Reliability Engineering and System Safety 241 (2024) 109664

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

Sampling-based adaptive design strategy for failure probability estimation

Tiexin Guo [a], Hongji Wang [a], Jinglai Li [b], Hongqiao Wang [a,*]

[a] School of Mathematics and Statistics, Central South University, Changsha, 410083, People's Republic of China
[b] School of Mathematics, University of Birmingham, Edgbaston, Birmingham, B15 2TT, UK

ARTICLE INFO

Keywords:
Adaptive design of experiment
Failure probability
Normalizing flows

ABSTRACT

Failure probability (FP) estimation problem is a crucial task in engineering. In this work we consider this problem in the situation that the underlying computer models are extremely expensive, which often arises in the practice, and in this setting, reducing the calls of computer model is of essential importance. We formulate the problem of estimating the failure probability with expensive computer models as an sequential experimental design for the limit state (i.e., the failure boundary) and propose a series of efficient adaptive design criteria to solve the design of experiment (DOE). Considering the remarkable achievements of neural networks, we aim to leverage this powerful tool for surrogate modeling and sampling purposes. In particular, the proposed method employs the deep neural network (DNN) as the surrogate of limit state function for efficiently reducing the calls of expensive computer experiment. A map from the Gaussian distribution to the posterior approximation of the limit state is learned by the normalizing flows for the ease of experimental design. Three normalizing-flows-based design criteria are proposed in this work for deciding the design locations based on the different assumption of generalization error. The accuracy and performance of the proposed method is demonstrated by both theory and practical examples. The relative error of FP estimation achieved by the proposed methods is consistently below ten percent.

1. Introduction

Real-life engineering systems are unavoidably subject to various uncertainties such as material properties [1], geometric parameters [2], boundary conditions [3] and applied load [4]. These uncertainties may lead to unexpected events, such as system failures or malfunctions. Accurate identification of failure region and evaluation of failure probability of a given system are essential tasks in many engineering fields such as risk management [5], structural design [6], reliability-based optimization [7], etc.

Conventionally the failure probability (FP) is often computed by constructing linear or quadratic expansions of the system model around the so-called most probable point, known as the first/second-order reliability method (FORM/SORM), see e.g., [8] and the references therein. It is well known that FORM/SORM may fail for systems with nonlinearity or multiple failure regions. The Monte Carlo (MC) simulation [9], which estimates the failure probability by repeatedly simulating the underlying system, is another popular method for solving such problems. The MC method makes no approximation to the underlying computer models and thus can be applied to any systems. On the other hand, the MC method is notorious for its slow convergence, and thus can become prohibitively expensive when the underlying computer model is computationally intensive and/or the system failures are rare and each sample requires a full-scale numerical simulation of the system. To reduce the computational effort, three kind of strategies are proposed. One kind of strategy is to solve FP in an augmented space. Bayes' rule is utilized firstly to include the solution of FPF in an augmented reliability problem, where the design parameters are treated as random variables with a predefined distribution [10]. This allows to estimate the FPF in a single simulation run. Based on the augmented space idea, lots of methods are proposed [11–13]. The second strategy is a reweighting approach, which builds a local approximation of FPF based on the information of a single reliability analysis. This kind of method focuses on the problem where only the distribution parameter is included in the analysis [14–17]. The third strategy is to construct a computationally inexpensive approximation of the true model, and then evaluate the approximate model in the MC simulations. Such approximate models are also known as response surfaces, surrogates, metamodels, and emulators, etc. These methods are referred to as the response surface (RS) methods [18–21] in this work. The response surface can often provide a reliable estimate of the failure probability, at a much lower computational cost than direct MC simulations. Machine learning-based methods in structural reliability analysis [18,22–28], An efficient reliability analysis method for structures with hybrid time-dependent

* Corresponding author.
E-mail address: Hongqiao.Wang@csu.edu.cn (H. Wang).

https://doi.org/10.1016/j.ress.2023.109664
Received 25 February 2023; Received in revised form 15 August 2023; Accepted 14 September 2023
Available online 19 September 2023

uncertainty [29],Adaptive network reliability analysis: Methodology and applications to power grid [30], A hybrid data-driven model for geotechnical reliability analysis [31], machine learning-based methods for reliability analysis develop fast in recent years and we only list parts of works which are representative and closely related to the method in this paper.

This study centers on a particular type of RS known as deep neural network (DNN) surrogates within a Bayesian inference framework. A Bayesian approach is investigated for identifying the failure boundary using a DNN model as a surrogate for the costly limit state function. Additionally, we aim to determine the optimal experimental design through the application of innovative normalizing flows, eliminating the need for an optimization procedure. The DNN surrogates have been widely used in machine learning [32], geostatistics [33], engineering optimizations [34], and most recently, uncertainty quantifications [35,36]. It has great ability in fitting and shows great potential on high dimensional fitting compared with other surrogates like, Gaussian process regression, Karhunen–Loeve expansion, support vector machine and so on. Rapid advancements have been made in the development of surrogate models for expensive systems, particularly in the domain of deep neural networks (DNNs). This includes the emergence of physics-based deep learning approaches, such as physics-informed neural networks (PINNs) [37], U-net architecture [38], Graph-based probabilistic geometric deep learning [39], Multiscale methods and deep learning [40,41], Surrogate models to accelerate the simulations [42]. Considering the probabilistic nature, Bayesian methods have developed rapidly in the field of parameter estimation/identification, and many Bayesian methods have been proposed in recent years [43–49]. This type of method is very suitable to cooperate with the variational method or MCMC method to calculate the posterior density of the parameters and give the uncertainty information of the parameter estimation.

In this work we consider the situation where the underlying computer models are extremely expensive and one can only afford a very limited number of simulations. In this setting, choosing the sampling points (i.e. the parameter values with which the simulation is performed) in the state space is of essential importance. Determining the sampling points for neural network can be cast as optimally designing computer experiments. A simple and straightforward idea aims to construct a surrogate that can accurately approximate the target function in the whole parameter space. As will be explained later, in the failure probability estimation or failure detection problems, only the sign of the target function is used. Thus with requiring surrogates to be globally accurate, the method may allocate considerable computational efforts to the regions not of interest, and use much more model simulations than necessary.

Several methods have been developed to determine the sampling points for the failure probability estimation. Most of these methods consist of sequentially finding the "best point" as a result of a heuristic balance between predicted closeness to the limit state, and high prediction uncertainty, e.g. [17,50–52]. Such methods are shown to be effective in many applications, while a major limitation is their point-wise sequential nature, which makes it unsuitable for problems in which multiple computer simulations can be performed parallel. The stepwise uncertainty reduction (SUR) method developed in [53,54] is one of the two exceptions, in which the authors proposed an optimal experimental design framework which determines multiple sampling points by minimizing the average variance of the failure probability. It should be noted that the design criteria in the SUR method is particularly developed for the goal of estimating the failure probability only. In practice, one is often interested in not only estimating the failure probability, but also identifying the events that can cause failures; the latter demands a design criteria for the goal of detecting the limit state, i.e., the boundaries of the failure domain. Another exception is the Gaussian process based failure boundary and probability estimation methods [24,55,56], which determine the multiple sampling points by maximizing the information gain based design criteria. The above multiple sampling points design methods would suffer from the same bottleneck that the difficulty of optimization would increase as the amount and the dimension of design points increase. This bottleneck seriously reduces the possibility of searching global optimal design and limits the application of these methods in real world. Simultaneously determining multiple optimal design points poses a significant challenge, particularly in cases involving high-dimensional limit state functions. Traditional frameworks for optimization-based design criteria have been developed, but achieving global maxima/minima numerically remains difficult. In order to address this issue, we propose an alternative approach: sampling-based optimal design. The objective of our work is twofold: first, to estimate the failure probability, and second, to infer the limit state in cases where the limit state function is computationally expensive. To achieve this, we leverage the success of neural networks as a powerful tool for surrogate modeling and sampling. We recast the neural network (NN) surrogate construction as a Bayesian inference to identify the distribution of limit state, and based on that, we propose three normalizing-flows-based design criteria to determine the sampling points. The proposed neural network based method could be easily applied in the high dimensional case and it avoids the step of optimization which is essential in the above optimal design methods. We compare the performance of the proposed method with that of the LSI by numerical examples.

We note that another line of research in failure probability estimation is to develop more efficient sampling schemes, such as the subset simulations [57], importance sampling [58–60], the cross-entropy method [61,62], hybrid enhanced MCS [23] etc. For practical engineering systems, computer simulations can be extremely time consuming. In many cases, one can only afford very limited number of simulations. In this case, even the most effective sampling method is not applicable. To this end, surrogates are needed even in those advanced sampling schemes and in particular the proposed method can be easily integrated into the aforementioned sampling schemes, resulting in more efficient estimation schemes. Examples of combining surrogates and efficient sampling schemes include [63–66].

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

2. Problem formulation

2.1. Failure probability estimation framework

In a general setting of failure probability estimation problem [55], we consider a d-dimensional random variable $\boldsymbol{x}$ that represents input variable with uncertainties and let $\Omega \subseteq \mathbb{R}^d$ be the state space of. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, where Ω is a sample space, $\mathcal{F}$ is a σ-field, and $\mathbb{P}$ is a probability measure on $(\Omega, \mathcal{F})$.We model a system using a real-valued function $g(\cdot) : \Omega \to R$, which is known as the limit state function or the performance function. The event of failure is defined as $g(\boldsymbol{x}) \leq 0$ and as a result the failure probability is

$$P = \mathbb{P}(g(\boldsymbol{x}) \leq 0) = \int_{\{\boldsymbol{x}\in\Omega | g(\boldsymbol{x})\leq 0\}} p_X(\boldsymbol{x}) d\boldsymbol{x} = \int_{\Omega} I_g(\boldsymbol{x}) p_X(\boldsymbol{x}) d\boldsymbol{x}, \tag{1}$$

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

$$I_g(\boldsymbol{x}) = \begin{cases} 1 & \text{if } g(\boldsymbol{x}) \leq 0, \\ 0 & \text{if } g(\boldsymbol{x}) > 0; \end{cases} \tag{2}$$

and $p_X(\boldsymbol{x})$ is the probability density function (PDF) of $\boldsymbol{x}$. In what follows we shall omit the integration domain when it is simply Ω. This is a general definition for failure probability, which is used widely in many

disciplines involving reliability analysis and risk management. P can be computed with the standard Monte Carlo (MC) estimation:

$$\hat{P} = \frac{1}{n}\sum_{i=1}^{n} I_g(\boldsymbol{x}_i), \tag{3}$$

where samples $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ are drawn from $p_X(\boldsymbol{x})$ which can be any probability density function. The failure probability can be estimated by MC method and this method does not require any assumptions on systems.

A high reliable estimate of the small failure probability, for example, $P \ll 1$, is required in many practical engineering systems. In this case, MC requires a rather large number of samples to produce a reliable estimate of the failure probability. For example, for $P \approx 10^{-3}$, MC simulation requires 10^5 samples to obtain an estimate with 10% coefficient of variation. On the other hand, in almost all practical cases, the limit state function $g(\boldsymbol{x})$ does not admit analytical expression and has to be evaluated through expensive computer simulations, which renders the crucial MC estimation of the failure probability prohibitive. To reduce the number of full-scale computer simulations, one can construct a computationally inexpensive surrogate $G(\boldsymbol{x})$ to replace the real function $g(\boldsymbol{x})$ in the MC estimation. In this work we choose the powerful Deep Neural Network (DNN) model as an efficient surrogate $G(\boldsymbol{x})$ due to its outstanding success in high dimensional scenario.

3. Methods

3.1. Deep neural network surrogate of expensive computer simulation

The powerful DNN have shown its strong fitting ability in lots of areas [67,68]. Here we employ it to construct the surrogate of real expensive limit state function $g(\boldsymbol{x})$. The data is obtained by the simulation of the limit state function $g(\cdot)$ with given $\boldsymbol{x}$. Dataset can be constructed denoted as $D = \{\boldsymbol{x}_i, y_i\}_{1,\dots,n}$, where $y_i = g(\boldsymbol{x}_i)$. With generality, we prefer to use the full connection neural network architecture which is the most simple neural network structure as the surrogate function $G(\cdot) := G_{out} \circ \dots \circ G_l \circ \dots \circ G_{in}(\cdot)$ in most cases. But for more challenging limit state functions like parametric PDE solvers, we prefer a more complex architecture, like Fourier Neural Operator (FNO) [69].

The basic idea of deep neural networks (DNNs) for surrogate model is that it can approximate an input–output map $G : \mathbb{R}^d \to \mathbb{R}$ through a hierarchical abstract layers of latent variables. A typical example is the feedforward neural network, which is also called multi-layer perception (MLP). It consists of a collection of layers that include an input layer $G_{in}(\cdot)$, an output layer $G_{out}(\cdot)$, and a number of hidden layers $G_k(\cdot), k = 1, \dots, K$. The size of the input layer and output layer are fixed and determined by the dimensionality of the input and output. Each element of $G_k(\cdot)$ is a neuron which calculates a weighted sum of an input vector plus bias and applies a non-linear function to produce an output. Specifically, given an d-dimensional input row vector $\boldsymbol{x} \in \mathbb{R}^d$, we can define a DNN with L hidden layers as following

$$\mathcal{NN}(\boldsymbol{x}) = W^{(L)} a^{(L)} + b^{(L)} \tag{4}$$

$$a^{(k+1)} = \sigma(W^{(k)} a^{(k)} + b^{(k)}), \quad k = 0, \dots, L-1. \tag{5}$$

Here $W^{(k)} \in \mathbb{R}^{d_{k+1} \times d_k}$, $b^{(k)} \in \mathbb{R}^{d_{k+1}}$ are the weights and biases of the network, d_k is the number of neurons in the kth layer and σ is the activation function. Notice that here $a^{(0)}$ is the input $\boldsymbol{x}$ and $d_0 = d$. Some popular choices for the activation function include sigmoid, hyperbolic tangent, rectified linear unit (ReLU), to name a few. In the current work, we shall use Swish as the activation function:

$$\sigma(z) = \frac{z}{1 + \exp(-z)}. \tag{6}$$

Once the network architecture is defined, one can resort to optimization tools to find the unknown parameters $\boldsymbol{\theta} = \{W^{(k)}, b^{(k)}\}$ based on the training data. Precisely, let $D := \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ be a set of training data, we can define the following minimization problem:

$$\arg\min \mathcal{J}(\boldsymbol{\theta}; D) = \frac{1}{N}\sum_{i=1}^{N} \|y_i - \mathcal{NN}(\boldsymbol{\theta}; \boldsymbol{x}_i)\|^2, \tag{7}$$

where $\mathcal{J}(\boldsymbol{\theta}; D)$ is the so called loss function. Solving this problem is generally achieved by the stochastic gradient descent (SGD) algorithm which minimizes the loss function by taking a negative step along an estimate of the gradient ∇ at iteration k. The gradients are usually computed through back propagation. At each iteration, SGD updates the solution by

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \epsilon \nabla_\theta \mathcal{J}(\boldsymbol{\theta}; D_M), \tag{8}$$

where ϵ is the learning rate and D_M is batch dataset. Recent algorithms that offer adaptive learning rates are available, such as AdaGrad [70], Adam and RMSProp [66], ect. The present work adopts Adam optimization algorithm.

For parametric PDE problem, we could employ a more efficient neural network architecture, named as Fourier Network Operator (FNO) [69], to construct a more complex and accurate surrogate. The last example in Section 4 will introduce the failure probability estimation problem in PDE situation in detail.

3.2. The sequential failure probability estimation framework

The failure probability can be estimated by the NN surrogate under the assumption that the data points are determined all in advance of performing computer simulations, which is often referred to as an open-loop design. But an accurate NN surrogate needs to be trained with a large number of data which is still computational intensive. In many applications, a more practical strategy is to choose the sampling points in a sequential fashion: determine a set of sampling points, perform simulations, determine another set of points based on the previous results, and so forth. A sequential (close-loop) design can be readily derived from the open-loop version. Simply speaking, the sequential design iterates as follows until a prescribed stopping criterion is met:

1 construct a NN model $G(\boldsymbol{x})$ for $g(\boldsymbol{x})$ using data-set D;
2 determine n sampling points $\{\boldsymbol{x}_1^*, \dots, \boldsymbol{x}_n^*\}$ with an open-loop design;
3 evaluate $y_i^* = g(\boldsymbol{x}_i^*), i = 1, \dots, n$ and let $D = \{D, (\boldsymbol{x}_1^*, y_1^*), \dots, (\boldsymbol{x}_n^*, y_n^*)\}$;

Note that the key in the sequential scheme is step 2, where we efficiently seek the more informative sampling points. Different from the traditional optimization based experimental design criteria [53–56], here we propose a series of novel normalizing-flows-based sampling strategies which map an common distribution to the posterior distribution of limit state and decide the design points based on the samples of it in Section 3.5. These criteria avoid the challenge of searching the global optimal which is a major problem in the field of optimization and reduces the undetermined time required for optimization. Before introducing the specific design criteria, we first define the posterior distribution of limit state which is an important concept in our method for the estimation of failure probability.

3.3. Posterior distribution of limit state

In the failure probability estimation, the limit state function g is only used in the indicator function $I_g(\boldsymbol{x})$ and so one is really interested in the sign of $g(\boldsymbol{x})$ rather than the precise value of it. To this end, the essential task in constructing surrogate for the failure probability estimation is to learn about the boundary of the failure domain. Here we emphasize that the indicator function $I_g(\boldsymbol{x})$ is a step function, but the limit state function $g(\boldsymbol{x})$ is continuous. Let $Z = \{\boldsymbol{z} := \boldsymbol{x} | \boldsymbol{x} \in \Omega, g(\boldsymbol{x}) = 0\}$ represents the boundary of the failure domain, i.e., the collection of solutions

of $g(\boldsymbol{x}) = 0$ and define the distribution of $\boldsymbol{z}$, $p(\boldsymbol{z}) = \frac{1}{C}\exp(-\frac{|g(\boldsymbol{z})-0|}{\lambda})$, where C is a normalization constant. Similarly, we define the posterior distribution of limit state in surrogate, i.e.,

$$p(\tilde{\boldsymbol{z}}|D) := p(G(\tilde{\boldsymbol{z}}) = 0|D) \propto \exp(-\frac{|G(\tilde{\boldsymbol{z}}) - 0|}{\lambda}), \tag{9}$$

where λ is scale parameter which scales the magnitude the output $G(\tilde{\boldsymbol{z}})$ and $G(\cdot)$ follows the description of Section 3.1. With the increment of data, the posterior distribution of limit state in surrogate $p(\tilde{\boldsymbol{z}}|D)$, would converge to the $p(\boldsymbol{z})$.

3.4. Density transformation via normalizing flows

Though we can obtain the unnormalized posterior density of limit state, we prefer calling the true posterior density and its samples. Here we employ the normalizing flows technique which have been shown its strong fitting ability in [71] to approximate the limit state posterior . NF attracts us with its cheap sampling procedure and density calculation.

The basic rule for transformation of densities considers an invertible, smooth map $f : \mathbb{R}^d \to \mathbb{R}^d$ with inverse $f^{-1} = h$, i.e. the composition $h \circ f(\boldsymbol{z}_0) = \boldsymbol{z}_0$. If we use this map to transform a random variable $\boldsymbol{z}_0$ with distribution $q(z_0)$, the resulting random variable $\tilde{\boldsymbol{z}} = f(\boldsymbol{z}_0)$ has a distribution:

$$q(\tilde{\boldsymbol{z}}) = q(\boldsymbol{z}_0)|\det \frac{\partial f^{-1}}{\partial \tilde{\boldsymbol{z}}}| = q(\boldsymbol{z}_0)|\det \frac{\partial f}{\partial \boldsymbol{z}_0}|^{-1}, \tag{10}$$

where the last equality can be seen by applying the chain rule (inverse function theorem) and the property of Jacobians of invertible functions. DNN is used to exactly approximate f for constructing arbitrarily complex densities. For completeness of the paper, we briefly introduce the normalizing flows in Appendix B.

3.5. Adaptive experimental design via normalizing flows

Normalizing-flows-based design (NFBD)

A simple and common adaptive strategy for experimental design is to take the experiments at the locations where we are interested [72–74]. In this problem we are interested in the distribution of limit state, $p(\tilde{\boldsymbol{z}}|D)$, and therefore a basic adaptive experimental design criterion is to use the sample points of $p(\tilde{\boldsymbol{z}}|D)$ as the design locations. We could obtain the transform map f by normalizing flows in Section 3.4 and then the sample points (design locations) of $p(\tilde{\boldsymbol{z}}|D)$, can be drawn easily by the transformation $\tilde{\boldsymbol{z}} = f(\boldsymbol{z}_0)$, where $\boldsymbol{z}_0$ are the samples of Gaussian distribution $q(\boldsymbol{z}_0)$.

Normalizing-flows-based design with fixed generalization (NFBD-FG)

The basic adaptive sampling strategy is effective, but would generate some less informative designs since there will be a lot of experiments lying in the high probability region of limit state and some of them would flock together. Two adjacent designs will provide duplicate information which is meaningless for improving the accuracy of surrogate. With the consideration of the data sparsity, interpolation uncertainty here is an inherent epistemic uncertainty associated with machine learning models, when they are used to predict new data points through interpolation/extrapolation. Here we give a fixed smoothness assumption about the limit state function and a straightforward strategy of reducing experimental calls can be obtained by adding a selection procedure on the NFBD criterion, named as NFBD-FG. For measuring the distance between a design candidate $\boldsymbol{z}$ and points in the current data-set D, we define a distance function

$$\rho(\boldsymbol{z}, \mathbb{D}) = \min(\|\boldsymbol{z} - \boldsymbol{x}_1\|_{L_\infty}, \dots, \|\boldsymbol{z} - \boldsymbol{x}_m\|_{L_\infty}), \tag{11}$$

where $\boldsymbol{x}_i$ represents the experiment location in $D = \{(\boldsymbol{x}_i, y_i)\}, i = 1, \dots, m$. This distance function could return the minimum distance value between the design candidate and data locations of current data-set D. Here we sequentially measure the distance between each proposed design candidate and the exiting experimental locations and accept the proposed sample with $1_{\rho(\tilde{\boldsymbol{z}},\mathbb{D})\le\epsilon}$, where ϵ is a preset value indicating a fixed generalization assumption.

Normalizing-flows-based design with adaptive generalization error (NFBD-AG)

Though we can simply use the distance function $\rho(\cdot,\cdot)$ to quantify the interpolation error of surrogate, the impact of it would be different in the input variable domain, i.e., the failure probability error caused by same interpolation error in high probability domain would be smaller than the one in low probability domain. Thus we propose an adaptive threshold strategy: the threshold value would not be constant but a function with respect to the probability density function value of input variable at proposed design points. It has been shown that when the expectation is calculated over all possible data distributions, as sample complexity increases, generalization error will decline following a power-law [75]

$$e(N) \sim \alpha N^{\gamma}. \tag{12}$$

This pow-law rule was observed in some empirical studies. It is worth noting that there are many choices of γ in real applications and in this paper we choose $\gamma = -0.5$ without loss of generality. In our generalization set as shown in Eq. (11) and the rule of acceptance $1_{\rho(\tilde{\boldsymbol{z}},\mathbb{D})\le\epsilon}$, we have $N = \frac{1}{\epsilon^d}$. Eq. (12) becomes

$$e(\epsilon) \sim \alpha(\frac{1}{\epsilon^d})^{-0.5}. \tag{13}$$

We note that more samples draw from the high probability domain of $p(\boldsymbol{x})$ for the computation of failure probability in MC methods and thus the same generalization error in different probability area of $p(\boldsymbol{x})$ would lead to different error bounds in the estimation of failure probability. For alleviating the above difficulty, we propose an adaptive generalization strategy in Eq. (14) which means the generalization error is not set up as a fixed value in advance, but adaptively adjusted with the probability of $p(\boldsymbol{x})$, named as NFBD-AG. The adaptive generalization error method could decline the generalization error in high probability area and allow a larger generalization error in low probability area.

$$e(\epsilon(\boldsymbol{x})) \propto \frac{1}{p(\boldsymbol{x})} \Rightarrow \epsilon(\boldsymbol{x}) \propto \beta\frac{1}{p(\boldsymbol{x})^{\frac{2}{d}}} \tag{14}$$

where $\beta = \frac{1}{\alpha^{\frac{d}{2}}}$.

Remark 1. In practice we could not know the exact value of β. Here we set a empirical preset threshold ϵ_0 corresponding to the median pdf value of $p_X(\boldsymbol{z})$ in each iteration and then we have $\beta = \epsilon_0 p(\boldsymbol{z}_{median})^{\frac{2}{d}}$. For preventing the value $\epsilon(\boldsymbol{z})$ too extreme, a bound $[0.1\epsilon_0, 10\epsilon_0]$ is used here to constrain the adaptive threshold $\epsilon(\boldsymbol{z})$ in a reasonable range, not too big (larger than $10\epsilon_0$) or too small (smaller than $0.1\epsilon_0$).

Discussion with similar studies

The proposed method shares similarities with other reliability methods that incorporate both Kriging and Monte Carlo Simulation in active learning approaches [24,26,28,50,55] . The AK-MCS method has demonstrated high efficiency in accurately estimating failure probabilities, requiring only a small number of calls to the performance function. In comparison to Gaussian process-based surrogate methods, the Deep neural network model exhibits superior fitting capabilities for numerous high-dimensional problems, making it a promising surrogate approach for estimating failure probabilities in high-dimensional scenarios. In these methods, density estimation or posterior sampling plays a crucial role. However, traditional kernel density estimation and Markov Chain Monte Carlo (MCMC) methods are time-consuming and limited in terms of dimensionality. Moreover, optimization procedures are indispensable, particularly in high-dimensional and/or parallel design settings. To address these challenges, our proposed method employs normalizing flows to generate samples and determine the "optimal" design. This approach proves to be efficient and overcomes the limitations of traditional density estimation and MCMC methods in terms of time and dimensionality.

3.6. Numerical implementation

Design via normalizing flows (DNF) algorithm

For clearly illustrating our proposed design via normalizing flows method, named as "DNF", and list the full algorithm as follows:

Algorithm 1: DNF method

Parameter: max number of simulations $N_{\max}$, number of initial data points N_0, number of design locations in each adaptive experimental design update N_D and the default threshold ϵ_0;
Output : estimation of failure probability P_f;

Randomly select N_0 points $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_0})$ and compute $y_i = g(\boldsymbol{x}_i)$. Obtain the initial data-set $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1,\cdots,N_0}$;
Train the NN surrogate function G and get the model parameters θ by minimizing the loss function with the data-set $\mathcal{D}$ (Section 3.1);
Estimate the failure probability $\hat{p}_f^0$ by Equation (3) with the NN surrogate $G(\boldsymbol{x})$ and compute $t_{\max} = \lceil \frac{N_{max} - N_0}{N_D} \rceil$;
for $t = 1$ **to** $t_{\max}$ **do**
  if $t < t_{max}$ **then**
    $N_t = N_D$;
  else
    $N_t = N_{\max} - N_0 - (t_{max} - 1) * N_D$
  end
  Train the normalizing flows and obtain the map from a common distribution $q(\boldsymbol{z}_0)$ to the posterior distribution of limit state $p(\tilde{z}|D)$ defined in (9);
  Determine N_t new design locations $D = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{N_t}\}$ by adaptive experimental design scheme (NFBD, NFBD-FG or NFBD-AG) with ϵ_0 (Section 3.5) ;
  Evaluate the corresponding values $\{y_1, \ldots, y_{N_t}\}$ at D by accessing the limit state function $g(\boldsymbol{x})$;
  Use the new data-set $\mathcal{D} = \mathcal{D} \cup \{(\boldsymbol{d}_i, y_i)\}_{i=1,\ldots,N_t}$ to retrain the NN surrogate G;
  Estimate the failure probability $\hat{p}_f^t$ by Equation (3) with the NN surrogate $G(\boldsymbol{x})$;
  if $|\frac{\hat{p}_f^t - \hat{p}_f^{t-1}}{\hat{p}_f^{t-1}}| > tolerance$ **then**
    $\hat{P}_f = \hat{P}_f^t$; Break the loop;
  end
end

Some remarks on the implementation of algorithm 1 are listed in order:

1 In normalizing flows the choice of $q(\boldsymbol{z}_0)$ can be any common probability density like uniform distribution, Gaussian distribution, etc., and in our numerical examples, we set $q(\boldsymbol{z}_0) = \mathcal{N}(\boldsymbol{z}_0; \boldsymbol{0}, I)$ as the Standard Gaussian distribution without loss of generality.
2 The threshold ϵ_0 is required in both NFBD-FG and NFBD-AG strategies, but unnecessary in NFBD strategy.
3 The stopping criterion, $|\frac{\hat{p}_f^t - \hat{p}_f^{t-1}}{\hat{p}_f^{t-1}}| < tolerance$, refers that the loop can be stopped when the relative error of the failure probability obtained by two adjacent iterations is smaller than a preset tolerance value, $tolerance$. In the numerical examples, we set $tolerance = 10\%$.

The performance of the DNF algorithm with NFBD, NFBD-FG and NFBD-AG design criteria

As aforementioned in Section 3.5, NFBD criterion is a greedy strategy without any generalization assumption and thus DNF algorithm with NFBD criterion would lead to an accurate estimation of failure probability with the data increase. But This criterion would be inefficient because the proposed design points may be close together and/or be allocated in the low probability domain. The assumption of generalization error in both NFBD-FG and NFBD-AG criteria could avoid the situation of points close together, but would inevitably introduce a certain irreducible generalization error. Compared with NFBD-FG, the generalization error assumed by NFBD-AG criterion would be small in high probability domain and big in low probability domain caused by its adaptive threshold function (14). In this setting, a relative smaller generalization assumption is given in high probability domain and bigger one in low probability domain, compared with ϵ_0.

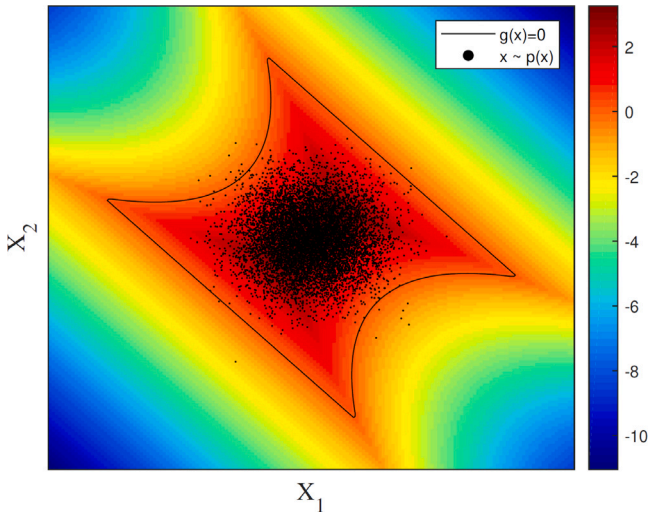

Fig. 1. For four branch system example, the limit state function and the samples drawn from the prior distribution $p(\boldsymbol{x})$. The solid line indicates the failure boundary.

4. Examples

In this section we first consider the failure probability estimation problem in two 2−d mathematical examples so that we can validate the approximate limit states with the exact ones. In Section 4.3, we apply our method to a computational intensive PDE simulation problem: the Darcy Flow equation, which is a changeling benchmark problem.

4.1. Four branch system

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

$$g(x_1, x_2) = \min \begin{Bmatrix} 3 + 0.1(x_1 - x_2)^2 - (x_1 + x_2)/\sqrt{2} \\ 3 + 0.1(x_1 - x_2)^2 + (x_1 + x_2)/\sqrt{2} \\ (x_1 - x_2) + 7/\sqrt{2} \\ (x_2 - x_1) - 7/\sqrt{2} \end{Bmatrix} \tag{15}$$

which is shown in Fig. 1. In this figure, the input random variable x_1 and x_2 are assumed to be independent and follow standard normal distribution, i.e., $p(x_i) \sim \mathcal{N}(0, 1), i = 1, 2$. We compute the failure probability with a standard MC estimation of 10^5 samples (the black dots in the figure), resulting an estimate of 2.05×10^{-3}. We regard the result of MC as the true probability of failure.

In the proposed adaptive experimental design methods, we first choose 25 points equally spaced in $\Omega = [-10, 10] \times [-10, 10]$ as the initial design points and then 2 design points determined in each iteration. In order to fully demonstrate the characteristics of different design criteria, we let the DNF algorithm terminates until the number of

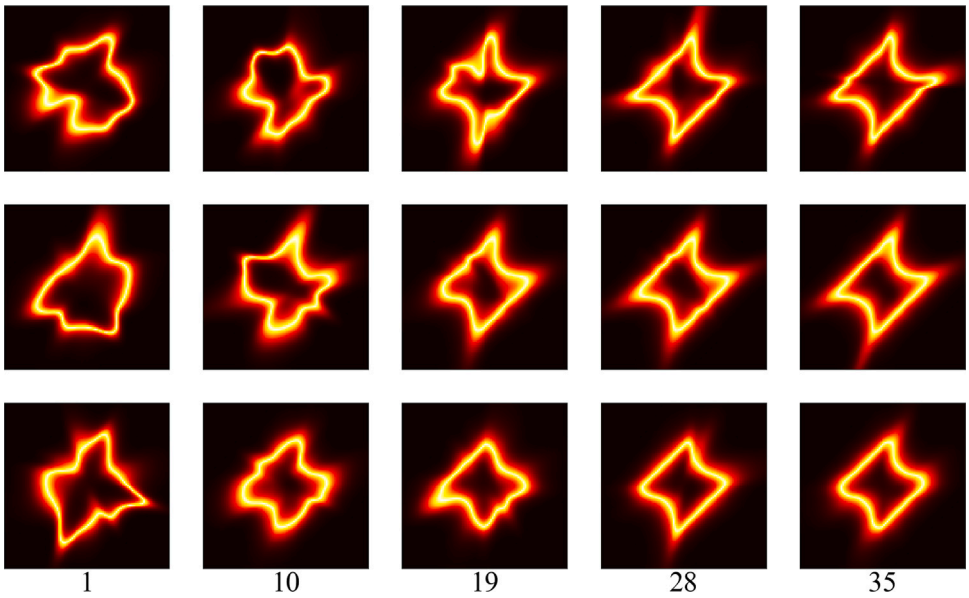

Fig. 2. For four branch system example, the approximation of limit state learned by NFBD (top row), NFBD-FG (middle row) and NFBD-AG (bottom row) criteria, with respect to the first, 10th, 19th, 28th and last iteration.

simulations reaches its max $N_{max} = 95$, resulting in totally 35 iterations. Fig. 2 shows the approximations of limit state learned by DNF with NFBD, NFBD-FG and NFBD-AG strategies with respect to the iterations. We can find that approximations computed by all the three criteria are bad as shown in the first column and are gradually approaching the real one as the number of iterations increases. The contours shown in the last column (also the 35th iteration) are pretty similar to the real limit state as shown in the black line of Fig. 1. It verifies that the failure boundary can be obtained by our normalizing-flows-based design framework.

We plot the design points and their corresponding approximations of limit state computed by these three design criteria in Fig. 3. We can see that both NFBD and NFBD-AG methods allocate more points near the boundary of the failure domain than NFBD-FG and it is reasonable because of the fixed generalization error assumption which makes the design points more sparse in Ω. We can find that the fitting accuracy of the failure boundary computed by NFBD are same both in the high probability domain and low probability domain, which is different with the one computed by NFBD-AG with same amount of data. But it is inevitable that there will be design points where information is repeated. NFBD-AG would like allocate more design points in failure boundary of the high probability domain resulting to an high accuracy estimation there. So NFBD-AG would quickly get an estimation in the correct order of magnitude.

We now compare the results of the three design strategies in Fig. 4. The left subfigure shows that all the three methods provide an acceptable fitting accuracy in the high probability area of failure boundary. We plot the estimation of logarithmic failure probability as a function of the number of iterations in Fig. 4 (right). In the figure we can see that the curve of our basis NFBD method oscillates greatly before 23th iteration and it asymptotically stably converges to the groundtruth. The curve of NFBD-FG method quickly stabilizes around the groundtruth, but the estimation accuracy does not increase significantly as the number of iterations increases. It is because a fixed generalization assumption could lead to a quick exploration for the failure boundary and a upper limit of the accuracy of this estimation. Compared with the other two methods, the curve of NFBD-AG performs well both on the speed of stabilization and the estimation accuracy, which benefits from its adaptive generalization error. The estimates of failure probability computed by NFBD, NFBD-FG and NFBD-AG in the last iteration are 2.13×10^{-3}, 2.18×10^{-3} and 2.15×10^{-3}, respectively, which are more accurate than the one 2.86×10^{-3} computed by Latin hypercube sampling method with the same number of evaluations. We can find that all these criteria result in a less than ten percent relative error.

4.2. Iso-probability lines

Our second example is the iso-probability lines and its specific expression of the system is

$$g(x_1, x_2) = b - x_2 - k(x_1 - e)^2, \tag{16}$$

where $b = 5$, $k = 0.5$ and $e = 0.1$. The prior of each parameter in $\boldsymbol{x} = [x_1, x_2]$ is also standard Normal distribution. 10^5 samples are used for MC estimation of the failure probability, resulting an estimate of 3.01×10^{-3}.

In this example, 5 points equally spaced in $\Omega = [-10, 10] \times [-10, 10]$ are chosen as the initial design points and 40 design points are determined by the DNF algorithm with one point determined in each iteration.

We plot the approximate posteriors obtained in the first, 10th, 20th, 30th and last iterations in Fig. 5, in which we can visualize how the quality of the approximation increases as the iterations proceed. We can find that all the approximate posterior distributions of limit state in the first twenty iterations are far from the real one and they are gradually approaching the real one (as shown in Fig. 6) in the end with the increase of iteration.

Fig. 6 shows all the design points determined by DNF with different criteria and we can find that the design samples computed by NFBD-AG allocates more points near the failure boundaries where $p(\boldsymbol{x})$ has larger PDF value. The distribution of these design points are different from the one computed by NFBD which allocates the more design points around the boundary of failure, and the one computed by NFBD-FG which sparsely allocates the design points around the failure boundary. In theory, NFBD-AG allocates the points with considering the value of $p(\tilde{\boldsymbol{z}}|D)$,i.e., the pdf value of the approximate posterior of limit state and $p(\boldsymbol{x})$.

Fig. 7 shows the estimation of logarithmic failure probability as a function of the number of iterations. We can see that the plot of NFBD-AG has the fastest convergence speed and a more stable performance after 27th iteration. The plot of NFBD has a slower convergence speed and the plot of NFBD-FG has a more unstable representation after convergence, compared with the plot of NFBD-AG. These results

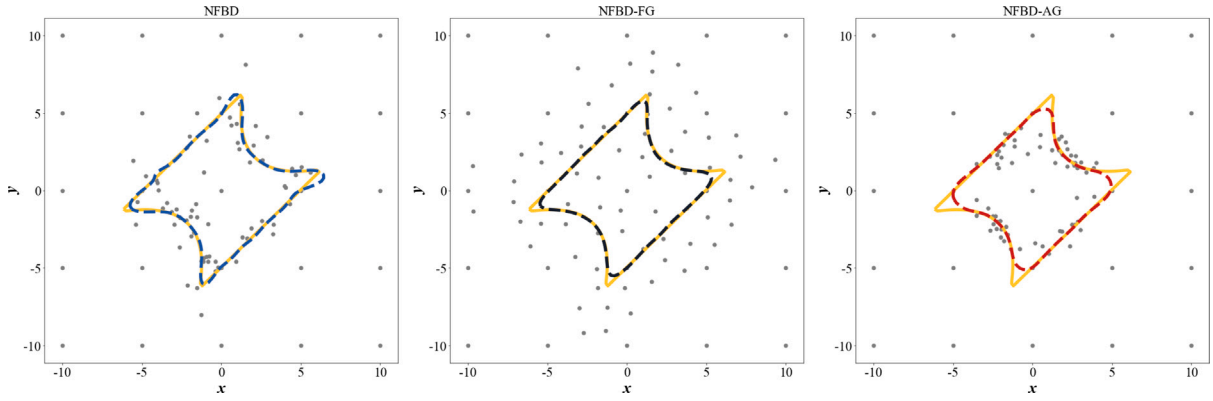

Fig. 3. The gray points represent the design locations determined by NFBD (left), NFBD-FG (middle) and NFBD-AG (right) strategies, respectively. The solid line is the true limit state and the dashed line represents its approximation learned by different strategies.

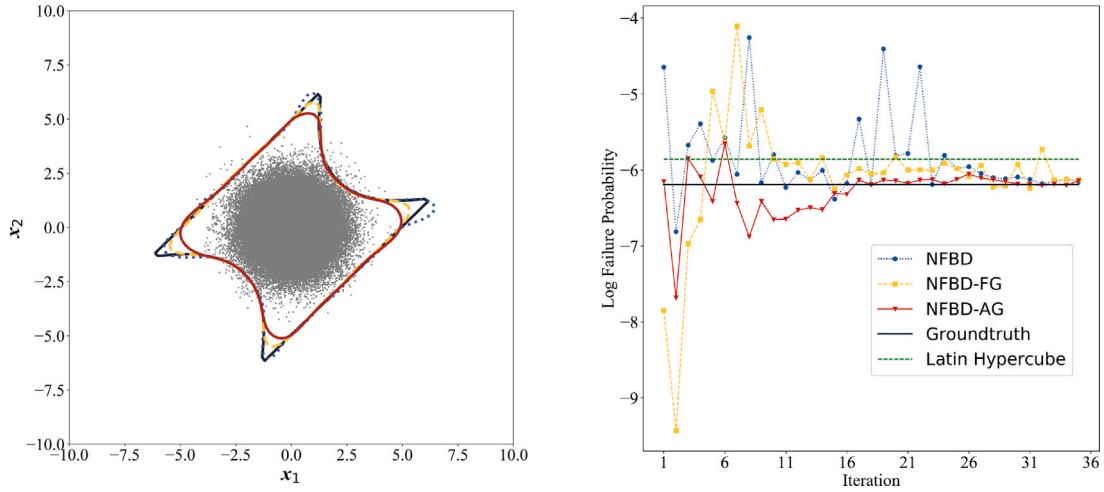

Fig. 4. The true limit state (solid line) and that computed with the NN surrogate (dashed line) are also shown in left figure. The estimation of logarithmic failure probability vs. iterations are shown in right figure.

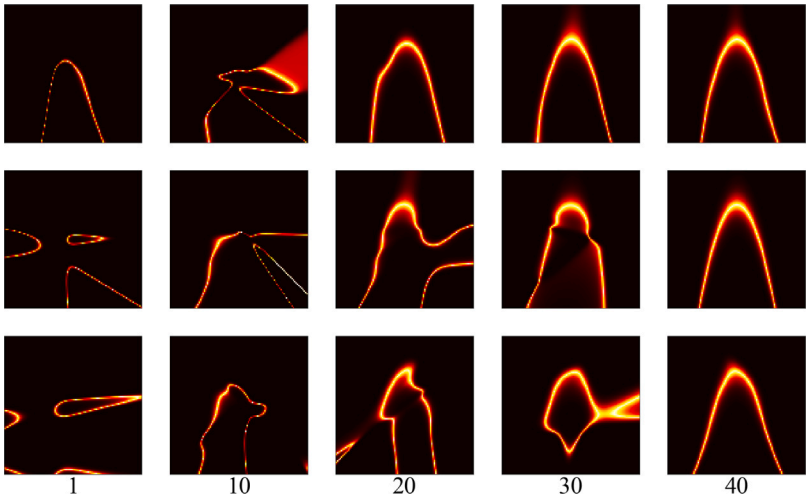

Fig. 5. For iso-probability lines example, the approximations of limit state learned by NFBD (top row), NFBD-FG (middle row) and NFBD-AG (bottom row) criteria, with respect to the first, 10th, 20th, 30th and last iterations, respectively.

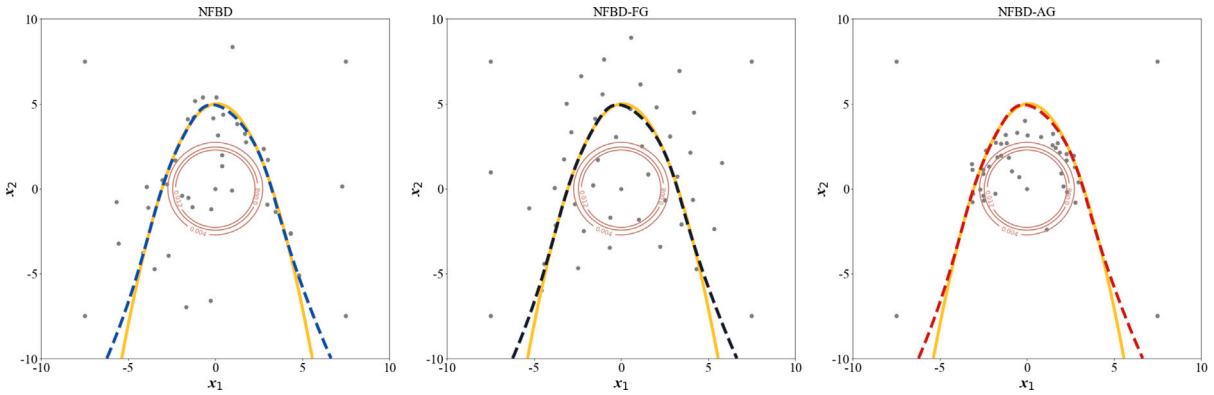

Fig. 6. For iso-probability lines example, the design locations (gray dots) and the approximation of limit state (dashed line) learned by NFBD, NFBD-FG and NFBD-AG, respectively. The contour lines represent the Gaussian distribution of $p(\boldsymbol{x})$.

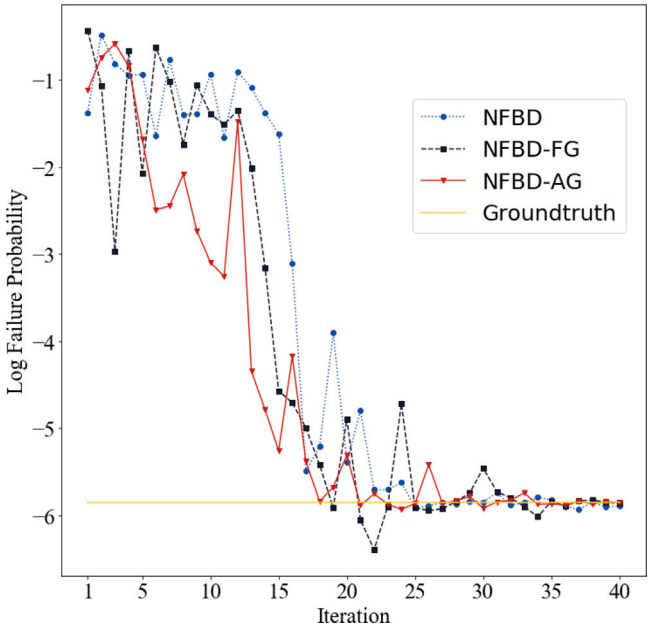

Fig. 7. For iso-probability lines example, the estimation of logarithmic failure probability vs. iterations.

verified the statement that NFBD has a slower convergence speed and NFBD-FG has a faster convergence speed with a large generalization error as mentioned in Section 3.6. In the last iteration, the estimates of failure probability computed by NFBD, NFBD-FG and NFBD-AG converge to 2.76×10^{-3}, 2.84×10^{-3} and 2.83×10^{-3}, respectively, which are more accurate than the one 2.69×10^{-3} computed by Latin hypercube sampling method with the same number of evaluations. The relative error computed by all these estimations are less than ten percent.

4.3. PDE problem cases

Particularly, We pay a attention on a specific scenario, the failure probability estimation of the parametric two-dimensional Darcy Flow equation. The input of this PDE is a function which called drift function and its output is the solution of PDE. The 2-d Darcy Flow equation is defined on the unit box which is the second order, linear, elliptic PDE

$$\begin{aligned} -\nabla\cdot(a(\boldsymbol{\xi};\boldsymbol{x})\nabla u(\boldsymbol{\xi})) &= f(\boldsymbol{\xi}) \quad \boldsymbol{\xi}\in(0,1)^2 \\ u(\boldsymbol{\xi}) &= 0 \quad \boldsymbol{\xi}\in\partial(0,1)^2 \end{aligned} \tag{17}$$

with a Dirichlet boundary where $a\in L^\infty((0,1)^2\times\mathbb{R}^d;\mathbb{R}_+)$ is the diffusion coefficient parametric by $\boldsymbol{x}\in\mathbb{R}^d$ and $f\in L^2((0,1)^2;\mathbb{R})$ is the forcing function which is given $f(\boldsymbol{\xi})=1$.

We are interest in the situation when $a(\boldsymbol{\xi};\boldsymbol{x})$ belong to a exponential random field, i.e. is random function, the solution $u(\boldsymbol{\xi})$ would be random and determined by the random parameter $\boldsymbol{x}$. The coefficients $a(\boldsymbol{\xi};\boldsymbol{x})$ are generated according to $\ln a\sim\mu$ where $\mu=\mathcal{N}(\mu_{\ln a},\sigma_{\ln a})$, where $\mu_{\ln a}=0$ and $\sigma_{\ln a}=((-\triangle+9I)^{-2})$ with zero Neumann boundary conditions on the Laplacian. To represent the random field $\ln a(\boldsymbol{\xi};\boldsymbol{x})$, we apply its Karhunen–Loeve expansion which takes the following form

$$\ln a(\xi)=\mu_{\ln a}+\sigma_{\ln a}\sum_{i=1}^{d}\sqrt{\theta_i}\mathcal{X}_i(\xi)x_i$$

where $\{\theta_i,\mathcal{X}_i\}$ are the eigenpairs of $\rho_{\ln a}$, which are known analytically for the applied exponential correlation model, and $x_i, i=1,\dots,d$ are independent standard normal random variables. The expansion is truncated after 4 terms, i.e., $d=4$ and Fig. 8 shows two samples of $a(\boldsymbol{\xi})$.

We employ this discrete representation of the log-diffusivity and used to define the failure event. Then the limit state function is defined as:

$$D_\epsilon(u(\boldsymbol{x}))=\max(u(\boldsymbol{x}))-\epsilon$$

where the threshold $\epsilon=0.082$.

In this example, we draw 5 initial design points by Latin hypercube sampling (LHS) method and 65 design points are determined by the DNF algorithm with 5 design points in each iteration. The estimations of failure probability computed by different criteria, with respect to different iterations, are shown in Fig. 9. We can find all the estimations of the proposed three criteria successfully converge to the true value 4.24×10^{-3}. Same as the previous two examples, the plot of NFBD slowly and accurately converge to the real value and the plot of NFBD-FG converge fast but not accurately. The estimation error of NFBD-FG are not improved with the increase of iteration. The curve of NFBD-AG quickly and smoothly converges to the real one. In the last iteration, the estimates of failure probability computed by NFBD, NFBD-FG and NFBD-AG converge to 4.22×10^{-3}, 4.88×10^{-3} and 4.88×10^{-3}, respectively, which are more accurate than the one 5.42×10^{-3} computed by Latin hypercube sampling method with the same number of evaluations. The relative error computed by all these estimations are less than ten percent.

5. Discussion and conclusion

The proposed method leverages the power of powerful deep neural network (DNN) models, which has clear advantages in accurately estimating high-dimensional failure probabilities. This approach has great application potential in the field of structural reliability, especially where complex limit states or finite element simulations are expensive. However, it is important to acknowledge that the challenges of overfitting and choosing an appropriate network architecture pose significant obstacles when implementing the proposed method.

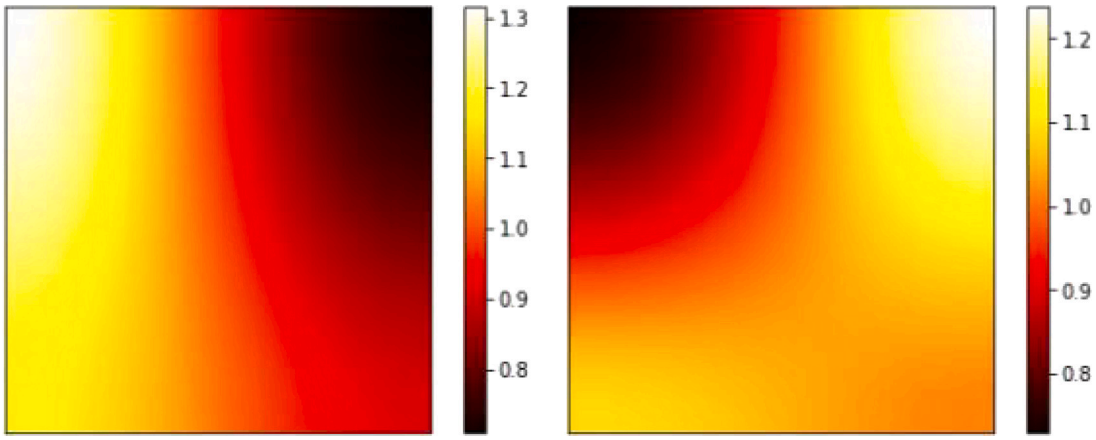

Fig. 8. For PDE problem example, two samples of $a(\xi)$ drawn from μ.

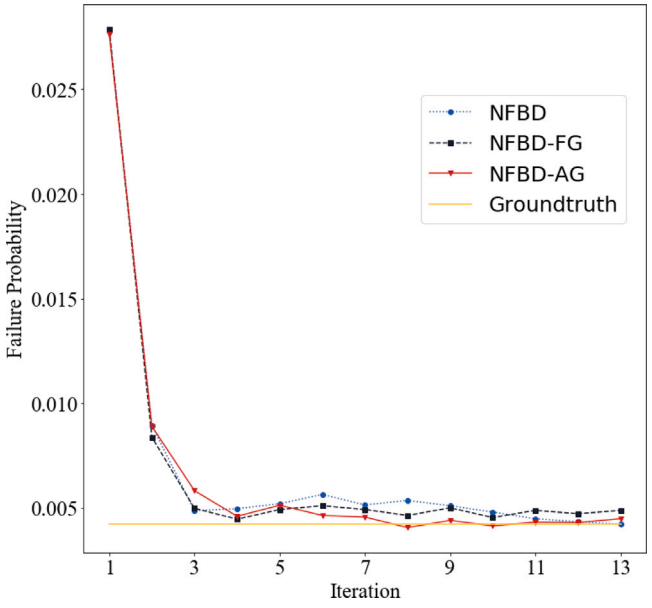

Fig. 9. For PDE problem example, the estimation of failure probability vs. iterations.

In conclusion, we have presented an adaptive design via normalizing flows (DNF) scheme for failure boundary detection and neural network model as the surrogate of computational intensive simulation for failure probability estimations. In particular, the method recasts the failure detection as inferring a contour $g(\boldsymbol{x}) = 0$ with Bayesian methods, and then adaptively determines the design locations for the inference problem. Three normalizing-flows-based design criteria, NFBD, NFBD-FG and NFBD-AG, are also presented. With numerical examples, we demonstrate that the proposed method can efficiently determine design points for failure detection and failure probability estimation.

In contrast to optimization-based optimal design criteria, these methods circumvent the challenges associated with high-dimensional optimization by introducing sampling-based optimal design criteria. Essentially, we address the complexity of estimating failure probabilities by shifting the focus from optimization to fitting: approximating the true failure boundary distribution using normalizing flows. The proposed method holds promise for integration with more efficient sampling techniques such as advanced importance sampling or augmented space techniques, enabling the development of more efficient methods for estimating failure probabilities.

CRediT authorship contribution statement

Tiexin Guo: Conceptualization, Validation, Writing – original draft, Visualization, Writing – review & editing. **Hongji Wang:** Methodology, Software, Writing – original draft, Formal analysis. **Jinglai Li:** Methodology, Writing – review & editing. **Hongqiao Wang:** Supervision, Conceptualization, Writing – original draft, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

Tiexin Guo acknowledges the support of NSFC 11971483. Hongqiao Wang acknowledges the support of NSFC 12101615 and the Natural Science Foundation of Hunan Province, China, under Grant 2022JJ40567. This work was carried out in part using computing resources at the High Performance Computing Center of Central South University.

Appendix A. Fourier neural operator

FNO methodology learns a map between two infinite dimensional spaces from a finite collection of observed input–output pairs. Let $D \subset \mathbb{R}^d$ be a bounded, open set and $\mathcal{A} = \mathcal{A}(D; \mathbb{R}^{d_a})$ and $\mathcal{U} = \mathcal{U}(D; \mathbb{R}^{d_u})$ be separable Banach spaces of function taking values in $\mathbb{R}^{d_a}$ and $\mathbb{R}^{d_u}$ respectively. Furthermore let $g^\dagger : \mathcal{A} \to \mathcal{U}$ be a (typically) non-linear map which arises as the solution operators of parametric PDEs, such as the 2-d Darcy Flow equation,

$$\begin{aligned} -\nabla \cdot (a(\xi)\nabla u(\xi)) &= f(\xi) \quad \xi \in (0,1)^2 \\ u(\xi) &= 0 \quad \xi \in \partial(0,1)^2. \end{aligned} \tag{18}$$

Suppose we have the observations $\{a_j, u_j\}_{j=1}^N$ where $a_j \sim \mu$ is an i.i.d. sequence from the probability measure μ supported on $\mathcal{A}$ and $u_j = g^\dagger(a_j)$ is possibly corrupted with noise. We aim to build an approximation of $g^\dagger$ by constructing a parametric map

$$G : \mathcal{A} \times \Theta \to \mathcal{U} \quad \text{or equivalently,} \quad G_\theta : \mathcal{A} \to \mathcal{U}, \theta \in \Theta \tag{19}$$

for some finite-dimensional parameter space Θ by choosing $\theta^* \in \Theta$ so that $G(\cdot, \theta^*) = G_{\theta^*} \approx g^\dagger$. This is a natural framework for learning in

infinite-dimensions as one could define a cost functional $C : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ and seek a minimizer of the problem

$$\min_{\theta \in \Theta} \mathbb{E}_{a\sim\mu}[C(G(a,\theta), G^{\dagger}(a))] \tag{20}$$

which directly parallels the classical finite-dimensional setting. We will approach this problem in the test-train setting by using a data-driven empirical approximation to the cost used to determine θ and to test the accuracy of the approximation. Details about FNO please refer [69].

Appendix B. Normalizing flows

Here we give a brief introduction of Normalizing Flows and more details please refer [71]. The basic rule for transformation of densities considers an invertible, smooth mapping $f : \mathbb{R}^d \to \mathbb{R}^d$ with inverse $f^{-1} = h$, i.e. the composition $h \circ f(z_0) = z_0$. If we use this mapping to transform a random variable z_0 with distribution $q(z_0)$, the resulting random variable $z = f(z_0)$ has a distribution:

$$q(\boldsymbol{z}) = q(\boldsymbol{z}_0)|\det \frac{\partial f^{-1}}{\partial \boldsymbol{z}}| = q(\boldsymbol{z}_0)|\det \frac{\partial f}{\partial \boldsymbol{z}_0}|^{-1}, \tag{21}$$

where the last equality can be seen by applying the chain rule (inverse function theorem) and is a property of Jacobians of invertible functions. We can construct arbitrarily complex densities by composing several simple maps and successively applying Eq. (21). The density $q_K(\boldsymbol{z}_K)$ obtained by successively transforming a random variable $\boldsymbol{z}_0$ with distribution q_0 through a chain of K transformations f_k is :

$$\boldsymbol{z}_K = f_K \circ \ldots \circ f_2 \circ f_1(\boldsymbol{z}_0) \tag{22}$$

$$\ln q_K(\boldsymbol{z}_K) = \ln q_0(\boldsymbol{z}_0) - \sum_{k=1}^{K} \ln \det |\frac{\partial f_k}{\partial \boldsymbol{z}_k}|, \tag{23}$$

where Eq. (22) will be used as a shorthand for the composition $f_K(f_{K-1}(\ldots f_1(\boldsymbol{z})))$. The path traversed by the random variables $\boldsymbol{z}_k = f_k(\boldsymbol{z}_{k-1})$ with initial distribution $q_0(\boldsymbol{z}_0)$ is called the flow and the path formed by the successive distributions q_k is a normalizing flow.

Consider a general probabilistic model with observation $\boldsymbol{y}$, latent variables $\boldsymbol{z}$ over which we must integrate, and model parameters $\boldsymbol{\theta}$. We introduce an approximate posterior distribution for the latent variable $q_\phi(\boldsymbol{z}|\boldsymbol{y})$ and follow the variational principle [76] to obtain a bound on the marginal likelihood:

$$\ln p_\theta(\boldsymbol{y}) = \ln \int p_\theta(\boldsymbol{y}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z} \tag{24}$$

$$= \ln \int \frac{q_\phi(\boldsymbol{z}|\boldsymbol{y})}{q_\phi(\boldsymbol{z}|\boldsymbol{y})} p_\theta(\boldsymbol{y}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z} \tag{25}$$

$$\geq \mathbb{D}_{KL}[q_\phi(\boldsymbol{z}|\boldsymbol{y}) \| p(\boldsymbol{z})] + \mathbb{E}_q[\ln p_\theta(\boldsymbol{y}|\boldsymbol{z})] = -\mathcal{F}(\boldsymbol{y}), \tag{26}$$

where we used Jensen's inequality to obtain the final equation, $\mathbb{D}_{KL}[\cdot\|\cdot]$ is Kullback–Leibler Divergence, $p_\theta(\boldsymbol{y}|\boldsymbol{z})$ is a likelihood function and $p(\boldsymbol{z})$ is a prior over the latent variables.

If we parameterize the approximate posterior distribution with a flow of length K, $q_\phi(\boldsymbol{z}|\boldsymbol{y}) := q_K(z_K)$, the free energy can be (24) can be written as an expectation over the initial distribution $q_0(\boldsymbol{z}_0)$:

$$\begin{aligned}\mathcal{F}(\boldsymbol{y}) &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{y})}\left[\ln q_\phi(\mathbf{z} \mid \boldsymbol{y}) - \ln p(\boldsymbol{y}, \mathbf{z})\right] \\ &= \mathbb{E}_{q_0(\boldsymbol{z}_0)}\left[\ln q_K\left(\mathbf{z}_K\right) - \ln p\left(\boldsymbol{y}, \mathbf{z}_K\right)\right] \\ &= \mathbb{E}_{q_0(\boldsymbol{z}_0)}\left[\ln q_0\left(\mathbf{z}_0\right)\right] - \mathbb{E}_{q_0(z_0)}\left[\ln p\left(\boldsymbol{y}, \mathbf{z}_K\right)\right]. \\ &- \mathbb{E}_{q_0(\boldsymbol{z}_0)}\left[\sum_{k=1}^{K} \ln \det |\frac{\partial f_{k,\phi}}{\partial \boldsymbol{z}_k}|\right]\end{aligned} \tag{27}$$

The neural network based transform function $f_{k,\phi}$ can be obtained by maximizing the free energy in Eq. (27).

References

[1] Leichsenring F, Jenkel C, Graf W, Kaliske M. Numerical simulation of wooden structures with polymorphic uncertainty in material properties. Int J Reliab Saf 2018;12:24–45.

[2] Goubergrits L, Hellmeier F, Bruening J, Spuler A, Hege H-C, Voss S, Janiga G, Saalfeld S, Beuing O, Berg P. (Match): uncertainty quantification of geometric rupture risk parameters. Biomed Eng Online 2018;18(2019):1–16.

[3] Xiu D, Karniadakis GE. Supersensitivity due to uncertain boundary conditions. Internat J Numer Methods Engrg 2004;61:2114–38.

[4] Kogiso N, Ahn W, Nishiwaki S, Izui K, Yoshimura M. Robust topology optimization for compliant mechanisms considering uncertainty of applied loads. J Adv Mech Des Syst Manuf 2008;2:96–107.

[5] Silva F, Lambe TW, Marr WA. Probability and risk of slope failure. J Geotech Geoenviron Eng 2008;134:1691–9.

[6] Rockafellar RT, Royset JO. On buffered failure probability in design and optimization of structures. Reliab Eng Syst Saf 2010;95:499–510.

[7] Enevoldsen I, Sørensen JD. Reliability-based optimization in structural engineering. Struct Saf 1994;15:169–96.

[8] Schuëller G, Pradlwarter H, Koutsourelakis P. A critical appraisal of reliability estimation procedures for high dimensions. Probab Eng Mech 2004;19:463–74. http://dx.doi.org/10.1016/j.probengmech.2004.05.004, URL https://www.sciencedirect.com/science/article/pii/S026689200400044X.

[9] Mooney CZ. Monte Carlo simulation, Vol. 116. Sage; 1997.

[10] Au S. Reliability-based design sensitivity by efficient simulation. Comput Struct 2005;83:1048–61.

[11] Ching J, Hsieh Y-H. Local estimation of failure probability function and its confidence interval with maximum entropy principle. Probab Eng Mech 2007;22:39–49.

[12] Taflanidis AA, Beck JL. An efficient framework for optimal robust stochastic system design using stochastic simulation. Comput Methods Appl Mech Engrg 2008;198:88–101.

[13] Liu W-S, Cheung SH. Reliability based design optimization with approximate failure probability function in partitioned design space. Reliab Eng Syst Saf 2017;167:602–11.

[14] Zou T, Mahadevan S. A direct decoupling approach for efficient reliability-based design optimization. Struct Multidiscip Optim 2006;31:190–200.

[15] Yuan X. Local estimation of failure probability function by weighted approach. Probab Eng Mech 2013;34:1–11.

[16] Yuan X, Lu Z. Efficient approach for reliability-based optimization based on weighted importance sampling approach. Reliab Eng Syst Saf 2014;132:107–14.

[17] Yuan X, Wang S, Valdebenito MA, Faes MG, Beer M. Sample regeneration algorithm for structural failure probability function estimation. Probab Eng Mech 2023;71:103387.

[18] Cheng K, Lu Z. Adaptive bayesian support vector regression model for structural reliability analysis. Reliab Eng Syst Saf 2021;206:107286.

[19] Faravelli L. Response-surface approach for reliability analysis. J Eng Mech 1989;115:2763–81.

[20] Gayton N, Bourinet JM, Lemaire M. Cq2rs: a new statistical approach to the response surface method for reliability analysis. Struct Saf 2003;25:99–121.

[21] Oakley J, O'Hagan A. Bayesian inference for the uncertainty distribution of computer model outputs. Biometrika 2002;89:769–84.

[22] Saraygord Afshari S, Enayatollahi F, Xu X, Liang X. Machine learning-based methods in structural reliability analysis: A review. Reliab Eng Syst Saf 2022;219:108223. http://dx.doi.org/10.1016/j.ress.2021.108223, URL https://www.sciencedirect.com/science/article/pii/S0951832021007018.

[23] Luo C, Keshtegar B, Zhu SP, Taylan O, Niu X-P. Hybrid enhanced monte carlo simulation coupled with advanced machine learning approach for accurate and efficient structural reliability analysis. Comput Methods Appl Mech Engrg 2022;388:114218.

[24] Wang J, Sun Z, Cao R. An efficient and robust kriging-based method for system reliability analysis. Reliab Eng Syst Saf 2021;216:107953.

[25] Li P, Wang Y. An active learning reliability analysis method using adaptive bayesian compressive sensing and monte carlo simulation (abcs-mcs). Reliab Eng Syst Saf 2022;221:108377.

[26] Xiong Y, Sampath S. A fast-convergence algorithm for reliability analysis based on the ak-mcs. Reliab Eng Syst Saf 2021;213:107693.

[27] Hong L, Li H, Fu J, Li J, Peng K. Hybrid active learning method for non-probabilistic reliability analysis with multi-super-ellipsoidal model. Reliab Eng Syst Saf 2022;222:108414.

[28] Huang S-Y, Zhang S-H, Liu L-L. A new active learning kriging metamodel for structural system reliability analysis with multiple failure modes. Reliab Eng Syst Saf 2022;228:108761.

[29] Zhang K, Chen N, Zeng P, Liu J, Beer M. An efficient reliability analysis method for structures with hybrid time-dependent uncertainty. Reliab Eng Syst Saf 2022;228:108794.

[30] Dehghani NL, Zamanian S, Shafieezadeh A. Adaptive network reliability analysis: Methodology and applications to power grid. Reliab Eng Syst Saf 2021;216:107973.

[31] Liu W, Li A, Fang W, Love PE, Hartmann T, Luo H. A hybrid data-driven model for geotechnical reliability analysis. Reliab Eng Syst Saf 2023;231:108985.
[32] Liu W, Wang Z, Liu X, Zeng N, Liu Y, Alsaadi FE. A survey of deep neural network architectures and their applications. Neurocomputing 2017;234:11–26.
[33] Wang H, Guan Y, Reich B. Nearest-neighbor neural networks for geostatistics. In: 2019 international conference on data mining workshops (ICDMW). IEEE; 2019, p. 196–205.
[34] Abd Elaziz M, Dahou A, Abualigah L, Yu L, Alshinwan M, Khasawneh AM, Lu S. Advanced metaheuristic optimization techniques in applications of deep neural networks: a review. Neural Comput Appl 2021;33:14079–99.
[35] Tripathy RK, Bilionis I. Deep uq: Learning deep neural network surrogate models for high dimensional uncertainty quantification. J Comput Phys 2018;375:565–88.
[36] Kabir HD, Khosravi A, Hosen MA, Nahavandi S. Neural network-based uncertainty quantification: A survey of methodologies and applications. IEEE access 2018;6:36218–34.
[37] Karniadakis GE, Kevrekidis IG, Lu L, Perdikaris P, Wang S, Yang L. Physics-informed machine learning. Nat Rev Phys 2021;3:422–40.
[38] Deshpande S, Lengiewicz J, Bordas S. Magnet: A graph u-net architecture for mesh-based simulations. 2022, arXiv preprint arXiv:2211.00713.
[39] Krokos V, Bordas S, Kerfriden P. A graph-based probabilistic geometric deep learning framework with online enforcement of physical constraints to predict the criticality of defects in porous materials. 2022, Available at SSRN 4384127.
[40] Krokos V, Bui Xuan V, Bordas SP, Young P, Kerfriden P. A bayesian multiscale cnn framework to predict local stress fields in structures with microscale features. Comput Mech 2022;69:733–66.
[41] Chen L, Cheng R, Li S, Lian H, Zheng C, Bordas SP. A sample-efficient deep learning method for multivariate uncertainty qualification of acoustic–vibration interaction problems. Comput Methods Appl Mech Engrg 2022;393:114784.
[42] Mazier A, Lavigne T, Lengiewicz J, Deshpande S, Urcun S, Bordas S. Towards real-time patient-specific breast simulations: from full-field information to surrogate model. 2022.
[43] Rappel H, Beex LAA, Bordas SPA. Bayesian inference to identify parameters in viscoelasticity. Mech Time-Dependent Mater 2018;22:221–58.
[44] Rappel H, Beex LAA, Hale JS, Noels L, Bordas SPA. A tutorial on bayesian inference to identify material parameters in solid mechanics. Arch Comput Methods Eng 2020;27:361–85.
[45] Rappel H, Beex LAA, Noels L, Bordas SPA. Identifying elastoplastic parameters with bayes' theorem considering output error, input error and model uncertainty. Probab Eng Mech 2019;55:28–41.
[46] Rappel H, Beex LAA. Estimating fibres' material parameter distributions from limited data with the help of bayesian inference. Eur J Mech A Solids 2019;75:169–96.
[47] Mohamedou M, Zulueta K, Chung CN, Rappel H, Beex L, Adam L, Arriaga A, Major Z, Wu L, Noels L. Bayesian identification of mean-field homogenization model parameters and uncertain matrix behavior in non-aligned short fiber composites. Compos Struct 2019;220:64–80.
[48] Rappel H, Wu L, Noels L, Beex LA. A bayesian framework to identify random parameter fields based on the copula theorem and gaussian fields: Application to polycrystalline materials. J Appl Mech 2019;86:121009.
[49] Peralta P, Ruiz RO, Rappel H, Bordas SP. Electromechanical properties identification for groups of piezoelectric energy harvester based on bayesian inference. Mech Syst Signal Process 2022;162:108034.
[50] Echard B, Gayton N, Lemaire M. Ak-mcs: an active learning reliability method combining kriging and monte carlo simulation. Struct Saf 2011;33:145–54.
[51] Bichon BJ, Eldred MS, Swiler LP, Mahadevan S, McFarland JM. Efficient global reliability analysis for nonlinear implicit performance functions. AIAA J 2008;46:2459–68.
[52] Yuan X, Qian Y, Chen J, Faes MG, Valdebenito MA, Beer M. Global failure probability function estimation based on an adaptive strategy and combination algorithm. Reliab Eng Syst Saf 2023;231:108937.
[53] Bect J, Ginsbourger D, Li L, Picheny V, Vazquez E. Sequential design of computer experiments for the estimation of a probability of failure. Stat Comput 2012;22:773–93.
[54] Chevalier C, Bect J, Ginsbourger D, Vazquez E, Picheny V, Richet Y. Fast parallel kriging-based stepwise uncertainty reduction with application to the identification of an excursion set. Technometrics 2014;56:455–65.
[55] Wang H, Lin G, Li J. Gaussian process surrogates for failure detection: A bayesian experimental design approach. J Comput Phys 2016;313:247–59.
[56] Renganathan A, Rao V, Navon I. Multifidelity gaussian processes for failure boundary and probability estimation. In: AIAA SCITECH 2022 forum. 2022, p. 0390.
[57] Au S-K, Beck JL. Estimation of small failure probabilities in high dimensions by subset simulation. Probab Eng Mech 2001;16:263–77.
[58] Engelund S, Rackwitz R. A benchmark study on importance sampling techniques in structural reliability. Struct Saf 1993;12:255–76.
[59] Wang Y, Xie B, Shiyuan E. Adaptive relevance vector machine combined with markov-chain-based importance sampling for reliability analysis. Reliab Eng Syst Saf 2022;220:108287.
[60] Tabandeh A, Jia G, Gardoni P. A review and assessment of importance sampling methods for reliability analysis. Struct Saf 2022;97:102216.
[61] Rubinstein RY, Kroese DP. Applications of ce to machine learning. In: The cross-entropy method. Springer; 2004, p. 251–70.
[62] Wang H, Zhou X. A cross-entropy scheme for mixtures. ACM Trans Model Comput Simul (TOMACS) 2015;25:1–20.
[63] Li J, Li J, Xiu D. An efficient surrogate-based method for computing rare failure probability. J Comput Phys 2011;230:8683–97.
[64] Li L, Bect J, Vazquez E. Bayesian subset simulation: a kriging-based subset simulation algorithm for the estimation of small probabilities of failure. 2012, arXiv preprint arXiv:1207.1963.
[65] Dubourg V, Sudret B, Deheeger F. Metamodel-based importance sampling for structural reliability analysis. Probab Eng Mech 2013;33:47–57.
[66] Zou F, Shen L, Jie Z, Zhang W, Liu W. A sufficient condition for convergences of adam and rmsprop. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 11127–35.
[67] Jumper J, Evans R, Pritzel A, Green T, Figurnov M, Ronneberger O, Tunyasuvunakool K, Bates R, dek A, Potapenko A, et al. Highly accurate protein structure prediction with alphafold. Nature 2021;596:583–9.
[68] Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 2019;378:686–707.
[69] Li Z, Kovachki N, Azizzadenesheli K, Liu B, Bhattacharya K, Stuart A, Anandkumar A. Fourier neural operator for parametric partial differential equations. 2020, arXiv preprint arXiv:2010.08895.
[70] Lydia A, Francis S. Adagrad—an optimizer for stochastic gradient descent. Int J Inf Comput Sci 2019;6:566–8.
[71] Rezende D, Mohamed S. Variational inference with normalizing flows. In: International conference on machine learning. PMLR; 2015, p. 1530–8.
[72] Papamakarios G, Sterratt D, Murray I. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In: The 22nd international conference on artificial intelligence and statistics. PMLR; 2019, p. 837–48.
[73] Chen Y, Gutmann MU. Adaptive Gaussian copula ABC. In: The 22nd international conference on artificial intelligence and statistics. PMLR; 2019, p. 1584–92.
[74] Blum MG, François O. Non-linear regression models for Approximate Bayesian Computation. Stat Comput 2010;20:63–73.
[75] Wang M, Ma C. Generalization error bounds for deep neural networks trained by sgd. 2022, arXiv preprint arXiv:2206.03299.
[76] Jordan MI, Ghahramani Z, Jaakkola TS, Saul LK. An introduction to variational methods for graphical models. Mach Learn 1999;37:183–233.