Active learning for system reliability analysis using PC-Kriging, subset simulation and sensitivity analysis

P. Parisi, M. Moustapha, S. Marelli and B. Sudret

Chair of Risk, Safety and Uncertainty Quantification
Stefano-Franscini-Platz 5
CH-8093 Zürich

Data Sheet

Journal:	Proc. 8th International Symposium on Reliability Engineering and Risk Management (ISRERM), Hannover (Germany), September 4-7
Report Ref.:	RSUQ-2022-008
Arxiv Ref.:	
DOI:	-
Date submitted:	December 16, 2021
Date accepted:	September 5, 2022

Active learning for system reliability analysis using PC-Kriging, subset simulation and sensitivity analysis

P. Parisi[1], M. Moustapha*[1], S. Marelli†[1], and B. Sudret‡[1]

[1] *Chair of Risk, Safety and Uncertainty Quantification, ETH Zürich, Switzerland*

September 23, 2022

Abstract

Structural reliability analysis aims at assessing the safety of structures which often operate under uncertain conditions. Approximation-, simulation- or surrogate-based methods are used in this context to estimate the failure probability. Surrogate-based methods are the least computationally intensive and consist in building a cheaper proxy of the original limit-state function, which is calibrated using a limited set of samples known as the experimental design. The latter is sequentially enriched to increase the accuracy of the surrogate in areas of interest, hence allowing for an accurate estimation of the failure probability.

A large number of such techniques has been recently developed in the literature (*e.g.*, active Kriging - Monte Carlo simulation). However most of these techniques consider a single limit-state function. These methods lose efficiency when used to solve *system reliability* problems, where failure is defined by a non-trivial combination of multiple limit-states. This is due to some peculiarities of the system problem such as the presence of disjoint failure domains or the uneven contribution of each limit-state to the overall failure.

In this work, we propose an efficient algorithm combining Kriging/PC-Kriging and subset simulation to solve system reliability problems in their most general setting, i.e., with an arbitrary combination of components. We devise a new learning function which first identifies candidate samples and then selects for enrichment the specific limit-state that contributes the most to system failure. The algorithm is validated on a set of analytical functions and compared with existing methods in the literature.

1 Introduction

Civil and engineering structures often operate in an environment not entirely known throughout their life cycle. The level of safety of the structure may be assessed through structural reliability

*moustapha@ibk.baug.ethz.ch

†marelli@ibk.baug.ethz.ch

‡sudret@ethz.ch

analysis. Generally, this is carried out in a probabilistic setting using a set of random variables $\boldsymbol{X} \sim f_{\boldsymbol{X}}$ that describe the state of the system and a set of so-called limit-state functions $\{g_j(\boldsymbol{x}), j = 1, \dots, p\}$. Ultimately, the goal is to estimate the probability of failure of the structure under these conditions. This is more specifically defined by:

$$P_f = \int_{\mathcal{D}_f} f_{\boldsymbol{X}}(\boldsymbol{x})\, d\boldsymbol{x}, \tag{1}$$

where $\mathcal{D}_f$ denotes the failure domain. In component reliability, *i.e.*, when $p = 1$, failure is conventionally defined by $g(\boldsymbol{x}) = g_1(\boldsymbol{x}) \leq 0$. The definition of the failure domain is more complex for system reliability problems, *i.e.*, when $p > 1$. Such systems may be described as an assembly of components in a series, parallel or general configuration [1, 2]. A series system is one that fails when any of its components does. In contrast, a parallel system is a more robust one that fails only when all the components do. The failure domain for these systems are defined respectively by $\mathcal{D}_f^{\text{series}} = \{\boldsymbol{x} \in \mathbb{X} : \min(g_1(\boldsymbol{x}), \dots, g_p(\boldsymbol{x})) \leq 0\}$ and $\mathcal{D}_f^{\text{parallel}} = \{\boldsymbol{x} \in \mathbb{X} : \max(g_1(\boldsymbol{x}), \dots, g_p(\boldsymbol{x})) \leq 0\}$. In this work, we consider a more generic configuration where a composition of Boolean operators (*e.g.*, a combination of max and min operators) defines the limit-state function. Denoting this function by h, the system limit-state function can then be cast as

$$h(g_1(\boldsymbol{x}), \dots, g_p(\boldsymbol{x})) = h(g(\boldsymbol{x})), \tag{2}$$

where h is assumed to take negative values in the failure domain.

The solution of structural reliability problems, which entails solving an integration problem over the implicitly defined failure domain (Eq. 1), generally involves computationally intensive methods based on simulation techniques. Examples include crude Monte Carlo simulation or more advanced variance reduction techniques such as importance sampling, subset simulation or line sampling [3]. These methods however require repetitive calls to the limit-state function, in the order of tens to hundreds of thousands, which makes them unaffordable when the limit-state function arises from an expensive-to-evaluate computational model.

An efficient solution to these computational drawbacks has been introduced by the use of *surrogate models* in a so-called active learning reliability framework [4]. The idea is to build a surrogate of the limit-state function to be used when locating the failure domain. This surrogate model is built by learning over a small set of limit-state function evaluations known as experimental design. Its efficiency comes from the fact that the limit-state function is evaluated sparingly and only in regions of interest, in a sequential scheme. This allows for a very precise identification of the limit-state surface and henceforth of the failure domain.

Milestone contributions in this field include the early work of [5, 6], which respectively introduced the so-called efficient global reliability analysis (EGRA) and active Kriging - Monte Carlo simulation (AK-MCS) that both rely on widely-used Gaussian process models as surrogates. Since then, multiple active learning reliability strategies have been introduced to efficiently solve Eq. (1), as evidenced by the recent reviews in [7] and [4]. Most of these contributions were however only introduced for component reliability problems and are not optimally efficient for system reliability problems. Indeed the latter is more complex due to the likely presence of multiple failure domains and the uneven contribution of the different limit states to failure. This calls for special algorithms capable of handling such peculiarities.

A few attempts have been made in that direction. The most direct adaptation of component active learning reliability algorithms to systems is derived by directly considering $h\left(g\left(\boldsymbol{x}\right)\right)$ as the function to approximate. The main limitation of such an approach is that h is by definition non-smooth and may present discontinuities, making it difficult to build an accurate surrogate model. Specific methods accounting directly for multiple limit-states have been introduced as well. [8, 9] proposed a composite learning function well suited for parallel and series system problems. More recently, [10] proposed the so-called ALK-TCR-SYS method which includes a scheme to select the most pertinent candidate sample for enrichment, leveraging on the knowledge of the system configuration (*i.e.*, parallel or series). Similarly, [11] analytically derived the so-called system expected risk function (SERF) which is embedded in a reliability-based design optimization framework.

All these approaches are limited to simple series or parallel configurations and only consider the sole adaptation of the learning function, ignoring other important components such as the surrogate model or the reliability estimation algorithm, which are equally important to devise an optimal solving strategy [4]. The latter is however important as it ultimately defines the accuracy of the estimated failure probability, regardless of that of the surrogate model.

In this paper, we propose an approach to efficiently solve the system reliability problem by building a single surrogate for each component and only enriching the ones that are the most relevant in a given iteration of the algorithm. We make use of both subset simulation [12] and Sobol' sensitivity [13] analysis to identify the enrichment candidates and the relevant limit-states to be evaluated. The proposed method is illustrated on two mathematical problems and its efficiency is assessed against similar methods found in the literature.

2 Proposed method

The proposed method follows the general framework for active learning reliability described in [4], with a few modifications aimed at enhancing efficiency with respect to system reliability. The flowchart in Figure 1 summarizes the main steps, which are the following:

1. **Initialization:** For each limit-state $g_j, j = 1 \dots, p$, an experimental design $\mathcal{D}_j^{(0)}$ is built by i. first sampling $m_j^{(0)}$ points in the input domain using space-filling techniques, such as Latin hypercube sampling, and ii. evaluating them using the corresponding limit-state function. The experimental design therefore reads $\mathcal{D}_j^{(0)} = \left\{ \left(\boldsymbol{\mathcal{X}}_j^{(i)}, \mathcal{Y}_j^{(i)} \right) : \mathcal{Y}_j = g_j \left(\boldsymbol{\mathcal{X}}^{(i)} \right) \in \mathbb{R}, \boldsymbol{\mathcal{X}}^{(i)} \in \mathbb{X} \subset \mathbb{R}^M, \right.$ $\left. i = 1 \dots m_j^{(0)} \right\}$. The initial number of samples $m_j^{(0)}$ is typically chosen small, *i.e.*, in the orders of tens of samples. We further assume in this work that they are the same for all

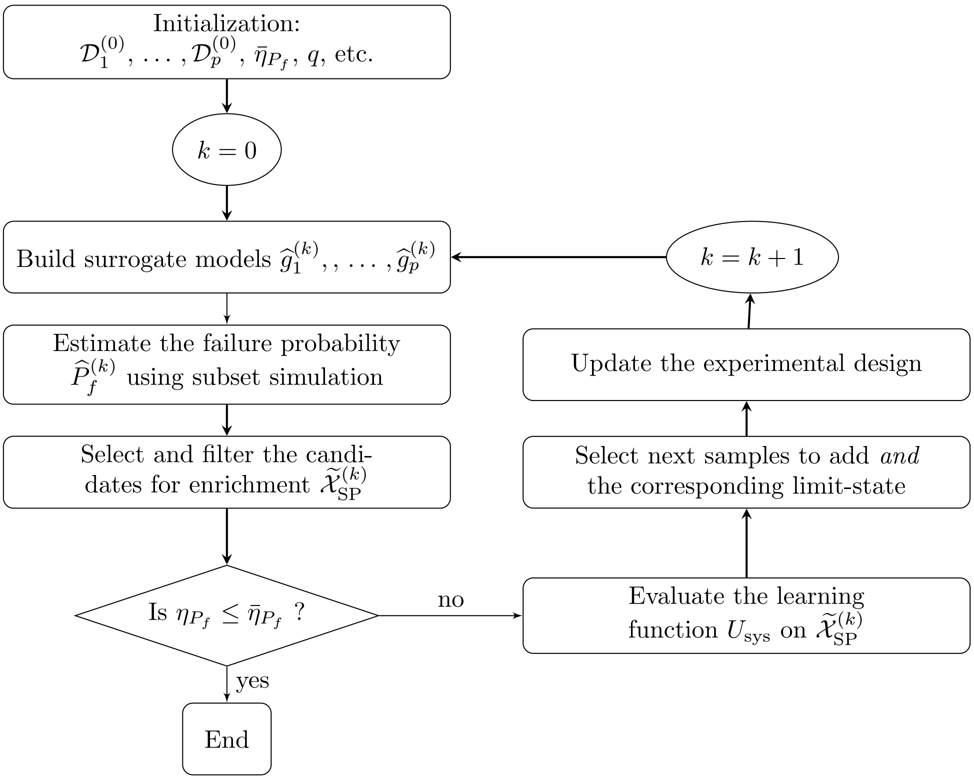
Initialization:
$\mathcal{D}_1^{(0)}, \dots, \mathcal{D}_p^{(0)}$, $\bar{\eta}_{P_f}$, q, etc.
$k = 0$
Build surrogate models $\widehat{g}_1^{(k)}, , \dots, \widehat{g}_p^{(k)}$
$k = k + 1$
Estimate the failure probability $\widehat{P}_f^{(k)}$ using subset simulation
Update the experimental design
Select and filter the candidates for enrichment $\widetilde{\mathcal{X}}_{\text{SP}}^{(k)}$
Select next samples to add *and* the corresponding limit-state
Is $\eta_{P_f} \leq \bar{\eta}_{P_f}$?
no
Evaluate the learning function U_{sys} on $\widetilde{\mathcal{X}}_{\text{SP}}^{(k)}$
yes
End

Figure 1: Flowchart of the proposed system active learning reliability algorithm.

limit-states, *i.e.*, $m_1^{(0)} = \dots = m_p^{(0)}$. Following this initialization, the algorithm enters a loop where the construction of a surrogate model, its enrichment and reliability analysis are alternatively performed.

2. **Surrogate:** At iteration k of the algorithm, a surrogate model $\widehat{g}_j^{(k)}$, herein Kriging or PC-Kriging, is built for each limit-state using the current experimental design. Details on the construction of the surrogate are given in Section 3.

3. **Reliability analysis:** The failure probability $\widehat{P}_f^{(k)}$ is then estimated using the current surrogate model. In this work, we consider subset simulation in a so-called *overkill setup*, which simply consists in choosing parameter settings that allow to decrease the variance of the estimated failure probability. As shown in [4], such a setting, which can be obtained in subset simulation by increasing the batch sample size and the target conditional failure probability, is extremely efficient compared to a traditional setting. More details regarding this setting are given in Section 4.

4. **Convergence:** Once the failure probability is estimated, the convergence of the algorithm is assessed. In this work, we simply check the stability of the estimated failure probability over two consecutive iterations. Convergence is assumed when the relative variation between iterations $\left|\widehat{P}_f^{(k)} - \widehat{P}_f^{(k-1)}\right| / \widehat{P}_f^{(k)}$ is below a given threshold $\bar{\eta}_{p_f}$.

5. **Enrichment:** If convergence is not achieved, the experimental design is enriched to improve

the local accuracy of the critical limit-states. The bulk of the developments of the proposed strategy lies in this step, as we aim not only to find the next best candidate sample as in traditional component reliability analysis, but also to select the most appropriate limit-states for enrichment. The goal is to be as efficient as possible by evaluating only the limit-states that contribute to failure in a given input space region. Details of the proposed approach are given in Section 5.

Once the experimental design is updated, the algorithm returns to Step 2 and iterates until convergence. In the next sections, we give more details regarding the important components of the framework.

3 Surrogate models

The proposed algorithm is built using a modular framework where any surrogate model can be employed. The learning function proposed in this work however relies on the surrogate prediction error. We therefore consider in this paper Kriging and polynomial chaos - Kriging (PC-Kriging), which are briefly described in the sequel.

Kriging: Kriging, also known as Gaussian process modelling, is a popular surrogate technique where the underlying function to approximate is assumed to be the realization of a stochastic process [14]:

$$\widehat{\mathcal{M}}^{\text{KRG}}(\mathbf{x}) = \boldsymbol{\beta}^T \mathbf{f}(\mathbf{x}) + \sigma^2 Z(\mathbf{x}, \omega). \tag{3}$$

The first part of this equation represents the mean of the process and is known as its trend. In its most common form, it may be written using polynomials, where $\boldsymbol{f} = \{f_j \ : \ j = 1, \dots, P\}$ are basis functions and $\boldsymbol{\beta} = \{\beta_j \ : \ j = 1, \dots, P\}$ are the corresponding coefficients. The second summand is the actual stochastic process with variance σ^2 and completely defined by the zero-mean unit-variance stationary Gaussian process (GP) $Z(\mathbf{x}, \omega)$ with auto-correlation function $R(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta})$.

The surrogate model is derived by fitting the parameters of the GP by learning over a set of observations of the model, known as experimental design. Following this learning stage, predictions can be made for any new sample, which is assumed to follow a multivariate Gaussian distribution:

$$\begin{Bmatrix} \widehat{Y}(\mathbf{x}) \\ \mathcal{Y} \end{Bmatrix} \sim \mathcal{N}_{N+1} \left(\begin{Bmatrix} \mathbf{f}(\mathbf{x})^T \boldsymbol{\beta} \\ \mathbf{F}\boldsymbol{\beta} \end{Bmatrix}, \ \sigma^2 \begin{Bmatrix} 1 & \mathbf{r}^T(\mathbf{x}) \\ \mathbf{r}(\mathbf{x}) & \mathbf{R} \end{Bmatrix} \right), \tag{4}$$

where $\mathbf{F}$ is the observation matrix with $F_{i,j} = f_j(\mathbf{x}^{(i)});\ i = 1, \dots, N;\ j = 1, \dots, P$, $\mathbf{r}(\mathbf{x})$ is the vector of the cross-correlations with $r_i = R(\mathbf{x}, \mathbf{x}^{(i)}; \boldsymbol{\theta}),\ i = 1, \dots, N$ and $\mathbf{R}$ is the correlation matrix with elements $R_{i,j} = R(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\theta}),\ i, j = 1, \dots, N$. The mean and the variance of this Gaussian random variable $\widehat{Y}(\mathbf{x})$ are given by:

$$\mu_{\widehat{Y}(\mathbf{x})} = \mathbf{f}^T(\mathbf{x})\hat{\boldsymbol{\beta}} + \mathbf{r}^T(\mathbf{x})\mathbf{R}^{-1}(\mathcal{Y} - \mathbf{F}\hat{\boldsymbol{\beta}}), \tag{5}$$

$$\sigma^2_{\widehat{Y}(\mathbf{x})} = \widehat{\sigma}^2\Big(1 - \mathbf{r}^T(\mathbf{x})\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}) + \mathbf{u}^T(\mathbf{x})(\mathbf{F}^T\mathbf{R}^{-1}\mathbf{F})^{-1}\mathbf{u}(\mathbf{x})\Big), \tag{6}$$

where $\hat{\boldsymbol{\beta}} = (\mathbf{F}^T\mathbf{R}^{-1}\mathbf{F})^{-1}\mathbf{F}^T\mathbf{R}^{-1}\mathcal{Y}$ is the weighted least-square estimate of the regression coefficients and $\mathbf{u}(\mathbf{x}) = \mathbf{F}^T\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}) - \mathbf{f}(\mathbf{x})$ is introduced for convenience.

The learning step consists in first selecting the correlation function, which is assumed to be an anistropic Gaussian kernel in this work, *i.e.*,

$$R(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{\theta}) = \prod_{i=1}^{M} R_i(x_i, x_i'; \theta_i) = \prod_{i=1}^{M} \exp\left(-\frac{1}{2}\left(\frac{x_i - x_i'}{\theta_i}\right)^2\right). \tag{7}$$

The optimal hyperparameters of the model $\left\{\widehat{\boldsymbol{\beta}}, \widehat{\sigma}, \boldsymbol{\theta}\right\}$ are obtained here by maximum likelihood estimation [15].

PC-Kriging PC-Kriging is an extension of universal Kriging developed in [16, 17], where the deterministic trend is chosen as polynomial chaos expansions (PCE), *i.e.*,

$$\widehat{\mathcal{M}}^{\text{PCK}}(\mathbf{x}) = \sum_{\boldsymbol{\alpha}\in\mathcal{A}} y_{\boldsymbol{\alpha}} \Psi_{\boldsymbol{\alpha}}(\mathbf{x}) + \sigma^2 Z(\mathbf{x}, \omega), \tag{8}$$

where $\sum_{\boldsymbol{\alpha}\in\mathcal{A}} y_{\boldsymbol{\alpha}} \Psi_{\boldsymbol{\alpha}}(\mathbf{x})$ is a truncated series of weighted orthonormal polynomials representing the GP trend. In this trend, $y_{\boldsymbol{\alpha}}$ are coefficients to be calibrated, $\mathcal{A}$ is a truncation set and $\Psi_{\boldsymbol{\alpha}} = \prod_{i=1}^{M} \Psi_{\alpha_i}(x_i)$ are orthonormal polynomials whose components Ψ_{α_i} are chosen in accordance with the probability distribution f_{X_i} of X_i.

The calibration of this surrogate model is a two-stage process where on the one hand, the coefficients of the trend are identified and on the other hand, the optimal Kriging hyperparameters are estimated. [16] proposed two approaches to achieve this. In this work, we consider the so-called optimal PC-Kriging approach where the surrogate model is built in an iterative scheme with the Kriging and PCE parts being jointly calibrated.

4 Subset simulation

Subset simulation is a variance-reduction reliability estimation algorithm [12]. It is particularly adapted to problems with small failure probability where the failure domain is in remote areas of the random space. The underlying principle is to decompose the failure event into multiple events whose probability of occurrence is magnitudes higher.

More specifically, let us consider a set of nested domains $\mathcal{D}_1 \supset \mathcal{D}_2 \supset \cdots \supset \mathcal{D}_m = \mathcal{D}_f$ such that $\mathcal{D}_f = \cap_{k=1}^{m} \mathcal{D}_k$. The intermediate domains are defined using a sequence of decreasing thresholds $t_1 > t_2 > \cdots > t_m = 0$ such that $\mathcal{D}_k = \{\mathbf{x} : g(\mathbf{x}) \leq t_k\}$. It follows that the failure probability can be recast as:

$$P_f = \mathbb{P}\left(\mathcal{D}_f\right) = \mathbb{P}\left(\cap_{k=1}^{m} \mathcal{D}_k\right) = \mathbb{P}\left(\mathcal{D}_1\right) \prod_{i=1}^{m-1} \mathbb{P}\left(\mathcal{D}_{i+1} | \mathcal{D}_i\right) \tag{9}$$

The initial failure probability $\mathbb{P}(\mathcal{D}_1)$ is estimated using Monte Carlo simulation and the remaining conditional failure probabilities are computed using a dedicated Markov Chain Monte Carlo (MCMC) simulation. The intermediate thresholds are set on-the-fly such that the $\mathbb{P}(\mathcal{D}_{i+1}|\mathcal{D}_i) = p_0$ is large enough. While in usual settings p_0 is chosen to be 0.1, we consider here $p_0 = 0.25$. The sample size for the initial Monte Carlo simulation and the intermediate MCMC steps is set to 10^5. These settings correspond to the aforementioned *overkill setup* [4]. This essentially allows us to reduce the variance of the failure probability estimate and, at the same time, enhance the robustness of the active learning scheme.

5 Enrichment strategy for system reliability analysis

The core of the novel strategy we propose lies in this step of the algorithm. In order to efficiently allocate computational resources, we resort to a four-step enrichment procedure, which is described in the following paragraphs.

5.1 Search space

In the first step of the enrichment we define the search space, *i.e.*, the set of samples that will eventually be considered as candidates for enrichment. We consider here all the samples generated during the reliability analysis in the previous iteration. As previously mentioned, subset simulation is used in this work in an overkill setup. The samples generated in this step are therefore the starting point for enrichment and are denoted by $\mathcal{X}_{\mathrm{SP}}$.

5.2 Learning function

A learning function is used to determine which points from $\mathcal{X}_{\mathrm{SP}}$ are more likely to locally improve the limit-state function surrogates and henceforth the accuracy of the estimated failure probability. Many such functions have been proposed in the context of component reliability analysis. They are built with the aim of a balanced exploitation and exploration of the input space. Exploration in this context means looking for new samples far away from existing ones in order to increase the overall knowledge of the limit-state function landscape, while exploitation tends to focus on those samples that are the closest to the currently estimated limit-state surfaces.

The starting point for the learning function we propose is the so-called *deviation number* proposed by [6] for component reliability, which reads

$$U(\boldsymbol{x}) = \frac{\left|\mu_{\widehat{g}}(\boldsymbol{x})\right|}{\sigma_{\widehat{g}}(\boldsymbol{x})}, \tag{10}$$

where $\mu_{\widehat{g}}$ and $\sigma^2_{\widehat{g}}$ are respectively the Kriging or PC-Kriging mean prediction and variance.

To adapt this learning function, we first start by noting that using a Kriging or PC-Kriging model to emulate separately each limit-state function, we obtain that for each sample $\boldsymbol{x} \in \mathcal{X}_{\text{SP}}$,

$$Z_j(\boldsymbol{x}) = \widehat{g}_j(\boldsymbol{x}) \sim \mathcal{N}\left(\mu_{\widehat{g}_j}(\boldsymbol{x}), \sigma^2_{\widehat{g}_j}(\boldsymbol{x})\right), \tag{11}$$

while the prediction for the system limit-state is given by

$$Z_{\text{sys}} = h\left(Z_1(\boldsymbol{x}), \ldots, Z_p(\boldsymbol{x})\right). \tag{12}$$

We then propose an adaptation of the U learning function which reads as follows:

$$U_{\text{sys}} = \frac{|\mu_{\text{sys}}(\boldsymbol{x})|}{\sigma_{\text{sys}}(\boldsymbol{x})}, \tag{13}$$

where $\mu_{\text{sys}}(\boldsymbol{x})$ and $\sigma_{\text{sys}}(\boldsymbol{x})$ are respectively the mean and standard deviation of the limit-state approximation $Z_{\text{sys}} = h\left(\widehat{g}(\boldsymbol{x})\right)$ defined in Eq. (12).

Since the distribution of Z_{sys} is generally not known, we estimate its moments empirically. It follows from Eq. 11 and from the mutual independence of the component-wise surrogate model predictions that the joint probability distribution of $\boldsymbol{Z}(\boldsymbol{x})$ is an independent multivariate Gaussian, *i.e.*,

$$\begin{aligned} \boldsymbol{Z}(\boldsymbol{x}) \sim & f_{\boldsymbol{Z}}\left(Z_1(\boldsymbol{x}), \ldots, Z_p(\boldsymbol{x})\right) = \\ & \prod_{j=1}^{p} f_Z\left(Z_j(\boldsymbol{x})\right) = \prod_{j=1}^{p} \mathcal{N}\left(\mu_{\widehat{g}_j}(\boldsymbol{x}), \sigma^2_{\widehat{g}_j}(\boldsymbol{x})\right). \end{aligned} \tag{14}$$

For each sample $\boldsymbol{x} \in \mathcal{X}_{\text{SP}}$, we can therefore draw N_s samples $\left(\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(p)}\right)$ from $f_{\boldsymbol{Z}}$ and empirically estimate the mean and variance as follows:

$$\begin{aligned} \widehat{\mu}_{\text{sys}}(\boldsymbol{x}) &= \frac{1}{N_s} \sum_{i=1}^{N_s} h\left(\boldsymbol{z}^{(i)}\right), \\ \widehat{\sigma}_{\text{sys}}(\boldsymbol{x}) &= \frac{1}{N_s - 1} \sum_{i=1}^{N_s} \left(h\left(\boldsymbol{z}^{(i)}\right) - \widehat{\mu}_{\text{sys}}(\boldsymbol{x})\right)^2. \end{aligned} \tag{15}$$

An estimate of the system learning function is then obtained by using these quantities, *i.e.*,

$$\widehat{U}_{\text{sys}}(\boldsymbol{x}) = \frac{|\widehat{\mu}_{\text{sys}}(\boldsymbol{x})|}{\widehat{\sigma}_{\text{sys}}(\boldsymbol{x})}. \tag{16}$$

The next best point to add is eventually selected as the minimizer of Eq. 16.

5.3 Filtering and clustering

Instead of adding a single point per enrichment iteration, we rather consider adding multiple ones at the same time. This helps accelerate convergence by simultaneously targeting multiple failure regions. To achieve this, we resort to clustering techniques.

Prior to clustering, we filter the candidate samples in $\mathcal{X}_{\text{SP}}$ by keeping only the samples that are most likely to improve the limit-state surface as defined by the learning function. This subset is defined by

$$\widetilde{\mathcal{X}}_{\text{SP}} = \left\{ \boldsymbol{x} \in \mathcal{X}_{\text{SP}} : \widehat{U}_{\text{sys}}\left(\boldsymbol{x}\right) < u_q \right\}, \tag{17}$$

where u_q is a lower q-quantile of $\widehat{U}_{\text{sys}}$ as computed on $\mathcal{X}_{\text{SP}}$. By choosing a small value of q, say in the order of $1\% - 5\%$, we can discard samples that are not relevant for enrichment.

This also makes the clustering step more efficient as only samples associated to relevant limit-states remain. In this work, we use DBSCAN [18] which is a density-based clustering technique. Its main advantage is that it does not require specifying a fixed number of clusters a priori. Instead, it learns from the data and returns the most optimal number of clusters together with their configuration. In practice, it often turns out that each cluster corresponds to one specific limit-state.

For each cluster we select for enrichment the point that minimizes the learning function, provided that its value is larger than a threshold set equal to 2. The latter condition allows us to ignore limit-states that are already accurately approximated. The set of enrichment samples selected as this stage is denoted by $\mathcal{X}_{\text{En}}$.

5.4 Sensitivity analysis

The final step of enrichment consists in defining which limit-state to evaluate for each sample in $\mathcal{X}_{\text{En}}$. We therefore need to define which component limit-state affects the most the system for each of these points. We resort to Sobol' sensitivity analysis [13] to achieve this. The model of interest in this case is the composition function h whose random inputs are jointly Gaussian. More specifically, as shown earlier, for each sample $\boldsymbol{x} \in \mathcal{X}_{\text{En}}$, the surrogate predictions, $\boldsymbol{Z}\left(\boldsymbol{x}\right) = \left\{Z_1\left(\boldsymbol{x}\right), \dots, Z_p\left(\boldsymbol{x}\right)\right\}$ follow a normal distribution. Using Sobol' indices, we can therefore estimate the shares of the variance of $h\left(\boldsymbol{Z}\left(\boldsymbol{x}\right)\right)$ associated to each component $\{1, \dots, p\}$. This quantity can be summarized into the so-called total Sobol' indices $\left\{S_1^T, \dots, S_p^T\right\}$. The specific limit-state to evaluate is eventually obtained by selecting the one corresponding to the largest total Sobol' index.

It is important to note at this stage that this step is not very computationally intensive, since the composition function h is known analytically and it is straightforward to sample from $\boldsymbol{Z} \sim f_{\boldsymbol{Z}}$. No evaluation of the limit-state is carried out in this step. Furthermore, only a few samples per iteration are selected for the computation of the Sobol' indices, which are obtained by Monte Carlo simulation.

6 Application examples

In this section, we validate the proposed method using two different mathematical functions. Convergence is assumed when the relative error introduced in Section 2 is below $\bar{\eta}_{P_f} = 0.01$ for two consecutive iterations. To assess the accuracy of the algorithm, we evaluate the following relative errors:

$$\begin{aligned} \Delta_{P_f} &= \frac{|P_f - P_{f,\text{ref}}|}{P_{f,\text{ref}}}, \\ \Delta_\beta &= \frac{|\beta - \beta_{\text{ref}}|}{\beta_{\text{ref}}}, \end{aligned} \tag{18}$$

where P_f (resp. β) is the estimated failure probability (resp. reliability index) and $P_{f,\text{ref}}$ (resp. β_{ref}) is the reference failure probability (resp. reliability index) obtained using the original model and a large Monte Carlo sample set.

To assess the robustness of the algorithm, the analysis is repeated 20 times. The median results are eventually used for a benchmark with existing methods in the literature. The analysis is carried out using UQLab [19], a modular framework for uncertainty quantification.

6.1 Four-branch function

The four-branch function is a popular two-dimensional function used to illustrate active learning reliability analysis methods. It can be seen as a series system with four components $g_1, \dots g_4$ defined as follows:

$$\begin{aligned} g_1(\boldsymbol{X}) &= 3 + 0.1(X_1 - X_2)^2 - \frac{1}{\sqrt{2}}(X_1 + X_2), \\ g_2(\boldsymbol{X}) &= 3 + 0.1(X_1 - X_2)^2 + \frac{1}{\sqrt{2}}(X_1 + X_2), \\ g_3(\boldsymbol{X}, \boldsymbol{a}) &= (X_1 - X_2) + \frac{a}{\sqrt{2}}, \\ g_4(\boldsymbol{X}, \boldsymbol{a}) &= (X_2 - X_1) + \frac{a}{\sqrt{2}}, \end{aligned} \tag{19}$$

where a is a constant set equal to 6 or 7 in the numerical examples.

The associated inputs follow a bivariate independent standard Gaussian distribution, *i.e.*,

$$\boldsymbol{X} = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right). \tag{20}$$

Running the algorithm 20 times, we obtain multiple estimates of the failure probability, which are slightly dependent on the initial experimental design. Figure 2 summarizes these estimates in boxplots for both Kriging and PC-Kriging. We can note that in the two cases the results are very close to the actual reference solution shown by the dotted black line. The scatter is relatively small even though there is a bias where the algorithm has the tendency to slightly underestimate the failure probability. The median relative errors are 0.83% and 1.03% respectively for Kriging and PC-Kriging. When considering the reliability index, these errors respectively become $\approx 0.01\%$ and 0.12%, respectively.

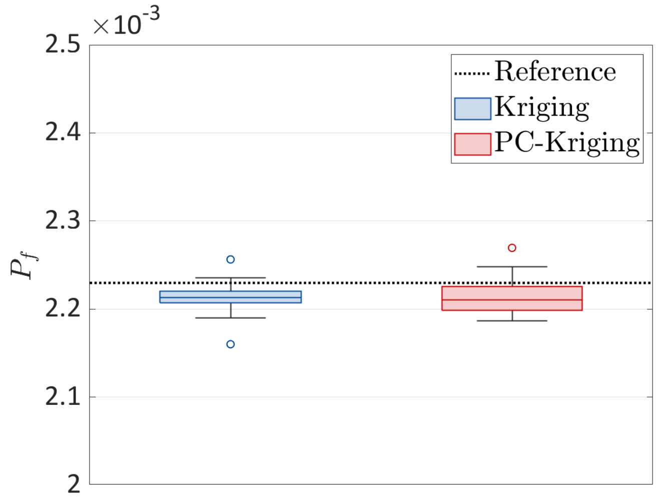

Figure 2: Example 1 - Estimated failure probabilities for 20 random repetitions of the algorithm.

The second criterion of interest to assess the proposed algorithm is its efficiency, *i.e.*, its capacity to find accurate solutions using the fewest possible calls to the original limit-state functions. Figure 3 shows the boxplots of the number of calls to each limit-state function for both Kriging and PC-Kriging. Overall, PC-Kriging is much more efficient as it is more adapted to approximate such a polynomial limit state function. Furthermore, limit-states g_3 and g_4 which are linear are well approximated by the initial experimental design and no further points are added to these limit-states during the enrichment process. Only g_1 and g_2 are enriched, with PC-Kriging converging much faster than Kriging.

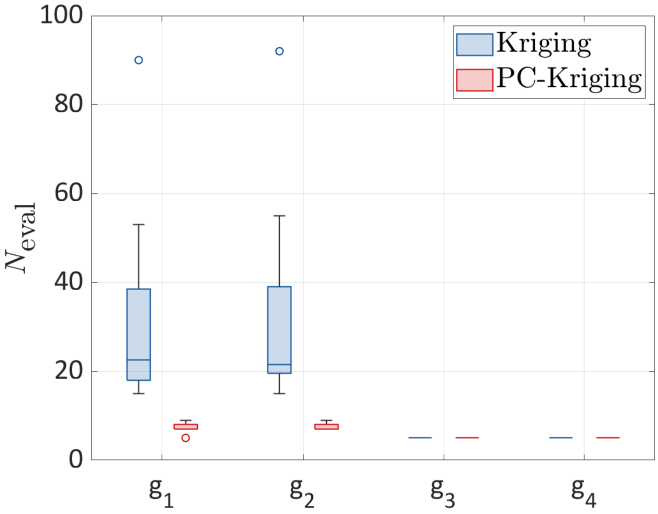

Figure 3: Example 1 - Number of added samples for each limit-state for 20 random repetitions of the algorithm.

The algorithms converge without any additional sample added to g_3 and g_4 thanks to two of its feature: i. the filtering step that allows us to ignore limit-states that are already accurate and ii. the Sobol' sensitivity analysis that allows us to target only the limit-state that is relevant for a given sample. This is illustrated in Figure 4, which corresponds to the fifth enrichment step of the solution scheme corresponding to the median results in terms of Δ_{P_f}.

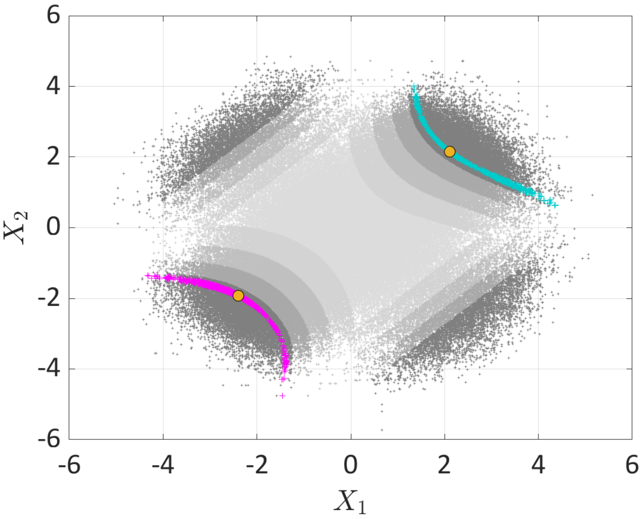

Figure 4: Example 1 - Illustration of the enrichment process.

In this figure, the candidates for enrichment $\mathcal{X}_{\text{SP}}$ are shown in gray. They have been generated by subset simulation in the previous iteration and the five shades of gray represent the intermediate steps of subset simulation. The colored samples in cyan and magenta represent the two clusters that have been identified by DBSCAN while the yellow dots show the two samples selected for enrichment. Each of the subsets actually correspond to one of the two limit-states g_1 and g_2. The limit-states, together with the initial experimental design samples, are shown in Figure 5 for reference.

Finally, we compare the proposed algorithms to various methods available in the literature. Table 1 mainly features active learning algorithms developed for component reliability and used in the context of multiple limit-state by approximating $h\left(g\left(\boldsymbol{x}\right)\right)$. Table 2 shows methods developed specifically for system reliability. In the two cases, we can see that the method we propose is the most efficient and accurate. Furthermore, the coefficient of variation of the estimated failure probability is comparatively small, thanks to the use of the overkill setup.

6.2 Liquid hydrogen tank

This example investigates the design of a liquid hydrogen fuel tank to be used on a space launch vehicle [22]. Built using a honeycomb sandwich composite material, the tank is subjected to various stresses caused by ullage pressure, head pressure, vehicle acceleration and fuel weight. There are three failure modes corresponding to von Mises stresses, isotropic strength and

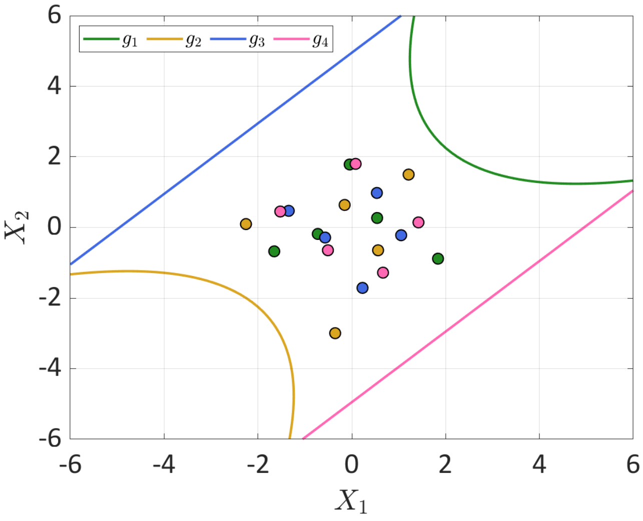

Figure 5: Example 1 - Initial experimental design and limit-state surface contours.

Table 1: Four-branch function ($a = 7$) performance comparison from [20].

Method	$\widehat{P}_f(\times 10^{-3})$	Error(%)	CoV(%)	N_{eval}
MCS	2.224	-	0.67	10^7
SuS	2.260	0.85	2.43	$28 \cdot 10^3$
IS	2.251	0.45	2.66	$14.4 \cdot 10^4$
AK-MCS	2.271	1.34	2.10	408
AK-IS	2.210	1.38	2.67	1446
AK-SS	2.228	0.31	2.51	268
Proposed (KRG)	2.223	0.82	0.55	53.5
Proposed (PCK)	2.223	1.03	0.55	25

Table 2: Four-branch function ($a = 6$) performance comparison from [21].

Method	$\widehat{P}_f(\times 10^{-3})$	Error(%)	CoV(%)	N_{eval}
MCS	4.48	-	0.47	10^7
AK-SYS	4.45	0.24	1.49	63.3
ALK-TCR	4.48	0.43	1.49	61.3
ALK-SIS	4.45	0.24	1.29	62.4
Proposed (KRG)	4.44	0.32	0.47	56
Proposed (PCK)	4.45	0.31	0.47	24

honeycomb buckling. The system fails when any one of the limit-state is reached. It is therefore a series system whose component limit-state functions read:

$$\begin{aligned} g_1(\boldsymbol{X}) &= 0.847 + 0.96x_1 + 0.986x_2 - 0.216x_3 \\ &\quad + 0.077x_1^2 + 0.11x_2^2 + 0.007x_3^2 \\ &\quad + 0.378x_1x_2 - 0.106x_1x_3 - 0.11x_2x_3, \\ g_2(\boldsymbol{X}) &= \frac{84000\ t_{\text{plate}}}{\sqrt{N_x^2 + N_y^2 - N_xN_y + 3N_{xy}^2}} - 1, \\ g_3(\boldsymbol{X}) &= \frac{8400\ t_{\text{plate}}}{|N_y|} - 1, \end{aligned} \tag{21}$$

where $x_1 = 4(t_{\text{plate}} - 0.075)$, $x_2 = 20(t_h - 1)$ and $x_3 = -6000\left(\frac{1}{N_{xy}} + 0.003\right)$. Five random variables describe the system, namely the plate thickness t_{plate}, the honeycomb thickness t_h and the running loads N_x, N_y and N_{xy}. They all follow a Gaussian distribution whose parameters are given in Table 3.

Table 3: Liquid hydrogen tank input marginal properties.

Variable	Distribution	Mean	Standard deviation
t_{plate}	Gaussian	0.07433	0.005
t_h	Gaussian	0.1	0.01
N_x	Gaussian	13	60
N_y	Gaussian	4751	48
N_{xy}	Gaussian	-684	11

The problem is solved using an initial experimental design of size 10 independently drawn for each of the limit-state functions. Figure 6 shows boxplots of the resulting failure probability estimates for 20 repetitions when using Kriging and PC-Kriging. The black dotted line corresponds to the reference solution. In the two cases, the solutions are close to the reference one and yield a median relative error which is smaller than 1%, with Kriging being slightly more accurate than PC-Kriging.

However, as shown in Figure 7, PC-Kriging is much more efficient than Kriging, as its average number of model evaluations is smaller. Furthermore, similarly to the previous example, limit-state components g_1 and g_3 are easier to approximate and require almost no additional samples after the initial step.

7 Conclusion

System reliability analysis specifically targets problems where there are multiple failure modes. One of the most efficient ways to solve such problems requires using surrogate models in a so-called active learning scheme. Such techniques were originally developed in the context of

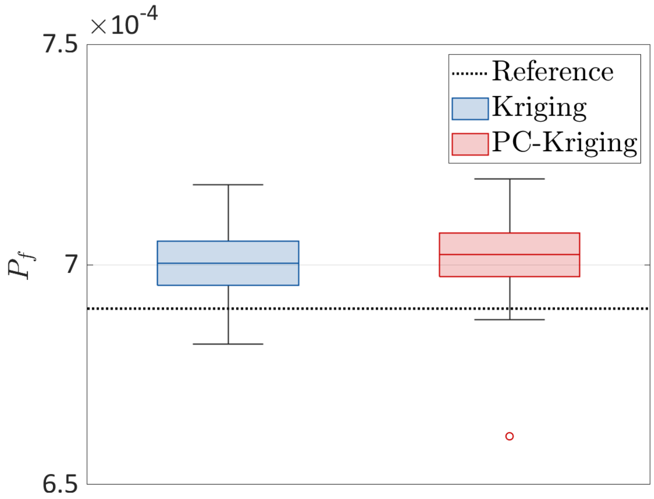

Figure 6: Example 2 - Estimated failure probabilities for 20 random repetitions of the algorithm.

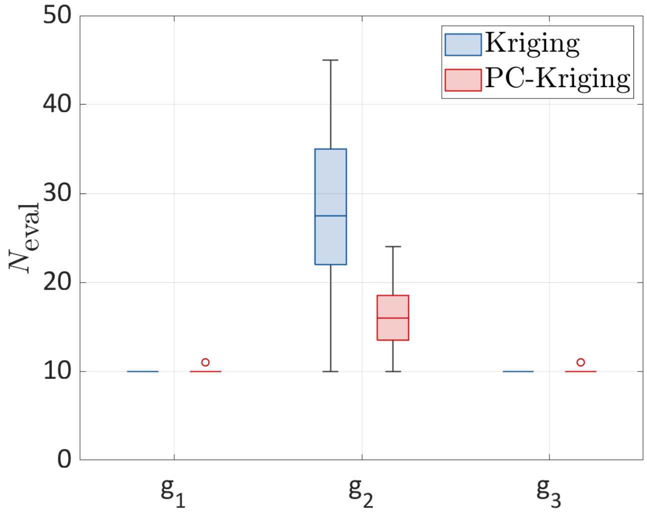

Figure 7: Example 2 - Number of added samples for each limit-state for 20 random repetitions of the algorithm.

component reliability analysis, where only one limit-state function describes the failure of the structure. The presence of multiple failure modes and the uneven contribution of each limit-state to failure make such an approach suboptimal for system reliability.

This work aimed at extending the currently used active learning scheme to efficiently solve system reliability problems, in their most general form, *i.e.*, considering any arbitrary combination of limit-states. We have specifically focused on combining Kriging and PC-Kriging with subset simulation and a novel enrichment scheme. The latter is the ingredient making the entire framework efficient as it allows us to select the samples that are most likely to improve the limit-states and to evaluate only the limit-state function that is the most impacted by the selected sample. This is achieved by using an adaptation of the popular U-learning function to system problems, a clustering technique that can identify and differentiate automatically failure regions and Sobol' sensitivity analysis to assess the impact of any sample to each limit-state.

The algorithm is validated and illustrated on two analytical examples. It is shown to be accurate, robust and efficient. Furthermore, the algorithm is shown to be more efficient than many similar techniques found in the literature.

References

[1] A. Der Kiureghian. *First- and second-order reliability methods*, chapter 14, pages 14–1 – 14–24. Engineering design reliability. CRC Press, Boca Raton , Fl., 2005.

[2] J. Song, W.-H. Kang, Y.-J. Lee, and J. Chun. Structural system reliability: Overview of theories and applications to optimization. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, 7(2), 2021. 03121001.

[3] A. T. Melchers, R. E. amd Beck. *Structural reliability analysis and prediction.* John Wiley & Sons, 2018.

[4] M. Moustapha, S. Marelli, and B. Sudret. Active learning for structural reliability: survey, general framework and benchmark. *Structural Safety*, 2021. (Accepted).

[5] B. J. Bichon, M. S. Eldred, L. Swiler, S. Mahadevan, and J. McFarland. Efficient global reliability analysis for nonlinear implicit performance functions. *AIAA Journal*, 46(10):2459–2468, 2008.

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

[7] R. Teixeira, M. Nogal, and A. O'Connor. Adaptive approaches in metamodel-based reliability analysis: A review. *Structural Safety*, 89:102019, 2021.

[8] B. J. Bichon, J. M. McFarland, and S. Mahadevan. Efficient surrogate models for reliability analysis of systems with multiple failure modes. *Reliability Engineering & System Safety*, 96(10):1386–1395, 2011.

[9] B. Echard, N. Gayton, M. Lemaire, and N. Relun. A combined importance sampling and Kriging reliability method for small failure probabilities with time-demanding numerical models. *Reliability Engineering & System Safety*, 111:232–240, 2013.

[10] X. Yang, C. Mi, D. Deng, and Y. Liu. A system reliability analysis method combining active learning Kriging model with adaptive size of candidate points. *Structural and Multidisciplinary Optimization*, 60:137–150, 2019.

[11] M. Xiao, J. Zhang, and L. Gao. A system active learning Kriging method for system

reliability-based design optimization with a multiple response model. *Reliability Engineering & System Safety*, 199:106935, 2020.

[12] S. K. Au and J. L. Beck. Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic Engineering Mechanics*, 16(4):263–277, 2001.

[13] I. M. Sobol'. Sensitivity estimates for nonlinear mathematical models. *Math. Modeling & Comp. Exp.*, 1:407–414, 1993.

[14] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning.* Adaptive computation and machine learning. MIT Press, Cambridge, Massachusetts, Internet edition, 2006.

[15] C. Lataniotis, S. Marelli, and B. Sudret. UQLab user manual – Kriging. Technical report, Chair of Risk, Safety & Uncertainty Quantification, ETH Zurich, 2017. Report # UQLab-V1.0-105.

[16] R. Schöbi, B. Sudret, and J. Wiart. Polynomial-chaos-based Kriging. *International Journal of Uncertainty Quantification*, 5(2):171–193, 2015.

[17] R. Schöbi, B. Sudret, and S. Marelli. Rare event estimation using Polynomial-Chaos-Kriging. *ASCE-ASME J. Risk Uncertainty Eng. Syst., Part A: Civ. Eng*, 3(2), 2017. D4016002.

[18] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. Han, and U. Fayyad, editors, *Proc. 2nd international conference on knowledge discovery and data mining, Portland, Oregon, USA, August 2-4, 1996.* AAAI press, 1996.

[19] S. Marelli and B. Sudret. UQLab: A framework for uncertainty quantification in Matlab. In *Vulnerability, Uncertainty, and Risk (Proc. 2nd Int. Conf. on Vulnerability, Risk Analysis and Management (ICVRAM2014), Liverpool, United Kingdom)*, pages 2554–2563, 2014.

[20] Chunyan Ling, Z. Lu, K. Feng, and Xiaobo Zhang. A coupled subset simulation and active learning Kriging reliability analysis method for rare failure events. *Structural and Multidisciplinary Optimization*, 60:2325–2341, 2019.

[21] Qing Guo, Yongshou Liu, Bingqian Chen, and Qin Yao. A variable and mode sensitivity analysis method for structural system using a novel active learning Kriging model. *Reliability Engineering & System Safety*, 206:107285, 2021.

[22] M. McDonald and S. Mahadevan. Design optimization with system-level reliability constraints. *Journal of Mechanical Design*, 130(2), 2008.