Journal of Computational Physics 313 (2016) 247–259

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Gaussian process surrogates for failure detection: A Bayesian experimental design approach

Hongqiao Wang [a], Guang Lin [b], Jinglai Li [c,*]

[a] Institute of Natural Sciences, Department of Mathematics, Shanghai Jiao Tong University, Shanghai 200240, China
[b] Department of Mathematics, School of Mechanical Engineering, Purdue University, 150 N. University Street, West Lafayette, IN 47907-2067, USA
[c] Institute of Natural Sciences, Department of Mathematics, MOE Key Laboratory of Scientific and Engineering Computing, Shanghai Jiao Tong University, Shanghai 200240, China

ARTICLE INFO

Article history:
Received 10 September 2015
Received in revised form 1 February 2016
Accepted 22 February 2016
Available online 26 February 2016

Keywords:
Bayesian inference
Experimental design
Failure detection
Gaussian processes
Monte Carlo
Response surfaces
Uncertainty quantification

ABSTRACT

An important task of uncertainty quantification is to identify the probability of undesired events, in particular, system failures, caused by various sources of uncertainties. In this work we consider the construction of Gaussian process surrogates for failure detection and failure probability estimation. In particular, we consider the situation that the underlying computer models are extremely expensive, and in this setting, determining the sampling points in the state space is of essential importance. We formulate the problem as an optimal experimental design for Bayesian inferences of the limit state (i.e., the failure boundary) and propose an efficient numerical scheme to solve the resulting optimization problem. In particular, the proposed limit-state inference method is capable of determining multiple sampling points at a time, and thus it is well suited for problems where multiple computer simulations can be performed in parallel. The accuracy and performance of the proposed method is demonstrated by both academic and practical examples.

1. Introduction

Real-life engineering systems are unavoidably subject to various uncertainties such as material properties, geometric parameters, boundary conditions and applied loadings. These uncertainties may cause undesired events, in particular, system failures or malfunctions, to occur. Accurate identification of failure modes and evaluation of failure probability of a given system is an essential task in many fields of engineering such as risk management, structural design, reliability-based optimization, etc.

In the many fields of applications such as structural engineering, system failures are often described with the limit state concepts. Simply put, the state space of the random parameters in a system is divided into two domains (while each domain may contain multiple sub-regions): the safe domain, the states in which may not cause system failures, and the failure domain the states in which may cause system failures. And the limit states represent the boundary between the two domains. The mathematical formulation of the limit states will be provided in Section 2.

Conventionally the failure probability is often computed by constructing linear or quadratic expansions of the system model around the so-called most probable point, known as the first/second-order reliability method (FORM/SORM); see e.g.,

[*] Corresponding author.
E-mail addresses: wanghongqiao@sjtu.edu.cn (H. Wang), guanglin@purdue.edu (G. Lin), jinglaili@sjtu.edu.cn (J. Li).

http://dx.doi.org/10.1016/j.jcp.2016.02.053
0021-9991/© 2016 Elsevier Inc. All rights reserved.

[33] and the references therein. It is well known that FORM/SORM may fail for systems with nonlinearity or multiple failure modes. The Monte Carlo (MC) simulation, which estimates the failure probability by repeatedly simulating the underlying system, is another popular method for solving such problems. The MC method makes no approximation to the underlying computer models and thus can be applied to any systems. On the other hand, the MC method is notorious for its slow convergence, and thus can become prohibitively expensive when the underlying computer model is computationally intensive and/or the system failures are rare and each sample requires to a full-scale numerical simulation of the system. To reduce the computational effort, one can construct a computationally inexpensive approximation to the true model, and then replace the true model with the approximate one in the MC simulations. Such approximate models are also known as response surfaces, surrogates, metamodels, and emulators, etc. These methods are referred to as the response surface (RS) methods [14,30,15] in this work. The emulator based response surface can often provide a reliable estimate of the failure probability, at a much lower computational cost than direct MC simulations.

In this work we focus on a specific kind of RS, the Gaussian processes (GP) surrogates [37], also known as kriging in many fields of applications. The GP surrogates have been applied to uncertainty quantification problems first in [25,27] and more recently in [7,8]. A number of GP-based methods have been also been successfully implemented for failure probability estimation [21,3]. In this work we consider the situation where the underlying computer models are extremely expensive and one can only afford a very limited number of simulations. In this setting, choosing the sampling points (i.e. the parameter values with which the simulation is performed) in the state space is of essential importance. Determining the sampling points for GP can be cast as to optimally design computer experiments and considerable efforts [32] have been devoted to it. Those methods aim to construct a surrogate that can accurately approximate the target function in the whole parameter space. As will be explained later, in the failure probability estimation or failure detection problems, only the sign of the target function is used. Thus by requiring surrogates to be globally accurate, the methods may allocate considerable computational efforts to the regions not of interest, and use much more model simulations than necessary.

Several methods have developed to determine sampling points for the failure probability estimation purposes. Most of these methods consist of sequentially finding the “best point” as a result of a heuristic balance between predicted closeness to the limit state, and high prediction uncertainty, e.g. [12,6]. Such methods are shown to be effective in many applications, while a major limitation is their point-wise sequential nature, which makes it unsuitable for problems in which multiple computer simulations can be performed parallelly. An exception is the stepwise uncertainty reduction (SUR) method developed in [4,9], in which the authors proposed an optimal experimental design framework which determines multiple sampling points by minimizing the average variance of the failure probability. It should be noted that the design criteria in the SUR method is particularly developed for the goal of estimating the failure probability only. In practice, one is often not only interested in estimating the failure probability, but also identifying the events that can cause failures; the latter demands a design criteria for the goal of detecting the limit state, i.e., the boundaries of the failure domain. In this work, we recast the surrogate construction as a Bayesian inference to identify the limit state, and based on that, we formulate an *information-theoretic* optimal experimental design, which uses the relative entropy from the prior to the posterior as the design criteria, to determine the sampling points. We also present an efficient numerical scheme for solving the resulting optimal design problem, modified from the simulation-based method developed in [17]. We compare the performance of the proposed limit-state inference (LSI) method with that of the SUR by numerical examples.

We note that another line of research in failure probability estimation is to develop more efficient sampling schemes, such as the subset simulations [2], importance sampling [13], the cross-entropy method [31,36], etc. The LSI method can be easily integrated into the aforementioned sampling schemes, resulting in more efficient estimation schemes. Examples of combining surrogates and efficient sampling schemes include [20,21,11], just to name a few.

The rest of this paper is organized as following. We first review the preliminaries of our work in Section 2, including the mathematical formulation of failure probability computation and the GP surrogates. Our Bayesian experimental design framework and its numerical implementations are presented in Section 3. Numerical examples are presented in Section 4 to demonstrate the effectiveness of the proposed method, and finally Section 5 offers some closing remarks.

2. Problem formulation

2.1. Failure detection and failure probability estimation

Here we describe the failure probability estimation problem in a general setting. We consider a probabilistic model where $\mathbf{x}$ is a d-dimensional random variable that represents the uncertainty in model and let $\Omega \subseteq R^d$ be the state space of $\mathbf{x}$. The system failure is often defined using a real-valued function $g(\cdot): \Omega \to R$, which is known as the *limit state function* or the *performance function*. Specifically, the event of failure is defined as $g(\mathbf{x}) < 0$ and as a result the failure probability is

$$P = \mathbb{P}(g(\mathbf{x}) < 0) = \int\limits_{\{\mathbf{x}\in\Omega | g(\mathbf{x})<0\}} p(\mathbf{x}) d\mathbf{x} = \int\limits_{\mathbf{x}\in\Omega} I_g(\mathbf{x}) p_X(\mathbf{x}) d\mathbf{x},$$

where $I_g(\mathbf{x})$ is an indicator function:

$$I_g(\mathbf{x}) = \begin{cases} 1 & \text{if } g(\mathbf{x}) < 0, \\ 0 & \text{if } g(\mathbf{x}) \geq 0; \end{cases}$$

and $p_X(\mathbf{x})$ is the probability density function (PDF) of $\mathbf{x}$. In what follows we shall omit the integration domain when it is simply Ω. This is a general definition for failure probability, which is used widely in many disciplines involving reliability analysis and risk management. P can be computed with the standard Monte Carlo estimation:

$$\hat{P} = \frac{1}{n}\sum_{i=1}^{n} I_g(\mathbf{x}_i), \tag{1}$$

where samples $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are drawn from distribution $p_X(\mathbf{x})$. In principle $p_X(\mathbf{x})$ can be any probability distribution and the method presented in the work does not require any assumptions on it.

In practice, many engineering systems require high reliability, namely the failure probability $P \ll 1$. In this case, MC requires a rather large number of samples to produce a reliable estimate of the failure probability. For example, for $P \approx 10^{-3}$, MC simulation requires 10^5 samples to obtain an estimate with 10% coefficient of variation. On the other hand, in almost all practical cases, the limit state function $g(\mathbf{x})$ does not admit analytical expression and has to be evaluated through expensive computer simulations, which renders the crucial MC estimation of the failure probability prohibitive. To reduce the number of full-scale computer simulations, one can construct a computationally inexpensive surrogate $G(\mathbf{x})$ to replace the real function $g(\mathbf{x})$ in the MC estimation. In this work we choose the Gaussian Process surrogates and we provide a description of GP in the next section.

2.2. The Gaussian process regression

The GP method constructs the approximation of $g(\mathbf{x})$ in a nonparametric Bayesian regression framework [37,25,27,29]. Specifically the target function $g(\mathbf{x})$ is cast as a Gaussian process whose mean is $\mu(\mathbf{x})$ and covariance is specified by a kernel $k(\mathbf{x}, \mathbf{x}')$, namely,

$$\mathrm{COV}[g(\mathbf{x}), g(\mathbf{x}')] = k(\mathbf{x}, \mathbf{x}').$$

The kernel $k(\mathbf{x}, \mathbf{x}')$ is positive semidefinite and bounded. Suppose that n computer simulations of the function $g(\mathbf{x})$ are performed at parameter values $\mathbf{X}^* := [\mathbf{x}_1^*, \ldots \mathbf{x}_n^*]$, yielding function evaluations $\mathbf{y}^* := [y_1^*, \ldots y_n^*]$, where

$$y_i^* = g(\mathbf{x}_i^*) \quad \text{for} \quad i = 1, \ldots, n.$$

Suppose we want to predict the function values at points $\mathbf{D} := [\mathbf{x}_1, \ldots \mathbf{x}_m]$, i.e., $\mathbf{y} = [y_1, \ldots y_m]$ where $y_i = g(\mathbf{x}_i)$. The sets $\mathbf{X}^*$ and $\mathbf{D}$ are often known as the training and the test points respectively. The joint prior distribution of $(\mathbf{y}^*, \mathbf{y})$ is,

$$\begin{bmatrix} \mathbf{y}^* \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\left(\begin{matrix} \mu(\mathbf{X}^*) \\ \mu(\mathbf{D}) \end{matrix}, \begin{bmatrix} K(\mathbf{X}^*, \mathbf{X}^*) & K(\mathbf{X}^*, \mathbf{D}) \\ K(\mathbf{D}, \mathbf{X}^*) & K(\mathbf{D}, \mathbf{D}) \end{bmatrix} \right), \tag{2}$$

where we use the notation $K(\mathbf{A}, \mathbf{B})$ to denote the matrix of the covariance evaluated at all pairs of points in set $\mathbf{A}$ and in set $\mathbf{B}$. The posterior distribution of $\mathbf{y}$ is also Gaussian:

$$\mathbf{y} \mid \mathbf{D}, \mathbf{X}^*, \mathbf{y}^* \sim \mathcal{N}(\mathbf{u}, \Sigma), \tag{3a}$$

where the posterior mean is

$$\mathbf{u} = \mu(\mathbf{D}) + K(\mathbf{D}, \mathbf{X}^*)K(\mathbf{X}^*, \mathbf{X}^*)^{-1}(\mathbf{y} - \mu(\mathbf{D})), \tag{3b}$$

and the posterior covariance matrix is

$$\Sigma = K(\mathbf{D}, \mathbf{D}) - K(\mathbf{D}, \mathbf{X}^*)K(\mathbf{X}^*, \mathbf{X}^*)^{-1}K(\mathbf{X}^*, \mathbf{D}). \tag{3c}$$

Here we only provide a brief introduction to the GP method tailored for our own purposes, and readers who are interested in further details may consult the aforementioned references.

3. The experimental design framework

A key question in constructing a GP surrogate is to determine the locations $\mathbf{D}$ where the true limit state function $g(\mathbf{x})$ is evaluated, which is often known as a design of computer experiments. In this section, we show that determining the sampling locations can be translated into a Bayesian experimental design problem whose goal is to find the locations of limit state with a given prior distribution.

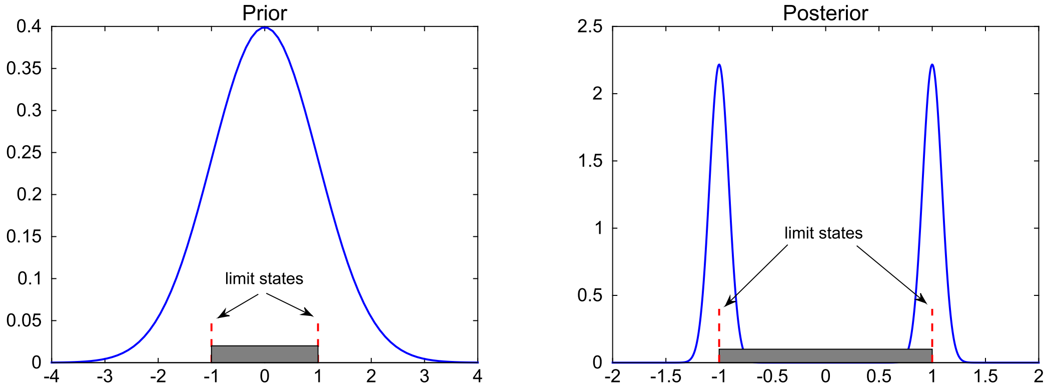

Fig. 1. A concept-illustrating example: $\Omega = R$, $x \sim \mathcal{N}(0, 1)$, failure region $(-1, 1)$, limit states $\{-1, 1\}$. Left: the prior distribution of z which is take to be the same as the distribution of x. Right: the posterior distribution of z.

3.1. Bayesian inference experiments

We formulate a Bayesian inference problem based on the following argument. In failure probability estimation, the limit state function g is only used in the indicator function $I_g(\mathbf{x})$ and so one is really interested in the sign of $g(\mathbf{x})$ rather than the precise value of it. To this end, the essential task in constructing surrogates for the failure probability estimation is to learn about the boundary of the failure domain. Here we emphasize that the indicator function $I_g(\mathbf{x})$ is a step function, but the limit state function $g(\mathbf{x})$ is continuous and follows the GP assumption described in Section 2.2. Let $\mathbf{Z} = \{\mathbf{z} := \mathbf{x} \mid \mathbf{x} \in \Omega, g(\mathbf{x}) = 0\}$ represents the boundary of the failure domain, i.e., the collection of solutions of $g(\mathbf{x}) = 0$. We want to identify $\mathbf{z}$ with the Bayesian inference, namely, to compute the distribution of $\mathbf{z}$ based on certain observed data. In our problem, the data is obtained by making observations, i.e., evaluate $g(\mathbf{x})$ at locations $\mathbf{D} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$, resulting in data $\mathbf{y} = (y_1, \ldots, y_n)$ with each $y_i = g(\mathbf{x}_i)$. Formally the distribution of $\mathbf{z}$ conditional on the data $\mathbf{y}$ is given by the Bayes' theorem:

$$p(\mathbf{z}|\mathbf{y}, \mathbf{D}) = \frac{p(\mathbf{y}|\mathbf{z}, \mathbf{D})p(\mathbf{z})}{p(\mathbf{y}|\mathbf{D})}, \tag{4}$$

where $p(\mathbf{y}|\mathbf{D})$ is the evidence. Loosely speaking, the posterior distribution $p(\mathbf{z}|\mathbf{y}, \mathbf{D})$ quantifies the possibility that a given point $\mathbf{z} \in \Omega$ is a limit state.

We now associate the parameter of interest $\mathbf{z}$ to the data $\mathbf{y}$, based on the assumption that the function $g(\mathbf{x})$ is a Gaussian process described in Section 2.2. Namely we let the training set be $\mathbf{X}^* = [\mathbf{z}]$ and the associated data set be $\mathbf{y}^* = [0]$, and the test set and the associated data set be $\mathbf{D}$ and $\mathbf{y}$ respectively. Now we assume the limit state function $g(\mathbf{x})$ is a GP whose prior distribution is as prescribed in Section 2.2. It follows immediately that the joint distribution of $\mathbf{y}$ and $\mathbf{y}^*$ is simply Eq. (2), and the conditional distribution $p(\mathbf{y}|\mathbf{z}, \mathbf{D})$ is given by Eqs. (3), which provides the likelihood function in Eq. (4). Finally to use Eq. (4), we also need to choose a prior distribution for $\mathbf{z}$. Since $\mathbf{Z}$ is a subset of Ω, without additional information, it is reasonable to assume that the prior of $\mathbf{z}$ is the same as the distribution of $\mathbf{x}$, i.e., $p(\mathbf{z}) = p_X(\mathbf{z})$. A concept-illustrating example is given in Fig. 1.

3.2. Optimal design criteria

As is mentioned earlier, determining the locations $\mathbf{D}$ can be formulated as a design of the Bayesian inference experiments. Following the decision-theoretic optimal experimental design framework, an objective for experimental design can be generally formed:

$$U(\mathbf{D}) = \int\int u(\mathbf{D}, \mathbf{y}, \mathbf{z})p(\mathbf{z}, \mathbf{y}|\mathbf{D})d\mathbf{z}d\mathbf{y} = \int\int u(\mathbf{D}, \mathbf{y}, \mathbf{z})p(\mathbf{z}|\mathbf{y}, \mathbf{D})p(\mathbf{y}|\mathbf{D})d\mathbf{z}d\mathbf{y}, \tag{5}$$

where $u(\mathbf{D}, \mathbf{y}, \mathbf{z})$ is a utility function, $U(\mathbf{D})$ is the expected utility. The utility function u should reflect the usefulness of an experiment at conditions $\mathbf{D}$, i.e., we will get a particular value outcome $\mathbf{y}$ at condition $\mathbf{D}$ by inputting a particular value of the parameters $\mathbf{z}$. Since we do not know the exact value of $\mathbf{z}$ and $\mathbf{y}$ in advance, we take the expectation of u over the joint distribution of $\mathbf{z}$ and $\mathbf{y}$, resulting in the expected utility $U(\mathbf{D})$. The optimal choice of $\mathbf{D}$ then can be obtained by maximizing the expected utility of the design space Ξ:

$$\mathbf{D}^* = \arg\max_{\mathbf{D} \in \Xi} U(\mathbf{D}). \tag{6}$$

A popular choice for the utility function is the relative entropy, also known as the Kullback–Leibler divergence (KLD), between the prior and the posterior distributions. For two distributions $p_A(\mathbf{z})$ and $p_B(\mathbf{z})$, the KLD from p_A to p_B is defined as

$$D_{KL}(p_A \parallel p_B) = \int p_A(\mathbf{z}) \ln \left[\frac{p_A(\mathbf{z})}{p_B(\mathbf{z})}\right] d\mathbf{z} = \mathbb{E}_A \left[\ln \frac{p_A(\mathbf{z})}{p_B(\mathbf{z})}\right] \tag{7}$$

where we define $0 \ln 0 \equiv 0$. This quantity is non-negative, non-symmetric, and reflects the difference in information carried by the two distributions. When KLD is used in our inference problem, the utility function becomes

$$u(\mathbf{D}, \mathbf{y}, \mathbf{z}) \equiv \mathcal{D}_{KL}(p(\cdot|\mathbf{y}, \mathbf{D}) \parallel p(\cdot)) = \int p(\tilde{\mathbf{z}}|\mathbf{y}, \mathbf{D}) \ln \left[\frac{p(\tilde{\mathbf{z}}|\mathbf{y}, \mathbf{D})}{p(\tilde{\mathbf{z}})}\right] d\tilde{\mathbf{z}}. \tag{8}$$

The utility function $u(\mathbf{D}, \mathbf{y}, \mathbf{z})$ in Eq. (8) can be understood as the information gain by performing experiments under conditions $\mathbf{D}$, and larger value of u implies that the experiment is more informative for parameter inference. Note that the utility function in Eq. (8) is independent of $\mathbf{z}$ and as such the expected utility $U(\mathbf{D})$ is reduced to

$$U(\mathbf{D}) = \int \mathcal{D}_{KL}(p(\cdot|\mathbf{y}, \mathbf{D}) \parallel p(\cdot)) p(\mathbf{y}|\mathbf{D})\, d\mathbf{y} = \int \int p(\mathbf{z}|\mathbf{y}, \mathbf{D}) \ln \left[\frac{p(\mathbf{z}|\mathbf{y}, \mathbf{D})}{p(\mathbf{z})}\right] d\mathbf{z}\, p(\mathbf{y}|\mathbf{D})\, d\mathbf{y}, \tag{9}$$

where $\tilde{\mathbf{z}}$ in Eq. (8) is replaced by $\mathbf{z}$ for simplicity. Next we discuss how to numerically solve the optimization problem Eq. (6), with $U(\mathbf{D})$ given by Eq. (9).

3.3. Numerical implementation

Following the recommendation of [17], we use the simultaneous perturbation stochastic approximation (SPSA) method, to solve the optimization problem Eq. (6). SPSA is a derivative-free stochastic optimization method that was first proposed by Spall [34,35], and we provide the detailed algorithm of SPSA in Appendix A. Note here that, since it is a derivative-free method, the algorithm only uses the function value of $U(\mathbf{D})$.

Next we discuss the evaluation of $U(\mathbf{D})$. To start, we re-write Eq. (9) as

$$\begin{aligned} U(\mathbf{D}) &= \int \int p(\mathbf{z}|\mathbf{y}, \mathbf{D}) \ln \left[\frac{p(\mathbf{z}|\mathbf{y}, \mathbf{D})}{p(\mathbf{z})}\right] d\mathbf{z}\, p(\mathbf{y}|\mathbf{D})\, d\mathbf{y} \\ &= -\frac{1}{2} \int \ln |\mathrm{C}|\, p(\mathbf{z})\, d\mathbf{z} - \int \int \ln[p(\mathbf{y}|\mathbf{D})] p(\mathbf{y}|\mathbf{D})\, d\mathbf{y} + Z \\ &= -\frac{1}{2} \mathbb{E}[\ln |\mathrm{C}|] + \mathbb{H}[p(\mathbf{y}|\mathbf{D})] + Z \end{aligned} \tag{10}$$

where $\mathbb{H}(\cdot)$ is the notation for entropy and Z is a constant independent of $\mathbf{D}$. The detailed derivation of Eq. (10) is given in Appendix B. Note that Eq. (10) typically has no closed-form expression and has to be evaluated with MC simulations. Draw m pairs of samples $\{(\mathbf{y}_1, \mathbf{z}_1), \ldots, (\mathbf{y}_m, \mathbf{z}_m)\}$ from $p(\mathbf{y}, \mathbf{z}|\mathbf{D})$, and the MC estimator of $\mathbb{E}[\ln |\mathrm{C}|]$ is

$$\hat{U}_1 = \frac{1}{m} \sum_{i=1}^{m} |\mathrm{C}(\mathbf{z}_i, \mathbf{D})|. \tag{11}$$

Recall that in [17], the entropy term $\mathbb{H}[p(\mathbf{y}|\mathbf{D})]$ is computed using a nested MC. For efficiency's sake, we use the resubstitution method developed in [1] to estimate the entropy. The basic idea of the method is rather straightforward: given a set of samples $\{\mathbf{y}_1, \ldots, \mathbf{y}_m\}$ of $p(\mathbf{y}|\mathbf{D})$, one first computes an estimator of the density $p(\mathbf{y}|\mathbf{D})$, say $\hat{p}(\mathbf{y})$, with certain density estimation approach, and then estimates the entropy with,

$$\hat{U}_2 = \frac{1}{m} \sum_{i=1}^{m} \hat{p}(\mathbf{y}_i). \tag{12}$$

Theoretical properties of the resubstitution method are analyzed in [18,16] and other entropy estimation methods can be found in [5]. In the original work [1], the distribution $\hat{p}$ is obtained with kernel density estimation, which can become very costly when the dimensionality of $\mathbf{y}$ gets high, and to address the issue, we use Gaussian mixture based density estimation method [24]. Without loss of generality we can simply set the constant $Z = 0$ and thus an estimate of U is simply

$$\hat{U} = \hat{U}_1 + \hat{U}_2, \tag{13}$$

which will be used to provide function values in the SPSA algorithm. Finally we reinstate that the numerical implementation in this work differs from a more general implementation outlined in [17] in the following two aspects: first, thanks to the Gaussianity of $p(\mathbf{y}|\mathbf{D})$, we can analytically integrate the variable $\mathbf{y}$ in the first integral on the right hand side of Eq. (10); second, we use Gaussian mixture density estimation method to approximate $p(\mathbf{y}|\mathbf{D})$ to avoid the nested MC integration. We note that there are other methods such as [22] that can be used to approximate the nested integral.

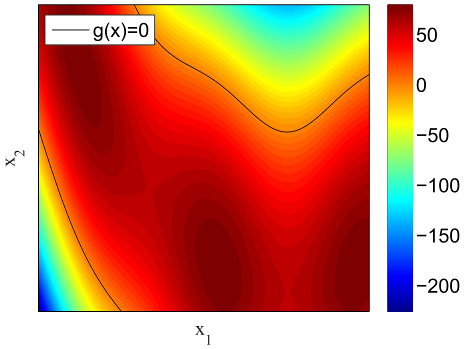

Fig. 2. The rescaled Branin function. The solid line is the limit state $g(\mathbf{x}) = 0$.

3.4. Sequential optimal design

Until this point we have presented our method under the assumption that the sampling points are determined all in advance of performing computer simulations, which is often referred to as an open-loop design. In many applications, a more practical strategy is to choose the sampling points in a sequential fashion: determine a set of sampling points, perform simulations, determine another set of points based on the previous results, and so forth. A sequential (close-loop) design can be readily derived from the open-loop version. For computational efficiency, we adopt the "greedy" approach for the sequential design, while noting that this approach can only obtain sub-optimal solutions in general. Simply speaking, the greedy sequential design iterates as follows until a prescribed stopping criteria is met:

(1) determine n sampling points with an open-loop design;
(2) perform simulations at the chosen sampling points;
(3) use the simulation results to update $p(\mathbf{z})$;
(4) return to step (1).

A key ingredient in the greedy procedure is that, in each loop, the prior $p(\mathbf{z})$ is updated using the data, i.e., the simulation results, obtained in the previous stage. To be specific, the prior distribution at step $k + 1$ is taken to be the posterior distribution of $\mathbf{z}$ at step k:

$$p_{k+1}(\mathbf{z}) = p_k(\mathbf{z}|\mathbf{y}, \mathbf{D}),$$

while the update of the likelihood remains unchanged in each step. Note that we do not have analytical expression for $p_{k+1}(\mathbf{z})$ or equivalently $p_k(\mathbf{z}|\mathbf{y}, \mathbf{D})$; however, our implementation only requires samples drawn from $p_{k+1}(\mathbf{z})$ which are available from the previous iteration. Finally we also need to specify a stopping criterion for the algorithm and following the previous works we stop the iterations when the difference the estimated probabilities in two consecutive iterations is smaller than a threshold value. The stopping criterion is chosen for its simplicity; we note that it may cause premature termination of the algorithm, and, if desired, more complicated stopping rules can be used, which, though, may result in higher computational cost.

4. Numerical examples

4.1. Branin function

To compare the performance of our method with SUR, we first test both methods on the rescaled Branin function

$$g(x_1, x_2) = 80 - [(15x_2 - \frac{5}{4\pi^2}(15x_1 - 5)^2 + \frac{5}{\pi}(15x_1 - 5) - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(15x_1 - 5) + 10], \tag{14}$$

which is plotted in Fig. 2. Here we assume that x_1 and x_2 are independent and each follows a uniform distribution in $[0, 1]$. The function is included in the R package KrigInv [10] as an example for the SUR method and the numerical results suggest that SUR performs well for this numerical example.

We first perform a standard MC with 10 000 samples, resulting in a failure probability estimate of 0.256, which will be used as the "true" failure probability to validate our results. In the experimental design, we first choose 4 points with the Latin hypercube sampling (LHS) method as the initial design points for both methods. We then sequentially choose 9 points with both methods, where one point is determined in each iteration. In all the three numerical examples, we choose a squared exponential covariance:

$$K(\mathbf{x}, \mathbf{x}') = \alpha \exp(-\frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{\beta}), \tag{15}$$

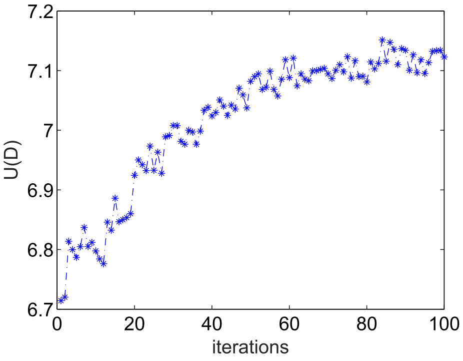

Fig. 3. The function $U(\mathbf{D})$ plotted against the SPSA iterations.

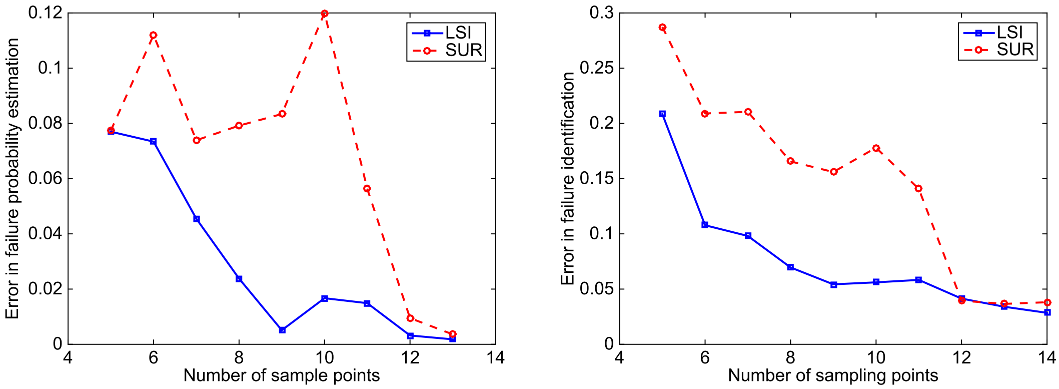

Fig. 4. Left: the error in failure probability estimation plotted against the number of design points. Right: the error in the failure detection plotted against the number of design points.

and the hyperparameters α and β are determined by maximizing the marginal likelihood function with the initial data points, where the implementation details can be found in (Ch. 5, [37]). If desired, the hyperparameters can also be updated with the maximum marginal likelihood method as more data points become available. The prior mean is determined with a quadratic regression. In all the three examples, the algorithm is terminated after 100 iterations in the SPSA method, and in each iteration, 5000 samples are used to evaluate the value of $U(\mathbf{D})$. To demonstrate the performance of the SPSA method, we plot the values of $U(\mathbf{D})$ against the number of iterations in Fig. 3.

We draw 10 000 realizations from the uniform distribution, and each time a new design point is added, we use the resulting GP surrogates to detect which points are in the failure domain and which are not. Using the detection results we compute the failure probability. To compare the performance of the two methods, we compute two quantities as the indicators of performance. The first is the error between the failure probability estimate and the true failure probability $|\hat{P} - P|$, which is referred to as *the error in failure probability estimation*. The second is the probability that a point is mis-identified:

$$\Theta_2 = \mathbb{P}[\mathbf{x} \in \{\mathbf{x}|g(\mathbf{x}) < 0, \hat{g}(\mathbf{x}) > 0\} \cup \{\mathbf{x}|g(\mathbf{x}) > 0, \hat{g}(\mathbf{x}) < 0\}],$$

where $\hat{g}(\mathbf{x})$ represents the surrogate, and this indicator is referred to as *the error in failure identification*. In Fig. 4, we plot both indicators as a function of the number of design points for our method and the SUR. The results of both indicators suggest that our LSI method have a better performance than SUR in this example.

4.2. Four branch system

Our second example is the so-called four branch system, which is a popular benchmark test case in reliability analysis. In this example the limit state function reads

$$g(x_1, x_2) = \min \left\{ \begin{array}{l} 3 + 0.1(x_1 - x_2)^2 - (x_1 + x_2)/\sqrt{2} \\ 3 + 0.1(x_1 - x_2)^2 + (x_1 + x_2)/\sqrt{2} \\ (x_1 - x_2) + 7/\sqrt{2} \\ (x_2 - x_1) - 7/\sqrt{2} \end{array} \right\},$$

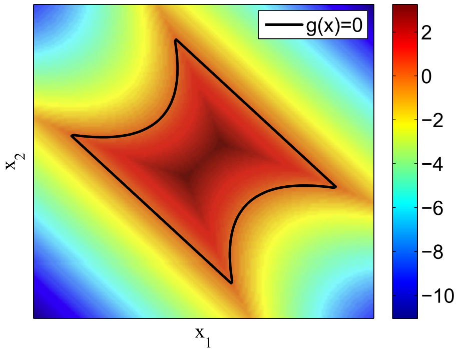

Fig. 5. The limit state function of the four-branch model. The solid lines in both figures are the limit state $g(\mathbf{x}) = 0$.

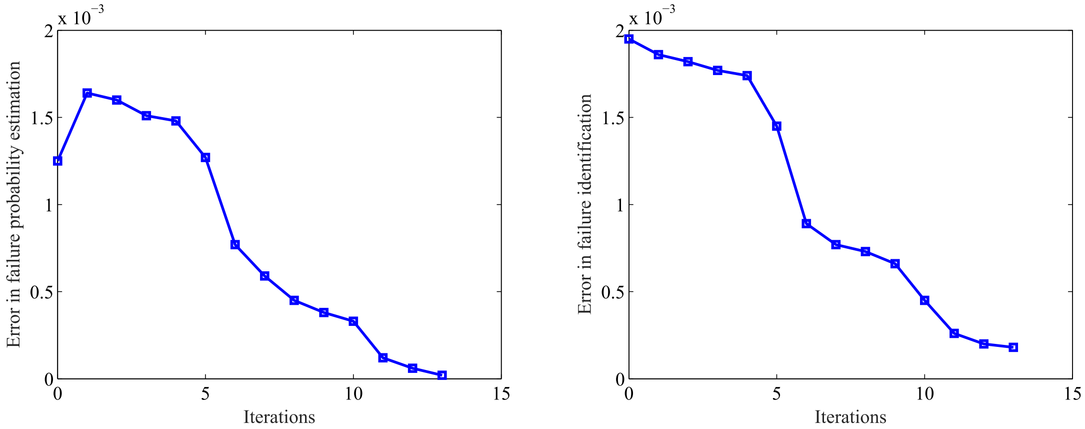

Fig. 6. Left: the probability of detection error plotted against the number of design points. Right: the error in the failure probability estimates plotted against the number of design points.

which is shown in Fig. 5. The input random variable x_1 and x_2 are assumed to be independent and follow standard normal distribution. We first compute the failure probability with a standard MC estimation of 10^5 samples, resulting an estimate of 2.34×10^{-3}, i.e., 234 samples fall in the failure domain.

The prior mean of the GP surrogate is computed with a quadratic regression. We perform the sequential design described in Section 3 with 4 sampling points determined in each iteration. The algorithm terminates in 13 iterations, resulting in totally 62 design points. We plot the errors in failure probability estimation as a function of the number of iterations in Fig. 6 (left), and the mis-detection probability, also as a function of the number of iterations in Fig. 6 (right). In Figs. 7, we plot the initial design points and the design points found in the first, the 7th, and the last iterations. As a comparison, we also computed the same number of design points with the Latin hypercube sampling (LHS) method, and in Fig. 8 (left), we show the design points obtained with both methods, and also shown in the figures are the GP surrogates constructed with the obtained design points. We can see that our method allocates more points near the boundary of the failure domain than LHS. We now compare the results of the two GP surrogates. The failure probability estimated with our method is 2.31×10^{-3}, which is very close to the true failure probability 2.34×10^{-3}, while the result of LHS is 1.19×10^{-3}, which clearly underestimates the probability. Also, we find that totally 19 samples are incorrectly identified: 11 safe samples are identified as failures, and 8 failure samples are identified as safe ones. For the surrogate constructed with LHS, 205 samples are incorrectly identified: 45 safe ones are identified as failures, and 160 failures are identified as safe ones. The numerical results indicate that our method can obtain much more accurate results than the conventionally used LHS method.

4.3. Clamped beam dynamics

Our last example is a practical problem which concerns the dynamic behavior of a beam clamped at both ends and a uniform pressure load is suddenly applied as shown in Fig. 9. We are interested in the dynamics of the deflections at the center of the beam caused by the pressure.

The Lagrangian formulation of the equilibrium equations without body forces can be written

$$\rho \frac{d^2\mathbf{u}}{dt^2} - \nabla \cdot (\mathbf{F} \cdot \mathbf{S}) = 0,$$

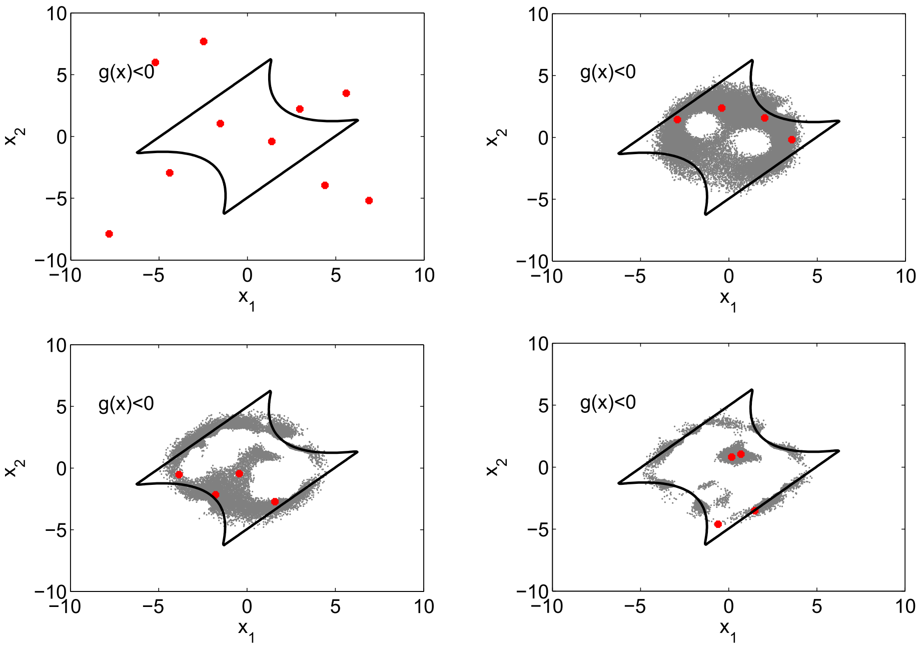

Fig. 7. Top: the left figure shows the initial design points and the right one shows the design points determined in the first iteration. Bottom: the left and the right figures are the design points determined in the 7th and the 13th iteration respectively.

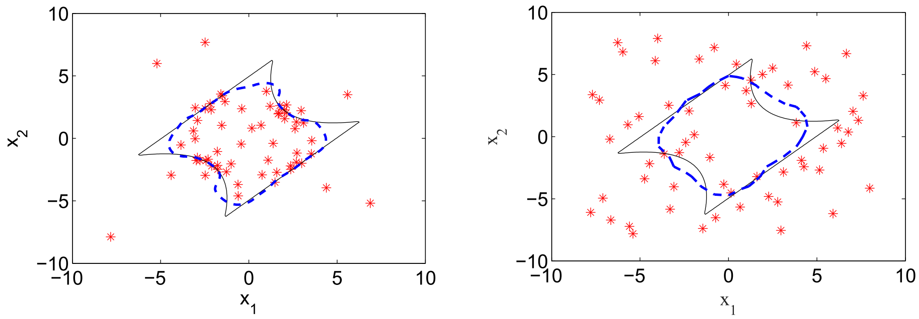

Fig. 8. The design points (asterisks), chosen by our method (left) and by the Latin hypercube (right). The true limit stage (solid line) and that computed with the GP surrogate (dashed line) are also shown in both plots.

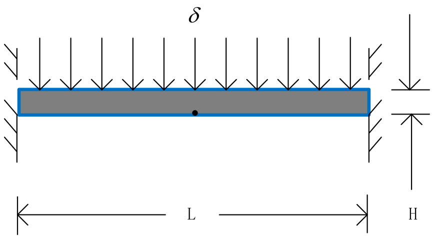

Fig. 9. Schematic illustration of the clamped beam.

where ρ is the material density, $\mathbf{u}$ is the displacement vector, $\mathbf{F}$ is the deformation gradient,

$$\mathbf{F} = \frac{\partial u_i}{\partial x_k} + \delta_{ik}$$

and $\mathbf{S}$ is the second Piola–Kirchoff stress tensor. For simplicity, we assume linear elastic constitutive relations and isotropic material. As a result, the constitutive relations may be written in matrix form:

Table 1
Mean and variance of the random variables in the clamped beam example.

parameter	L	h	ρ	E	ν	δ
mean	5	0.1	7.77×10^{-4}	3×10^{7}	0.3	200
variance	6.25×10^{-2}	2.5×10^{-5}	1.51×10^{-9}	3.6×10^{13}	2.25×10^{-4}	40

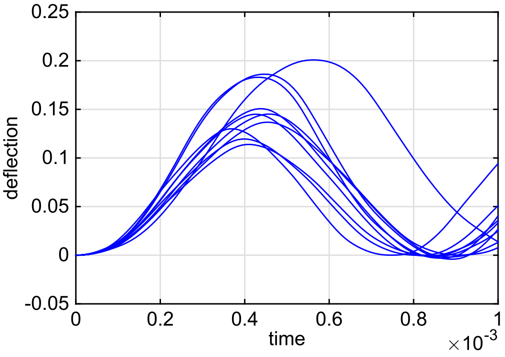

Fig. 10. Dynamics of the deflection at the beam center for ten randomly generated samples.

$$\mathbf{S} = \begin{Bmatrix} S_{11} \\ S_{22} \\ S_{12} \end{Bmatrix} = \begin{bmatrix} C_{11} & C_{12} & \\ C_{12} & C_{22} & \\ & & 2G_{12} \end{bmatrix} \begin{Bmatrix} E_{11} \\ E_{22} \\ E_{12} \end{Bmatrix}$$

where

$$E_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} + \frac{\partial u_k}{\partial x_i}\frac{\partial u_k}{\partial x_j}\right),$$

and

$$C_{11} = C_{22} = \frac{E}{1-\nu^2}, \quad C_{12} = \frac{E\nu}{1-\nu^2}, \quad G_{12} = \frac{E}{2(1+\nu)}.$$

Here E is Young's modulus and ν is Poisson's ratio. The initial conditions are

$$u(0,\mathbf{x}) = 0, \quad \frac{\partial u}{\partial t}(0,\mathbf{x}) = 0.$$

Readers who are interested in more details about the Lagrangian formulation for nonlinear elasticity can consult, for example, [23].

We assume in this example that the beam length L, the height h, the material density ρ, the Young's module E, the Poisson ratio ν, and the applied load δ are random. All the random parameters follow a normal distribution and are independent to each other. The means and the variances of the parameters are summarized in Table 1. To demonstrate the statistical variations of the beam dynamics, we randomly generate 10 parameter samples and plot the 10 resulting beam dynamics in Fig. 10. We define the failure as the maximum deflection at the center of the beam being larger than a threshold value $u_{\max}$. To be specific, we take $u_{\max} = 0.23$. We first run standard MC with 10^5 samples to compute the failure probability, resulting in an estimate of 3.35×10^{-3}.

In the GP method, we use 64 initial sampling points drawn by the hyper Latin cube approach. Our algorithm determines 4 points in each iteration and it is terminated after 25 iterations, resulting in totally 164 sampling points. As before, we plot the errors in failure probability estimation and the mis-detection probability, both against the number of iterations, in Figs. 11. In both plots we can see that the accuracy of the estimations improve rapidly as the number of sampling points increases, indicating that our methods can effectively identify good sampling points for this practical problem. Our failure probability estimate is 3.35×10^{-3} while 12 points in the failure domain is mis-identified as safe ones and 10 safe ones are mis-identified as failures.

5. Conclusions

In conclusion, we have presented an experimental design based method to construct GP surrogate for failure probability estimations. In particular, the method recasts the failure detection as inferring a contour $g(\mathbf{x}) = 0$ with Bayesian methods, and then determines the optimal sampling locations for the inference problem. An efficient numerical implementation of the resulting design problem, based on the SPSA algorithm, is also presented. With numerical examples, we demonstrate

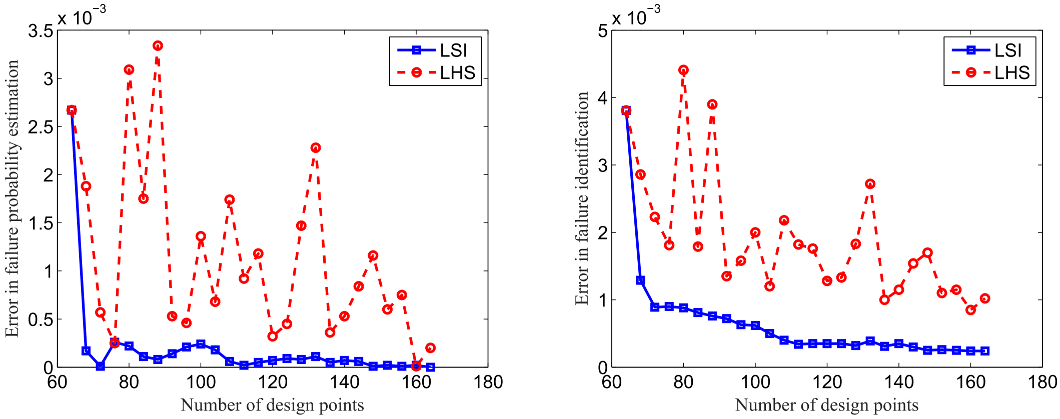

Fig. 11. Left: the error in failure probability estimation plotted against the number of design points. Right: the error in the failure identification plotted against the number of design points. In both plots, the solid lines with squares are the results of our method and the dashed lines with circles are the results of LHS.

that the proposed LSI method can effectively and efficiently determine sampling points for failure detection and failure probability estimation.

There are some improvements and extensions of the method that we plan to investigate in the future. First, this work focuses on the method itself, and the examples provided are typical benchmark problems that are used to demonstrate the performance of the method. In the future, we plan to apply the proposed method to real-world applications. Secondly, the computational efficiency of the method can be improved by utilizing some existing tools. For example, Oakley [26] and Zitrou et al. [38] present an efficient method based on the Bayesian quadrature [28,19] to compute the expectations of the utility function. It is possible to apply such a method to computing $U(\mathbf{D})$ in the present problem. Thirdly, from an applied point of view, it is highly desirable to be able to quantify the uncertainty in the estimation result, e.g. to provide an uncertainty bound of it. This issue is very important in practice and should be investigated in future studies. Finally, in addition to failure probability estimations, the similar ideas can be used to address many other problems as well. In particular, we plan to use the method to solve Bayesian inverse problems with computationally intensive forward models.

Acknowledgement

We would like to thank the two anonymous reviewers for their very helpful comments. The work was partially supported by the National Natural Science Foundation of China under grant number 11301337. G. Lin would like to acknowledge the support of the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program as part of the Multifaceted Mathematics for Complex Energy Systems (M^{2}ACS) project and part of the Collaboratory on Mathematics for Mesoscopic Modeling of Materials project, and NSF Grant DMS-1115887.

Appendix A. The SPSA method

Here we briefly introduce the SPSA method, in the context of our specific applications. In each step, the method only uses two random perturbations to estimate the gradient regardless of the problem's dimension, which makes it particularly attractive for high dimensional problems. Specifically, in step j, one first draw a n_d dimensional random vector $\mathbf{\Delta}_j = [\delta_{j,1}, \ldots \delta_{j,n_d}]$, where $n_d = \dim(\mathbf{D})$, from a prescribed distribution that is symmetric and of finite inverse moments. The algorithm then updates the solution using the following equations:

$$\mathbf{D}_{j+1} = \mathbf{D}_j - a_j b_j(\mathbf{D}_j), \tag{A.1a}$$

$$b_j(\mathbf{D}_j) = \frac{\hat{U}(\mathbf{D}_j + c_j\mathbf{\Delta}_j) - \hat{U}(\mathbf{D}_j - c_j\mathbf{\Delta}_j)}{2c_j}\mathbf{\Delta}_j^{-1}, \tag{A.1b}$$

where $\hat{U}(\cdot)$ is computed with Eq. (13) and

$$\mathbf{\Delta}_j^{-1} = \left[\Delta_{j,1}^{-1}, \ldots, \Delta_{j,n_d}^{-1}\right]^T, \quad a_k = \frac{a}{(A+k+1)^\alpha}, \quad c_k = \frac{c}{(k+1)^\gamma},$$

with A, α, c, and γ being algorithm parameters. Following the recommendation of [35], we choose $\mathbf{\Delta}_j \sim$ Bernoulli(0.5) and $A = 100$, $\alpha = 0.602$, $c = 1$, and $\gamma = 0.101$ in this work.

Appendix B. Derivation of Equation (3.6)

From Eq. (9), we have

$$\begin{aligned} U(\mathbf{D}) &= \int\int p(\mathbf{z}|\mathbf{y},\mathbf{D}) \ln\left[\frac{p(\mathbf{z}|\mathbf{y},\mathbf{D})}{p(\mathbf{z})}\right] d\mathbf{z}\, p(\mathbf{y}|\mathbf{D})\, d\mathbf{y} \\ &= \int\int \ln\left[\frac{p(\mathbf{y}|\mathbf{z},\mathbf{D})}{p(\mathbf{y}|\mathbf{D})}\right] p(\mathbf{y}|\mathbf{z},\mathbf{D})\, p(\mathbf{z})\, d\mathbf{z}\, d\mathbf{y} \\ &= \int\int \ln(p(\mathbf{y}|\mathbf{z},\mathbf{D}))p(\mathbf{y}|\mathbf{z},\mathbf{D})\, p(\mathbf{z})\, d\mathbf{z}\, d\mathbf{y} - \int\int \ln[p(\mathbf{y}|\mathbf{D})]p(\mathbf{y}|\mathbf{z},\mathbf{D})\, p(\mathbf{z})\, d\mathbf{z}\, d\mathbf{y} \end{aligned} \tag{B.1}$$

Recall that $p(\mathbf{y}|\mathbf{z},\mathbf{D})$ is multivariate normal distribution:

$$p(\mathbf{y}|\mathbf{z},\mathbf{D}) = \frac{1}{(2\pi)^{n/2}\cdot|\mathsf{C}|^{1/2}}\cdot \exp[-\frac{1}{2}(\mathbf{y}-\mathbf{u})'\mathsf{C}^{-1}(\mathbf{y}-\mathbf{u})],$$

because $\mathbf{u}$ and C only depend on $\mathbf{z}$ and $\mathbf{D}$ ($\mu=\mu(z,d), \mathsf{C}=\mathsf{C}(z,d)$), we change of variable:

$$\mathbf{s}=\mathsf{C}^{-1/2}(\mathbf{y}-\mathbf{u})$$

$$\begin{aligned} &\int\int \ln(p(\mathbf{y}|\mathbf{z},\mathbf{D}))p(\mathbf{z}|\mathbf{z},\mathbf{D})\, p(\mathbf{z})\, d\mathbf{z}\, d\mathbf{y} \\ &= \int\int \left(-\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln|\mathsf{C}| - \frac{1}{2}(\mathbf{y}-\mathbf{u})'\mathsf{C}^{-1}(\mathbf{y}-\mathbf{u})\right)\frac{1}{(2\pi)^{n/2}|\mathsf{C}|^{1/2}}\exp[-\frac{1}{2}(\mathbf{y}-\mathbf{u})'\mathsf{C}^{-1}(\mathbf{y}-\mathbf{u})]p(\mathbf{z})d\mathbf{z}d\mathbf{y} \\ &= \int\int \left(-\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln|\mathsf{C}| - \frac{1}{2}\mathbf{s}'\mathbf{s}\right)\frac{1}{(2\pi)^{n/2}}\exp(-\frac{1}{2}\mathbf{s}'\mathbf{s})p(\mathbf{z})d\mathbf{z}d\mathbf{s} \\ &= -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\int \ln|\mathsf{C}|d\mathbf{z} - \int\int \left(\frac{1}{2}\mathbf{s}'\mathbf{s}\right)\frac{1}{(2\pi)^{n/2}}\exp(-\frac{1}{2}\mathbf{s}'\mathbf{s})p(\mathbf{z})d\mathbf{z}d\mathbf{s} \\ &= -\frac{1}{2}\int \ln|\mathsf{C}|p(\mathbf{z})d\mathbf{z} - \int \left(\frac{1}{2}\mathbf{s}'\mathbf{s}\right)\frac{1}{(2\pi)^{n/2}}\exp(-\frac{1}{2}\mathbf{s}'\mathbf{s})d\mathbf{s} - \frac{n}{2}\ln(2\pi) \end{aligned} \tag{B.2}$$

Note that the second integral on the right hand side of Eq. (B.2) actually does not depend on $\mathbf{D}$ and so we can define a constant Z such that

$$Z = -\int\int \left(\frac{1}{2}\mathbf{s}'\mathbf{s}\right)\frac{1}{(2\pi)^{n/2}}\exp(-\frac{1}{2}\mathbf{s}'\mathbf{s})d\mathbf{s} - \frac{n}{2}\ln(2\pi).$$

It follows immediately that

$$\int\int \ln(p(\mathbf{y}|\mathbf{z},\mathbf{D}))p(\mathbf{z}|\mathbf{z},\mathbf{D})\, p(\mathbf{z})\, d\mathbf{z}\, d\mathbf{y} = -\frac{1}{2}\mathbb{E}_z[\ln|\mathbf{C}|] + Z,$$

which in turns yields Eq. (10).

References

[1] Ibrahim Ahmad, Pi-Erh Lin, et al., A nonparametric estimation of the entropy for absolutely continuous distributions (corresp.), IEEE Trans. Inf. Theory 22 (3) (1976) 372–375.

[2] S.K. Au, J. Beck, Estimation of small failure probabilities in high dimensions by subset simulation, Probab. Eng. Mech. 16 (2001) 263–277.

[3] Mathieu Balesdent, Jerome Morio, Julien Marzat, Kriging-based adaptive importance sampling algorithms for rare event estimation, Struct. Saf. 44 (2013) 1–10.

[4] Julien Bect, David Ginsbourger, Ling Li, Victor Picheny, Emmanuel Vazquez, Sequential design of computer experiments for the estimation of a probability of failure, Stat. Comput. 22 (3) (2012) 773–793.

[5] Jan Beirlant, Edward J. Dudewicz, Laszlo Gyorfi, Edward C. Van der Meulen, Nonparametric entropy estimation: an overview, Int. J. Math. Stat. Sci. 6 (1) (1997) 17–39.

[6] Barron J. Bichon, Michael S. Eldred, Laura Painton Swiler, Sandaran Mahadevan, John M. McFarland, Efficient global reliability analysis for nonlinear implicit performance functions, AIAA J. 46 (10) (2008) 2459–2468.

[7] Ilias Bilionis, Nicholas Zabaras, Multi-output local Gaussian process regression: applications to uncertainty quantification, J. Comput. Phys. 231 (17) (2012) 5718–5746.

[8] Ilias Bilionis, Nicholas Zabaras, Bledar A. Konomi, Guang Lin, Multi-output separable Gaussian process: towards an efficient, fully Bayesian paradigm for uncertainty quantification, J. Comput. Phys. 241 (2013) 212–239.

[9] Clement Chevalier, David Ginsbourger, Julien Bect, Emmanuel Vazquez, Victor Picheny, Yann Richet, Fast parallel kriging-based stepwise uncertainty reduction with application to the identification of an excursion set, Technometrics 56 (4) (2014).

[10] Clément Chevalier, Victor Picheny, David Ginsbourger Kriginv, An efficient and user-friendly implementation of batch-sequential inversion strategies based on kriging, Comput. Stat. Data Anal. 71 (2014) 1021–1034.

[11] Vincent Dubourg, B. Sudret, F. Deheeger, Metamodel-based importance sampling for structural reliability analysis, Probab. Eng. Mech. 33 (2013) 47–57.

[12] B. Echard, N. Gayton, M. Lemaire, Ak-mcs: an active learning reliability method combining kriging and Monte Carlo simulation, Struct. Saf. 33 (2) (2011) 145–154.
[13] S. Engelund, R. Rackwitz, A benchmark study on importance sampling techniques in structural reliability, Struct. Saf. 12 (1993) 255–276.
[14] L. Faravelli, Response surface approach for reliability analysis, J. Eng. Mech. 115 (12) (1989) 2763–2781.
[15] N. Gayton, J.M. Bourinet, M. Lemaire, CQ2RS: a new statistical approach to the response surface method for reliability analysis, Struct. Saf. 25 (2003) 99–121.
[16] Peter Hall, Sally C. Morton, On the estimation of entropy, Ann. Inst. Stat. Math. 45 (1) (1993) 69–88.
[17] Xun Huan, Youssef M. Marzouk, Simulation-based optimal Bayesian experimental design for nonlinear systems, J. Comput. Phys. 232 (1) (2013) 288–317.
[18] Joe Harry, Estimation of entropy and other functionals of a multivariate density, Ann. Inst. Stat. Math. 41 (4) (1989) 683–697.
[19] Marc Kennedy, Bayesian quadrature with non-normal approximating functions, Stat. Comput. 8 (4) (1998) 365–375.
[20] Jing Li, Jinglai Li, Dongbin Xiu, An efficient surrogate-based method for computing rare failure probability, J. Comput. Phys. 230 (24) (2011) 8683–8697.
[21] Li Ling, Julien Bect, Emmanuel Vazquez, Bayesian subset simulation: a kriging-based subset simulation algorithm for the estimation of small probabilities of failure, arXiv:1207.1963, 2012.
[22] Quan Long, Marco Scavino, Raúl Tempone, Suojin Wang, Fast estimation of expected information gains for Bayesian experimental designs based on Laplace approximations, Comput. Methods Appl. Mech. Eng. 259 (2013) 24–39.
[23] Lawrence E. Malvern, Introduction to the Mechanics of a Continuous Medium. Number Monograph, 1969.
[24] Geoffrey McLachlan, David Peel, Finite Mixture Models, John Wiley & Sons, 2004.
[25] Jeremy Oakley, Anthony O'Hagan, Bayesian inference for the uncertainty distribution of computer model outputs, Biometrika 89 (4) (2002) 769–784.
[26] Jeremy E. Oakley, Decision-theoretic sensitivity analysis for complex computer models, Technometrics 51 (2) (2009) 121–129.
[27] Jeremy E. Oakley, Anthony O'Hagan, Probabilistic sensitivity analysis of complex models: a Bayesian approach, J. R. Stat. Soc., Ser. B, Stat. Methodol. 66 (3) (2004) 751–769.
[28] Anthony O'Hagan, Bayes–Hermite quadrature, J. Stat. Plan. Inference 29 (3) (1991) 245–260.
[29] Anthony O'Hagan, J.F.C. Kingman, Curve fitting and optimal design for prediction, J. R. Stat. Soc. B (1978) 1–42.
[30] M.R. Rajashekhar, B.R. Ellingwood, A new look at the response surface approach for reliability analysis, Struct. Saf. 12 (1993) 205–220.
[31] R.Y. Rubinstein, D.P. Kroese, The Cross-Entropy Method, Springer Science + Business Media, Inc., New York, NY, 2004.
[32] Jerome Sacks, William J. Welch, Toby J. Mitchell, Henry P. Wynn, Design and analysis of computer experiments, Stat. Sci. (1989) 409–423.
[33] G.I. Schueller, H.J. Pradlwarter, P.S. Koutsourelakis, A critical appraisal of reliability estimation procedures for high dimensions, Probab. Eng. Mech. 19 (2004) 463–474.
[34] James C. Spall, Multivariate stochastic approximation using a simultaneous perturbation gradient approximation, IEEE Trans. Autom. Control 37 (3) (1992) 332–341.
[35] James C. Spall, Implementation of the simultaneous perturbation algorithm for stochastic optimization, IEEE Trans. Aerosp. Electron. Syst. 34 (3) (1998) 817–823.
[36] Hui Wang, Xiang Zhou, A cross-entropy scheme for mixtures, ACM Trans. Model. Comput. Simul. 25 (1) (2015) 6.
[37] Christopher KI Williams, Carl Edward Rasmussen, Gaussian processes for machine learning, MIT Press 2 (3) (2006) 4.
[38] Athena Zitrou, Tim Bedford, Alireza Daneshkhah, Robustness of maintenance decisions: uncertainty modelling and value of information, Reliab. Eng. Syst. Saf. 120 (2013) 60–71.