Efficient Generalized Surrogate-Assisted Evolutionary Algorithm for High-Dimensional Expensive Problems

Xiwen Cai, Liang Gao, and Xinyu Li

Abstract—**Engineering optimization problems usually involve computationally expensive simulations and many design variables. Solving such problems in an efficient manner is still a major challenge. In this paper, a generalized surrogate-assisted evolutionary algorithm is proposed to solve such high-dimensional expensive problems. The proposed algorithm is based on the optimization framework of the genetic algorithm (GA). This algorithm proposes to use a surrogate-based trust region local search method, a surrogate-guided GA (SGA) updating mechanism with a neighbor region partition strategy and a prescreening strategy based on the expected improvement infilling criterion of a simplified Kriging in the optimization process. The SGA updating mechanism is a special characteristic of the proposed algorithm. This mechanism makes a fusion between surrogates and the evolutionary algorithm. The neighbor region partition strategy effectively retains the diversity of the population. Moreover, multiple surrogates used in the SGA updating mechanism make the proposed algorithm optimize robustly. The proposed algorithm is validated by testing several high-dimensional numerical benchmark problems with dimensions varying from 30 to 100, and an overall comparison is made between the proposed algorithm and other optimization algorithms. The results show that the proposed algorithm is very efficient and promising for optimizing high-dimensional expensive problems.**

Index Terms—**High-dimensional expensive problems, multiple surrogates, prescreening strategy, simplified Kriging, surrogate-assisted evolutionary algorithm, surrogate-guided crossover operation, trust region method.**

I. Introduction

EVOLUTIONARY or metaheuristic algorithms, such as the genetic algorithm (GA) [1], particle swarm optimization (PSO) [2], and differential evolution (DE) [3] are very popular and widely applied in engineering optimization. Previous studies [4]–[8] have shown that these algorithms can handle high-dimensional optimization problems well. However, in the optimization process, the function calls are usually numerous and can reach hundreds of thousands. This process is acceptable because these common optimization problems are usually very inexpensive and the time to run a function evaluation can be negligible. However, if these metaheuristic algorithms are applied to expensive optimization problems that involve computationally expensive simulations, the computational cost will be tremendous and even prohibitive. According to Simpson *et al.* [9], Ford Motor Company spends approximately 36–160 h running one car crash simulation, which means a function call in the common optimization problems. The differences between expensive and common optimization problems are presented in Fig. 1. The response function of expensive problems is usually a simulation model [finite element analysis (FEA), computational fluid dynamics (CFD), and so on]. One function evaluation refers to a simulation and is usually very expensive compared with a function evaluation in common optimization problems. A promising approach to reduce the computation time for optimizing highly time-consuming problems is to employ computationally inexpensive approximation models (surrogates) to replace in part the computationally expensive function evaluations. With the "curse of dimensionality," common surrogates, such as Kriging [10], radial basis function (RBF) [11], support vector regression (SVR) [12], polynomial response surface (PRS) [13], and so on usually cannot provide highly accurate results for problems with more than ten dimensions according to previous studies [14]–[16]. The deeper reasons could be that only limited data points can be utilized and the optimal hyperparameters of these surrogates are difficult to be obtained in approximating high-dimensional expensive

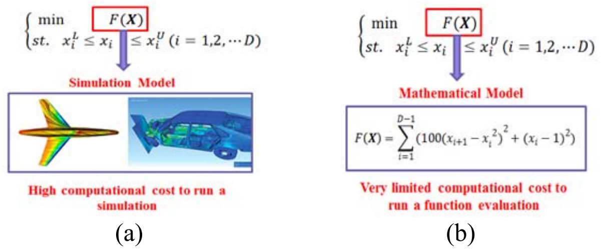
min F(X)
st. $x_i^L \le x_i \le x_i^U$ ($i = 1,2,\cdots D$)
Simulation Model
High computational cost to run a simulation
(a)
min F(X)
st. $x_i^L \le x_i \le x_i^U$ ($i = 1,2,\cdots D$)
Mathematical Model
$F(\mathbf{X}) = \sum_{i=1}^{D-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$
Very limited computational cost to run a function evaluation
(b)

Fig. 1. Differences between (a) expensive and (b) common optimization problems.

Manuscript received July 9, 2018; revised December 6, 2018, February 27, 2019, and April 17, 2019; accepted May 23, 2019. Date of publication May 31, 2019; date of current version March 31, 2020. This work was supported in part by the National Natural Science Foundation for Distinguished Young Scholars of China under Grant 51825502, in part by the National Natural Science Foundation of China under Grant 51805180 and Grant 51775216, in part by the Natural Science Foundation of Hubei Province under Grant 2018CFA078, and in part by the Program for Huazhong University of Science and Technology Academic Frontier Youth Team under Grant 2017QYTD04. *(Corresponding author: Liang Gao.)*

The authors are with the State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: 864533724@qq.com; gaoliang@mail.hust.edu.cn; lixinyu@mail.hust.edu.cn).

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.2019.2919762

problems. Therefore, the surrogate-based optimization methods, which heavily rely on the accuracy of surrogates, could be unsuitable for handling high-dimensional expensive problems. For instance, some surrogate-based optimization methods optimize by directly using the optimum of the built global surrogate. If the global surrogate is not accurate, an improved optimum could be difficult to be found. In addition, the lengthy time for Kriging metamodeling could be another reason that makes the Kriging-based optimization methods unsuitable for high-dimensional problems [17], [18]. Moreover, efficient exploration in high-dimensional design space is usually very difficult within limited data points.

On the one hand, the surrogate-assisted evolutionary or metaheuristic algorithms reviewed in [19]–[21] have recently received increasing attention for addressing such high dimensional expensive optimization problems with more than 20 dimensions. These algorithms usually assume the metaheuristic algorithms to be the primary optimization frameworks and consider the surrogates to be additional tools to accelerate the convergence of the basic metaheuristic algorithms. These algorithms usually do not have high accuracy requirements for the surrogates. According to our investigation, surrogate-assisted metaheuristic algorithms usually have some specific ways to accelerate the convergence of metaheuristic algorithms. These ways could be hybridly utilized to improve the optimization efficiency of metaheuristic algorithms. The ways include the use of surrogate prescreening, optima of surrogates, surrogate-based local search, and so on.

First, metaheuristic algorithms assisted by surrogate prescreening are widely used for optimizing high-dimensional expensive problems. Some prescreening strategies are based on the Kriging surrogate (also known as the Gaussian model). These strategies usually utilize the Kriging-based infill sampling criteria, such as expected improvement (EI) [22], [23], probability of improvement (PoI) [24], and lower confidence bound (LCB) [25]–[28] to prescreen promising candidate offspring points produced by the basic metaheuristic algorithms. Among these algorithms, the LCB-assisted DE algorithm (called GPEME), proposed by Liu *et al.* [25], shows high efficiency in optimization for medium-scale expensive problems. Developed by Liu *et al.* [29], this algorithm can solve expensive problems with inequality constraints. Notably, some non-Kriging-based infilling criteria [30], [31] can also be utilized for prescreening. In addition, some prescreening strategies are based on the predicted response of surrogates. These strategies usually use the surrogate's response function to rank the candidate offspring points produced by basic metaheuristic algorithms and then select the points with small predicted responses as the offspring points. For example, Fonseca *et al.* [32] used a surrogate for fitness inheritance to assist a GA in solving optimization problems with limited computational budget. Regis [33] utilized an RBF surrogate to identify the most promising trial position for each particle in a swarm. Regis [34] also combined a similar prescreening strategy with evolutionary programming for optimization of high-dimensional constrained expensive problems. Mallipeddi and Lee [35] used surrogates to generate competitive offspring points among trial offspring points. Gong *et al.* [36] used an inexpensive density function model to select the most promising candidate offspring point. Sun *et al.* [37] used an RBF surrogate to select the best candidate offspring particle in an RBF-assisted social learning PSO. Moreover, some prescreening strategies are based on a comparison of the candidate point's predicted response provided by surrogates with the father point's real response, and this comparison is then used to decide whether the candidate point should be evaluated by the real response function. For example, Praveen and Duvigneau [38] proposed an RBF-assisted PSO algorithm for an aerodynamic shape design. This algorithm uses a surrogate to screen promising particles among the candidate offspring swarm particles in the optimization process. A similar algorithm has also been proposed by Sun *et al.* [39]. The difference is that they use an accurate two-layer surrogate construction scheme for improving optimization efficiency. In addition, Elsayed *et al.* [40] used a Kriging surrogate to suitably select the parameters of a DE algorithm to accelerate its convergence.

Second, metaheuristic algorithms assisted by the optima of surrogates usually use the predicted global optimum provided by surrogates to replace the current best population point if this optimum is better than the current best population point. For example, Parno *et al.* [41] used a Kriging surrogate to improve the efficiency of the PSO. Tang *et al.* [42] used a hybrid global surrogate model consisting of a quadratic polynomial model and RBF to develop a surrogate-based PSO. Regis [33] used the RBF's global optimum obtained in a local region to assist the PSO. Yu *et al.* [43] used the optimum provided by a local RBF surrogate built around the current best particle to speed up a PSO search process.

Third, metaheuristic algorithms assisted by surrogate-based local search usually use the surrogate-based local search first and then use the search mechanism of basic metaheuristic algorithms for global optimization. For example, Ong *et al.* [44] employed a trust region method for the interleaved use of exact models for the objective and constrained functions with computationally inexpensive RBF surrogates during a local search. The GA operator was then run for global optimization. Local search based on a trust region method and surrogates is very common in surrogate-assisted metaheuristic algorithms. These algorithms can be read about in references, such as [45]–[48]. In addition, Wang *et al.* [49] recently used an RBF-based interior point local search method to assist DE optimization of constrained expensive problems. Although current surrogate-assisted metaheuristic algorithms can handle high-dimensional expensive problems well, most of these algorithms still need a large number of function evaluations that are usually greater than thousands to obtain good optimization results in the optimization process. Moreover, these algorithms are usually developed for optimizing problems whose dimensions are usually less than 30. For example, the generalized surrogate single-objective memetic algorithm (GS-SOMA) proposed by Lim *et al.* [47] needs 8000 function evaluations for 30-D problems. The surrogate-assisted DE algorithm (ESMDE) proposed by Mallipeddi and Lee [35] needs more than 10 000 function evaluations for 30-D problems.

On the other hand, methods based on cut-high-dimensional model representation (Cut-HDMR) [50] are also promising to ease the curse of dimensionality. It can decompose a high-dimensional problem into multiple low-dimensional problems. Current studies [51]–[54] demonstrate that Cut-HDMR combined with surrogates exhibits excellent approximation ability for high-dimensional problems. However, the hierarchical structure of Cut-HDMR could limit its applicability for optimization of high-dimensional expensive problems because Cut-HDMR requires the sample points to be distributed in a regular way, thereby conflicting with the irregularly distributed sample points in the global optimization process. To overcome this shortcoming, some Cut-HDMR-based global optimization algorithms [55]–[57] have been proposed to improve the optimization efficiency of high-dimensional expensive problems.

To further improve the efficiency of optimizing high-dimensional expensive problems, an efficient generalized surrogate-assisted evolutionary algorithm is proposed. To validate the performance of the proposed algorithm, the algorithm is tested on several problems with dimensions varying from 30 to 100. The results show that the proposed algorithm is very promising for optimizing high-dimensional expensive problems whose dimensions are greater than 30.

The remainder of this paper is organized as follows. Section II presents the background theories involved in the proposed algorithm. Section III presents the proposed efficient generalized surrogate-assisted evolutionary algorithm. Section IV presents the discussions and experimental results and Section V concludes this paper.

II. Background

A. Radial Basis Function

In this paper, an RBF surrogate [11] is used in the optimization process of the proposed algorithm. Some studies [14], [58], [59] demonstrate that an RBF can usually obtain a more accurate approximation for high-dimensional problems than other common surrogates, including PRS, Kriging, and SVR. Another merit of an RBF is that its metamodeling speed is very fast when compared with Kriging. The RBF surrogate is defined as follows.

Given n distinct points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in R^D$ and the function values $f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_n)$, the interpolating form of the cubic RBF can be expressed as follows:

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \lambda_i \Phi_i(\|\boldsymbol{x} - \boldsymbol{x}_i\|) + p(\boldsymbol{x}) \tag{1}$$

where $\Phi_i(\cdot)$ is the ith basis function, $\|\cdot\|$ is the Euclidean norm and λ_i denotes the weight of the ith basis function. In this paper, the cubic function is used as the basis function: $\Phi(r) = r^3$. Other possible choices of the basis functions include the thin plate spline, multiquadric, and Gaussian forms. $p(\boldsymbol{x})$ is a linear polynomial function $\boldsymbol{b}^T\boldsymbol{x} + a$.

The unknown parameters ($\lambda_1, \lambda_2, \ldots, \lambda_n \in R^D$, $\boldsymbol{b} \in R^D$, $a \in R$) of the RBF can be obtained as the solution of the following linear equations:

$$\begin{pmatrix} \boldsymbol{\Phi} & \boldsymbol{P} \\ \boldsymbol{P^T} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{c} \end{pmatrix} = \begin{pmatrix} \boldsymbol{F} \\ \boldsymbol{0} \end{pmatrix} \tag{2}$$

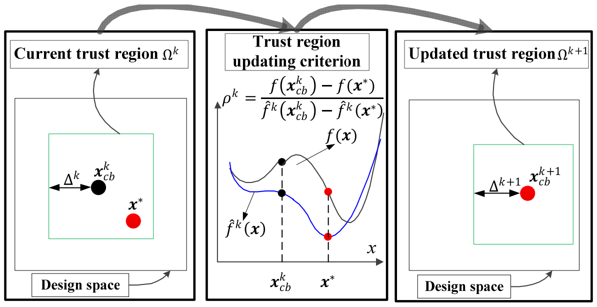
Current trust region Ω^k
Trust region updating criterion
Updated trust region Ω^{k+1}
$\rho^k = \frac{f(x_{cb}^k) - f(x^*)}{\hat{f}^k(x_{cb}^k) - \hat{f}^k(x^*)}$
Δ^k
x_{cb}^k
x^*
$f(x)$
$\hat{f}^k(x)$
x
Δ^{k+1}
x_{cb}^{k+1}
Design space
Design space

Fig. 2. Search process of trust region method based on surrogates.

where Φ is an $n \times n$ matrix with $\Phi_{ij} = \Phi(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|)$ and

$$\boldsymbol{P} = \begin{pmatrix} \boldsymbol{x}_1{}^T & 1 \\ \boldsymbol{x}_2{}^T & 1 \\ \vdots & \vdots \\ \boldsymbol{x}_n{}^T & 1 \end{pmatrix}, \boldsymbol{\lambda} = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{pmatrix}, \boldsymbol{c} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_d \\ a \end{pmatrix}, \boldsymbol{F} = \begin{pmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \vdots \\ f(\boldsymbol{x}_n) \end{pmatrix}.$$

If rank($\boldsymbol{P}$) $= D + 1$, the matrix $\begin{pmatrix} \boldsymbol{\Phi} & \boldsymbol{P} \\ \boldsymbol{P^T} & \boldsymbol{0} \end{pmatrix}$ is nonsingular and system (2) has a unique solution [60]. Thus, a unique RBF model is obtained.

B. Trust Region Method Based on Surrogates

The trust region method [61]–[64] is a local search method. As this method exhibits good local search ability and guarantees convergence, the method is widely used in engineering practice. The optimization of the trust region method based on surrogates (STR) can be expressed as

$$\begin{aligned} &\text{Minimize: } \hat{f}^k\left(\boldsymbol{x} + \boldsymbol{x}_{cb}^k\right) \\ &\text{Subject to: } \boldsymbol{x} \in \left[\boldsymbol{x}_{cb}^k - \Delta^k, \boldsymbol{x}_{cb}^k + \Delta^k\right] \end{aligned} \tag{3}$$

where $k = 0, 1, 2, \ldots, k_{\max}$ is the iteration of the trust region method, $\boldsymbol{x}_{cb}^k$ is the current best population point, and Δ^k is the radius of the trust region Ω^k in the kth iteration. The STR search process is depicted in Fig. 2, where $\boldsymbol{x}^*$ is the predicted optimum of $\hat{f}^k(\boldsymbol{x} + \boldsymbol{x}_{cb}^k)$ and ρ^k is the trust ratio, which is used to update the radius of trust regions. According to [47] and [65], the trust region in the optimization process can be updated as follows:

$$\Delta^{k+1} = \begin{cases} 0.25\Delta^k & \text{if } \rho^k \le 0.25 \\ \Delta^k & \text{if } 0.25 \le \rho^k \le 0.75 \\ \xi\Delta^k & \text{if } \rho^k \ge 0.75 \end{cases} \tag{4}$$

where $\xi = 2$, if $\|\boldsymbol{x}^* - \boldsymbol{x}_{cb}^k\| = \Delta^k$; $\xi = 1$, if $\|\boldsymbol{x}^* - \boldsymbol{x}_{cb}^k\| < \Delta^k$. In the iteration process of the trust region method, the center point $\boldsymbol{x}_{cb}^k$ is updated as: if $\rho^k \le 0$, $\boldsymbol{x}_{cb}^{k+1} = \boldsymbol{x}_{cb}^k$; otherwise, $\boldsymbol{x}_{cb}^{k+1} = \boldsymbol{x}^*$.

C. Prescreening Strategy Based on Kriging

The prescreening strategy based on Kriging can be seen as a general method to accelerate the search speed of basic evolutionary algorithms. This strategy is widely studied in surrogate-assisted evolutionary computation. Kriging [66] is

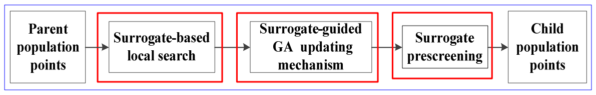

Fig. 3. Framework of the GSGA.

an interpolative Bayesian metamodeling technique. It estimates unknown responses through a combination of a known function $f_i(\boldsymbol{x})$ (a linear model, such as a polynomial trend) plus a stochastic process $Z(\boldsymbol{x})$ with zero mean

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{D} \beta_i f_i(\boldsymbol{x}) + Z(\boldsymbol{x}). \tag{5}$$

The coefficients β_i $(i = 1 \cdots D)$ are regression parameters. The random function $Z(\boldsymbol{x})$ can be achieved by a stochastic process $Z(\boldsymbol{x})$ with mean zero and process variance σ^2. The spatial covariance function is given as

$$\mathrm{Cov}(Z(\boldsymbol{v}), Z(\boldsymbol{w})) = \sigma^2 R(\boldsymbol{v}, \boldsymbol{w}, \boldsymbol{\theta}) \tag{6}$$

where $R(\boldsymbol{v}, \boldsymbol{w})$ is the correlation function between two points $\boldsymbol{v}$ and $\boldsymbol{w}$. The common correlation functions include the Gaussian correlation function, exponential correlation function, and so on. $\boldsymbol{\theta}$ is known as the correlation parameter. Its value reflects the importance of different variables. To obtain the Kriging metamodel, the correlation parameters $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_D]$ can be calculated by optimizing the maximum likelihood estimation function (MLS). The quality of the solutions of $\boldsymbol{\theta}$ has a great impact on the accuracy of the Kriging surrogate. According to previous studies, $\boldsymbol{\theta}$ can be optimized by the MLS function $K(\boldsymbol{\theta})$ as

$$\max_{\theta} \ K(\boldsymbol{\theta}) = \psi(\boldsymbol{R}, D, \sigma) \tag{7}$$

where $\psi(\cdot)$ is a transformed MLS function, and $\boldsymbol{R}$ is the correlation matrix.

The main idea of surrogate prescreening is to use the surrogates to select promising candidate child/offspring population points (individuals in the population) for exact function evaluations, thus saving the time cost of evaluating bad candidate child population points. The efficiency of traditional evolutionary algorithms is improved. Additional details on surrogate-assisted evolutionary algorithms assisted by surrogate prescreening can be referred to in [25] and [26].

III. Proposed Efficient Generalized Surrogate-Assisted Evolutionary Algorithm

In this section, a new generalized surrogate-assisted evolutionary algorithm is proposed. GA, a typical evolutionary algorithm, is selected as our research base. Therefore, our algorithm is called GSGA. The generalized optimization framework of the GSGA is plotted in Fig. 3. The main contribution of the GSGA is to propose and hybridly utilize three strategies, including a surrogate-based local search strategy, an SGA updating mechanism, and a surrogate prescreening strategy, within the optimization framework. The first local search strategy aims to obtain an accurate optimum and thus guide the GSGA to search rapidly. The second SGA updating mechanism aims to generate competitive child population

Algorithm 1 Pseudocode of the Initialization in the GSGA

```
1. NFE=0.
% NFE is the number of function evaluations
2. Generate $N$ parent population points $\mathbf{P} = [\boldsymbol{x}_1; \boldsymbol{x}_2; \cdots \boldsymbol{x}_N]$ by
   Latin hypercube sampling method. Evaluate all points (individuals) in the population using the function evaluations $f(\cdot)$,
   $\mathrm{NFE} = \mathrm{NFE} + N$.
3. $N_1 = round(N \times 0.9)$.
% $N_1$ means the number of individuals that are created for crossover
4. If $mod(N_1, 2) \neq 0$
       $N_1 = N_1 - 1$.
   End If.
5. $N_2 = round(N * 0.1)$.
% $N_2$ is the number of individuals that mutate
6. $N_3 = N - N_s$.
% $N_3$ denotes the number of survived individuals. They represent the individuals with the smallest responses (fitness values)
```

points (individuals in the population), thereby accelerating the GSGA optimization process. The third prescreening strategy aims to select the more promising child population points and further improve the optimization efficiency of the GSGA. The first strategy is used for local search. The other two strategies are used for global search. The hybrid use of these strategies could make the GSGA very efficient for global optimization. In summary, the GSGA sufficiently utilizes information provided by surrogates in the optimization process. This algorithm could be very suitable for the optimization of high-dimensional expensive problems.

First, Algorithm 1 is provided below to show the pseudocode of the initialization in the GSGA optimization. In the following pseudocodes of the GSGA in this section, some functions, including round($\cdot$), mod($\cdot$), sum($\cdot$), cumsum($\cdot$), find($\cdot$), and randi($\cdot$) are cited from MATLAB. The use of these functions can be referred to in the explanation files of MATLAB.

A. Surrogate-Based Local Search

As the optimization process continues, the global optimum can be found in a small region. To obtain a more accurate estimate of the global optimum, the trust region local search method based on an RBF surrogate is used. This method is only used around the best population point with the smallest response in the GSGA. The best population point will be updated by the optimum found through the STR local search. The idea of using STR local search in the GSGA actually refers to the previous research [33], [42], [43]. The research indicated that the optima provided by surrogates can effectively accelerate the convergence of evolutionary or metaheuristic algorithms. Therefore, it is reasonable to use the STR local search in the GSGA to improve the optimization efficiency of the GA.

In the STR local search in the GSGA, the local RBF surrogate $\hat{f}_L(\boldsymbol{x}) = \sum_{i=1}^{n_L} \lambda_i \Phi_i(\|\boldsymbol{x} - \boldsymbol{x}_i\|) + p(\boldsymbol{x})$ is built by using the evaluated points in the trust region $[\boldsymbol{x}_{\text{best}} - \Delta^k, \boldsymbol{x}_{\text{best}} + \Delta^k] \cap [\text{lb}, \text{ub}]$ around the best population point x_{best}. n_L is the number of evaluated points used for training the RBF surrogate. The initial radius Δ^0 of the trust region is set as the half distance between the maximum response point and minimum response point in the initial trust region made of $5 \times D$ evaluated points

Algorithm 2 Pseudocode of the STR in the GSGA

```
1. Sort the population points in an ascending order according to their responses. The population is now $\boldsymbol{P} = [\boldsymbol{x}_1; \boldsymbol{x}_2; \cdots \boldsymbol{x}_N]$. The response $f(\boldsymbol{x}_1)$ of individual $\boldsymbol{x}_1$ is the smallest. The response $f(\boldsymbol{x}_N)$ of individual $\boldsymbol{x}_N$ is the largest.
% Run the STR local search for the current best population point $\boldsymbol{x}_1$ and update this point with the improved point
2. While $k < k_{\max}$ do
% $k$ is the iteration variable of the STR local search, and its initial value is $k = 0$; $k_{\max}$ is the maximum number of iterations of the STR local search
3.     Build the local RBF surrogate $\hat{f}_L(\boldsymbol{x})$ by using the evaluated points $[\boldsymbol{X}_L, f(\boldsymbol{X}_L)]$ in the trust region $[\boldsymbol{x}_1 - \Delta^k, \boldsymbol{x}_1 + \Delta^k] \cap [\mathbf{lb}, \mathbf{ub}]$ around the current best population point $\boldsymbol{x}_1$;
4.     Obtain predicted point $\boldsymbol{x}_1^*$ by minimizing $\hat{f}_L(\boldsymbol{x})$ in the trust region $[\boldsymbol{x}_1 - \Delta^k, \boldsymbol{x}_1 + \Delta^k] \cap [\mathbf{lb}, \mathbf{ub}]$. Evaluate $\boldsymbol{x}_1^*$ by using $f(\cdot)$, NFE=NFE+1;
5.     If $f(\boldsymbol{x}_1^*) < f(\boldsymbol{x}_1)$
6.         update the best population point, $\boldsymbol{x}_1 = \boldsymbol{x}_1^*$;
7.     End If
% Trust region updating
8.     Calculate the trust ratio $\rho^k = (f(\boldsymbol{x}_1) - f(\boldsymbol{x}_1^*))/(\hat{f}_L(\boldsymbol{x}_1) - \hat{f}_L(\boldsymbol{x}_1^*))$;
9.     Obtain the radius $\Delta^{k+1}$ of the trust region in the $k$+1-*th* iteration according to equation (4). $k = k + 1$;
10. End While
```

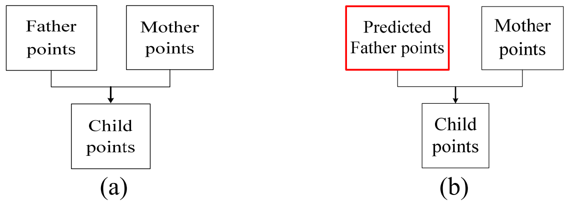

Fig. 4. Crossover difference between the (a) GA and (b) SGA.

around $\boldsymbol{x}_{\text{best}}$. Moreover, to obtain an accurate RBF surrogate in the STR local search, the number of evaluated training points should be set larger than a certain number, which is set to be $5 \times D$ in this paper. If this number does not satisfy this condition, evaluated points surrounding the trust region should be added until it reaches $5 \times D$ or until all evaluated points have been added.

After initialization, the optimization of the GSGA will go to the main loop. Algorithm 2 is provided below to show the pseudocode of the STR local search in the GSGA.

B. Surrogate-Guided GA Updating Mechanism

In this section, an RBF SGA updating mechanism is proposed to produce the competitive candidate child population points. The main procedures of a GA can be divided into three steps: 1) selection; 2) crossover; and 3) mutation. Here, the proposed SGA method only changes the crossover operation of the GA. The other procedures of the proposed method are the same as those of the GA. It should be noted that Rasheed [67] also proposed a guided crossover operator. Instead of using surrogates, this crossover operator uses the population points of the GA to select a promising direction for exploration. The crossover difference between the GA and SGA is shown in Fig. 4. In the SGA crossover operation, the selected parent population points are first divided into father points and mother points. The father points $\boldsymbol{x}_{i,\text{father}}$, $i = 1, 2, \ldots, n$ (or mother points) are then replaced by the predicted points $x^*_{i,\text{father}}$ of surrogates in their corresponding neighbor region. Then, the crossover operation can be performed for the predicted father points and mother points. Whether the father points are replaced by the predicted father points is determined by

$$\boldsymbol{x}^*_{i,\text{father}} = \begin{cases} X_{\text{Nbest}}; & \text{if } \hat{f}_{Li}(X_{\text{Nbest}}) < f(\boldsymbol{x}_{i,\text{father}}) \\ \boldsymbol{x}_{i,\text{father}}; & \text{else} \end{cases}$$
$$X_{\text{Nbest}} = \operatorname{argmin}\left(\hat{f}_{Li}(\boldsymbol{x})\right) \tag{8}$$

where $\hat{f}_{Li}(\boldsymbol{x})$ is the predicted response function of the local surrogate built by all the evaluated points in the neighbor region of $\boldsymbol{x}_{i,\text{father}}$ and X_{Nbest} is the predicted minimum of $\hat{f}_{Li}(\boldsymbol{x})$ in the neighbor region.

The main idea behind the improved crossover operation in the SGA is that surrogates provide an optimization direction. This crossover operation can lead the population points to search in the regions with lower responses. In addition, the crossover operation of the GA also guides the population points to search in a global way. Therefore, the SGA mechanism sufficiently utilizes the predictive ability of surrogates and the global search ability of the GA, enabling the GSGA to search efficiently for expensive high-dimensional optimization problems within limited sample points.

In the SGA updating mechanism, it is important to determine the size of the neighbor region of $x_{i,\text{father}}$. If the size of this region is too large, the predicted father points could be duplicated by other population points. Therefore, the diversity of population points in the SGA will be reduced, making the SGA search locally. However, if the size of the neighbor region is too small, the optimum information provided by surrogates will not be utilized sufficiently, and there will be no significant improvement in the efficiency of the GA. An algorithm that was proposed by Kitayama *et al.* [68] to determine the widths of RBF is introduced to obtain a suitable size for the neighbor region. The radius r_i of the neighbor region around father point $\boldsymbol{x}_{i,\text{father}}$ is determined by

$$r_i = \alpha \times \frac{D_{i,\max}}{\sqrt{D}\sqrt[D]{N-1}}, i = 1, 2, \ldots, n \tag{9}$$

where $D_{i,\max}$ denotes the maximum distance between the ith father point and the other population points, D is the dimension of the design space, N is the population size, n is the number of father points, and α is an important parameter used to control the size of the neighbor region. Then, the neighbor region of $x_{i,\text{father}}$ can be defined as $[\boldsymbol{x}_{i,\text{father}} - r_i, \boldsymbol{x}_{i,\text{father}} + r_i] \cap [\mathbf{lb}, \mathbf{ub}]$, where $\mathbf{lb}$ and $\mathbf{ub}$ denote the lower bound and upper bound of the design space, respectively. The minimum predicted response point X_{Nbest} can be obtained by using the global optimization algorithm to minimize $\hat{f}_{Li}(x)$ in the neighbor region.

In this paper, the default value of parameter α is set as 0.5, which is suitable to maintain the population's diversity by making the predicted father points not duplicated by other population points and to greatly utilize the surrogates' optimum information. To validate this viewpoint, it is first given that the

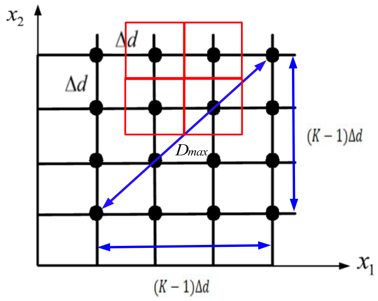

Fig. 5. Distribution of the K-level full factorial design points in the 2-D design space.

sample points (population points) are uniformly distributed. The radius of each sample point can be expressed as

$$r = 0.5\frac{D_{\max}}{\sqrt{D}\sqrt[D]{N}} \tag{10}$$

where $D_{\max}$ denotes the maximum distance between any two different sample points.

Next, let us consider a K-level full factorial design in an ideal 2-D design space with the same bounds for each variable. The distribution of the sample points (black dots) is depicted in Fig. 5, in which Δd is the regular interval between any two points, and $D_{\max}$ can be calculated as

$$D_{\max} = \sqrt{D}(K-1)\Delta d, \quad \text{where} \quad N = K^D. \tag{11}$$

Then, we can obtain the following:

$$\frac{r}{\Delta d} = 0.5\frac{\sqrt{D}(K-1)\Delta d}{\sqrt{D}\sqrt[D]{N}\Delta d} = 0.5\frac{K-1}{K} = 0.5 - \frac{0.5}{K}. \tag{12}$$

We can see from (12) that, with an increase in sampling points ($K \to \infty$), the ratio of ($r/\Delta d$) will be close to 0.5. Therefore, the neighbor regions belonging to the population points in the SGA are not duplicated, as indicated by the red squares in Fig. 5. The predicted father point corresponds to a population point. Obtaining the predicted father points is equal to obtaining the predicted population points in the neighbor regions of red squares. The predicted father or population points will not be duplicated by other population points. Therefore, $\alpha = 0.5$ is a suitable setting to obtain the predicted father points.

From the proposed SGA updating mechanism, it can be observed that the combination of surrogates and the GA in the GSGA is different from that in other surrogate-assisted metaheuristic algorithms. In the GSGA, the SGA combines surrogates and the evolutionary algorithms themselves in a tight way rather than in a separate way. The predicted optima provided by the local surrogates can direct the GA to search accurately and quickly. Moreover, the neighbor region partition strategy enables the GSGA to make full use of the space information. The predicted optima found in the neighbor regions can be used to make the GA search in a greedier way. Additionally, this strategy can avoid the obtained optima from being duplicated and the diversity of population points from being decreased. The global search ability of GSGA is thus maintained. Moreover, the GSGA is robust because of the multiple surrogates used in its optimization process. This algorithm will still be useful if some surrogates are badly built. In summary, the SGA updating mechanism is a special characteristic of GSGA. This mechanism fuses surrogates with evolutionary algorithms in the evolutionary optimization process.

Algorithm 3 Pseudocode of the SGA in the GSGA

```
% Obtain the predicted population points
1.  For $i = 1$ to $N$ do
2.     Build the local RBF surrogate $\hat{f}_{Li}(\boldsymbol{x})$ by using the evaluated points in the region $[\boldsymbol{x}_1 - r_i, \boldsymbol{x}_1 + r_i] \cap [\mathbf{lb}, \mathbf{ub}]$ around the population point $\boldsymbol{x}_i$. $r_i = 0.5 \times \frac{D_{i,max}}{\sqrt{D}\sqrt[D]{N-1}}$, $D$ is the problem's dimension, and $D_{i,max}$ is the maximum distance between $\boldsymbol{x}_i$ and the other population points. The number of training points is set to be larger than $5 \times D$ to build an accurate RBF surrogate. If this condition is not satisfied, the evaluated points surrounding $\boldsymbol{x}_i$ will be utilized.
3.     Obtain the best neighbor point $X_{Nbest}$ of $\boldsymbol{x}_i$ by minimizing $\hat{f}_{Li}(x)$ in the local region $[\boldsymbol{x}_1 - r_i, \boldsymbol{x}_1 + r_i] \cap [\mathbf{lb}, \mathbf{ub}]$. The predicted population point $\hat{\boldsymbol{x}}_i = X_{Nbest}$;
4.  End For
% GA operators
%% Rank selection
5.  $\boldsymbol{pdf} = (N + 1 - [1 : N]^T)/(N \times (N + 1)/2)$;
6.  $\boldsymbol{pdf} = \boldsymbol{pdf}/sum(\boldsymbol{pdf})$;
7.  $\boldsymbol{cdf} = cumsum(\boldsymbol{pdf})$;
% The population points and predicted population points after selection are set as PS = [] and PPS = [], respectively.
8.  For $i = 1$ to $N_1$ do
9.     Index $= find(\text{rand} \le \boldsymbol{cdf}, 1, \text{'first'})$;
10.    $\mathbf{PS} = [\mathbf{PS}; \boldsymbol{x}_{\text{Index}}]$, $\mathbf{PPS} = [\mathbf{PPS}; \hat{\boldsymbol{x}}_{\text{Index}}]$;
11. End For
%% Uniform crossover
% Divide PS into father points FP = [] and mother points MP = []; Divide PPS into predicted father points PFP = [] and predicted mother points PMP = []
12. For $i = 1 : 2 : N_1$ do
13.    $\mathbf{FP} = [\mathbf{FP}; \mathbf{PS(i, :)}]$, $\mathbf{MP} = [\mathbf{MP}; \mathbf{PS(i+1, :)}]$;
14.    $\mathbf{PFP} = [\mathbf{FP}; \mathbf{PPS(i, :)}]$, $\mathbf{PMP} = [\mathbf{PMP}; \mathbf{PPS(i+1, :)}]$;
15. End For
% The child population points after crossover are defined as Child = [], Child1 and Child2
16. For $i = 1$ to $N_1/2$ do
17.    For $j = \mathbf{1}$ to $\boldsymbol{D}$ do
18.       If rand $\le P_{cross}$
% rand is a random number between 0 and 1; $P_{cross}$ is the crossover probability
19.          $\mathbf{Child1}(i, j) = \mathbf{PFP}(i, j)$;
20.          $\mathbf{Child2}(i, j) = \mathbf{MP}(i, j)$;
21.       Else
22.          $\mathbf{Child1}(i, j) = \mathbf{MP}(i, j)$;
23.          $\mathbf{Child2}(i, j) = \mathbf{PF}\boldsymbol{P}(i, j)$;
24.       End If
25.    End For
26. End For
27. $\mathbf{Child} = [\mathbf{Child}; \mathbf{Child1}; \mathbf{Child2}]$;
%% Mutation
28. For $i = 1$ to $N_2$ do
29.    $j = randi([1\ \text{N}_1])$;
30.    For $k = \mathbf{1}$ to D do
31.       If rand $\le P_{mut}$
32.          $\mathbf{Child}(j, k) = \mathbf{lb(1)} + (\mathbf{ub(1)} - \mathbf{lb(1)}) \times \text{rand}$
% It is assumed that the lower and upper bounds are the same for all variables; $P_{mut}$ is the mutation probability
33.       End If
34.    End For
35. End For
```

Algorithm 3 is provided below to show the pseudocode of SGA in the GSGA. The implementation of the code in lines 1–4 is lightly different from the above description in part B, which aims to obtain the predicted father points. The code is used to obtain the predicted population points. However, the same effect is achieved because one predicted father point corresponds to one population point. Obtaining the predicted father points is thus equivalent to obtaining the predicted population points. The reason for such coding is that it is easy for us to realize GSGA programming.

C. Prescreening Strategy Based on Simplified Kriging

In this section, a simplified Kriging is proposed for prescreening the promising candidate child population points and improving the optimization efficiency of the GSGA. This approach can ease the computational burden of Kriging metamodeling for high-dimensional problems. The prescreening strategy based on Kriging has proven to be an efficient way to accelerate the convergence of an evolutionary algorithm in previous studies. However, a major disadvantage of Kriging is that its metamodeling time increases exponentially with increasing problem dimensions, making it unsuitable for the optimization of high-dimensional problems. To address this problem, a simplified Kriging is proposed for approximating high-dimensional problems. Through investigation and experiments, it can be found that the very high computational cost of Kriging metamodeling for high-dimensional problems is incurred by obtaining the optimal correlation parameters by maximizing the likelihood function $K(\theta_1, \theta_2, \ldots, \theta_D)$. Specifically, the computation of the correlation matrix is very time consuming in the process of calculating the response of $K(\boldsymbol{\theta})$. For high-dimensional problems, obtaining an optimal $\boldsymbol{\theta}$ will need many evaluations of $K(\boldsymbol{\theta})$, thus making Kriging metamodeling extremely time consuming. To reduce the time cost of Kriging metamodeling, we consider that every correlation parameter has the same weight. The optimization problem then becomes a 1-D optimization problem of maximizing $K(\theta_1, \theta_2, \ldots, \theta_D)$, $\theta_1 = \theta_2 = \cdots = \theta_D$. Therefore, only a very limited number of evaluations of $K(\boldsymbol{\theta})$ are required to obtain a relatively optimal $\boldsymbol{\theta}$ and an available Kriging surrogate is thus built. The simplified Kriging may not be that accurate. However, surrogate-assisted evolutionary algorithms usually do not have a high accuracy requirement for Kriging. These algorithms only require that Kriging can rank candidate points by using the metric values of the Kriging infilling criterion. Thus, the prescreening strategy based on simplified Kriging will be suitable for optimizing high-dimensional problems. In the optimization of $K(\theta_1, \theta_2, \ldots, \theta_D), \theta_1 = \theta_2 = \cdots = \theta_D$, the dividing rectangle (DIRECT) optimization algorithm [69] is used, as it is an efficient global optimizer for low-dimensional problems. This idea of tuning correlation parameters with the same weight has been discussed by Toal *et al.* [70]. The difference is that they used a GA to optimize $K(\theta_1, \theta_2, \ldots, \theta_D), \theta_1 = \theta_2 = \cdots = \theta_D$. The results of their research show that simplified Kriging may be a good choice for the optimization of high-dimensional problems. A more accurate correlation parameters' tuning strategy can be found in a study by Welch *et al.* [71]. Moreover, the Kriging with partial least squares, recently proposed by Bouhlel *et al.* [72], can also be used to ease the computational burden of Kriging metamodeling for high-dimensional problems. In this paper, the well-known EI infilling criterion proposed by Jones *et al.* [73] is used for prescreening candidate child population points. This criterion can be expressed as $\mathrm{EI}(\boldsymbol{x}) = \begin{cases} \left(f_{\min} - \hat{f}(\boldsymbol{x})\right)\Phi\left(\frac{f_{\min}-\hat{f}(\boldsymbol{x})}{\hat{s}(\boldsymbol{x})}\right) + \hat{s}(\boldsymbol{x})\phi\left(\frac{f_{\min}-\hat{f}(\boldsymbol{x})}{\hat{s}(\boldsymbol{x})}\right), & \text{if } \hat{s}(\boldsymbol{x}) > 0 \\ 0, & \text{if } \hat{s}(\boldsymbol{x}) = 0 \end{cases}$ where $\hat{s}(\boldsymbol{x})$ is the predicted error, $f_{\min}$ is the current minimum response, and $\hat{f}(\boldsymbol{x})$ is the predicted response.

Algorithm 4 Pseudocode of the EI Prescreening in the GSGA

```
1. Build the simplified Kriging surrogate by using all evaluated
   points and the DACE toolbox, and build the EI function $EI(\cdot)$;
2. Calculate the EI values **EIChild** of the child population points
   **Child**;
3. **For** $i = 1$ to $N_1$ **do**
4.     **EIChild**$(i) = EI($**Child**$(i, :))$;
5. **End For**
6. Sort the child population points **Child** in descending order
   according to their EI values. The points with high EI values
   in **Child** will be selected for exact function evaluations;
7. **NEWChild** = **Child**$[1 : N_s, :]$, Evaluate all points in
   **NEWChild** using the function evaluations $f(\cdot)$, NFE=NFE+$N_s$.
   If a point in **NEWChild** is equal to one of the evaluated points,
   it will not be evaluated by $f(\cdot)$;
% Formulate the new population points
8. **P1** = **P**$[1: N_3, :]$, **P** = [], **P** = [**P1**; **NEWChild**];
```

In the prescreening process, the N_s candidate child population points with larger EI values will be selected for exact function evaluations because the points with larger EI values are considered more promising for the global optimum according to the EI criterion.

Algorithm 4 is provided below to show the pseudocode of the EI prescreening in the GSGA. Stopping the GSGA optimization is determined by whether the allowed computational budget is exhausted. This factor can be represented by the maximum number of function evaluations (MaxNFE) in the numerical study. In this paper, the main loop of the GSGA optimization including the implementation of Algorithms 2–4 is continued until NFE is larger than MaxNFE.

D. Parameter Settings of the Proposed GSGA

The basic parameters of the GSGA are set as presented in Table I. A cubic RBF is used in our research. In the STR local search, the maximum number of iterations $k_{\max}$ is set to 3 according to the research performed by Ong *et al.* [44] and Lim *et al.* [47]. To obtain an accurate global optimum, the sequential quadratic programing solver in MATLAB 2011a is used with ten different start points to obtain the predicted optima of surrogates in the STR local search. In the SGA updating mechanism, the PSO with a constriction factor (CPSO) algorithm [74] is used to obtain the predicted optima of local surrogates. In the optimization of the correlation parameters θ in the simplified Kriging, the DIRECT optimization algorithm is used to optimize the maximum likelihood function. The MaxNFE is set to 1000. The simplified

TABLE I
PARAMETER SETTINGS OF THE GSGA

Population size (N)	50
Crossover probability (P_{cross})	0.9
Mutation probability (P_{mut})	0.1
Maximum number of function evaluations (MaxNFE)	1000
Maximum number of iterations in the trust region local search (k_{max})	3
Number of selected child population points (N_s)	10
Evolutionary operators	Uniform crossover, mutation and rank selection

TABLE II
CHARACTERISTICS OF THE BENCHMARK PROBLEMS

Fun.	Name	D	Design space	Global optimum	Property
F1	Ellipsoid	30 50 100	$[-5.12, 5.12]^D$	0	Unimodal
F2	Rosenbrock	30 50 100	$[-2.048, 2.048]^D$	0	Multimodal with narrow valley
F3	Ackley	30 50 100	$[-32.768, 32.768]^D$	0	Multimodal
F4	Griewank	30 50 100	$[-600, 600]^D$	0	Multimodal
F5	Shifted Rotated Rastrigin	30 50 100	$[-5, 5]^D$	-330	Very complicated multimodal
F6	Rotated Hybrid Composition Function	30 50 100	$[-5, 5]^D$	120	Very complicated multimodal
F7	Rotated Hybrid Composition Function with a narrow basin for the global optimum	30 50 100	$[-5, 5]^D$	10	Very complicated multimodal

Kriging is improved based on the DACE toolbox in 2002 [66]. In the prescreening strategy, $N_s = 10$ candidate child population points are selected for exact function evaluations. The codes of the GSGA are programmed in MATLAB 2011a.

IV. Discussion and Experimental Study

To evaluate the performance of the proposed algorithm, several widely used unimodal and multimodal benchmark problems are adopted. The dimension (D) of these problems varies from 30 to 100. These problems have been widely tested by many researchers, including Lim *et al.* in 2010 [47], Liu *et al.* in 2014 [25], Sun *et al.* in 2017 [37], and Yu *et al.* in 2018 [43]. Problems 5–7 are cited from CEC 2005 [75]. These problems were also tested by some of the researchers indicated above. The characteristics of these problems are summarized in Table II. To obtain a robust optimization result, 20 runs are carried out on the proposed GSGA and the GA variants for each problem. The proposed algorithm is run on a PC with an Intel Core i5-4570 CPU @ 3.20 GHz and 8 GB RAM. It should be noted that all convergence curves in this paper display the mean optimization results.

A. Discussion of the Proposed Algorithm

In this section, we discuss the proposed algorithm in detail through experimental studies. First, to demonstrate the effects of the proposed strategies on the proposed optimization algorithm, we conduct a comprehensive study of the variants of the proposed algorithm. These variants include the GA plus the surrogate-based trust region local search strategy (GA+s1), GA plus SGA mechanism strategy (GA+s2), GA plus Kriging-based EI prescreening strategy (GA+s3), GA plus hybrid strategies of s1, s2, and s3 (GA+s1+s2, GA+s1+s3 and GA+s2+s3), and the proposed GSGA. These algorithms are tested on the 50-D Ackley function, and their convergence curves are plotted in Fig. 6. It can be observed that different strategies work in the proposed optimization framework. However, strategies s1 and s2 play a more important role than s3 in the optimization process and this phenomenon is explainable. Strategies s1 and s2 use the optima provided by surrogates. Therefore, these strategies provide the search direction for the GA. If the optima provided by surrogates are accurate, these strategies will greatly accelerate the search speed of GA. Strategy s3 uses the EI criterion of Kriging to prescreen promising candidate child population points, and this strategy does not fundamentally change the search mechanism of the GA. This strategy only improves the quality of child population points in the GA and saves some improper evaluations of bad child population points. Therefore, strategy s3 does not have a clear performance improvement for the GA. In addition, we can take a closer look at the effects of the hybrid proposed strategies by comparing GA+s1+s2 and GA+s1+s3 and GA+s2+s3. It can be observed that different hybrid strategies mutually influence the proposed algorithm. The combination of these strategies has a more positive effect on improving the optimization efficiency of the GA than the standalone strategy. The combination of the three strategies can improve the GA's optimization efficiency the most. This finding validates the rationality of the hybrid use of different strategies in the proposed optimization framework. Strategy s1 is used to improve the optimization accuracy. Strategies s2 and s3 are used to improve the global optimization efficiency. Strategy s2 is used to produce competitive candidate child population points and strategy s3 is used to select more competitive child population points for exact function evaluations. The hybrid use of these strategies thus makes the proposed algorithm have a high optimization efficiency and accuracy.

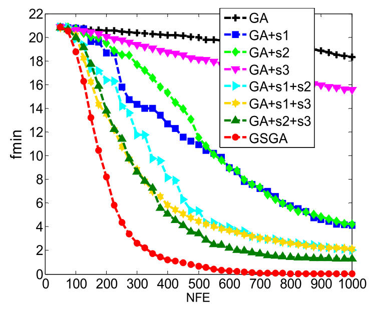

Fig. 6. Convergence curves of the proposed algorithm and its variants on the 50-D Ackley function (F3).

In addition, the simplified Kriging surrogate is checked through 2-D, 50-D, and 100-D Rosenbrock and Ackley functions to demonstrate its suitability for high-dimensional problems. Two metrics, including metamodeling time Time (s) and

TABLE III
METAMODELING TIME AND ACCURACY OF DIFFERENT KRIGING MODELS

Fun	Dimension	2D				50D				100D			
	Sample size	10		50		100		500		200		500	
	Metrics	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time
Rosenbrock	$\text{Kriging}_{\text{DACE}}$	459.6841	**0.0066**	**0.1105**	**0.0108**	6677.9853	1.9167	4854.5151	**38.6608**	9125.8184	22.7319	7336.5615	140.3592
	$\text{Kriging}_{\text{Ga5000}}$	**432.5930**	3.4426	5.6396	7.4668	6677.9853	122.7182	4854.5151	2063.8568	9125.8184	398.0359	7336.5615	2075.9298
	$\text{Kriging}_{\text{Direct}}$	466.6021	0.0339	0.4682	0.0920	**6677.1504**	**1.7539**	**2878.4089**	46.0165	**9121.5394**	**8.1425**	**6543.0403**	**89.1170**
Ackley	$\text{Kriging}_{\text{DACE}}$	1.8808	**0.0069**	1.1976	**0.0105**	0.2145	**1.6015**	0.1633	**36.8094**	0.150528	26.6172	0.1195	117.5162
	$\text{Kriging}_{\text{Ga5000}}$	1.8833	3.3760	1.1832	7.4199	0.2145	105.0835	0.1633	1706.5336	0.150528	462.0953	0.1195	1705.0723
	$\text{Kriging}_{\text{Direct}}$	**1.7795**	0.0363	**1.1496**	0.0733	0.2145	1.6643	**0.1191**	41.9006	**0.150466**	**11.2578**	**0.1095**	**48.4557**

accuracy RMSE, should be considered. The RMSE metric can be referred to in Jin *et al.* [14] and Le *et al.* [48]. A total of 5×10^4 randomly distributed test points are selected to calculate the RMSE values. Ten runs of Kriging metamodeling are carried out to obtain robust metric values. The sample size in metamodeling is set as 10 and 50 for 2-D problems, 100 and 500 for 50-D problems and 200 and 500 for 100-D problems. For comparison, two commonly used Kriging models, one that uses a GA [70] to optimize the correlation parameters θ and another that uses the DACE toolbox's Hooke and Jeeves optimization algorithm [76] to optimize parameters θ, are also tested. These models are labeled $\text{Kriging}_{\text{DACE}}$ and $\text{Kriging}_{\text{Ga5000}}$. The simplified Kriging that uses the DIRECT optimization algorithm is called $\text{Kriging}_{\text{Direct}}$. The mean testing results are reported in Table III. The improved Kriging models are all based on the Kriging in the DACE toolbox. The optimizing range of θ is set as [0.01 20]. The regression model is "@regpoly1." The correlation model is "@corrgauss." The MaxNFE in the GA is set to 5000. Table III shows that, for the 2-D Rosenbrock function, $\text{Kriging}_{\text{DACE}}$, and $\text{Kriging}_{\text{Ga5000}}$ perform more accurately than $\text{Kriging}_{\text{Direct}}$ with 20 sample points. $\text{Kriging}_{\text{DACE}}$ also performs more accurately than $\text{Kriging}_{\text{Direct}}$ with 50 sample points. The reason can be attributed to the fact that for low-dimensional problems, the GA or Hooke and Jeeves algorithm can obtain an accurate $\boldsymbol{\theta}$ within a limited number of evaluations of the likelihood function. However, for the 50-D and 100-D Rosenbrock functions, $\text{Kriging}_{\text{Direct}}$ clearly performs more accurately than $\text{Kriging}_{\text{DACE}}$ and $\text{Kriging}_{\text{Ga5000}}$. The accuracy performance of $\text{Kriging}_{\text{DACE}}$ and $\text{Kriging}_{\text{Ga5000}}$ is similar. This finding shows that the GA and Hooke and Jeeves optimization algorithms all cannot obtain an accurate $\boldsymbol{\theta}$ within limited function evaluations. In addition, for the 50-D and 100-D problems, the time cost of $\text{Kriging}_{\text{Direct}}$ is very limited. This model only needs 89 s to build a Kriging model by using 500 sample points, while $\text{Kriging}_{\text{Ga5000}}$ uses 2045 s (over 0.5 h). Compared with the Kriging metamodeling time cost reported in previous studies, such as Liu *et al.* (2014), the time cost of $\text{Kriging}_{\text{Direct}}$ metamodeling is also quite low. Moreover, $\text{Kriging}_{\text{Direct}}$ has another merit. This mode could be highly suitable for metamodeling of symmetrical problems. In the metamodeling process of $\text{Kriging}_{\text{Direct}}$, the same weight of different variables ($\theta_1 = \theta_2 = \cdots = \theta_D$) is expected. The Ackley function is symmetrical for different axes. The accuracy of $\text{Kriging}_{\text{Direct}}$ is always higher than that of $\text{Kriging}_{\text{DACE}}$ and $\text{Kriging}_{\text{Ga5000}}$. In summary, the simplified Kriging has good accuracy and fast metamodeling speed. This approach is very suitable for the optimization of high-dimensional expensive problems.

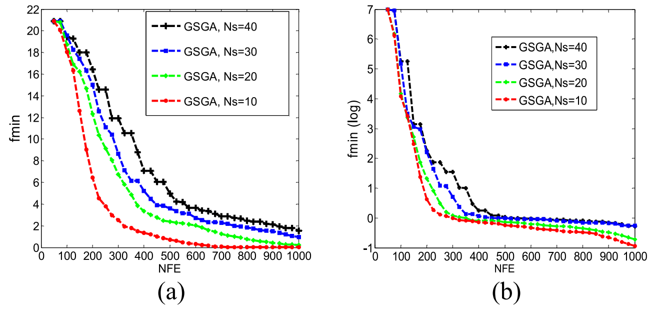

Fig. 7. Convergence curves of the GSGA with different N_s values on the (a) 50-D Ackley (F3) and (b) Griewank (F4) functions.

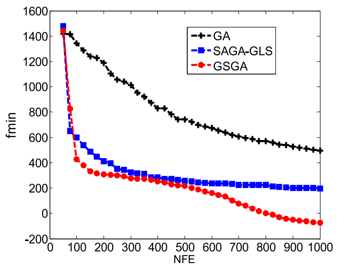

Fig. 8. Convergence curves of the GA, SAGA-GLS, and GSGA on the 50-D shifted rotated Rastrigin function (F5).

Moreover, the effect of parameter N_s in the prescreening strategy is also discussed. GSGAs with different N_s values are tested on the 50-D Ackley and Griewank functions. The N_s values are set to 10, 20, 30 and 40. The convergence curves of different GSGAs are plotted in Fig. 7. The GSGA's optimization efficiency gradually increases with decreasing N_s. This conclusion is similar to a study performed by Liu *et al.* [25]. The difference is that they use the LCB criterion to prescreen the candidate child population points. In summary, the EI infilling criterion provided by the simplified Kriging can effectively select promising candidate child population points, improving the optimization efficiency of the proposed GSGA. For convenience, $N_s = 10$ is used in our research, because a smaller N_s value will increase the number of iterations of the GSGA and thus increase the execution time of the GSGA.

In addition, a comparison is made between GSGA and the surrogate-assisted evolutionary algorithm (SAGA-GLS) in [24], because these two algorithms can be considered generalized surrogate-assisted evolutionary algorithms. Both algorithms use surrogate-based local search and surrogate prescreening strategies in their optimization processes. The

TABLE IV
OPTIMIZATION RESULTS FOR THE 30-D PROBLEMS

Fun.	Algorithm	Best	Worst	Mean	Std.	t-test
F1	GA	1.6682E+02	5.2733E+02	2.9262E+02	9.1593E+01	+
	GPEME	1.5500E-02	1.6470E-01	**7.6200E-02**	4.0100E-02	≈
	SHPSO	4.4782E-02	7.2024E-01	2.1199E-01	1.5229E-01	+
	GSGA	5.1545E-03	3.2678E-01	**7.2725E-02**	9.3597E-02	
F2	GA	3.0519E+02	8.1527E+02	5.5535E+02	1.3359E+02	+
	GS-SOMA	N/A	N/A	2981	N/A	N/A
	GPEME	2.624E+01	8.8235E+01	4.6177E+01	2.5520E+01	+
	SHPSO	2.7726E+01	2.9290E+01	2.8566E+01	4.0441E-01	+
	GSGA	2.5691E+01	2.9043E+01	**2.7598E+01**	1.2954E+00	
F3	GA	1.2487E+01	1.6339E+01	1.4699E+01	1.0262E+00	+
	GS-SOMA	N/A	N/A	20	N/A	N/A
	GPEME	1.9491E+00	4.9640E+00	3.0105E+00	9.2500E-01	+
	SHPSO	5.6091E-01	2.9574E+00	1.4418E+00	7.7404E-01	+
	GSGA	6.5059E-03	5.7251E-02	**2.3087E-02**	1.4142E-02	
F4	GA	4.5474E+01	1.1498E+02	7.1876E+01	1.8529E+01	+
	GS-SOMA	N/A	N/A	365	N/A	N/A
	GPEME	7.3680E-01	1.0761E+00	9.9690E-01	1.0800E-01	+
	SHPSO	7.0609E-01	1.0275E+00	9.2053E-01	8.8062E-02	+
	GSGA	9.5108E-02	3.8395E-01	**2.2280E-01**	7.8443E-02	
F5	GA	-4.1720E+01	1.7493E+02	4.6944E+01	5.8320E+01	+
	GS-SOMA	N/A	N/A	>50	N/A	N/A
	GPEME	-5.7068E+01	1.8033E+01	-2.1861E+01	3.6449E+01	+
	SHPSO	-1.3297E+02	-5.9993E+01	-9.2830E+01	2.2544E+01	+
	GSGA	-2.4520E+02	-1.5902E+02	**-2.0302E+02**	2.4871E+01	
F6	GA	4.5128E+02	7.5636E+02	5.9478E+02	7.8189E+01	+
	GS-SOMA	N/A	N/A	>665	N/A	N/A
	SHPSO	3.2715E+02	6.4948E+02	**4.6433E+02**	8.5125E+01	≈
	GSGA	2.7556E+02	5.6312E+02	**4.2470E+02**	1.0625E+02	
F7	GA	1.0072E+03	1.1391E+03	1.0623E+03	3.3178E+01	+
	GS-SOMA	N/A	N/A	>1118.8	N/A	N/A
	GPEME	9.3316E +02	9.9286E + 02	9.5859E+02	2.5695E+01	+
	SHPSO	9.2248E +02	9.6363E + 02	9.3961E+02	9.0177E+00	+
	GSGA	9.1882E+02	9.3885E+02	**9.2725E+02**	6.0434E+00	

GSGA adds to the use of the strategy of the SGA updating mechanism (s2). This strategy can be considered a special characteristic of the GSGA and discriminates the GSGA from SAGA-GLS and other surrogate-assisted evolutionary algorithms. To show the effects of such a difference, a GA, SAGA-GLS, and GSGA are tested on the 50-D shifted rotated Rastrigin function (F5). The parameter settings of SAGA-GLS refer to those in [24]. However, for a fair comparison, the parameters of the GA and the prescreening strategy of SAGA-GLS are set the same as those in the GSGA. The convergence curves of the GA, SAGA-GLS, and GSGA are plotted in Fig. 8. Compared with SAGA-GLS, the GSGA has a greater improvement on the GA's optimization efficiency and can obtain a more accurate optimum. The higher efficiency of the GSGA can be attributed to the use of strategy s2. Specifically, in the process of obtaining the improved population (father) points, strategy s2 does not require additional exact function evaluations. However, in SAGA-GLS, the STR local search is used to obtain the improved population points, and this method requires a large number of exact function evaluations to evaluate the optima of surrogates. The additional exact function evaluations could make SAGA-GLS not optimize very efficiently. Moreover, in strategy s2, the neighbor partition strategy can maintain the diversity of the improved population points, thus enhancing the GSGA's global optimization ability. However, with the movement of trust regions in the STR local search, the improved population points in SAGA-GLS could be duplicated and the population diversity could decrease. Therefore, the global optimization ability of SAGA-GLS is weakened.

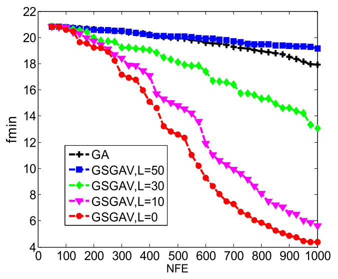

Fig. 9. Convergence curves of the GSGAV with different L values on the 50-D Ackley function (F3).

Finally, a potential advantage of the proposed GSGA is that it is very robust because of many surrogates used in the optimization process. Surrogates can sometimes be badly built because of improper parameter settings, irrational distribution of sample points, and other factors and surrogates' accuracy

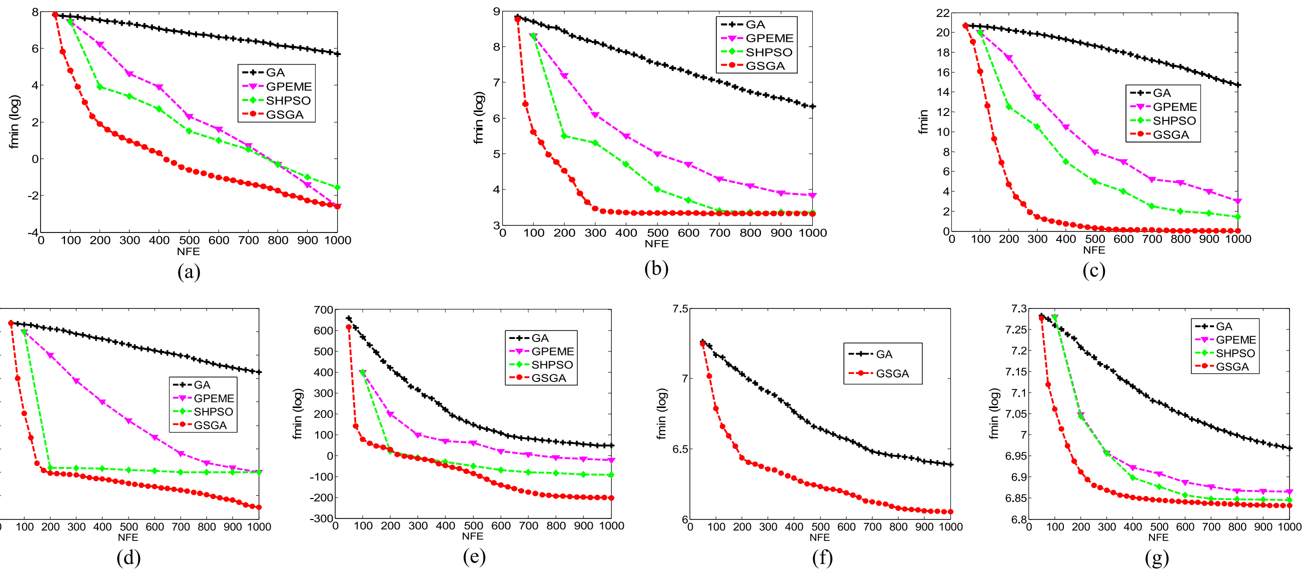

Fig. 10. Convergence curves of different algorithms for the 30-D benchmark problems. (a) F1. (b) F2. (c) F3. (d) F4. (e) F5. (f) F6. (g) F7.

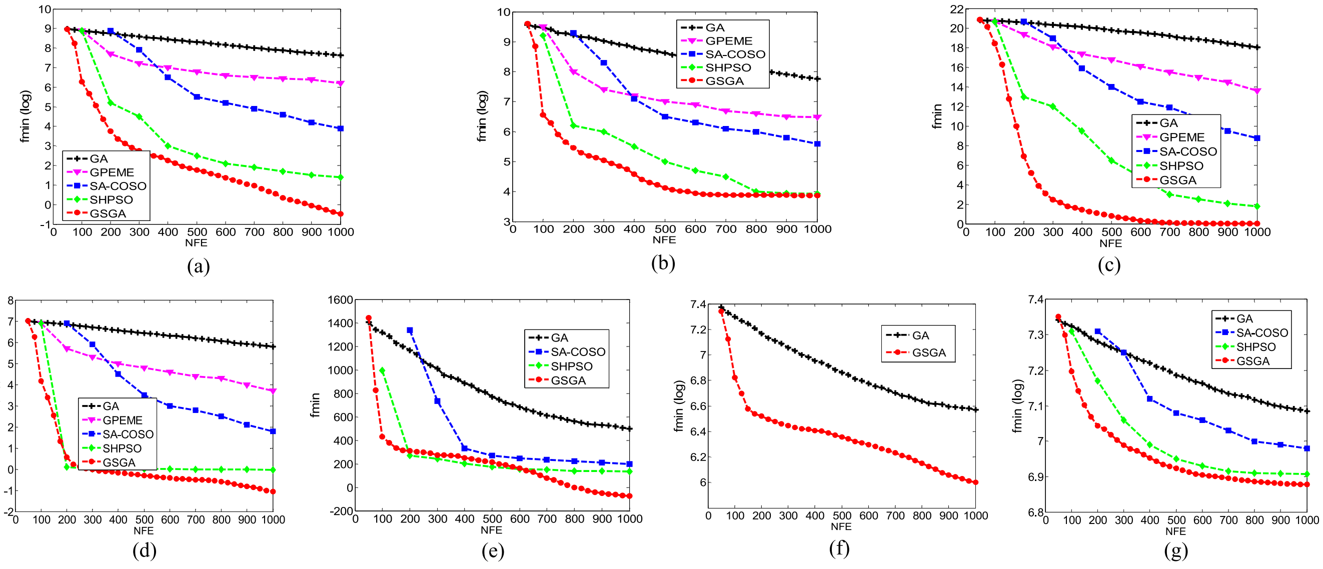

Fig. 11. Convergence curves of different algorithms for the 50-D benchmark problems. (a) F1. (b) F2. (c) F3. (d) F4. (e) F5. (f) F6. (g) F7.

thus cannot be guaranteed. Therefore, if a single surrogate or few surrogates are used in surrogate-assisted algorithms, potentially wrong information provided by the bad surrogates can mislead the search direction of the basic evolutionary or metaheuristic algorithms. However, because of the multiple surrogates used in the optimization process, the GSGA cannot be greatly influenced by a badly built surrogate or a few badly built surrogates. To test this viewpoint, we design a variant of the proposed GSGA (GSGAV). It is similar to the GA+s2 algorithm and only adds strategy 2 in the GA search process. In addition, some RBF surrogates are purposely built inaccurately in the metamodeling process of the GSGAV. The predicted responses of these surrogates are set to 0. The number of such inaccurate surrogates is set as L, and they are randomly selected. The GA and GSGAVs with different L values are tested on the 50-D Ackley function. The L values are set as 50, 30, 10, and 0. The convergence curves of the GA and different GSGAVs are plotted in Fig. 9. The GSGAV with $L = 50$ means that all built RBF surrogates are totally inaccurate. Fig. 9 shows that the poorly built RBF surrogates do have an effect on GSGAV. With a decrease in the number of badly built surrogates in the GSGAV, the optimization efficiency gradually improves. In particular, the GSGAV with $L = 50$ performs even worse than the basic GA. This finding is because the badly built surrogates provide incorrect optimum information and thus mislead the GA to search in an incorrect direction. This GSGAV search is equal to a random search. However, when only several surrogates are badly built in the GSGAV, its optimization efficiency is always better than the GA. Therefore, the proposed GSGA has good robustness because of the multiple surrogates used. The optimization efficiency of this algorithm can avoid being greatly influenced by several badly built surrogates in the optimization process.

In summary, the proposed strategies in the proposed optimization framework has been discussed in detail. The results show that these strategies have a positive effect on the proposed GSGA. The simplified Kriging surrogate is proven to be very suitable for high-dimensional expensive optimization problems. A small number of selected candidate child population points can help to speed up the search process of the GSGA. The GSGA has the advantage of good robustness in optimization because of the multiple surrogates used.

B. Comparison Results on 30-D Benchmark Problems

In this section, we make a performance comparison of different optimization algorithms, including the basic GA, GS-SOMA [47], GPEME [25], surrogate-assisted hierarchical PSO (SHPSO) [43], and the proposed GSGA. These algorithms are tested on 30-D benchmark problems. As the codes of GS-SOMA, GPEME, and SHPSO are not available online and not replicated by us, their optimization results are only cited from the above research papers. The optimization results of GS-SOMA are obtained from convergence figures. Therefore, only the mean optimal values are listed. The statistical results of GS-SOMA are thus unable to be obtained by t-test. The optimization results of the different algorithms are listed in Table IV, where Std. represents standard deviation. The t-test results at a confidence level of 95% that compare the GSGA to other algorithms are also listed in Table IV, where "+," "−," and "≈" indicate that the GSGA is significantly better than, significantly worse than, or comparable to the compared algorithm, respectively. The convergence curves of different benchmark functions are given in Fig. 10. The approximate convergence figures for GPEME and SHPSO are created based on the figures provided in their relative research papers.

The statistical results in Table IV show that the GSGA performs significantly better than or comparably well to the other compared algorithms. For instance, compared with the recently proposed SHPSO, the GSGA obtains better average results on almost all problems except for the rotated hybrid composition function (F6). Fig. 10 shows that the GSGA achieves significantly better convergence performance on most of the test problems. For the ellipsoid function (F1), the GSGA can obtain optimization results comparable to GPEME. The GSGA can converge to the optima faster than GPEME within 1000 function evaluations. However, GPEME can converge quickly at a constant speed and tends to converge more accurately than the GSGA. The high convergence accuracy of GPEME can be attributed to the DE/best/1 mutation strategy used, which is biased toward local search. For function F6, although the optimization results of the GSGA are comparable to those of SHPSO, the GSGA can obtain a more accurate mean optimization result. This finding shows that the performance of the GSGA is not robust for this function. The optimization results of the GSGA vary from 275 to 563. The reason could be attributed to the high complexity of the F6 function, which makes it difficult for the GSGA to escape from local optima. A similar phenomenon also happens on SHPSO, whose Std. value of the optimization results for F6 also reaches 85.

TABLE V
OPTIMIZATION RESULTS FOR THE 50-D PROBLEMS

Fun	Algorithm	Mean	Std.	t-test
F1	GA	2.0513E+03	2.8019E+02	+
	GPEME	2.2108E+02	8.1612E+01	+
	SA-COSO	5.1475E+01	1.6246E+01	+
	SHPSO	4.0281E+00	2.0599E+00	+
	GSGA	**6.2108E-01**	4.8487E-01	
	GSGA	Best	Median	Worst
		2.0371E-01	4.8669E-01	1.8685E+00
F2	GA	2.3357E+03	5.0027E+02	+
	GPEME	2.5828E+02	8.0188E+01	+
	SA-COSO	2.5258E+02	4.0744E+01	+
	SHPSO	5.0800E+01	3.0305E+00	+
	GSGA	**4.8214E+01**	7.6660E-01	
	GSGA	Best	Median	Worst
		4.6844E+01	4.8484E+01	4.9108E+01
F3	GA	1.8044E+01	5.1246E-01	+
	GPEME	1.3233E+01	1.5846E+00	+
	SA-COSO	8.9318E+00	1.0668E+00	+
	SHPSO	1.8389E+00	5.6370E-01	+
	GSGA	**2.1605E-02**	2.3796E-02	
	GSGA	Best	Median	Worst
		6.0733E-03	1.1110E-02	7.6502E-02
F4	GA	3.2820E+02	6.0478E+01	+
	GPEME	3.6646E+01	1.3176E+01	+
	SA-COSO	6.0062E+00	1.1043E+00	+
	SHPSO	9.4521E-01	6.1404E-02	+
	GSGA	**3.4657E-01**	7.1520E-02	
	GSGA	Best	Median	Worst
		2.7263E-01	3.2877E-01	4.4222E-01
F5	GA	4.9903E+02	8.2562E+01	+
	SA-COSO	1.9916E+02	3.0599E+01	+
	SHPSO	1.3442E+02	3.2256E+01	+
	GSGA	**-7.5825E+01**	4.9991E+01	
	GSGA	Best	Median	Worst
		-1.3969E+02	-7.5519E+01	1.2095E+01
F6	GA	7.1389E+02	8.5637E+01	+
	SHPSO	4.7438E+02	4.2029E+01	+
	GSGA	**4.0331E+02**	8.7598E+01	
	GSGA	Best	Median	Worst
		2.7183E+02	3.9084E+02	5.2487E+02
F7	GA	1.1932E+03	3.6502E+01	+
	SA-COSO	1.0809E+03	3.2859E+01	+
	SHPSO	9.9660E+02	2.2145E+01	+
	GSGA	**9.7078E+02**	1.8182E+01	
	GSGA	Best	Median	Worst
		9.4377E+02	9.6975E+02	1.0027E+03

C. Comparison Results for 50-D and 100-D Benchmark Problems

Most recently, optimization for higher dimensional problems has been studied by many researchers, such as Liu *et al.* [25], Sun *et al.* [37], Yu *et al.* [43], and so on. In order to further evaluate the efficiency of the GSGA on high-dimensional problems, here we perform experiments on 50-D and 100-D benchmark problems. Three surrogate-assisted algorithms, GPEME [25], SA-COSO [37], and SHPSO [43], are used for comparison. Tables V and VI give the statistical results for problems with 50-D and 100-D, respectively. The optimization results of these algorithms are cited from the corresponding studies. Because no experimental results of GPEME on F5, F6, and F7 were reported in Liu *et al.* [25], we do not compare the performance of these problems with

TABLE VI
OPTIMIZATION RESULTS FOR THE 100-D PROBLEMS

Fun	Algorithm	Mean	Std.	t-test
F1	GA	1.5841E+04	1.6892E+03	+
	SA-COSO	1.0332E+03	3.1718E+02	+
	SHPSO	7.6106E+01	2.1447E+01	+
	GSGA	**1.2329E+01**	9.3949E+00	
	GSGA	Best	Median	Worst
		2.6030E+00	1.0128E+01	2.7158E+01
F2	GA	1.2340E+04	1.8513E+03	+
	SA-COSO	2.7142E+03	1.1702E+02	+
	SHPSO	1.6559E+02	2.6366E+01	+
	GSGA	**1.0909E+02**	1.1763E+01	
	GSGA	Best	Median	Worst
		9.9743E+01	1.0604E+02	1.3938E+02
F3	GA	1.9884E+01	1.6680E-01	+
	SA-COSO	1.5756E+01	5.0245E-01	+
	SHPSO	4.1134E+00	5.9247E-01	+
	GSGA	**1.3189E+00**	9.6892E-01	
	GSGA	Best	Median	Worst
		1.5661E-01	1.2019E+00	2.8076E+00
F4	GA	1.2043E+03	6.9572E+01	+
	SA-COSO	6.3353E+01	1.9021E+01	+
	SHPSO	1.0704E+00	2.0485E-02	+
	GSGA	**7.0674E-01**	7.0635E-02	
	GSGA	Best	Median	Worst
		5.8091E-01	7.1981E-01	8.0469E-01
F5	GA	1.8001E+03	1.0843E+02	+
	SA-COSO	1.2731E+03	1.1719E+02	+
	SHPSO	8.0173E+02	7.2252E+01	+
	GSGA	**6.7250E+02**	2.9794E+01	
	GSGA	Best	Median	Worst
		6.2040E+02	6.8650E+02	7.0528E+02
F6	GA	8.3733E+02	6.3232E+01	+
	SHPSO	5.1619E+02	3.2060E+01	+
	GSGA	**4.4721E+02**	1.4258E+01	
	GSGA	Best	Median	Worst
		4.2241E+02	4.5033E+02	4.7264E+02
F7	GA	1.4495E+03	3.2954E+01	+
	SA-COSO	1.3657E+03	3.0867E+01	+
	SHPSO	1.4198E+03	3.8238E+01	+
	GSGA	**1.2567E+03**	2.4563E+01	
	GSGA	Best	Median	Worst
		1.2204E+03	1.2627E+03	1.2872E+03

50 dimensions. SA-COSO also did not test 50-D and 100-D F6 functions. The corresponding optimization results are also not listed. Tables V and VI show that the GSGA obtains better results with 1000 expensive function evaluations on all 50-D and 100-D benchmark problems. Fig. 11 provides the convergence profiles of the GSGA and the compared algorithms on the 50-D benchmark problems. The convergence profiles of different algorithms on the 100-D benchmark problems are attached in the supplementary file because of page limit. The performance superiority of GSGA on the 50-D and 100-D problems is more clear. The GSGA performs better than all the compared algorithms on all tested problems. For the 50-D F1, F3, and F4 functions, the GSGA can obtain the approximate global optimum within 1000 function evaluations. According to the convergence curves for the 50-D F4 (Griewank) function, SHPSO can initially converge faster than the GSGA. A reason could be that SHPSO is based on the PSO search mechanism, which could be more biased toward local search than the GA. Moreover, the accurate local metamodeling strategy in SHPSO could make SHPSO search fast in the initial optimization stage. However, the final optimization accuracy of SHPSO is worse than that found by the GSGA because SHPSO converges slowly in the later optimization process. This finding shows that the GSGA not only has high global search ability but also has high local search ability. In summary, the GSGA is very efficient for optimization of high-dimensional expensive problems.

V. CONCLUSION

This paper introduced a new efficient GSGA algorithm for the optimization of high-dimensional expensive problems. The optimization framework of the proposed algorithm is based on the basic GA. Compared with the GA, the GSGA first uses the optimum provided by the surrogate-based trust region local search method to guide the GA to search accurately. In addition, an SGA updating mechanism with a neighbor region partition strategy is proposed to guide the GA's crossover operation. Finally, a simplified Kriging-based EI prescreening strategy is also proposed to further improve the optimization efficiency of the proposed algorithm.

To validate the proposed GSGA, this algorithm is tested by several widely used high-dimensional numerical benchmark problems with dimensions varying from 30 to 100. The proposed strategies in the proposed optimization framework have been validated. The results show that these strategies can enhance the optimization efficiency of the proposed GSGA. Moreover, an overall performance comparison among the GA, GSGA, and four other surrogate-assisted metaheuristic algorithms, including GS-SOMA, GPEME, SA-COSO, and SHPSO, is made. The results show that the GSGA has a faster convergence speed and higher optimization accuracy than the other compared algorithms for most of 30-D benchmark problems and all 50-D and 100-D benchmark problems. The performance superiority of the GSGA over the other compared algorithms indicates that the proposed generalized surrogate-assisted evolutionary optimization framework in the GSGA is very efficient.

Although the proposed GSGA has achieved good optimization results for the benchmark problems, some aspects should still be focused on to further improve the optimization efficiency of the GSGA for high-dimensional expensive problems. First, the MaxNFE could be increased to study the performance of the GSGA. In our research, this maximum is set to 1000 because the optimization problem is considered very expensive (a function evaluation could need minutes or hours). Second, the execution time of the GSGA is a little expensive. The GSGA needs approximately 3 h to run an optimization process for the 100-D Ackley function (one function evaluation needs approximately 10.8 s). The reasons are attributed to the optimization for multiple surrogates, Kriging metamodeling and computer power. Therefore, how to shorten the execution time of the GSGA (e.g., use inexpensive surrogate-based prescreening strategies in the optimization process, use efficient optimization algorithms to optimize the built RBF surrogates, or use powerful computers) and make it suitable for optimization within a large number of function evaluations is worth studying. Third, our algorithm is based on the GA optimization framework. This algorithm can also be developed to the optimization frameworks of other metaheuristic algorithms. The corresponding performance should also be worth studying. Fourth, multifidelity surrogate [77] is an efficient metamodeling tool for improving the optimization efficiency of expensive problems. The GSGA integrated with multifidelity surrogates will also be worth studying to greatly improve the optimization efficiency of high-dimensional expensive problems.

REFERENCES

[1] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1998.

[2] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. IEEE Conf. 6th Int. Symp.*, 1995, pp. 39–43.

[3] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[4] Y.-W. Leung and Y. Wang, "An orthogonal genetic algorithm with quantization for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 5, no. 1, pp. 41–53, Feb. 2001.

[5] J.-T. Tsai, T.-K. Liu, and J.-H. Chou, "Hybrid Taguchi-genetic algorithm for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 4, pp. 365–377, Aug. 2004.

[6] H. Wang, H. Sun, C. Li, S. Rahnamayan, and J.-S. Pan, "Diversity enhanced particle swarm optimization with neighborhood search," *Inf. Sci.*, vol. 223, pp. 119–135, Feb. 2013.

[7] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.

[8] R. Mallipeddi, P. N. Suganthan, Q.-K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Appl. Soft Comput.*, vol. 11, no. 2, pp. 1679–1696, 2011.

[9] T. W. Simpson, A. J. Booker, D. Ghosh, A. A. Giunta, P. N. Koch, and R. J. Yang, "Approximation methods in multidisciplinary analysis and optimization: A panel discussion," *Struct. Multidiscipl. Optim.*, vol. 27, no. 5, pp. 302–313, 2004.

[10] N. Cressie, "The origins of Kriging," *Math. Geol.*, vol. 22, no. 3, pp. 239–252, 1990.

[11] H.-M. Gutmann, "A radial basis function method for global optimization," *J. Glob. Optim.*, vol. 19, no. 3, pp. 201–227, 2001.

[12] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, no. 3, pp. 199–222, 2004.

[13] G. G. Wang, "Adaptive response surface method using inherited latin hypercube design points," *J. Mech. Design*, vol. 125, no. 2, pp. 210–220, 2003.

[14] R. Jin, W. Chen, and T. W. Simpson, "Comparative studies of metamodelling techniques under multiple modelling criteria," *Struct. Multidiscipl. Optim.*, vol. 23, no. 1, pp. 1–13, 2001.

[15] S. Shan and G. G. Wang, "Development of adaptive RBF-HDMR model for approximating high dimensional problems," in *Proc. ASME Int. Design Eng. Techn. Conf. Comput. Inf. Eng. Conf.*, 2009, pp. 727–740.

[16] S. Shan and G. G. Wang, "Metamodeling for high dimensional simulation-based design problems," *J. Mech. Design*, vol. 132, no. 5, 2010, Art. no. 051009.

[17] G. H. Cheng, A. Younis, K. H. Hajikolaei, and G. G. Wang, "Trust region based mode pursuing sampling method for global optimization of high dimensional design problems," *J. Mech. Design*, vol. 137, no. 2, 2015, Art. no. 021407.

[18] H. Wang, Y. Jin, and J. Doherty, "Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2664–2677, Sep. 2017.

[19] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm Evol. Comput.*, vol. 1, no. 2, pp. 61–70, 2011.

[20] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution—An updated survey," *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.

[21] R. T. Haftka, D. Villanueva, and A. Chaudhuri, "Parallel surrogate-assisted global optimization with expensive functions—A survey," *Struct. Multidiscipl. Optim.*, vol. 54, no. 1, pp. 3–13, 2016.

[22] M. T. M. Emmerich, K. C. Giannakoglou, and B. Naujoks, "Single-and multiobjective evolutionary optimization assisted by Gaussian random field metamodels," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 456–474, Aug. 2006.

[23] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, "Expensive multiobjective optimization by MOEA/D with Gaussian process model," *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 421–439, Jun. 2010.

[24] Z. Zhou, Y.-S. Ong, P. B. Nair, A. J. Keane, and K. Y. Lum, "Combining global and local surrogate models to accelerate evolutionary optimization," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 37, no. 1, pp. 66–76, Jan. 2007.

[25] B. Liu, Q. Zhang, and G. G. E. Gielen, "A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 180–192, Apr. 2014.

[26] B. Liu, Q. Chen, Q. Zhang, G. Gielen, and V. Grout, "Behavioral study of the surrogate model-aware evolutionary search framework," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2014, pp. 715–722.

[27] B. Liu, H. Yang, and M. J. Lancaster, "Global optimization of microwave filters based on a surrogate model-assisted evolutionary algorithm," *IEEE Trans. Microw. Theory Techn.*, vol. 65, no. 6, pp. 1976–1985, Jun. 2017.

[28] B. Liu, V. Grout, and A. Nikolaeva, "Efficient global optimization of actuator based on a surrogate model assisted hybrid algorithm," *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5712–5721, Jul. 2018.

[29] B. Liu, Q. Zhang, and G. Gielen, "A surrogate-model-assisted evolutionary algorithm for computationally expensive design optimization problems with inequality constraints," in *Simulation-Driven Modeling and Optimization*. Cham, Switzerland: Springer, 2016, pp. 347–370.

[30] D. Guo, Y. Jin, J. Ding, and T. Chai, "Heterogeneous ensemble-based infill criterion for evolutionary multiobjective optimization of expensive problems," *IEEE Trans. Cybern.*, vol. 49, no. 3, pp. 1012–1025, Mar. 2019.

[31] F. Li, X. Cai, and L. Gao, "Ensemble of surrogates assisted particle swarm optimization of medium scale expensive problems," *Appl. Soft Comput.*, vol. 74, pp. 291–305, Jan. 2019.

[32] L. G. Fonseca, A. C. Lemonge, and H. J. C. Barbosa, "A study on fitness inheritance for enhanced efficiency in real-coded genetic algorithms," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2012, pp. 1–8.

[33] R. G. Regis, "Particle swarm with radial basis function surrogates for expensive black-box optimization," *J. Comput. Sci.*, vol. 5, no. 1, pp. 12–23, 2014.

[34] R. G. Regis, "Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 326–347, Jun. 2014.

[35] R. Mallipeddi and M. Lee, "An evolving surrogate model-based differential evolution algorithm," *Appl. Soft Comput.*, vol. 34, pp. 770–787, Sep. 2015.

[36] W. Gong, A. Zhou, and Z. Cai, "A multioperator search strategy based on cheap surrogate models for evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 5, pp. 746–758, Oct. 2015.

[37] C. Sun, Y. Jin, R. Cheng, J. Ding, and J. Zeng, "Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 644–660, Aug. 2017.

[38] C. Praveen and R. Duvigneau, "Low cost PSO using metamodels and inexact pre-evaluation: Application to aerodynamic shape design," *Comput. Methods Appl. Mech. Eng.*, vol. 198, nos. 9–12, pp. 1087–1096, 2009.

[39] C. Sun, Y. Jin, J. Zeng, and Y. Yu, "A two-layer surrogate-assisted particle swarm optimization algorithm," *Soft Comput.*, vol. 19, no. 6, pp. 1461–1475, 2015.

[40] S. M. Elsayed, T. Ray, and R. A. Sarker, "A surrogate-assisted differential evolution algorithm with dynamic parameters selection for solving expensive optimization problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2014, pp. 1062–1068.

[41] M. D. Parno, T. Hemker, and K. R. Fowler, "Applicability of surrogates to improve efficiency of particle swarm optimization for simulation-based problems," *Eng. Optim.*, vol. 44, no. 5, pp. 521–535, 2012.

[42] Y. Tang, J. Chen, and J. Wei, "A surrogate-based particle swarm optimization algorithm for solving optimization problems with expensive black box functions," *Eng. Optim.*, vol. 45, no. 5, pp. 557–576, 2013.

[43] H. Yu, Y. Tan, J. Zeng, C. Sun, and Y. Jin, "Surrogate-assisted hierarchical particle swarm optimization," *Inf. Sci.*, vols. 454–455, pp. 59–72, Jul. 2018.

[44] Y. S. Ong, P. B. Nair, and A. J. Keane, "Evolutionary optimization of computationally expensive problems via surrogate modeling," *AIAA J.*, vol. 41, no. 4, pp. 687–696, 2003.

[45] Z. Zhou, Y. S. Ong, M. H. Nguyen, and D. Lim, "A study on polynomial regression and Gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2005, pp. 2832–2839.

[46] Z. Zhou, Y. S. Ong, M. H. Lim, and B. S. Lee, "Memetic algorithm using multi-surrogates for computationally expensive optimization problems," *Soft Comput.*, vol. 11, no. 10, pp. 957–971, 2007.

[47] D. Lim, Y. Jin, Y.-S. Ong, and B. Sendhoff, "Generalizing surrogate-assisted evolutionary computation," *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 329–355, Jun. 2010.

[48] M. N. Le, Y. S. Ong, S. Menzel, Y. Jin, and B. Sendhoff, "Evolution by adapting surrogates," *Evol. Comput.*, vol. 21, no. 2, pp. 313–340, 2013.

[49] Y. Wang, D.-Q. Yin, S. Yang, and G. Sun, "Global and local surrogate-assisted differential evolution for expensive constrained optimization problems with inequality constraints," *IEEE Trans. Cybern.*, vol. 49, no. 5, pp. 1642–1656, May 2019.

[50] M. Meckesheimer, A. J. Booker, R. R. Barton, and T. W. Simpson, "Computationally inexpensive metamodel assessment strategies," *AIAA J.*, vol. 40, no. 10, pp. 2053–2060, 2002.

[51] S. Shan and G. G. Wang, "Turning black-box functions into white functions," *J. Mech. Design*, vol. 133, no. 3, 2011, Art. no. 031003.

[52] H. Wang, L. Tang, and G. Li, "Adaptive MLS-HDMR metamodeling techniques for high dimensional problems," *Expert Syst. Appl.*, vol. 38, no. 11, pp. 14117–14126, 2011.

[53] X. Cai, H. Qiu, L. Gao, P. Yang, and X. Shao, "An enhanced RBF-HDMR integrated with an adaptive sampling method for approximating high dimensional problems in engineering design," *Struct. Multidiscipl. Optim.*, vol. 53, no. 6, pp. 1209–1229, 2016.

[54] X. Cai, H. Qiu, L. Gao, and X. Shao, "Metamodeling for high dimensional design problems by multi-fidelity simulations," *Struct. Multidiscipl. Optim.*, vol. 56, no. 1, pp. 151–166, 2017.

[55] L. Tang, H. Wang, and G. Li, "Advanced high strength steel springback optimization by projection-based heuristic global search algorithm," *Mater. Design*, vol. 43, pp. 426–437, Jan. 2013.

[56] K. H. Hajikolaei, G. H. Cheng, and G. G. Wang, "Optimization on metamodeling-supported iterative decomposition," *J. Mech. Design*, vol. 138, no. 2, 2016, Art. no. 021401.

[57] D. Wu, K. H. Hajikolaei, and G. G. Wang, "Employing partial metamodels for optimization with scarce samples," *Struct. Multidiscipl. Optim.*, vol. 57, no. 3, pp. 1329–1343, 2018.

[58] A. A. Mullur and A. Messac, "Metamodeling using extended radial basis functions: A comparative approach," *Eng. Comput.*, vol. 21, no. 3, p. 203, 2006.

[59] A. Díaz-Manríquez, G. Toscano, and C. A. Coello Coello, "Comparison of metamodeling techniques in evolutionary algorithms," *Soft Comput.*, vol. 21, no. 19, pp. 5647–5663, 2017.

[60] M. J. D. Powell, "The theory of radial basis functions in 1990," in *Advances in Numerical Analysis, Volume II of Wavelets, Subdivision Algorithms, and Radial Basis Functions*, W. Light, Ed., Oxford Univ. Press, 1992, pp. 105–210.

[61] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to Derivative-Free Optimization*. Philadelphia, PA, USA: SIAM, 2009.

[62] S. M. Wild and C. Shoemaker, "Global convergence of radial basis function trust-region algorithms for derivative-free optimization," *SIAM Rev.*, vol. 55, no. 2, pp. 349–371, 2013.

[63] R. G. Regis and S. M. Wild, "CONORBIT: Constrained optimization by radial basis function interpolation in trust regions," *Optim. Methods Softw.*, vol. 32, no. 3, pp. 552–580, 2017.

[64] T. Long, X. Li, R. Shi, J. Liu, X. Guo, and L. Liu, "Gradient-free trust-region-based adaptive response surface method for expensive aircraft optimization," *AIAA J.*, vol. 56, no. 2, pp. 862–873, 2018.

[65] J. Y. Cho and M. H. Oh, "Sequential approximate optimization procedure based on sample-reusable moving least squares meta-model and its application to design optimizations," *Comput. Model. Eng. Sci.*, vol. 66, no. 3, pp. 187–213, 2010.

[66] S. N. Lophaven, H. B. Nielsen, and J. Søndergaard, *DACE: A MATLAB Kriging Toolbox*. Lyngby, Denmark: Tech. Univ. Denmark, 2002.

[67] K. Rasheed, "Guided crossover: A new operator for genetic algorithm based optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 1999, pp. 1535–1541.

[68] S. Kitayama, M. Arakawa, and K. Yamazaki, "Sequential approximate optimization using radial basis function network for engineering optimization," *Optim. Eng.*, vol. 12, no. 4, pp. 535–557, 2011.

[69] D. E. Finkel, *DIRECT Optimization Algorithm User Guide*, vol. 2, Center Res. Sci. Comput., North Carolina State Univ., Raleigh, NC, USA, pp. 1–14, 2003.

[70] D. J. J. Toal, N. W. Bressloff, and A. J. Keane, "Kriging hyper parameter tuning strategies," *AIAA J.*, vol. 46, no. 5, pp. 1240–1252, 2008.

[71] W. J. Welch, R. J. Buck, J. Sacks, H. P. Wynn, T. J. Mitchell, and M. D. Morris, "Screening, predicting, and computer experiments," *Technometrics*, vol. 34, no. 1, pp. 15–25, 1992.

[72] M. A. Bouhlel, N. Bartoli, R. G. Regis, A. Otsmane, and J. Morlier, "Efficient global optimization for high-dimensional constrained problems by using the Kriging models combined with the partial least squares method," *Eng. Optim.*, vol. 50, no. 12, pp. 1–16, 2018.

[73] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *J. Glob. Optim.*, vol. 13, no. 4, pp. 455–492, 1998.

[74] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, Feb. 2002.

[75] P. N. Suganthan *et al.*, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," School EEE, Nanyang Technol. Univ., Singapore, KanGAL Rep. 2005005, 2005.

[76] J. S. Kowalik and M. R. Osborne, *Methods for Unconstrained Optimization Problems*. New York, NY, USA: Elsevier, 1968.

[77] X. Cai, H. Qiu, L. Gao, L. Wei, and X. Shao, "Adaptive radial-basis-function-based multifidelity metamodeling for expensive black-box problems," *AIAA J.*, vol. 55, no. 7, pp. 2424–2436, 2017.

Xiwen Cai received the B.Sc. degree in mechanical design and theory from the Wuhan Institute of Technology, Wuhan, China, in 2012, and the Ph.D. degree in mechanical design and theory from the Huazhong University of Science and Technology (HUST), Wuhan, in 2017.

He is currently a Post-Doctoral Fellow with the School of Mechanical Science and Engineering, HUST. His current research interests include surrogates, surrogate-based optimization, surrogate-assisted evolutionary computation, and machine learning.

Liang Gao received the B.Sc. degree in mechatronic engineering from Xidian University, Xi'an, China, in 1996, and the Ph.D. degree in mechatronic engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2002.

He is a Professor of the Department of Industrial and Manufacturing Systems Engineering, the Deputy Director of State Key Laboratory of Digital Manufacturing Equipment and Technology, and the Vice Dean of Research and Development Office, HUST. He was supported by the Program for New Century Excellent Talents in University in 2008 and the National Science Fund for Distinguished Young Scholars of China in 2018. His current research interests include intelligent optimization algorithms, big data, and deep learning with theirs application in design and manufacturing. He has published over 200 papers indexed by SCIE, authored 7 monographs.

Dr. Gao currently serves as the Co-Editor-in-Chief for *IET Collaborative Intelligent Manufacturing*, an Associate Editor of *Swarm and Evolutionary Computation*, and the *Journal of Industrial and Production Engineering*.

Xinyu Li received the Ph.D. degree in industrial engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2009.

He is an Associate Professor with the Department of Industrial and Manufacturing Systems Engineering, State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, HUST. He had published over 80 refereed papers. His current research interests include intelligent optimization algorithm, intelligent scheduling, and machine learning.