A hierarchical neural hybrid method for failure probability estimation

Li, Ke; Tang, Kejun; Li, Jinglai; Wu, Tianfan; Liao, Qifeng

DOI:
10.1109/access.2019.2934980

Document Version
Publisher's PDF, also known as Version of record

Citation for published version (Harvard):
Li, K, Tang, K, Li, J, Wu, T & Liao, Q 2019, 'A hierarchical neural hybrid method for failure probability estimation', *IEEE Access*, vol. 7, pp. 112087 - 112096. https://doi.org/10.1109/access.2019.2934980

Received July 24, 2019, accepted July 31, 2019, date of publication August 15, 2019, date of current version August 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2934980

A Hierarchical Neural Hybrid Method for Failure Probability Estimation

KE LI[1], KEJUN TANG[1], JINGLAI LI[2], TIANFAN WU[3], AND QIFENG LIAO[1]

[1]School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China
[2]Department of Mathematical Sciences, University of Liverpool, Liverpool L69 3BX, U.K.
[3]Viterbi School of Engineering, University of Southern California, Los Angeles, CA 90007, USA

Corresponding author: Qifeng Liao (liaoqf@shanghaitech.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 11601329 and Grant 11771289, and in part by the Science Challenge Project under Grant TZ2018001.

ABSTRACT Failure probability evaluation for complex physical and engineering systems governed by partial differential equations (PDEs) are computationally intensive, especially when high-dimensional random parameters are involved. Since standard numerical schemes for solving these complex PDEs are expensive, traditional Monte Carlo methods which require repeatedly solving PDEs are infeasible. Alternative approaches which are typically the surrogate based methods suffer from the so-called ''curse of dimensionality'', which limits their application to problems with high-dimensional parameters. For this purpose, we develop a novel hierarchical neural hybrid (HNH) method to efficiently compute failure probabilities of these challenging high-dimensional problems. Especially, multifidelity surrogates are constructed based on neural networks with different levels of layers, such that expensive highfidelity surrogates are adapted only when the parameters are in the suspicious domain. The efficiency of our new HNH method is theoretically analyzed and is demonstrated with numerical experiments. From numerical results, we show that to achieve an accuracy in estimating the rare failure probability (e.g., 10^{-5}), the traditional Monte Carlo method needs to solve PDEs more than a million times, while our HNH only requires solving them a few thousand times.

INDEX TERMS Hierarchical method, hybrid method, PDEs, rare events.

I. INTRODUCTION

Due to lack of knowledge or measurement of realistic model parameters, modern complex physical and engineering systems are often modeled by partial differential equations (PDEs) with high-dimensional random parameters. For example, groundwater flow problems are modeled by stochastic diffusion equations, and acoustic scattering problems are modeled by Helmholtz equations with random inputs. When conducting risk management, it is essential to compute failure probabilities of these stochastic PDE models. A standard method to compute the failure probability is the traditional Monte Carlo sampling method [1]. However, this method requires repeatedly solving complex PDEs to generate a large number samples to capture failure probabilities associated with rare events. There are two main computational challenges: first, it is expensive to solve each complex PDE using standard numerical schemes (e.g., finite elements [2]); second, the complex PDEs need to be repeatedly solved many times for computing failure probabilities.

As alternatives, gradient-based methods and simulation-based methods are developed and are briefly reviewed as follows. Classical gradient-based methods are the first-order reliability method (FORM) [3], second-order reliability method (SORM) [4] and a statistical technical response surface method (RSM) [5], [6]. Moreover, Simulation-based method relies on Monte Carlo (MC) sampling and the failure probability can be approximated by the ratio between number of failure samples and the total amount. To deal with rare events, conditional simulation [7] and importance sampling (IS) [8] are efficient improvements of the traditional Monte Carlo method.

However, the above methods suffer from the ''curse of dimensionality'', which limits their applications for complex problems with high-dimensional inputs.

In this work, we propose a novel hierarchical neural hybrid (HNH) method to resolve the challenges discussed above. Our method combines the advantages of MC,

The associate editor coordinating the review of this article and approving it for publication was Nagarajan Raghavan.

RSM and neural network to solve high-dimensional failure probability problems. The main advantages of HNH are three-fold. First, HNH is a universal solver based on neural networks which can efficiently approximate any complex system including PDEs with any given accuracy [9]. Second, multifidelity surrogates are constructed and expensive fine-fidelity surrogates are adopted only for samples close to the suspicious domain, which results in an overall efficient computational procedure. Finally, only a few samples generated through solving given PDEs with standard numerical schemes are required to construct the surrogates and to modify the failure probability estimation.

Our Contributions are as Follows:

- Our method is based on a combination of neural networks and the hybrid method. The hybrid method is an extremely effective approach which can capture failure probability using a few samples without losing accuracy. And our new method utilizes the feature of neural network as a universal approximation to solve complex systems with high-dimensional inputs.
- We propose a novel hierarchical neural hybrid (HNH) method. Given the fact that a sufficient deep neural network surrogate is still expensive to approximate complex systems, we employ multifidelity models to replace the single fine-fidelity deep model. Our method uses coarse-fidelity surrogates as a preconditioning scheme and uses fine-fidelity surrogates to correct the estimation. Our HNH method can significantly accelerate the computational procedure for failure probability estimation without sacrificing accuracy.

Paper Organization: The failure probability, the hybrid method and multilayer neural network are briefly reviewed in Section 2. Our method is presented in Section 3, where the rigorous error analysis for HNH is conducted. In Section 4, the efficiency of HNH is demonstrated with a high-dimensional structural safety problem, stochastic diffusion equations and Helmholtz equations. Finally some concluding remarks are offered in Section 5.

II. PRELIMINARIES

A. FAILURE PROBABILITY

In practical engineering problems, the various uncertainties ultimately affects the structural safety. Reliability analysis has become increasingly attracted in engineering analysis, and it can measure the structural safety by considering these uncertainties [10], [11]. In a general setting, let Z be a n_z-dimensional random vector $Z = (Z_1, Z_2, \ldots, Z_{n_z}) : \Omega \to R^{n_z}$, where $F_Z(z) = \text{Prob}(Z \le z)$ is the distribution function, Ω is the probability space. Given a (scalar) limit state function $g(Z)$, $g(Z) < 0$ defines a failure domain Ω_f and $g(Z) \ge 0$ defines safe domain. For some specific setting in section 4, we give additional definitions of Ω_f. The failure probability P_f is defined

$$P_f = \text{Prob}(Z \in \Omega_f) = \int_{\Omega_f} dF_Z(z) = \int \chi_{\Omega_f}(z) dF_Z(z), \tag{1}$$

where χ means the characteristic function

$$\chi_{\Omega_f}(z) = \begin{cases} 1 & \text{if } z \in \Omega_f, \\ 0 & \text{if } z \notin \Omega_f. \end{cases} \tag{2}$$

B. HYBRID METHOD

Hybrid method [12] can enhance the performance of Monte Carlo sampling with surrogate models. Instead of computing (1) directly, we employ surrogate model $\hat{g}$ to evaluate the failure probability

$$\hat{P_f} = \int \chi_{\{\hat{g}(z)<0\}}(z) q(z) dz, \tag{3}$$

where $q(z)$ denotes an arbitrary distribution of variable z. Specifically, we utilize MC to evaluate (3) by generating samples $\{z^{(i)}\}_{i=1}^M$ from $q(z)$

$$\hat{P}_f^{mc} = \frac{1}{M} \sum_{i=1}^{M} \chi_{\{\hat{g}(z)<0\}}(z^{(i)}). \tag{4}$$

where M denotes the number of samples.

Following the idea of the response surface method (RSM) [13], [14] and design from [15], the procedure of failure probability estimation can be specified as follows.

Definition 1: For a given real parameter γ, a limit state function g and its surrogate model $\hat{g}$, the failure probability of $g(Z) < 0$ can be represented as

$$\begin{aligned} \text{Prob}(g<0) &\approx \text{Prob}(\gamma; g, \hat{g}) \\ &= \text{Prob}(\{\hat{g} < -\gamma\}) + \text{Prob}(\{|\hat{g}| \le \gamma\} \cap \{g < 0\}), \end{aligned} \tag{5}$$

where $(-\gamma, \gamma)$ is called the suspicious domain.
With MC method, we have

$$\begin{aligned} \widehat{\text{Prob}}(g < 0) \approx \widehat{\text{Prob}}(\gamma; g, \hat{g}) &= \frac{1}{M} \sum_{i=1}^{M} (\chi^{(h)}_{\{\hat{g}<0\}}(z^{(i)})) \\ &= \frac{1}{M} \sum_{i=1}^{M} [\chi^{(h)}_{\{\hat{g}<-\gamma\}}(z^{(i)}) \\ &\quad + \chi^{(h)}_{\{|\hat{g}|\le\gamma\}}(z^{(i)}) \cdot \chi_{\{g<0\}}(z^{(i)})]; \end{aligned} \tag{6}$$

where γ in the equation is an arbitrary positive number, which denotes the range of re-verifying domain. It is clear that the computational cost will increase with γ growing. So choosing a small enough γ without sacrificing accuracy is a traditional difficulty.

In this paper, we employ neural networks as the surrogate model $\hat{g}$, and we call the method as a direct combination of hybrid method and neural network as neural hybrid (NH) method.

C. MULTILAYER NEURAL NETWORK SURROGATE

We employ full connected multilayer perceptron as the basic of our surrogate, which is a feed-forward type network with only adjoining layers connected. Network generally consists

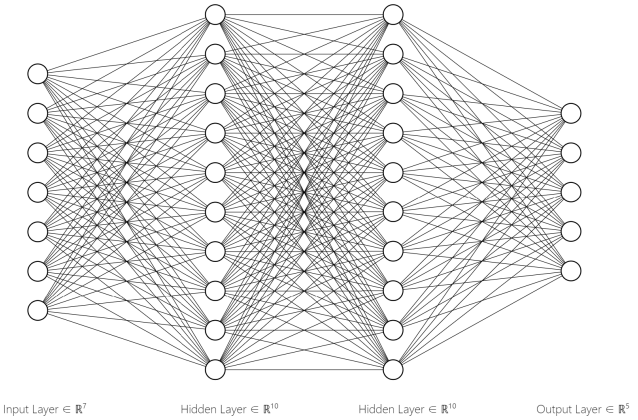


FIGURE 1. Schematic representation of a multilayer neural network which contains 7 input units, 5 output units and 2 hidden layers.

of input, hidden and output layers, see Figure 1. For illustration only, the network depicted consists of 2 layers with 10 neurons in each layer and σ denotes an element-wise operator

$$\sigma(x) = (\phi(x_1), \phi(x_2), \ldots, \phi(x_{10})), \tag{7}$$

where ϕ is called activation function. We recall the network space with given data according to [16]

$$\mathfrak{N}_i^n(\sigma) = \left\{ h(x) : \mathbb{R}^k \to \mathbb{R} | h(x) = \sum_{j=1}^{n} \beta_j \sigma\left(\alpha_j^T x - \theta_j\right) \right\} \tag{8}$$

where σ is any activation function, $\mathbf{x} \in \mathbb{R}^k$ is one set of observed data, $\beta \in \mathbb{R}^n$, $\alpha \in \mathbb{R}^{k\times n}$ and $\theta \in \mathbb{R}^{k\times n}$ denote coefficients of networks. The learning rule called back propagation algorithm [17] is widely used in neural network. Multilayer perceptron is a mapping from $n-$dimension Euclidean space to $m-$dimension Euclidean space if there are n input units and m output units. We can also choose residual neural networks [18] as surrogates for efficiency, but we just present our idea of this work using the fully-connected neural network for simplicity.

Via the comparison of the accuracy using neural networks with different layers in [18], the results of deeper neural networks are more accurate after sufficient training procedure. Therefore, we call deep networks fine-fidelity and shadow networks coarse fidelity in the following. In order to reduce the risk of overfitting, we employ cross validation and dropout [19].

III. HIERARCHICAL NEURAL HYBRID METHOD

In the former part, we introduced the hybrid method to speed up solving complex system by using an accurate enough surrogate. However, the computational cost of using a fine-fidelity model is also expensive, though it is much cheaper than the original system. Here we propose a hierarchical neural hybrid (HNH) method which constructs a hierarchy surrogate model to accelerate the standard NH method. The hierarchical surrogate models here are neural networks with different layers. Let $g^{(1)}, g^{(2)}, \cdots, g^{(L)}$ denote L surrogates, with $\ell = 1$ being the coarsest and $\ell = L$ being the finest, each $g^{(\ell)}(\ell = 1, \ldots, L)$ is a feedforward neural network with P_ℓ layers and can approximate the limit state function g well, illustrated in Figure 2. It should be noticed that training data are the same for constructing hierarchical models off-line. Compared with NH method with single fine-fidelity, the hierarchical surrogate models can reduce the running time.

The idea of our method can be illustrated in Figure 3. Discrete solutions from true PDE models are referred as groundtruth, but the computational procedure can be extremely expensive. In former part we introduce that solutions from deeper networks are more accurate but more expensive. If we use multifidelity to combine different networks, the cost can be reduced and the accuracy can be kept. Our hierarchical neural hybrid method uses the true model to improve the accuracy and the cost does not increase much, because our method only needs to solve the discrete PDEs a few times.

A. THE HNH METHOD

Hybrid method combines the robustness of MC and the feasibility of RSM. However, the computation cost is still expensive if an enough accurate surrogate model is repeatedly applied. Following the idea of multifidelity approaches [20]–[23], we suppose that the cost can be reduced without losing accuracy by using a hierarchy surrogate model in the neural hybrid method. Our HNH method iterates through the level $\ell = 0, \cdots, L$, where with $\ell > 0$ increasing the accuracy increases except $g^{(0)} = g$ means the original system. First, the HNH method evaluates the state function at Z by using $g^{(1)}$ instead of neural hybrid method using $g^{(L)}$, and then sorts $\{|g_1(z^{(i)})|\}_{i=1}^M$ in an ascending order. Second, the HNH method divides the data set into L parts according to the value of $\{|g_1(z^{(i)})|\}_{i=1}^M$. For part $1, \cdots, L-1$, HNH employs $g^{(L)}, \cdots, g^{(2)}$ to modify the prediction of failure with a threshold ϵ_{opt} to stop the modification. The modification procedure is presented in Algorithm 1. Third, our HNH method uses the modified results of prediction to run the iterative hybrid method introduced in III-B and Algorithm 2. A detailed illustration of HNH method can be found in Figure 4.

B. ITERATION ALGORITHM OF HNH METHOD

Here, we provide our integrated iteration algorithm for computing failure probability of high-dimensional problems through HNH method.

The hybrid method is applied which utilizes surrogate models to enhance the performance of Monte Carlo sampling. We present the probability formula defined in equation (5) and (6).

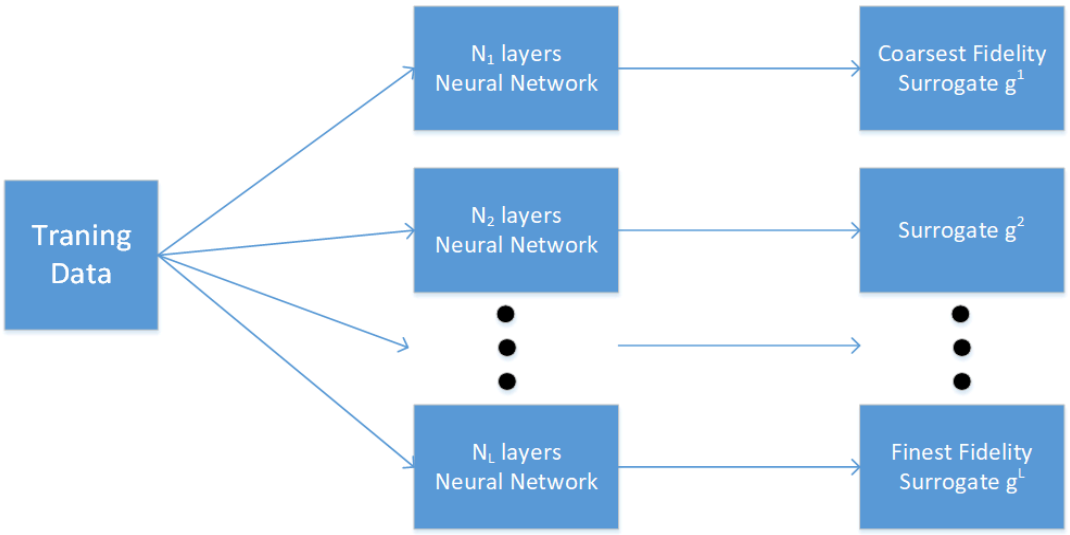


FIGURE 2. Illustration of hierarchical training procedure.

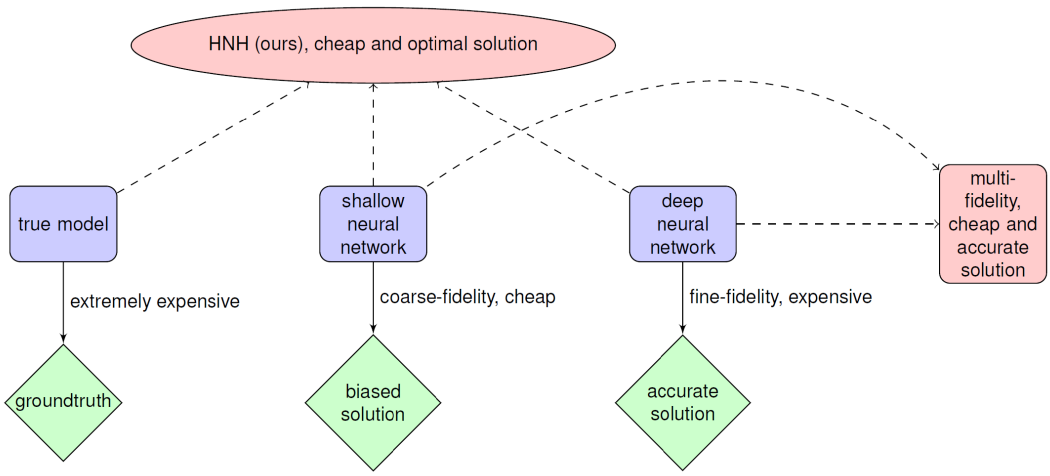


FIGURE 3. The idea of our HNH method.

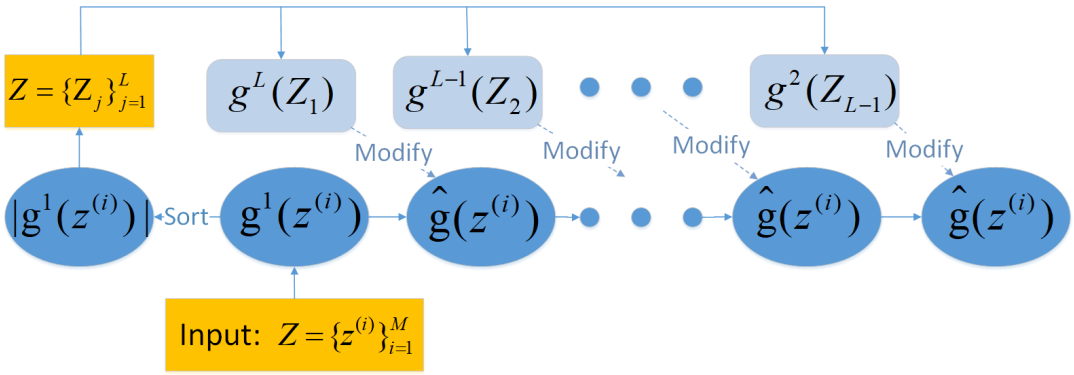


FIGURE 4. Illustration of hierarchical neural hybrid method.

Hybrid method with γ can enhance the performance of MC, but it is often hard to get the threshold γ. So we use iterative scheme to avoid choosing γ.

Let δM be the number of samples generated from g in each iteration, ϵ_{opt} be a given tolerance. The formal description of HNH method is presented in Algorithm 2.

Algorithm 1 Modification Procedure of Hierarchical Neural Hybrid Method

Input: $g^{(1)}, g^{(2)}, \cdots, g^{(L)}, Z = \{z^{(i)}\}_{i=1}^{M}, \eta$.
Evaluate $g^{(1)}(z^{(1)}), g^{(1)}(z^{(2)}), \cdots, g^{(1)}(z^{(M)})$ and get label $\{\chi_{\{\hat{g}<0\}}(z^{(i)})\}_{i=1}^{M} = \{\chi_{\{g_1<0\}}(z^{(i)})\}_{i=1}^{M}$.
Sort $\{z^{(i)}\}_{i=1}^{M}$ such that the value $\{|g^{(1)}(z^{(i)})|\}_{i=1}^{M}$ is in an ascending order, and the sorted order is denoted as $\{z^{(j)}\}_{j=1}^{M}$.
Split dataset $Z = \{z^{(j)}\}_{j=1}^{M}$ into L parts and each part has the same number of samples.
for $i = 1$ **to** $L-1$ **do**
  $\epsilon = 0$.
  for $j = \frac{(i-1)M}{L} + 1$ **to** $\frac{iM}{L}$ **do**
    $\Delta = [-\chi_{\{g^{(1)}<0\}} z^{(j)} + \chi_{\{g^{(-i+L-1)}<0\}} z^{(j)}]$.
    $\chi_{\{\hat{g}<0\}}(z^{(j)}) = \chi_{\{g^{(1)}<0\}}(z^{(j)}) + \Delta$.
    $\epsilon = \epsilon + \frac{L}{M}[-\chi_{\{g^{(1)}<0\}} z^{(j)} + \chi_{\{g^{(-i+L-1)}<0\}} z^{(j)}]$.
  end for
  if $\epsilon < \eta$ **then**
    break
  end if
end for
Return: Label $\{\chi_{\{\hat{g}<0\}}(z^{(j)})\}_{j=1}^{M}$.

Algorithm 2 Iteration Algorithm of Hierarchical Neural Hybrid Method

Input: $g^{(1)}, \cdots, g^{(L)}, Z = \{z^{(i)}\}_{i=1}^{M}, \eta, \delta M, \epsilon_{opt}$.
Initialize: $Z^{(0)} = \emptyset, k = 0, \epsilon = 10\epsilon_{opt}$.
Obtain $\{\chi_{\{\hat{g}<0\}}(z^{(j)})\}_{j=1}^{M}$ using algorithm 1.
$P_f^{(k)} = \frac{1}{M}\sum_{j=1}^{M} \chi_{\{\hat{g}<0\}}(z^{(j)})$.
while $\epsilon > \epsilon_{opt}$ **do**
  $k = k + 1$.
  $\delta Z^{(k)} = \{z^{(j)}\}_{j=(k-1)\delta M+1}^{k\delta M}$.
  $Z^{(k)} = Z^{(k-1)} \cup \delta Z^{(k)}$.
  $\delta P = \frac{1}{M} \sum_{z^{(j)} \in \delta Z^{(k)}} [-\chi_{\{\hat{g}<0\}}(z^{(j)}) + \chi_{\{g<0\}}(z^{(j)})]$.
  Update the failure probability:
  $P_f^{(k)} = P_f^{(k-1)} + \delta P$.
  $\epsilon = |P_f^{(k)} - P_f^{(k-1)}|$.
end while
Return: $P_f^{(k)}$.

C. THE COMPUTATIONAL PROCEDURE AND THE COMPUTATIONAL COMPLEXITY OF HNH

We construct a hierarchy of neural network model denoted as $g^{(1)}$,$g^{(2)}$,$\cdots$,$g^{(L)}$, where the number of layers are $P_1, P_2, \cdots, P_L$ in a ascending order and there are N neurons in each layer (see Algorithm 1). Our input data set is $Z \in \mathbb{R}^{M\times r}$, $r \ll M$. For the classical single-fidelity neural network $g^{(L)}$, the computational complexity is $(M \times P_L \times N^2)$. In our HNH method, the computational complex is $(M \times P_1 \times N^2 + \frac{M}{L}P_L N^2 + \cdots + \frac{M}{L}P_{L+1-\xi}N^2)$ if the modifications finished after m-th iteration, where $\xi = \text{ceil}(\frac{mL}{M})$. The computational cost of our HNH method will be reduced significantly compared to NH method since $m \ll M$ and $\xi \ll L$ for an acceptable surrogate model.

D. ANALYSES OF HNH METHOD

Assumption 1: Let $C > 1, a > 1, 0 < \rho < 1, M$ is the size of input data Z, $\mathcal{F}^{\ell} = \{z \in Z | \{g^{(\ell)}(z) > \eta\} \cap \{g(z) < 0\}\}$. The models $g^{(\ell)}$ satisfy

$$\int_{\Omega_\eta} \chi_{\mathcal{F}^\ell} dF_Z(z) \le (\frac{1}{C}\frac{1}{1+\exp(ax)})^{\ell}, \tag{9}$$

where $\mathcal{G}$ is sorted $\{|g^{(1)}(z^{(i)})|\}_{i=1}^{M}$, $\eta = \mathcal{G}_{\ell M/L}$, $\eta_{max} = \mathcal{G}_{(1-\rho)M}$, $x = \eta/\eta_{max}$.

This assumption takes the formula of inverse of the Sigmoid function. It is a reasonable assumption for it fitting the numerical results well.

Assumption 2: For any $\epsilon > 0$, given C and a, η_{max}, η_t is a threshold written as

$$\eta_t = \ln(\frac{1}{C\cdot\epsilon})\frac{\eta_{max}}{a}. \tag{10}$$

Theorem 1: Failure probability is P_f, P_f^h is calculated by neural hybrid method, $\hat{P}_f$ is the failure probability evaluated by HNH, $g(Z) \in L_{\Omega}^{p}$, $p \ge 1$ is the given system function, $\hat{g}(Z)$ is the HNH surrogate in L^p- norm, $g^{(\ell)}(Z)$ is a hierarchy of surrogate models for $\ell = 1, \cdots, L$, for any $\epsilon > 0$, there exists η_t, for any $\eta > \eta_t$,

$$|P_f - \hat{P}_f| \le \epsilon. \tag{11}$$

Proof:

$$|P_f - \hat{P}_f| \le |P_f - P_f^h| + |P_f^h - \hat{P}_f|. \tag{12}$$

For $\ell = 1, \cdots, L$,

$$\begin{aligned} |P_f^h - \hat{P}_f| &= \sum_{\ell=1,\cdots,L-1} \int_{\Omega_{\eta_\ell}} \chi_{\{g^{L-\ell}(z)>\eta\}\cap\{g^L(z)<0\}} dF_z(z) \\ &= \sum_{\ell=1,\cdots,L-1} \int_{\Omega_{\eta_\ell}} \chi_{\{\mathcal{U}-\{g^{L-\ell}(z)>\eta\}^{\complement}\cup\{g^L(z)<0\}^{\complement}\}} dF_z(z) \end{aligned} \tag{13}$$

where $\mathcal{U}$ means the universal set, $\eta_1 > \cdots > \eta_{L-1}$.

Choose ℓ, for any $\epsilon > 0$, there exists η_t, for any $\eta > \eta_t$, we can obtain

$$\begin{aligned} &\int_{\Omega_{\eta_{L-\ell}}} \chi_{\{g^\ell(z)>\eta\}\cap\{g^L(z)<0\}} dF_z(z) \\ &= \int_{\Omega_{\eta_{L-\ell}}} \chi_{\{\mathcal{U}-\{g^\ell(z)>\eta\}^{\complement}\cup\{g^L(z)<0\}^{\complement}\}} dF_z(z) \\ &= 1 - (1 - \frac{1}{C}\frac{1}{1+\exp(ax)})^{\ell} - (\frac{1}{C}\frac{1}{1+\exp(ax)})^{L} \\ &\le 1 - (1 - \frac{1}{C}\frac{1}{1+\exp(ax)})^{L} - (\frac{1}{C}\frac{1}{1+\exp(ax)})^{L} \\ &\le (\frac{1}{C}\frac{1}{1+\exp(ax)}) - (\frac{1}{C}\frac{1}{1+\exp(ax)})^{L} < \epsilon. \end{aligned} \tag{14}$$

Upon combining (13)-(14), we obtain

$$|P_f^h - \hat{P}_f| < \epsilon \tag{15}$$

Then, with the definition of failure probability, we have

$$\begin{aligned} &|P_f - P_f^h| \\ &= \int_{\Omega_\eta} \chi_{\{g^{(L)}(Z)>\eta\}\cap\{g(Z)<0\}} dF_Z(z) \\ &\leq (\frac{1}{C}\frac{1}{1+\exp(ax)})^L < \epsilon. \end{aligned} \tag{16}$$

So that

$$|P_f - \hat{P}_f| < \epsilon. \tag{17}$$

Before the estimation of expectation, we present the definition of multifidelity surrogate $\hat{g}(z)$ obtained by HNH.

Definition 2: (HNH Surrogate) For input data $Z = \{z^{(i)}\}_{i=1}^M$, M is the number of total samples, $g^{(\ell)}$, $\ell = 1, \cdots, L$ is a hierarchy of surrogate models, m means the times of modifications and $\xi = ceil(\frac{mL}{M})$, then $\hat{g}$ can be presented as

$$\hat{g} = \frac{1}{L}\sum_{i=1}^{\xi-1} g^{(L+1-i)} + \frac{mL-(\xi-1)M}{ML} g^{(L-\xi+1)} + \frac{M-m}{M} g^{(1)}, \tag{18}$$

when $\xi = 1$, the first term at the right side equals zero.

Assumption 3: $g(Z)$ is given system function, $g^{(\ell)}(Z)$ is a hierarchy of surrogate models for $\ell = 1, \cdots, L$, Z is the dataset. While the models are trained with labeled data, the neural networks can be regarded as unbiased approximators. The expectations are given as

$$\mathbb{E}[g] = \mathbb{E}[g^{(1)}] = \cdots = \mathbb{E}[g^{(L)}]. \tag{19}$$

Theorem 2: The HNH surrogate $\hat{g}(Z)$ is an unbiased estimator of $g(Z)$.

Proof: Suppose HNH modifications finish after m-th iteration, $\xi = ceil(\frac{mL}{M})$,

$$\begin{aligned} \mathbb{E}[\hat{g}] &= \frac{1}{M}[\frac{M}{L}\mathbb{E}[g^{(1)} + (-g^{(1)} + g^{(L)})] \\ &\quad + \cdots \\ &\quad + \frac{M}{L}\mathbb{E}[g^{(1)}] \\ &\quad + \frac{mL - M\xi + M}{L}\mathbb{E}[(-g^{(1)} + g^{(L-\xi+1)})] \\ &\quad + \frac{(L-\xi)M}{L}\mathbb{E}[g^{(1)}]] \\ &= \frac{1}{M}[\frac{M}{L}L \cdot \mathbb{E}[g]] \\ &= \mathbb{E}[g]. \end{aligned} \tag{20}$$

IV. NUMERICAL EXPERIMENTS

In this section we provide three numerical examples to test the performance of our HNH method. For the multivariate benchmark, we use MC to obtain the reference solution. In the test of diffusion and Helmholtz equations, we employ finite element method to calculate the solution. For these three studies, we trained hierarchical neural networks with 6, 15 and 30 layers, and each layer has 500 neurons. The selection of the number of layers and neurons is empirical. All timings conduct on an Intel Core i5-7500, 16GB RAM, Nvidia GTX 1080Ti processor with MATLAB 2018a and Tensorflow under Python 3.6.5. The number of samples we generated in the following tests is related to the limit of RAM. In the time comparison, we set the runtime of HNH method as a unit.

In numerical tests, the numbers of training samples are 10^3, and the total cost includes these parts. While traditional approaches are expensive or infeasible in high-dimensional problems, we compare with the neural hybrid method proposed in this work. It should be noted that after modifications using fine-fidelity networks, the only difference between HNH and NH is runtime, which means that the error plots of two methods in the following are the same with respect to the number of discrete PDE solves.

A. MULTIVARIATE BENCHMARK

Here we consider a high-dimensional multivariate benchmark problem in the field of structural safety from [24].

$$g(X) = \beta n^{\frac{1}{2}} - \sum_{i=1}^{n} X_i, \tag{21}$$

where $\beta = 3.5$, $n = 50$ and $X_i \sim N(0, 1)$.

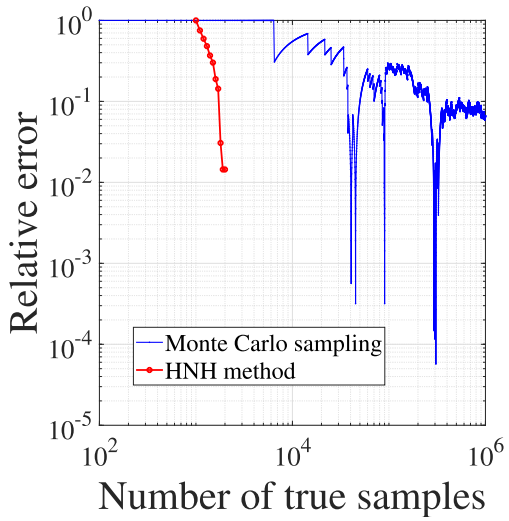
Then we construct our surrogate model using neural network as

$$\hat{g} = \phi_n(w_n\phi_{n-1}(\cdots\phi_1(w_1x + b_1) + \cdots) + b_n), \tag{22}$$

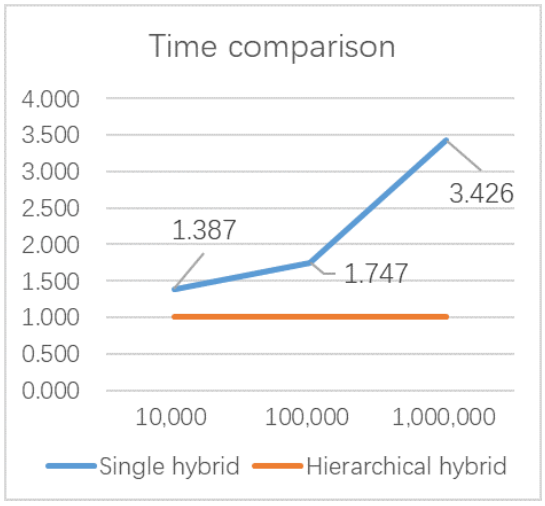
where $\{\phi_j\}_{j=1}^n$ are projections, $\{w_j\}_{j=1}^n$ are weights, $\{b_j\}_{j=1}^n$ are biases and x is input variable generated from the given distribution.

We consider the failure probability $P_f = \text{Prob}(g(X) < 0)$. With 5×10^6 MC sampling, the reference estimation is $P_f^{mc} = 2.218 \times 10^{-4}$.

We generated 5×10^6 samples for evaluating reference estimation, so we set 10^6 samples for the error estimation. In Figure 5(a), blue line is the error of Monte Carlo estimation, and red line is the error of probability evaluated by HNH method. The error decreases quickly and iteration stops automatically since the ϵ between the last two iterations is lower than the threshold ϵ_{opt} **(last two red circles are too close to identify in Figure 5(a)). The phenomenon perfectly fits our Assumption 1 and 2, for the error decreasing to zero when** $g(z) \geq \eta_t$**.** It means that there will be no more modification when $g(z) \geq \eta_t$, and the failure probability estimation converges. So an acceptable probability ($P_f = 2.25 \times 10^{-4}$) can be obtained with 2×10^3 sample generated



(a) Absolute error



(b) Time comparison

FIGURE 5. Performances on accuracy and time of HNH in multivariate benchmark.

from origin system, where 10^3 samples are for training models. Compared with reference value which obtained by using 5×10^6 MC sampling, the absolute error is about 10^{-6} and relative error is 1.443% with less than 0.4% training samples generated from origin system g. In Figure 5(b), we show the timing comparison between our HNH method and our NH method. We set the running time of NH method as the unit time in three orders of magnitude. With number of samples growing, our HNH method is more efficient.

B. DIFFUSION EQUATION

In this numerical test, we consider a diffusion problem. The governing equations of the diffusion problem are

$$\begin{aligned} -\nabla \cdot (a(x,\xi)\nabla u(x,\xi)) &= 1 \quad \text{in} \quad D \times \Gamma \\ u(x,\xi) &= 0 \quad \text{on} \quad \partial D_D \times \Gamma \\ \frac{\partial u(x,\xi)}{\partial n} &= 0 \quad \text{on} \quad \partial D_N \times \Gamma. \end{aligned} \tag{23}$$

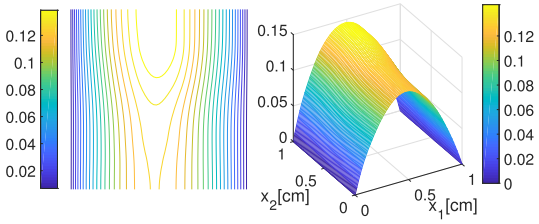


FIGURE 6. The plots show the values of the diffusion equation, where left one is the contour.

where the equation setting can be found in [25]. We can employ finite element method (FEM) and the weak form of (23) is to find an appropriate $u(x,\xi) \in H_0^1(D)$ s.t. $\forall v \in H_0^1(D)$, $(a\nabla u, \nabla v) = (1, v)$. And the finite element solver is from the former work [26], [27].

In our numerical study, the spatial domain is $D(0,1) \times (0,1)$. Dirichlet boundary conditions are applied on the left ($x = 0$) and right ($x = 1$) boundaries. Neumann conditions are applied on the top and bottom conditions. The problem is discretized on a uniform 65×65 grid, and $N_h = 4225$ is the spatial degrees of freedom.

The coefficient $a(x,\xi)$ of the diffusion problem is regarded as a random field, where $a_0(x)$ is mean function, σ is standard deviation and covariance function,

$$\text{Cov}(x,y) = \sigma^2 \exp\left(-\frac{|x_1 - y_1|}{L} - \frac{|x_2 - y_2|}{L}\right), \tag{24}$$

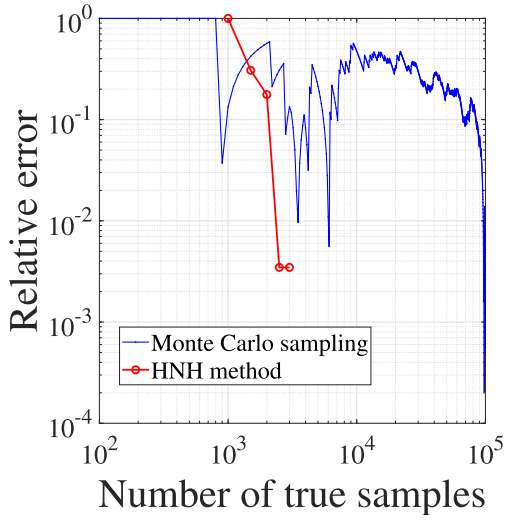
where L is the correlation length. We employ Karhunen-Loéve (KL) expansion [28], [29] to approximate the random field

$$a(x,\xi) \approx a_0(x) + \sum_{k=1}^{d} \sqrt{\lambda_k} a_k(x)\, \xi_k, \tag{25}$$

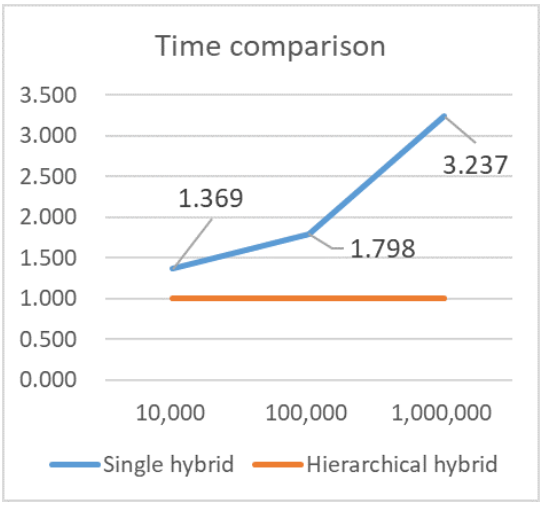
where $a_k(x)$ and λ_k are the eigenfunctions and eigenvalues of (24), $\{\xi_k\}_{k=1}^d$ are random variables. We set $a_0(x) = 1$, $\sigma = 0.42$, and the number d we choose is large enough to capture 95% of total variance of the exponential covariance function [30]. In this paper, we set $d = 48$ for correlation length $L = 0.8$.

The numerical results solved by FEM using IFISS [27] are shown in Figure 6, and we choose $X = [0.5; 0.5]$ as the point sensor placed. For $d = 48$, with 10^6 MC sampling, the reference failure probability for Prob$(u(x,\xi) > 0.19)$ is $P_f = 1.2 \times 10^{-3}$.

In Figure 7(a), our HNH method uses 2×10^3 samples from origin model, and it captures the the failure probability as 1.15×10^{-3}, relative error is 4.17% (the reference solution computed by MC using 10^6 samples). The result is better than MC with around 10^5 samples. In Figure 7(b), our HNH method is more than three times faster than NH method when the magnitude is 10^6. Surrogate method is widely used in huge magnitude problem. NH method is an efficient surrogate method, and our HNH method is more outstanding.



(a) Absolute error



(b) Time comparison

FIGURE 7. Performances on accuracy and time of HNH in diffusion equation for $d = 48$.

C. HELMHOLTZ EQUATION

We now consider the Helmholtz equation

$$-\Delta u - k^2 u = 0, \tag{26}$$

where k is coefficient, and we set the homogeneous term.

We employ MATLAB PDE solver to obtain accurate solutions of Helmholtz equation shown in Figure 8. We choose $X = [0.7264; 0.4912]$ as the point sensor placed. The reference failure probability for $\text{Prob}(u(x, k) > 1.09)$ is $P_f = 2.08 \times 10^{-3}$ computed by MC using 10^5 samples.

Figure 9(a) shows the relative error compared with reference failure probability. Our method captures the failure probability ($P_f = 2.16 \times 10^{-3}$) with only 1500 samples generated from Helmholtz equation where 10^3 for training and 500 for modification. The absolute error is about 8×10^{-5} and relative error is 3.85%. The result is more accurate than MC with 10^4 samples. The estimation of standard MC does not converge within 10^4 samples. Figure 9(b) shows the speedup of our HNH approach compared with our former

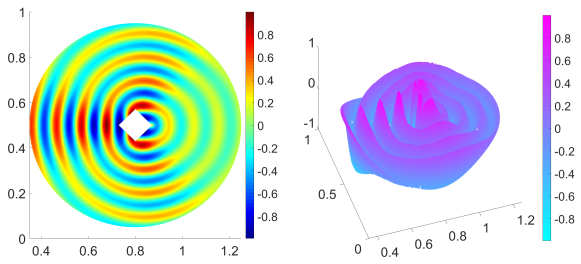
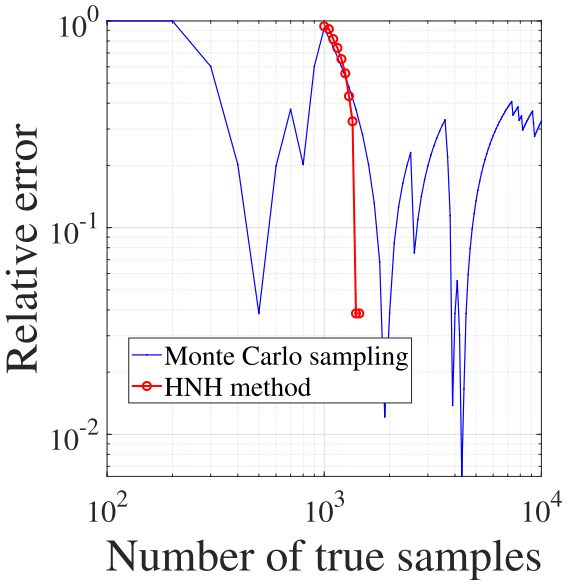
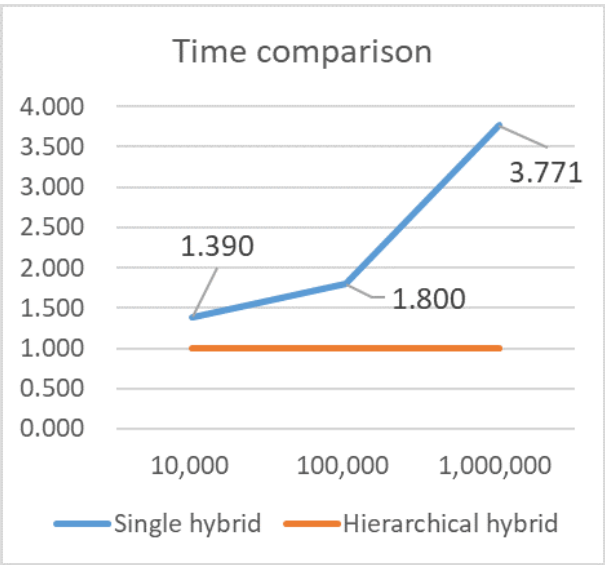


FIGURE 8. A snapshot of the value of the Helmholtz equation.



(a) Absolute error



(b) Time comparison

FIGURE 9. Performances on accuracy and time of HNH in Helmholtz equation.

TABLE 1. Runtime for HNH method and its corresponding neural hybrid (NH) method with different number of samples for three examples in the former part.

Type	10^4 / (ms)	10^5 / (ms)	10^6 / (ms)
NH in benchmark	374.1	556.5	2659.5
HNH in benchmark	**269.6**	**318.6**	**776.4**
NH in diffusion	365.6	563.8	2630.7
HNH in diffusion	**267.1**	**313.5**	**812.7**
NH in Helmholtz	359.4	536.5	2449.3
HNH in Helmholtz	**258.5**	**298.1**	**649.5**

neural hybrid method. Considering the performance in both accuracy and efficiency, our HNH approach performs well.

V. CONCLUSION

Conducting adaptivity is a main concept for efficiently estimating failure probabilities of complex PDE models with high-dimensional inputs. In this work, our HNH procedure adaptively fits the hierarchical structures for this problem. To finally show the efficiency of HNH, we compare it with a direct combination of neural network and hybrid method (which is referred to as NH). Table 1 shows the running times of HNH and NH to achieve the same given accuracy for the three test problems. It can be seen that, as the sample size increases, the efficiency of HNH becomes more clear, e.g., for the Helmholtz problem with $M = 10^6$, the running time of HNH is around a quarter of the time of NH. Moreover, our method employs neural network as a surrogate to overcome the limitations of standard polynomial chaos for high-dimensional problems. From the numerical examples, to achieve the same accuracy, it is clear that HNH only solves the PDEs several thousand times, while traditional MC needs to solve PDEs more than 10^5 times. Due to the universal nature of the neural network, our HNH method can be extend to general surrogate modelling problems.

ACKNOWLEDGMENT

(Ke Li and Kejun Tang contributed equally to this work.)

REFERENCES

[1] N. Metropolis and S. Ulam, "The Monte Carlo method," *J. Amer. Statist. Assoc.*, vol. 44, no. 247, pp. 335–341, 1949.

[2] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics* (Numerical Mathematics and Scie). New York, NY, USA: Oxford Univ. Press, 2014.

[3] A. Der Kiureghian and T. Dakessian, "Multiple design points in first and second-order reliability," *Struct. Saf.*, vol. 20, no. 1, pp. 37–49, 1998.

[4] M. Hohenbichler, S. Gollwitzer, W. Kruse, and R. Rackwitz, "New light on first- and second-order reliability methods," *Struct. Saf.*, vol. 4, no. 4, pp. 267–284, 1987.

[5] M. R. Rajashekhar and B. R. Ellingwood, "A new look at the response surface approach for reliability analysis," *Struct. Saf.*, vol. 12, no. 3, pp. 205–220, Oct. 1993.

[6] C. G. Bucher and U. Bourgund, "A fast and efficient response surface approach for structural reliability problems," *Struct. Saf.*, vol. 7, no. 1, pp. 57–66, 1990.

[7] B. M. Ayyub and R. H. Mccuen, "Simulation-based reliability methods," in *Probabilistic Structural Mechanics Handbook*. Boston, MA, USA: Springer, 1995, pp. 53–69.

[8] R. E. Melchers, "Importance sampling in structural systems," *Struct. Saf.*, vol. 6, pp. 3–10, Jul. 1989.

[9] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, 1989.

[10] K. Song, Y. Zhang, X. Yu, and B. Song, "A new sequential surrogate method for reliability analysis and its applications in engineering," *IEEE Access*, vol. 7, pp. 60555–60571, 2019.

[11] Y. Huang, Q. Xu, S. Abedi, T. Zhang, X. Jiang, and G. Lin, "Stochastic security assessment for power systems with high renewable energy penetration considering frequency regulation," *IEEE Access*, vol. 7, pp. 6450–6460, 2018.

[12] J. Li and D. Xiu, "Evaluation of failure probability via surrogate models," *J. Comput. Phys.*, vol. 229, no. 23, pp. 8966–8980, Nov. 2010.

[13] S.-K. Au and J. Beck, "Estimation of small failure probabilities in high dimensions by subset simulation," *Probabilistic Eng. Mech.*, vol. 16, no. 4, pp. 263–277, Oct. 2001.

[14] A. I. Khuri and S. Mukhopadhyay, "Response surface methodology," *WIREs Comput. Statist.*, vol. 2, no. 2, pp. 128–149, 2010.

[15] J. Li, J. Li, and D. Xiu, "An efficient surrogate-based method for computing rare failure probability," *J. Comput. Phys.*, vol. 230, no. 24, pp. 8683–8697, Oct. 2011.

[16] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Netw.*, vol. 4, no. 2, pp. 251–257, 1991.

[17] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[20] L. W.-T. Ng and K. E. Willcox, "A multifidelity approach to aircraft conceptual design under uncertainty," in *Proc. 10th AIAA Multidisciplinary Design Optim. Conf.*, Jan. 2014, p. 0802.

[21] R. C. Aydin, F. A. Braeu, and C. J. Cyron, "General multi-fidelity framework for training artificial neural networks with computational models," *Frontiers Mater.*, vol. 6, Apr. 2019. doi: 10.3389/fmats.2019.00061.

[22] X. Zhu and D. Xiu, "A multi-fidelity collocation method for time-dependent parameterized problems," in *Proc. 19th AIAA Non-Deterministic Approaches Conf.*, Jan. 2017, p. 1094.

[23] X. Meng and G. E. Karniadakis, "A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems," Feb. 2019, *arXiv:1903.00104*. [Online]. Available: https://arxiv.org/abs/1903.00104

[24] S. Engelund and R. Rackwitz, "A benchmark study on importance sampling techniques in structural reliability," *Struct. Saf.*, vol. 12, no. 4, pp. 255–276, Nov. 1993.

[25] Q. Liao and G. Lin, "Reduced basis ANOVA methods for partial differential equations with high-dimensional random inputs," *J. Comput. Phys.*, vol. 317, pp. 148–164, Jul. 2016.

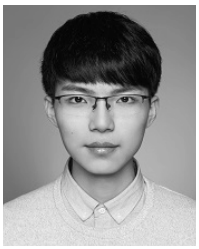
[26] Q. Liao and D. Silvester, "Implicit solvers using stabilized mixed approximation," *Int. J. Numer. Methods Fluids*, vol. 71, no. 8, pp. 991–1006, Jun. 2013.

[27] H. C. Elman, A. Ramage, and D. J. Silvester, "IFISS: A computational laboratory for investigating incompressible flow problems," *SIAM Rev.*, vol. 56, no. 2, pp. 261–273, May 2014.

[28] R. G. Ghanem and P. D. Spanos, "Stochastic finite element method: Response statistics," in *Stochastic Finite Elements: A Spectral Approach*. New York, NY, USA: Springer, 1991, pp. 101–119.

[29] I. Babuška, F. Nobile, and R. Tempone, "A stochastic collocation method for elliptic partial differential equations with random input data," *SIAM J. Numer. Anal.*, vol. 45, no. 3, pp. 1005–1034, 2007.

[30] C. E. Powell and H. C. Elman, "Block-diagonal preconditioning for spectral stochastic finite-element systems," *IMA J. Numer. Anal.*, vol. 29, no. 2, pp. 350–375, Apr. 2009.

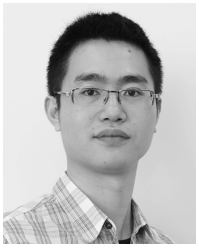


KE LI received the B.S. degree in information and computational science from Chongqing University. He is currently pursuing the master's degree with the School of Information Science and Technology, ShanghaiTech University. His research interests include uncertainty quantification, domain decomposition, and deep learning.

KEJUN TANG received the B.S. degree in computational mathematics from Yantai Univetsity. He is currently pursuing the Ph.D. degree with the School of Information Science and Technology, ShanghaiTech Univetsity. His research interests include tensor methods, machine learning, and scientific computing.

JINGLAI LI received the B.Sc. degree in applied mathematics from Sun Yat-sen University and the Ph.D. degree in mathematics from SUNY Buffalo. After obtaining his Ph.D. degree, he did Postdoctoral Research at Northwestern University and MIT. Prior to arriving at Liverpool, he was an Associate Professor with Shanghai Jiao Tong University. He is currently a Reader with the Department of Mathematical Sciences, University of Liverpool. His current research interests include scientific computing, computational statistics, uncertainty quantification, and data science and their applications in various scientific and engineering problems.

TIANFAN WU received the B.S. degree in mathematics and statistics from Miami University, Oxford. He is currently pursuing the master's degree with the Viterbi School of Engineering, University of Southern California. His research interests include data science and computational statistics.



QIFENG LIAO received the Ph.D. degree in applied numerical computing from the School of Mathematics, The University of Manchester, in December 2010. From January 2011 to June 2012, he held a postdoctoral position at the Department of Computer Science, University of Maryland, College Park. From July 2012 to February 2015, he held a postdoctoral position at the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology. He is currently an Assistant Professor with the School of Information Science and Technology, ShanghaiTech University. His research interest includes efficient numerical methods for PDEs with high-dimensional random inputs, and his work was supported by the National Natural Science of China and Shanghai Young East Scholar.