**Yifan Tang**
School of Aerospace Engineering,
Beijing Institute of Technology,
Key Laboratory of Dynamics and Control of Flight Vehicle, Ministry of Education,
Beijing 100081, China
e-mail: bit_steventang@163.com

**Teng Long**
School of Aerospace Engineering,
Beijing Institute of Technology,
Key Laboratory of Dynamics and Control of Flight Vehicle, Ministry of Education,
Beijing 100081, China
e-mail: tenglong@bit.edu.cn

**Renhe Shi**[1]
School of Aerospace Engineering,
Tsinghua University,
Beijing 100084, China
e-mails: srenhe@163.com; shirenhe@bit.edu.cn

**Yufei Wu**
School of Aerospace Engineering,
Beijing Institute of Technology;
Key Laboratory of Dynamics and Control of Flight Vehicle, Ministry of Education,
Beijing 100081, China
e-mail: bitwuyufei@163.com

**G. Gary Wang**
School of Mechatronic Systems Engineering,
Simon Fraser University,
Surrey, BC V3 T 0A3, Canada
e-mail: gary_wang@sfu.ca

# Sequential Radial Basis Function-Based Optimization Method Using Virtual Sample Generation

*To further reduce the computational expense of metamodel-based design optimization (MBDO), a novel sequential radial basis function (RBF)-based optimization method using virtual sample generation (SRBF-VSG) is proposed. Different from the conventional MBDO methods with pure expensive samples, SRBF-VSG employs the virtual sample generation mechanism to improve the optimization efficiency. In the proposed method, a least squares support vector machine (LS-SVM) classifier is trained based on expensive real samples considering the objective and constraint violation. The classifier is used to determine virtual points without evaluating any expensive simulations. The virtual samples are then generated by combining these virtual points and their Kriging responses. Expensive real samples and cheap virtual samples are used to refine the objective RBF metamodel for efficient space exploration. Several numerical benchmarks are tested to demonstrate the optimization capability of SRBF-VSG. The comparison results indicate that SRBF-VSG generally outperforms the competitive MBDO methods in terms of global convergence, efficiency, and robustness, which illustrates the effectiveness of virtual sample generation. Finally, SRBF-VSG is applied to an airfoil aerodynamic optimization problem and a small Earth observation satellite multidisciplinary design optimization problem to demonstrate its practicality for solving real-world optimization problems.* [DOI: 10.1115/1.4046650]

*Keywords: design optimization, metamodeling, virtual sample generation*

## 1 Introduction

**1.1 Research Background and Literature Review.** Metamodel-based design optimization (MBDO) methods have been widely employed in simulation-driven engineering design applications to alleviate the computational burden [1]. In MBDO methods, a metamodel is constructed to approximate computationally expensive simulations, e.g., computational fluid dynamics (CFD) and finite element analysis (FEA). Since often only a limited number of samples are required to construct a metamodel, the MBDO methods improve the optimization efficiency as compared with traditional numerical optimization techniques such as genetic algorithm (GA) and particle swarm optimization (PSO) [2]. Long et al. [3] and Jin et al. [4] compared commonly used metamodeling methods systematically and pointed out that Kriging (KRG) [5] and radial basis function (RBF) [6] generally outperform other methods in terms of approximation accuracy, robustness, and efficiency with limited samples.

In the past decades, some adaptive MBDO methods have been developed, where the metamodels are adaptively updated for global exploration and local exploitation according to certain infill sampling strategies. For instance, the efficient global optimization (EGO) [7] and its variants (e.g., MSEGO [8], SuperEGO [9]) gradually infill new points by maximizing the expected improvement criterion; the mode-pursing sampling (MPS) [10] and its variants (e.g., TR-MPS [11], CiMPS [12]) generate discriminative sequential samples toward the global optimum according to the specific probability density distribution function from metamodel predictions [13]; the adaptive response surface method (ARSM) [14] and its variants (e.g., TR-ARSM [15], ARSM-ISES [16]) adaptively reshape the search space for bias sampling to improve the optimization efficiency and convergence. In recent years, state-of-the-art technologies such as machine learning and high-dimensional (HD) modeling representation (HDMR) have been applied to MBDO methods to further enhance their optimization capacity for black-box optimization problems. For instance, Shi et al. [17] proposed a sequential radial basis function using the support vector machine method (SRBF-SVM), where the fuzzy c-means clustering method and the support vector machine (SVM) classifier are utilized to identify the interesting sampling region where the global optimum is probably located. A partial metamodel-based optimization method combined with the trust region was proposed by Wu et al. [18] to enhance the global convergence with limited computational resources. Rouhi et al. [19] presented a multistep metamodel-based optimization method, where the design space is reduced by narrowing down the side constraints of the design variables around the optimum points obtained from the previous steps. Li et al. [20] integrated EGO into HDMR to improve the capability of EGO for high-dimensional problems. Ran et al. [21] proposed a hybrid metamodel-based two-level global optimization method to search the significant design space, where a local search is performed to obtain the global optimum. Moreover, some penalty-free constraint handling mechanisms have been developed for solving expensive constrained black-box problems such as ConstrLMSRBF [22], COBRA [23], KCGO [24], FSRBF [25], and FLT-AKM [26].

Note that most aforementioned MBDO methods purely use samples from expensive simulations for optimization. Thus, the

[1]Corresponding author.

Contributed by the Design Automation Committee of ASME for publication in the Journal of Mechanical Design. Manuscript received June 10, 2019; final manuscript received January 19, 2020; published online May 21, 2020. Assoc. Editor: Wei Chen.

  

computational cost is still high when an excessive number of simulation evaluations are required to build a sufficiently accurate metamodel. To address this issue, some novel metamodeling techniques such as multimodel fusion [27–29] and gradient-enhanced metamodels [30–32] were developed. In multimodel fusion methods, cheap low-fidelity samples are integrated with a limited number of expensive high-fidelity samples to enhance the metamodeling accuracy and optimization efficiency [33]. However, the variable fidelity models or data are not available for some problems, which limits the application of multimodel fusion in practices. For gradient-enhanced metamodeling methods, the inexpensive gradients are incorporated into the construction of a metamodel to improve the approximation accuracy [30]. However, gradient-enhanced metamodeling methods are only suitable for specific problems (e.g., structural and aerodynamic optimization) whose gradient information is cheap to obtain. Thus, it is still necessary and promising to investigate other approaches to further enhance the efficiency of MBDO methods.

Recently, the virtual sample generation (VSG) technique becomes attractive in machine learning [34]. VSG technique aims at constructing artificial samples to enlarge the training sample sets [35]. Generally, virtual samples can be constructed by prior knowledge, addition of noises, and existing datasets [36]. The prior knowledge-based VSG method generates virtual samples by extracting the prior knowledge hidden in the questions, but the adaptability is low. The addition-of-noises based VSG method generates virtual samples by adding small normal noises to each sample, but this approach lacks sufficient justification [36].

Different from the prior knowledge and addition of noises methods, the VSG method based on existing datasets generates a great number of new virtual samples with the information hidden in the raw samples, which releases the requirement of prior knowledge about relevant research fields [37]. In the literature, Chongfu [38] proposed an information diffusion theory to generate virtual samples. Li et al. [39] proposed the mega-trend-diffusion technique by considering the position information of each training sample. Lin et al. [40] estimated the domain ranges of datasets to help generate virtual samples. Li and Lin [41] constructed virtual samples based on possibility functions, which is estimated by Gaussian kernel functions and intervalization technique [42]. In addition to the aforementioned applications to classification tasks, some researchers are committed to applying the VSG method to regression tasks. For instance, Li et al. [43] proposed a procedure based on the nonparametric approach and fuzzy techniques to generate virtual samples considering attribute dependence, which improves the forecasting accuracy of multiple linear regression model and support vector regression model. Chen et al. [37] proposed a PSO-based virtual sample generation method for regression tasks. Based on the work in Ref. [37], Gong et al. [44] combined the Monte Carlo method with PSO to further address the data scarcity issue.

In general, only a small number of samples are available in the early stage of the engineering optimization practices. Thus, the metamodel construction is essentially a regression task with small-scale samples. However, no study has been reported on the application of VSG to MBDO methods according to the authors' best knowledge.

**1.2 Motivations and Contributions.** To further release the computational burden of MBDO methods, a novel sequential radial basis function-based optimization method using virtual sample generation (SRBF-VSG) is proposed in this paper. In the proposed method, the VSG technique is employed to reduce the required expensive simulation calls for metamodeling, which is an innovative endeavor in the MBDO field. Moreover, a VSG-based metamodel management strategy is developed to balance the exploration and exploitation of design space, which provides SRBF-VSG with promising optimization capacity.

The remainder of the paper is organized as follows. Section 2 briefly reviews the fundamental mathematical tools used in SRBF-VSG. Section 3 presents details of the proposed SRBF-VSG and the VSG-based metamodel management strategy. In Sec. 4, SRBF-VSG is tested on a number of benchmark problems through comparing with a number of well-known MBDO methods. The detailed iteration process on a numerical benchmark is illustrated to investigate the benefits of using KRG-based virtual samples. In Sec. 5, SRBF-VSG is applied to an airfoil aerodynamic optimization problem and an Earth observation satellite optimization problem. Finally, Sec. 6 presents the conclusions and the future work.

## 2 Mathematical Foundations of SRBF-VSG

In this section, RBF/KRG metamodels and the least squares support vector machine (LS-SVM) classifier are briefly reviewed.

**2.1 Kriging and Radial Basis Function Metamodels.** KRG is an interpolation metamodeling technique as shown in Eq. (1):

$$f_{KRG}(\boldsymbol{x}) = G(\boldsymbol{x}) + Z(\boldsymbol{x}) \tag{1}$$

where $G(\boldsymbol{x})$ is a global model indicating the changing trend of approximation objects in the design space and $Z(\boldsymbol{x})$ is a stochastic Gaussian process determining the approximation ability of the KRG metamodel. The covariance matrix of $Z(\boldsymbol{x})$ is given as follows:

$$Cov(\boldsymbol{Z}(\boldsymbol{x}_i), \boldsymbol{Z}(\boldsymbol{x}_j)) = \sigma^2 \boldsymbol{R}[R(\boldsymbol{x}_i, \boldsymbol{x}_j)] \tag{2}$$

where $\sigma^2$ is the process variance of $Z(\boldsymbol{x})$; $\boldsymbol{R}$ is the correlation matrix; and $R(.)$ is a Gaussian correlation function that controls the smoothness of the KRG metamodel.

RBF is a multidimensional interpolation method based on discrete samples. The formulation of RBF metamodel is given as Eq. (3):

$$\begin{cases} f_{RBF}(\boldsymbol{x}) &= \boldsymbol{\beta}^T \boldsymbol{\phi}(\boldsymbol{x}) \\ \boldsymbol{\phi}(\boldsymbol{x}) &= [\phi(\|\boldsymbol{x} - \boldsymbol{x}_1\|) \quad \dots \quad \phi(r_i) \quad \dots \quad \phi(\|\boldsymbol{x} - \boldsymbol{x}_{n_s}\|)]^T \\ \boldsymbol{\beta} &= [\beta_1 \quad \beta_2 \quad \dots \quad \beta_{n_s}]^T \end{cases} \tag{3}$$

where $\boldsymbol{x}$ is the unknown point; $n_s$ is the number of constructing points; $\phi(r_i)$ is the radial basis function; $r_i = \|\boldsymbol{x} - \boldsymbol{x}_i\|$, $i = 1, 2, \dots, n_s$ is the Euclidean distance of two points; $\boldsymbol{x}_i$ is the $i$-th constructing point; and $\boldsymbol{\beta}$ is a coefficient vector of the RBF, which can be obtained by considering the interpolation property. In this paper, the multiquadratic basis function is used as the radial basis function, as given in Eq. (4):

$$\phi(r) = (r^2 + c^2)^{1/2} \tag{4}$$

where $c$ is the shape coefficient. More details about RBF and KRG metamodels can be found in Refs. [5,6]. A MATLAB toolbox SURROGATES [45] is employed to construct RBF and KRG metamodels in this paper. All tuning parameters are set to be the default values.

Comparison studies in Refs. [3] and [4] have proved that KRG generally has better approximation accuracy than RBF in the condition of relatively sparse samples and lack of prior knowledge, whereas the construction expense of KRG is more expensive than that of RBF.

**2.2 Least Squares Support Vector Machine Classifier.** LS-SVM is a natural extension of SVM to improve the training efficiency. The LS-SVM model utilizes the least square linear system as the loss function, and the inequality constraint functions in SVM are revised as the equality constraints [46].

For a given training set $\boldsymbol{S}$ with $n_t$ training samples:

$$\boldsymbol{S} = \{(\boldsymbol{x}_i, y_i) | i = 1, 2, 3, \cdots, n_t\} \tag{5}$$

where $\boldsymbol{x}_i$ is an input vector and $y_i \in \{-1, +1\}$ represents a corresponding desired output. The input space is mapped into a higher dimensional feature space by using a nonlinear function $\boldsymbol{\varphi}(.)$. The desired optimal classification hyperplane must satisfy the following conditions:

$$\begin{cases} \boldsymbol{w}^T \boldsymbol{\varphi}(\boldsymbol{x}_i) + b \geq +1 & y_i = +1 \\ \boldsymbol{w}^T \boldsymbol{\varphi}(\boldsymbol{x}_i) + b \leq -1 & y_i = -1 \end{cases} \tag{6}$$

where $\boldsymbol{w}$ is the normal vector of the hyperplane and $b$ is an offset quantity. Then, the classification model can be described as follows:

$$f(\boldsymbol{x}_i) = \text{sgn}(\boldsymbol{w}^T \boldsymbol{\varphi}(\boldsymbol{x}_i) + b) \tag{7}$$

To find the optimal hyperplane, the LS-SVM model is obtained by solving the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{w},b,\boldsymbol{e}} \quad & l_2(\boldsymbol{w}, b, \boldsymbol{e}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \gamma \sum_{i=1}^{n_t} e_i^2 \\ \text{s.t.} \quad & y_i[\boldsymbol{w}^T \boldsymbol{\varphi}(\boldsymbol{x}_i) + b] = 1 - e_i, \quad i = 1, \ldots, n_t \end{aligned} \tag{8}$$

where $e_i$ is a slack variable and $\gamma$ is a small positive regularization constant to balance the fitness error and model complexity. The optimization problem is solved by the Lagrange multiplier method, as shown in Eq. (9):

$$L_2(\boldsymbol{w}, b, \boldsymbol{e}; \boldsymbol{\alpha}) = l_2(\boldsymbol{w}, b, \boldsymbol{e}) - \sum_{i=1}^{n_t} \alpha_i \{y_i[\boldsymbol{w}^T \boldsymbol{\varphi}(\boldsymbol{x}_i) + b] - 1 + e_i\} \tag{9}$$

where $\alpha_i$ is a positive Lagrange multiplier. Thus, the classification model can be formulated as follows:

$$f(\boldsymbol{x}_i) = \text{sgn}\left[\sum_{i=1}^{n} y_i \alpha_i \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_i) + b\right] \tag{10}$$

where $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_i)$ is a kernel function.

More details are available in Ref. [46]. A MATLAB toolkit LSSVMlab [47] is employed for classifier training with default tuning parameters in this paper.

## 3 Methodology of Sequential Radial Basis Function Using Virtual Sample Generation

In this section, the overall procedure of SRBF-VSG is first presented. Then, a VSG-based metamodel management strategy is discussed about how SRBF-VSG reduces the computational budget of metamodel updating and improves the optimization performance.

### 3.1 Overall Procedure of SRBF-VSG.

A general engineering optimization problem can be formulated as follows:

$$\begin{aligned} & \min f(\boldsymbol{x}) \\ \text{s.t.} \quad & g_i(\boldsymbol{x}) \le 0, \qquad i = 1, 2, \ldots, n \\ & \boldsymbol{x}_L \le \boldsymbol{x} \le \boldsymbol{x}_U \end{aligned} \tag{11}$$

where $f(\boldsymbol{x})$ is the objective; $g_i(\boldsymbol{x})$ is the $i$-th computationally expensive constraint; $n$ is the number of expensive constraints; and $\boldsymbol{x}_L$ and $\boldsymbol{x}_U$ denote the lower and upper bounds of the design space.

When $f(\boldsymbol{x})$ and $g_i(\boldsymbol{x})$ are replaced by RBF metamodels $\tilde{f}(\boldsymbol{x})$ and $\tilde{g}_i(\boldsymbol{x})$, respectively, the original optimization problem in Eq. (11) can be rewritten as Eq. (12):

$$\begin{aligned} & \min \quad \tilde{f}(\boldsymbol{x}) \\ \text{s.t.} \quad & \tilde{g}_i(\boldsymbol{x}) \le 0, \qquad i = 1, 2, \ldots, n \\ & \boldsymbol{x}_L \le \boldsymbol{x} \le \boldsymbol{x}_U \end{aligned} \tag{12}$$

Before introducing the methodology of SRBF-VSG, some concepts in SRBF-VSG are explained in Table 1 based on the notions of samples, cheap points, and superior cheap points defined in Ref. [17]. In SRBF-VSG, the RBF metamodels are gradually refined using the newly added real samples and virtual samples. The flowchart of SRBF-VSG is shown in Fig. 1, and the procedure is explained in details as follows:

*Step 1.* Configuration parameters are determined including the design space $[\boldsymbol{x}_L, \boldsymbol{x}_U]$, the number of initial real samples $n_i$, the number of cheap points $n_c$, and the number of virtual samples generated at each iteration $n_{vs}$. The iteration counter $k$ is set to be one, i.e., $k = 1$.

*Step 2.* The maximin latin hypercube design (LHD) method is employed to generate initial real points, which are stored in the real points database $\boldsymbol{X}_r$. Their real objective and constraint responses are calculated and then stored in the real objective response database $\boldsymbol{Y}_r$ and the real constraint response database $\boldsymbol{C}_r$, respectively. Finally, the real samples database $\boldsymbol{S}_r$ is updated by the aggregate of $\boldsymbol{X}_r$, $\boldsymbol{Y}_r$, and $\boldsymbol{C}_r$.

*Step 3.* The RBF metamodels $\tilde{f}(\boldsymbol{x})$ and $\tilde{g}_i(\boldsymbol{x})$ are constructed to approximate $f(\boldsymbol{x})$ and $g_i(\boldsymbol{x})$, respectively. In this method, $\tilde{f}(\boldsymbol{x})$ is constructed or updated based on real and virtual samples, while $\tilde{g}_i(\boldsymbol{x})$ is constructed or updated only based on real samples.

*Step 4.* The GA is employed to perform the global optimization on the current RBF metamodels to get the pseudo optimum point $\boldsymbol{x}^{(k)}$, which is regarded as the current potential global optimum. The expensive simulations are invoked to calculate real responses $f(\boldsymbol{x}^{(k)})$ and $g_i(\boldsymbol{x}^{(k)})$ at $\boldsymbol{x}^{(k)}$. In the end, the pseudo optimum point $\boldsymbol{x}^{(k)}$ and its responses are stored in the historical pseudo optimum samples database and real samples database.

*Step 5.* The termination criteria in Eq. (13), containing the optimality criterion $C_1$, the approximation criterion $C_2$, and the computational budget criterion $C_3$ [3], are checked to verify whether the optimization process should be terminated.

$$\begin{aligned} & (C_1 \quad \text{AND} \quad C_2) \quad \text{OR} \quad C_3 \\ & C_1: \left( \left| \frac{f(\boldsymbol{x}^{(k)}) - f(\boldsymbol{x}^{(k-1)})}{f(\boldsymbol{x}^{(k)})} \right| \le \varepsilon \middle|\middle| \left| f(\boldsymbol{x}^{(k)} - f(\boldsymbol{x}^{(k-1)}) \right| \le \varepsilon \right) \\ & C_2: \; err \le err\_max \\ & C_3: \; n_r > n_{max} \end{aligned} \tag{13}$$

**Table 1 Definitions and characteristics of some concepts**

| | Definition | Characteristics |
|---|---|---|
| Point | Pure coordinate | |
| Sample | A combination of a point and its response | |
| Real sample | A combination of a point and its real response generated by invoking expensive simulations | (1) The amount of real samples indicates the computational expense of an optimization method.<br>(2) The point in a real sample is named as real point. |
| Virtual sample | A combination of a point and its virtual response generated without calling expensive simulations | (1) The virtual response is generated by a KRG metamodel.<br>(2) The expense to generate virtual samples is negligible.<br>(3) The point in a virtual sample is named as a virtual point. |
| Cheap point [17] | A point produced by design of computer experiment in the entire design space | (1) The actual response of a cheap point is not calculated.<br>(2) The computational cost is negligible.<br>(3) The cheap points are tested by the trained LS-SVM classifier. |
| Superior cheap point [17] | A cheap point classified by LS-SVM with positive value +1 | (1) The superior cheap point has a promising optimality and feasibility performance. |

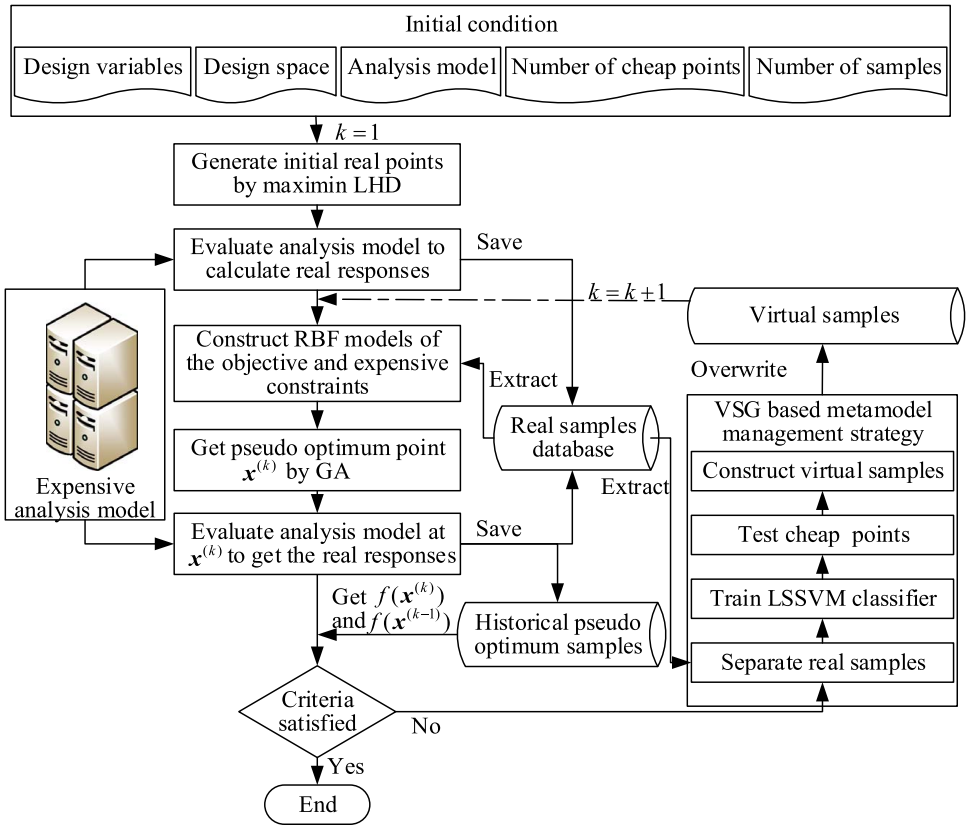


**Fig. 1 Flowchart of SRBF-VSG**

where $\varepsilon$ is the optimality tolerance; *err_max* is the approximation accuracy tolerance; $n_r$ is the number of real samples; and $n_{max}$ is the cap of $n_r$. The approximation accuracy *err* at $\boldsymbol{x}^{(k)}$ is defined as Eq. (14):

$$err = \begin{cases} \left| \dfrac{f(\boldsymbol{x}^{(k)}) - \tilde{f}(\boldsymbol{x}^{(k)})}{f(\boldsymbol{x}^{(k)})} \right|, & |f(\boldsymbol{x}^{(k)})| > 0.01 \\ |f(\boldsymbol{x}^{(k)}) - \tilde{f}(\boldsymbol{x}^{(k)})|, & |f(\boldsymbol{x}^{(k)})| \le 0.01 \end{cases} \tag{14}$$

If both $C_1$ and $C_2$ are satisfied, or $C_3$ is meet, SRBF-SVG terminates and outputs the current best solution; otherwise, the procedure jumps to *Step 6*.

*Step 6*. A VSG-based metamodel management strategy is applied to refining the RBF metamodel for the objective. Virtual samples set for the next iteration $\boldsymbol{S}_v^{(k)}$, including virtual points set $\boldsymbol{X}_v^{(k)}$ and virtual objective responses set $\boldsymbol{Y}_v^{(k)}$, is constructed. Then, the procedure returns to *Step 3* to refit the RBF metamodels. The details of VSG-based metamodel management strategy are presented in Sec. 3.2. The iteration counter is increased by one, i.e., $k = k + 1$.

**3.2 Virtual Sample Generation-Based Metamodel Management Strategy.** In the authors' previously developed SRBF-SVM [17], an interesting sampling region is identified by the SVM for sequential sampling to refine the RBF metamodels. Inspired by the machine learning enhanced metamodeling framework in the conventional SRBF-SVM, a VSG-based metamodel management strategy is developed in SRBF-VSG, where the LS-SVM classifier is used to determine the virtual samples. The procedure of this strategy exhibited in Algorithm 1 is detailed as follows:

*Step 1 (line 1)*: The constraint violation set $\boldsymbol{h}$ is calculated in terms of Eq. (15):

$$\begin{aligned} \boldsymbol{h} &= [h_1, \ldots, h_i, \ldots, h_{n_r}] \\ h_i &= \sum_{i=1}^{n} \max(0, g_i(\boldsymbol{x}_r^i)) \end{aligned} \tag{15}$$

where $g_i(\boldsymbol{x}_r^i)$ are constraint information of real points stored in $\boldsymbol{C}_r$. If the $i$-th real point $\boldsymbol{x}_r^i$ is feasible, the corresponding constraint violation $h_i = 0$; otherwise $h_i > 0$.

*Step 2 (lines 2–14)*: Real points are separated into two categories with different classification values, according to constraint violations and objective responses, as shown in Eq. (16):

$$\begin{aligned} &\hat{\boldsymbol{Y}}_r = [\hat{y}_2^1, \ldots, \hat{y}_r^i, \ldots, \hat{y}_r^{n_r}] \\ &\text{if} \quad n_f \ge 2 \\ &\hat{y}_r^i = \begin{cases} +1, & \boldsymbol{x}_r^i \in \boldsymbol{X}_{feas} \quad \text{AND} \quad y_r^i \le P_{thresh} \\ -1, & \text{otherwise} \end{cases} \\ &\text{else} \\ &\hat{y}_r^i = \begin{cases} +1, & \boldsymbol{x}_r^i \in \boldsymbol{X}_{feas} \\ -1, & \text{otherwise} \end{cases} \end{aligned} \tag{16}$$

where $\hat{\boldsymbol{Y}}_r$ is the classification value set; $n_f$ is the number of feasible real points; $\boldsymbol{x}_r^i$ is the $i$-th real point; $y_r^i$ is the actual response of $\boldsymbol{x}_r^i$; $\boldsymbol{X}_{feas}$ is the feasible point set to store feasible and quasi-feasible real points; and $P_{thresh}$ is the threshold to determine the classification value $\hat{y}_r^i$ of $\boldsymbol{x}_r^i$ and the amount of superior cheap points [17]. The corresponding objective responses of points in $\boldsymbol{X}_{feas}$ are stored in the feasible objective responses set $\boldsymbol{Y}_{feas}$. The quasi-feasible real points are not strictly feasible, but their constraint violations are the

lowest in $\boldsymbol{h}$. Both $\boldsymbol{X}_{feas}$ and $\boldsymbol{Y}_{feas}$ are set as NULL initially at each optimization iteration.

When no less than two feasible real points exist in $\boldsymbol{X}_r$ (*lines 2–9*), feasible real points and their objective responses are stored in $\boldsymbol{X}_{feas}$ and $\boldsymbol{Y}_{feas}$, respectively. Next, the threshold $P_{thresh}$ is constructed as Eq. (17) [17]:

$$P_{thresh} = \min(\boldsymbol{Y}_{feas}) + \eta_k(\max(\boldsymbol{Y}_{feas}) - \min(\boldsymbol{Y}_{feas})) \quad (17)$$

where $\eta_k$ is a shrinking coefficient at the $k$-th optimization iteration. Then, the classification values are constructed: if the $i$-th real point $\boldsymbol{x}_r^i$ is a member of $\boldsymbol{X}_{feas}$ and its corresponding response $y_r^i$ is smaller than $P_{thresh}$, its classification value $\hat{y}_r^i$ is set as $+1$; otherwise, $\hat{y}_r^i$ is set as $-1$. In this paper, if there is only one positive element in $\hat{\boldsymbol{Y}}_r$, $P_{thresh}$ is modified as Eq. (18) [17], and $\hat{\boldsymbol{Y}}_r$ is reconstructed.

$$P_{thresh} = P_{thresh} + \lambda_k |P_{thresh}| \quad (18)$$

where $\lambda_k$ is a modification coefficient at the $k$-th iteration.

**Algorithm 1** Metamodel Management Strategy

```
1  h ← GetConstraintViolation (S_r)
2  If no less than two feasible real points exist
3      construct X_feas and Y_feas, respectively
4      P_thresh = ConstructThreshold (Y_feas)
5      Ŷ_r = ∅
6      While less than two positive values exist in Ŷ_r
7          Ŷ_r = SetClassificationResults (X_r, Y_r, X_feas, P_thresh)
8          P_thresh = ModifyThreshold (P_thresh)
9      End
10 Else
11     select (n_v − n_f) quasi-feasible real points stored in X_qfeas
12     X_feas = x_feas ∪ X_qfeas
13     Ŷ_r = SetClassificationResults (X_r, Y_r, X_feas)
14 End
15 X_sup = ∅
16 While X_sup = ∅
17     Classifier ← TrainLS-SVM (X_r, Ŷ_r)
18     generate n_c cheap points and store them in X_cheap
19     Foreach x_c^i in X_cheap
20         If LS-SVMpredict (Classifier, x_c^i) = +1
21             X_sup = X_sup ∪ x_c^i
22         End
23     End
24     Modify parameters in the strategy
25 End
26 KRG ← ConstructKRG (X_r, Y_r)
27 Y_sup = KRGpredict (X_sup)
28 X_v^(k) = ∅
29 If no less than two feasible real points exist
30     Foreach x_sup^i in X_sup
31         If y_sup^i < P_thresh
32             X_v^(k) = X_v^(k) ∪ x_sup^i
33         End
34     End
35 Else
35     X_v^(k) = X_sup
37 End
38 If more than n_vs elements exist in X_v^(k)
39     X_v^(k) ← FCM (X_v^(k), n_vs)
40 End
41 Y_v^(k) = KRGpredict (X_v^(k))
42 S_v^(k) = {X_v^(k), Y_v^(k)}
43 Return S_v^(k)
```

When less than two feasible real points exist (*lines 10–14*), infeasible real points are sorted in the ascending order in terms of constraint violations. The first ($n_v$—$n_f$) infeasible real points are selected as the quasi-feasible ones stored in the set $\boldsymbol{X}_{qfeas}$. The feasible and newly generated quasi-feasible real points are used to update the set $\boldsymbol{X}_{feas}$. If the $i$-th real point $\boldsymbol{x}_r^i$ is a member of $\boldsymbol{X}_{feas}$, its classification value $\hat{y}_r^i$ is set as $+1$; otherwise, $\hat{y}_r^i$ is set as $-1$.

*Step 3 (lines 15–25)*: The superior cheap points set $\boldsymbol{X}_{sup}$ is initially set to be NULL. $n_c$ cheap points are generated randomly in the whole design space. The samples in ($\boldsymbol{X}_r$, $\hat{\boldsymbol{Y}}_r$) are employed to train a binary classifier LS-SVM to classify cheap points. If the classification value of a cheap point predicted by the trained LS-SVM classifier is positive, i.e., $+1$, this cheap point is regarded to be superior, which is believed to have lower objective and constraint violation values. This superior cheap point is added to the set $\boldsymbol{X}_{sup}$. If no superior cheap points exist, the coefficients $\eta_k$ and $\lambda_k$, whose initial values are set as 0.1 [17] at the $k$-th optimization iteration, are increased as shown in Eq. (19) to release the classification condition, and the procedure returns to *Step 2*:

$$\begin{aligned} \eta_k &= \eta_k + 0.1\eta_k \\ \lambda_k &= \lambda_k + 0.1\lambda_k \end{aligned} \quad (19)$$

*Step 4 (lines 26–28)*: An objective KRG metamodel is constructed based on real samples in the set $\boldsymbol{S}_r$. The approximated objective responses at superior cheap points are generated by evaluating the constructed KRG metamodel. The virtual points set as $\boldsymbol{X}_v^{(k)}$ for the ($k$+1)-th iteration is set as NULL initially.

*Step 5 (lines 29–37)*: When less than two feasible real points exist, $\boldsymbol{X}_v^{(k)}$ is equal to $\boldsymbol{X}_{sup}$; otherwise, if the approximate objective response $y_{sup}^i$ of the $i$-th superior cheap point $\boldsymbol{x}_{sup}^i$ is smaller than $P_{thresh}$, $\boldsymbol{x}_{sup}^i$ is considered as a virtual point, which is stored in the set $\boldsymbol{X}_v^{(k)}$. In this paper, virtual points can be considered as a subset of superior cheap points.

*Step 6 (lines 38–40)*: If more than $n_{vs}$ virtual points exist in $\boldsymbol{X}_v^{(k)}$, $n_{vs}$ cluster centers of virtual points are obtained by the fuzzy $c$-means clustering method, and $\boldsymbol{X}_v^{(k)}$ is overwritten by these cluster centers; otherwise, $\boldsymbol{X}_v^{(k)}$ does not need to be updated.

*Step 7 (lines 41–43)*: The constructed KRG metamodel in *Step 4* is employed to predict the virtual objective responses of virtual points. These responses are stored in virtual objective responses set $\boldsymbol{Y}_v^{(k)}$. Finally, the virtual samples set $\boldsymbol{S}_v^{(k)}$ is overwritten by the tuple of $\{\boldsymbol{X}_v^{(k)}, \boldsymbol{Y}_v^{(k)}\}$.

## 4 Tests on Numerical Benchmarks Problems and Discussions

In this section, some numerical benchmarks are applied to testing the global convergence, optimization efficiency, and robustness of SRBF-VSG, compared with SRBF-SVM, EGO, MSEGO, MPS, CiMPS, TR-MPS, COBRA, and ARSM-ISES. These benchmarks are considered as the complex and time-consuming problems although they have explicit function expressions.

**4.1 Description of Benchmarks and Test Procedure.** In this paper, unconstrained benchmarks represent problems without expensive constraints; constrained benchmarks represent problems with expensive constraints. Basic information of numerical

**Table 2 Tuning parameters of SRBF-VSG**

| Tuning parameters | Value |
|---|---|
| $n_i$ | $4n_v$ |
| $n_c$ | $100\,n_v^2\,d_{max}$ |
| $n_{vs}$ | $n_v$ |
| $\varepsilon$ | 0.001 |
| *err_max* | 0.0001 |

**Table 3 Optimization results comparison on unconstrained LD benchmarks**

| | | BR | SC | SE | PK | HN |
|---|---|---|---|---|---|---|
| Dimension | | 2 | 2 | 2 | 2 | 6 |
| Analytic global optimum solution | | 0.397 | −1.032 | −1.457 | −6.551 | −3.322 |
| SRBF-VSG | VR_OPT | [0.398, 0.398] | [−1.032, -1.031] | [−1.457, 2.866] | [−6.551, −3.050] | [−3.322, −3.191] |
| | MD_OPT | 0.398 | −1.032 | −1.457 | −6.551 | −3.321 |
| | VR_NFE | [20, 30] | [19, 34] | [18, 23] | [18, 23] | [49, 69] |
| | AV_NFE | 25.7 | 23 | 19.6 | 19.3 | 61.7 |
| SRBF-SVM | VR_OPT | [0.398, 0.398] | [−1.032, −1.031] | [−1.457, 2.866] | [−6.551, −3.047] | [−3.321, −3.037] |
| | MD_OPT | 0.398 | −1.032 | −1.457 | −6.549 | −3.307 |
| | VR_NFE | [20, 34] | [20, 34] | [32, 32] | [16, 34] | [56, 88] |
| | AV_NFE | 25.6 | 26.8 | 32 | 22.2 | 71.2 |
| EGO | VR_OPT | [0.398, 0.400] | [−1.032, −1.031] | [−1.456, −1.436] | [−6.550, −6.383] | [−3.316, −3.308] |
| | MD_OPT | 0.398 | −1.031 | −1.453 | −6.550 | −3.313 |
| | VR_NFE | [32, 41] | [27, 37] | [52, 52] | [26, 52] | [66, 74] |
| | AV_NFE | 36.1 | 32.6 | 52 | 42.6 | 68.8 |
| MSEGO | VR_OPT | [0.398, 0.431] | [−1.024, −0.987] | [−1.456, −1.454] | [−6.498, −5.079] | [−3.208, −3.052] |
| | MD_OPT | 0.398 | −1.024 | −1.456 | −6.498 | −3.145 |
| | VR_NFE | [36, 132] | [130, 132] | [70, 123] | [129, 132] | [176, 176] |
| | AV_NFE | 112.6 | 131.2 | 109.6 | 130.4 | 176 |
| MPS | VR_OPT | [0.398, 1.393] | [−1.032, −1.029] | [−1.457, 6.538] | [−6.551, −3.040] | [−3.322, −3.149] |
| | MD_OPT | 0.398 | −1.032 | −1.457 | −6.551 | −3.322 |
| | VR_NFE | [14, 174] | [24, 47] | [12, 71] | [20, 57] | [365, 1091] |
| | AV_NFE | 69.2 | 32.9 | 39.1 | 39.8 | 613.4 |
| ARSM-ISES | VR_OPT | [0.398, 0.399] | [−1.032, −1.030] | [−1.457, 2.866] | [−6.551, −6.550] | [−3.322, −3.193] |
| | MD_OPT | 0.398 | −1.032 | −1.457 | −6.551 | −3.322 |
| | VR_NFE | [29, 58] | [25, 38] | [22, 35] | [22, 55] | [142, 288] |
| | AV_NFE | 39.8 | 31.7 | 29.4 | 35.4 | 188.6 |

benchmarks is listed in Tables 3–5 [16,23]. More details are shown in the Appendix. If less than ten design variables exist, the problem is considered as low-dimensional (LD), otherwise HD [16].

Tuning parameters of SRBF-VSG are listed in Table 2, where $n_v$ is the dimension of design space and $d_{max}$ is the maximum range of values for all dimensions with two decimal places. In this paper, $n_{max}$ is set according to competitive MBDO methods; the GA optimizer integrated in MATLAB is utilized to perform the global optimization on the RBF metamodels; the *rand*(.) function in MATLAB is used to generate cheap points; and the fuzzy *c*-means cluster method in MATALB is applied to obtaining the cluster centers.

To reduce influences of randomness, each numerical benchmark is tested for ten runs [16,17]. The variation ranges of optimum (i.e., VR_OPT) and the number of function evaluations (i.e., VR_NFE) are recorded to indicate the optimization robustness. Besides, the average number of function evaluations (AV_NFE) reflects the optimization efficiency. The median optimum (MD_OPT) indicates the global convergence.

**Table 4 Optimization results comparison on unconstrained HD benchmarks**

| | | R10 | F16 | HD1 |
|---|---|---|---|---|
| Dimension | | 10 | 16 | 10 |
| Analytic global optimum solution | | 0.000 | 25.875 | 0.000 |
| SRBF-VSG | VR_OPT | [0.182, 7.772] | [25.875, 25.876] | [0.339, 0.627] |
| | MD_OPT | 4.793 | 25.876 | 0.538 |
| | VR_NFE | [1100, 1100] | [130, 130] | [800, 800] |
| | AV_NFE | 1100 | 130 | 800 |
| SRBF-SVM | VR_OPT | [18.307, 89.255] | [26.141, 27.452] | [0.506, 1.091] |
| | MD_OPT | 42.301 | 26.862 | 0.546 |
| | VR_NFE | [1100, 1100] | [130, 130] | [800, 800] |
| | AV_NFE | 1100 | 130 | 800 |
| TR-MPS | VR_OPT | N/A | N/A | N/A |
| | MD_OPT | 9.570 | 25.912 | 2.019 |
| | VR_NFE | N/A | N/A | N/A |
| | AV_NFE | 6483.3 | 726.3 | 7137.0 |
| MPS | VR_OPT | [70.057, 272.384] | [29.387, 30.615] | [3.326, 5.854] |
| | MD_OPT | 29.387 | 29.387 | 3.326 |
| | VR_NFE | [4198, 4204] | [916, 931] | [2007, 2012] |
| | AV_NFE | 4200.6 | 921.0 | 2008.8 |
| ARSM-ISES | VR_OPT | [2.714, 66.174] | [25.875, 25.887] | [0.505, 0.557] |
| | MD_OPT | 3.147 | 25.875 | 0.519 |
| | VR_NFE | [1023, 4197] | [462, 916] | [802, 2006] |
| | AV_NFE | 2638.0 | 661.0 | 1408.6 |

**Table 5 Optimization results comparison on constrained numerical benchmarks**

| | | G6 | G4 | G7 |
|---|---|---|---|---|
| Dimension | | 2 | 5 | 10 |
| Analytic global optimum solution | | −6961.8139 | −30665.539 | 24.3062 |
| SRBF-VSG | VR_OPT | [−6961.2, −5647.3] | [−30492.5, −29811.1] | [41.70, 129.02] |
| | MD_OPT | −6945.9 | −30233.1 | 98.75 |
| | VR_NFE | [21, 36] | [38, 56] | [187, 187] |
| | AV_NFE | 25.1 | 50.6 | 187 |
| SRBF-SVM | VR_OPT | [−6961.9, −6961.5] | [−30097.8, −29018.0] | [24.45, 1540.05] |
| | MD_OPT | −6961.8 | −29359.9 | 30.20 |
| | VR_NFE | [16, 18] | [29, 45] | [110, 442] |
| | AV_NFE | 17.4 | 34.2 | 187.4 |
| COBRA local | VR_OPT | [−6944.5, −6450.5] | [−30665.5, −30664.6] | [24.48, 29.33] |
| | MD_OPT | −6834.5 | −30665.2 | 25.32 |
| | VR_NFE | [53, 53] | [56, 56] | [61, 61] |
| | AV_NFE | 53 | 56 | 61 |
| COBRA global | VR_OPT | [−6929.7, −6341.1] | [−30664.9, −30664.9] | [24.52, 28.60] |
| | MD_OPT | −6799.05 | −30664.9 | 25.28 |
| | VR_NFE | [53, 53] | [56, 56] | [61, 61] |
| | AV_NFE | 53 | 56 | 61 |
| CiMPS | VR_OPT | [−6961.8, −6961.8] | [−30665.5, −30665.5] | N/A |
| | MD_OPT | −6961.8 | −30665.5 | N/A |
| | VR_NFE | [104 (>1E4), 104 (1E4)] | [108 (425), 116 (475)] | N/A |
| | AV_NFE | 104 (1E4) | 109.8 (460.2) | N/A |

For SRBF-VSG, SRBF-SVM, and COBRA, one function evaluation produces the values of the objective and all constraints at once; for CiMPS, the number of constraint function evaluation is presented in the parenthesis.

### 4.2 Test Results

*4.2.1 Unconstrained Low-Dimensional Benchmarks.* In this part, some unconstrained LD benchmarks are used to compare the capability of SRBF-VSG with SRBF-SVM, EGO, MSEGO, MPS, and ARSM-ISES. The comparison results are summarized in Table 3. Note that the results of competitive MBDO methods are directly cited from Refs. [16] and [17].

The best global optima in Table 3 show that SRBF-VSG, SRBF-SVM, MPS, and ARSM-ISES can successfully capture the true global optimum for all the numerical problems. Their median optima are all equal or close to analytic global optima. Results in Table 3 also indicate that compared with SRBF-SVM, MPS, and ARSM-ISES, the NFE consumed by SRBF-VSG is lower and has much smaller variation range, which demonstrates the merits of SRBF-VSG in optimization efficiency and robustness. Besides, note that both EGO and MSEGO fail to find the analytic optima for some benchmarks, e.g., EGO fails in HN, and MSEGO fails in PK and HN. For the rest benchmarks, EGO and MSEGO generally consume larger NFE to find inferior solutions, compared with SRBF-VSG, SRBF-SVM, MPS, and ARSM-ISES.

To intuitively illustrate the performance difference between the proposed SRBF-VSG and the conventional SRBF-SVM, the boxplots of optima obtained by the two competitive methods are compared in Fig. 2. The boxplots also show that SRBF-VSG generally outperforms the conventional SRBF-SVM in terms of global convergence, which demonstrates the effectiveness of virtual sample generation.

In conclusion, the aforementioned comparison results indicate that SRBF-VSG can converge to the analytical global optima with a higher efficiency for unconstrained LD problems, compared with SRBF-SVM, MPS, ARSM-ISES, EGO, and MSEGO.

*4.2.2 Unconstrained High-Dimensional Benchmarks.* In this part, three unconstrained HD benchmarks are utilized to test the capability of SRBF-VSG. Considering that EGO and MSEGO are inefficient for solving HD problems [16], SRBF-VSG is only compared with SRBF-SVM, MPS, TR-MPS, and ARSM-ISES in this study. Comparison results are summarized in Table 4. Since the variation ranges of solutions obtained by TR-MPS are not available in Ref. [16], they are labeled as N/A in Table 4. Note that the results of MPS, TR-MPS, and ARSM-ISES are cited directly from Ref. [16]. SRBF-VSG and SRBF-SVM are operated with the same maximum NFE for each benchmark.

The comparison results in Table 4 show that the best solutions obtained by SRBF-VSG are closer to analytical optima than those obtained by SRBF-SVM, TR-MPS, MPS, and ARSM-IESE. Although the median optima obtained by SRBF-VSG are comparable with those obtained by ARSM-ISES, they are much better than those obtained by SRBF-VSG, TR-MPS, and MPS. Besides, SRBF-VSG has the fewest mean NFE values for R10, F10, and

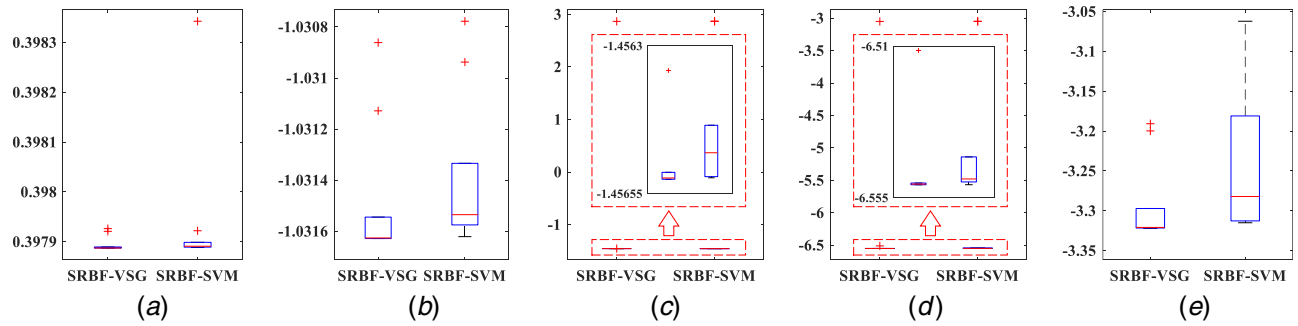


**Fig. 2 Boxplots of optima for unconstrained LD benchmarks: (*a*) BR, (*b*) SC, (c) SE, (*d*) PK, and (e) HN**

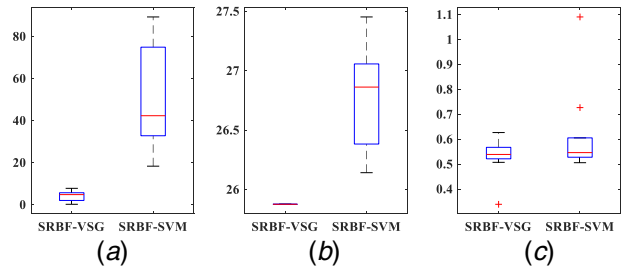


**Fig. 3 Boxplots of optima for unconstrained HD benchmarks: (*a*) R10, (*b*) F16, and (*c*) HD1**

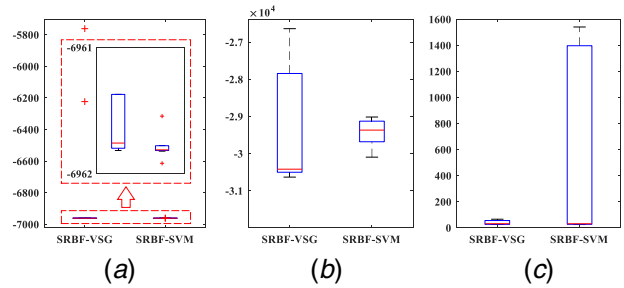


**Fig. 4 Boxplots of optima for constrained benchmarks: (*a*) G6, (*b*) G4, and (*c*) G7**

HD1 problems. Moreover, the boxplots of optima in Fig. 3 also indicate that the proposed SRBF-VSG performs better than SRBF-SVM for solving the unconstrained HD benchmarks.

Aforementioned comparative studies demonstrate that SRBF-VSG has a comparable or better global convergence, and a better optimization efficiency for unconstrained HD problems, compared with SRBF-SVM, TR-MPS, MPS, and ARSM-IESE.

*4.2.3 Constrained Numerical Benchmarks.* Since standard EGO, MSEGO, and MPS cannot handle expensive constrained optimization problems [17], SRBF-VSG is compared with SRBF-SVM, COBRA (COBRA global and COBRA local). Comparison results in Table 5 indicate that COBRA has the best performance on G7 in terms of efficiency and global convergence. Besides, SRBF-VSG, SRBF-SVM, and CiMPS can successfully capture the analytic optimum and produce comparable median optima for G6 benchmark. However, SRBF-VSG and SRBF-SVM consume much fewer function evaluations than CiMPS. For G4, the best and the median optima obtained by SRBF-VSG are only about 0.56% and 1.41% higher, respectively, than those obtained by COBRA and CiMPS. But the mean function evaluations required by SRBF-VSG is about 90% of that required by COBRA, and only about 17.75% of that required by CiMPS. In addition, the boxplots of the optima from SRBF-VSG and SRBF-SVM for constrained benchmarks are displayed in Fig. 4.

The comparison results prove that compared with SRBF-SVM, SRBF-VSG has a similar global convergence and a better efficiency in G4 and G7. SRBF-VSG outperforms COBRA and CiMPS on LD constrained problems with a better efficiency and a comparable global convergence, whereas COBRA has advantages on HD constrained problems and CiMPS is not suitable to solve problems with lots of variables and constraints.

**4.3 Discussion.** In this paper, the idea of virtual samples is introduced for metamodel-based optimization, which distinguishes SRBF-VSG from the existing MBDO methods. In this section, the SC problem in Eq. (A2) is solved for ten runs to further illustrate the benefits of KRG-based virtual sampling. In this work, three different cases of the proposed method are discussed. In Case-I, the original SRBF-VSG in this paper is used for optimization; in Case-II, the virtual samples in SRBF-VSG are replaced with real samples

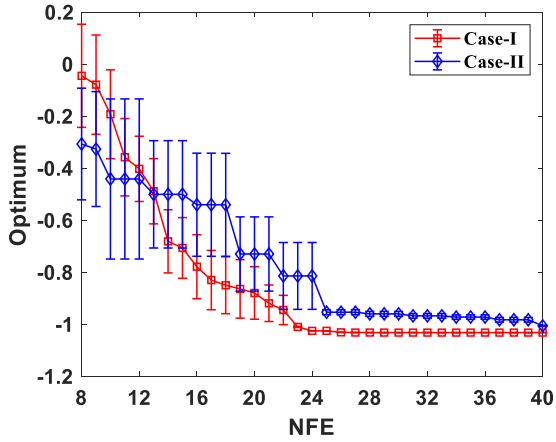


**Fig. 5 Error bars of optima in Case-I and Case-II**

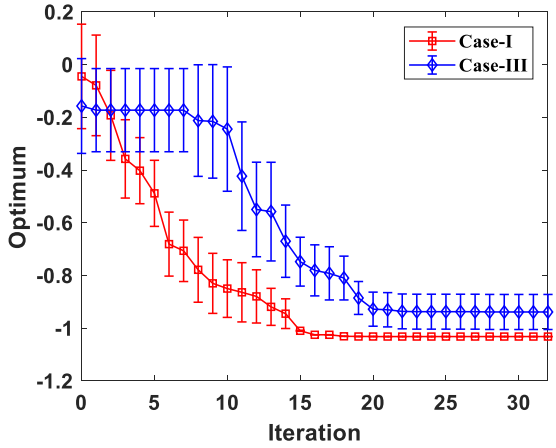


**Fig. 6 Error bars of optima in Case-I and Case-III**

for optimization; and in Case-III, a variant of SRBF-VSG is implemented to solve the optimization problem where the virtual samples are generated by RBF instead of KRG. The optimization results are shown in Figs. 5 and 6.

As shown as Fig. 5, Case-I obtains a better optimum than Case-II after 40 function calls. Moreover, Case-I using virtual samples significantly convergences faster than Case-II within 24 function calls. It indicates that the nonlinear characteristic of the black-box

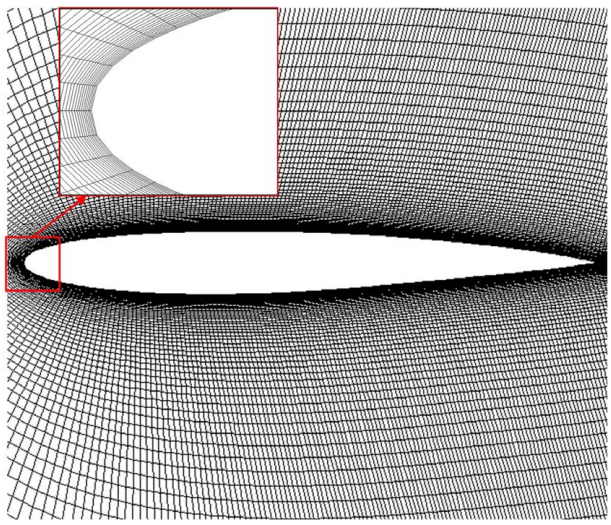

**Fig. 7 C-H type structural grids**

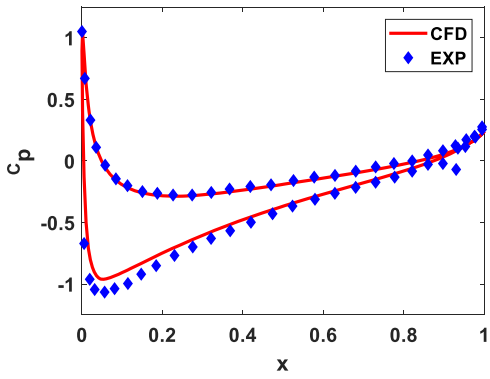


**Fig. 8 Comparison of *Cp***

function can be further mined by the virtual samples to enhance the approximation performance of metamodels, which improves the global exploration and local exploitation ability of the optimization method.

The error bar in Fig. 6 shows that Case-I takes fewer iterations (i.e., 18) to converge to a better optimum compared with Case-III, which indicates that the virtual samples from KRG are more effective than those from RBF for optimization. This can be attributed to the promising approximation accuracy of KRG with limited sparse samples.

## 5 Engineering Optimization Problems

In this section, SRBF-VSG is employed to solve an airfoil aerodynamic optimization problem and a small Earth observation satellite multidisciplinary design optimization (MDO) problem, which are presented as follows.

**5.1 Airfoil Aerodynamic Optimization Problem.** NACA0012 is chosen as the baseline airfoil, whose geometry is parametrized by the shape function method [48]. High-fidelity CFD model is applied to evaluate the aerodynamic performance. The flow field around the airfoil is simulated by the Navier–Stokes equation incorporated with Spalart–Allmaras turbulence model; and FLUENT software is employed as the CFD solver [15]. C-H type structural grids (shown in Fig. 7) are used to improve the simulation accuracy and efficiency. As shown in Fig. 8, the coefficient of pressure distribution ($C_p$) obtained by CFD simulations under the flight condition (i.e., 0.6 Mach number and 2 deg angle of attack) agrees well with the wind-tunnel experimental data, which indicates that the CFD model used in this paper provides acceptable accuracy for this optimization problem.

The purpose of this optimization problem is to maximize the lift-to-drag ratio via consecutively modifying the baseline airfoil under the given flight condition, i.e., 0.8 Mach number and 2.5 deg angle of attack. This aerodynamic optimization problem can be described as follows:

$$\begin{aligned} \max \quad & L/D = C_L/C_D \\ \text{s.t.} \quad & C_L \geq C_L^{basic} \\ & t_{max} \geq 0.8 t_{max}^{basic} \\ & -0.005 \leq x_{1,2,\ldots,i,\ldots,10} \leq 0.005 \end{aligned} \tag{20}$$

**Table 6 Comparison for solution results**

| | *L/D* | | |
|---|---|---|---|
| Optimization method | Optimum range | Mean | NFE |
| SRBF-VSG | [16.3,17.0] | 16.6 | 50 |
| SRBF-SVM | [15.4,16.6] | 15.9 | 80 |
| NLPQL | [14.3,17.1] | 16.7 | 120 |
| MIGA | [15.2,16.4] | 15.8 | 500 |

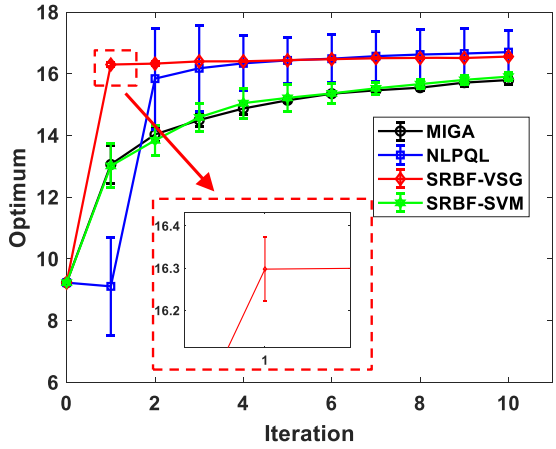


**Fig. 9 Error bars of optima obtained by different methods**

**Table 7 Comparison of baseline and optimized airfoils**

| Parameter | Baseline | Optimal | Change (%) |
|---|---|---|---|
| $C_L$ | 0.381 | 0.5211 | +36.77 |
| $C_D$ | 0.0413 | 0.0306 | −25.72 |
| $t_{max}$ | 0.1043 | 0.0969 | −7.09 |
| *L/D* | 9.232 | 17.014 | +84.29 |

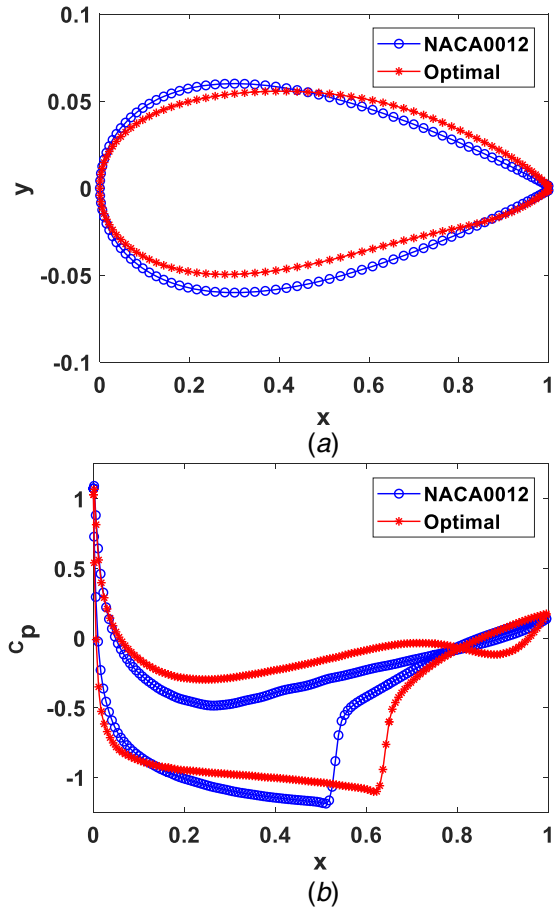


**Fig. 10 Comparison of baseline and optimal airfoils: (*a*) geometry and (*b*) Cp**

**Table 8 Optimal design variables of the satellite MDO problem**

| Design variables | Sign (unit) | Range | Optimum |
|---|---|---|---|
| Aperture of radiometer | $D_1$ (mm) | [120, 280] | 217.58 |
| Aperture of CCD camera | $D_2$ (mm) | [5, 15] | 7.69 |
| Focal length of radiometer | $F_1$ (mm) | [400, 900] | 724.25 |
| Focal length of CCD camera | $F_2$ (mm) | [10, 50] | 37.47 |
| Area of solar arrays | $A_{sa}$ (mm) | [3, 10] | 3.04 |
| Capability of storage battery | $C_s$ (AH) | [20, 100] | 20.63 |
| Thickness of joint ring | $T_R$ (mm) | [10.5, 19.5] | 14.7 |
| Honeycomb core thickness of structure plates | $T_{BH}$ (mm) | [7.0, 13.0] | 8.9 |
| Skin thickness of the structure plates | $T_{SH}$ (mm) | [0.07, 0.13] | 0.076 |

**Table 9 Constraint information of the optimized satellite**

| Constraints | Sign | Range | Value |
|---|---|---|---|
| Signal-to-noise ration of ocean color scanner | $SNR_1$ | ≥300 | 416.9 |
| Signal-to-noise ration of CCD camera | $SNR_2$ | ≥500 | 800.8 |
| Resolution of ocean color scanner | $R_1$ | ≤1100 m | 1038.2 |
| Resolution of CCD camera | $R_2$ | ≤250 m | 225.4 |
| Noise equivalent temperature difference | $NE\Delta R$ | ≤0.2 K | 0.2 |
| Surplus power of satellite | $g_w$ | ≥0 W | 0.9 |
| Discharge depth of battery | $DOD$ | ≤0.3 | 0.16 |
| Natural frequency in the X coordinate | $f_x$ | ≥20 Hz | 22.1 |
| Natural frequency in the Y coordinate | $f_y$ | ≥20 Hz | 23.5 |

where $L/D$ is the lift-to-drag ratio; $C_L$ is the lift coefficient; $C_D$ is the drag coefficient; $C_L^{basic}$ is the lift coefficient of the baseline airfoil; $t_{max}$ and $t_{max}^{basic}$ are the maximum thickness of the optimized and baseline airfoil, respectively; and $x_i$ is the coefficient of shape function.

The optimization results of SRBF-VSG are compared with those obtained from nonlinear programming by SRBF-SVM, the quadratic Lagrangian (NLPQL) method [49], and the multiisland genetic algorithm (MIGA) method [50]. The optimization problem is solved for ten runs by SRBF-VSG, SRBF-SVM, NLPQL, and MIGA, respectively. The max iterations of NLPQL is set as 10; the subpopulation size of MIGA is set as 50; the number of islands of MIGA is set as 1; the number of generations is set as 10; and other parameters of NLPQL and MIGA are set as default.

The solution results are summarized in Table 6. Although the mean optimal $L/D$ obtained by SRBF-VSG is about 0.6% lower than that obtained by NLPQL, SRBF-VSG only takes about 41.7% computational expense of that consumed by NLPQL. Besides, SRBF-VSG costs only 10% computational budget of MIGA and captures a mean optimal $L/D$ 5.1% higher than that obtained by MIGA. Moreover, the required NFE of SRBF-VSG is 37.5% less than that of SRBF-SVM, while the mean optimal $L/D$ of SRBF-VSG is about 4.4% higher than that of SRBF-SVM. By comparing error bars of SRBF-VSG and NLPQL in Fig. 9, SRBF-VSG can converge to a comparable optimal $L/D$ value within fewer iterations, which indicates the better convergence property of SRBF-VSG. Moreover, the variance of SRBF-VSG is generally smaller than that of SRBF-SVM, MIGA, and NLPQL, which demonstrates the stronger robustness of SRBF-VSG. Thus, the aforementioned comparative studies prove that the proposed SRBF-VSG outperforms SRBF-SVM, NLPQL, and MIGA methods for this problem in terms of efficiency and global convergence.

Table 7 lists the comparison results of the baseline airfoil and the optimal airfoil obtained by SRBF-VSG. The lift coefficient and the lift-to-drag ratio are improved by 36.77% and 84.29%, respectively. The drag coefficient and maximum thickness are reduced by 25.72% and 7.09%, respectively. The geometry and pressure distribution

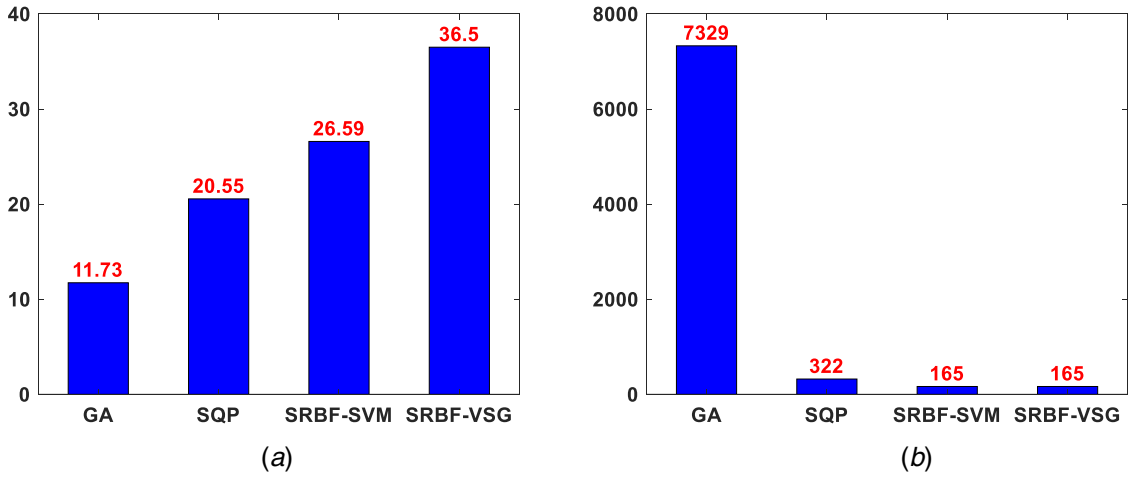


**Fig. 11 Comparison of results obtained by different methods on the satellite MDO problem: (*a*) total mass reduction and (*b*) NFE**

comparison of the baseline and the optimal airfoil are exhibited in Figs. 10(*a*) and 10(*b*). The effective angle of attack is increased in the optimal airfoil, which results in improvement of *L*/*D*.

**5.2 Earth Observation Satellite Multidisciplinary Design Optimization Problem.** An earth observation satellite MDO problem in Ref. [17] is also investigated to further demonstrate the performance of SRBF-VSG. The optimization results of SRBF-VSG are compared with those from GA, SQP, and SRBF-SVM presented in Ref. [17]. The max number of function evaluations of both SRBF-VSG and SRBF-SVM are set as 165. The obtained optimum reflects the methods' convergence, whereas the number of function evaluations reflects the optimization efficiency. The satellite MDO problem aims to minimize the total mass of the satellite, as formulated in Eq. (21) [17]

$$\begin{aligned} &\min \quad M_{satellite} = m_{payload} + m_{bat} + m_{solar} + m_{str} + m_{other} \\ &\text{where} \quad X = [D_1, D_2, F_1, F_2, A_{sa}, C_s, T_R, T_{BH}, T_{SH}]^T \\ &\text{s.t.} \quad \begin{cases} SNR_1 \geq 300,\ SNR_2 \geq 500,\ R_1 \leq 1100\,\text{m},\ R_2 \leq 250\,\text{m} \\ NE\Delta T \leq 0.2\,\text{K},\ g_w \geq 0,\ DOD \leq 25\% \\ f_x \geq 20\,\text{Hz},\ f_y \geq 20\,\text{Hz} \end{cases} \end{aligned} \tag{21}$$

where the information of design variables and constraints [17,51] are listed in Tables 8 and 9, respectively. The total mass of the satellite $M_{satellite}$ consists of the payload mass $m_{payload}$, the battery mass $m_{bat}$, the solar arrays mass $m_{solar}$, the structure mass $m_{str}$, and the fixed mass $m_{other} = 198$ kg. Details of the variables, constraints, and disciplinary analysis models are discussed in Ref. [51], and more information about the optimization problem can be found in Ref. [17].

The optimization results obtained by SRBF-VSG are summarized in Table 8. The constraints at the optimum shown in Table 9 indicate that SRBF-VSG successfully captures a feasible optimum. The reduced total mass and required NFE are exhibited in Figs. 11(*a*) and 11(*b*), respectively. Compared with the initial total mass of the satellite (i.e., 366.7 kg [17]), the total mass obtained by SRBF-VSG reduces 36.5 kg. Compared with GA, SQP, and SRBF-SVM, the mass reduction of SRBF-VSG is more obvious, as detailed in Fig. 11(*a*). Besides, the number of function evaluations required by SRBF-VSG is only 2.25% and 51.24% of that required by GA and SQP, respectively.

The comparative studies on two aforementioned real-world engineering problems demonstrate that the proposed SRBF-VSG method outperforms the competitive methods in terms of global convergence, optimization efficiency, and robustness. The effectiveness and practicality of SRBF-VSG in solving real-world engineering optimization problems have also been demonstrated.

## 6 Conclusions and Future Work

In this paper, a novel sequential radial basis function using virtual sample generation (SRBF-VSG) method is proposed to further improve the optimization efficiency of MBDO. The proposed method centers on a VSG-based metamodel management strategy. In this strategy, a LS-SVM classifier is trained based on real samples' optimality and feasibility. Then, virtual samples are constructed including virtual points and virtual responses. Virtual points are generated from the superior cheap points obtained by the trained classifier, and virtual responses are generated by calculating a KRG metamodel of the objective. The virtual samples are combined with the pseudo optimum of each iteration to update the objective RBF metamodel, which significantly reduces the budget of updating metamodels. Several numerical benchmarks are used to test and validate the overall performance of the proposed SRBF-VSG. The comparison results demonstrate that in terms of global convergence, efficiency, and robustness, SRBF-VSG generally outperforms SRBF-SVM, EGO, MSEGO, MPS, CiMPS, TR-MPS, COBRA, and ARSM-ISES. The capability of solving real-world engineering design optimization problems of SRBF-VSG is demonstrated by an airfoil aerodynamic optimization problem and an earth observation satellite MDO problem. The results of comparative studies show that the proposed method has a promising effectiveness and practicality in solving expensive optimization problems.

In future work, some further researches are expected to be investigated as follows:

(1) The idea of virtual samples is expected to be incorporated with more MBDO methods to test its effectiveness.
(2) Other VSG mechanisms such as the mega-trend-diffusion technique will be investigated to further improve the quality of virtual samples.
(3) The proposed SRBF-VSG in this paper is mainly developed for solving middle- and low-dimensional optimization problems. In the future, some novel techniques such as high-dimensional model representation (HDMR) and manifold learning are expected to be combined with virtual samples to tackle the high-dimensional optimization problems.


## Acknowledgment

This work is sponsored by the National Natural Science Foundation of China (Grant No. 51675047), and Chinese Postdoctoral Science Foundation (Grant No. 2019M660668), and the Graduate Technological Innovation Project of Beijing Institute of Technology, China (Grant No. 2018CX10001). The authors appreciate the online SURROGATES and LSSVMlab toolkits from F. A. C. Viana and K. D. Brabanter, respectively.


## Appendix

Branin function (BR)

$$f(\boldsymbol{x}) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{\pi}{5}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 \tag{A1}$$
$$x_1 \in [-5, 10], \quad x_2 \in [0, 15]$$

Six-hump camelback function (SC)

$$f(\boldsymbol{x}) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4 \tag{A2}$$
$$x_{1,2} \in [-2, 2]$$

Sasena function (SE)

$$\begin{aligned} f(\boldsymbol{x}) = {} & 2 + 0.01(x_2 - x_1^2)^2 + (1 - x_1)^2 \\ & + 2(2 - x_2)^2 + 7\sin(0.5x_1)\sin(0.7x_1x_2) \end{aligned} \tag{A3}$$
$$x_{1,2} \in [0, 5]$$

Peaks function (PK)

$$\begin{aligned} f(\boldsymbol{x}) = {} & 3(1 - x_1)^2\exp(-x_1^2) - (1 + x_2)^2 \\ & - 10\left(\frac{x_1}{5} - x_1^3 - x_2^5\right)\exp(-x_1^2 - x_2^2) \\ & - \frac{1}{3}\exp((-x_1 + 1)^2 - x_2^2) \end{aligned} \tag{A4}$$
$$x_{1,2} \in [-3, 3]$$

Hartman function (HN)

$$f(\boldsymbol{x}) = \sum_{i=1}^{4} c_i \exp\left[\sum_{j=1}^{n} \alpha_{ij}(x_j - p_{ij})^2\right], i = 1, 2, \ldots, 6 \tag{A5}$$
$$x_{1,2,\ldots,6} \in [0, 1]$$

where $\alpha_{ij}$, $c_i$, and $p_{ij}$ are described in Ref. [17].

10D SUR-TI-14 function (HD1)

$$f(\boldsymbol{x}) = (x_1 - 1)^2 + (x_{10} - 1)^2 + 10\sum_{i=1}^{10}(10 - i)(x_i^2 - x_{i+1})^2 \qquad x_{1,\ldots,10} \in [-3, 2] \tag{A6}$$

10D Rosenbrock function (R10)

$$f(\boldsymbol{x}) = \sum_{i=1}^{9}(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2) \qquad x_{1,\ldots,10} \in [-5, 5] \tag{A7}$$

Sixteen-variable function (F16)

$$f(\boldsymbol{x}) = \sum_{i=1}^{16}\sum_{j=1}^{16} a_{ij}(x_i^2 + x_i + 1)(x_j^2 + x_j + 1) \qquad x_i \in [-1, 1], \quad i = 1, 2, \ldots, 16 \tag{A8}$$

where $a_{ij}$ is described in Ref. [17].

G4

$$\begin{aligned}
\min \quad & f(\boldsymbol{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 \\
& \quad +37.293239x_1 - 40792.14175 \\
\text{s.t.} \quad & u = 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 \\
& \quad - 0.0022053x_3x_5 \\
& v = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 \\
& \quad + 0.0021813x_3^2 \\
& w = 9.300964 + 0.0047026x_3x_5 + 0.0012547x_1x_3 \\
& \quad + 0.0019085x_3x_4 \\
& g_1(\boldsymbol{x}) = -u \le 0; \qquad g_2(\boldsymbol{x}) = u - 92 \le 0; \\
& g_3(\boldsymbol{x}) = -v + 90 \le 0; \quad g_4(\boldsymbol{x}) = v - 100 \le 0; \\
& g_5(\boldsymbol{x}) = -w + 20 \le 0; \quad g_6(\boldsymbol{x}) = w - 25 \le 0 \\
& 78 \le x_1 \le 102, 33 \le x_2 \le 45, 27 \le x_i \le 45, \quad i = 3, 4, 5
\end{aligned} \tag{A9}$$

G7

$$\begin{aligned}
\min \quad & f(\boldsymbol{x}) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 \\
& \quad + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 \\
& \quad + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45 \\
\text{s.t.} \quad & g_1(\boldsymbol{x}) = (4x_1 + 5x_2 - 3x_7 + 9x_8 - 105)/105 \le 0 \\
& g_2(\boldsymbol{x}) = (10x_1 - 8x_2 - 17x_7 + 2x_8)/370 \le 0 \\
& g_3(\boldsymbol{x}) = (-8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12)/158 \le 0 \\
& g_4(\boldsymbol{x}) = (3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120)/1258 \le 0 \\
& g_5(\boldsymbol{x}) = (5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40)/816 \le 0 \\
& g_6(\boldsymbol{x}) = (0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30)/834 \le 0 \\
& g_7(\boldsymbol{x}) = (x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6)/788 \le 0 \\
& g_8(\boldsymbol{x}) = (-3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10})/4080 \le 0 \\
& \quad -10 \le x_i \le 10, \quad i = 1, 2, \ldots, 10
\end{aligned} \tag{A10}$$

G6

$$\begin{aligned}
\min \quad & f(\boldsymbol{x}) = (x_1 - 10)^3 + (x_2 - 20)^3 \\
& x_1 \in [13, 100] \quad x_2 \in [0, 100] \\
\text{s.t.} \quad & \begin{cases} g_1(\boldsymbol{x}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0 \\ g_2(\boldsymbol{x}) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0 \end{cases}
\end{aligned} \tag{A11}$$

## References


[1] Wang, G. G., and Shan, S., 2007, "Review of Metamodeling Techniques in Support of Engineering Design Optimization," ASME J. Mech. Des., **129**(4), pp. 370–380.

[2] Duan, X., Wang, G. G., Kang, X., Niu, Q., Naterer, G., and Peng, Q., 2009, "Performance Study of Mode-Pursuing Sampling Method," Eng. Optimiz., **41**(1), pp. 1–21.

[3] Long, T., Liu, J., Wang, G. G., Liu, L., Shi, R. H., and Guo, X. S., 2016, "Discuss on Approximate Optimization Strategies Using Design of Computer Experiments and Metamodels for Flight Vehicle Design," Chin. J. Mech. Eng-En., **52**(14), pp. 79–105.

[4] Jin, R., Chen, W., and Simpson, T. W., 2001, "Comparative Studies of Metamodelling Techniques Under Multiple Modelling Criteria," Struct. Multidiscip. O., **23**(1), pp. 1–13.

[5] Cressie, N., 1990, "The Origins of Kriging," Math Geol, **22**(3), pp. 239–252.

[6] Buhmann, M. D., 2003, *Radial Basis Functions: Theory and Implementations* (Process of Cambridge Monographs on Applied and Computational Mathematics), Cambrige University, Cambridge, UK.

[7] Jones, D. R., Schonlau, M., and Welch, W. J., 1998, "Efficient Global Optimization of Expensive Black-Box Functions," J. Global Optim., **13**(4), pp. 455–492.

[8] Viana, F. A., Haftka, R. T., and Watson, L. T., 2013, "Efficient Global Optimization Algorithm Assisted by Multiple Surrogate Techniques," J. Global Optimiz., **56**(2), pp. 669–689.

[9] Sasena, M. J., Parkinson, M., Reed, M. P., Papalambros, P. Y., and Goovaerts, P., 2005, "Improving an Ergonomics Testing Procedure Via Approximation-Based Adaptive Experimental Design," ASME J. Mech. Des., **127**(5), pp. 1006–1013.

[10] Wang, L., Shan, S., and Wang, G. G., 2004, "Mode-Pursuing Sampling Method for Global Optimization on Expensive Black-Box Functions," Eng. Optimiz., **36**(4), pp. 419–438.

[11] Cheng, G. H., Younis, A., Hajikolaei, K. H., and Wang, G. G., 2015, "Trust Region Based Mode Pursuing Sampling Method for Global Optimization of High Dimensional Design Problems," ASME J. Mech. Des., **137**(2), p. 021407.

[12] Kazemi, M., Wang, G. G., Rahnamayan, S., and Gupta, K., 2011, "Metamodel-Based Optimization for Problems With Expensive Objective and Constraint Functions," ASME J. Mech. Des., **133**(1), p. 014505.

[13] Cai, X., Qiu, H., Gao, L., Yang, P., and Shao, X., 2017, "A Multi-Point Sampling Method Based on Kriging for Global Optimization," Struct. Multidiscip. O., **56**(1), pp. 71–88.

[14] Wang, G. G., Dong, Z., and Aitchison, P., 2001, "Adaptive Response Surface Method—A Global Optimization Scheme for Approximation-Based Design Problems," Eng. Optimiz., **33**(6), pp. 707–734.

[15] Long, T., Li, X., Shi, R., Liu, J., Guo, X., and Liu, L., 2017, "Gradient-Free Trust-Region-Based Adaptive Response Surface Method for Expensive Aircraft Optimization," AIAA J., **56**(2), pp. 862–873.

[16] Long, T., Wu, D., Guo, X. S., Wang, G. G., and Liu, L., 2015, "Efficient Adaptive Response Surface Method Using Intelligent Space Exploration Strategy," Struct. Multidiscip. O., **51**(6), pp. 1335–1362.

[17] Shi, R., Liu, L., Long, T., and Liu, J., 2016, "Sequential Radial Basis Function Using Support Vector Machine for Expensive Design Optimization," AIAA J., **55**(1), pp. 214–227.

[18] Wu, D., Hajikolaei, K. H., and Wang, G. G., 2018, "Employing Partial Metamodels for Optimization With Scarce Samples," Struct. Multidiscip. O., **57**(3), pp. 1329–1343.

[19] Rouhi, M., Ghayoor, H., Hoa, S. V., and Hojjati, M., 2017, "Computational Efficiency and Accuracy of Multi-Step Design Optimization Method for Variable Stiffness Composite Structures," Thin Wall. Struct., **113**, pp. 136–143.

[20] Li, E., Wang, H., and Ye, F., 2016, "Two-Level Multi-Surrogate Assisted Optimization Method for High Dimensional Nonlinear Problems," Appl. Soft Comput., **46**, pp. 26–36.

[21] Ran, R., Li, W., and Li, Y., 2018, "A Two-Level Global Optimization Method Based on Hybrid Metamodel for Expensive Problems," Adv. Mech. Eng., **10**(4), pp. 1–14.

[22] Regis, R. G., 2011, "Stochastic Radial Basis Function Algorithms for Large-Scale Optimization Involving Expensive Black-Box Objective and Constraint Functions," Comput. Oper. Res., **38**(5), pp. 837–853.

[23] Regis, R. G., 2014, "Constrained Optimization by Radial Basis Function Interpolation for High-Dimensional Expensive Black-Box Problems With Infeasible Initial Points," Eng. Optimiz., **46**(2), pp. 218–243.

[24] Li, Y., Wu, Y., Zhao, J., and Chen, L., 2017, "A Kriging-Based Constrained Global Optimization Algorithm for Expensive Black-Box Functions With Infeasible Initial Points," J. Global Optim., **67**(1), pp. 343–366.

[25] Shi, R., Liu, L., Long, T., Wu, Y., and Tang, Y., 2019, "Filter-based Sequential Radial Basis Function Method for Spacecraft Multidisciplinary Design Optimization," AIAA J., **57**(3), pp. 1019–1031.

[26] Shi, R., Liu, L., Long, T., Wu, Y., and Tang, Y., 2019, "Filter-Based Adaptive Kriging Method for Black-Box Optimization Problems With Expensive Objective and Constraints," Comput. Method Appl. M., **347**, pp. 782–805.

[27] Forrester, A. I. J., Sóbester, A., and Keane, A. J., 2007, "Multi-Fidelity Optimization Via Surrogate Modelling," Proc. Roy. Soc. A Math. Phys. Eng. Sci., **463**(2088), pp. 3251–3269.

[28] Xiong, Y., Chen, W., and Tsui, K.-L., 2008, "A New Variable-Fidelity Optimization Framework Based on Model Fusion and Objective-Oriented Sequential Sampling," ASME J. Mech. Des., **130**(11), p. 111401.

[29] Hanyu, Y., Clemens, M., and Seifert, J., 2015, "Dimension Reduction for the Design Optimization of Large Scale High Voltage Devices Using Co-Kriging Surrogate Modeling," IEEE T. Magn., **51**(3), pp. 1–4.

[30] Han, Z. H., Zhang, Y., Song, C. X., and Zhang, K. S., 2017, "Weighted Gradient-Enhanced Kriging for High-Dimensional Surrogate Modeling and Design Optimization," AIAA J., **55**(12), pp. 4330–4346.

[31] Hao, P., Feng, S., Zhang, K., Li, Z., Wang, B., and Li, G., 2018, "Adaptive Gradient-Enhanced Kriging Model for Variable-Stiffness Composite Panels Using Isogeometric Analysis," Struct. Multidiscip. O., **58**(1), pp. 1–16.

[32] Bouhlel, M. A., and Martins, J. R. R. A., 2019, "Gradient-Enhanced Kriging for High-Dimensional Problems," Eng. Comput., **35**(1), pp. 157–173.

[33] Shi, R., Liu, L., Long, T., Wu, Y., and Tang, Y., 2018, "Dual-Sampling Based Co-Kriging Method for Design Optimization Problems With Multi-Fidelity Models," 2018 Multidisciplinary Analysis and Optimization Conference, AIAA, Atlanta, GA, June 25–29, Paper No. AIAA-18-3747.
[34] Tokuda, I. T., Hoang, H., and Kawato, M., 2017, "New Insights Into Olivo-Cerebellar Circuits for Learning From a Small Training Sample," Curr. Opin. Neurobiol., **46**, pp. 58–67.
[35] Poggio, T., and Vetter, T., 1992, "Recognition and Structure from One 2D Model View: Observations on Prototypes, Object Classes and Symmetries," Laboratory Massachusetts Institute of Technology, Cambridge, MA, Report No. AI-M-1347.
[36] Yang, J., Yu, X., Xie, Z. Q., and Zhang, J. P., 2011, "A Novel Virtual Sample Generation Method Based on Gaussian Distribution," Knowl Based. Syst., **24**(6), pp. 740–748.
[37] Chen, Z. S., Zhu, B., He, Y. L., and Yu, L. A., 2017, "A PSO Based Virtual Sample Generation Method for Small Sample Sets: Applications to Regression Datasets," Eng. Appl. Artif. Intel., **59**, pp. 236–243.
[38] Chongfu, H., 1997, "Principle of Information Diffusion," Fuzzy Set. Syst., **91**(1), pp. 69–90.
[39] Li, D. C., Wu, C. S., Tsai, T. I., and Lina, Y. S., 2007, "Using Mega-Trend-Diffusion and Artificial Samples in Small Data Set Learning for Early Flexible Manufacturing System Scheduling Knowledge," Comput. Oper. Res., **34**(4), pp. 966–982.
[40] Lin, L. S., Li, D. C., and Pan, C. W., 2016, "Improving Virtual Sample Generation for Small Sample Learning With Dependent Attributes," 2016 5th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI), IEEE, Kumamoto, June 10–14, pp. 715–718.
[41] Li, D. C., and Lin, Y. S., 2006, "Using Virtual Sample Generation to Build Up Management Knowledge in the Early Manufacturing Stages," Eur. J. Oper. Res., **175**(1), pp. 413–434.
[42] Angelier, J., 1990, "Inversion of Field Data in Fault Tectonics to Obtain the Regional Stress—III. A New Rapid Direct Inversion Method by Analytical Means," Geophys. J. Int., **103**(2), pp. 363–376.
[43] Li, D. C., Lin, W. K., Lin, L. S., Chen, C. C., and Huang, W. T., 2016, "The Attribute-Trend-Similarity Method to Improve Learning Performance for Small Datasets," Int. J. Prod. Res., **55**(7), pp. 1898–1913.
[44] Gong, H. F., Chen, Z. S., Zhu, Q. X., and He, Y. L., 2017, "A Monte Carlo and PSO Based Virtual Sample Generation Method for Enhancing the Energy Prediction and Energy Optimization on Small Data Problem: An Empirical Study of Petrochemical Industries," Appl. Energ., **197**, pp. 405–415.
[45] Viana, F. A. C., and Goel, T., 2010, "Surrogates Toolbox User's Guide," http://sites.google.com/site/felipeacviana/surrogatestoolbox, Accessed April 30, 2020.
[46] Suykens, J. A., and Vandewalle, J., 1999, "Least Squares Support Vector Machine Classifiers," Neural Process. Lett., **9**(3), pp. 293–300.
[47] Karsmakers, P., Ojeda, F., Alzate, C. et al., 2010, *LS-SVMlab Toolbox User's Guide. Version 1.8*, Katholieke Universiteit Leuven, Leuven, Belgium.
[48] Kulfan, B., and Bussoletti, J., 2006, "Fundamental" Parameteric Geometry Representations for Aircraft Component Shapes," Proc. AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, VA, Sept. 6–8, Paper No. AIAA-06-6948.
[49] Schittkowski, K., 1986, "NLPQL: A Fortran Subroutine Solving Constrained Nonlinear Programming Problems," Ann. Oper. Res., **5**(2), pp. 485–500.
[50] Tanese, R., 1989, "Distributed Genetic Algorithms," Proceedings Third International Conference on Genetic Algorithms (ICGA), George Mason University, Fairfax, VA, June 4–7, pp. 434–439.
[51] Wu, B. B., Huang, H., Chen, S. Y., and Wu, W. R., 2013, "Multi-Disciplinary Design Optimization of Ocean Satellites Based on Analytical Target Cascading Strategy," Chinese J. Aeronaut., **34**(1), pp. 9–16.