain the diversity of solutions. Then, nine multimodal multi-objective test problems are designed to assess the performance of the algorithms. In addition, a real-world problem is solved by the proposed algorithm to further prove its effectiveness.

The rest of this paper is organized as follows. Section II reviews the related works. Section III introduces the details of the proposed GS-MOPSO. Section IV describes the experimental settings. And Section V reports the experimental results and the relevant analysis. Section VI summarizes the conclusions and describes the future work.

II. RELATED WORK
A. PARTICLE SWARM OPTIMIZATION
Particle swarm optimizer (PSO), proposed by Kennedy and Eberhart in 1995 [22], mimics a flock of birds’ foraging behavior to solve the optimization problems. Each particle can be viewed as a potential solution, which flies through the solution space [23]–[25]. The position and velocity of the particle are dynamically adjusted according to its historical personal best position ($p_{best}$) and the historical best position of its neighborhood ($n_{best}$). The velocity and position of the particle are updated according to (2) and (3),

$$v_{i+1}(t) = wv_i(t) + C_1r_1(x_{p_{best}} - x_i(t)) + C_2r_2(x_{n_{best}} - x_i(t))$$
$$x_{i+1}(t) = x_i(t) + v_{i+1}(t)$$

where $v_i(t)$, $x_i(t)$ denote the velocity and position of particle $i$ of the $t$th generation, respectively. $r_1$ and $r_2$ are random numbers within the range $[0, 1]$; $w$ represents the inertia weight; $C_1$ and $C_2$ are the acceleration coefficients.

Owing to its simplicity, effectiveness, and reliability, PSO has been successfully used in many theoretic research and engineering applications. For practical purposes, various strategies are introduced into PSO to enhance the performance of the algorithm [26]–[32]. For instance, in order to avoid falling into local optimal solutions and improve the global searching ability, a stochastic inertia weight strategy is proposed [32]. In the proposed strategy, the inertia weight is adjusted by using the characteristics of random variables. The corresponding inertia weight update equation is as follows

$$w = \mu_{min} + (\mu_{max} - \mu_{min}) \cdot \text{rand}(0, 1) + \sigma \cdot N(0, 1)$$

where $\sigma$ is constant. $\mu_{min}$ and $\mu_{max}$ stand for the minimum and maximum value of the inertia weight, respectively. $N(0, 1)$ means random number of standard normal distribution.

B. PRIOR WORKS ON MULTIMODAL MULTI-OBJECTIVE OPTIMIZATION
Maintaining the distribution of solutions in the decision space has been done by a few researchers. Deb and Tiwari [33] introduced the concept of crowding distance into the decision space and proposed the Omni-optimizer to preserve the diversity of the solution in the decision space. Ulrich et al. [34] introduced decision space diversity into the hypervolume metric to increase the diversity in both decision space and objective space. Chan and Ray [35] adopted Lebesgue contribution and neighborhood count to enhance the diversity of population in the decision and objective space.
Rudolph et al. [36] employed a restart strategy to obtain a well-distributed Pareto set. Zhou et al. [37] proposed a probabilistic model based on MOEA to approximate the PS and the PF. Xia et al. [38] integrated crowding estimation method into MOEA, in which crowding distances were applied to guarantee diversity in the decision space. These works consider the distribution in the decision space, which produces positive effects on ameliorating the performance of the algorithm for handling multi-objective problems. However, the multi-objective problems where multiple PSs corresponding to the same PF is not well studied.

Multimodal multi-objective optimization aims at locating multiple equivalent PSs. Therefore, niching techniques are incorporated into multi-objective algorithms by some researchers to promote the diversity of the population and improve the distribution of the solutions. Liang et al. [10] proposed a decision space based niching NSGAII (called DNSGAII), which employs the niching method to maintain the diversity in the decision space. Tanabe and Ishibuchi [39] proposed a decomposition-based evolutionary algorithm with addition and deletion operators to find more equivalent PSs. The results demonstrate that the proposed algorithm can maintain the population diversity in the decision space. Yue et al. [11] introduced a ring topology and special crowding distance into particle swarm optimization algorithm, in which a ring topology was deployed to form stable niches for maintaining multiple PSs, and a special crowding distance was utilized to balance the diversity in the both objective and decision space. Liang et al. [16] integrated self-organizing map network into particle swarm algorithm, in which the self-organizing map network is employed to establish the neighborhood of the individuals to identify a larger number of Pareto solutions. Liu et al. [15] introduced a multimodal multi-objective evolutionary algorithm using two-archive and recombination strategies to promote a good diversity solution in the decision space. Qu et al. [17] proposed a self-organized speciation based multi-objective particle swarm optimizer to locate multiple optimal solutions. The results show that the proposed method is competitive. Li et al. [40] combined reinforcement learning mechanism into differential evolution algorithm, in which the reinforcement learning mechanism is applied to increase the diversity of the solution set in the decision space.

C. MOTIVATION

There is more than one PS for multimodal multi-objective problems. For example, for MMF2 [11], the function has two PSs, namely $PS_1$ and $PS_2$, as shown in Fig. 2. In Fig. 2, A and B are particles, and C and D are candidate $nbest$ solutions.

To locate more Pareto solutions, observed two problems are considered to be solved. The first one is how to guide the particles close to one of the PS. Particle A is far away from $PS_1$ and $PS_2$, and has relatively large rank according to non-dominated sorting approach compared with other solutions (i.e., B, C and D). Particle A and solution C are more likely to be in the same niche as they are close to each other compared with solution D. If solution C is selected as $nbest$ to guide the update of particle A, it is conducive to converge and locate the solution effectively. In this way, the population can also form multiple niches that can converge to different solutions. The second one is how to search more potential solutions after locating one of the PS. In Fig. 2, particle B is close to $PS_2$, which indicated that it in a potential area. And it has a lower rank in the population in accordance with non-dominated sorting approach. If a local search is performed on particle B, it is helpful to obtain multiple desired solutions. Based on the above discussion, for the larger rank particle, candidate solution which is close to it is used to guide its movement and thereby close to the true PS. In contrast, for the lower rank particle, the particle evolves within its own niche by local search using the proposed Gaussian sampling mechanism. And the detail of the proposed multi-objective particle swarm optimization algorithm based on Gaussian sampling (GS-MOPSO) is described in Section III.

III. PROPOSED ALGORITHM

In this section, we present the main framework of the multi-objective particle swarm optimization algorithm based on Gaussian sampling (GS-MOPSO), which is composed of two components: Gaussian sampling mechanism, and external archive maintenance strategy. The details are described in the following subsections.

A. MAIN FRAMEWORK OF THE GS-MOPSO

Algorithm 1 outlines the main framework of the proposed GS-MOPSO, where the notation $POP_i$ represents the $i$th particle at the $i$th generation, and $PBA_i$ denotes the $i$th particle’s personal historical best positions. $PBA$ retains each particle’s historical best positions, which is advantageous for each particle to improve its position for the next generation by learning its own historical experience.

The procedure of GS-MOPSO is as follows. First, the particle population ($POP$), the personal best archive ($PBA$) and the external archive ($EXA$) are initialized. Then, each particle in $POP$ is assigned a rank value by employing the fast non-dominated sort approach. The purpose of this step is to obtain the maximum rank value of the whole population ($MaxRank$) and the rank value of each particle ($Rank$) for Gaussian sampling mechanism. Next, the first solution from
Algorithm 1 Framework of GS-MOPSO

1. Input: maximum number of iterations $\text{MaxIter}$, population size $N$.
2. Output: the non-dominated solutions in external archive $\text{EXA}$.
3. Initialize a random population $\text{POP}_0$, and Evaluation $\text{POP}_0$.
4. //Initialize personal best archive $\text{PBA}$ and external archive $\text{EXA}$.
5. for $i = 1: N$
6. $\text{PBA}[i] = \text{POP}_0(i)$.
7. end for
8. $\text{EXA} = \text{POP}_0$.
9. for $t = 1: \text{MaxIter}$
10. Assign rank value for each particle in $\text{POP}_t$ according to fast non-dominated sort approach.
11. for $i = 1: N$
12. //Select $pbest$ and $nbest$.
13. $pbest = \text{The first one in sorted PBA}[i]$.
14. Select a solution that is closest to the $i$th particle from $\text{EXA}$ as $nbest$.
15. Update $\text{POP}_t(i)$ to $\text{POP}_{t+1}(i)$ according to (4), (2), and (3).
16. Perform Gaussian sampling mechanism.
17. Evaluation($\text{POP}_{t+1}(i)$).
18. //Update $\text{PBA}$.
19. Put $\text{POP}_{t+1}(i)$ into $\text{PBA}[i]$ and remove solutions dominated by $\text{POP}_{t+1}(i)$.
20. end for
21. end for
22. Update $\text{EXA}$ according to external archive maintenance strategy.

Algorithm 2 Gaussian Sampling Mechanism

1. Requirement: rank value of the $i$th particle $\text{Rank}_i$, maximum rank value $\text{maxRank}$, maximum number of iterations $\text{MaxIter}$, current iteration number $t$, personal best position $pbest$, neighborhood best position $nbest$, population $\text{POP}$, personal best archive $\text{PBA}$, external archive $\text{EXA}$.
2. if $\text{rand} > \text{Rank}_i/\text{maxRank}$
3. if $\text{rand} > t / \text{MaxIter}$
4. //Global Gaussian sampling.
5. $\text{sampleSet} = \{pbest; nbest\}$
6. $\text{POP}_{t+1}(i) = N(\text{mean}(\text{sampleSet}), \text{std}(\text{sampleSet})+\varepsilon)$
7. else
8. //Local Gaussian sampling.
9. Divide the set $\{\text{POP}; \text{PBA}; \text{EXA}\}$ into $N$ clusters by using $K$-means clustering method.
10. Identify the cluster which the particle $\text{POP}_t(i)$ belongs, and assign it to the $\text{sampleSet}$.
11. $\text{POP}_{t+1}(i) = N(\text{mean}(\text{sampleSet}), \text{std}(\text{sampleSet})+\varepsilon)$.
12. end if
13. end if

the sorted $\text{PBA}[i]$ is chosen as the $pbest$. The solution closest to the $i$th particle in $\text{EXA}$ is chosen as the $nbest$. Afterward, $\text{POP}_t(i)$ is updated to $\text{POP}_{t+1}(i)$ according to (4), (2), and (3). Then, the Gaussian sampling mechanism is employed. After evaluation, $\text{POP}_{t+1}(i)$ is stored into $\text{PBA}[i]$ and all solutions dominated by $\text{POP}_{t+1}(i)$ are removed. Finally, $\text{EXA}$ is updated according to external archive maintenance strategy. Repeat the above steps until termination conditions are satisfied.

**B. GAUSSIAN SAMPLING MECHANISM**

Particle swarm optimization is easy to plunge into the local optimal, and appear premature stagnation phenomenon. And it has a slow convergence to the optimal solution and cannot locate more than one solution without the niching technique. According to Section II.C, more potential solutions can be obtained by searching in the neighborhood of lower rank particle, so as to improve the disadvantages of the algorithm. Therefore Gaussian sampling mechanism is proposed.

The Gaussian sampling mechanism includes global Gaussian sampling and local Gaussian sampling, which aims to achieve a trade-off between global exploration and local exploitation. In the early search stage, global Gaussian sampling is employed to explore the search space comprehensively and to seek global optimal solutions quickly for promoting the exploration capability of the algorithm. The main idea is to guide the particle to move toward the global optimum by learning the optimal information of particles. In the latter stage of the search process, local Gaussian sampling is adopted to exploit the neighborhood of the promising solution and to locate more optimal solutions in the decision space for enhancing the exploitation capability of the algorithm. The main idea is to induce multiple neighborhoods in decision space and to guide particles to evolve in their own neighborhood.

The procedure of Gaussian sampling mechanism is shown in Algorithm 2, while $\text{rand}$ is a random number uniformly distributed between 0 and 1, $\text{Rank}_i$ is the rank of the $i$th particle, $t$ is current iteration, $\text{MaxIter}$ is maximum number of iterations, $N()$ is a Gaussian random function, $\text{mean}()$ is the arithmetic mean, $\text{std}()$ refers to the standard deviation, and $\varepsilon$ is a small number to avoid $\text{std}()$ equals zero. The process starts from judging $\text{rand} < \text{Rank}_i/\text{maxRank}$ to decide whether to adopt Gaussian sampling mechanism. If $\text{rand} < \text{Rank}_i/\text{maxRank}$, the Gaussian sampling mechanism is adopted. Then if $\text{rand} > t / \text{MaxIter}$, the global Gaussian sampling is performed, otherwise the local Gaussian sampling is employed.

For global Gaussian sampling, the best positions found by population and individual are used to evolve particle.
By combining the best position found so far, the particle can be guided toward the global optimal direction effectively. The steps are as follows. First, a sampling set (sampleSet) is defined, and contains pbest, nbest. Then, according to the sample set, a new position is generated by Gaussian distribution. And the update formula is expressed as follows:

\[
POP_{t+1}(i) = N(\text{mean}(\text{sampleSet}), \text{std}(\text{sampleSet}) + \epsilon)
\]

As a result, the new position will be around the pbest and nbest, which increase the chance of finding potential solutions.

For local Gaussian sampling, particle’s neighborhood information is adopted to generate its next position. The procedures are as follows. First, according to K-means clustering method [41]–[43], the set \([POP; PBA; EXA]\) is divided into \(N\) clusters (neighborhoods). Then, the neighborhood to which the particle belongs is identified, and assigns it to the variable sampleSet. Next, the position of particle is updated using Gaussian distribution. And the update formula is given below:

\[
POP_{t+1}(i) = N(\text{mean}(\text{sampleSet}), \text{std}(\text{sampleSet}) + \epsilon)
\]

In this way, particles fine search in their own neighborhood. Therefore, it has the ability to quickly find approximate solutions and the search accuracy can be improved at the same time.

To summarize, the new position generated by particles is randomly distributed around the optimal solution in a Gaussian distribution to enhance the population diversity and avoid falling into a local optimal solution. Moreover, by searching the neighborhood of the optimal solution, more solutions in the decision space are located. More specifically, in the early stage of iteration, pbest is far from nbest, and the standard deviation will be large, which facilitates to explore in a wider range for particles. Then by continuously learning the pbest and nbest, the particles gradually approach and converge to the optimal solution. In the later stage of iteration, particles exploit their neighborhood according to the neighborhood information. Each particle evolves in its own neighborhood. Therefore particles can find the optimal solution in its neighborhood, so the whole population can locate more solutions in different neighborhoods. Besides, by using the information of PBA and EXA, the particles move toward the optimal solution promptly and can find more potential solutions. Thus the search efficiency and accuracy of the algorithm are improved.

The process for the Gaussian sampling mechanism can be summarized as follows:

**Step 1:** Check whether the condition \(\text{rand} < \text{Rank} / \text{maxRank}\) is satisfied, if so, go to Step 2; otherwise, exit the program.

**Step 2:** Global Gaussian sampling. If \(\text{rand} > \text{t/MaxIter}\), Steps 2.1-2.3 are invoked; else, go directly to Step 3.

**Step 2.1:** A variable sampleSet is defined and set to an empty value.

**Step 2.2:** Assign \([pbest; nbest]\) to sampleSet.

**Step 2.3:** Calculate the mean and standard deviation of sampleSet. Use formula (A5) to update the position of the particle \(POP_{t+1}(i)\). Then go to Step 4.

**Step 3:** Local Gaussian sampling.

**Step 3.1:** Set variable sampleSet to an empty value.

**Step 3.2:** Divide \([POP; PBA; EXA]\) into \(N\) clusters by using K-means method.

**Step 3.3:** Identify the cluster to which the \(i\)th particle belongs, and allocate it to sampleSet.

**Step 3.4:** Compute the mean and standard deviation of sampleSet. Update particle’s position \(POP_{t+1}(i)\) according to equation (A6). Then go to Step 4.

**Step 4:** Return the position \(POP_{t+1}(i)\).

### C. EXTERNAL ARCHIVE MAINTENANCE STRATEGY

The external archive is utilized to store the non-dominant solutions found so far and to guide the particles toward the true Pareto set [44], [45]. In general, a maximum size is defined for the external archive. When the size of solutions in the external archive reaches its predefined maximum value, the external archive is maintained to determine which solution can be retained into the EXA. If the diversity of the non-dominant solutions in EXA is poor, the particles in the population will gather in a certain region.

Suppose that the maximum size of the external archive is six, and six solutions need to be remained in the external archive. The distribution of non-dominant solutions in one iteration is shown in Fig.3 as a simple example. In Fig. 3(a), there are eleven solutions, and the solutions A, B, and C belong to the same cluster and are quite close. Compared to adding all three solutions to the external archive (see Fig. 3(b)), maintaining only one of them into the external archive (see Fig. 3(c)) can greatly increase diversity with the same archive size. Based on the above idea, an external archive maintenance strategy is proposed to obtain the even distribution of non-dominant solutions in the decision space. External archive maintenance strategy at the \(i\)th generation is described as following.

First, copy all individuals from archive EXA and population POP, to a combined population \(R_i\). The population \(R_i\) is of size \(2N\). Then, the population is sorted according to the non-dominated sort method. In this process, individuals are assigned to several fronts \(F_1, F_2, \ldots, F_l\) according to the level of non-domination, where \(l\) is the value of the last
non-dominated front. If the size of $F_1$ is less than $N$, the first $N$ better order individuals are added to the $EXA$ directly. Otherwise archive maintenance strategy will be performed. The standard $K$-mean clustering algorithm divides $F_1$ into $N$ clusters. Then randomly select a solution from each cluster and add it to the $EXA$. In this way, the external archive $EXA$ is updated.

In theory, the $K$-mean clustering method divides similar solutions into the same cluster, where these solutions are dissimilar from solutions in other clusters. Therefore, only one of the solutions in the same cluster is stored in the external archive, which not only avoids over-exploitation of a certain region by the particle in the evolution process, but also provides well-distributed solutions for the external archive.

### IV. EXPERIMENTAL DESIGN

#### A. TEST FUNCTIONS

The research on multimodal multi-objective optimization is still in the emerging phase, related benchmark functions from the literature are relatively few. Furthermore, many researchers are becoming interested in this area [11], it is essential to design complicated test problems for comparing the effectiveness of the multimodal multi-objective algorithms systematically. Thus, according to the approach of designing test problem [10], [11], nine multimodal multi-objective test problems (i.e., F1–F9) are proposed in this paper. Table 1 lists the relevant features of these nine test problems in this work, where $n$ is the number of variables, and $m$ is the number of objectives.

As is shown in Table 1, the function $F_1$ has four linear PSs, and its PF is composed of discontinuous pieces. $F_2$ has two PSs, and its PSs and PF are linear. $F_3$ and $F_4$ are originated from unimodal multi-objective problem UF2 [46], which have four PSs. $F_5$ developed from UF2 [46] has two PSs, where one of the PS is an irregular geometry which is shown in Appendix A. The building block of $F_6$ is from UF7 [46]. The PSs of $F_6$ are nonlinear, and its PF is linear. $F_7$ and $F_8$ are constructed based on UF4 [46]. $F_7$ has convex PF, and $F_8$ has concave PF. $F_9$ has eight PSs, which is more complex than $F_7$ with two PSs. The PSs shapes of $F_9$ include the nonlinear and linear geometry simultaneously, which increases the difficulty degree of locating multiple solutions. Further details of the nine test problems are in the attached Appendix. In addition, thirty-one other test functions, called MMF1-8 [11], MMF10-MMF15 [47], MMF1_e [47], MMF14_a [47], MMF15_a [47], MMF10-1-MMF15_1 [47], MMF15_a_1 [47], MMF16_11-MMF16_13 [47], SYM-PART simple [36], SYM-PART rotated [36], SYM-PART rot.+ trans. [36], map-based test problem (denoted as MBP) [48], and the Omni-test function [33], are included as test problems in the experiments.

#### B. PERFORMANCE INDICATORS

Pareto Sets Proximity ($PSP$) [11] and Inverted Generational Distance ($IGD$) [49], [50] are used to evaluate the performance of algorithm. $PSP$ reflects the similarity between the true PSs and the obtained solutions, and $IGD$ indicator can measure both convergence and diversity of the obtained solutions in the objective space. $PSP$ is used to compare the performance of the algorithm in the decision space, while $IGD$ is utilized to assess the performance of the algorithm in the objective space. The larger value of $PSP$ means the obtained solutions in decision space are well distributed. And an algorithm with a larger $IGD$ is considered to be a better design for multimodal multi-objective optimization. The small value of $IGD$ means the convergence and diversity of obtained solutions in objective space are good.

#### C. COMPETING ALGORITHMS

To demonstrate the effectiveness of algorithm, the proposed GS-MOPSO is compared with the following twelve multimodal multi-objective algorithms: ZS-MO_Ring_PSO_SCD (denoted as ZS-MRPS for simplicity) [51], DE_RLRF [40], SSMPSO [17], MMOPIO [52], MODE [53], TriMOEATA&R [15], PEN-MOBA [54], NMOHSA [55], SPSO-MM [16], MO_Ring_PSO_SCD (denoted as MRPS) [11], DN-NSGAII [10], and Omni-optimizer [33].

Note that the source code of the MRPS algorithm can be found at http://www5.zzu.edu.cn/cilab/info/1025/1131.htm. The source code of the SPSO-MM algorithm is available at http://www5.zzu.edu.cn/cilab/info/1025/1231.htm. The source code of the MMOPIO algorithm is located at http://www5.zzu.edu.cn/cilab/info/1025/1230.htm. The source code of the MOMODE algorithm can be downloaded at http://www5.zzu.edu.cn/cilab/info/1025/1231.htm. The source codes of the other algorithms are obtained from the respective authors.

#### D. PARAMETER SETTINGS

For fair comparison, the population size is set to 800 and the maximal evaluation is set to 80,000 according to the original study [11]. For each problem, 25 independent runs are carried out. In GS-MOPSO, both $C_1$ and $C_2$ are set to 2.05 [11], and $\mu_{min}$, $\mu_{max}$ and $\sigma$ are respectively set to 0.4, 0.9, 0.15[56], and the parameter $\epsilon$ is set to $10^{-2}$. Discussion about the
TABLE 2. Mean and standard deviation values of \( \text{PSP} \) obtained by different variants.

| Parameter | GS-MOPSO | MOPSO-I | MOPSO-II | MOPSO |
|-----------|-----------|---------|---------|-------|
| MMF1      | 124.15(1.74) | 54.95(5.12) | 3.14(0.69) | 3.11(0.25) |
| MMF2      | 276.19(29.37) | 109.45(12.95) | 13.44(2.68) | 5.56(0.48) |
| MMF3      | 289.70(12.69) | 141.23(32.66) | 13.12(1.46) | 7.03(1.37) |
| MMF4      | 212.08(11.73) | 108.53(25.29) | 3.74(0.45) | 3.70(0.45) |
| MMF5      | 54.36(2.94) | 27.98(1.13) | 2.20(0.18) | 1.82(0.17) |
| MMF6      | 60.62(2.01) | 28.02(2.13) | 2.84(0.52) | 2.68(0.15) |
| MMF7      | 203.69(6.87) | 67.14(15.30) | 5.02(0.92) | 2.92(0.33) |
| MMF8      | 105.41(6.10) | 48.30(4.18) | 1.70(0.45) | 1.87(0.23) |
| MMF10     | 372.09(15.25) | 50.64(8.80) | 39.31(4.14) | 19.24(3.60) |
| MMF11     | 996.38(45.90) | 639.78(20.82) | 36.08(7.88) | 27.07(4.38) |
| MMF12     | 1818.98(77.07) | 1349.24(60.81) | 105.77(21.13) | 96.97(11.69) |
| MMF13     | 82.70(4.40) | 54.83(3.28) | 9.99(1.61) | 8.27(0.11) |
| MMF14     | 50.48(0.67) | 27.49(1.11) | 4.64(0.23) | 3.91(0.13) |
| MMF15     | 60.07(0.91) | 39.06(2.18) | 5.93(1.72) | 5.73(0.30) |
| MMF16_a   | 8.99(0.66) | 2.93(1.56) | 1.17(0.31) | 0.39(0.33) |
| MMF16_b   | 42.01(0.20) | 22.81(0.35) | 3.71(0.32) | 3.09(0.21) |
| MMF15_a   | 53.67(5.23) | 29.49(2.17) | 5.08(1.18) | 4.88(0.19) |
| MMF10_c   | 6.16(0.45) | 5.77(0.11) | 5.54(1.28) | 3.92(0.02) |
| MMF11_c   | 2.69(1.73) | 1.88(0.17) | 1.77(1.02) | 0.43(1.48) |
| MMF12_c   | 2.56(3.21) | 2.13(0.79) | 1.57(1.33) | 0.51(1.56) |
| MMF13_c   | 2.84(0.56) | 2.43(1.40) | 1.81(0.14) | 1.17(0.40) |

FIGURE 4. The comparison of PSs obtained by multiobjective PSO algorithms on MMF3.

parameter \( \varepsilon \) is covered in Section 5.4. The other parameter settings of the compared algorithms are set as their respective references [10], [11], [15]–[17], [33], [40], [51]–[55].

E. RUNNING ENVIRONMENT

All experiments are conducted independently on the same computer. The hardware conditions of the computer are as follows: CPU is Intel Xeon E5-2640 with 2.0GHz and 8GB main memory; the software platform is Windows 8.1 operating system; the simulation software is MATLAB R2016a. All the algorithms are coded and run using MATLAB.

V. RESULTS AND DISCUSSIONS

A. EXPERIMENTAL VERIFICATION OF THE EFFECTIVENESS OF THE PROPOSED ALGORITHM

To illustrate the effectiveness of Gaussian sampling mechanism and of external archive maintenance strategy, the proposed GS-MOPSO is compared with the basic MOPSO [11], MOPSO-I (MOPSO only with the Gaussian sampling mechanism), and MOPSO-II (MOPSO only with the external archive maintenance strategy). The mean \( \text{PSP} \) values on forty test functions are presented in Table 2. Moreover, the Wilcoxon’s rank sum test is applied to determine the statistical significance of the advantage of GS-MOPSO. “+”, “−”, and “=” in Table 2 denote the performance of GS-MOPSO is better than, worse than, and equal to that of the other algorithm. Furthermore, the PSs obtained by these four algorithms on MMF3 are presented in Fig. 4.

It is observed that the \( \text{PSP} \) value obtained by MOPSO-I and MOPSO-II are marginally larger than that of MOPSO, while the \( \text{PSP} \) value achieved by GS-MOPSO is much greater than that of MOPSO. In addition, the rank sum test results show that there are significant differences among GS-MOPSO and the other three algorithms. Moreover, GS-MOPSO locates more Pareto solutions in the decision space than other three algorithms as shown in Fig. 4. The reasons are discussed in the following paragraphs.

Gaussian sampling mechanism enables the algorithm to locate enough optimal solutions. This mechanism can guide particles to search for the potential region by using optimal information. Meanwhile, it establishes multiple niches, and particles evolve independently in their own niche
| TABLE 3. The mean and standard deviation values of $PSP$ obtained by different algorithms. |
|-----------------------------------------------|
| **Algorithm** | **Mean** | **SD** |
|-----------------|----------|--------|
| GS-MOPSO        | 20.9724  | 3.83   |
| ZS-MRPS         | 20.9724  | 3.83   |
| DE-RLRF         | 20.9724  | 3.83   |
| SM-GMOPO        | 20.9724  | 3.83   |
| MPMOPPO         | 20.9724  | 3.83   |
| TMOMOAP & R     | 20.9724  | 3.83   |
| PEN-MOBA        | 20.9724  | 3.83   |
| NMOHSA          | 20.9724  | 3.83   |
| SM-POMPO-MM     | 20.9724  | 3.83   |
| MRPS            | 20.9724  | 3.83   |
| DNS-GMOPO       | 20.9724  | 3.83   |
| Omni-optimizer  | 20.9724  | 3.83   |

**MMF1**

- **G. Li et al.: Multi-Objective Particle Swarm Optimization Based on Gaussian Sampling**
  
  **TABLE 3. The mean and standard deviation values of $PSP$ obtained by different algorithms.**

| **Algorithm** | **Mean** | **SD** |
|-----------------|----------|--------|
| GS-MOPSO        | 20.9724  | 3.83   |
| ZS-MRPS         | 20.9724  | 3.83   |
| DE-RLRF         | 20.9724  | 3.83   |
| SM-GMOPO        | 20.9724  | 3.83   |
| MPMOPPO         | 20.9724  | 3.83   |
| TMOMOAP & R     | 20.9724  | 3.83   |
| PEN-MOBA        | 20.9724  | 3.83   |
| NMOHSA          | 20.9724  | 3.83   |
| SM-POMPO-MM     | 20.9724  | 3.83   |
| MRPS            | 20.9724  | 3.83   |
| DNS-GMOPO       | 20.9724  | 3.83   |
| Omni-optimizer  | 20.9724  | 3.83   |
to fine search for more solutions. The introducing of external archive maintenance strategy helps obtain a good distribution of optimal solutions in the decision space. The similar solutions are removed preferentially. In this way, if solutions are not similar (crowded) in decision space, they are able to survive and maintain in the archive. Therefore, the distribution of obtained solution is improved.

In conclusion, both Gaussian sampling mechanism and external archive maintenance strategy make GG-MOPSO more effective in solving multimodal multi-objective problems.

B. COMPARISON WITH OTHER ALGORITHMS

The mean values and standard deviations of the PSP metric for all the algorithms are presented in Table 3. The results show that the PSP values achieved by GS-MOPSO are highest on thirty-two test functions. In addition, according to the rank sum test, GS-MOPSO has better or similar performance as compared with the other compared algorithms on thirty-six test functions.

With respect to MMF3, the PSP value obtained by MMOPIO is a bit larger than GS-MOPSO. For MMF13_l, PEN-MOBA has a much larger PSP value than GS-MOPSO. For MMF15_a_l and F3, NMOHSA obtains the better PSP values than that of GS-MOPSO. However, the rank sum test results indicate that GS-MOPSO shows an equivalent performance with MMOPIO, PEN-MOBA, and NMOHSA.

For MMF10, the SMPSO-MM is superior to GS-MOPSO. The reason may be that the SMPSO-MM employed the self-organizing map to extract neighboring relation information and help find the topological properties of the PSs, which makes the SMPSO-MM more suitable for solving MMF10. For MMF11_l and MMF12_l, GS-MOPSO is surpassed by ZS-MRPS. The key reason is that ZS-MRPS adopted the zoning search strategy to explore different subspaces and preserve the diversity of solutions for each subspace, which is more efficient for solving MMF11_l and MMF12_l. For F1, DE_RLRF obtains a higher PSP value. The reason is that DE_RLRF used the reinforcement learning to dynamically adjust the evolution direction of the population, which is more conducive to quickly find multiple PSs of F1. For the remaining thirty-two test functions, GS-MOPSO performs better than other algorithms.

C. COMPARISON OF THE CONVERGENCE BEHAVIORS OF DIFFERENT ALGORITHMS

The convergence behavior of the algorithm is investigated on one typical test problem F6, which has two PSs in a three-dimensional decision space. The search space of F6 is divided into two sub-regions, namely Region 1\{x₁ ∈ [−1, 0], x₂ ∈ [−1, 1], x₃ ∈ [−1, 1]\} and Region 2\{x₁ ∈ [0, 1],
| GS-MPPO | ZS-MRPS | DE_RLRF | SMMPSO | MMPOPO | MMODE | trimOEATDA | PEN-MOBA | MIMOSA | SMPSO-MM | MRPS | DNN-GSAI | Omni-optimizer |
|---------|---------|---------|--------|--------|--------|------------|----------|---------|----------|------|----------|---------------|
| MF1     | 4.87E-04 | 6.47E-04 | 8.97E-04 | 8.68E-04 | 6.93E-04 | 8.20E-04 | 3.27E-03 | 6.93E-04 | 6.80E-04 | 6.79E-04 | 1.04E-03 | 7.02E-04 | 8.89E-04 |
| MF2     | 1.21E-04 | 1.51E-04 | 1.75E-04 | 1.78E-04 | 1.43E-04 | 1.80E-04 | 1.68E-04 | 1.51E-04 | 1.52E-04 | 1.52E-04 | 1.54E-04 | 1.52E-04 | 1.51E-04 |
| MF3     | 3.88E-04 | 4.26E-04 | 6.64E-04 | 2.79E-03 | 1.21E-03 | 2.56E-03 | 3.43E-03 | 2.51E-03 | 3.56E-03 | 5.19E-03 | 6.16E-03 | 7.54E-04 |
| MF4     | 5.26E-04 | 5.19E-04 | 9.94E-04 | 8.36E-04 | 6.19E-04 | 7.95E-04 | 6.60E-02 | 6.07E-04 | 6.79E-04 | 5.99E-04 | 9.35E-04 | 6.52E-04 |
| MF5     | 1.57E-04 | 3.65E-04 | 1.25E-03 | 1.25E-03 | 1.25E-03 | 1.25E-03 | 1.25E-03 | 1.25E-03 | 1.25E-03 | 1.25E-03 | 1.25E-03 | 1.25E-03 |
| MF6     | 3.26E-04 | 3.26E-04 | 3.26E-04 | 3.26E-04 | 3.26E-04 | 3.26E-04 | 3.26E-04 | 3.26E-04 | 3.26E-04 | 3.26E-04 | 3.26E-04 | 3.26E-04 |
| MF7     | 4.41E-04 | 4.81E-04 | 8.91E-04 | 7.22E-04 | 4.66E-04 | 6.20E-04 | 7.59E-04 | 5.37E-04 | 6.79E-04 | 6.96E-04 | 1.08E-03 | 8.88E-04 |
| MF8     | 8.63E-04 | 8.63E-04 | 8.63E-04 | 8.63E-04 | 8.63E-04 | 8.63E-04 | 8.63E-04 | 8.63E-04 | 8.63E-04 | 8.63E-04 | 8.63E-04 | 8.63E-04 |
| MF9     | 1.78E-03 | 1.78E-03 | 1.78E-03 | 1.78E-03 | 1.78E-03 | 1.78E-03 | 1.78E-03 | 1.78E-03 | 1.78E-03 | 1.78E-03 | 1.78E-03 | 1.78E-03 |
| MF10    | 2.77E-03 | 2.77E-03 | 2.77E-03 | 2.77E-03 | 2.77E-03 | 2.77E-03 | 2.77E-03 | 2.77E-03 | 2.77E-03 | 2.77E-03 | 2.77E-03 | 2.77E-03 |
| MF11    | 1.18E-03 | 1.18E-03 | 1.18E-03 | 1.18E-03 | 1.18E-03 | 1.18E-03 | 1.18E-03 | 1.18E-03 | 1.18E-03 | 1.18E-03 | 1.18E-03 | 1.18E-03 |
| MF12    | 2.44E-03 | 2.44E-03 | 2.44E-03 | 2.44E-03 | 2.44E-03 | 2.44E-03 | 2.44E-03 | 2.44E-03 | 2.44E-03 | 2.44E-03 | 2.44E-03 | 2.44E-03 |
| MF13    | 4.22E-03 | 4.22E-03 | 4.22E-03 | 4.22E-03 | 4.22E-03 | 4.22E-03 | 4.22E-03 | 4.22E-03 | 4.22E-03 | 4.22E-03 | 4.22E-03 | 4.22E-03 |
| MF14    | 5.05E-03 | 5.05E-03 | 5.05E-03 | 5.05E-03 | 5.05E-03 | 5.05E-03 | 5.05E-03 | 5.05E-03 | 5.05E-03 | 5.05E-03 | 5.05E-03 | 5.05E-03 |
| MF15    | 9.09E-04 | 9.09E-04 | 9.09E-04 | 9.09E-04 | 9.09E-04 | 9.09E-04 | 9.09E-04 | 9.09E-04 | 9.09E-04 | 9.09E-04 | 9.09E-04 | 9.09E-04 |
| MF16    | 8.96E-04 | 8.96E-04 | 8.96E-04 | 8.96E-04 | 8.96E-04 | 8.96E-04 | 8.96E-04 | 8.96E-04 | 8.96E-04 | 8.96E-04 | 8.96E-04 | 8.96E-04 |
| MF17    | 2.07E-03 | 2.07E-03 | 2.07E-03 | 2.07E-03 | 2.07E-03 | 2.07E-03 | 2.07E-03 | 2.07E-03 | 2.07E-03 | 2.07E-03 | 2.07E-03 | 2.07E-03 |
| MF18    | 2.47E-03 | 2.47E-03 | 2.47E-03 | 2.47E-03 | 2.47E-03 | 2.47E-03 | 2.47E-03 | 2.47E-03 | 2.47E-03 | 2.47E-03 | 2.47E-03 | 2.47E-03 |
| MF19    | 4.02E-03 | 4.02E-03 | 4.02E-03 | 4.02E-03 | 4.02E-03 | 4.02E-03 | 4.02E-03 | 4.02E-03 | 4.02E-03 | 4.02E-03 | 4.02E-03 | 4.02E-03 |
| MF20    | 4.87E-03 | 4.87E-03 | 4.87E-03 | 4.87E-03 | 4.87E-03 | 4.87E-03 | 4.87E-03 | 4.87E-03 | 4.87E-03 | 4.87E-03 | 4.87E-03 | 4.87E-03 |

The proportions for each region from the 1st to the 100th generation are shown in Fig. 7 (note that the GS-MRPS uses the whole population to search one of the regions in one iteration, so the proportion of solutions in one region is 1 in an iteration, and the proportion of solutions in the remaining regions is 0. Hence it is excluded from this comparison study). Fig. 7 shows that GS-MPPO maintains almost equal proportions in the two regions and the proportion in each...
G. Li et al.: Multi-Objective Particle Swarm Optimization Based on Gaussian Sampling

TABLE 4. (Continued.) The mean and standard deviation values of \( IGD \) obtained by different algorithms.

| Algorithm | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | MBP |
|-----------|----|----|----|----|----|----|----|----|----|-----|
| GS-MOPSO | 4.43E-03 | 3.66E-03 | 2.91E-03 | 9.32E-04 | 1.09E-03 | 1.17E-02 | 5.49E-03 | 1.37E-03 | 1.57E-02 | 2.02E-01 |
| ZS-MRPS | 1.25E-02 | 3.68E-03 | 6.83E-03 | 9.12E-03 | 4.43E-03 | 5.78E-02 | 1.19E-03 | 4.45E-03 | 1.86E-02 | 5.47E-01 |
| DE_RLRF | 2.12E-03 | 5.32E-03 | 9.89E-03 | 8.29E-03 | 2.32E-03 | 4.25E-03 | 1.94E-03 | 4.00E-03 | 2.92E-03 | 2.54E-01 |
| SSMOPSO | 3.72E-03 | 3.81E-03 | 3.90E-03 | 5.82E-03 | 6.18E-03 | 4.31E-03 | 3.15E-03 | 7.39E-03 | 4.77E-03 | 6.51E-03 |
| MMOPIO | 2.25E-02 | 3.67E-03 | 4.25E-03 | 5.09E-02 | 5.95E-03 | 9.04E-02 | 7.78E-03 | 5.83E-03 | 3.16E-02 | 1.23E-02 |
| MMODE | 7.42E-03 | 3.61E-03 | 4.25E-03 | 5.95E-03 | 7.66E-03 | 3.67E-03 | 4.38E-03 | 6.03E-03 | 2.63E-02 | 6.31E-03 |
| SMPSO-MM | 5.13E-03 | 3.61E-03 | 4.54E-03 | 6.58E-03 | 7.66E-03 | 3.67E-03 | 4.38E-03 | 6.03E-03 | 2.63E-02 | 6.31E-03 |
| MO_Ring_PSO_SCD | 1.41E-02 | 3.94E-03 | 5.43E-03 | 9.34E-03 | 7.66E-03 | 3.67E-03 | 4.38E-03 | 6.03E-03 | 2.63E-02 | 6.31E-03 |
| TriMOEATA&R | 3.49E-03 | 3.61E-03 | 4.54E-03 | 5.95E-03 | 7.66E-03 | 3.67E-03 | 4.38E-03 | 6.03E-03 | 2.63E-02 | 6.31E-03 |
| DN_NSAGII | 3.49E-03 | 3.61E-03 | 4.54E-03 | 5.95E-03 | 7.66E-03 | 3.67E-03 | 4.38E-03 | 6.03E-03 | 2.63E-02 | 6.31E-03 |
| Omni-optimizer | 3.49E-03 | 3.61E-03 | 4.54E-03 | 5.95E-03 | 7.66E-03 | 3.67E-03 | 4.38E-03 | 6.03E-03 | 2.63E-02 | 6.31E-03 |

D. PARAMETER SENSITIVITY ANALYSIS

The parameter \( \varepsilon \) is designed to prevent the standard deviation from being a zero vector in Gaussian sampling mechanism. A large \( \varepsilon \) allows particles to explore a wide search space. However, this may lead to particles to deviate from

FIGURE 5. Comparison of PSs obtained by different algorithms on F8.
the original sample area. A small $\varepsilon$ would make particle to stagnate and even hardly escape from local optima, which is not conducive to locate more equivalent solutions. To investigate the sensitivity of the parameter $\varepsilon$, $\varepsilon = \{0, 10^{-1}, 10^{-2}, 10^{-3}, \ldots, 10^{-15}\}$ are tested on six test functions (MMF1-MMF6). Fig. 8 shows the distributions of the $PSP$ values obtained by the proposed GS-MOPSO with different values of $\varepsilon$.

As reported in Fig.8, it can be observed that the parameter $\varepsilon$ does affect the performance of algorithm. For MMF1, when the parameter $\varepsilon$ is set to $10^{-1}$ or $10^{-2}$, the algorithm achieves relatively high $PSP$ value and exhibits a satisfying performance. When $\varepsilon$ decreases from $10^{-2}$ to $10^{-4}$, the $PSP$ value gradually approaches the minimum $PSP$ value. Then, the $PSP$ value fluctuates slightly at the minimum $PSP$ value as $\varepsilon$ decreases from $10^{-4}$ to $0$. And a similar trend is also seen on MMF4, MMF5, and MMF6. For MMF2 and MMF3, the algorithm performance improves as $\varepsilon$ decreases from $10^{-1}$ to $10^{-3}$. When the value of $\varepsilon$ to be $10^{-3}$ GS-MOPSO obtains the highest $PSP$ value. Then the $PSP$ value fluctuates around the maximal $PSP$ value as $\varepsilon$ decreases from $10^{-3}$ to $0$. It is concluded that GS-MOPSO with $\varepsilon \in \{10^{-1}, 10^{-2}\}$ has better performance on MMF1, MMF4, MMF5, and MMF6, while GS-MOPSO performs better on MMF2 and MMF3 when
ε ∈ \{10^{-2}, 10^{-3}, 10^{-4}, \ldots, 10^{-15}, 0\}. Overall, when ε = 10^{-2}, GS-MOPSO has an acceptable performance on the six test functions.

E. APPLICATION
To further verify the algorithm’s performance, this section considers its application to a real-world multimodal multi-objective problem. One of the well-known problems is map-based test problem (MPB), which has three disconnected Pareto optimal subset [48]. This problem is solved with the proposed GS-MOPSO and the other twelve algorithms. And the results are provided in Table 3. From Table 3, the proposed GS-MOPSO obtains the highest \(PSP\) value than that of the other algorithms on the MBP. It can be concluded that GS-MOPSO shows a superior performance relative to the other algorithms. In addition, the Pareto solution sets obtained by each algorithm are displayed in Fig.9. It can be observed that the GS-MOPSO is able to cover all the three Pareto optimal regions. In contrast, for ZS-MRPS, MMOPIO, MMODE, PEN-MOBA, NMOHSA, SMPSO-MM, and MO_Ring_PSO_SCD, most parts of the Pareto optimal sets are obtained. For DE_RLRF, SSMOPSO, TriMOEATA&R, DN-NSGAII, and Omni-optimizer, only a few solutions are found. In conclusion, the GS-MOPSO algorithm is effective in solving the practical multimodal multi-objective problem.

F. THE ROLE OF GAUSSIAN SAMPLING
The role of the Gaussian sampling mechanism is to quickly identify the global optimal solution according to the best information of particles in the early stage of iteration, and to locate more potential solutions by exploiting the neighborhood of the optimal solution in the later stage of iteration. In order to verify its effectiveness, the MOPSO with the Gaussian sampling mechanism (denoted as MOPSO-I) is compared with the basic MOPSO. The results are provided in Table 2. It can be seen that the performance of MOPSO-I is better than that of MOPSO on all test functions, which indicated that the Gaussian sampling mechanism improves the performance of the algorithm.

The Gaussian sampling mechanism consists of two parts: global Gaussian sampling and local Gaussian sampling. These two parts play different roles in the searching process, which are analyzed as follows.

The global Gaussian sampling is designed to increase the exploration capability of the algorithm, while local Gaussian sampling is designed to boost the exploitation capability of the algorithm. Therefore, the role of global Gaussian sampling is considered to explore the whole search space and to find the global optimum as quickly as possible. The role of local Gaussian sampling is considered to exploit the neighborhood of the optimal solution and to obtain more promising solutions as accurately as possible.

To further analyze the role of Gaussian sampling, MOPSO-I compares with MOPSO-I-v1 (MOPSO-I without the global Gaussian sampling) and MOPSO-I-v2 (MOPSO-I without the local Gaussian sampling) on forty test functions. Table 5 presents the mean and standard deviation values of \(PSP\) obtained by MOPSO-I, MOPSO-I-v1, and MOPSO-I-v2. It can be observed that the performance of MOPSO-I is significantly better than or equal to that of MOPSO-I-v1 on all the test functions, which indicates that global Gaussian sampling can help this algorithm to obtain diverse solutions and improve its exploration ability. MOPSO-I is significantly superior to or equal to
MOPSO-I-v2 on all the test functions, which reveals that the local Gaussian sampling can help this algorithm to converge on promising solutions and enhance its exploitation power.

To intuitively illustrate the role of Gaussian sampling, the SYM-PART simple test function as an example is chosen. This test function has nine Pareto sets. The final population distribution of different algorithms on SYM_PART simple is shown in Fig. 10. It can be observed from Fig. 10(a) that MOPSO-I is capable to locate all optima solutions. In Fig.10(b), the whole population is distributed around the optimal solutions, which denotes that the local Gaussian sampling makes the MOPSO-I-v2 has good

| TABLE 5. Mean and standard deviation values of PSP obtained by different variants. |
|-----------------------------------|-------------------------------|-------------------------------|-----------------------------------|-------------------------------|-------------------------------|
| MOPSO-I                          | MOPSO-I-v1                     | MOPSO-I-v2                     | MOPSO-I                          | MOPSO-I-v1                     | MOPSO-I-v2                     |
|-----------------------------------|-------------------------------|-------------------------------|-----------------------------------|-------------------------------|-------------------------------|
| MMF1                              | 54.95(5.12)                   | 39.29(2.63)+                 | 27.57(2.76)+                     | 2.78(1.72)                    | 2.70(1.65)+                   |
| MMF2                              | 109.45(12.93)                 | 111.54(25.23)=               | 96.21(37.38)=                    | 4.82(0.47)                    | 4.73(0.54)=                   |
| MMF3                              | 141.23(32.66)                 | 30.50(23.99)=                | 53.93(26.70)+                    | 7.58(0.12)                    | 7.07(0.58)=                   |
| MMF4                              | 108.53(5.29)                  | 85.98(12.60)=                | 88.17(17.89)=                    | 2.87(0.20)                    | 2.18(0.75)=                   |
| MMF5                              | 27.98(1.13)                   | 20.18(1.30)=                 | 23.24(1.54)=                     | 4.69(0.92)                    | 4.70(0.91)=                   |
| MMF6                              | 28.02(2.13)                   | 25.52(0.66)=                 | 24.63(1.71)=                     | 4.69(0.92)                    | 4.64(0.82)=                   |
| MMF7                              | 67.14(15.36)                  | 65.62(9.36)=                 | 49.41(8.74)=                     | 18.96(4.26)                   | 14.51(5.41)=                  |
| MMF8                              | 48.30(4.18)                   | 39.20(4.94)=                 | 42.74(8.48)=                     | 18.79(1.19)                   | 18.00(4.82)=                  |
| MMF9                              | 50.64(8.80)                   | 50.69(10.67)=                | 22.05(9.80)=                     | 11.71(2.44)                   | 8.72(7.24)=                   |
| MMF10                             | 63.78(20.82)                  | 60.18(42.09)=                | 629.52(53.99)=                   | 10.16(0.60)                   | 2.46(0.53)=                   |
| MMF11                             | 134.92(60.81)                 | 1315.71(80.45)=              | 1342.59(36.26)=                  | 20.06(2.07)                   | 9.36(2.71)=                   |
| MMF12                             | 54.83(3.28)                   | 47.07(2.61)=                 | 39.24(5.19)=                     | 24.48(1.10)                   | 11.14(9.07)=                  |
| MMF13                             | 27.49(1.11)                   | 26.23(1.15)=                 | 26.84(0.84)=                     | 25.87(5.07)                   | 22.62(2.92)=                  |
| MMF15                             | 39.06(2.18)                   | 37.24(3.08)=                 | 37.68(0.99)=                     | 61.22(8.74)                   | 60.48(19.92)=                 |
| MMF1_e                            | 2.93(1.56)                    | 1.18(1.47)=                  | 0.68(0.32)=                      | 4.23(1.37)                    | 3.87(1.93)=                   |
| MMF14_e                           | 22.81(0.35)                   | 21.35(1.18)=                 | 22.23(1.03)=                     | 10.47(1.59)                   | 10.27(1.31)=                  |
| MMF15_e                           | 29.49(2.17)                   | 22.05(1.16)=                 | 20.48(1.14)=                     | 13.77(1.59)                   | 14.93(3.89)=                  |
| MMF10_e                           | 5.77(0.11)                    | 5.04(0.13)=                  | 5.74(0.62)=                      | 46.26(3.04)                   | 46.29(3.73)=                  |
| MMF11_e                           | 1.88(0.17)                    | 1.51(2.04)=                  | 0.41(0.06)=                      | 19.89(1.45)                   | 18.66(3.34)=                  |
| MMF12_e                           | 2.13(0.79)                    | 1.87(2.82)=                  | 0.45(0.08)=                      | 1.96(0.04)                    | 1.50(0.07)=                   |
| MMF13_e                           | 2.43(1.40)                    | 2.07(0.40)=                  | 1.78(0.03)=                      | 32/80                        | 29/110                       |

FIGURE 9. The comparison of PSs obtained by different algorithms on map-based test problem.
convergence performance. In Fig. 10 (c), the population distributed about the whole search space, which hints that the global Gaussian sampling makes the MOPSO-I-v2 has good diversity performance.

Overall, the comparison results illustrate that Gaussian sampling can help increase the performance of the algorithm.

G. LIMITATIONS AND FUTURE WORK

Though the proposed algorithm has good optimization performance on the majority of test functions, there are limitations of our research. On the one hand, the proposed algorithm does not perform well on test functions with discontinuous PS (e.g., MMF12_l, and F1) in Table 3. The reason is that the algorithm does not design relevant strategies according to the geometrical characteristics of PS. In fact, few strategies have been proposed by analyzing the characteristics of PS. Due to the PS geometry may be connected/disconnected, linear/nonlinear and other complex shapes, locating more solutions on complex PS is a very challenging task. One possible way to alleviate this difficulty is to adopt multiple strategies to guide the evolution of the population according to the different characteristics of PS. On the other hand, the parameter $\varepsilon$ is set to a fixed value in the proposed algorithm. According to the analysis in Section V.D, the parameter $\varepsilon$ affects the performance of the algorithm. In fact, the comparison algorithms (e.g. ZS-MRPS, SSMOPSO, and Tri-MOEATA&R) have similar situations. The reason is that it is difficult to determine the relationship between the parameter and the characteristics of PS. Therefore, future work should look deeply into this matter.

VI. CONCLUSION

In this paper, a multi-objective particle swarm optimization algorithm based on Gaussian sampling (GS-MOPSO) is proposed to solve multimodal multi-objective optimization problems. The Gaussian sampling mechanism adopts historical information of particles to divide multiple neighborhoods, and constantly search in the particle’s neighborhood to locate more potential solutions in the decision space. In addition, an external archive maintenance strategy is utilized to improve the quality of the obtained solutions in the decision space. Moreover, a set of multimodal multi-objective test problems are proposed in this paper, i.e., F1−F9.

The proposed GS-MOPSO algorithm is compared with twelve multimodal multi-objective algorithms. All the algorithms are tested on forty multimodal multi-objective test functions. The experimental results show that GS-MOPSO has better performance than the compared algorithms in terms of PSP and is able to locate more and better distributed Pareto-optimal solutions. Finally, the proposed GS-MOPSO is applied to a real world problem, the map-based test problem. Results indicate that GS-MOPSO is feasible and effective.

In a future study, some of the real-world problems will be modeled to generate new multimodal multi-objective test functions with different characteristics. In addition, future work should further extend the problem set with multimodal many-objective or higher dimensional optimization.

APPENDIX

A. DETAILS OF TEST FUNCTIONS

F1

\[
x_2 = \begin{cases} 
0 \leq |x_2| \leq 0.5 |1.5 \leq |x_2| \leq 2 \\
0.5 \leq |x_2| < 1.5 
\end{cases} 
\]

\[
f_1 = |x_1| \\
f_2 = 1 - |x_1| + 2(x_2 - \text{sgn}(0.7 |x_1| \pi + \frac{\pi}{4}))^2. 
\]

where $0 \leq x_1 \leq 10$, $-2 \leq x_2 \leq 2$.

Its true PS is

\[
\begin{cases} 
x_1 = x_1 \\
x_2 = \text{sgn}(0.7 |x_1| \pi + \frac{\pi}{4}) 
\end{cases}
\]

where $0 \leq x_1 \leq 10$.

Its true PF is

\[
f_2 = 1 - f_1 
\]

where $0 \leq f_1 \leq 10$.

Its true PS and PF are illustrated in Fig. 11.
FIGURE 11. Illustration of the true PSs and PF of F1.

F2

\[
x_2 = \begin{cases} 
  x_2 & 0 \leq x_2 \leq 3 \\
  |x_2| & -3 \leq x_2 < 0 
\end{cases}
\]

\[
f_2 = \begin{cases} 
  f_1 = |x_1| \\
  f_2 = 1 - |x_1| + 2(x_2 - |\sin(|x_1| \pi + \frac{\pi}{2})| - \frac{\pi}{4})^2
\end{cases}
\]

(A5)

where \(-10 \leq x_1 \leq 10, -3 \leq x_2 \leq 3.

Its true PS is

\[
x_2 = x_2 \\
x_1 = \sqrt{|x_2|} \\
\]

(A7)

Its true PF is

\[
f_2 = 1 - f_1
\]

(A8)

where \(0 \leq f_1 \leq 10.

Its true PS and PF are illustrated in Fig. 12.

F3

\[
x_2 = \begin{cases} 
  x_2 & 0 \leq x_2 \leq 2 \\
  |x_2| & -2 \leq x_2 < 0 
\end{cases}
\]

(A9)

where \(-2 \leq x_2 \leq 0.

Its true PS and PF are illustrated in Fig. 12.

F4

\[
x_2 = \begin{cases} 
  x_2 & 0 \leq x_2 \leq 2 \\
  |x_2| & -2 \leq x_2 < 0 
\end{cases}
\]

(A13)

where \(-10 \leq x_1 \leq 10, -3 \leq x_2 \leq 3.

Its true PS is

\[
x_2 = x_2 \\
x_1 = \sqrt{|x_2|} \\
\]

(A11)

Its true PF is

\[
f_2 = 1 - \sqrt{f_1}
\]

(A12)

where \(0 \leq f_1 \leq 1.

Its true PS and PF are illustrated in Fig. 13.
Its true PF is

\[ f_1 = x_1 \]

\[ f_2 = \begin{cases} 
1 - \sqrt{x_1} + 2(4|x_2| - \sqrt{x_1})^2 \\
-2 \cos(\frac{20(|x_2| - \sqrt{x_1})\pi}{\sqrt{2}}) + 2 
\end{cases} \\
0 \leq |x_2| \leq 0.5, 0.5 < |x_2| < 1 \& 0.25 < x_1 \leq 1 \\
1 - \sqrt{x_1} + 2(4|x_2| - 0.5 - \sqrt{x_1})^2 \\
-\cos(\frac{20(|x_2| - 0.5 - \sqrt{x_1})\pi}{\sqrt{2}}) + 2 \\
1 \leq |x_2| \leq 1.5, 0 \leq x_1 < 0.25 & 0.5 < |x_2| < 1 \\
\end{cases} \]

(A14)

where 0 \leq x_1 \leq 1, -1.5 \leq x_2 \leq 1.5.

Its true PS is

\[ \begin{cases} 
x_2 = x_2 \\
x_1 = \begin{cases} 
0 \leq |x_2| \leq 0.5, 0.5 < |x_2| < 1 & 1 \& 0.25 < x_1 \leq 1 \\
(x_2 - 0.5)^2 1 \leq |x_2| \leq 1.5, 0 \leq x_1 < 0.25 & 0.5 < |x_2| < 1 
\end{cases} 
\end{cases} \]

(A15)

Its true PF is

\[ f_2 = 1 - \sqrt{f_1} \]

(A16)

where 0 \leq f_1 \leq 1.

Its true PS and PF are illustrated in Fig. 14.

F5

\[ f_1 = \begin{cases} 
|x_1| + \frac{2}{|J_1|} \sum_{j \in J_1} x_j \\
[x_j - \sin(6\pi |x_1| + \frac{j\pi}{n})]^2 - 1 \leq x_1 \leq 0 \\
x_1 + \frac{2}{|J_1|} \sum_{j \in J_1} x_j^2 0 < x_1 \leq 1 \\
1 - \sqrt{|x_1| + \frac{2}{|J_2|} \sum_{j \in J_2} x_j^2} - 1 \leq x_1 \leq 0 \\
1 - \sqrt{|x_1| + \frac{2}{|J_2|} \sum_{j \in J_2} x_j^2} 0 < x_1 \leq 1 
\end{cases} \]

(A17)

where -1 \leq x_i \leq 1, J_1 = \{j|j\text{ is odd and } 2 \leq j \leq n\}, J_2 = \{j|j\text{ is even and } 2 \leq j \leq n\}, and

\[ x_j = \begin{cases} 
0.3x_1^3 \cos(24\pi x_1 + \frac{4\pi j}{n}) + 0.6x_1 \\
\cos(6\pi x_1 + \frac{j\pi}{n}) j \in J_1 \\
x_j = \begin{cases} 
0.3x_1^3 \cos(24\pi x_1 + \frac{4\pi j}{n}) + 0.6x_1 \\
\sin(6\pi x_1 + \frac{j\pi}{n}) j \in J_2 
\end{cases} 
\end{cases} \]
Its true PS is
\[
x_2 = \begin{cases} 
\sin(6\pi |x_1| + j\pi) \\
[0.3x_1^2 \cos(24\pi x_1 + 4j\pi) + 0.6x_1] \\
\cos(6\pi x_1 + j\pi) \\
[0.3x_1^2 \cos(24\pi x_1 + 4j\pi) + 0.6x_1] \\
sin(6\pi x_1 + j\pi) \\
\end{cases} \quad j \in J_1 \\
\begin{cases} 
x_j - \sin(4.5\pi |x_1| + j\pi) \\
\sin(4.5\pi |x_1| + j\pi) \\
\end{cases} \quad j \in J_2 \\
0 < x_1 \leq 1 \\
\] (A18)

Its true PF is
\[
f_2 = 1 - \sqrt{f_1} \\
\] (A19)

where $0 \leq f_1 \leq 1$.

Its true PS and PF are illustrated in Fig. 15.

F6
\[
\begin{cases} 
f_1 = \sqrt{x_1} + \frac{2}{|J_1|} \sum_{j \in J_1} y_j^2 \\
f_2 = 1 - \sqrt{x_1} + \frac{2}{|J_2|} \sum_{j \in J_2} y_j^2 \\
\end{cases} \\
\] (A20)

where $-1 \leq x_i \leq 1$, $J_1 = \{j | j \text{ is odd and } 2 \leq j \leq n\}$, $J_2 = \{j | j \text{ is even and } 2 \leq j \leq n\}$, and $y_j = \begin{cases} 
x_j - \sin(4.5\pi |x_1| + j\pi) \\
\sin(4.5\pi |x_1| + j\pi) \\
\end{cases} \quad 0 \leq x_1 \leq 1$.

Its true PS is
\[
x_2 = \begin{cases} 
\sin(4.5\pi |x_1| + j\pi) \quad -1 \leq x_1 < 0 \\
\sin(4\pi |x_1| + j\pi) \\
\end{cases} \quad 0 \leq x_1 \leq 1 \\
\] (A21)

Its true PF is
\[
f_2 = 1 - f_1 \\
\] (A22)

where $0 \leq f_1 \leq 1$.

Its true PS and PF are illustrated in Fig. 16.

F7
\[
\begin{cases} 
f_1 = |x_1| + \frac{2}{|J_1|} \sum_{j \in J_1} y_j^2 \\
f_2 = 1 - |x_1| + \frac{2}{|J_2|} \sum_{j \in J_2} y_j^2 \\
\end{cases} \\
\] (A23)
where $-1 \leq x_1 \leq 1$, $J_1 = \{j|j \text{ is odd and } 2 \leq j \leq n\}$, $J_2 = \{j|j \text{ is even and } 2 \leq j \leq n\}$, and $y_j = \begin{cases} x_j - [0.3 \cos(24\pi |x_1| + \frac{4j\pi}{n})] & j \in J_1 \\ \cos(6\pi |x_1| + \frac{j\pi}{n}) & j \in J_1 \\ x_j - [0.3 \cos(24\pi |x_1| + \frac{4j\pi}{n})] & j \in J_2 \\ \sin(6\pi |x_1| + \frac{j\pi}{n}) & j \in J_2 \end{cases}$.

Its true PS is

$$x_j = \begin{cases} 0.3 \cos(24\pi |x_1| + \frac{4j\pi}{n}) \cos(6\pi |x_1| + \frac{j\pi}{n}) & j \in J_1 \\ 0.3 \cos(24\pi |x_1| + \frac{4j\pi}{n}) \sin(6\pi |x_1| + \frac{j\pi}{n}) & j \in J_2 \end{cases}$$  \hspace{1cm} (A24)

where $-1 \leq x_1 \leq 1$.

Its true PF is

$$f_2 = 1 - \sqrt{f_1}$$  \hspace{1cm} (A25)

where $0 \leq f_1 \leq 1$.

Its true PS and PF are illustrated in Fig. 17.

F8

$$\begin{cases} f_1 = |x_1| + \frac{2}{|J_1|} \sum_{j \in J_1} h(y_j) \\ f_2 = 1 - x_1^2 + \frac{2}{|J_2|} \sum_{j \in J_2} h(y_j) \end{cases}$$  \hspace{1cm} (A26)

where $J_1 = \{j|j \text{ is odd and } 2 \leq j \leq n\}$, $J_2 = \{j|j \text{ is even and } 2 \leq j \leq n\}$, $-1 \leq x_1 \leq 1$, $-1 \leq x_2 \leq 1$, $-1 \leq x_3 \leq 1$, and $y_j = x_j - |x_1|^{0.5(1.0 + \frac{3(0-j)}{n-2})}$, $h(t) = \frac{|t|}{1 + |t|}, j = 2, \ldots, n$.

Its true PS is

$$x_j = x_1^{0.5(1.0 + \frac{3(0-j)}{n-2})}$$  \hspace{1cm} (A27)

Its true PF is

$$f_2 = 1 - f_1^2$$  \hspace{1cm} (A28)

where $0 \leq f_1 \leq 1$.

Its true PS and PF are illustrated in Fig. 18.
Its true PF is

\[ f_1 + f_2 + f_3 = 1 \quad 0 \leq f_1, f_2, f_3 \leq 1 \]  

(A31)

Its true PS and PF are illustrated in Fig. 19.

**REFERENCES**

[1] H. Li, K. Deb, and Q. Zhang, “Variable-length Pareto optimization via decomposition-based evolutionary multiobjective algorithm,” *IEEE Trans. Evol. Comput.*, vol. 23, no. 6, pp. 987–999, Dec. 2019.

[2] K. Li, R. Chen, G. Fu, and X. Yao, “Two-archive evolutionary algorithm for constrained multiobjective optimization,” *IEEE Trans. Evol. Comput.*, vol. 23, no. 2, pp. 305–315, Apr. 2019.

[3] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, “A new two-stage evolutionary algorithm for many-objective optimization,” *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 748–761, Oct. 2019.

[4] D. H. Moraes, D. S. Sanches, J. da Silva Rocha, J. M. C. Garbelini, and M. F. Castoldi, “A novel multi-objective evolutionary algorithm based on subpopulations for the bi-objective traveling salesman problem,” *Soft Comput.*, vol. 23, no. 15, pp. 6157–6168, Aug. 2019.

[5] J. Luo, Y. Yang, X. Li, Q. Liu, M. Chen, and K. Gao, “A decomposition-based multi-objective evolutionary algorithm with quality indicator,” *Swarm Evol. Comput.*, vol. 39, pp. 339–355, Apr. 2018.

[6] Y. Zhang, D.-W. Gong, J.-Y. Sun, and B.-Y. Qu, “A decomposition-based archiving approach for multi-objective evolutionary optimization,” *Inf. Sci.*, vol. 430–431, pp. 397–413, Mar. 2018.

[7] W. L. Wang, W. K. Li, Z. Wang, and L. Li, “Opposition-based multi-objective whale optimization algorithm with global grid ranking,” *Neurocomputing*, vol. 341, pp. 41–59, May 2019.

[8] S. Lalwani, H. Sharma, S. C. Satapathy, K. Deep, and J. C. Bansal, “A survey on parallel particle swarm optimization algorithms,” *Arabian J. Sci. Eng.*, vol. 44, no. 4, pp. 2899–2923, Apr. 2019.

[9] T. Chugh, K. Sindhiya, J. Hakanen, and K. Miettinen, “A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms,” *Soft Comput.*, vol. 23, no. 9, pp. 3137–3166, May 2019.

[10] J. J. Liang, C. T. Yue, and B. Y. Qu, “Multimodal multi-objective optimization: A preliminary study,” in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, Jul. 2016, pp. 2454–2461.

[11] C. Yue, B. Qu, and J. Liang, “A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems,” *IEEE Trans. Evol. Comput.*, vol. 22, no. 5, pp. 805–817, Oct. 2018.

[12] Y. Liu, H. Ishibuchi, and Y. Nojima, “A double-niched evolutionary algorithm and its behavior on polygon-based problems,” in *Proc. Int. Conf. Parallel Problem Solving Nature*. Cham, Switzerland: Springer, 2018, pp. 262–273.

[13] Y. Peng, H. Ishibuchi, and K. Shang, “Multi-modal multi-objective optimization: Problem analysis and case studies,” in *Proc. IEEE Symp. Ser. Comput. Intel. (SSCI)*, Xi’an, China, Dec. 2019, pp. 1865–1872.

[14] Y. Liu, H. Ishibuchi, G. G. Yen, Y. Nojima, and N. Masuyama, “Handling imbalance between convergence and diversity in the decision space in evolutionary multi-modal multi-objective optimization,” *IEEE Trans. Evol. Comput.*, vol. 24, no. 3, pp. 551–565, Jun. 2020, doi: 10.1109/TEVC.2019.2938557.

[15] Y. Liu, G. G. Yen, and D. Gong, “A multimodal multiobjective evolutionary algorithm using two-archive and recombination strategies,” *IEEE Trans. Evol. Comput.*, vol. 23, no. 4, pp. 660–674, Aug. 2019.

[16] J. J. Liang, Q. Guo, C. Yue, and B. Y. Qu, “A self-organizing multi-objective particle swarm optimization algorithm for multimodal multi-objective problems,” in *Proc. Int. Conf. Swarm Intell.*, Shanghai, China, 2018, pp. 550–560.

[17] B. Qu, C. Li, J. Liang, L. Yan, K. Yu, and Y. Zhu, “A self-organized speciation based multi-objective particle swarm optimizer for multimodal multi-objective problems,” *Appl. Soft Comput.*, vol. 86, Jan. 2020, Art. no. 105886.

[18] R. Thomsen, “Multimodal optimization using crowding-based differential evolution,” in *Proc. IEEE Congr. Evol. Comput.*, vol. 2, Jun. 2004, pp. 1382–1389.

[19] D. E. Goldberg and J. Richardson, “Genetic algorithms with sharing for multimodal function optimization,” in *Proc. 2nd Int. Conf. Genetic Algorithms Genetic Algorithms Their Appl.*. Hillsdale, NJ, USA, 1987, pp. 41–49.

[20] A. Petrovski, “A clearing procedure as a niching method for genetic algorithms,” in *Proc. IEEE Int. Conf. Evol. Comput.*, May 1996, pp. 798–803.

[21] J.-P. Li, M. E. Balazs, G. T. Parks, and P. J. Clarkson, “A species conserving genetic algorithm for multimodal function optimization,” *Evolution Comput.*, vol. 10, no. 3, pp. 207–234, Sep. 2002.

[22] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proc. IEEE Congr. Evol. Comput.*, vol. 23, no. 4, pp. 660–674, Aug. 2019.

[23] D. E. Goldberg, “Genetic algorithms in search, optimization, and machine learning,” Boston, MA, USA: Addison-Wesley, 1989.

[24] M. Ghasemi, E. Akbari, A. Rahimnejad, S. E. Razavi, S. Ghavidel, and L. Li, “Phasor particle swarm optimization: A simple and efficient variant of PSO,” *Soft Comput.*, vol. 23, no. 19, pp. 9701–9718, Oct. 2019.

[25] L. Li, “Phasor particle swarm optimization: A simple and efficient variant of PSO,” *Soft Comput.*, vol. 23, no. 19, pp. 9701–9718, Oct. 2019.

[26] X. Zhang, X. Zheng, R. Cheng, J. Qiu, and Y. Jin, “A competitive mechanism based multi-objective particle swarm optimizer with fast convergence,” *Inf. Sci.*, vol. 427, pp. 63–76, Feb. 2018.

[27] Y. Chen, L. Li, H. Peng, J. Xiao, and Q. Wu, “Dynamic multi-swarm differential learning particle swarm optimizer,” *Swarm Evol. Comput.*, vol. 39, pp. 209–221, Apr. 2018.

[28] Q. Yang, W.-N. Chen, J.-H. Deng, Y. Li, T. Gu, and J. Zhang, “A level-based learning swarm optimizer for large-scale optimization,” *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 578–594, Aug. 2018.

[29] F. Kong, J. Jiang, and Y. Huang, “An adaptive multi-swarm competition particle swarm optimizer for large-scale optimization,” *Mathematica*, vol. 7, no. 6, pp. 521–525, 2018.

[30] D. Wang, D. Tan, and L. Liu, “Particle swarm optimization algorithm: An overview,” *Soft Comput.*, vol. 22, no. 2, pp. 387–408, Jan. 2018.

[31] N. Lynn, M. Z. Ali, and P. N. Suganthan, “Population topologies for particle swarm optimization and differential evolution,” *Swarm Evol. Comput.*, vol. 39, pp. 24–35, Apr. 2018.

[32] J. Yao, J. Pan, Y. Han, and L. Wang, “Application of particle swarm optimization with stochastic inertia weight and adaptive mutation in target localization,” in *Proc. Int. Conf. Comput. Appl. Syst. Modelling (ICCASM)*, Oct. 2010, pp. 251–254.
[33] K. Deb and S. Tiwari, “Omnivar: A procedure for single and multi-objective optimization,” in Proc. Int. Conf. Evol. Multi-Criterion Optim., 2005, pp. 47–61.

[34] T. Ulrich, J. Bader, and E. Zitzler, “Integrating decision space diversity into hyper-volume-based multiobjective search,” in Proc. 12th Genet. Evol. Comput. Conf., New York, NY, USA, 2010, pp. 455–462.

[35] K. P. Chan and T. Ray, “An evolutionary algorithm to maintain diversity in the parametric and the objective space,” in Proc. Int. Conf. Comput. Intell., Robot. Auton. Syst., 2005, pp. 13–16.

[36] G. Rudolph, B. Naujoks, and M. Preuss, “Capabilities of EMOA to detect and preserve equivalent Pareto subsets,” in Proc. Int. Conf. Evol. Multi-Criterion Optim., 2007, pp. 36–50.

[37] A. Zhou, Q. Zhang, and Y. Jin, “Approximating the set of Pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm,” IEEE Trans. Evol. Comput., vol. 13, no. 5, pp. 1167–1189, Oct. 2009.

[38] H. Xia, J. Zhuang, and D. Yu, “Combining crowding estimation in objective and decision space with multiple selection and search strategies for multi-objective evolutionary optimization,” IEEE Trans. Cybern., vol. 44, no. 3, pp. 378–393, Mar. 2014.

[39] R. Tanabe and H. Ishibuchi, “A decomposition-based evolutionary algorithm for multi-modal multi-objective optimization,” in Proc. Int. Conf. Parallel Problem Solving Nature. Cham, Switzerland: Springer, 2018, pp. 249–261.

[40] Z. Li, L. Shi, C. Yue, Z. Shang, and B. Qu, “Differential evolution based on reinforcement learning with fitness ranking for solving multimodal multiobjective problems,” Swarm Evol. Comput., vol. 49, pp. 234–244, Sep. 2019.

[41] J. Azezeh, R. Rasras, and Z. Alqadi, “Adaptation of MATLAB K-means clustering function to create Color Image Features,” Int. J. Res. Adv. Eng. Technol., vol. 5, no. 2, pp. 10–18, 2019.

[42] H. Xie, L. Zhang, C. P. Lim, Y. Yu, C. Liu, H. Liu, and J. Walters, “Improving K-means clustering with enhanced firefly algorithms,” Appl. Soft Comput., vol. 84, Nov. 2019, Art. no. 105763.

[43] M. Pelikan and D. E. Goldberg, “Genetic algorithms, clustering, and the breaking of symmetry,” in Proc. Parallel Problem Solving Nature (PPSN VI), Paris, France, 2000, pp. 385–394.

[44] X. Li, C. Lu, L. Gao, S. Xiao, and L. Wen, “An effective multiobjective algorithm for energy-efficient scheduling in a real-life welding shop,” IEEE Trans. Ind. Informat., vol. 14, no. 12, pp. 5400–5409, Dec. 2018.

[45] T. Fan, J. Wang, and M. Feng, “Application of multi-objective firefly algorithm based on archive learning in robot path planning,” Int. J. Intell. Inf. Database Syst., vol. 12, no. 3, pp. 199–211, 2019.

[46] Q. Zhang, A. Zhou, and S. Zhao, “Multiobjective optimization test instances for the CEC 2009 special session and competition,” Dept. Comput. Sci. Electron. Eng., Univ. Essex, Colchester, U.K., Dept. Elect. Electron. Eng., Nanyang Technol. Univ., Singapore, Tech. Rep. CES-487, 2008.

[47] J. J. Liang, P. N. Suganthan, and B. Y. Qu, “Problem definitions and evaluation criteria for the CEC 2020 special session on multimodal multiobjective optimization,” in Proc. IEEE Conge. Evol. Comput., Zhengzhou, China, 2019, pp. 1–22.

[48] H. Ishibuchi, N. Akedo, and Y. Nojima, “A many-objective test problem for visually examining diversity maintenance behavior in a decision space,” in Proc. 13th Annu. Conf. Genetic Evol. Comput., 2011, pp. 649–656.

[49] H. Ishibuchi, R. Imada, Y. Setoguchi, and Y. Nojima, “Reference point specification in inverted generational distance for triangular linear Pareto front,” IEEE Trans. Evol. Comput., vol. 22, no. 6, pp. 961–975, Dec. 2018.

[50] Y. Tian, R. Cheng, X. Zhang, F. Cheng, and Y. Jin, “An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility,” IEEE Trans. Evol. Comput., vol. 22, no. 4, pp. 609–622, Aug. 2018.

[51] Q. Fan and X. Yan, “Solving multimodal multiobjective problems through zoning search,” IEEE Trans. Syst., Man, Cybern. Syst., early access, Oct. 9, 2019, doi: 10.1109/TCMC.2019.2944338.

[52] Y. Hu, J. Wang, J. Liang, K. Yu, H. Song, Q. Guo, C. Yue, and Y. Wang, “A self-organizing multimodal multi-objective pigeon-inspired optimization algorithm,” Sci. China Inf. Sci., vol. 62, no. 7, p. 70206, Jul. 2019.

[53] J. Liang, W. Xu, C. Yue, K. Yu, H. Song, O. D. Crisalle, and B. Qu, “Multimodal multiobjective optimization with differential evolution,” Swarm Evol. Comput., vol. 44, pp. 1028–1059, Feb. 2019.

[54] L. Yan, G. S. Li, Y. C. Jiao, B. Y. Qu, C. T. Yue, and S. K. Qu, “A performance enhanced niching multi-objective bat algorithm for multimodal multi-objective problems,” in Proc. IEEE Conge. Evol. Comput., Jun. 2019, pp. 1275–1282.

[55] B. Y. Qu, G. S. Li, and Q. Q. Guo, “A niching multi-objective harmony search algorithm for multimodal multi-objective problems,” in Proc. IEEE Conge. Evol. Comput., Wellington, New Zealand, Jun. 2019, pp. 1267–1274.

[56] C. Gan, W. Cao, M. Wu, and X. Chen, “A new bat algorithm based on iterative local search and stochastic inertia weight,” Expert Syst. Appl., vol. 104, pp. 202–212, Aug. 2018.