Adaptive Experiment Design for Probabilistic Integration

Pengfei Wei [a,*,b], Xing Zhang [a], Michael Beer [b,c,d]

[a] *School of Mechanics, Civil Engineering and Architecture, Northwestern Polytechnical University, Xi'an 710072, China*

[b] *Institute for Risk and Reliability, Leibniz Universität Hannover, Callinstr. 34, Hannover, Germany*

[c] *Institute for Risk and Uncertainty, University of Liverpool, Peach Street, L69 7ZF Liverpool, United Kingdom*

[d] *International Joint Research Center for Engineering Reliability and Stochastic Mechanics, Tongji University, Shanghai 200092, China*

Abstract: Probabilistic integration is a Bayesian inference technique for numerical integration, and has received much attention in the community of scientific and engineering computations. The most appealing advantages are the ability to improve the integration accuracy by making full use of the spatial correlation information among the design points, and the treatment of discretization error as a source of epistemic uncertainty being explicitly propagated to the integration results. This paper aims to develop an adaptive algorithm for further improving the efficiency and accuracy of the probabilistic integration when it is applied to the time-consuming computer simulators. A learning function is first extracted from the posterior variance of the integration and is shown to be especially useful for identifying the design point, by adding which to the training data set, the most reduction of the posterior variance of integration can be achieved. Based on this learning function, an adaptive experiment design algorithm is then developed for actively producing optimal design points. Results of the experiment tests and engineering application show that, with the same number of design points, the developed design strategy always produce more accurate and robust integration results, than the three kinds of commonly used random sampling design strategies (i.e., Monte Carlo design, Latin-hypercube design and

* Corresponding author at: School of Mechanics, Civil Engineering and Architecture, Northwestern Polytechnical University, Xi'an 710072, China

Email address: pengfeiwei@nwpu.edu.cn (P. Wei)

Sobol sequence).

Keywords: Probabilistic Integration; Epistemic Uncertainty; Gaussian Process Regression; Experiment Design; Kernel Function; Bayesian Inference

1. Introduction

The numerical integration algorithms based on, e.g., discretization and stochastic simulation, play a core role in almost all areas of modern scientific and engineering computations such as computational mechanics, uncertainty quantification and computational physics. However, due to the limited computational resources and the increasing complexity of the simulators, those well-established algorithms have reached the ceiling, but still cannot satisfy the needs of scientific and engineering computations, especially when it comes to the time-consuming computer simulators such as finite element analysis of multi-physics fields [1]. Pursuing numerical integrations with better efficiency and accuracy is always on the way.

There are three important elements in a numerical integration algorithm, i.e., (i) the design points at which the values of the integrand need to be computed (the most time-consuming part), (ii) the integration rule with which the integration is calculated, and (iii) the discretization errors of different forms which need to be controlled and measured. Based on these three elements, the available algorithms can be divided into three groups, named as deterministic integration, stochastic simulation, and probabilistic integration respectively.

The deterministic integration, such as the classical Gaussian-Hermite integration and the sparse grid integration [2], is based on the well-designed integration points such that the integral errors can be limited to zero for specific orders of polynomial integrands. For implicit integrands with unknown behavior, it is difficult to assess the numerical errors, and for high-dimensional integrals, the required number of integrand calls can be extremely demanding. The computational cost of stochastic simulation is commonly less sensitive to

the dimension, and the convergence of this group of algorithms is promised by the law of large numbers and the central-limit theorem. This group of methods includes the crude Monte Carlo (MC) simulation [3], the quasi MC simulation (such as Latin Hypercube Sampling (LHS) [4] and Sobol sequence [5]) and the advanced MC simulation especially those developed for estimating the probability of rare events [6][7]. A typical character of stochastic simulation is that they regard the discretization errors as a kind of statistical error, and measure the errors by the variance of the estimator [3]. Many approaches have been developed for controlling the variance (thus the integration error) of the estimator such as control variates [8] and control functionals [9].

The (Bayesian) probabilistic integration is a branch of the Bayesian probabilistic numerical methods which aim at treating the mathematical quantities in numerical computations (such as the solution of partial differential equations) with the philosophy of uncertainty quantification and Bayesian inference [1][10]. Among the past decade, it has received more and more attention in statistical computation and also become a research frontier in many other disciplines such as computational mechanics. Compared with the deterministic integration and the stochastic simulation, the probabilistic integration has two promising characters. First, the spatial correlations among the integration points are integrated into the integration rule to improve the efficiency and accuracy; second, the discretization errors are treated as a kind of epistemic uncertainty, and are analytically formulated for the integration outcomes by posterior variance [11]. Recent studies have shown that the probabilistic integration can outperform the classical deterministic integration and stochastic simulation by several orders of magnitude on efficiency [12].

The probabilistic integration methods are generally based on stochastic process regression models, and in most cases, the Gaussian Process Regression (GPR) is utilized. Under this framework, the performance of the numerical integration can be affected by the kernel functions of the GPR model [13], the prior information of the inference [14] and the

experiment design strategies for generating training points [15], while the focus of this work is on experiment design. The first work on this topic can be dated back to 1991 by O'Hagan [15], where the Gaussian-Hermite integration points were utilized. More recently, the random sampling design strategies, such as MC design, LHS design, importance sampling, are utilized, and the resulting methods are termed as Bayesian MC simulation [16], which has been applied for sensitivity analysis [17] and structural reliability analysis [18].

In this paper, we develop an adaptive experiment design for creating the optimal design points iteratively for probabilistic integration by starting from a small number of random design points. For doing this, a learning function (a concept borrowed from structural reliability analysis [19][20]) is firstly established, which measures the overall contribution of the prediction error at each site to the posterior variance of integration, with the consideration of its correlations with those of all the other sites. The maximum value of the learning function informs the site by adding which to the training data the integration accuracy can be improved the most. Then, based on the learning function, an adaptive experiment design algorithm is proposed for implementing the probabilistic integrations actively. Experiment tests and engineering application show that the proposed algorithm always outperforms the commonly used random sampling design strategies.

The rest of this paper is organized as follows. Section 2 reviews the GPR model and the probabilistic integration, with some more insightful interpretations of the probabilistic integration outcomes. In section 3, the learning function, as well as the adaptive experiment design algorithm, are developed, followed by three numerical experiment test examples, and an engineering application for demonstrating the effectiveness of the proposed algorithm in section 4. Section 5 gives conclusions.

2. Probabilistic Integration

2.1. Problem statement

Consider a supervised learning problem with training data set $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}$ of size

N, where $\boldsymbol{x}$ is a n-dimensional row-wise vector of input variables, and y is a scalar output variable. We rearrange the training data of input variables as a $(N \times n)$-dimensional sample matrix X with its i-th row being $\boldsymbol{x}^{(i)}$, and the training data of the output variable in a N-dimensional column-wise vector $\boldsymbol{y}$. Throughout this paper, we assume that the functional relationship $y = g(\boldsymbol{x})$ between $\boldsymbol{x}$ and y is deterministic, and the training data $\mathcal{D}$ is noise-free. The above assumption is consistent with the settings in computer simulation such as the finite element analysis, where the model response function is commonly abbreviated as g-function. Then our target is to numerically estimate the n-dimensional integral:

$$d = \Pi\left[g(\boldsymbol{x})\right] = \int g(\boldsymbol{x})\pi(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \tag{1}$$

, where $\Pi[\cdot]$ refers to the integral of its argument with respect to the weight density $\pi(\boldsymbol{x})$. For simplicity, we assume that each x_i follows independent standard Gaussian distribution with zero mean and unit variance, and its marginal density function is denoted as $\phi_i(x_i)$. Then we have $\pi(\boldsymbol{x}) = \prod_{i=1}^{n} \phi_i(x_i)$. In subsection 2.3, we will give the reason why we make this assumption, and will also show how to handle the problem if $\pi(\boldsymbol{x})$ is not standard Gaussian density.

Probabilistic integration is driven by the stochastic process surrogate model learning from the training data $\mathcal{D}$. Different types of stochastic process models, i.e., the Student-t process and the Gaussian process (GP), can be assumed, which reflects part of the prior knowledge imposed on the Bayesian probabilistic integration [11]. In this paper, we only consider the GPR model, but the developed method can also be extended to the other types of stochastic process models such as the Student-t process which shows heavier tails [11]. The notations and symbols in this paper are mostly inherited from Ref. [11].

2.2. Gaussian Process Regression model

Given a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, a GP model can be defined as $f : \mathcal{X} \times \Omega \to \mathbb{R}$,

which means that for each $\omega \in \Omega$, $f(\cdot, \omega)$ is a realization of the GP, and given each $\boldsymbol{x} \in \mathcal{X}$ with $\mathcal{X}$ indicating a subspace of $\mathbb{R}^n$ and defining the support of $\boldsymbol{x}$, $f(\boldsymbol{x}, \cdot)$ is a Gaussian random variable.

Given the above notations, the g-function can be approximated by $y = g(\boldsymbol{x}) \cong f(\boldsymbol{x}, \omega) + \varepsilon$, where ε is a (Gaussian) noise random variable utilized for characterizing noise of data and/or the part of $g(\boldsymbol{x})$ that cannot be interpreted by the GP model $f(\boldsymbol{x}, \omega)$. In this paper, we only consider the deterministic simulation models, thus the training data is always noise-free, and we use the noise-free version of the GP model, that is:

$$y = g(\boldsymbol{x}) \cong f(\boldsymbol{x}, \omega). \tag{2}$$

With the noise-free setting, the GPR model degrades into a numerical interpolation method. Given the above assumption, the GP model $f(\boldsymbol{x}, \omega)$, or the GPR model $f_{\mathcal{D}}(\boldsymbol{x})$ if it is trained from the data $\mathcal{D}$, is uniquely characterized by its mean function $m(\boldsymbol{x}) = \mathbb{E}_{\omega}[f(\boldsymbol{x}, \omega)]$ and covariance function $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\omega}[(g(\boldsymbol{x}, \omega) - m(\boldsymbol{x}))(g(\boldsymbol{x}', \omega) - m(\boldsymbol{x}'))]$, where $\mathbb{E}_{\omega}[\cdot]$ indicates the expectation taken over all $\omega \in \Omega$. In practical application, the mean function $m(\boldsymbol{x})$ can be assumed to be made of any type of basis functions, e.g., zero, constant and linear, and this assumption reflects the user's prior knowledge imposed on the mean of the GP model. For example, if a linear mean function is assumed, it is formulated as:

. $$m(\boldsymbol{x}) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i \tag{3}$$

, where $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_n)$ is a set of hyper-parameters for the mean function. Below we always denote the hyper-parameters for $m(\boldsymbol{x})$ as $\boldsymbol{\beta}$ no matter which kind of basis functions is utilized.

The covariance function $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ is also called kernel function, and different forms of kernel function can be assumed (see Chapter 4 of Ref. [21] for more details), which reflects

the user's prior knowledge on the structure of the covariance function. In this paper, the squared exponential kernel with different correlation length parameter for each input variable is utilized, and it is formulated as:

$$\kappa(\boldsymbol{x},\boldsymbol{x}') = \sigma_0^2 \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{x}')^{\top}\right) \tag{4}$$

, where $\Sigma = \mathrm{diag}\left(\sigma_1^2,\ldots,\sigma_n^2\right)$ with σ_i ($i=1,\ldots,n$) being the length scale of x_i. This kernel function has been integrated into the Matlab GPR toolbox with training function being "fitrgp", which is used in the experimental tests of this paper.

With the above definitions, the GP model $f(\boldsymbol{x},\omega)$ is uniquely defined by the hyper-parameters $\boldsymbol{\beta}$, σ_0 and Σ. The values of the hyper-parameters are estimated numerically by, e.g., maximizing the likelihood, and one can refer to Chapter 5 of Ref. [21] for details. Once the hyper-parameters being determined from the training data $\mathcal{D}$, the posterior prediction of the GPR model at a new site $\boldsymbol{x}$ is a Gaussian random variable with mean and variance being

$$\mathbb{E}_{\omega}\left[f_{\mathcal{D}}(\boldsymbol{x})\right] = m(\boldsymbol{x}) + \boldsymbol{\kappa}(\boldsymbol{x},X)^{\top} K^{-1}\left(\boldsymbol{y}-\boldsymbol{m}(X)\right) \tag{5}$$

, and

$$\mathbb{V}_{\omega}\left[f_{\mathcal{D}}(\boldsymbol{x})\right] = \kappa(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{\kappa}(\boldsymbol{x},X)^{\top} K^{-1}\boldsymbol{\kappa}(\boldsymbol{x},X) \tag{6}$$

, respectively, where $\boldsymbol{\kappa}(\boldsymbol{x},X)$ indicates a N-dimensional column-wise vector with the i-th component being $\kappa\left(\boldsymbol{x},\boldsymbol{x}^{(i)}\right)$, and K is a $(N\times N)$-dimensional matrix with the (i,j)-th element being $\kappa\left(\boldsymbol{x}^{(i)},\boldsymbol{x}^{(j)}\right)$.

From Eq. (5), the GPR prediction can be regarded as the prior mean $m(\boldsymbol{x})$ plus a linear combination of the kernel function $\boldsymbol{\kappa}(\boldsymbol{x},X)$ over all training data. Eq. (6) indicates that the posterior variance $\mathbb{V}_{\omega}\left[f_{\mathcal{D}}(\boldsymbol{x})\right]$ is equal to the difference between the prior variance $\kappa(\boldsymbol{x},\boldsymbol{x})$ and the term $\boldsymbol{\kappa}(\boldsymbol{x},X)^{\top} K^{-1}\boldsymbol{\kappa}(\boldsymbol{x},X)$ which represents the information being learned from the training data [21]. The (subjective) posterior Gaussian probability

distribution defined by Eqs. (5) and (6) reflects the epistemic uncertainty on the value of $g(\boldsymbol{x})$, and the probabilistic integration rule can propagate this epistemic uncertainty to the estimation of the integral, and thus provides a measure of the error of the probabilistic integration.

2.3. Probabilistic integration rule

Based on the well-trained GPR model $f_{\mathcal{D}}(\mathbf{x})$, the induced integration $\Pi[f_{\mathcal{D}}]$ is also a Gaussian random variable with posterior mean and variance formulated as [11]:

$$\hat{d} = \mathbb{E}_{\omega}\left[\Pi(f_{\mathcal{D}})\right] = \Pi\left[m(\boldsymbol{x})\right] + \Pi\left[\boldsymbol{\kappa}(\boldsymbol{x}, X)^{\top}\right] K^{-1}\left(\boldsymbol{y} - \boldsymbol{m}(X)\right) \tag{7}$$

, and

$$\operatorname{var}\left(\hat{d}\right) = \mathbb{V}_{\omega}\left[\Pi(f_{\mathcal{D}})\right] = \Pi\Pi\left[\kappa(\boldsymbol{x}, \boldsymbol{x}')\right] - \Pi\left[\boldsymbol{\kappa}(\boldsymbol{x}, X)^{\top}\right] K^{-1} \Pi\left[\boldsymbol{\kappa}(\boldsymbol{x}, X)\right] \tag{8}$$

, respectively, where $\Pi\Pi\left[\kappa(\boldsymbol{x}, \boldsymbol{x}')\right]$ refers to the integral of $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ with respect to both arguments under the weight density $\pi(\boldsymbol{x})$ and $\pi(\boldsymbol{x}')$.

Eq. (7) indicates that the posterior mean of the probabilistic integration equals the integral $\Pi\left[m(\boldsymbol{x})\right]$ of the prior mean function plus the improvement part (either positive or negative) $\Pi\left[\boldsymbol{\kappa}(\boldsymbol{x}, X)^{\top}\right] K^{-1}\left(\boldsymbol{y} - \boldsymbol{m}(X)\right)$ learned from the training data; while Eq. (8) reveals that the posterior variance of the probabilistic integration equals the prior variance of the probabilistic integration minus $\Pi\left[\boldsymbol{\kappa}(\boldsymbol{x}, X)^{\top}\right] K^{-1} \Pi\left[\boldsymbol{\kappa}(\boldsymbol{x}, X)\right]$ (positive), which is a measure of the reduction of the integration variance learned from the training data. Therefore, the posterior variance of the integration in Eq. (8) can be interpreted as the residual epistemic uncertainty on the integral after learning some information from the training data.

To generate the analytical expressions for the posterior mean and variance of the integrations in Eqs. (7) and (8), we need first to generate the closed-form expressions for the integral $\Pi\left[m(\boldsymbol{x})\right]$ of the prior mean, the integral $\Pi\Pi\left[\kappa(\boldsymbol{x}, \boldsymbol{x}')\right]$ of the prior kernel and the integral $\Pi\left[\boldsymbol{\kappa}(\boldsymbol{x}, X)\right]$ of the data-based kernel. Derivation of $\Pi\left[m(\boldsymbol{x})\right]$ is trivial

for the commonly used (zero, constant, linear and polynomial) basis functions. However, to generate closed-form expressions for $\Pi\Pi\left[\kappa(\boldsymbol{x},\boldsymbol{x}')\right]$ and $\Pi\left[\boldsymbol{\kappa}(\boldsymbol{x},X)\right]$, some specific properties are required for the forms of the kernel κ and the density π. Table 1 of Ref. [11] provides a non-exhaustive list of distribution π and kernel κ pairs that result in closed-form expressions for both $\Pi\Pi\left[\kappa(\boldsymbol{x},\boldsymbol{x}')\right]$ and $\Pi\left[\boldsymbol{\kappa}(\boldsymbol{x},X)\right]$. Fortunately, the pair of the Gaussian distribution and the squared exponential kernel given by Eq. (4) is listed in this Table, and this is why we assume each x_i follows standard Gaussian distribution. For non-Gaussian weight density, there are two ways to break the obstacle. The first way is to find the corresponding form of the kernel function, and one can refer to Ref. [11] for more details. The second way is to perform a preprocessing for the integral to transform the input variables into standard Gaussian random variables by using e.g., Nataf transformation and Rosenblatt transformation [22], and we show how to do this with the second test example in Section 4.

. Given the standard Gaussian distribution and the squared exponential kernel, the closed-form expressions for $\Pi\left[\boldsymbol{\kappa}(\boldsymbol{x},X)\right]$ and $\Pi\Pi\left[\kappa(\boldsymbol{x},\boldsymbol{x}')\right]$ are given as [16]:

$$\Pi\left[\boldsymbol{\kappa}(\boldsymbol{x},X)\right]=\sigma_0^2\left|\Sigma^{-1}+I\right|^{-1/2}\exp\left[-\frac{1}{2}\mathrm{vec}\left\{\mathrm{diag}\left[X\left(\Sigma+I\right)^{-1}X^{\top}\right]\right\}\right] \tag{9}$$

, and

$$\Pi\Pi\left[\kappa(\boldsymbol{x},\boldsymbol{x}')\right]=\sigma_0^2\left|2\Sigma^{-1}+I\right|^{-1/2} \tag{10}$$

, where $\mathrm{vec}\left\{\mathrm{diag}[\cdot]\right\}$ means formulating a column-wise vector with the diagonal elements of the argument.

It is obvious that the performance of the probabilistic integration rule in Eqs. (7) and (8) depends on the design of the training data $\mathcal{D}$. One of the common ways to generate $\mathcal{D}$ is by random sampling design such as MC, LHS, and Sobol sequence. In the next section, we present the adaptive experiment design for creating $\mathcal{D}$ based on the information we learn from the pre-trained GRP model.

3. Adaptive Experiment Design

As has been interpreted, the variance $\mathbb{V}_{\omega}\left[\Pi\left(f_{\mathcal{D}}\right)\right]$ in Eq. (8) summarizes the epistemic uncertainty of the probabilistic integration $\Pi\left(f_{\mathcal{D}}\right)$ by integrating the epistemic uncertainty of the GPR posterior prediction $f_{\mathcal{D}}\left(\boldsymbol{x}\right)$ at all points $\boldsymbol{x}\in\mathcal{X}$. Thus, for reducing the epistemic uncertainty of the probabilistic integration the most, initially, we need to find the point $\boldsymbol{x}\in\mathcal{X}$, at which the epistemic uncertainty of prediction contributes the most to the posterior variance $\mathbb{V}_{\omega}\left[\Pi\left(f_{\mathcal{D}}\right)\right]$. For determining this point, we define the following learning function:

$$h^{+}\left(\boldsymbol{x}\right)=\Pi'\left[\kappa\left(\boldsymbol{x},\boldsymbol{x}'\right)\right]-\boldsymbol{\kappa}\left(\boldsymbol{x},X\right)^{\top}K^{-1}\Pi'\left[\boldsymbol{\kappa}\left(X,\boldsymbol{x}'\right)\right] \tag{11}$$

, where $\Pi'\left[\cdot\right]$ denotes the integral of its argument with respect to $\boldsymbol{x}'$ under the weight density $\pi\left(\boldsymbol{x}'\right)$. Then, it is easy to prove that:

$$\mathbb{V}_{\omega}\left[\Pi\left(f_{\mathcal{D}}\right)\right]=\Pi\left[h^{+}\left(\boldsymbol{x}\right)\right]. \tag{12}$$

Obviously, $h^{+}\left(\boldsymbol{x}\right)$ measures the contribution of the epistemic uncertainty of the posterior prediction, at the site $\boldsymbol{x}$, to the posterior variance $\mathbb{V}_{\omega}\left[\Pi\left(f_{\mathcal{D}}\right)\right]$, with the consideration of its correlations with all the other sites in $\mathcal{X}$. Thus if we add the point with the maximum value of $h^{+}\left(\boldsymbol{x}\right)$ into the training set, it is expected that a great reduction of the posterior variance $\mathbb{V}_{\omega}\left[\Pi\left(f_{\mathcal{D}}\right)\right]$ can be achieved. This is why we name it as learning function.

Further, if all $\boldsymbol{x}\in\mathcal{X}$ have the same value of $h^{+}\left(\boldsymbol{x}\right)$, it is reasonable to take the point with the highest weight density value as the most important site, thus we define another learning function as:

$$h\left(\boldsymbol{x}\right)=h^{+}\left(\boldsymbol{x}\right)\pi\left(\boldsymbol{x}\right). \tag{13}$$

With this definition, the variance $\mathbb{V}_{\omega}\left[\Pi\left(f_{\mathcal{D}}\right)\right]$ equals the integration of $h\left(\boldsymbol{x}\right)$ with uniform weight. Thus, by sequentially adding the point with the maximum value of $h\left(\boldsymbol{x}\right)$ to the training data set $\mathcal{D}$, the posterior variance $\mathbb{V}_{\omega}\left[\Pi\left(f_{\mathcal{D}}\right)\right]$ is expected to decrease most efficiently.

Given the Gaussian distribution and squared exponential kernel function, the closed-form expression for the learning function $h(\boldsymbol{x})$ can be analytically derived. In Eq. (11), $\Pi'\left[\boldsymbol{\kappa}(X,\boldsymbol{x}')\right]$ is nothing but the transposition of $\Pi\left[\boldsymbol{\kappa}(\boldsymbol{x},X)\right]$ in Eq. (9), and we only need to derive the closed-form expression for $\Pi'\left[\kappa(\boldsymbol{x},\boldsymbol{x}')\right]$, which is formulated as:

$$\Pi'\left[\kappa(\boldsymbol{x},\boldsymbol{x}')\right]=\sigma_0^2\left|\Sigma^{-1}+I\right|^{-1/2}\exp\left[-\frac{1}{2}\boldsymbol{x}(\Sigma+I)^{-1}\boldsymbol{x}^{\top}\right]. \tag{14}$$

Besides the leaning function, we need also to present a stopping criteria for the algorithm. Let denote the posterior coefficient of variation (C.O.V.) of the integration as $\mathrm{cov}\left(\hat{d}\right)=\sqrt{\mathbb{V}_{\omega}\left[\Pi(f_{\mathcal{D}})\right]}\Big/\left|\mathbb{E}_{\omega}\left[\Pi(f_{\mathcal{D}})\right]\right|$, then the stopping criterion can be defined as $\mathrm{cov}\left(\hat{d}\right)<\epsilon$, where ϵ is a user-specified threshold, and can be set to be a small value, e.g., $1\sim5\%$. However, it is found that, when the initial sample size is very small, there is a possibility that this stopping criterion is satisfied although the integration results do not converge. This can be easily avoided by a delayed judgment, which means finishing the algorithm only when the stopping criteria is satisfied for several (e.g., three) times in succession. Then, based on the learning function $h(\boldsymbol{x})$ and the stopping criteria, an adaptive probabilistic integration algorithm is developed with the flowchart shown in Figure 1. In the initialization of the algorithm, the user need also to specify the initial size N_0 of the training data set $\mathcal{D}$ so as to initiate the algorithm. Depending on our experience, the value of this parameter is problem-dependent. Commonly, the smaller the better, but N_0 should also be large enough such that these points do not (approximately) lie on one hyperplane (unless the integrand is approximately linear), in case the algorithm may finish before the estimation converges, due to small posterior covariance of integration. If no prior information is available on the behavior of the g-function, then the N_0 training points can be obtained by any random sampling scheme such as LHS design, but the sampling density should not necessarily set to be $\pi(\boldsymbol{x})$. For example, one can generate N_0 uniform LHS samples in the support of $\pi(\boldsymbol{x})$.

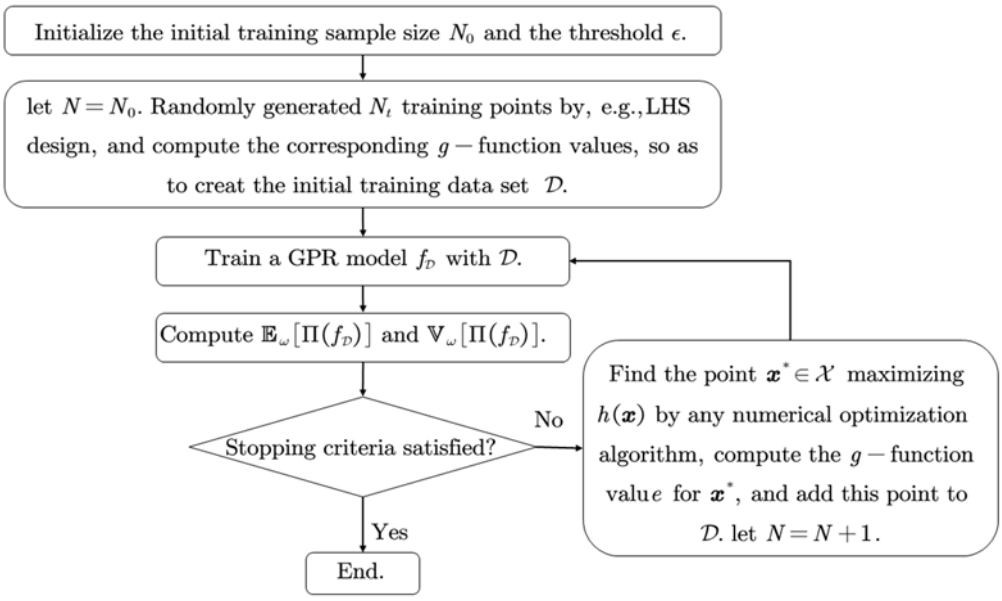

Figure 1 Flowchart of the adaptive experiment design algorithm

One of the key steps of the algorithm is solving the optimization problem which aims at finding the point $\boldsymbol{x}^* \in \mathcal{X}$ maximizing the learning function. For doing this efficiently and accurately, it is better to provide the closed-form gradients of the learning function, which are given in the Appendix. It is obvious that the learning function is a smooth function that has continuous gradients up to infinite order over $\mathcal{X}$, thus the Hessian matrix can also be provided if necessary. For high dimensional problem (e.g., $n > 20$), solving the optimization problem numerically can be time-consuming, and in this case, one can also firstly produce a candidate training data set $\mathcal{D}_{\text{cand}}$ of large size (say 1e4), and then in each iteration, add the point in $\mathcal{D}_{\text{cand}}$, with the maximum value of learning function, to the training data set $\mathcal{D}$. However, this procedure generally needs more training samples (thus g-function calls) since the selected training data in each iteration is sub-optimal.

4. Numerical tests and engineering application

We first introduce a one-dimensional integration problem to illustrate the details of the adaptive training process of the proposed algorithm, and then the Ishigami function and a polynomial integrand with two different settings to test the performance of the algorithm

in the multivariate cases. At last, the proposed algorithm is applied to a dam seepage model to estimate the expected seepage. The performance of the algorithm is compared with three random sampling design strategies (including MC design, LHS design, and Sobol sequence) without adaptive learning.

4.1. One-dimensional integration

For illustrating the adaptive learning process, consider the estimation of the expectation of the *g*-function $g(x) = x^2 \sin(2x) + 1$ with respect to the scalar standard Gaussian random variable x. The true value of the integration is 1.

By setting $N_0 = 3$ and $\epsilon = 1\%$, the proposed algorithm adaptively produces six more training points, and the training process is schematically shown in Figure 2, where the first row shows the comparison of the 95% prediction intervals when the training data size N equals to 3, 5 and 9, respectively, together with the training points and the true *g*-function; the second row presents the learning function $h(\boldsymbol{x})$ as well as the maximum point to be added in the next iteration; and the last row shows the induced posterior density function of the integration, together with the true value of integration. As can be seen, at each iteration, by adding the point specified by the maximum value of $h(\boldsymbol{x})$ to the training data, both the GPR prediction and the induced probabilistic integration can be improved largely. With totally nine training points, both the GPR model and the induced probabilistic integration are accurate enough.

For illustrating the improvement of the adaptive experiment design with respect to the random sampling design, we also implement the probabilistic integration by MC design, LHS design and Sobol sequence with the same number of training points, and the results are compared in Figure 3 and Table 1. As can be seen from Table 1, among the four design strategies, only the adaptive design and Sobol sequence produce posterior 95% confidence intervals with the true value being included, and the confidence interval produced by the adaptive experiment design is much narrow than all the three random sampling design

strategies. From Figure 3, it can also be seen that the adaptive experiment design produces much better results for both regression and probabilistic integration than all the other three kinds of random sampling design strategies. The reason is that, with the adaptive experiment design, the behavior of the g-function reflected by the GPR model is taken into consideration, while the random sampling design strategies do not consider this kind of information. This indicates that the GPR model itself contains valuable information for improving its performance as well as that of the induced probabilistic integration.

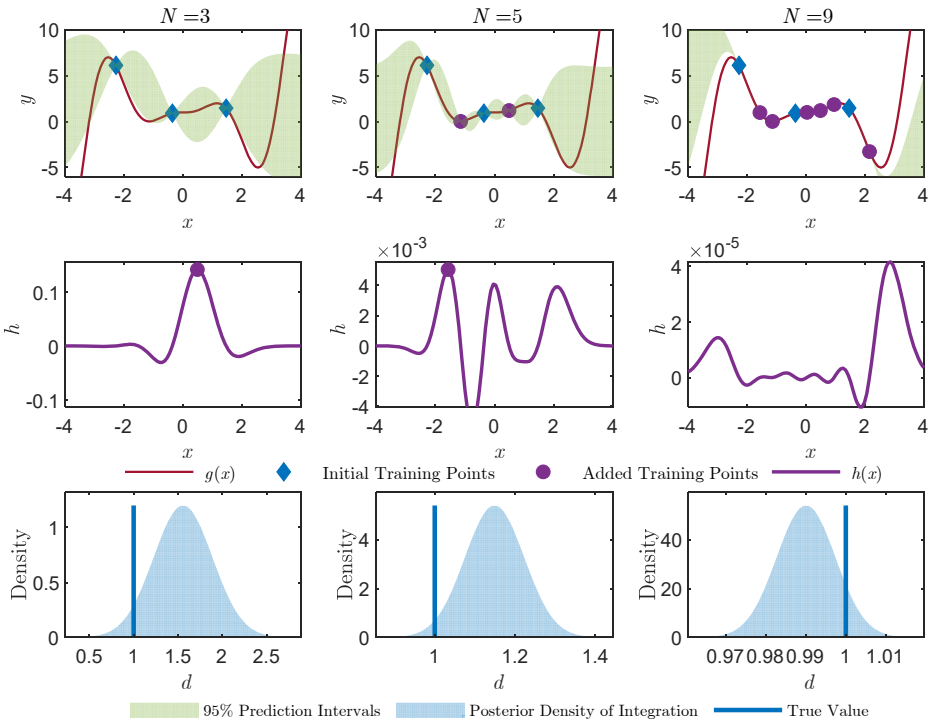

Figure 2 Adaptive training details of the one-dimensional integral, where the first column refers to the initial GPR model generated with three random training samples, the second column shows the details with two more training samples adaptively added, and the last column gives the final results with totally nine training points.

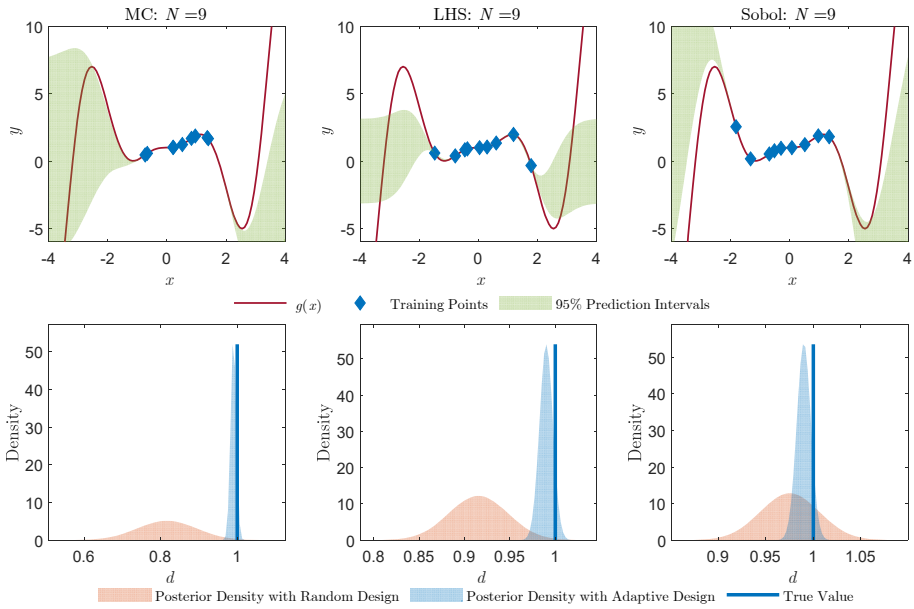

Figure 3 Comparison of the one-dimensional probabilistic integration driven by adaptive experiment design with those driven by crude MC design (first column), LHS design (second column) and Sobol sequence design (third column).

Table 1 Probabilistic integration results for the one-dimensional integrand

Design strategies	Means	C.O.V. (%)	95% Confidence intervals	N
Adaptive Design	0.9900	0.7459	[0.9755, 1.0045]	9
MC	0.8165	9.4500	[0.6653, 0.9677]	
LHS	0.9152	3.5619	[0.8513, 0.9791]	
Sobol	0.9752	3.1885	[0.9143, 1.0361]	
True Value	1			

4.2. Ishigami function

The Ishigami function is a highly nonlinear closed-form model response function widely used in the sensitivity analysis community for testing the performance of different sensitivity indices and related numerical methods [23]. The response function is formulated as:

$$y(\boldsymbol{x}) = \sin x_1 + a\sin^2 x_2 + bx_3^4 \sin x_1 \tag{15}$$

, where a and b are both constants, and set to be 7 and 0.25 respectively, x_1, x_2 and

x_3 are three random input variables, all of which follow a uniform distribution $U(-\pi,\pi)$. Our aim is to estimate the expectation of the model response y, and the analytical value is $0.5a = 3.5$.

Since the input variables follow a uniform distribution, it may be better to use one of the first three kernel functions listed in Table 1 of Ref. [11] so that both the posterior expectation and variance of the integration can be analytically derived. However, as above-stated, in this paper, we only use the squared exponential kernel function given in Eq.(4), thus it is necessary to transform each input variable into a standard Gaussian variable. Let $P(\cdot|-\pi,\pi)$ denote the cumulative distribution function (CDF) of $U(-\pi,\pi)$, and $\Phi(\cdot)$ indicate the CDF of standard Gaussian distribution. Then, the transformation is given as $x_i = T_i(z_i) = P^{-1}(\Phi(z_i)|-\pi,\pi)$. Let $\mathbf{T}(\mathbf{z}) = (T_1(z_1), T_2(z_2), T_3(z_3))$, the expectation of y can be equivalently formulated as $d = \Pi[y(\mathbf{T}(\mathbf{z}))]$. With the above nonlinear transformation, the nonlinearity of the integrand may increase, making the problem more challenging for probabilistic integration. The g-function against z with one variable integrated out is shown in Figure 4, which indicates the nonlinearity.

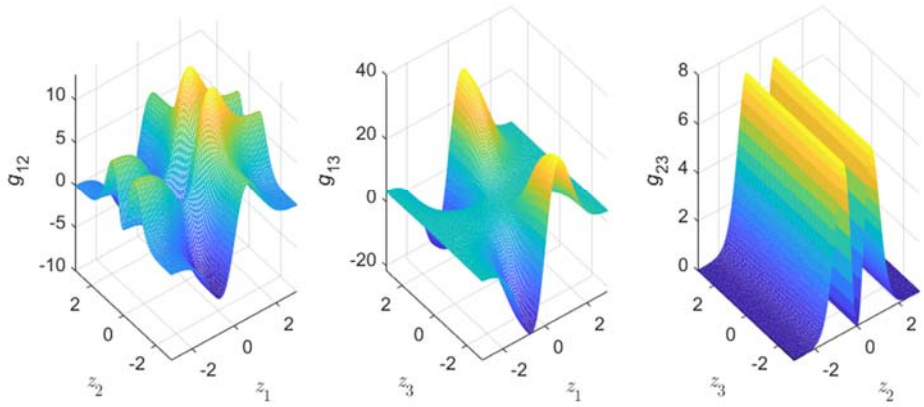

Figure 4 Plot of the Ishigami function again standard Gaussian variables with one variable being integrated out.

For this example, we use the linear basis functions for training the GPR model due to the high nonlinearity. We start the adaptive experiment design by setting $N_0 = 10$ and

$\epsilon = 4\%$. With this setting, the adaptive experiment design produces totally 77 training points before reaching the stopping criteria. Then we implement the probabilistic integration by MC design, LHS design and Sobol sequence with the same data size, and the results are compared with those generated by adaptive design in Table 2 and Figure 5. Both the posterior confidence intervals listed in Table 2 and the posterior density shown in Figure 5 demonstrate that, although the same number of g-function calls are consumed, the adaptive experiment design produces much accurate and robust estimation of the integral than all the three random sampling design strategies.

The active learning process of the adaptive design is illustrated by Figure 6, where the maximum value of the learning function against each training step is shown. It is seen that the general trend is descending, but not always the case in each adjacent steps. This is fair since the GPR model may give biased predictions with small variation in some important local regions where there is no training data. It is also seen that a very small value of $h_{\max}$ does not necessarily mean small variation of the posterior estimation of the integration.

Table 2 Probabilistic integration results for Ishigami function

Experiment Design strategies	Means	C.O.V. (%)	95% Confidence intervals	N
Adaptive Design	3.4516	3.9940	[3.1814, 3.7218]	77
MC	4.1169	10.4361	[3.2748, 4.9590]	
LHS	3.2681	9.2764	[2.6739, 3.8623]	
Sobol	3.6077	10.6419	[2.8552, 4.3602]	
True Value	3.5			

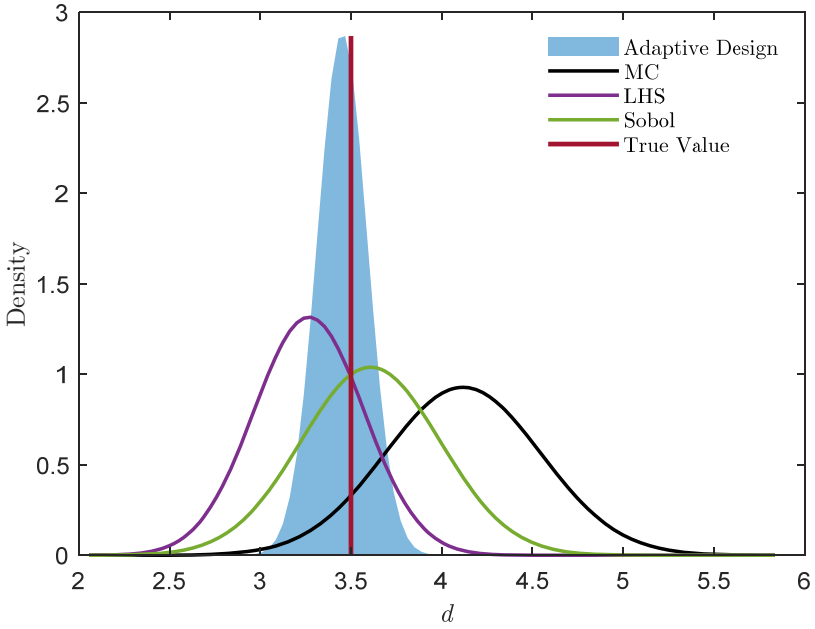

Figure 5 comparison of the probabilistic integration results of Ishigami function with training data generated by adaptive experiment design and three random sampling design strategies.

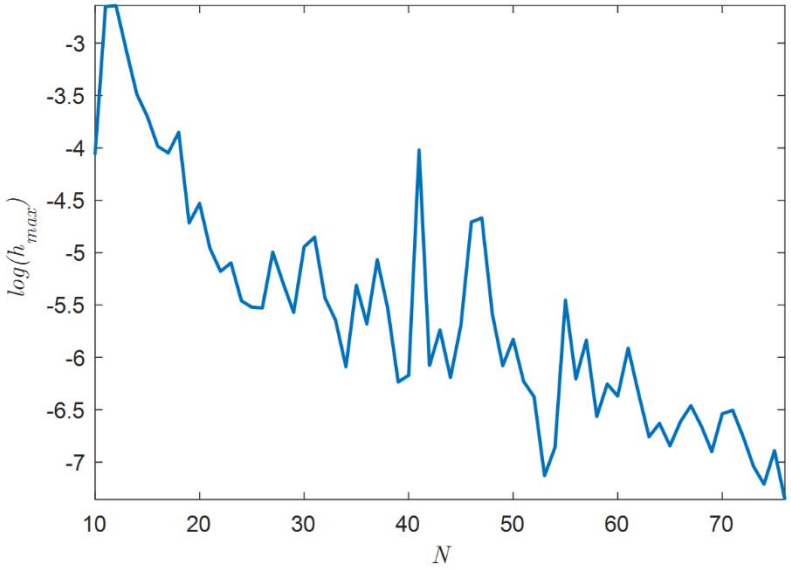

Figure 6 Logarithmic plot of the maximum value of the learning function against the training data size.

4.3. Polynomial integrand

We develop a polynomial integrand formulated as:

$$y(\boldsymbol{x}) = a + \sum_{i=1}^{n} \frac{i}{n} x_i + \sum_{i=1}^{n-1} \frac{i}{n} x_i x_{i+1} + \sum_{i=1}^{n-2} \frac{i}{n} x_i x_{i+1} x_{i+2} \tag{16}$$

, where each input variable x_i follows standard Gaussian distribution, n is the input dimension, and a is a constant, which equals the expectation of y, i.e., $d = \mathbb{E}\left[y(\mathbf{x}) \right] = a$. Our target is to estimate this expectation, and we set the value of a to be one. We consider two cases for this example. In case one, n is set to be ten, while in case two, it is set to be twenty.

For both cases, we set $N_0 = 30$ and $\epsilon = 4\%$, and the constant basis function is utilized for GPR model. For case one, the adaptive experiment design algorithm produces 58 more training points before meeting the stopping criteria, thus the total number of g-function calls is 88. We then use the three kinds of random sampling design with the same data size to do the probabilistic integration, and the results for case one are compared in Table 3 and Figure 7. From Table 3 it is also seen that the posterior 95% confidence interval produced by adaptive design contains the true value 1, and the interval length is much smaller than those generated by the three random sampling design strategies. It can also be found that the 95% confidence interval produced by LHS design even excludes the true value. These results indicate that, 88 training points are far from being enough for all the three random sampling design strategies to produce the same quality of posterior probabilistic integration results as the adaptive experiment design, thus further reveal that the adaptive experiment design outperforms all the three kinds of random sampling design for the probabilistic integration.

For illustrating the adaptive design process, we also plot the maximum value of learning function $h(\boldsymbol{x})$ at each iteration in Figure 8. It is seen that the maximum value of h does not always reduce at each iteration step, but its total varying trend is descending, indicating that the reduction of the posterior variance of the probabilistic integration.

Table 3 Probabilistic integration results for case 1 of the polynomial integrand

Experiment design strategies	Means	C.O.V. (%)	95% Confidence Intervals	N
Adaptive design	0.9746	3.9225	[0.8997, 1.0495]	88
MC	0.9439	9.0633	[0.7762,1.1116]	
LHS	1.1731	7.3033	[1.0052,1.3410]	
Sobol	0.8930	11.0325	[0.6999,1.0861]	
True Value	1			

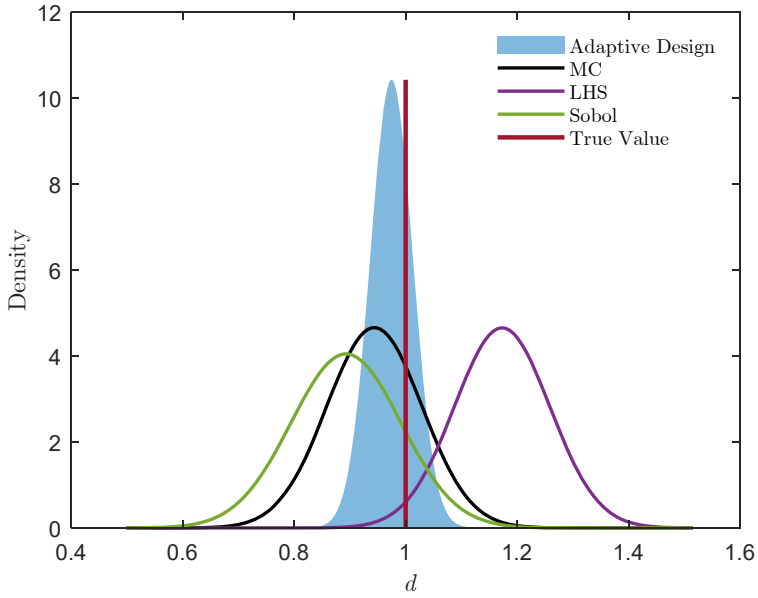

Figure 7 Comparison of the integration results for case one of the polynomial integrand generated by adaptive design and random sampling design with the same size of training data.

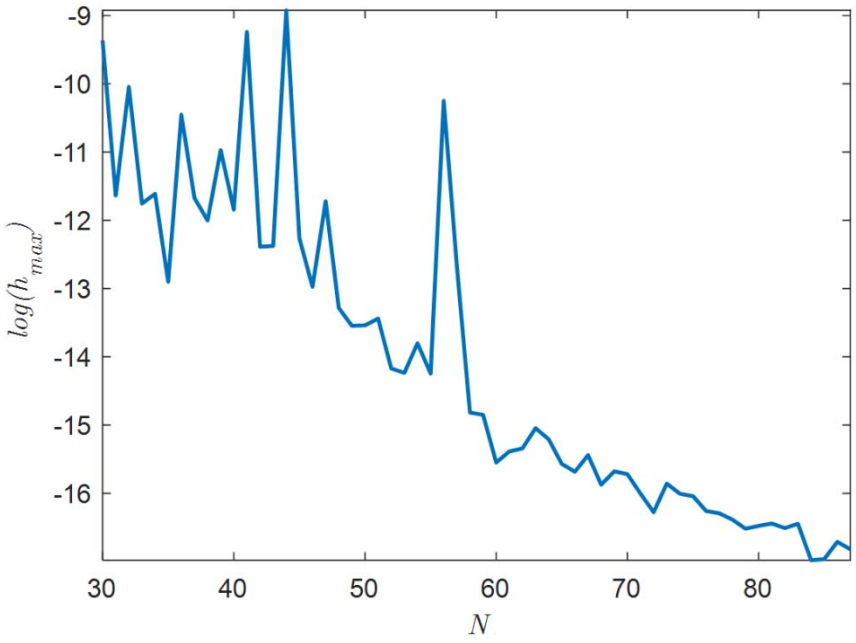

Figure 8 Logarithmic plot of the maximum value of the learning function at each iteration step for case one of the polynomial examples.

For case two, the results are compared in Table 4 and Figure 9. Compared with case one, the adaptive experiment design requires more g-function calls to achieve the same level of accuracy, which is fair due to the higher dimension. Generally, the number of required g-function calls increases with respect to the nonlinearity of g-function and the dimension of the integral. It is shown that, in this case, the adaptive design still produces much more accurate and robust results than the other three kinds of design strategies. This demonstrates that the adaptive experiment design also outperforms the random sampling design for higher-dimensional problems.

Table 4 Results of probabilistic integration for case two of the polynomial example

Experiment design strategies	Means	C.O.V. (%)	95% Confidence Intervals	N
Adaptive design	0.9627	3.9771	[0.8876, 1.0377]	226
MC	0.8761	13.7542	[0.6399, 1.1123]	
LHS	0.9142	11.9235	[0.7005, 1.1278]	
Sobol	1.2803	9.4102	[1.0442, 1.5165]	
True Value	1			

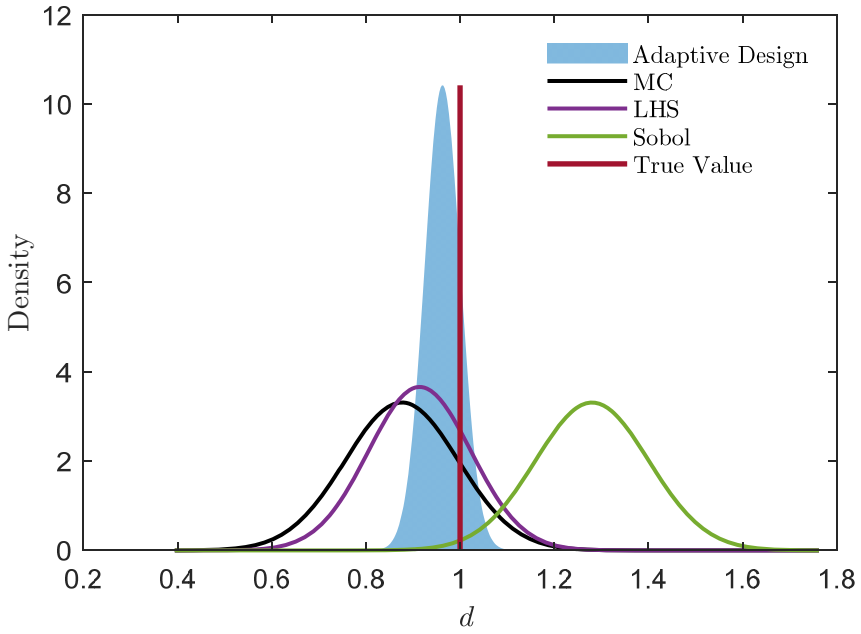

Figure 9 Probabilistic integration results for case two of the polynomial test example.

4.4. Application to a dam seepage model

We then apply the adaptive experiment design to a dam seepage model adapted from Ref. [24]. This model was established to predict the confined seepage below a dam, with elevation shown in Figure 10. One can refer to Ref. [24] for the detailed description of the model. The dam rests over a soil made of two permeable layers and one impermeable layer, where the vertical and horizontal permeability of silty sand layer are denoted as $k_{yy,1}$ and $k_{xx,1}$ respectively, and those of the silty gravel layer are indicated by $k_{yy,2}$ and $k_{xx,2}$ respectively. The depth of the water is denoted as h_D. All these five input variables are random variables with distribution information shown in Table 5.

The governing PDE of this model is formulated as:

$$k_{xx,1}\frac{\partial^2 h_W}{\partial x^2}+k_{yy,i}\frac{\partial^2 h_W}{\partial^2 y}=0,\ \ i=1,2 \tag{17}$$

, where h_W is the hydraulic head over segment AB in Figure 10, with boundary conditions

described in [24]. A finite element model with 3413 nodes and 1628 quadratic triangular elements is established to solve the above PDE numerically. Once h_W being solved, the seepage q can be computed. For example, along the CD segment, the seepage q can be estimated by:

$$q = -\int_{\mathrm{CD}} k_{yy,2} \frac{\partial h_W}{\partial y} \mathrm{d}x \,. \tag{18}$$

The unit of q is [L/h/m], where "L" is the volume, "h" means hour and "m" indicates meter. This application aims to estimate the expected seepage which can be formulated as:

$$d = \mathbb{E}\Big[q\big(k_{xx,1}, k_{yy,1}, k_{xx,2}, k_{yy,2}, h_D\big)\Big]. \tag{19}$$

As shown in Table 5, all the five input variables do not follow Gaussian distribution, thus a nonlinear transformation needs to be carried out for each variable before implementing the probabilistic integration.

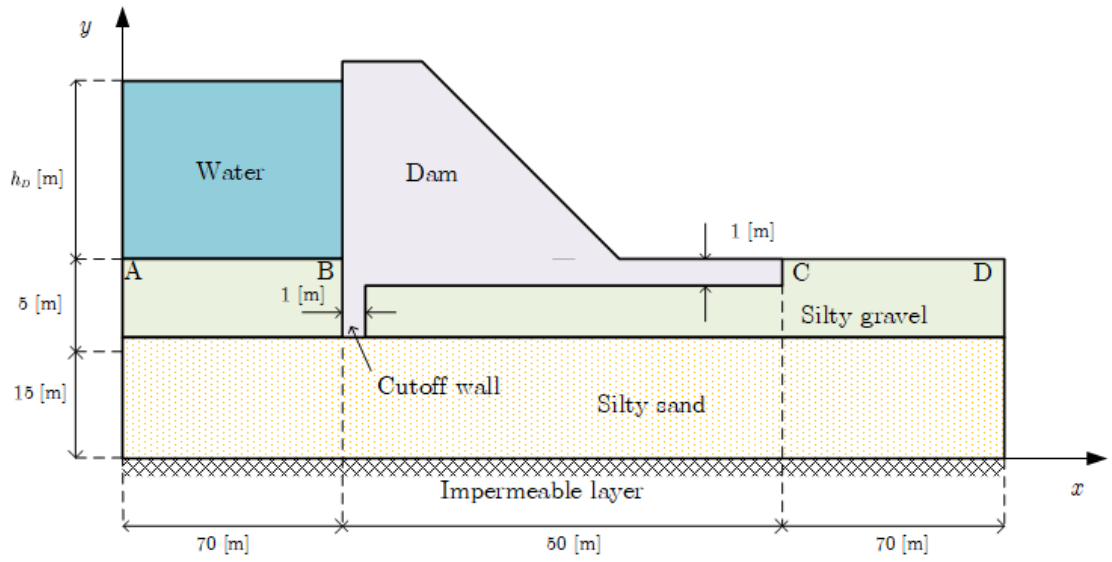

Figure 10 Elevation of a dam

Table 5 Distribution information of the input variables of the dam seepage model

Input variables	$k_{xx,1}$	$k_{yy,1}$	$k_{xx,2}$	$k_{yy,2}$	h_D
Distribution type	Lognormal	Lognormal	Lognormal	Lognormal	Uniform
Distribution Parameters	$\mu = \sigma = 5\times10^{-7}$	$\mu = \sigma = 2\times10^{-7}$	$\mu = \sigma = 5\times10^{-6}$	$\mu = \sigma = 2\times10^{-6}$	[7, 10]

Units	[m/s]	[m/s]	[m/s]	[m/s]	[m]

For implementing the adaptive experiment design, we set $N_0 = 8$ and $\epsilon = 0.5\%$. The adaptive design automatically produces eight more training points, thus the total number of g-function calls is sixteen. The results of the probabilistic integration driven by adaptive design and the three kinds of random sampling design strategies are compared in Table 6 and Figure 11. For this example, the true value of d cannot be derived analytically, thus we use the stochastic simulation driven by LHS design with 1×10^5 samples to calculate the reference solutions, and they are reported in both Table 6 and Figure 11. As can be seen, with the same number of g-function calls, the adaptive experiment design produces much better results than all the other three kinds of design, demonstrating the effectiveness of the adaptive design.

For illustrating the learning process of the adaptive experiment design, the posterior variance of the integration at each iteration step is schematically shown in Figure 12. As can be seen, based on the eight initial training points, with only one more training point being added to the training data set, the posterior variance is largely reduced.

Table 6 Results of probabilistic integration for the dam seepage model, where the reference results are estimated by stochastic simulation with 1×10^5 LHS random samples

Experiment design strategies	Means ($\times10^{-6}$)	C.O.V. (%)	95% Confidence Intervals ($\times10^{-6}$)	N
Adaptive design	2.1836	0.3697	[2.1677, 2.1994]	16
MC	2.0508	1.8704	[1.9756, 2.1260]	
LHS	2.1643	3.3202	[2.0235, 2.3052]	
Sobol	2.2009	2.0711	[2.1116, 2.2902]	
Ref. results	2.1965	0.1673	[2.1893, 2.2036]	10^5

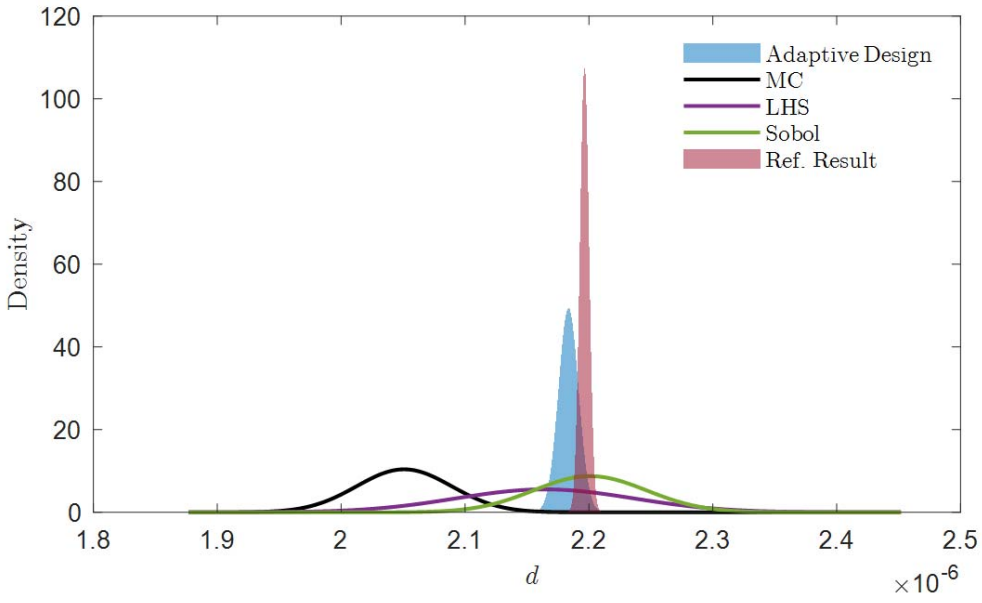

Figure 11 Schematic comparison of results for the dam seepage model.

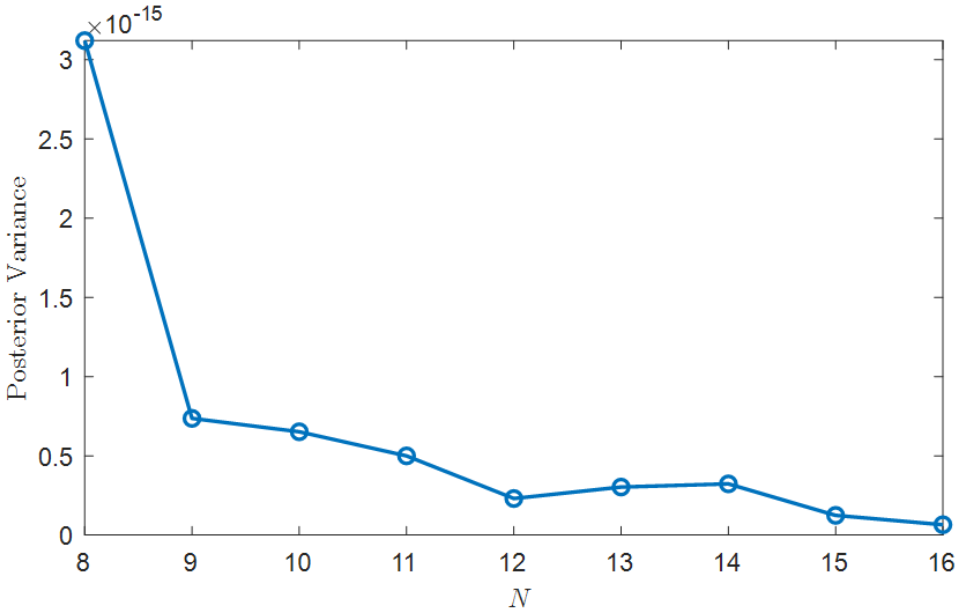

Figure 12 Plot of the posterior variance of the probabilistic integration driven by the adaptive experiment design against the iteration steps.

4.5. Final remarks

The results of the above four test examples have comprehensively proved the effectiveness of the developed adaptive experiment design strategy for improving

probabilistic integration. The success of the strategy is attributed to the spatial correlation information revealed by the GPR model. For totally non-smooth integrands, the strategy may lose its advantage. Fortunately, in most real-world applications, the integrands are smooth or at least piecewisely smooth, indicating that the proposed strategy is of wide applicability.

It's worth mentioning that, in the specific area of reliability analysis, represented by Active learning combining Kriging and MC Simulation (AK-MCS) [19], tremendous active learning strategies have been developed for efficiently estimating the failure probability [20][25]. We denote these methods as AK-MCS class methods, and next, we discuss the differences and links between the adaptive probabilistic integration and those AK-MCS class methods.

For reliability analysis, the target is to estimate the failure probability which is also defined by an integral:

$$p_f = \int I_F(\boldsymbol{x})\pi(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \tag{20}$$

, where $I_F(\boldsymbol{x})$ is the indicator function of the failure domain $F = \{\boldsymbol{x} : g(\boldsymbol{x}) < 0\}$, which equals to one if $g(\boldsymbol{x}) < 0$ or zero if $g(\boldsymbol{x}) \geqslant 0$. The AK-MCS method is then based on creating a sample pool $S = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(N)}\}$ following $\pi(\boldsymbol{x})$, then adaptively selecting training points from S, one by one, with the target to correctly predict the signs for all the points in S by the trained GPR model, and this way to estimate the failure probability with the predicted signs by:

$$\hat{p}_f = \frac{1}{N}\sum_{i=1}^{N} \hat{I}_F\left(\boldsymbol{x}^{(i)}\right) \tag{21}$$

, where $\hat{I}_F\left(\boldsymbol{x}^{(i)}\right)$ is the indicator function value predicted by the trained GPR model for $g(\boldsymbol{x})$.

The difference between AK-MCS and adaptive experiment design is obvious. For failure

probability estimation, the integrand is the (non-smooth) indicator function. The target of the adaptive design process in AK-MCS is to correctly predict the sign for each point in the sample pool S instead of the corresponding g-function values. Since the posterior probability of misjudging the signs of the g-function around this surface is high, the resultant training points are mostly located around the failure surface $g(\boldsymbol{x})=0$. Thus, the AK-MCS class methods provide a high-quality local approximation for the g-function around the failure surface, but for the area far from the failure surface, the accuracy of the g-function approximation is not promised. However, for the integration problem concerned in this work, the integrand is the (commonly smooth) g-function itself, and lack of approximation of the g-function value at any point may contribute significantly to the integration error (posterior variance). Thus the induced training points may spread over the full space of $\boldsymbol{x}$. Another difference lies in the estimators. As shown by Eq. (21), the AK-MCS class methods provide a numerical estimation of the failure probability based on the MCS estimator. Although the variance of the estimator can be easily generated, this variance only accounts for the discretize error caused by the limited sample size in the sample pool, but does not count the error caused by the lack of GPR approximation. Whereas, in the probabilistic integration, given the closed-form expressions for both posterior mean and posterior variance, the error due to the limited sample size does not exist, and that caused by the lack of GPR approximation is explicitly indicated by the posterior variance.

There are also links between these two groups of methods. Based on the rationale of the probabilistic integration, we can further get an in-depth understanding of the AK-MCS class methods from the perspective of Bayesian inference. For failure probability estimation, the integrand is the indicator function $I_F(\boldsymbol{x})$. Given the GPR approximation of the g-function, the induced surrogate model for the indicator function is a Bernoulli process, but the closed-form expressions of the posterior mean and posterior variance of the integration

are not available. However, with the sample pool S, it is possible to derive numerical approximations (e.g., MC estimators) for the posterior mean and posterior variance for the failure probability. For example, the posterior mean can be approximated by

$$\mathbb{E}_\omega\left(\hat{p}_f\right)=\frac{1}{N}\sum_{i=1}^{N}\Phi\left(-\frac{\mathbb{E}_\omega\left[f_{\mathcal{D}}\left(\boldsymbol{x}^{(i)}\right)\right]}{\sqrt{\mathbb{V}_\omega\left[f_{\mathcal{D}}\left(\boldsymbol{x}^{(i)}\right)\right]}}\right) \tag{22}$$

, where $\Phi\left(-\mathbb{E}_\omega\left[f_{\mathcal{D}}\left(\boldsymbol{x}^{(i)}\right)\right]\Big/\sqrt{\mathbb{V}_\omega\left[f_{\mathcal{D}}\left(\boldsymbol{x}^{(i)}\right)\right]}\right)$ is the posterior probability that the g-function value at $\boldsymbol{x}^{(i)}$ being less than zero. One can also derive the MCS estimator for the posterior variance of the failure probability. One note that, in real-world applications, once the AK-MCS algorithms converged, Eq. (22) provides quite similar results as Eq. (21). The above understanding may provide a basis for further improving the AK-MCS class methods by making full use of spatial correlation information, and we will investigate this in our future work.

5. Conclusions

Probabilistic integration, as a kind of Bayesian inference technique based on the stochastic process regression model, has achieved great attention around these years in the community of statistical computation, but may still be unfamiliar to the community of deterministic computer simulation. Compared with the classical deterministic integration and stochastic simulation, the probabilistic integration, on the one hand, makes the best use of the spatial correlation information among integration points to largely improve the integration accuracy, and on the other hand, treats the discretization error as a kind of epistemic uncertainty which allows the analytical propagation to the integration results. The above two characters make probabilistic integration appealing to also deterministic computer simulation due to the large potential improvement on both efficiency and accuracy.

Based on the two appealing characters, we developed an adaptive experiment design strategy for further reducing the required number of g-function calls, each of which can be time-consuming for computer simulation. The key component of this design strategy is the learning function $h(\boldsymbol{x})$, which is effective for finding the unknown point by adding which the posterior variance of the integration can be reduced the most. As has been interpreted, this is due to the fact the $h(\boldsymbol{x})$ measures the contribution of the prediction error (epistemic uncertainty) at each non-training site $\boldsymbol{x}$ with the consideration of its correlation with all the other sites.

The results of the several experimental tests show that the proposed adaptive experiment design always outperforms the commonly used three kinds of random sampling design strategies (MC design, LHS design, and Sobol sequence) since, with the same number of g-function calls, the adaptive experiment design always produces more accurate and robust results. The application to higher dimensional problems needs to be further investigated, by utilizing e.g., dimension reduction techniques, and this will be carried out in future work.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (NSFC 51905430). The first author is also supported by the Alexander von Humboldt Foundation of Germany and the Top International University Visiting Program for Outstanding Young Scholars of Northwestern Polytechnical University.

Appendix: Gradients of Learning Function

Given the definition of the learning function in Eq. (13), its gradient with respect to each x_i can be derived as:

$$\frac{\partial h(\boldsymbol{x})}{\partial x_i} = \frac{\partial \Pi'\left[\kappa(\boldsymbol{x},\boldsymbol{x}')\pi(\boldsymbol{x})\right]}{\partial x_i} - \frac{\partial\left\{\boldsymbol{\kappa}(\boldsymbol{x},X)^{\top}\pi(\boldsymbol{x})\right\}}{\partial x_i} K^{-1}\Pi'\left[\boldsymbol{\kappa}(X,\boldsymbol{x}')\right] \tag{A1}$$

where

$$\frac{\partial \Pi'\left[\kappa(\boldsymbol{x},\boldsymbol{x}')\pi(\boldsymbol{x})\right]}{\partial x_i} = -\frac{\sigma_i^2+2}{\sigma_i^2+1} x_i \Pi'\left[\kappa(\boldsymbol{x},\boldsymbol{x}')\right]\pi(\boldsymbol{x}) \tag{A2}$$

, and

$$\frac{\partial\left\{\boldsymbol{\kappa}(\boldsymbol{x},X)^\top \pi(\boldsymbol{x})\right\}}{\partial x_i} = -\left(\frac{x_i - X_{,i}}{\sigma_i^2} + x_i\right)^\top \times \boldsymbol{\kappa}(\boldsymbol{x},X)^\top \pi(\boldsymbol{x}) \tag{A3}$$

with $X_{,i}$ being the i-th column of X and $\times$ means outer product.

■

Reference:

[1]. Cockayne, J., Oates, C. J., Sullivan, T. J., & Girolami, M. (2019). Bayesian probabilistic numerical methods. *SIAM Review*, *61*(4), 756-789.

[2]. Gerstner, T., & Griebel, M. (1998). Numerical integration using sparse grids. Numerical algorithms, 18(3-4), 209.

[3]. Rubinstein, R.Y. & Kroese, D. P. Simulation and the Monte Carlo methods. Third edition. New Jersey: John Wiley & Sons, 2017

[4]. Park, J. S. (1994). Optimal Latin-hypercube designs for computer experiments. Journal of statistical planning and inference, 39(1), 95-111.

[5]. Tuffin, B. (1996). On the use of low discrepancy sequences in Monte Carlo methods. Monte Carlo Methods and Applications, 2(4), 295-320.

[6]. Au, S. K., & Beck, J. L. (2001). Estimation of small failure probabilities in high dimensions by subset simulation. Probabilistic engineering mechanics, 16(4), 263-277.

[7]. Schuëller, G. I., Pradlwarter, H. J., & Koutsourelakis, P. S. (2004). A critical appraisal of reliability estimation procedures for high dimensions. Probabilistic engineering mechanics, 19(4), 463-474.

[8]. Li, W., Chen, R., & Tan, Z. (2016). Efficient sequential Monte Carlo with multiple proposals and control variates. Journal of the American Statistical Association, 111(513), 298-313.

[9]. Oates, C. J., Girolami, M., & Chopin, N. (2017). Control functionals for Monte Carlo integration. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 79(3), 695-718.

[10]. Hennig P, Osborne MA, Girolami M. Probabilistic numerics and uncertainty in computations. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences. 2015 Jul 8;471(2179):20150142.

[11]. Briol, F. X., Oates, C. J., Girolami, M., Osborne, M. A., & Sejdinovic, D. (2019). Probabilistic integration: A role in statistical computation?. Statistical Science, 34(1), 1-22.

[12]. Owen, A. B. (2019). Comment: Unreasonable Effectiveness of Monte Carlo. Statistical Science, 34(1), 29-33.

[13]. Jagadeeswaran, R., & Hickernell, F. J. (2019). Fast automatic Bayesian cubature using lattice sampling. Statistics and Computing, 29(6), 1215-1229.

[14]. Karvonen, T., Oates, C. J., & Sarkka, S. (2018). A Bayes-Sard cubature method. Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

[15]. O'Hagan, A. (1991). Bayes–hermite quadrature. Journal of statistical planning and inference, 29(3), 245-260.

[16]. Rasmussen, C. E., & Ghahramani, Z. (2003). Bayesian Monte Carlo. Conference on neural information processing systems (NeurIPS 2002), 505-512.

[17]. Zhou, Y., Lu, Z., Cheng, K., & Yun, W. (2019). A Bayesian Monte Carlo-based method for efficient computation of global sensitivity indices. Mechanical Systems and Signal Processing, 117, 498-516.

[18]. Cadini, F., & Gioletta, A. (2016). A Bayesian Monte Carlo-based algorithm for the estimation of small failure probabilities of systems affected by uncertainties. Reliability Engineering & System Safety, 153, 15-27.

[19]. Echard, B., Gayton, N., & Lemaire, M. (2011). AK-MCS: an active learning reliability method combining Kriging and Monte Carlo simulation. Structural Safety, 33(2), 145-154.

[20]. Wei, P., Tang, C., & Yang, Y. (2019). Structural reliability and reliability sensitivity analysis of extremely rare failure events by combining sampling and surrogate model methods. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, 233(6): 943-957.

[21]. Rusmassen, C. E., & Williams, C. K. I. (2005). Gaussian process for machine learning. Cambridge: The MIT Press

[22]. Lebrun, R., & Dutfoy, A. (2009). Do Rosenblatt and Nataf isoprobabilistic transformations really differ?. Probabilistic Engineering Mechanics, 24(4), 577-584.

[23]. Wei, P., Lu, Z., & Song, J. (2013). A new variance-based global sensitivity analysis technique. Computer Physics Communications, 184(11), 2540-2551.

[24]. Valdebenito, M. A., Jensen, H. A., Hernandez, H. B., & Mehrez, L. (2018). Sensitivity estimation of failure probability applying line sampling. Reliability Engineering & System Safety, 171, 99-111.

[25]. Dubourg, V., Sudret, B., & Deheeger, F. (2013). Metamodel-based importance sampling for structural reliability analysis. Probabilistic Engineering Mechanics, 33, 47-57.