Final version

Submitted to special issue on Selected Papers from IDETC 2019, Journal of Mechanical Design

Robustness Metric for Robust Design Optimization under Time- and Space-Dependent Uncertainty through Metamodeling

Xinpeng Wei

Graduate Research Assistant

Department of Mechanical and Aerospace Engineering

Missouri University of Science and Technology

160 Toomey Hall

400 West 13th Street

Rolla, MO 65409-0500, USA

E-mail: weixinp@mst.edu

Xiaoping Du, Ph.D.

Professor

Department of Mechanical and Energy Engineering

Indiana University – Purdue University Indianapolis

723 W. Michigan Street

Indianapolis, IN 46202-5195, USA

E-mail: duxi@iu.edu

ABSTRACT

Product performance varies with respect to time and space in many engineering applications. This paper discusses how to measure and evaluate the robustness of a product or component when its quality characteristics are functions of random variables, random fields, temporal variables, and spatial variables. At first, the existing time-dependent robustness metric is extended to the present time- and space-dependent problem. The robustness metric is derived using the extreme value of the quality characteristics with respect to temporal and spatial variables for the nominal-the-better type quality characteristics. Then a metamodel-based numerical procedure is developed to evaluate the new robustness metric. The procedure employs a Gaussian Process regression method to estimate the expected quality loss that involves the extreme quality characteristics. The expected quality loss is obtained directly during the regression model building process. Three examples are used to demonstrate the robustness analysis method. The proposed method can be used for robustness analysis during robust design optimization under time- and space-dependent uncertainty.

1. INTRODUCTION

Robust design optimization (RDO) [1] is an optimization design methodology for improving the quality of a product through minimizing the effect of the causes of variation without eliminating the causes [2]. It allows for the use of low grade materials and reduces labor and material cost while improving reliability and reducing operating cost [2]. RDO has been used to improve product quality in industrial applications [3, 4]. Over the last three decades, it has gained much attention from many research fields, such as operations research [5-7], aerospace [8, 9], structural mechanics [10, 11], vibration control [12, 13], automobile [14-16], and fatigue analysis [17, 18]. Methods to solve RDO can be roughly grouped into three categories: probabilistic methods [19-21], deterministic methods [22-26], and metamodel-based methods [27-32]. Probabilistic methods perform the robust optimization using the probability distributions of random variables. Deterministic methods incorporate a non-statistical index, such as the gradient of a response, into the optimization problem to obtain a robust optimum [32]. Metamodel-based methods employ computationally cheap surrogate models to improve the efficiency of RDO.

Robustness analysis, which evaluates and predicts the robustness of a design, is repeated for a number of times during RDO. Many metrics that measure the robustness exist in literature. The most common metric is the Taguchi's quality loss function (QLF) [2]. This metric measures not only the distance between the average quality characteristics (QCs) and their targets, but also the variation in the QCs [33]. There are also other robustness metrics, such as the signal-to-noise ratio [2], the percentile difference [34], and the worst-case QCs [35].

Most of the above robustness metrics are defined for static QCs that do not change over time and space. Some of the metrics could be used for dynamics problems, but they are only applicable for situations where the targets of QCs vary with signals [36, 37], instead of with time. To deal

with problems involving time-dependent QCs, Goethals et al. [38] proposed to use the weighted sum of mean values of a QLF at discretized time instances to measure the robustness. The weighted-sum method, however, does not take into consideration of the autocorrelation of the time-dependent QLF, which is modeled as a stochastic process. To overcome this drawback, Du [33] proposed to use the maximum value of the time-dependent QLF to measure the time-dependent robustness.

In addition to the above static and time-dependent problems, more general is the time- and space-dependent (TSD) problem [39]. In many engineering applications, QCs vary with both time and space. There are at least two reasons for the TSD QCs. (1) A QC is a function of TSD variables, such as the wind load and road conditions. (2) The QC itself is a function of temporal and spatial variables. A typical example is a wind turbine. Since the wind speed varies with time and location, it is usually modeled as a TSD random field, subjected to which, the QC of the turbine is hence TSD.

There is a need to define a new robustness metric for the optimization involving TSD problems. The object of this work is to derive a robustness metric for TSD problems and develop a numerical method to evaluate it. We use the expectation of the maximum value of a TSD QLF to measure the robustness. For the former, we employ the same strategy in [33], and for the latter we use a metamodeling method to manage the computational efficiency because of the involvement of the expensive multidimensional global optimization [40-43] with respect to temporal and spatial parameters. An efficient method based on the Gaussian process model [44-47] is then proposed. The contributions of this work are twofold. First, a TSD robustness metric is defined. It can take into consideration of all information of the TSD QLF, including its autocorrelation. Therefore, it

is mathematically a rigorous metric for the TSD problems. Second, a Gaussian process based method is developed to effectively compute the TSD robustness metric.

The proposed TSD robustness metric is actually an extension of the time-dependent robustness metric proposed in [33]. The similarity is that both the proposed TSD robustness metric and the time-dependent robustness metric use the maximum value of the QLF to measure the robustness. However, this study deals with a more general and complicate problem because the time-dependent problem is only a special case of the TSD problem. From the perspective of mathematical models, the new robustness metric needs the multidimensional global optimization with respect to both temporal and spatial parameters, while the time-dependent one involves unidimensional global optimizations with respect to only a temporal parameter. In addition, the new QLF may include random fields in its input.

The paper is organized as follows. Section 2 briefly reviews the time-dependent robustness metric, whose extension to TSD problems is discussed with a new robustness metric in Section 3, followed by a meta-modeling numerical procedure for the new metric in Section 4. Four examples are given in Section 5, and conclusions are provided in Section 6.

2. REVIEW OF STATIC AND TIME-DEPENDENT ROBUSTNESS METRICS

Nominal-the-best, smaller-the-better, and larger-the-better are three types of QCs [33]. In this work, we only focus on the nominal-the-best type. The discussions, however, can be extended to the other two types.

2.1. Static robustness metric

The most common robustness metric is the QLF. Let a QC be defined as

$$Y = g(\mathbf{X}) \tag{1}$$

where $\mathbf{X} = (X_1, X_2 \ldots, X_N)$ are N input random variables. Then the QLF is

$$L = A(Y - m)^2 \tag{2}$$

where m is the target value of Y, and A is a constant determined by a monetary loss. The robustness is measured by the expectation, or the mean E_L of L, which is calculated by

$$E_L = A[(\mu_Y - m)^2 + \sigma_Y^2] \tag{3}$$

where μ_Y and σ_Y are the mean and standard deviation of Y, respectively. The smaller is E_L, the better is the robustness because μ_Y (the average QC) is closer to the target m and σ_Y (variation of the QC) is smaller.

2.2. Time-dependent robustness metric

A time-dependent QC is given by

$$Y = g(\mathbf{X}, t) \tag{4}$$

Note that the input of $g(\cdot)$ may also include random processes, which can be transformed into functions with respect to random variables and t [48]. Thus Eq. (4) does not lose generality. At instant t, the QLF is given as

$$L(t) = A(t)[Y - m(t)]^2 = A(t)[g(\mathbf{X}, t) - m(t)]^2 \tag{5}$$

$L(t)$ can measure only the quality loss at a specific time instant t and is thus called point quality loss function (P-QLF). To measure the quality loss of a product over a time interval $[\underline{t}, \overline{t}]$, Du [33] proposed to use the extreme value or the worst-case value of $L(t)$ over $[\underline{t}, \overline{t}]$. The worst-case quality loss is called interval quality loss function (I-QLF) and is given by

$$L(\underline{t}, \overline{t}) = \max_{t \in [\underline{t}, \overline{t}]} L(t) = \max_{t \in [\underline{t}, \overline{t}]} \{A(t)[g(\mathbf{X}, t) - m(t)]^2\} \tag{6}$$

Note that $L(\underline{t}, \overline{t})$ is a random variable while $L(t)$ is a random process. Like static problems, the expectation $E_L(\underline{t}, \overline{t})$ of $L(\underline{t}, \overline{t})$ is also used as the time-dependent robustness metric given by

$$E_L(\underline{t}, \overline{t}) = \mathrm{E}\left[L(\underline{t}, \overline{t})\right] \tag{7}$$

where $\mathrm{E}(\cdot)$ stands for expectation. Minimizing $E_L(\underline{t}, \overline{t})$ reduces both the deviation of the QC from its target and the variation in the QC over time interval $[\underline{t}, \overline{t}]$. When $\mathbf{X}$ is fixed to a specific realization $\mathbf{x}$, Eq. (6) actually shows a unidimensional global optimization problem. Multiple samples of $L(\underline{t}, \overline{t})$ are necessary to calculate $E_L(\underline{t}, \overline{t})$ using Eq. (7), and hence multiple unidimensional global optimizations is required to obtain $E_L(\underline{t}, \overline{t})$.

3. A NEW ROBUSTNESS METRIC FOR TIME- AND SPACE-DEPENDENT QCS

In TSD problems, in addition to random variables and random processes, static random fields and time-dependent random fields are also involved. For convenience, we do not distinguish random processes, static random fields or time-dependent random fields. In this paper, we generally call them random fields. Let $\mathbf{Z} = (\mathrm{S}_1, \mathrm{S}_2, \mathrm{S}_3, t)$ be the vector comprising the three spatial parameters (*x*-, *y*-, and *z*-coordinates) and the time. Note that for problems in one-dimensional and two-dimensional space, $\mathbf{Z} = (\mathrm{S}_1, t)$ and $\mathbf{Z} = (\mathrm{S}_1, \mathrm{S}_2, t)$, respectively. Also note that random fields can be transformed into functions with respect to random variables and $\mathbf{Z}$ [48]. Without loss of generality, a TSD QC is then given by

$$Y = g(\mathbf{X}, \mathbf{Z}) \tag{8}$$

With the TSD QC, the QLF is given by

$$L(\mathbf{X}, \mathbf{Z}) = A(\mathbf{Z})[Y - m(\mathbf{Z})]^2 = A(\mathbf{Z})[g(\mathbf{X}, \mathbf{Z}) - m(\mathbf{Z})]^2 \tag{9}$$

$L(\mathbf{X}, \mathbf{Z})$ measures the qualify loss at any specific point $\mathbf{z} \in \Omega$, where Ω is the domain of $\mathbf{Z}$, and it is also a P-QLF.

Before defining the TSD robustness metric, we propose some criteria of robustness metrics for the TSD problems, inspired by the criteria of the robustness metrics for time-dependent problems given in [33]. The criteria are as follows:

(a) The metric must represent the maximum quality loss over Ω. This reflects the fact that the quality loss is not reversible. If a quality loss, including the maximum quality loss, has occurred, there is no way to turn back.

(b) The metric should increase or at least stay the same with the expansion of Ω, given that other conditions stay unchanged. The reason is that when a product involves a larger space and/or is put into service for a longer period of time, the robustness should be worse or at least the same.

(c) The metric should capture the autocorrelation of the P-QLF $L(\mathbf{X}, \mathbf{Z})$ over Ω. Since $L(\mathbf{X}, \mathbf{Z})$ is a random field, its autocorrelation is an important property. Two different random fields with the same marginal distribution at any point may have very different performances if they do not share the same autocorrelation.

(d) Minimizing the metric will lead to optimizing the mean QCs and minimizing the variations of the QCs over Ω. This criterion comes from the purpose of the robust optimization [49].

Based on the above criteria, we define the TSD robustness metric $E_L(\Omega)$ as

$$E_L(\Omega) = \mathrm{E}[L_{\max}(\mathbf{X}, \Omega)] \tag{10}$$

where

$$L_{\max}(\mathbf{X}, \Omega) = \max_{\mathbf{z} \in \Omega} L(\mathbf{X}, \mathbf{z}) \tag{11}$$

is the maximum value of $L(\mathbf{X}, \mathbf{Z})$ and is called the domain quality loss function (D-QLF). The definition of $L_{\max}(\mathbf{X}, \Omega)$ ensures that $E_L(\Omega)$ meet Criterion (a) naturally. Let $\acute{\Omega} \subset \Omega$, then it is obvious that

$$L_{\max}\left(\mathbf{x}, \acute{\Omega}\right) \leq L_{\max}(\mathbf{x}, \Omega) \tag{12}$$

and hence $E_L\left(\acute{\Omega}\right) \leq E_L(\Omega)$. Therefore, $E_L(\Omega)$ meets Criterion (b). Since $L_{\max}(\mathbf{X}, \Omega)$ is the maximum value distribution [50, 51] of $L(\mathbf{X}, \mathbf{Z})$, the autocorrelation of $L(\mathbf{X}, \mathbf{Z})$ is necessary for computing $L_{\max}(\mathbf{X}, \Omega)$. Different autocorrelation functions of $L(\mathbf{X}, \mathbf{Z})$ will lead to different distributions of $L_{\max}(\mathbf{X}, \Omega)$, and hence $E_L(\Omega)$ can capture the autocorrelation of $L(\mathbf{X}, \mathbf{Z})$, indicating that $E_L(\Omega)$ meets Criterion (c). Since $L(\mathbf{X}, \mathbf{Z})$, and hence $L_{\max}(\mathbf{X}, \Omega)$ and $E_L(\Omega)$, are nonnegative, minimizing $E_L(\Omega)$ requires that the QC $g(\mathbf{X}, \mathbf{Z})$ gets close to its target $m(\mathbf{Z})$ as much as possible. Therefore, $E_L(\Omega)$ also meets Criterion (d).

4. A META-MODLING APPROACH TO ROBUSTNESS ANALYSIS

The robustness metric defined in Eq. (10) involves the extreme value of a general random field. It is not an easy task to evaluate the robustness metric for a given design. In this section, we discuss our proposal numerical method for the robustness analysis.

4.1. Overview of the proposed robustness analysis

The main idea of the proposed robustness analysis method is to train a Gaussian process model $\hat{L}(\mathbf{X}, \mathbf{Z})$ for $L(\mathbf{X}, \mathbf{Z})$. Replacing $L(\mathbf{X}, \mathbf{Z})$ in Eq. (11) with $\hat{L}(\mathbf{X}, \mathbf{Z})$, we can approximate $L_{\max}(\mathbf{X}, \Omega)$ with $\hat{L}_{\max}(\mathbf{X}, \Omega)$ as follows:

$$\hat{L}_{\max}(\mathbf{X}, \Omega) = \max_{\mathbf{z} \in \Omega} \hat{L}(\mathbf{X}, \mathbf{z}) \tag{13}$$

Then the Monte Carlo simulation (MCS) [52] is used to compute $E_L(\Omega)$ by

$$E_L(\Omega) = \frac{1}{n_{\mathrm{MCS}}} \sum_{i=1}^{n_{\mathrm{MCS}}} \hat{L}_{\max}\left(\mathbf{x}^{(i)}, \Omega\right) \tag{14}$$

where n_{MCS} is the sample size, and $\mathbf{x}^{(i)}$ is the *i*-th sample of $\mathbf{X}$. Since $\hat{L}(\mathbf{X}, \mathbf{Z})$ is computationally much cheaper than $L(\mathbf{X}, \mathbf{Z})$, the proposed method can significantly improve the efficiency. Generally, a larger number of training points of $L(\mathbf{X}, \mathbf{Z})$ is preferred to train $\hat{L}(\mathbf{X}, \mathbf{Z})$ for higher accuracy, but the efficiency will decrease because $L(\mathbf{X}, \mathbf{Z})$ in engineering applications is often a black-box function whose evaluation needs expensive numerical procedures or simulations [53].

To balance the accuracy and the efficiency, we do not require $\hat{L}(\mathbf{X}, \mathbf{Z})$ to be accurate globally. Instead, we only need it to be locally accurate at samples of $\mathbf{X}$ in Eq. (14). To this end, we employ the efficient global optimization (EGO) [54, 55] to adaptively add training points to update $\hat{L}(\mathbf{X}, \mathbf{Z})$.

To have a quick overview to the proposed method, we give a simplified flowchart in Fig. 1. There are in total eight steps in the proposed method. Details of Step 2 will be given in Subsection 4.2. The EGO, which comprises Steps 3 through 5, will be detailed in Subsection 4.3. We propose two stopping criteria in Steps 4 and 7. Detailed information is given in Subsection 4.4. The implementation of the algorithm and the detailed flowchart will be given in Subsection 4.5. In Subsection 4.6, we discuss how to deal with a more general problem that involves random fields.

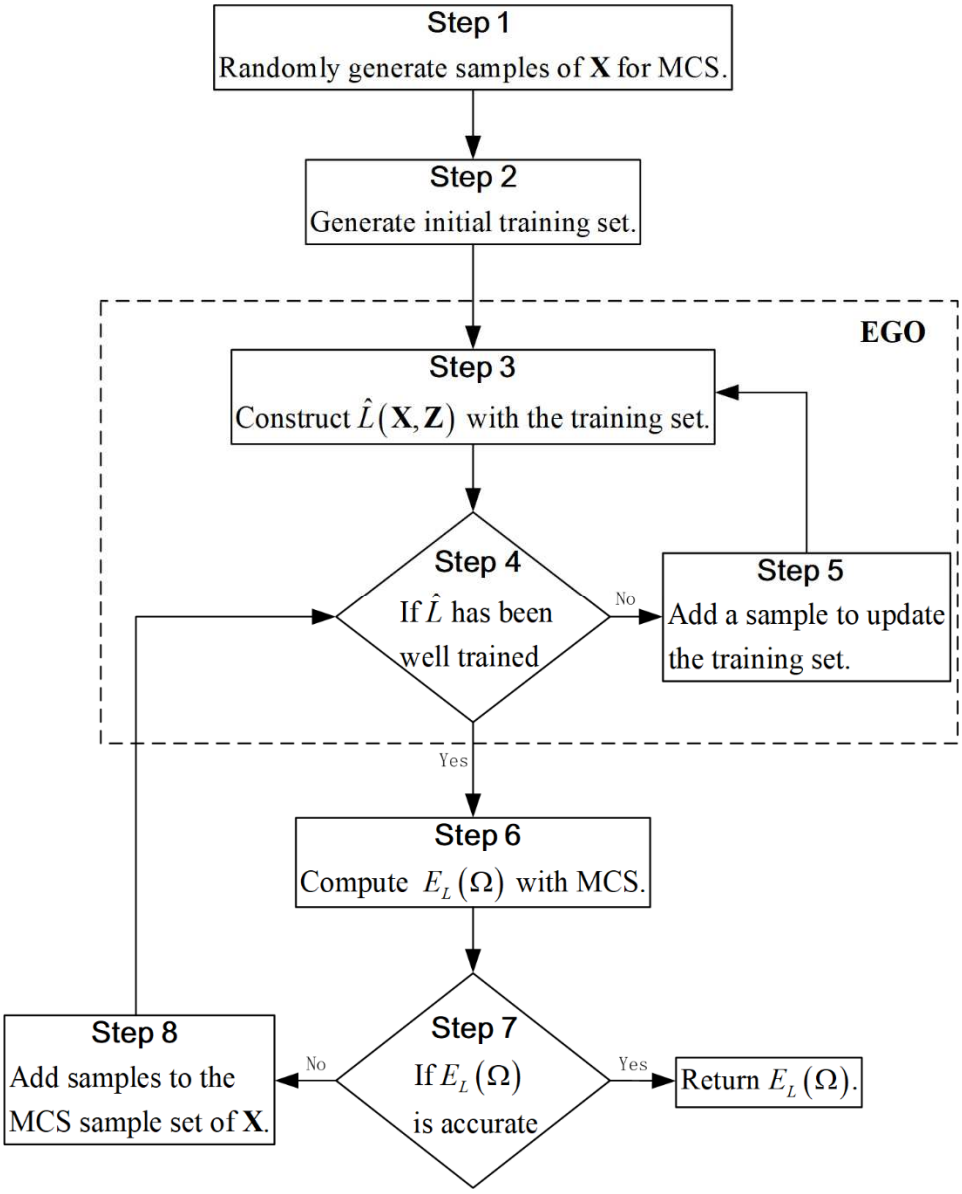

Fig. 1 Simplified flowchart

4.2 Initial training set

The principle of generating initial training set for building a Gaussian process model is to spread the initial training points evenly. Commonly used sampling methods include random sampling, Latin hypercube sampling and Hammersley sampling [56]. In this study, we employ the

Hammersley sampling method because it has better uniformity properties over a multidimensional space [57].

Since the dimension of the entire input vector $(\mathbf{X}, \mathbf{Z})$ is $N + N_{\mathrm{Z}}$, where N_{Z} is the dimension of $\mathbf{Z}$, the Hammersley sampling method generates initial training points in a hypercube $[0,1]^{N+N_{\mathrm{Z}}}$. To get initial training points of $\mathbf{X}$, we can simply use the inverse probability method to transform the training points from the hypercube space to the X-space. As for the initial training points of $\mathbf{Z}$, we treat all components of $\mathbf{Z}$ as if they were independent uniform random variables and then also use the inverse probability method to transform the training points from the hypercube space to the Z-space.

Samples of a row random vector are assembled into a matrix. For example, the initial training points $\mathbf{x}^{\mathrm{in}}$ of $\mathbf{X} = (X_1, X_2 \ldots, X_N)$ are

$$\mathbf{x}^{\mathrm{in}} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \ddots & x_N^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \ddots & x_N^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(n_{\mathrm{in}})} & x_2^{(n_{\mathrm{in}})} & \ddots & x_N^{(n_{\mathrm{in}})} \end{bmatrix} \tag{15}$$

where n_{in} is the total number of initial training points. With $\mathbf{x}^{\mathrm{in}}$ and the initial training points $\mathbf{z}^{\mathrm{in}}$ of $\mathbf{Z}$, we then obtain initial training points $\mathbf{l}^{\mathrm{in}}$ of $L(\mathbf{X}, \mathbf{Z})$ by calling Eq. (9). Finally, we get the initial training set $(\mathbf{x}^{\mathrm{trn}}, \mathbf{z}^{\mathrm{trn}}, \mathbf{l}^{\mathrm{trn}}) = \left(\mathbf{x}^{\mathrm{in}}, \mathbf{z}^{\mathrm{in}}, \mathbf{l}^{\mathrm{in}}\right)$, where the superscript *trn* represents training points.

4.3 Employment of EGO

EGO is based on the Gaussian process model. With the training set $(\mathbf{x}^{\mathrm{trn}}, \mathbf{z}^{\mathrm{trn}}, \mathbf{l}^{\mathrm{trn}})$ we can build $\hat{L}(\mathbf{X}, \mathbf{Z})$. Because only a limit number of training points are used, $\hat{L}(\mathbf{X}, \mathbf{Z})$ has model uncertainty (or epistemic uncertainty), which is measured by $\sigma(\mathbf{X}, \mathbf{Z})$.

Practically, when $\hat{L}(\mathbf{X}, \mathbf{Z})$ is available, we need to discretize Ω to compute the maximum value $\hat{L}_{\max}(\mathbf{X}, \Omega)$ with Eq. (13). If we discretize Z_j, the j-th element of $\mathbf{Z}$, into m_j points, then Ω will be discretized into $n_\Omega = \prod_{i=1}^{N_\mathrm{Z}} m_j$ points. For convenience, we denote the n_Ω points of $\mathbf{Z}$ by $\mathbf{z}^\Omega$, whose dimension is $n_\Omega \times N_\mathrm{Z}$. Then Eq. (13) is rewritten as

$$\hat{L}_{\max}(\mathbf{X}, \Omega) = \max_{\mathbf{z} \in \mathbf{z}^\Omega} \hat{L}(\mathbf{X}, \mathbf{z}) \tag{16}$$

Since $\hat{L}_{\max}(\mathbf{X}, \Omega)$ may not be the exact global maximum, we need to add training points of $(\mathbf{X}, \mathbf{Z}, L)$ to update $\hat{L}(\mathbf{X}, \mathbf{Z})$ so that the $\hat{L}_{\max}(\mathbf{X}, \Omega)$ will be more accurate. To determine how to add a new training point, we use the well-known expected improvement (EI) learning function [55] given by

$$\mathrm{EI}(\mathbf{x}, \mathbf{z}) = \left(\hat{L} - \hat{L}_{\max}\right)\Phi\left(\frac{\hat{L} - \hat{L}_{\max}}{\sigma(\mathbf{x}, \mathbf{z})}\right) + \sigma(\mathbf{x}, \mathbf{z})\varphi\left(\frac{\hat{L} - \hat{L}_{\max}}{\sigma(\mathbf{x}, \mathbf{z})}\right) \tag{17}$$

where $\hat{L} = \hat{L}(\mathbf{x}, \mathbf{z})$ and $\hat{L}_{\max} = \hat{L}_{\max}(\mathbf{x}, \Omega)$; $\Phi(\cdot)$ and $\varphi(\cdot)$ are the cumulative distribution function and probability density function of a standard Gaussian variable, respectively. $\mathrm{EI}(\mathbf{x}, \mathbf{z})$ means that the exact $L_{\max}(\mathbf{x}, \Omega)$ is expected to be $\mathrm{EI}(\mathbf{x}, \mathbf{z})$ larger than the current $\hat{L}_{\max}(\mathbf{x}, \Omega)$. In other words, if we add a training point at $(\mathbf{x}, \mathbf{z})$ to update $\hat{L}(\mathbf{X}, \mathbf{Z})$, we expect to update current $\hat{L}_{\max}(\mathbf{x}, \Omega)$ to $\hat{L}_{\max}(\mathbf{x}, \Omega) + \mathrm{EI}(\mathbf{x}, \mathbf{z})$. In principle, we should update $\hat{L}_{\max}(\mathbf{x}, \Omega)$ by a step size as large as possible so that the algorithm converges quickly. Therefore, we determine the next training point $\left(\mathbf{x}^{(\mathrm{next})}, \mathbf{z}^{(\mathrm{next})}\right)$ by

$$\left(\mathbf{x}^{(\mathrm{next})}, \mathbf{z}^{(\mathrm{next})}\right) = \arg \max_{\mathbf{x} \in \mathbf{x}^{\mathrm{MCS}},\ \mathbf{z} \in \mathbf{z}^\Omega} \mathrm{EI}(\mathbf{x}, \mathbf{z}) \tag{18}$$

where $\mathbf{x}^{\mathrm{MCS}}$ represents the MCS population of $\mathbf{X}$. Eq. (18) indicates a double-layer optimization. The reason is that whenever we want to optimize $\mathbf{Z}$ in Eq. (18), we must fix $\mathbf{X}$ to a specific

realization $\mathbf{x}$ so that Eq. (16) can be used to calculate $\hat{L}_{\max}$. With Eq. (9), we can obtain the next training point $\mathrm{l}^{(\mathrm{next})}$ of $L(\mathbf{X}, \mathbf{Z})$. Then the training set $(\mathbf{x}^{\mathrm{trn}}, \mathbf{z}^{\mathrm{trn}}, \mathbf{l}^{\mathrm{trn}})$ is updated through

$$\begin{cases} \mathbf{x}^{\mathrm{trn}} = \begin{bmatrix} \mathbf{x}^{\mathrm{trn}} \\ \mathbf{x}^{(\mathrm{next})} \end{bmatrix} \\ \mathbf{z}^{\mathrm{trn}} = \begin{bmatrix} \mathbf{z}^{\mathrm{trn}} \\ \mathbf{z}^{(\mathrm{next})} \end{bmatrix} \\ \mathbf{l}^{\mathrm{trn}} = \begin{bmatrix} \mathbf{l}^{\mathrm{trn}} \\ \mathrm{l}^{(\mathrm{next})} \end{bmatrix} \end{cases} \tag{19}$$

The updated training set $(\mathbf{x}^{\mathrm{trn}}, \mathbf{z}^{\mathrm{trn}}, \mathbf{l}^{\mathrm{trn}})$ is used to refine $\hat{L}(\mathbf{X}, \mathbf{Z})$. Then $\hat{L}_{\max}(\mathbf{X}, \Omega)$ in Eq. (16) and hence $E_L(\Omega)$ in Eq. (14) are also updated. With similar procedures, training points are iteratively added into the training set, and $E_L(\Omega)$ is updated iteratively until stopping criteria are satisfied.

4.4. Stopping criteria

In this subsection, we discuss two stopping criteria in Steps 4 and 7 shown in Fig. 1. The purpose of the stopping criterion in Step 4 is to judge whether more training points are necessary to update $\hat{L}(\mathbf{X}, \mathbf{Z})$. A straightforward stopping criterion is

$$\max_{\mathbf{x} \in \mathbf{x}^{\mathrm{MCS}},\ \mathbf{z} \in \mathbf{z}^{\Omega}} \left| \mathrm{EI}(\mathbf{x}, \mathbf{z}) / \hat{L}_{\max}(\mathbf{x}, \Omega) \right| \le c \tag{20}$$

where c is a threshold, which usually takes a small positive number, such as 0.005. This stopping criterion guarantees that for any $\mathbf{x} \in \mathbf{x}^{\mathrm{MCS}}$, the absolute value of the expected improvement rate of $\hat{L}_{\max}(\mathbf{x}, \Omega)$ is small enough. In other words, this stopping criterion guarantees that the n_{MCS} samples of $\hat{L}_{\max}(\mathbf{X}, \Omega)$ are all accurate enough so that $E_L(\Omega)$ is accurate enough. The threshold c, however, does not directly measure the accuracy of $E_L(\Omega)$. As a result, it is hard to determine a proper value for c. If we set a too small value to c, it may result in unnecessary iterations and hence an unnecessary computational cost. To resolve this problem, we propose a new stopping criterion

$$W = \left| \left\{ \underset{\mathbf{x} \in \mathbf{x}^{\mathrm{MCS}}}{\mathrm{mean}} \left[\max_{\mathbf{z} \in \mathbf{z}^{\Omega}} \mathrm{EI}(\mathbf{x}, \mathbf{z}) \right] \right\} / E_L(\Omega) \right| \leq w \tag{21}$$

where w is another threshold, which usually takes a small positive number, such as 0.005. Since $\max_{\mathbf{z} \in \mathbf{z}^{\Omega}} \mathrm{EI}(\mathbf{x}, \mathbf{z})$ is the maximum expected improvement of $\hat{L}_{\max}(\mathbf{x}, \Omega)$, $\underset{\mathbf{x} \in \mathbf{x}^{\mathrm{MCS}}}{\mathrm{mean}} \left[\max_{\mathbf{z} \in \mathbf{z}^{\Omega}} \mathrm{EI}(\mathbf{x}, \mathbf{z}) \right]$ is the expected maximum improvement of $E_L(\Omega)$. Then, W is the absolute value of the expected improvement rate of $E_L(\Omega)$. W directly measures the accuracy of $E_L(\Omega)$, and so we can set the value of w according to specific engineering requirement. For example, if we set $w = 0.005$, no more training points will be added if adding more training points can change current $E_L(\Omega)$ by no more than 0.5%. As a result, if n_{MCS} is sufficiently large, the relative error of the obtained $E_L(\Omega)$ is expected to be between -0.5% and 0.5%.

Step 7 mainly deals with the following question: How many samples of $\hat{L}_{\max}(\mathbf{X}, \Omega)$ are enough to obtain accurate $E_L(\Omega)$? Since $L_{\max}(\mathbf{X}, \Omega)$ is a random variable, the sample size needed to estimate its mean value $E_L(\Omega)$ is dependent on the standard deviation $\sigma(\Omega)$ of $L_{\max}(\mathbf{X}, \Omega)$. Since the sample size is n_{MCS}, the deviation coefficient Γ of $E_L(\Omega)$ is

$$\Gamma = \frac{\sigma(\Omega)}{E_L(\Omega)\sqrt{n_{\mathrm{MCS}}}} \tag{22}$$

where $E_L(\Omega)$ is estimated by Eq. (14), and $\sigma(\Omega)$ is estimated by

$$\sigma(\Omega) = \sqrt{\frac{1}{n_{\mathrm{MCS}} - 1} \sum_{i=1}^{n_{\mathrm{MCS}}} \left[\hat{L}_{\max}(\mathbf{x}^{(i)}, \Omega) - E_L(\Omega) \right]^2} \tag{23}$$

Eq. (22) shows that the larger is n_{MCS}, the smaller Γ will we obtain. A smaller Γ means that the estimated $E_L(\Omega)$ is more accurate. Therefore, we use the following stopping criterion in Step 7:

$$\Gamma \leq \gamma \tag{24}$$

where γ is a threshold, which usually takes a small positive number, such as 0.005.

If the stopping criterion in Eq. (24) is not satisfied, how many samples do we need to add to the current sample set $\mathbf{x}^{\mathrm{MCS}}$? Combining Eq. (22) and Eq. (24), we have

$$n_{\mathrm{MCS}} \geq \left[\frac{\sigma(\Omega)}{E_L(\Omega)\gamma}\right]^2 \tag{25}$$

It means that to meet the stopping criterion in Eq. (24), we should use a sample size at least $\left[\frac{\sigma(\Omega)}{E_L(\Omega)\gamma}\right]^2$. For convenience, let $n_0 = \mathrm{round}\left\{\left[\frac{\sigma(\Omega)}{E_L(\Omega)\gamma}\right]^2\right\}$, where $\mathrm{round}(\cdot)$ represents the operation to get the nearest integer. Then the number n_{add} of samples we should add to the current sample set $\mathbf{x}^{\mathrm{MCS}}$ is

$$n_{\mathrm{add}} = n_0 - n_{\mathrm{MCS}} \tag{26}$$

However, when $\hat{L}(\mathbf{X}, \mathbf{Z})$ is too rough at the first several training iterations, both $E_L(\Omega)$ and $\sigma(\Omega)$ may have very poor accuracy. As a result, n_{add} determined by Eq. (26) may be misleading. To resolve this problem, we set a threshold $\tilde{n}_{\mathrm{add}}$ for n_{add}. Then n_{add} is modified to

$$n_{\mathrm{add}} = \begin{cases} \tilde{n}_{\mathrm{add}}, \text{if } n_0 - n_{\mathrm{MCS}} > \tilde{n}_{\mathrm{add}} \\ n_0 - n_{\mathrm{MCS}}, \text{otherwise} \end{cases} \tag{27}$$

4.5 Implementation

In this subsection, we give the detailed procedure of the proposed method. The detailed flowchart is shown in Fig. 2. The total number n_{call} of function evaluations in Eq. (9) is used to measure the main computational cost of the proposed method, since Eq. (9) usually involves the computation of an expensive black-box function.

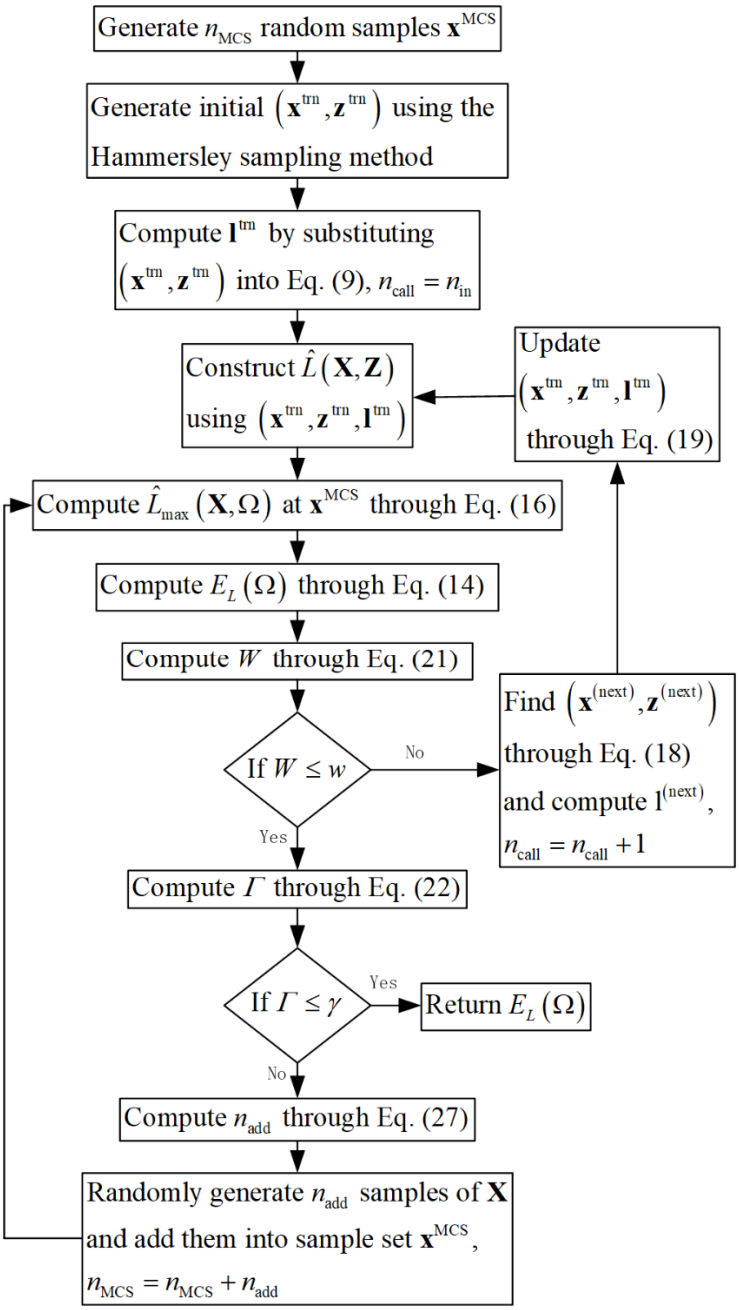

Fig. 2 Detailed flowchart

The strategy of the extreme value in this study is similar to what the nested extreme response surface approach [58] employs, because both methods use the same EGO to solve the global optimization problem. But the problem in the former method is the multidimensional global optimization with respect to time and space while the problem in the latter method is a

unidimensional one with respect to time. As a result, the learning functions and stopping criteria of the two methods are different.

4.6 Extension to problems with input random fields

When the TSD QC $g(\cdot)$ involves input random fields, it is straightforward to use the series expansion of the random fields so that the above implementation of the proposed method still holds. For example, a QC is given as

$$Y = g(\mathbf{X}, \mathbf{H}(\mathbf{Z}), \mathbf{Z}) \tag{28}$$

where $\mathbf{H}(\mathbf{Z})$ is a vector of random fields. To easily present the idea, we assume there is only one random filed, given by $H(\mathbf{Z})$. Widely used series expansions for random fields include, but are not limited to, the Karhunen-Loeve series expansion (K-L), the orthogonal series expansion (OSE), and the expansion optimal linear estimation method (EOLE) [48]. Since Ω is discretized into $\mathbf{z}^{\Omega}$, the autocorrelation coefficient function of $H(\mathbf{Z})$ is discretized into the autocorrelation coefficient matrix $\mathbf{M}_H$ with dimension $n_\Omega \times n_\Omega$. Then the EOLE expansion $H(\boldsymbol{\xi}, \mathbf{Z})$ of $H(\mathbf{Z})$ is given by

$$H(\boldsymbol{\xi}, \mathbf{z}) = \mu_H(\mathbf{z}) + \sigma_H(\mathbf{z}) \sum_{k=1}^{n_\Omega} \frac{\xi_k}{\sqrt{\lambda_k}} \mathbf{V}_k \mathbf{M}_H(:, k), \mathbf{z} \in \mathbf{z}^{\Omega} \tag{29}$$

where $\mu_H(\mathbf{z})$ is the mean value function of $H(\mathbf{Z})$, $\sigma_H(\mathbf{z})$ is the standard deviation function of $H(\mathbf{Z})$, $\xi_k, k = 1,2, \dots, n_\Omega$ are n_Ω independent standard Gaussian variables, λ_k is the k-th eigenvalue of $\mathbf{M}_H$, $\mathbf{V}_k$ is the k-th (row) eigenvector of $\mathbf{M}_H$, and $\mathbf{M}_H(:, k)$ is the k-th column of $\mathbf{M}_H$. Note that the eigenvalues are sorted from the largest to the smallest. Usually only the first p ($p \le n_\Omega$) eigenvalues are significant. Therefore, Eq. (29) is practically truncated, and only the first p orders are kept:

$$H(\boldsymbol{\xi}, \mathbf{z}) = \mu_H(\mathbf{z}) + \sigma_H(\mathbf{z}) \sum_{k=1}^{p} \frac{\xi_k}{\sqrt{\lambda_k}} \mathbf{V}_k \mathbf{M}_H(:, k), \mathbf{z} \in \mathbf{z}^{\Omega} \tag{30}$$

Then the dimension of $\boldsymbol{\xi}$ is p. With the expansion, Eq. (28) is rewritten as

$$Y = g[\mathbf{X}, H(\boldsymbol{\xi}, \mathbf{Z}), \mathbf{Z}] \tag{31}$$

or equivalently as

$$Y = g(\widetilde{\mathbf{X}}, \mathbf{Z}) \tag{32}$$

where $\widetilde{\mathbf{X}} = (\boldsymbol{\xi}, \mathbf{X})$. Eq. (32) shares the same format with Eq. (8), and hence the above implementation in Subsection 4.5 is also applicable.

The direct implementation this way, however, may suffer from the curse of dimensionality. Since many random variables, i.e. $\boldsymbol{\xi}$, are in the series expansion $H(\boldsymbol{\xi}, \mathbf{Z})$, the dimension of $\boldsymbol{\xi}$ and hence that of $g(\widetilde{\mathbf{X}}, \mathbf{Z})$ is high. As a result, the dimension of the surrogate model $\hat{L}(\widetilde{\mathbf{X}}, \mathbf{Z})$ is also high. The high-dimensional surrogate model has as least two drawbacks. First, it is not cheap anymore, losing its expected advantages. Second, more training points are needed for an acceptable accuracy. To overcome the drawbacks, we build a surrogate model $\hat{L}(\mathbf{X}, H, \mathbf{Z})$ with respect to $\mathbf{X}, H$ and $\mathbf{Z}$. Note that the entire random field H is treated as only one variable for $\hat{L}(\mathbf{X}, H, \mathbf{Z})$. Then the surrogate model $\hat{L}(\widetilde{\mathbf{X}}, \mathbf{Z})$ with respect to $\widetilde{\mathbf{X}}$ and $\mathbf{Z}$ is obtained through

$$\hat{L}(\widetilde{\mathbf{X}}, \mathbf{Z}) = \hat{L}[\mathbf{X}, H(\boldsymbol{\xi}, \mathbf{Z}), \mathbf{Z}] \tag{33}$$

Since the truncated series expansion $H(\boldsymbol{\xi}, \mathbf{Z})$ in Eq. (30) has a simple closed-form expression, if $\hat{L}(\mathbf{X}, H, \mathbf{Z})$ is accurate and efficient, so will be $\hat{L}(\widetilde{\mathbf{X}}, \mathbf{Z})$ in Eq. (33). Since the dimension of $\hat{L}(\mathbf{X}, H, \mathbf{Z})$ is $(p-1)$ lower than that of $\hat{L}(\widetilde{\mathbf{X}}, \mathbf{Z})$, it is more efficient to train $\hat{L}(\mathbf{X}, H, \mathbf{Z})$ with higher accuracy. To build $\hat{L}(\mathbf{X}, H, \mathbf{Z})$, we need the training points $\mathbf{h}^{\mathrm{trn}}$ of H. $\mathbf{h}^{\mathrm{trn}}$ can be obtained simply by substituting $(\boldsymbol{\xi}^{\mathrm{trn}}, \mathbf{z}^{\mathrm{trn}})$ into Eq. (30). Similarly, when $(\tilde{\mathbf{x}}^{(\mathrm{next})}, \mathbf{z}^{(\mathrm{next})})$ is determined by Eq. (18), the next training point $h^{(\mathrm{next})}$ of H is obtained by substituting $(\boldsymbol{\xi}^{(\mathrm{next})}, \mathbf{z}^{(\mathrm{next})})$ into Eq. (30).

Note that $\tilde{\mathbf{x}}^{(\text{next})} = \left(\boldsymbol{\xi}^{(\text{next})}, \mathbf{x}^{(\text{next})}\right)$. When more than one input random fields are involved, the procedure of building and updating the surrogate model $\hat{L}$ is similar.

However, it should be mentioned that Eq. (33) is not suitable for all problems involving input random fields. Roughly speaking, the problems involving input random fields, including random processes which are unidimensional random fields, can be grouped into two categories. To distinguish the two categories, we first need to make it clear that whenever $\mathbf{Z}$ in Eq. (28) is fixed to a specific realization $\mathbf{z}^{(i)}$, $\mathbf{H}\left(\mathbf{z}^{(i)}\right)$ and $g\left(\mathbf{X}, \mathbf{H}\left(\mathbf{z}^{(i)}\right), \mathbf{z}^{(i)}\right)$ are a random vector and a random variable, respectively. If the randomness, or uncertainty, of the output random variable $g\left(\mathbf{X}, \mathbf{H}\left(\mathbf{z}^{(i)}\right), \mathbf{z}^{(i)}\right)$ only comes from the input random variables $\mathbf{X}$ and $\mathbf{H}\left(\mathbf{z}^{(i)}\right)$, then the problem belongs to Category 1. If the randomness of $g\left(\mathbf{X}, \mathbf{H}\left(\mathbf{z}^{(i)}\right), \mathbf{z}^{(i)}\right)$ comes from not only $\mathbf{X}$ and $\mathbf{H}\left(\mathbf{z}^{(i)}\right)$ but also $\mathbf{H}\left(\mathbf{z}^{(j)}\right), j \neq i$, then the problem belongs to Category 2. Eq. (33) can only deal with problems in Category 1. When dealing with Category 2 problems, we cannot treat the entire random field as a single input variable to $\hat{L}$ anymore. Instead, we must treat each component of $\boldsymbol{\xi}$ as an input variable to $\hat{L}$, resulting in a high-dimension Gaussian process model. Currently the Gaussian process model cannot work well for high-dimensional problems. Therefore, we only consider Category 1 when input random fields are involved, and in the example section, both Example 3 and Example 4 belong to Category 1.

5. NUMERICAL EXAMPLES

In this section, we use four examples to test the proposed method. The first one is a mathematical example. With a low-dimension case in this example, we aim at clearly illustrating the detailed procedure of the proposed method. Then we test the method by setting a higher dimensionality for this example. The second one is an engineering example involving only random

variables while the third one, also an engineering example, involves both random variables and unidimensional random fields. The last engineering example involves multidimensional random fields.

The direct MCS is also used to compute the TSD robustness metric. MCS calls the original QLF model in Eq. (11) directly. The sample size of MCS is set to 10^5. The results of MCS are treated as the exact ones for the accuracy comparison. For all examples, the convergence thresholds w, γ, and n_{in}, are set to 0.005, 0.005, and 10, respectively. Both methods share the same discretization of Ω.

5.1. A math problem

The QC is given by

$$Y = \sum_{i=1}^{N} X_i^2 + 0.1(Z_1 + Z_2 + 5)^2 \sin(0.1Z_2) \prod_{i=1}^{N} X_i \tag{34}$$

where $(X_1, X_2 \ldots, X_N)$ are N independent and identically distributed normal variables with mean and standard deviation being 1 and 0.02, respectively. The domain Ω of $\mathbf{Z} = (Z_1, Z_2)$ is $[0,2] \times [0,5]$. $m(\mathbf{Z})$ is given as

$$m(\mathbf{Z}) = 0.1(Z_1 + Z_2 + 5)^2 \sin(0.1Z_2) \tag{35}$$

and $A(\mathbf{Z}) = \$1000$. Z_1 and Z_2 are discretized into 20 and 50 points, respectively; so there are $n_\Omega = 10^3$ discretization points in $\mathbf{z}^\Omega$.

For an easy demonstration, we first consider a low-dimension case and set $N = 2$. Using the Hammersley sampling, we generate 10 initial training points in the hypercube $[0,1]^{N+N_Z} = [0,1]^4$. The initial training points are given in Table 1. Then we transform the first two data columns into Z-space, using the inverse probability method mentioned in Subsection 4.2, to get the initial training points $\mathbf{z}^{\text{in}}$. Note that each training point in $\mathbf{z}^{\text{in}}$ is rounded to the nearest one in $\mathbf{z}^\Omega$. The

last two data columns are transformed into X-space to generate $\mathbf{x}^{\text{in}}$. Substituting those points in $\left(\mathbf{x}^{\text{in}}, \mathbf{z}^{\text{in}}\right)$ one by one into Eq. (9), we obtain 10 initial training points $\mathbf{l}^{\text{in}}$ of L. $\mathbf{z}^{\text{in}}$, $\mathbf{x}^{\text{in}}$ and $\mathbf{l}^{\text{in}}$ are given in Table 2. Since Eq. (9) is called 10 times, $n_{\text{call}} = 10$. Initially we set $n_{\text{MCS}} = 400$.

Table 1 Initial training points in hypercube space

Point number	Data			
1	0.0000	0.5000	0.3333	0.2000
2	0.1000	0.2500	0.6667	0.4000
3	0.2000	0.7500	0.1111	0.6000
4	0.3000	0.1250	0.4444	0.8000
5	0.4000	0.6250	0.7778	0.0400
6	0.5000	0.3750	0.2222	0.2400
7	0.6000	0.8750	0.5556	0.4400
8	0.7000	0.0625	0.8889	0.6400
9	0.8000	0.5625	0.0370	0.8400
10	0.9000	0.3125	0.3703	0.0800

With the initial training points, we build the initial $\hat{L}(\mathbf{X}, \mathbf{Z})$. Then using Eq. (14) and Eq. (21) we obtain $E_L(\Omega) = \$\,4044.5$ and $W = 1.41\%$, respectively. $W = 1.41\%$ means that if we add more training points to update $\hat{L}(\mathbf{X}, \mathbf{Z})$, we expect to improve the current $E_L(\Omega)$ by 1.41%. Since 1.41% is larger than the threshold value 0.5%, more training points are needed. The learning function in Eq. (18) locates the next training point $\left(\mathbf{z}^{(\text{next})}, \mathbf{x}^{(\text{next})}\right)$ at $(0, 0, 0.9263, 0.9630)$. Substituting $\left(\mathbf{z}^{(\text{next})}, \mathbf{x}^{(\text{next})}\right)$ into Eq. (9), we obtain $\text{l}^{(\text{next})} = 3188.0$ and then update $n_{\text{call}} = n_{\text{call}} + 1 = 11$. The new training point is added into current training set to update $\hat{L}(\mathbf{X}, \mathbf{Z})$. With the process going on, more and more training points are added. In total 7 training points are added one by one, which are given in Table 3. n_{call} becomes $10 + 7 = 17$. The update of W is shown in Fig. 3. In Iteration 8, the 7^{th} training point shown in Table 3 is added to update $\hat{L}(\mathbf{X}, \mathbf{Z})$, resulting in $W = 0.45\% < 0.5\%$. Therefore, no more new training points are needed. Note that in all the eight iterations, $n_{\text{MCS}} = 400$ remains unchanged.

Table 2 Initial training points in X-space and Z-space

Point number	$\mathbf{z}^{\text{in}}$		$\mathbf{x}^{\text{in}}$		$\mathbf{l}^{\text{in}}$
1	0.0000	2.5000	0.9914	0.9832	3664.4
2	0.2000	1.2500	1.0086	0.9949	4036.1
3	0.4000	3.7500	0.9756	1.0051	3618.5
4	0.6000	0.6250	0.9972	1.0168	4128.0
5	0.8000	3.1250	1.0153	0.9650	3657.3
6	1.0000	1.8750	0.9847	0.9858	3639.8
7	1.2000	4.3750	1.0028	0.9970	3993.8
8	1.4000	0.3125	1.0244	1.0072	4277.8
9	1.6000	2.8125	0.9643	1.0199	3722.3
10	1.8000	1.5625	0.9934	0.9719	3586.5

Table 3 Added training points

Iteration	$\mathbf{z}^{(\text{next})}$		$\mathbf{x}^{(\text{next})}$		$\mathrm{l}^{(\text{next})}$
1	0.0000	0.0000	0.9263	0.9630	3188.0
2	2.0000	0.0000	0.9769	1.0544	4269.2
3	2.0000	5.0000	0.9263	0.9630	1082.4
4	0.0000	5.0000	1.0557	0.9859	5208.2
5	0.0000	5.0000	1.0222	1.0460	6107.1
6	2.0000	0.0000	1.0161	0.9451	3707.9
7	2.0000	0.0000	0.9330	0.9927	3444.1

To check if 400 samples are sufficient to obtain accurate $E_L(\Omega)$, we calculate Γ using Eq. (22), which results in $\Gamma = 0.0064$. Since 0.0064 is larger than the threshold $\gamma = 0.005$, the sample size $n_{\text{MCS}} = 400$ is not sufficiently large and hence we need to increase it. From Eqs. (25) and (26), we know that n_{MCS} should be increased by at least 247. However, according to Eq. (27) we only increase it by 100, because we set the hyperparameter $\tilde{n}_{\text{add}} = 100$. The reason for limiting the increasing step of n_{MCS} has been given in Subsection 4.4. Then with the updated $n_{\text{MCS}} = 400 + 100 = 500$ and updated $\mathbf{x}^{\text{MCS}}$, we calculate W again to check if $\hat{L}(\mathbf{X}, \mathbf{Z})$ is still accurate.

Fig. 3 shows that $W < 0.5\%$ in Iteration 9; hence $\hat{L}(\mathbf{X}, \mathbf{Z})$ is still accurate. We calculate Γ and then W again and repeat the process until both $W < 0.5\%$ and $\Gamma < 0.005$ are satisfied.

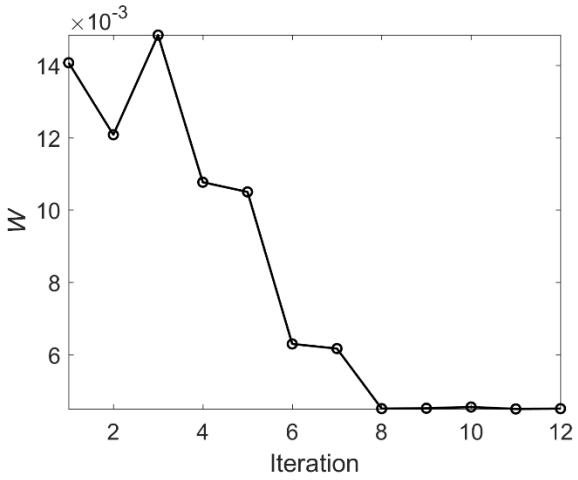

Fig. 3 Update of W

Table 4 Robustness analysis results

Methods	Proposed method	MCS
$E_L(\Omega)(\$)$	4.28×10^3	4.35×10^3
Relative error (%)	-1.5	-
n_{MCS}	677	10^5
n_{call}	17	10^8

The final results, as well as the results obtained directly by MCS, are given in Table 4. The robustness computed by the proposed method is $\$4.28 \times 10^3$, and the robustness by MCS is $\$4.35 \times 10^3$. The proposed method is very accurate, with a small relative error of -1.5%. In addition to the 10 initial training points, 7 more training points are added adaptively to update the Gaussian process model, and hence the proposed method costs 17 function calls. The proposed method adaptively increases the sample size, obtaining accurate results with only 677 samples.

To test the proposed method with higher dimensionality, we set $N = 8$ while keeping other parameters unchanged. The results obtained from the proposed method and MCS are given in Table 5. The robustness computed by the proposed method is $\$6.70 \times 10^4$, and the robustness by

MCS is $\$6.67 \times 10^4$. The relative error is 0.5%, and 70 function calls and only 400 samples are used by the proposed method. In this case, $N = 8$ and $N_Z = 2$, and hence the dimensionality of $\hat{L}(\mathbf{X}, \mathbf{Z})$ is 10. The two cases show that the proposed method works well for both low dimension and moderate dimension in this example problem.

Table 5 Robustness analysis results

Methods	Proposed method	MCS
$E_L(\Omega)(\$)$	6.70×10^4	6.67×10^4
Relative error (%)	0.5	-
n_{MCS}	400	10^5
n_{call}	70	10^8

Note that the second case needs a smaller sample size ($n_{\text{MCS}} = 400$) than that ($n_{\text{MCS}} = 677$) of the first case, although the dimensionality is higher in the second case. The reason is that the deviation coefficient $\frac{\sigma(\Omega)}{E_L(\Omega)}$ of $L_{\max}(\mathbf{X}, \Omega)$ in the first case is larger than that in the second case.

5.2. A slider mechanism

Shown in Fig. 4 is a slider mechanism [39]. The spatial variables are the offset H and the initial angle θ_0 with the following ranges: $H \in [14.85, 15.15]$ m and $\theta_0 \in [-2^\circ, 2^\circ]$. The time span is $t \in [0\ , 0.1\pi]$ s. Then the $\mathbf{Z}$ vector is (H, θ_0, t). The random variable vector is $\mathbf{X} = (L_1, L_2)$, which includes two independent random link lengths $L_1 \sim N(15, 0.015^2)$ m and $L_2 \sim N(35, 0.035^2)$ m. The QC, or the actual position of the slider, is

$$Y = L_1 \cos(\theta_0 + \omega t) + \sqrt{L_2^2 - (H + L_1 \sin(\theta_0 + \omega t))^2} \tag{36}$$

where $\omega = 1$ rad/s is the angular velocity. The target QC is

$$m(\mathbf{Z}) = 15 \cos(\omega t) + \sqrt{35^2 - (15 + 15 \sin(\omega t))^2} \tag{37}$$

and $A(\mathbf{Z}) = \$1000/\mathrm{m}^2$. The intervals of h, θ_0 and t are all evenly discretized into 20 points. Accordingly, $\Omega = [14.85, 15.15]\ \mathrm{m} \times [-2^\circ, 2^\circ] \times [0, 0.1\pi]\ \mathrm{s}$ is discretized into $n_\Omega = 20 \times 20 \times 20 = 8 \times 10^3$ points.

Setting the initial value of n_{MCS} to 1000 and $\tilde{n}_{\mathrm{add}}$ to 100, the robustness analysis results are given in Table 6. The proposed method is accurate and efficient with 31 function calls.

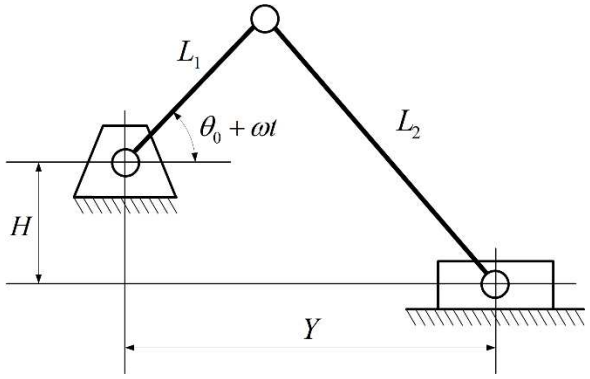

Fig.4 A slider mechanism

Table 6 Robustness analysis results

Methods	Proposed method	MCS
$E_L(\Omega)$ (\$)	88.43	88.02
Relative error (%)	0.5	-
n_{MCS}	1492	10^5
n_{call}	31	8×10^8

5.3. A cantilever beam

Shown in Fig. 5 is a cantilever beam. Its span $L = 1$ m. Due to the machining error, the diameter of its cross section is not a deterministic. Instead, it is modeled as a one-dimensional stationary Gaussian random field $D(x)$. The mean value μ_D and standard deviation σ_D of $D(x)$ are 0.1 m and 0.001 m, respectively. Its autocorrelation coefficient function $\rho_D(x_1, x_2)$ is given as

$$\rho_D(x_1, x_2) = \exp[-(x_1 - x_2)^2] \tag{38}$$

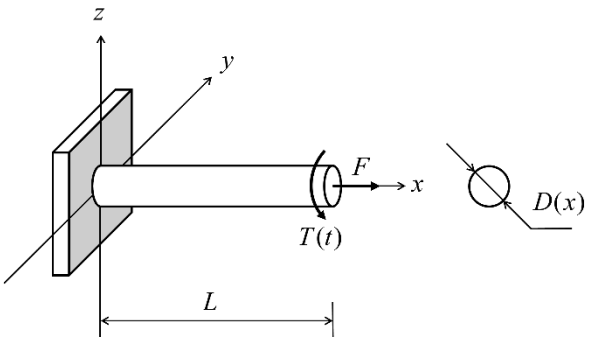

Fig. 5 A cantilever beam

The beam is subjected to a torsion $T(t)$ and a tensile force F at the right endpoint. F is a normal variable with mean μ_F and standard deviation σ_F being 1000 N and 100 N, respectively. $T(t)$ is a stationary Gaussian process with mean μ_T and standard deviation σ_T being 200 N · m and 20 N · m, respectively. Its autocorrelation coefficient function $\rho_T(t_1, t_2)$ is given by

$$\rho_T(t_1, t_2) = \exp\left[-\left(\frac{t_1 - t_2}{2}\right)^2\right] \tag{39}$$

The maximum von Misses stress of the beam is the QC and is given by

$$Y = \sqrt{\left[\frac{4F}{\pi D(x)^2}\right]^2 + 3\left[\frac{16T(t)}{\pi D(x)^3}\right]^2} \tag{40}$$

The target $m(\mathbf{Z}) = 0$ and $A(\mathbf{Z}) = \$1000/(\text{Mpa})^2$. The domain Ω of $\mathbf{Z} = (x, t)$ is $[0, 1]$ m $\times$ $[0, 5]$ yr and is evenly discretized into $n_\Omega = 20 \times 50 = 1000$ points.

With $\rho_D(x_1, x_2)$ we can get the autocorrelation coefficient matrix M_D of the one-dimensional random field $D(x)$. Since x is discretized evenly into 20 points, the dimension of M_D is 20×20. The most significant three eigenvalues of M_D are 17.0693, 2.7182 and 0.2026. We use EOLE to

generate the series expansion of $D(x)$ and only keep the first three orders. Similarly, we use EOLE to generate the series expansion of $T(t)$ and only keep the first six orders.

Setting the initial value of n_{MCS} to 1000 and $\tilde{n}_{\mathrm{add}}$ to 1000, the robustness analysis results are given in Table 7. The robustness computed by the proposed method and by MCS are $\$3.85 \times 10^3$ and 3.88×10^3, respectively. The relative error of the robustness computed by the proposed method is only −0.7%. The proposed method calls the original quality loss function 13 times, showing its high efficiency.

Table 7 Robustness analysis results

Methods	Proposed method	MCS
$E_L(\Omega)$ (\$)	3.85×10^3	3.88×10^3
Relative error (%)	-0.7	-
n_{MCS}	1000	10^5
n_{call}	13	10^8

5.4. An electron accelerator

Shown in Fig. 6 is an electron accelerator, which is used to accelerate electrons to a higher speed. Electrons are horizontally emitted from the electrode, then enter the electric field $E(w,h)$ in the accelerator, and finally fly out from the accelerator. The initial velocity of the electrons is a time-dependent stationary Gaussian random field $V_0(w,h,t)$, whose mean value μ_{V_0} and standard deviation σ_{V_0} are 500,000 m/s and 50,000 m/s, respectively. Its autocorrelation coefficient function $\rho_{V_0}(w_1,h_1,t_1;w_2,h_2,t_2)$ is given by

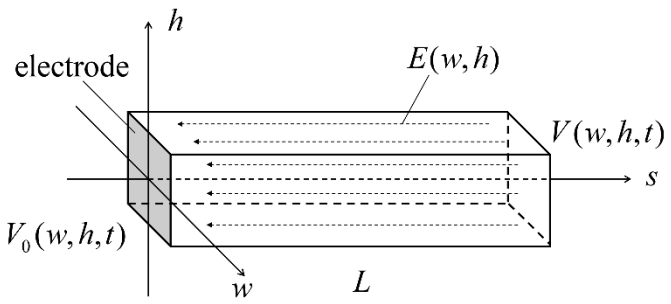

Fig. 6 An electron accelerator

$$\rho_{V_0}(w_1,h_1,t_1;w_2,h_2,t_2)=\exp\left[-\left(\frac{w_1-w_2}{0.2}\right)^2-\left(\frac{h_1-h_2}{0.2}\right)^2-\left(\frac{t_1-t_2}{4}\right)^2\right] \tag{41}$$

The length of the accelerator L is a random variable that follows a normal distribution $N(1,0.01^2)$ m. The electric field $E(w,h)$ is a two-dimensional stationary Gaussian random field, whose mean value μ_E and standard deviation σ_E are 10 N/C and 1 N/C, respectively. Its autocorrelation coefficient function $\rho_E(w_1,h_1;w_2,h_2)$ is given by

$$\rho_E(w_1,h_1;w_2,h_2)=\exp\left[-\left(\frac{w_1-w_2}{0.1}\right)^2-\left(\frac{h_1-h_2}{0.1}\right)^2\right] \tag{42}$$

If the acceleration time and the interaction among the electrons are negligible, the velocity $V(w,h,t)$ of the electrons after acceleration is given by

$$V(w,h,t)=\sqrt{\frac{2qE(w,h)L}{m}+V_0^2(w,h,t)} \tag{43}$$

where $q=1.6\times10^{-19}$ C and $m=9.109\times10^{-31}$ kg are the electric quantity and mass of an electron, respectively. The target velocity V_t is given by

$$V_t=\sqrt{\frac{2q\mu_E L}{m}+\mu_{V_0}^2} \tag{44}$$

In this example, $\mathbf{Z} = (w, h, t) \in \Omega = [-0.05, 0.05]\ \text{m} \times [-0.05, 0.05]\ \text{m} \times [0, 10]\ \text{s}$ and $m(\mathbf{Z}) = \$10^{-8}/(\text{m/s})^2$. Ω is evenly discretized into $n_\Omega = 10 \times 10 \times 20 = 2000$ points. We also use EOLE to generate the series expansions of both $V_0(w, h, t)$ and $E(w, h)$, and the first 20 and 8 orders are kept, respectively.

Setting the initial value of n_{MCS} to 2×10^4 and $\tilde{n}_{\text{add}}$ to 10^3, the robustness analysis results are given in Table 8. The robustness computed by the proposed method is $\$ 271.25$. Again, the proposed method is both accurate, with relative error being -0.1%, and efficient, with only 40 function evaluations. Although two multidimensional random fields are involved, the efficiency is still high. The high efficiency is achieved by using the method described in Subsection 4.6, and the dimension of this problem is only three, including one random variable and two random fields.

Table 8 Robustness analysis results

Methods	Proposed method	MCS
$E_L(\Omega)$ ($)	271.25	271.44
Relative error (%)	-0.1	-
n_{MCS}	22476	10^5
n_{call}	40	2×10^8

6. CONCLUSIONS

Existing robustness analysis methods only consider the static or time-dependent problems. More general are time-and space-dependent problems. In this paper, a new robustness metric is proposed for time-and space-dependent problems. The new metric has the following features:

- The robustness is measured by the expected maximum quality loss over the domain of interest, which consists of the space and period of time under consideration.
- This metric can fully take into consideration of the autocorrelation of the time- and space-dependent quality loss function at all points of the domain of interest.

- Minimizing the expected maximum quality loss will reduce both the deviation of a quality characteristic (QC) from its target and the standard deviation of the QC in the domain of interest, thereby maximizing the robustness.

An efficient robustness analysis method is developed to quantify the robustness metric based on the Gaussian process model and efficient global optimization. The method can accommodate QCs that are general functions of random variables, random processes, and random fields, temporal variables, and spatial variables.

Possible future work includes the following tasks: Further improve the efficiency of the proposed robustness analysis method, develop other robustness analysis methods, and investigate possible robustness metrics when multiple time- and space-dependent QCs are considered.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under grant CMMI 1923799 (formerly 1727329).

REFERENCES

[1] Taguchi, G., 1993, Taguchi on robust technology development: bringing quality engineering upstream, ASME Press, New York.
[2] Phadke, M. S., 1995, Quality engineering using robust design, Prentice Hall PTR.
[3] Du, X., and Chen, W., 2000, "Towards a better understanding of modeling feasibility robustness in engineering design," Journal of Mechanical Design, 122(4), pp. 385-394.
[4] Huang, B., and Du, X., 2007, "Analytical robustness assessment for robust design," Structural and Multidisciplinary Optimization, 34(2), pp. 123-137.
[5] Mulvey, J. M., Vanderbei, R. J., and Zenios, S. A., 1995, "Robust optimization of large-scale systems," Operations research, 43(2), pp. 264-281.
[6] Goh, J., and Sim, M., 2010, "Distributionally robust optimization and its tractable approximations," Operations research, 58(4-part-1), pp. 902-917.
[7] Fabozzi, F. J., Huang, D., and Zhou, G., 2010, "Robust portfolios: contributions from operations research and finance," Annals of Operations Research, 176(1), pp. 191-220.
[8] Alexandrov, N. M., and Lewis, R. M., 2002, "Analytical and computational aspects of collaborative optimization for multidisciplinary design," AIAA journal, 40(2), pp. 301-309.
[9] Sobieszczanski-Sobieski, J., and Haftka, R. T., 1997, "Multidisciplinary aerospace design optimization: survey of recent developments," Structural optimization, 14(1), pp. 1-23.

[10] Doltsinis, I., Kang, Z., and engineering, 2004, "Robust design of structures using optimization methods," Computer Methods in Applied Mechanics and Engineering, 193(23-26), pp. 2221-2237.
[11] Lagaros, N. D., Plevris, V., and Papadrakakis, M., 2007, "Reliability based robust design optimization of steel structures," International Journal for Simulation Multidisciplinary Design Optimization, 1(1), pp. 19-29.
[12] Cheng, J., Liu, Z., Wu, Z., Li, X., and Tan, J., 2015, "Robust optimization of structural dynamic characteristics based on adaptive Kriging model and CNSGA," Structural and Multidisciplinary Optimization, 51(2), pp. 423-437.
[13] Roy, B. K., and Chakraborty, S., 2015, "Robust optimum design of base isolation system in seismic vibration control of structures under random system parameters," Structural Safety, 55, pp. 49-59.
[14] Fang, J., Gao, Y., Sun, G., and Li, Q., 2013, "Multiobjective reliability-based optimization for design of a vehicledoor," Finite Elements in Analysis and Design, 67, pp. 13-21.
[15] Hwang, K., Lee, K., and Park, G., 2001, "Robust optimization of an automobile rearview mirror for vibration reduction," Structural and Multidisciplinary Optimization, 21(4), pp. 300-308.
[16] Sun, G., Li, G., Zhou, S., Li, H., Hou, S., and Li, Q., 2011, "Crashworthiness design of vehicle by using multiobjective robust optimization," Structural and Multidisciplinary Optimization, 44(1), pp. 99-110.
[17] Lee, T. H., and Jung, J., 2006, "Metamodel-based shape optimization of connecting rod considering fatigue life," Key Engineering Materials, 306-308, pp. 211-216.
[18] Li, F., Meng, G., Sha, L., and Zhou, L., 2011, "Robust optimization design for fatigue life," Finite Elements in Analysis and Design, 47(10), pp. 1186-1190.
[19] Carpinelli, G., Caramia, P., and Varilone, P., 2015, "Multi-linear Monte Carlo simulation method for probabilistic load flow of distribution systems with wind and photovoltaic generation systems," Renewable Energy, 76, pp. 283-295.
[20] Li, F., Luo, Z., Sun, G., and Zhang, N., 2013, "An uncertain multidisciplinary design optimization method using interval convex models," Engineering Optimization, 45(6), pp. 697-718.
[21] Du, X., Guo, J., and Beeram, H., 2008, "Sequential optimization and reliability assessment for multidisciplinary systems design," Structural and Multidisciplinary Optimization, 35(2), pp. 117-130.
[22] Kim, N.-K., Kim, D.-H., Kim, D.-W., Kim, H.-G., Lowther, D. A., and Sykulski, J. K., 2010, "Robust optimization utilizing the second-order design sensitivity information," IEEE transactions on Magnetics, 46(8), pp. 3117-3120.
[23] Papadimitriou, D., and Giannakoglou, K., 2013, "Third‐order sensitivity analysis for robust aerodynamic design using continuous adjoint," International Journal for Numerical Methods in Fluids, 71(5), pp. 652-670.
[24] Li, M., Hamel, J., and Azarm, S., 2010, "Optimal uncertainty reduction for multi-disciplinary multi-output systems using sensitivity analysis," Structural and Multidisciplinary Optimization, 40(1-6), p. 77.
[25] Siddiqui, S., Azarm, S., and Gabriel, S., 2011, "A modified Benders decomposition method for efficient robust optimization under interval uncertainty," Structural and Multidisciplinary Optimization, 44(2), pp. 259-275.
[26] Siddiqui, S., Gabriel, S. A., and Azarm, S., 2015, "Solving mixed-integer robust optimization problems with interval uncertainty using Benders decomposition," Journal of the Operational Research Society, 66(4), pp. 664-673.

[27] Chatterjee, T., Chowdhury, R., and Ramu, P., 2019, "Decoupling uncertainty quantification from robust design optimization," Structural and Multidisciplinary Optimization, 59(6), pp. 1969-1990.
[28] Chatterjee, T., Chakraborty, S., and Chowdhury, R., 2019, "A critical review of surrogate assisted robust design optimization," Archives of Computational Methods in Engineering, 26(1), pp. 245-274.
[29] Steuben, J., and Turner, C. J., "Robust Optimization Exploration Using NURBs-Based Metamodeling Techniques," Proc. ASME 2010 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers Digital Collection, pp. 239-250.
[30] Steuben, J. C., and Turner, C., 2012, "Robust optimization of mixed-integer problems using NURBs-based metamodels," Journal of Computing Information Science in Engineering, 12(4), p. 041010.
[31] Steuben, J. C., Turner, C. J., and Crawford, R. H., 2013, "Robust engineering design optimization with non-uniform rational B-splines-based metamodels," Engineering Optimization, 45(7), pp. 767-786.
[32] Zhou, H., Zhou, Q., Liu, C., and Zhou, T., 2018, "A kriging metamodel-assisted robust optimization method based on a reverse model," Engineering Optimization, 50(2), pp. 253-272.
[33] Du, X., 2012, "Toward Time-Dependent Robustness Metrics," Journal of Mechanical Design, 134(1), p. 011004.
[34] Du, X., Sudjianto, A., and Chen, W., 2004, "An integrated framework for optimization under uncertainty using inverse reliability strategy," Journal of Mechanical Design, 126(4), pp. 562-570.
[35] Gu, X., Renaud, J. E., Batill, S. M., Brach, R. M., and Budhiraja, A. S., 2000, "Worst case propagated uncertainty of multidisciplinary systems in robust design optimization," Structural and Multidisciplinary Optimization, 20(3), pp. 190-213.
[36] Tsui, K.-L., 1999, "Modeling and analysis of dynamic robust design experiments," IIE Transactions, 31(12), pp. 1113-1122.
[37] Wu, F.-C., 2009, "Robust design of nonlinear multiple dynamic quality characteristics," Computers & Industrial Engineering, 56(4), pp. 1328-1332.
[38] Goethals, P. L., and Cho, B. R., 2011, "The development of a robust design methodology for time‐oriented dynamic quality characteristics with a target profile," Quality and Reliability Engineering International, 27(4), pp. 403-414.
[39] Wei, X., and Du, X., 2019, "Uncertainty Analysis for Time-and Space-Dependent Responses With Random Variables," Journal of Mechanical Design, 141(2), p. 021402.
[40] Törn, A., and Žilinskas, A., 1989, Global optimization, Springer.
[41] Strongin, R. G., and Sergeyev, Y. D., 1992, "Global multidimensional optimization on parallel computer," Parallel Computing, 18(11), pp. 1259-1273.
[42] Strongin, R. G., and Sergeyev, Y. D., 2013, Global optimization with non-convex constraints: Sequential and parallel algorithms, Springer Science & Business Media.
[43] Sasena, M. J., Papalambros, P., and Goovaerts, P., 2002, "Exploration of metamodeling sampling criteria for constrained global optimization," Engineering Optimization, 34(3), pp. 263-278.
[44] Williams, C. K., and Rasmussen, C. E., 2006, Gaussian processes for machine learning, MIT Press Cambridge, MA.
[45] Lophaven, S. N., Nielsen, H. B., and Søndergaard, J., 2002, DACE: a Matlab kriging toolbox, Citeseer.

[46] Zhu, Z., and Du, X., 2016, "Reliability analysis with Monte Carlo simulation and dependent Kriging predictions," Journal of Mechanical Design, 138(12), p. 121403.
[47] Hu, Z., and Mahadevan, S., 2016, "A single-loop kriging surrogate modeling for time-dependent reliability analysis," Journal of Mechanical Design, 138(6), p. 061406.
[48] Sudret, B., and Der Kiureghian, A., 2000, Stochastic finite element methods and reliability: a state-of-the-art report, Department of Civil and Environmental Engineering, University of California Berkeley.
[49] Gabrel, V., Murat, C., and Thiele, A., 2014, "Recent advances in robust optimization: An overview," European Journal of Operational Research, 235(3), pp. 471-483.
[50] Bovier, A., 2005, "Extreme values of random processes," Lecture Notes Technische Universität Berlin.
[51] Kotz, S., and Nadarajah, S., 2000, Extreme value distributions: theory and applications, World Scientific, London.
[52] Mooney, C. Z., 1997, Monte carlo simulation, Sage Publications.
[53] Zienkiewicz, O. C., Taylor, R. L., Nithiarasu, P., and Zhu, J., 1977, The finite element method, McGraw-hill London.
[54] Hu, Z., and Du, X., 2015, "Mixed efficient global optimization for time-dependent reliability analysis," Journal of Mechanical Design, 137(5), p. 051401.
[55] Jones, D. R., Schonlau, M., and Welch, W. J., 1998, "Efficient global optimization of expensive black-box functions," Journal of Global Optimization, 13(4), pp. 455-492.
[56] Chen, W., Tsui, K.-L., Allen, J. K., and Mistree, F., 1995, "Integration of the response surface methodology with the compromise decision support problem in developing a general robust design procedure," ASME Design Engineering Technical Conference, pp. 485-492.
[57] Hosder, S., Walters, R., and Balch, M., 2007, "Efficient sampling for non-intrusive polynomial chaos applications with multiple uncertain input variables," Proc. 48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, p. 1939.
[58] Wang, Z., and Wang, P., 2012, "A Nested Extreme Response Surface Approach for Time-Dependent Reliability-Based Design Optimization," Journal of Mechanical Design, 134(12), pp. 121007-121014.

List of Table Captions

List of Figure Captions

