Journal of Mechanical Engineering Research and Developments
ISSN: 1024-1752
Website: http://www.jmerd.org
Vol. 39, No. 2, 2016, pp. 278-295

High-Efficient Sequential Approximate Strategy for Reliability-Based Robust Design Optimization

X. M. Lai[†‡*], Q. F. Lai[†], H. Huang[†], C. Wang[§††], Y. Zhang[†], L. Yan[†],

J. H. Yang[†], & S. R. Liao[†]

[†]College of Mechanical Engineering and Automation, Huaqiao University, Xiamen, 361021, China,
*Email: laixiongming@hqu.edu.cn
[‡]Key Laboratory of High Performance Complex Manufacturing, Central South University, Changsha, 410083, China
[§]College of Computer Science and Technology, Huaqiao University, Xiamen, 361021, China
[††]State Key Laboratory for Strength and Vibration of Mechanical Structures, Xi'an Jiaotong University, Xi'an 710049, China

ABSTRACT: To solve reliability-based robust design optimization (RBRDO) problem in engineering practice, it requires repeated performance of actual complex computation-intensive model simulation within the nesting optimization. In order to improve the computation efficiency, this paper presents an improved formulation of the RBRDO, based on which the RBRDO problem is transformed into approximate sequential sub-optimization problems (ASSOP).In the outer optimization, a fast method to calculate the performance-measure-approach (PMA) functions in ASSOP is introduced, which only needs adaptive limited computation number of the actual simulation models. In the inner optimization, the PMAs in each sub-optimization problem are further expressed as the explicit functions of design vectors, which make each sub-optimization problem deterministic and accelerate the computation speed without performing the actual complex computation-intensive model simulation. Numerical examples shows that the above method is quite efficient since it can effectively reduce the computation number of the actual simulation model and the method itself is applicable to integrate into the existing commercial software such as finite element software. Hence, the method presented in this paper is of high application value to solve complex RBRDO problems in engineering practice.

KEYWORDS: Reliability-Based robust design optimization; Performance measure approach; Trust region; Kriging model.

INTRODUCTION

In engineering practice, the performance of structure varies and fluctuates due to uncertainties in design conditions. Some of them, such as loading, material properties, operating conditions and etc., are uncontrollable by the designer. The other uncertainties are controllable, the typical example of which is physical dimension of parts caused by manufacturing or measurement errors. Therefore, it is reasonable to explore the effect of uncertainties, which poses the task of robust design of structures with uncertainties.

The purpose of robust design is to make a product or response of a system insensitive to (or robust against) "hard-to-control" input parameters (called "noise factors"), by carefully adjusting "easy-to-control" input parameters (called "design parameters" or "control factors") [1]. With the development of computational capabilities and the progress in the optimization techniques, recent years has witnessed a growing research interest in the study of a reliability-based robust design optimization (RBRDO) method.

Depending on different aims, treatment of objective function and constraints, various statements of structural optimization problems under uncertainties have been formulated. Du et al. [2] employed an inverse strategy that used percentile performance for assessing the robustness of both the objective function and probabilistic constraints. Aiming at structural systems, Doltsinis and Kang [3-4] proposed a perturbation-based stochastic FE analysis technique and further integrated it into solving structural robust design problem. Youn et al. [5] used the PMI approach to estimate the product quality loss in RBRDO problem. Mourelatos and Liang [6] advocated a preference aggregation method to choose the best solution to a multi-objective optimization problem in robust design considering the designer's preferences. Lee et al.[7] developed a RBRDO method by using the univariate dimension reduction method and compared it to the performance moment integration method and the percentile difference method for accuracy and efficiency. Yadav et al.[8] proposed a model to combine both RDO and RBDO approaches

DOI: 10.7508/jmerd.2016.02.003

in a hybrid quality loss function-based multi-objective optimization model. Rathod et al. [9] presented a comparative study of different formulation approaches of RBRDO models and their performances and also proposed an evolutionary multi-objective genetic algorithm (MOGA) to one of the promising hybrid quality loss function-based RBRDO model. More recently, Shahraki and Noorossana [10] used a multi-objective evolutionary algorithm to solve the RBRDO problems in which the correlations among multiple objective functions were considered.

Despite the theoretical advances of RBRDO, a serious basic problem of computational obstacles still exists when treating most real-life engineering problems [11]. Generally, the computation of a single deterministic evaluation for complex simulation model are already time-consuming, let alone the repeated evaluations in RBRDO. Due to the difficulty, only a few studies address this issue for RBRDO. Harzheim and Warnecke [12], Gu and Lu [13] and Sun et al. [14] performed robust optimization of practical engineering designs using a cheap explicit surrogate model instead of the actual computation-intensive simulation model, and a remarkable reduction in the computational cost was achieved. But there exists another three problems. Firstly, the appropriate sampling size required for constructing the approximate surrogate model is difficult to determine. If it is too large, the computation consumption is too expensive. Otherwise, the accuracy of the constructed surrogate model is compromised. Secondly, the RBRDO is a typical nesting optimization, in which it necessitates rebuilding the surrogate model in the inner optimization according to the iterative design vector updated from the outer optimization and therefore the total evaluation number of the actual simulation model for samples may be considerable. Naturally, they still require disproportionately big computational effort for practical applications. Thirdly, when carrying out the Monte Carlo sampling (MCS) method and its variants on the surrogate models, the evaluations of the probabilistic characteristics (e.g. mean and standard deviation) vary to some degree since they are sensitive to sampling size, and they may make the optimization solutions unreliable. In this paper, the above problem can be overcomed. First of all, an improved formulation of the RBRDO is presented. Then the improved RBRDO problem is transformed into approximate sequential sub-optimization problems (ASSOP).In the outer optimization, a fast method to calculate the performance-measure-approach (PMA) functions in ASSOP is introduced, which only needs adaptive limited computation number of the actual simulation models. During this process, the Monte Carlo sample is avoided as well as the first and the third problem. In the inner optimization, the PMAs in each sub-optimization problem are further expressed as the explicit functions of design vectors, which make each sub-optimization problem deterministic and accelerate the computation speed without performing the actual complex computation-intensive model simulation. In so doing, the second problem is overcomed.

IMPROVED FORMULATION OF THE RBRDO PROBLEM

Consider an engineering design problem stated using the conventional optimization model in Eq. (1):

$$\begin{aligned} &\min \quad f(\mathbf{d},\mathbf{X};\boldsymbol{\mu}_{\mathbf{d}}) \\ &\text{s.t:} \ -g_j(\mathbf{d},\mathbf{X};\boldsymbol{\mu}_{\mathbf{d}}) \le 0 \quad j=1...q \\ &\boldsymbol{\mu}_{\mathbf{d}}^L \le \boldsymbol{\mu}_{\mathbf{d}} \le \boldsymbol{\mu}_{\mathbf{d}}^R, \mathbf{X}_{\mathbf{d}}^L \le \mathbf{d} \le \mathbf{X}_{\mathbf{d}}^R, \mathbf{X}^L \le \mathbf{X} \le \mathbf{X}^R \end{aligned} \tag{1}$$

where $,i=1\sim N, j=1\sim M$ is the original system performance function. K_{kri}^t are constraint functions, also called performance functions. $\mathbf{d}=[d_1,...d_m]^T$ is a vector of the controllable random design variables and $\mathbf{X}=[X_1,...X_n]^T$ is a vector of the uncontrollable random variables. $\mathbf{u}_{MPP}^t$ is the mean value of the controllable design variables $\left\|\mathbf{u}_{MPP}^t - \mathbf{u}_{MPP}^{t-1}\right\| / \left\|\mathbf{u}_{MPP}^{t-1}\right\| \le \varepsilon_1$. The superscripts L and R denote lower and upper bounds of interval, respectively. Both $\mathbf{d}$ and $\mathbf{X}$ are treated as random variables in the inner optimization, while $\boldsymbol{\mu}_{\mathbf{d}}$ is treated as the design variable in the outer optimization. Additionally, both $\mathbf{d}$ and $\mathbf{X}$ are treated as independent normal variables, which are most frequently used in engineering applications. For the sake of convenient statement in the following paper, let $\mathbf{z}=[\mathbf{d},\mathbf{X}]$ presents the norm random variables, then $\mathbf{z}^L=[\mathbf{X}_{\mathbf{d}}^L,\mathbf{X}^L]$ and $\mathbf{z}^R=[\mathbf{X}_{\mathbf{d}}^R,\mathbf{X}^R]$ are the lower and upper random variation bounds for $\mathbf{z}$, respectively. Correspondingly, Eq. (1) can be expressed as:

$$\begin{aligned} &\min \quad f(\mathbf{z};\boldsymbol{\mu}_{\mathbf{d}}) \\ &\text{s.t:} \ -g_j(\mathbf{z};\boldsymbol{\mu}_{\mathbf{d}}) \le 0 \quad j=1...q \\ &\boldsymbol{\mu}_{\mathbf{d}}^L \le \boldsymbol{\mu}_{\mathbf{d}} \le \boldsymbol{\mu}_{\mathbf{d}}^R, \mathbf{z}^L \le \mathbf{z} \le \mathbf{z}^R \end{aligned} \tag{2}$$

Apparently, f and g_j in Eq. (3) also have variations within certain interval. Thus, in robust design, the robustness of a design objective can be achieved by simultaneously minimizing both the mean performance μ_f and the performance variance σ_f^2, shown as below:

$$\min\ [\mu_f, \sigma_f] \tag{3}$$

Generally, the deterministic optimum designs without considering the uncertainty of design variables and random variables frequently push design constraints to the limit of boundaries and lead objective performance variation to large fluctuation. In order to stop pushing design constraints to the limit of boundaries, the reliable design indicates an optimization subjecting to probabilistic bounds on the constraints as [10]:

$$P\{g_j(\mathbf{z};\boldsymbol{\mu}_\mathbf{d}) \le 0\} \le p_f^j \quad j = 1...q \tag{4}$$

where p_f^j $(j = 1...q)$ are desired probabilities of constraint failure.

Based on Eq. (3) and (4), the uncertainties in Eq. (1) can be changed as following multi-objective optimization problem:

$$\begin{aligned} &\min \quad [\mu_f, \sigma_f] \\ &\text{s.t:}\ P\{g_j(\mathbf{z};\boldsymbol{\mu}_\mathbf{d}) \le 0\} \le p_f^j \quad j = 1...q \\ &\boldsymbol{\mu}_\mathbf{d}^L \le \boldsymbol{\mu}_\mathbf{d} \le \boldsymbol{\mu}_\mathbf{d}^R, \mathbf{z}^L \le \mathbf{z} \le \mathbf{z}^R \end{aligned} \tag{5}$$

The literature shows that the weighted sum (WS) method is widely used to form a composite objective function for achieving the trade-off among the performance measures. Hence, Eq. (5) is expressed by the WS method as:

$$\begin{aligned} &\min \quad W = (1-\alpha)\mu_f / \phi + \alpha\sigma_f / \varphi \\ &\text{s.t:}\ P\{g_j(\mathbf{z};\boldsymbol{\mu}_\mathbf{d}) \le 0\} \le p_f^j \quad j = 1...q \\ &\boldsymbol{\mu}_\mathbf{d}^L \le \boldsymbol{\mu}_\mathbf{d} \le \boldsymbol{\mu}_\mathbf{d}^R, \mathbf{z}^L \le \mathbf{z} \le \mathbf{z}^R \end{aligned} \tag{6}$$

where the weighting factor $\alpha \in [0,1]$ is determined by the importance of minimization and robus tness. μ_f and σ_f are the mean value and standard deviation of $, i = 1 \sim N, j = 1 \sim M$. ϕ and φ are normalization factors of the two objectives.

Eq. (6) is a nesting optimization. In the outer optimization, the iterative design vector is updated. In the inner optimization, the probabilistic uncertainties are obtained by an amount of evaluations of the actual simulation model. Thus, for practical structures, especially for ones with computation-intensive simulation model, the computation cost caused by the nesting optimization of the actual simulation model is extremely high and generally unacceptable.

When dealing with the probabilistic uncertainty in RBRDO, performance measure approach (PMA) is adopted in this paper. It judges whether a given design satisfies the probabilistic constraint with a given target reliability index β^j by solving the following optimization problem:

$$\begin{aligned} &\min \quad G_j(\mathbf{U};\boldsymbol{\mu}_\mathbf{d}) \\ &\text{s.t:} \quad \|\mathbf{U}\| = \beta^j \end{aligned} \tag{7}$$

where $\mathbf{U}$ is the standard normal random vectors transformed from $\mathbf{z}$ by using Rosenblatt transformation[16-17], i.e. $\mathbf{U} = T(\mathbf{z})$. For example, if $\mathbf{z}$ are normal variables, then $\mathbf{z} = \boldsymbol{\sigma}_\mathbf{z}\mathbf{U} + \boldsymbol{\mu}_\mathbf{z}$. $G_j(\mathbf{U};\boldsymbol{\mu}_\mathbf{d}) \equiv g_j(\mathbf{z};\boldsymbol{\mu}_\mathbf{d})$. The optimum point of Eq. (7) is called the most probable point (MPP) and denoted by $\mathbf{U}_j^*$ in U-space.

Figure 1 illustrates the concept of PMA. The MPP is found on the circle of radius β^j for which the constraint function $G_j(\mathbf{U};\boldsymbol{\mu}_\mathbf{d})$ reaches its maximum value. Then the original probabilistic constraint function, $P\{g_j(\mathbf{z};\boldsymbol{\mu}_\mathbf{d}) \le 0\} \le p_f^j$, is replaced by the deterministic function, $G_j(\mathbf{U}_j^*;\boldsymbol{\mu}_\mathbf{d}) \ge 0$.

The distribution of $,i=1\sim N, j=1\sim M$ is shown in Figure 2. *f_{p0.5}* (i.e. μ_f) is the mean value of $,i=1\sim N, j=1\sim M$, i.e. μ_f . p1 is a left-tail percentile. p2 is a right-tail percentile and, in general, p1 + p2 =1. If the given target reliability index β^f for $,i=1\sim N, j=1\sim M$ is equal to 3, p1=1-Φ(3)≈0.00135 and p2=1-Φ(3) ≈0.99865. It means that the probability of covering the distribution range of $,i=1\sim N, j=1\sim M$ is 0.9973, i.e. the shaded area in Figure 2.

Then f_{p1} can be approximated by the optimum result obtained by minimizing the following optimization problem:

$$\begin{aligned} \min \quad & F(\mathbf{U};\boldsymbol{\mu}_\mathbf{d}) \\ \text{s.t:} \quad & \|\mathbf{U}\| = \beta^f \end{aligned} \tag{8}$$

where $F(\mathbf{U};\boldsymbol{\mu}_\mathbf{d}) \equiv f(\mathbf{z};\boldsymbol{\mu}_\mathbf{d})$. Suppose the optimum solution is $\mathbf{U}^*_{p1}$, then $f_{p1} = F(\mathbf{U}^*_{p1};\boldsymbol{\mu}_\mathbf{d})$. Similarly, f_{p2} can be approximated by the optimum result obtained by maximizing the objective function $F(\mathbf{U};\boldsymbol{\mu}_\mathbf{d})$ in Eq. (9). Suppose the corresponding optimum solution is $\mathbf{U}^*_{p2}$, then $f_{p2} = F(\mathbf{U}^*_{p2};\boldsymbol{\mu}_\mathbf{d})$. Since f_{p1}, f_{p2} and $G_j(\mathbf{U}^*_j;\boldsymbol{\mu}_\mathbf{d})$ are obtained by PMA, each is called the PMA function, for the sake of convenient statement in the following paper.

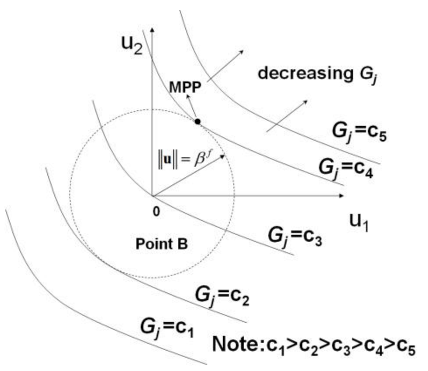


Figure 1. Concept of performance measure approach.

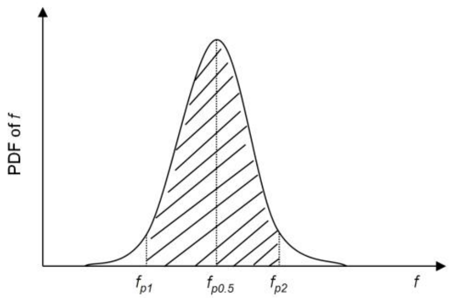


Figure 2.The distribution of f.

There are several advantages of using the percentile performance difference (i.e. f_{p2}- f_{p1}) to replace the conventional standard deviation σ_f in Eq. (6) for robustness assessment [2]. But it's not applicable to the case that the performance function is not monotonic [7]. For example, if f(z) = z 2 and z ~ N(0, 1) and β^f is 3, then two MPPs become 1.732 and -1.732. Thus, the percentile performance difference (i.e. f_{p2}- f_{p1}) is 0. Therefore the following improvement is introduced by considering three characteristic values, i.e. $f_{p0.5}$, f_{p1} and f_{p2}.

$$\min \quad [\mu_f, \max(|f_{p1}-\mu_f|, |f_{p2}-\mu_f|)] \tag{9}$$

By using the WS method, Eq. (9) is expressed as:

$$W = (1-\alpha)\mu_f/\phi + \alpha\Delta_f/\varphi \tag{10}$$

where Δ_f is $\max(|f_{p1}-\mu_f|, |f_{p2}-\mu_f|)$.

Through above treatments, the uncertain optimization Eq. (6) can be formulated as:

$$\begin{aligned}
&\min \quad W = \alpha\mu_f / \phi + (1-\alpha)\Delta_f / \varphi \\
&\text{s.t:} \ \ g_j(\mathbf{z}_j^*;\boldsymbol{\mu}_\mathbf{d}) \geq 0 \quad j = 1...q \\
&\quad \Delta_f = \max(|\, f_{p1} - \mu_f \,|, |\, f_{p2} - \mu_f \,|) \\
&\quad f_{p1} = F(\mathbf{U}_{p1}^*;\boldsymbol{\mu}_\mathbf{d}) \equiv f(\mathbf{z}_{p1}^*;\boldsymbol{\mu}_\mathbf{d}) \\
&\quad f_{p2} = F(\mathbf{U}_{p2}^*;\boldsymbol{\mu}_\mathbf{d}) \equiv f(\mathbf{z}_{p2}^*;\boldsymbol{\mu}_\mathbf{d}) \\
&\quad g_j(\mathbf{z}_j^*;\boldsymbol{\mu}_\mathbf{d}) \equiv G_j(\mathbf{U}_j^*;\boldsymbol{\mu}_\mathbf{d}) \\
&\quad \boldsymbol{\mu}_\mathbf{d}^L \leq \boldsymbol{\mu}_\mathbf{d} \leq \boldsymbol{\mu}_\mathbf{d}^R, \mathbf{z}^L \leq \mathbf{z} \leq \mathbf{z}^R
\end{aligned} \tag{11}$$

where $\mathbf{z}_j^*, \mathbf{z}_{p1}^*, \mathbf{z}_{p2}^*$ are inverse Rosenblatt transformation from $\mathbf{U}_j^*$, $\mathbf{U}_{p1}^*$ and $\mathbf{U}_{p2}^*$, respectively.

Using penalty function method, Eq. (11) can be formulated to:

$$\begin{aligned}
&\min \quad H_p = W + \kappa\sum_{j=1}^{q}\max(0, -g_j(\mathbf{z}_j^*;\boldsymbol{\mu}_\mathbf{d})) \\
&\quad W = \alpha\mu_f / \phi + (1-\alpha)\Delta_f / \varphi \\
&\quad \Delta_f = \max(|\, f_{p1} - \mu_f \,|, |\, f_{p2} - \mu_f \,|) \\
&\quad f_{p1} = F(\mathbf{U}_{p1}^*;\boldsymbol{\mu}_\mathbf{d}) \equiv f(\mathbf{z}_{p1}^*;\boldsymbol{\mu}_\mathbf{d}) \\
&\quad f_{p2} = F(\mathbf{U}_{p2}^*;\boldsymbol{\mu}_\mathbf{d}) \equiv f(\mathbf{z}_{p2}^*;\boldsymbol{\mu}_\mathbf{d}) \\
&\quad g_j(\mathbf{z}_j^*;\boldsymbol{\mu}_\mathbf{d}) \equiv G_j(\mathbf{U}_j^*;\boldsymbol{\mu}_\mathbf{d}) \\
&\quad \boldsymbol{\mu}_\mathbf{d}^L \leq \boldsymbol{\mu}_\mathbf{d} \leq \boldsymbol{\mu}_\mathbf{d}^R, \mathbf{z}^L \leq \mathbf{z} \leq \mathbf{z}^R
\end{aligned} \tag{12}$$

where κ is a penalty factor which is always specified as a large value.

A conventional and general method to solve Eq. (12) is illustrated in Figure 3. It is still a typical nesting optimization problem. Apparently, an amount of evaluations of the actual simulation model are carried out when computing the PMA functions in the inner optimization and such jobs should also be repeated each time when the design vector $\boldsymbol{\mu}_\mathbf{d}$ is updated in the outer optimization. In realistic engineering applications, the key problem lies in minimizing the evaluation number of the actual simulation model in the inner optimization from the perspective of computation consumption. Hence, the above conventional method is inefficient. The presented improved formulation of the RBRDO can be conveniently solved by the sequential approximate strategy method suggested in the following paper which can minimize the computation number of the PMA functions in ASSOP. It is of great significance for the RBRDO method to apply in engineering complex problems.

AN EFFICIENT SEQUENTIAL APPROXIMATE STRATEGY OF RBRDO

An efficient sequential approximate strategy of RBRDO is presented in this section. It mainly

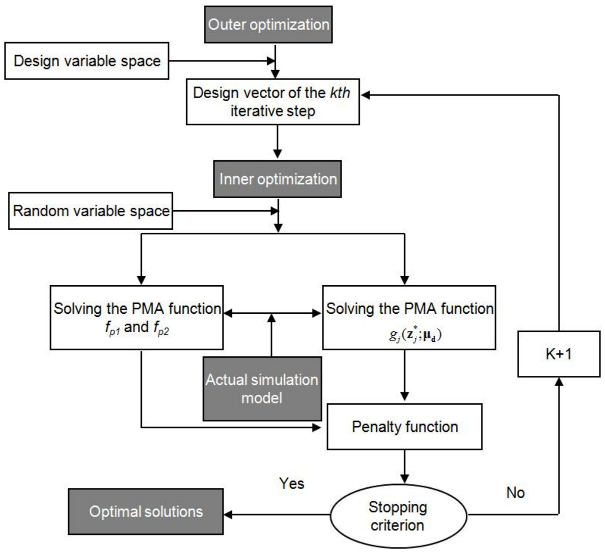


Figure 3. The flowchart of nesting optimization based on actual simulation model.

contains three key steps. Firstly, formulating local approximate analytical expressions respectively towards the PMA functions f_{p1} , f_{p2} and $G_j(\mathbf{U}_j^*;\boldsymbol{\mu}_\mathbf{d})$ in Eq. (12). Secondly, transforming the RBRDO problem into ASSOP. Finally, the fast computation method for determining the PMA functions in Eq. (12) is proposed.

Local explicit functions for the PMA function

The computational efficiency of the inner optimization is further increased by replacing the original PMA functions including f_{p1} , f_{p2} and $G_j(\mathbf{U}_j^*;\boldsymbol{\mu}_\mathbf{d})$ with equivalent ones. The objective of this replacement is to reduce the computational cost using repeated original PMA function estimates with respect to the updated design vector during the inner optimization. The equivalent PMA functions are represented using approximate functions dependent on the design vector. Then in the inner optimization the original PMA functions can be replaced by the explicit functions. Hence, the uncertainty optimization problems can be transformed into deterministic ones, then the computation number for the actual model in each PMA function can be minimized, which greatly improves the computational efficiency. In fact, the MPP is the implicit function of the design vector. Take f_{p1} as example. $\mathbf{U}_{\mathrm{p1}}^*$ in f_{p1} $(= f(\mathbf{z}_{\mathrm{p1}}^*;\boldsymbol{\mu}_\mathbf{d}) \equiv F(\mathbf{U}_{\mathrm{p1}}^*;\boldsymbol{\mu}_\mathbf{d}))$ is determined by the design vector $\boldsymbol{\mu}_\mathbf{d}$. Hence, $F(\mathbf{U}_{\mathrm{p1}}^*;\boldsymbol{\mu}_\mathbf{d})$ in nature is the function only with respect to the design vector c_1 . In RBRDO, it 's much more possible to construct a local approximation around a current optimal design c_2 at the kth outer optimization, say

$$f_{\mathrm{p1}} = f(\mathbf{z}_{\mathrm{p1}}^*;\boldsymbol{\mu}_\mathbf{d}) \equiv F(\mathbf{U}_{\mathrm{p1}}^*(\boldsymbol{\mu}_\mathbf{d});\boldsymbol{\mu}_\mathbf{d}) \approx F(\mathbf{U}_{\mathrm{p1}}^{*k}(\boldsymbol{\mu}_\mathbf{d}^k);\boldsymbol{\mu}_\mathbf{d}^k) + \mathbf{a}(\boldsymbol{\mu}_\mathbf{d} - \boldsymbol{\mu}_\mathbf{d}^k)^T \tag{13}$$

where $F(\ )$ can be computed by the method presented in Section 3.3. Note that $\mathbf{U}_{\mathrm{p1}}^*$ is deliberately written as the function of $\boldsymbol{\mu}_\mathbf{d}^k$ in Eq. (13), i.e. $\mathbf{U}_{\mathrm{p1}}^*(\boldsymbol{\mu}_\mathbf{d})$, and the $(\boldsymbol{\mu}_\mathbf{d})$ in $\mathbf{U}_{\mathrm{p1}}^*(\boldsymbol{\mu}_\mathbf{d})$ are ignored in the following paper for the sake of convenience. Similarly, $\mathbf{U}_{\mathrm{p1}}^{*k}(\boldsymbol{\mu}_\mathbf{d}^k)$ is also abbreviated as $\mathbf{U}_{\mathrm{p1}}^{*k}$ in the following paper, whose superscript k denotes that the value of $\mathbf{U}_{\mathrm{p1}}^*$ is computed at $\boldsymbol{\mu}_\mathbf{d}^k$. $\mathbf{a} = [a_1, a_2, ..., a_m]$ denotes the constant coefficients, which can be calculated by:

$$a_i = \frac{\partial f(\mathbf{z}^*_{p1};\boldsymbol{\mu}_\mathbf{d})}{\partial \mu_{di}}\bigg|_{\boldsymbol{\mu}_\mathbf{d}=\boldsymbol{\mu}^k_\mathbf{d}} \equiv \frac{\partial F(\mathbf{U}^*_{p1};\boldsymbol{\mu}_\mathbf{d})}{\partial \mu_{di}}\bigg|_{\boldsymbol{\mu}_\mathbf{d}=\boldsymbol{\mu}^k_\mathbf{d}}$$
$$= \frac{\partial F(\mathbf{U}^{*k}_{p1};\boldsymbol{\mu}_\mathbf{d})}{\partial U^{*k}_{p1,i}}\bigg|_{\boldsymbol{\mu}_\mathbf{d}=\boldsymbol{\mu}^k_\mathbf{d}} \frac{\partial U^{*k}_{p1,i}}{\partial \mu_{di}}$$
$$= \frac{\partial F(\mathbf{U}^{*k}_{p1};\boldsymbol{\mu}_\mathbf{d})}{\partial U^{*k}_{p1,i}}\bigg|_{\boldsymbol{\mu}_\mathbf{d}=\boldsymbol{\mu}^k_\mathbf{d}} \frac{\partial T(z^{*k}_{p1,i})}{\partial \mu_{di}} \tag{14}$$
$$= \frac{\partial F(\mathbf{U}^{*k}_{p1};\boldsymbol{\mu}_\mathbf{d})}{\partial U^{*k}_{p1,i}}\bigg|_{\boldsymbol{\mu}_\mathbf{d}=\boldsymbol{\mu}^k_\mathbf{d}} \frac{\partial T(d^{*k}_{p1,i})}{\partial \mu_{di}}$$

where i=1~m. As mentioned before, $\mathbf{z}=[\mathbf{d},\mathbf{X}]$. Hence, first m components of $\mathbf{z}^{*k}_{p1}$, namely $z^{*k}_{p1,i}(i=1\sim m)$, are corresponding to the random variable $\mathbf{d}$. Here $z^{*k}_{p1,i}(i=1\sim m)$ are expressed as $d^{*k}_{p1,i}(i=1\sim m)$. If d_i is a normal variable, then $\partial T(d^{*k}_{p1,i})/\partial\mu_{di} = -1/\sigma_{di}$ with the help of Eq. (15). μ_{di} and σ_{di} are the mean and standard deviation of the ith component of $\mathbf{d}$. $\partial F(\mathbf{U}^*_{p1};\boldsymbol{\mu}_\mathbf{d})/\partial U_i$ can be derived from the Kriging model described hereafter.

$$U^{*k}_{p1,i} = T(z^{*k}_{p1,i}) = T(d^{*k}_{p1,i}) = (d^{*k}_{p1,i} - \mu_{di})/\sigma_{di} \tag{15}$$

In fact, it is impossible to find the real value of the MPP $\mathbf{U}^{*k}_{p1}$, however, it can be approximated by iterative optimization computation from Eq. (7), which denotes as $\tilde{\mathbf{U}}^{*k}_{p1}$. Correspondingly, f_{p1} in Eq. (13) is approximately represented as:

$$\tilde{f}_{\mathrm{p1}}(\boldsymbol{\mu}_\mathbf{d}) = F(\tilde{\mathbf{U}}^{*k}_{\mathrm{p1}}(\boldsymbol{\mu}^k_\mathbf{d});\boldsymbol{\mu}^k_\mathbf{d}) + \mathbf{a}(\boldsymbol{\mu}_\mathbf{d} - \boldsymbol{\mu}^k_\mathbf{d})^T \tag{16}$$

In this way, $\tilde{f}_{\mathrm{p1}}(\boldsymbol{\mu}_\mathbf{d})$ represents the approximate explicit function with respect only to the design vector $\boldsymbol{\mu}_\mathbf{d}$.

Through above similar treatments, f_{p2} and $g_j(\mathbf{z}^*_j;\boldsymbol{\mu}_\mathbf{d})$ can be formulated as the following approximate analytical expressions:

$$\tilde{f}_{\mathrm{p2}}(\boldsymbol{\mu}_\mathbf{d}) = F(\tilde{\mathbf{U}}^{*k}_{\mathrm{p2}};\boldsymbol{\mu}^k_\mathbf{d}) + \mathbf{b}(\boldsymbol{\mu}_\mathbf{d} - \boldsymbol{\mu}^k_\mathbf{d})^T \tag{17}$$

$$\tilde{g}_j(\boldsymbol{\mu}_\mathbf{d}) = G_j(\tilde{\mathbf{U}}^{*k}_{\mathrm{j}};\boldsymbol{\mu}^k_\mathbf{d}) + \mathbf{c}(\boldsymbol{\mu}_\mathbf{d} - \boldsymbol{\mu}^k_\mathbf{d})^T \tag{18}$$

where $\mathbf{b}$ and $\mathbf{c}$ are the constant coefficients. $\tilde{f}_{\mathrm{p2}}(\boldsymbol{\mu}_\mathbf{d})$ and $\tilde{g}_j(\boldsymbol{\mu}_\mathbf{d})$ are also the approximate explicit function only with respect to the design vector $\boldsymbol{\mu}_\mathbf{d}$.

Sequential approximate RBRDO

In the optimization process, a sequence of approximate optimizations is constructed. At the kth iterative step, the approximate optimization for Eq. (12) can be formulated as:

$$\begin{aligned}
&\min \quad W = \alpha\mu_f/\phi + (1-\alpha)\Delta_f/\varphi \\
&\text{s.t:} \quad g_j(\mathbf{z}_j^*;\boldsymbol{\mu}_\mathbf{d}) \geq 0 \quad j = 1\ldots q \\
&\Delta_f = \max(|f_{p1}(\boldsymbol{\mu}_\mathbf{d}) - \mu_f|, |f_{p2}(\boldsymbol{\mu}_\mathbf{d}) - \mu_f|) \\
&f_{p1} = F(\mathbf{U}_{p1}^*;\boldsymbol{\mu}_\mathbf{d}) \equiv f(\mathbf{z}_{p1}^*;\boldsymbol{\mu}_\mathbf{d}) \\
&f_{p2} = F(\mathbf{U}_{p2}^*;\boldsymbol{\mu}_\mathbf{d}) \equiv f(\mathbf{z}_{p2}^*;\boldsymbol{\mu}_\mathbf{d}) \\
&g_j(\mathbf{z}_j^*;\boldsymbol{\mu}_\mathbf{d}) \equiv G_j(\mathbf{U}_j^*;\boldsymbol{\mu}_\mathbf{d}) \\
&\max[\boldsymbol{\mu}_\mathbf{d}^L, \boldsymbol{\mu}_\mathbf{d}^k - \boldsymbol{\delta}^k] \leq \boldsymbol{\mu}_\mathbf{d} \leq \min[\boldsymbol{\mu}_\mathbf{d}^R, \boldsymbol{\mu}_\mathbf{d}^k + \boldsymbol{\delta}^k], \mathbf{z}^L \leq \mathbf{z} \leq \mathbf{z}^R
\end{aligned} \tag{19}$$

where μ_f^k, $\boldsymbol{\delta}^k$ and $\boldsymbol{\mu}_\mathbf{d}^k$ are the mean values of f, the trust region radius vector and the current design vector at the kth iterative step, respectively. $\boldsymbol{\delta}^k$ is temporary for each iterative step, and changes with the proceeding of the outer optimization. By using the local explicit functions for the PMA value suggested in Section 3.3, the following approximate optimization model for Eq. (19) is given by:

$$\begin{aligned}
&\min \quad \tilde{W}(\boldsymbol{\mu}_\mathbf{d}) = \alpha\mu_f/\phi + (1-\alpha)\tilde{\Delta}_f/\varphi \\
&\text{s.t:} \quad \tilde{g}_j(\boldsymbol{\mu}_\mathbf{d}) \geq 0 \quad j = 1\ldots q \\
&\tilde{\Delta}_f = \max(|\tilde{f}_{p1}(\boldsymbol{\mu}_\mathbf{d}) - \mu_f|, |\tilde{f}_{p2}(\boldsymbol{\mu}_\mathbf{d}) - \mu_f|) \\
&\max[\boldsymbol{\mu}_\mathbf{d}^L, \boldsymbol{\mu}_\mathbf{d}^k - \boldsymbol{\delta}^k] \leq \boldsymbol{\mu}_\mathbf{d} \leq \min[\boldsymbol{\mu}_\mathbf{d}^R, \boldsymbol{\mu}_\mathbf{d}^k + \boldsymbol{\delta}^k]
\end{aligned} \tag{20}$$

where $\tilde{f}_{p1}(\boldsymbol{\mu}_\mathbf{d})$, $\tilde{f}_{p2}(\boldsymbol{\mu}_\mathbf{d})$ and $\tilde{g}_j(\boldsymbol{\mu}_\mathbf{d})$ are approximate explicit representations only with respect to the design vector around a current design $\boldsymbol{\mu}_\mathbf{d}^k$, i.e. Eqs. (16)-(18). Correspondingly, Δ_f and $W(\boldsymbol{\mu}_\mathbf{d})$ computed by the above approximate PMA functions are also written as $\tilde{\Delta}_f$ and $\tilde{W}(\boldsymbol{\mu}_\mathbf{d})$, respectively. Obviously, Eq.(20) is the deterministic optimization and it can be further changed as the following penal format like Eq. (12):

$$\begin{aligned}
&\min \quad \tilde{H}_p(\boldsymbol{\mu}_\mathbf{d}) = \tilde{W}(\boldsymbol{\mu}_\mathbf{d}) + \kappa\sum_{j=1}^{q}\max(0, -\tilde{g}_j(\boldsymbol{\mu}_\mathbf{d})) \\
&\tilde{W}(\boldsymbol{\mu}_\mathbf{d}) = \alpha\mu_f/\phi + (1-\alpha)\tilde{\Delta}_f/\varphi \\
&\tilde{\Delta}_f = \max(|\tilde{f}_{p1}(\boldsymbol{\mu}_\mathbf{d}) - \mu_f|, |\tilde{f}_{p2}(\boldsymbol{\mu}_\mathbf{d}) - \mu_f|) \\
&\max[\boldsymbol{\mu}_\mathbf{d}^L, \boldsymbol{\mu}_\mathbf{d}^k - \boldsymbol{\delta}^k] \leq \boldsymbol{\mu}_\mathbf{d} \leq \min[\boldsymbol{\mu}_\mathbf{d}^R, \boldsymbol{\mu}_\mathbf{d}^k + \boldsymbol{\delta}^k]
\end{aligned} \tag{21}$$

where $\tilde{H}_p(\boldsymbol{\mu}_\mathbf{d})$ (termed as "approximate penalty function") is computed also based on the above approximate PMA functions. And particle swarm optimization (PSO) will also be applied to work with Eq. (21) in this paper, since it possesses a fine global convergence performance and has been widely used in an amount of literatures[18-19].

PSO makes use of a velocity vector to update the current position of each particle in the swarm, i.e.

$$v_{id}^{k+1} = \omega v_{id}^k + c_1\delta_1(P_{id}^k - x_{id}^k) + c_2\delta_2(G_{id}^k - x_{id}^k) \tag{22}$$

$$x_{id}^{k+1} = x_{id}^k + v_{id}^k \tag{23}$$

where v_{id}^k and v_{id}^k are respectively the dth velocity component and the dth location component of particle i at iteration k. v_{id}^{k+1} and x_{id}^{k+1} are respectively the dth velocity component and the dth location component of particle i at iteration k+1. P_{id}^k is the dth dimension of the personal best location of particle i, and G_{id}^k is the dth dimension of the global best location of the whole swarm. c_1 and c_2 denote the cognition learning factor and the social learning factor, respectively. δ_1 and δ_2 are random numbers drawn from a uniform distribution in [0, 1]. ω is inertia weight. As for the examples illustrated in this paper, ω=0.7298, $c_1 = c_2$=1.4962. For more information about the PSO, readers please refer to Section 2.2 in [20].

Since Eq. (21) is the deterministic optimization and the actual model simulation is avoided, the computational effort required by PSO is acceptable.

The side constraints for the design vector $\boldsymbol{\mu}_{\mathbf{d}}$ in the approximate problems are changed at each design cycle in order to protect the quality of the approximation between Eq. (12) and Eq. (21). Hence, the choice of the trust region radius vector $\boldsymbol{\delta}^k$ is the key step. The trust region method provides a strategy to adjust $\boldsymbol{\delta}^k$ at each design cycle. At the kth iterative step, a trust region criterion is applied to check the fidelity of the current approximation optimization in Eq. (21), which is quantified by the approximate degree of the penalty function in this paper. Hence, the trust index containing the information of the penalty function is defined as follows:

$$\rho^k = (H_p(\boldsymbol{\mu}_{\mathbf{d}}^k) - H_p(\boldsymbol{\mu}_{\mathbf{d}}^{*k})) / (H_p(\boldsymbol{\mu}_{\mathbf{d}}^k) - \tilde{H}_p(\boldsymbol{\mu}_{\mathbf{d}}^{*k})) \tag{24}$$

where $\boldsymbol{\mu}_{\mathbf{d}}^{k*}$ is the optimum for the kth approximation optimization problem in Eq. (21). In Eq. (24), the approximate penalty function $\tilde{H}_p$ can be obtained by solving Eq. (21). And it also necessitates the computation of H_p, which denotes the actual value of the penalty function (termed as "actual penalty function") and is always unknown and implicit for most of the complex engineering problems. In the following text, details of the determining process of the actual penalty function will be given.

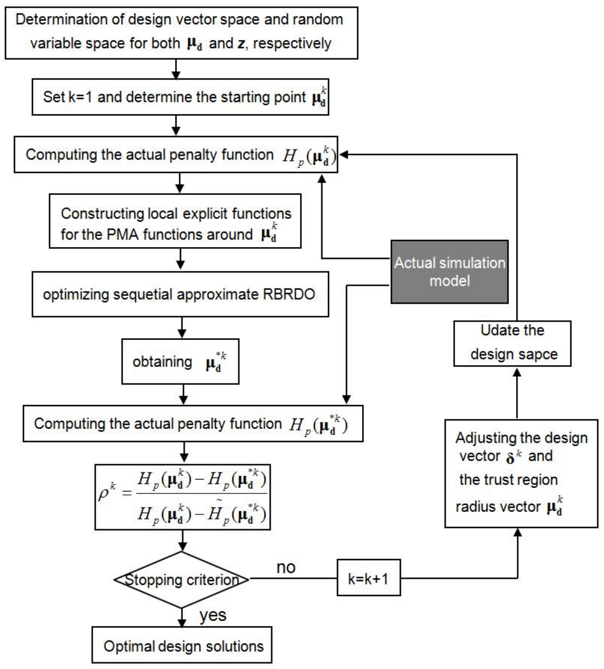
According to the value of ρ^k, the current trust region radius vector can be updated. If $\rho^k \le 0$, it signifies that the approximation is bad and the trust region radius $\boldsymbol{\delta}^k$ should be reduced. If $\rho^k \approx 1$, it implies that a good approximation is made and the trust region radius $\boldsymbol{\delta}^k$ should be increased when $\left\| \boldsymbol{\mu}_{\mathbf{d}}^{k*} - \boldsymbol{\mu}_{\mathbf{d}}^k \right\| = \boldsymbol{\delta}^k$. If $\rho^k \ge 1$, it also means that the approximation is bad, but the search direction is right. Finally, if $0 < \rho^k < 1$, the trust region radius vector should be unchanged, reduced, or expanded according to how close to 0 or 1 ρ^k is. In general, there are many trust region methods that can be used to adjust the trust region radius for sequential approximate optimization[21]. They have shown good convergence properties and are frequently used in the area of structural optimization. In this paper, the following updating procedure is adopted[11, 22], which has been proved to be adequate in the context of the present study.

$$\begin{cases} \boldsymbol{\delta}^{k+1} = 0.5\boldsymbol{\delta}^k, \boldsymbol{\mu}_{\mathbf{d}}^{k+1} = \boldsymbol{\mu}_{\mathbf{d}}^k & if \quad \rho^k \le 0 \\ \boldsymbol{\delta}^{k+1} = 0.5\boldsymbol{\delta}^k, \boldsymbol{\mu}_{\mathbf{d}}^{k+1} = \boldsymbol{\mu}_{\mathbf{d}}^{k*} & if \quad 0 \le \rho^k \le 0.25 \\ \boldsymbol{\delta}^{k+1} = \boldsymbol{\delta}^k, \boldsymbol{\mu}_{\mathbf{d}}^{k+1} = \boldsymbol{\mu}_{\mathbf{d}}^{k*} & if \quad 0.25 \le \rho^k \le 0.75 \\ \boldsymbol{\delta}^{k+1} = 2\boldsymbol{\delta}^k, \boldsymbol{\mu}_{\mathbf{d}}^{k+1} = \boldsymbol{\mu}_{\mathbf{d}}^{k*} & if \quad \rho^k \ge 0.75 \quad and \quad \left\| \boldsymbol{\mu}_{\mathbf{d}}^{k*} - \boldsymbol{\mu}_{\mathbf{d}}^k \right\| = \boldsymbol{\delta}^k \\ \boldsymbol{\delta}^{k+1} = \boldsymbol{\delta}^k, \boldsymbol{\mu}_{\mathbf{d}}^{k+1} = \boldsymbol{\mu}_{\mathbf{d}}^{k*} & if \quad \rho^k \ge 0.75 \quad and \quad \left\| \boldsymbol{\mu}_{\mathbf{d}}^{k*} - \boldsymbol{\mu}_{\mathbf{d}}^k \right\| \prec \boldsymbol{\delta}^k \end{cases} \tag{25}$$

Generally, when finish each inner optimization in Eq. (21), the above trust index is computed in order to check whether the design vector $\boldsymbol{\mu}_{\mathbf{d}}^k$ and the trust region radius vector $\boldsymbol{\delta}^k$ need be changed in the next inner iteration. Figure 4 outlines the main sequential approximate strategy for the RBRDO. At each iterative step, the actual penalty function $H_p(\boldsymbol{\mu}_{\mathbf{d}}^k)$ is computed first, whose computation process contains sample generation, actual simulation and the gradient information gathering necessary for building the local explicit functions for the PMA functions, i.e. Eqs. (16)- (18) in the subsequent step. After creating all of the local explicit PMA functions, then solve the sequential approximate RBRDO in Eq. (24), an optimal design vector $\boldsymbol{\mu}_{\mathbf{d}}^{k*}$ can be achieved for the current iterative step. After that, the actual penalty function $H_p(\boldsymbol{\mu}_{\mathbf{d}}^k)$ should be computed. Consequently, the design vector and the trust region radius vector are updated according to the value of the trust index. This process will be repeated until the stopping criterion is satisfied. In this paper, the maximum iterate number is employed as stopping criterion.

Fast method for determining the PMA function

Observed from Eq. (12), it necessitates the computation of the PMA functions, including f_{p1} , f_{p2} and $G_j(\mathbf{U}_j^*;\boldsymbol{\mu}_\mathbf{d})$ to compute the actual penalty function H_p at the arbitrary design vector $\boldsymbol{\mu}_\mathbf{d}$. To quickly determine the PMA functions, the conventional method is to construct the explicit surrogate models for the performance functions in



Determination of design vector space and random variable space for both $\boldsymbol{\mu}_\mathbf{d}$ and $\mathbf{z}$, respectively
Set k=1 and determine the starting point $\boldsymbol{\mu}_\mathbf{d}^k$
Computing the actual penalty function $H_p(\boldsymbol{\mu}_\mathbf{d}^k)$
Constructing local explicit functions for the PMA functions around $\boldsymbol{\mu}_\mathbf{d}^k$
optimizing sequetial approximate RBRDO
obtaining $\boldsymbol{\mu}_\mathbf{d}^{*k}$
Computing the actual penalty function $H_p(\boldsymbol{\mu}_\mathbf{d}^{*k})$
$\rho^k = \frac{H_p(\boldsymbol{\mu}_\mathbf{d}^k)-H_p(\boldsymbol{\mu}_\mathbf{d}^{*k})}{H_p(\boldsymbol{\mu}_\mathbf{d}^k)-\tilde{H}_p(\boldsymbol{\mu}_\mathbf{d}^{*k})}$
Stopping criterion
no
yes
Optimal design solutions
k=k+1
Adjusting the design vector $\boldsymbol{\delta}^k$ and the trust region radius vector $\boldsymbol{\mu}_\mathbf{d}^k$
Udate the design sapce
Actual simulation model

Figure 4. The efficient sequential approximate strategy of RBRDO.

Eqs. (7)-(8), respectively, with the help of limited samples. Then based on the surrogate models, solve the Eq. (7) by the optimization algorithm. Finally, the MPP and the PMA value can be obtained. To solve complex engineering problems, it's of great importance to minimize the sampling size and the evaluation number of the actual simulation model on the premise of ensuring the computation accuracy. This paper presents a fast PMA-function-computing method based on the performance function reconstruction. The main idea of this method is as follows: firstly, create the initial prediction model of the performance functions and predict approximate location of MPP by using Kriging model fitting high efficient initial samples. Secondly, increase new samples around the MPP and reconstruct the performance functions to improve the approximation precision of Kriging model at the MPP, so as to improve the computation accuracy of the PMA functions at the MPP.

Initial choice of sampling points

High dimensional model representation (HDMR) is a general set of quantitative model assessment and analysis tools for capturing the high dimensional relationships between sets of input and output model variables[23]. The samples selected in this method greatly contribute to the improvement of the modeling precision. Therefore referring to the above method, the proposed samples in U-space are:

$$\begin{gathered}(\mu_1,\mu_2,...,\mu_{i-1},U_i^j,\mu_{i+1},...,\mu_N)\\ U_i^j=-M+M/floor(M/2)*(j-1)*\sigma_i\end{gathered}\tag{26}$$

where $N=m+n\,;i=1\sim N$; μ_i and σ_i are mean and standard deviation of the random variable U_i with respect to random vector $\mathbf{U}$ at dimension i; j=1~M; the value of M is normally specified as 3,5,7 or 9. Floor() rounds the element in the bracket to the nearest integer which is less than or equal to the element itself. As for the examples illustrated in this paper, let M=3. In U-space, $\mu_i=0$, σ_i =1. Then the samples in original space can be derived from $\mathbf{z} = T^{-1}(\mathbf{U})$. Finally, compute the responses of the above samples by running the actual simulation model and combine the samples and their responses as the initial sampling set.

Reconstruction of performance function based on Kriging model

Though HDMR can well approximate the performance function in a global sense, there are still big errors with its prediction for arbitrary unknown local point, on the premise of limited sample points. In the application of this paper, the accurate MPP can be obtained by increasing new sample points according to certain strategy to approximate the surface of the performance function in the vicinity of the MPP. However, new samples and their response information are unable to be integrated into HDMR for constructing the performance function. Here Kriging model is adopted to fit the above initial sample set, so as to obtain a better overall prediction model and get the initial position of MPP with the help of Eq. (8). Amore comprehensive study on the use of extrapolation is available in the literature [24].Then compute the real response value of the performance function at MPP by actual model simulation . After that, treat the MPP and its response as the new sample and add to the above initial sample set, and repeatedly reconstruct the performance function based on Kriging model. Since Kriging model has an excellent ability of local prediction in the vicinity of a given sample, this ability are repeatedly used for reconstruction and approximating the surface of the performance function around MPP with the help of the new samples, which are also around MPP. Meanwhile, the new derived samples can gradually approximate the MPP. The steps to achieve above approach are as follows:

(1) Set t=0. Create the initial sample set $B_t = \{(\mu_1, \mu_2, ..., \mu_{i-1}, u_i^j, \mu_{i+1}, ..., \mu_N)\}$,$i = 1 \sim N, j = 1 \sim M$, and compute the output response set Y_t corresponding to B_t by the actual model simulation.

Combine the set B_t and Y_t into S_t , namely $S_t = \{B_t, Y_t\}$.

(2) Use Kriging model to fit above initial sample set S_t to obtain prediction model K_{kri}^t of the performance function.

(3) Obtain MPP, namely $\mathbf{u}_{MPP}^t$ based on the above prediction model combined with Eq. (8). Then compute the response of the performance function at $\mathbf{u}_{MPP}^t$ by actual model simulation. If t>0 and $\left\|\mathbf{u}_{MPP}^t - \mathbf{u}_{MPP}^{t-1}\right\| / \left\|\mathbf{u}_{MPP}^{t-1}\right\| \le \varepsilon_1$, $\| \partial K_{kri}^t / \partial \mathbf{u}\Big|_{\mathbf{u}=\mathbf{u}_{MPP}^t} - \partial K_{kri}^{t-1} /$

$\partial \mathbf{u}\Big|_{\mathbf{u}=\mathbf{u}_{MPP}^{t-1}} \| \le \varepsilon_2$, then turn to step （6）, otherwise turn to step（4）. For the examples presented in the following paper, take ε_1=0.01, ε_2=0.001.

(4) Treat $\mathbf{u}_{MPP}^t$ and its performance function response as a new sample and add to S_t to obtain the new set S_{t+1}. Reconstruct the performance functions based on the new set S_{t+1} to get a new prediction model K_{kri}^{t+1} .

(5) Set t=t+1, repeat step（3）.

(6) Computation stops, and get the predicted value of the PMA function at $\mathbf{u}_{MPP}^t$.

The above approach for calculation of the actual PMA function is illustrated in Figure 5.

Note that the number of the initial samples is (M-1)*N+1. While for the number of reconstruction, it is quite small. As shown by the illustrated examples in this paper, several times of reconstruction are sufficient. In other words, based on the effective initial samples to explore the global space, the presented method can quickly converge to predict the local desired MPP with a few extra samples. Hence, the presented method is quite efficient. Moreover, the sampling size for the PMA function is acceptable and adaptive for each outer updating design vector $\boldsymbol{\mu}_{\mathbf{d}}^k$ in RBRDO. While for the other conventional method using sampling methods, such as Latin Hypercube Design, Monte Carlo and so on, the choice of the sampling size is tricky. If the sampling size is too small, the constructed surrogate model will not be accurate. If too large, the overall computational resources will be costly. Hence, the presented method well overcomes this problem.

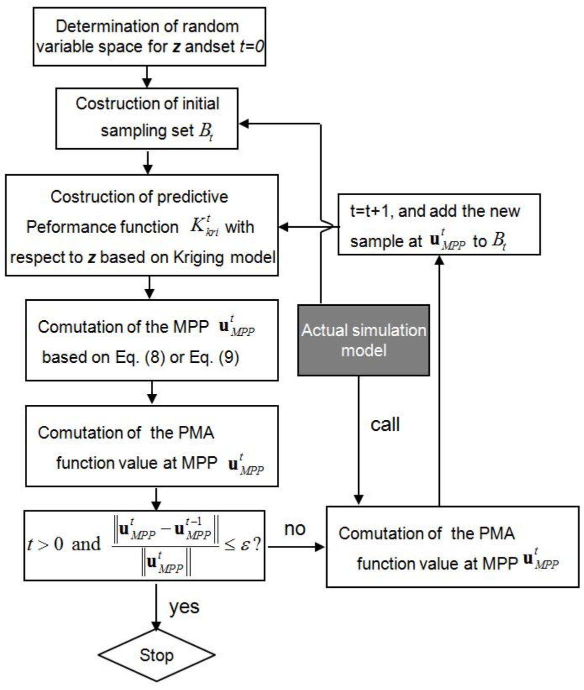

Determination of random variable space for **z** andset $t=0$
Costruction of initial sampling set B_t
Costruction of predictive Peformance function K^t_{kri} with respect to **z** based on Kriging model
t=t+1, and add the new sample at $\mathbf{u}^t_{MPP}$ to B_t
Comutation of the MPP $\mathbf{u}^t_{MPP}$ based on Eq. (8) or Eq. (9)
Actual simulation model
call
Comutation of the PMA function value at MPP $\mathbf{u}^t_{MPP}$
$t>0$ and $\frac{\|\mathbf{u}^t_{MPP}-\mathbf{u}^{t-1}_{MPP}\|}{\|\mathbf{u}^t_{MPP}\|}\le\varepsilon$?
no
Comutation of the PMA function value at MPP $\mathbf{u}^t_{MPP}$
yes
Stop

Figure 5. Calculation of the PMA function at MPP.

By using the proposed method, the PMA functions including f_{p1}, f_{p2} and $G_j(\mathbf{U}^*_j;\boldsymbol{\mu}_\mathbf{d})$ can be fast computed at the design vector $\boldsymbol{\mu}_\mathbf{d}$, based on which the actual penalty function H_p can be directly computed. It is worth mentioning that the obtained penalty function is not really an actual one in a rigorous sense. In fact, it is also an approximation. Because this approximation can be ensured to be precise enough, it is still called "actual penalty function" to distinguish from the "approximate penalty function".

NUMERICAL EXAMPLES

Application 1

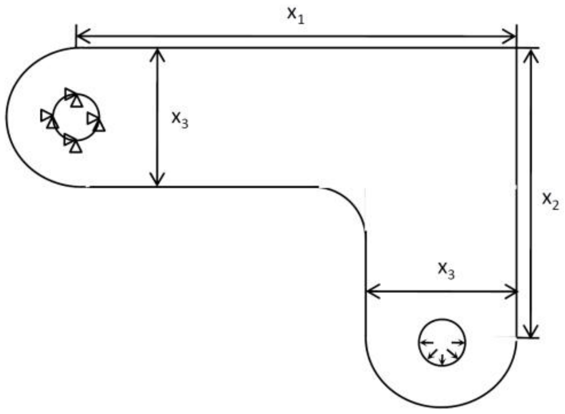


Figure 6. Angle bracket design problem.

The design of a corner angle bracket is studied using the proposed method. The bracket is made of A36 steel with Young's modulus of 210 GPa and Poisson's ratio of 0.27. The upper left pin hole is constrained (welded) around its entire circumference and a tapered pressure load is applied to the bottom of the lower right-hand pin hole with the magnitudes shown in Figure 6.The random factor considered mainly here is $\mathbf{z}=[x_1,x_2,x_3,\rho,E,Nu,P]$. All variables are assumed to have Gaussian distributions, with the mean of the former four being the design variables. The above concerned parameters are shown in Table. 1.

Table 1. Input parameters for the angle bracket design problem.

Name	Mean Value	Range of the variation (ξ)	Standard Deviation	Distribution Type
x1(mm)	μ_{x1}	±2	ξ/3	Normal
x2 (mm)	μ_{x2}	±2		Normal
x3 (mm)	μ_{x3}	±2		Normal
Density ρ (kg/m3)	7.85×10^3	$\pm2.355\times10^3$		Normal
Young's Modulus E (GPa)	210	±63		Normal
Poisson's ratio Nu	0.27	±0.081		Normal
Pressure P (Mpa)	10	±3		Normal

In size optimization, the angle bracket shall ensure minimum displacement and at the same time ensure that the maximum stress shall not be more than 250Mpa, namely:

Minimize the maximum displacement:

$$\min d_{\max}(\mathbf{d},\mathbf{X};\boldsymbol{\mu}_\mathbf{d})$$

subject to:

stress constraint

$$g_1(\mathbf{d},\mathbf{X};\boldsymbol{\mu}_\mathbf{d}) = 250 - \sigma_{\max}(\mathbf{d},\mathbf{X};\boldsymbol{\mu}_\mathbf{d}) \le 0 \tag{27}$$
$$100\text{mm}\le\mu_{x1}\le180\text{mm};80\text{mm}\le\mu_{x2}\le150\text{mm};$$
$$30\text{mm}\le\mu_{x3}\le60\text{mm}$$

In the optimization process, the factors α , ϕ , φ , κ , β^f and β^j are specified as 0.5, 0.015, 0.0029, 1000, 3 and 3, respectively. The maximum iterate number is set to 8 for the sequential optimization process. The maximum generations and the population size for PSO are specified as 100 and 10. The structure is modeled using the FEM. The initial sampling size for the construction of each PMA function is 15 (=(3-1)×7+1). Table 2 shows the optimization results. It can be found that at iteration number 8 a stationary optimal design solution is obtained, and the corresponding penalty function converges to 0.0618. At this optimal design, the values of μ_f , f_{p1} and f_{p2} are 0.0791, 0.0511and 0.1236, respectively, and the constraints are satisfied with both $g_1(\mathbf{z}_1^{*k};\boldsymbol{\mu}_\mathbf{d}^k)$ greater than 0. Additionally, Table 3 indicates that the reconstruction of the PMA functions i.e. $g_1(\mathbf{z}_1^{*k};\boldsymbol{\mu}_\mathbf{d}^k)$ is efficient, since limited times of reconstruction are OK. The total number of the FEM evaluations is 649. If solving the application by the conventional nesting optimization using PSO as the optimization operator, the computation is intensive. For the outer PSO with 100 generations and 10 individuals per generation, a total of 1000 individuals are generated and each individual represents a trial design vector. For each individual, the inner PSO will be called fourth to compute 4 PMA functions, and each call will need 1000 evaluations of the actual simulation models. Thus, the total evaluation number of the actual objective function is 1000×1000×4 = 4×10^6. As a result, the ratio of the computation cost between the two methods is 649 to 4×10^6. Thus using the presented method, the optimization efficiency can be improved exponentially. The above convergence processes are provided in Figure 7. Obviously, the presented method still has a high convergence speed.

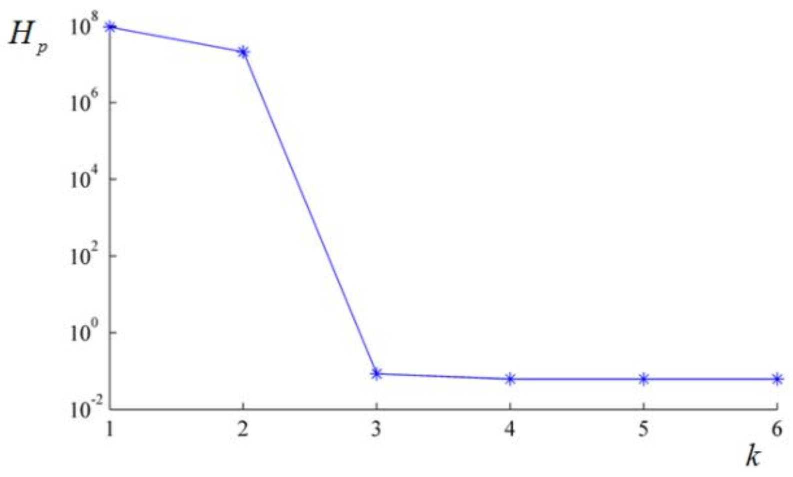


Figure 7. Iteration history of the angle bracket.

Table 2. Optimization results of the angle bracket (m).

Iteration number k	Design vector $\boldsymbol{\mu}_{\mathbf{d}}^{k}$	Penalty function $H_p(\boldsymbol{\mu}_{\mathbf{d}}^{k})$	μ_f	The value of PMA functions at $\boldsymbol{\mu}_{\mathbf{d}}^{k}$		
				f_{p1}	f_{p2}	$g_1(\mathbf{z}_1^{*k};\boldsymbol{\mu}_{\mathbf{d}}^{k})$
1	(150,100,50)	9.42×10^7	0.4190	0.2638	0.6605	-94.16
2	(135,90,55)	2.13×10^7	0.2338	0.1509	0.3653	-21.27
3	(105,110,60)	0.0848	0.1083	0.0699	0.1695	69.64
4~6	(100,80,60)	0.0618	0.0791	0.0511	0.1236	83.12

Table 3. The number of the PMA function evaluations for the angle bracket.

Iteration number k	the PMA function at $\boldsymbol{\mu}_{\mathbf{d}}^{k}$			the PMA function at $\boldsymbol{\mu}_{\mathbf{d}}^{*k}$		
	f_{p1}	f_{p2}	$g_1(\mathbf{z}_1^{*k};\boldsymbol{\mu}_{\mathbf{d}}^{k})$	f_{p1}	f_{p2}	$g_1(\mathbf{z}_1^{*k};\boldsymbol{\mu}_{\mathbf{d}}^{k})$
1	15+5	15+3	15+2	15+2	15+2	15+13
2	15+2	15+2	15+6	15+2	15+2	15+6
3	15+2	15+3	15+8	15+2	15+3	15+8
4~6	15+2	15+2	15+2	15+2	15+2	15+2

Application 2

An optimization of the piezoresistive pressure sensor is considered in this example. The sensor consists of a silicon (Si) substrate which is etched to produce a membrane and is bonded to a glass substrate. A piezoresistive layer (PZT) is applied onto the membrane near the edges to convert the mechanical stress into an electrical voltage. By measuring the voltage on the piezo, the pressure applied to the membrane may be found. Figure 8 below shows a schema of the pressure sensor.

Figure 9 shows the main size design of the pressure sensor . The random factor considered is $\mathbf{z}=[a,b,c,d,E,Nu,P]$. All variables in $\mathbf{z}$ are assumed to be normal and statistically independent. Here E, Nu and P are respectively Young modulus, Poisson's ratio and pressure. The design vector is the mean value of the former four components in $\mathbf{z}$, i.e. $\boldsymbol{\mu}_{\mathbf{d}} = [\mu_a, \mu_b, \mu_c, \mu_d]$.The above concerned parameters are shown in Table 4.

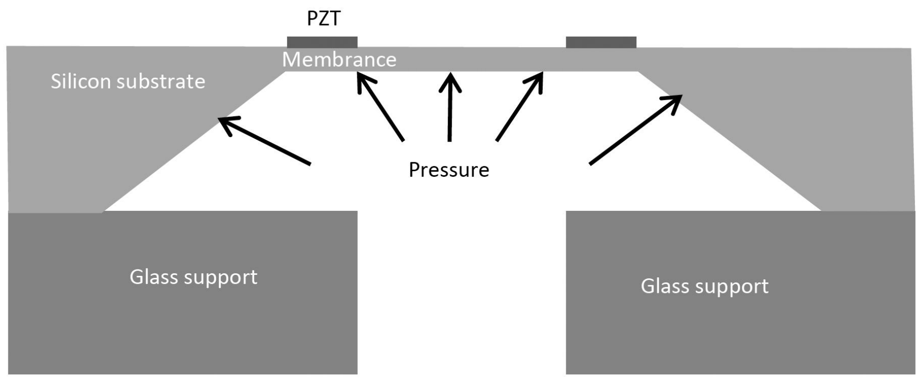


Figure 8. Schema of pressure sensor.

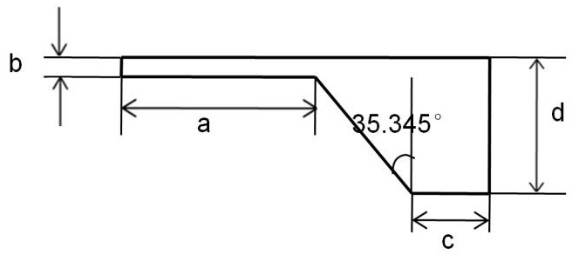


Figure 9. The main size of pressure sensor.

Table 4. Input parameters for the pressure sensor design problem.

Name	Mean Value	Range of the variation (ξ)	Standard Deviation	Distribution Type
a(m)	μ_a	$\pm 0.1\times10^{-3}$		Normal
b(m)	μ_b	$\pm 0.002\times10^{-3}$		Normal
c(m)	μ_c	$\pm 0.02\times10^{-3}$		Normal
d(m)	μ_d	$\pm 0.05\times10^{-3}$		Normal
Density ρ (kg/m^3)	2.34×10^{3}	±702	ξ/3	Normal
Young's Modulus E (MPa)	1.689×10^{11}	$\pm 5.067\times10^{10}$		Normal
Poisson's ratio Nu	0.0624	±0.01872		Normal
The pressure P (Mpa)	0.05	±0.015		Normal

The objective of the design is to minimize the stress of the pressure sensor, which will simultaneously satisfy the displacement and fundamental frequency constraints under given loads, namely: Minimize the maximum stress:

$$\min \sigma_{\max}(\mathbf{d}, \mathbf{X}; \boldsymbol{\mu}_{\mathbf{d}})$$

subject to:

displacement constraint

$$g_1(\mathbf{d}, \mathbf{X}; \boldsymbol{\mu}_{\mathbf{d}}) = 0.03 - d_{\max}(\mathbf{d}, \mathbf{X}; \boldsymbol{\mu}_{\mathbf{d}}) \le 0 \tag{28}$$

fundamental frequency constraint

$$g_2(\mathbf{d}, \mathbf{X}; \boldsymbol{\mu}_{\mathbf{d}}) = 20000 - \omega(\mathbf{d}, \mathbf{X}; \boldsymbol{\mu}_{\mathbf{d}}) \le 0$$
$$0.5\times10^{-3}\text{m} \le \mu_a \le 2\times10^{-3}\text{m};\ 0.01\times10^{-3}\text{m} \le \mu_b \le 0.05\times10^{-3}\text{m}$$
$$0.1\times10^{-3}\text{m} \le \mu_c \le 1\times10^{-3}\text{m};\ 0.3\times10^{-3}\text{m} \le \mu_d \le 0.8\times10^{-3}\text{ m}$$

Table 5. Optimization results of the pressure sensor (m).

Iteration number k	Design vector $\boldsymbol{\mu}_{\mathbf{d}}^k (10^{-3})$	Penalty function $H_p(\boldsymbol{\mu}_{\mathbf{d}}^k)$	μ_f	The value of PMA functions at $\boldsymbol{\mu}_{\mathbf{d}}^k$			
				f_{p1}	f_{p2}	$g_2(\mathbf{z}_1^{*k}; \boldsymbol{\mu}_{\mathbf{d}}^k)$	$g_2(\mathbf{z}_1^{*k}; \boldsymbol{\mu}_{\mathbf{d}}^k)$
1	(2.0,0.05,1.0,0.8)	37.55	55.88	38.43	75.10	0.0176	10243
2	(1.8,0.05,0.9,0.7975)	30.37	44.99	30.85	60.74	0.0248	7951
3	(1.8,0.05,0.9,0.7975)	30.37	44.99	30.85	60.74	0.0248	7951
4	(1.6,0.05,0.9975,0.7975)	23.95	35.48	24.33	47.90	0.0267	4733
5	(1.6,0.05,0.9975,0.7975)	30.37	35.48	24.33	47.90	0.0267	4733
6	(1.6,0.05,0.9975,0.7975)	30.37	35.48	24.33	47.90	0.0267	4733
7	(1.5,0.05,0.9988,0.7575)	21.24	31.37	21.47	42.48	0.0274	2616
8	(1.5,0.05,0.9988,0.7575)	21.24	31.37	21.47	42.48	0.0274	2616
9~15	(1.4,0.05,0.9988,0.7575)	18.70	27.52	18.79	37.40	0.0280	22.12

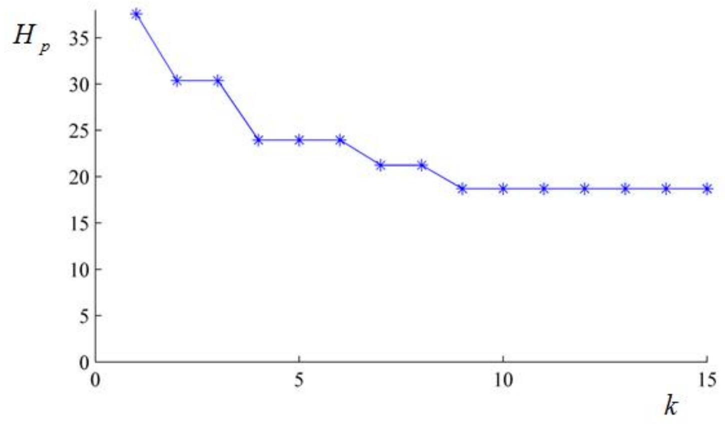


Figure 10. Iteration history of the pressure sensor design.

Table 6. The number of the PMA function evaluations for the pressure sensor.

Iteration number k	the PMA function at $\boldsymbol{\mu}_{\mathbf{d}}^k$				the PMA function at $\boldsymbol{\mu}_{\mathbf{d}}^{*k}$			
	f_{p1}	f_{p2}	$g_1(\mathbf{z}_1^{*k}; \boldsymbol{\mu}_{\mathbf{d}}^k)$	$g_2(\mathbf{z}_1^{*k}; \boldsymbol{\mu}_{\mathbf{d}}^k)$	f_{p1}	f_{p2}	$g_1(\mathbf{z}_1^{*k}; \boldsymbol{\mu}_{\mathbf{d}}^k)$	$g_2(\mathbf{z}_1^{*k}; \boldsymbol{\mu}_{\mathbf{d}}^k)$
1	17+2	17+2	17+3	17+3	17+2	17+2	17+2	17+2
2~15	17+2	17+2	17+2	17+2	17+2	17+2	17+2	17+2

The device is modeled using an axisymmetric FE model (FEM). The evaluation number of FEM in the objective function and constraints is concerned. The factors α , ϕ , φ , κ , β^f and β^j are specified as 0.5, 1, 1, 1000, 3 and 3, respectively. The maximum iterate number is set to 15 for the sequential optimization process. For PSO, the maximum generations and the population size are specified as 200 and 20. The initial sampling size for the construction of each PMA function is 17 (=(3-1)×8+1). Table 5 shows the optimization results. It can be found that at iteration number 9 a stationary optimal design solution is obtained, and the corresponding penalty function converges to 18.7. At this optimal design, the values of μ_f , f_{p1} and f_{p2} are 27.52, 18.79 and 37.40, respectively, and the constraints are satisfied with both $g_1(\mathbf{z}_1^{*k};\boldsymbol{\mu}_{\mathbf{d}}^k)$ and $g_2(\mathbf{z}_1^{*k};\boldsymbol{\mu}_{\mathbf{d}}^k)$ greater than zero. Additionally, Table 6 indicates that the reconstruction for each PMA function, i.e. $g_1(\mathbf{z}_1^{*k};\boldsymbol{\mu}_{\mathbf{d}}^k)$ and $g_2(\mathbf{z}_1^{*k};\boldsymbol{\mu}_{\mathbf{d}}^k)$ is of high efficiency, since no more than 3 times of construction is required. The total number of the FEM evaluations is 2074. If solving the application by the conventional nesting optimization using PSO as the optimization operator, the computation is quite time-consuming. For the outer PSO with 200 generations and 20 individuals per generation, a total of 4000 individuals are generated and each individual represents a trial design vector. For each individual, the inner PSO will be called fourth to compute 4 PMA functions, and each call will need 4000 evaluations of the actual simulation models. Thus, the total evaluation number of the actual objective function is $4000\times4000\times4 = 6.4\times10^7$. As a result, the ratio of the computation cost between the two methods is 2074 to 6.4×10^7. Thus using the presented method, the optimization efficiency can be improved exponentially. The above convergence processes are provided in Figure 10. Obviously, the present method still has a high convergence speed.

CONCLUSION

The paper presents an improved formulation of the RBRDO, which can be conveniently solved by the RBRDO algorithm based on high-efficient sequential approximate strategy. This new algorithm firstly transforms the RBRDO problem into ASSOP. Then a fast method to calculate the PMA functions in ASSOP is further introduced during the outer optimization process of solving ASSOP. Meanwhile, the trust region method is used to manage such sequence of the approximation sub-optimization problems, whose solutions can gradually converge to original ones. In the inner optimization, the PMAs in each sub-optimization problem are further expressed as the explicit functions of design vectors, which make each sub-optimization problem deterministic and accelerate the computation speed. Numerical examples show that the above method is quite efficient since it can effectively reduce the computation number of the actual simulation model and the method itself is applicable to integrated into the existing commercial software such as finite element software. It should be noticed that the presented method is based on a precondition of low uncertainty level. For higher level, it needs further study. Fortunately, this precondition can be commonly satisfied in practice as the uncertainty always fluctuates slightly in nominal value. In general, the presented method is of great significance for reducing the actual model computation number in practical engineering complex RBRDO problems and promoting the application of robust design methods.

ACKNOWLEDGMENTS

This work was financially supported by National Natural Science Foundation of China (Grant No. 51305143, Grant No.51205141, Grant No.51205141, Grant No. 51305142, Grant No. 51405168), Open Research Fund of Key Laboratory of High Performance Complex Manufacturing, Central South University (Grant No. Kfkt2015-11), the Start-up Foundation of High-talent Research Project from Huaqiao University (Grant No. 12BS204), Natural Science Foundation of Fujian Province of China(No. 2014J01191, No.2014J01189), the General Financial Grant from the China Postdoctoral Science Foundation(Grant No:2014M552429), project of Xiamen science and technology plan (3502Z20143041), introduction of talents Huaqiao University Scientific Research Projects(Project No.12BS217) and the Promotion Program for Young and Middle-aged Teachers in Science and Technology Research at Huaqiao University under Grant ZQN-PY212.

REFERENCES

[1] Z. Liu, S. Atamturktur, C.H. Juang, Performance based robust design optimization of steel moment resisting frames, J. Constr. Steel. Res. 89 (2013) 165–174.

[2] X.Du , A. Sudjianto, W. Chen, An integrated framework for optimization under uncertainty using inverse reliability strategy,ASME, J. Mech. Design. 126(2004) 562–570.

[3] I.Doltsinis, Z.Kang,Robust design of structures using optimization methods, Comput. Method. Appl. M. 193(2004)2221–2237.

[4] I.Doltsinis, Z.Kang, G.Cheng, Robust design of non-linear structures using optimization methods, Comput. Method. Appl. M. 194(2005)1779–1795.

[5] B.D.Youn, K.K.Choi, K.Yi, Performance moment integration (PMI) method for quality assessment in reliability-based robust optimization, Mech. Based. Des. Struc. 33(2005) 185–213.

[6] Z.P. Mourelatos, J.Liang, A methodology for trading-off performance and robustness under uncertainty, J. Mech. Design. 128(2005) 856-863.

[7] I.Lee, K.K. Choi, L. Du, D.Gorsich, Dimension reduction method for reliability-based robust design optimization, Comput. Struct, 86(2008) 1550–1562.

[8] O.P.Yadav, S.S. Bhamare, A.Rathore, Reliability-based robust design optimization: A multi-objective framework using hybrid quality loss function, Qual. Reliab. Eng. Int. 26(2010) 27-41.

[9] V.Rathod, O.P.Yadav, A.Rathore, R. Jain, Optimizing reliability-based robust design model using multi-objective genetic algorithm, Comput. Ind. Eng. 66(2013) 301–310.

[10] A.F.Shahraki, R. Noorossana, Reliability-based robust design optimization: A general methodology using genetic algorithm, Comput. Ind. Eng. 74(2014)199–207.

[11] C.Jiang, X.Han, G.P. Liu, A sequential nonlinear interval number programming method for uncertain structures, Comput. Method. Appl. M. 197(2008) 4250–4265.

[12] L.Harzheim, U. Warnecke,Robustness optimization of the position of an anti-roll bar link to avoid the toggling of a rear axle stabilizer, Struct. Multidiscip. O. 42(2010) 315–323.

[13] X.Gu, J.Lu, Reliability-based robust assessment for multiobjective optimization design of improving occupant restraint system performance, Comput. Ind. 65(2014) 1169–1180.

[14] G.Sun, G. Li, S. Zhou, H. Li, S. Hou, Q. Li, Crashworthiness design of vehicle by using multiobjective robust optimization, Struct. Multidiscip. O. 44(2011) 99-110.

[15] H.Yu, F. Gillot, M. Ichchou, Reliability based robust design optimization for tuned mass damper in passive vibration control of deterministic/uncertain structures, J. Sound. Vib. 332(2013) 2222–2238.

[16] M.Rosenblatt, Remarks on a multivariate transformation, Ann. Math. Stat. 23(1952) 470–472.

[17] T.M.Cho, B.C. Lee, Reliability-based design optimization using convex linearization and sequential optimization and reliability assessment method, Struct. Saf. 33(2011) 42–50.

[18] G.G.Dimopoulos, Mixed-variable engineering optimization based on evolutionary and social metaphors, Comput. Method. Appl. M. 196(2007) 803–817.

[19] C.Praveen, R.Duvigneau, Low cost PSO using metamodels and inexact pre-evaluation: Application to aerodynamic shape design, Comput. Method. Appl. M. 198(2009) 1087–1096.

[20] P.Wu, L.Gao, D. Zou, S. Li, An improved particle swarm optimization algorithm for reliability problems. 50(2011) 71–81.

[21] A.R.Conn, N.I. Gould, P.L. Toint, Global convergence of a Class of trust region algorithms for optimization with simple bounds SIAM. Journal on Numerical Analysis. 25(1988) 433–460.

[22] J.F.Rodriguez, J.E. Renaud, L.T. Watson, Convergence of trust region augmented Lagrangian methods using variable fidelity approximation data, Struct Optimization. 15(1998) 141–156.

[23] R.howdhury, B. Rao, Hybrid High Dimensional Model Representation for reliability analysis, Comput. Method. Appl. M. 198(2009) 753-765.

[24] S.N.ophaven, H.B. Nielsen, J.øndergaard, 2005, DACE, a Matlab kriging toolbox, http://www.imm.dtu.dk/-hbn/dace/.

[25] G.G.Wang, Adaptive response surface method using inherited latin hypercube design points, J. Mech. Des., Trans. ASME. 125(2003) 210–220.