Engineering Structures 84 (2015) 382–394

Contents lists available at ScienceDirect

Engineering Structures

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

Spectral representation-based neural network assisted stochastic structural mechanics

Dimitris G. Giovanis, Vissarion Papadopoulos *

Institute of Structural Analysis and Antiseismic Research, National Technical University of Athens, Iroon Polytechniou 9, Zografou Campus, Athens 15780, Greece

ARTICLE INFO

Article history:
Received 6 June 2014
Revised 27 November 2014
Accepted 28 November 2014
Available online 17 December 2014

Keywords:
Monte Carlo simulation
Spectral representation
Neural networks
Latin hypercube sampling

ABSTRACT

This paper explores the applicability of artificial neural networks (ANN) for predicting the spread of structural response under the presence of uncertain parameters described as random fields. The use of ANN is carried out in combination with Monte Carlo simulation (MCS) for calculating response statistics in stochastic analysis of structural systems using finite elements. To this extent, the ANNs are trained with a few samples, following a conventional MCS procedure and used henceforth to predict the stochastic response for the rest of samples. The basic idea is to achieve a dimensionality reduction of the input ANN training space by using as input vector the random phase angles of the spectral representation method instead of the random variables describing the uncertain input parameters. A further improvement of the efficiency of the proposed approach is achieved by exploiting the uniform distribution of the random phase angles, in order to span efficiently the training space using a latin hypercube sampling (LHS) technique. The advantage of this approach over conventional computation of stochastic response via a standard stochastic finite element-based MCS is the fast and reliable prediction of the required response sample space which can be accomplished at a fraction of computing time and is independent of the size of the finite element model. Numerical results are presented, demonstrating the efficiency and the applicability of the proposed methodology as well as its distinct advantages over existing ANN-based stochastic finite element methodologies (SFEM).

1. Introduction

The impact of uncertainties in the design process of engineering structures is an important field with growing interest. While most of modern design codes rely on partial safety factors calibrated to target structural reliability, stochastic modeling of the uncertainties in the loading, geometrical, and material properties becomes more and more attractive leading to more rational estimations of structural safety and reliability. Several probabilistic structural analysis methods have been proposed in the past, the simplest being the description of the uncertainties by a set of correlated random variables, where each variable represents a material parameter, load factor, or geometrical property. In several engineering applications however, the description of uncertain parameters using random variables can be insufficient. This is due to the fact that certain physical quantities are often expected to vary randomly in space or time. The probabilistic description of such quantities requires the consideration of random fields. This approach is generally referred as stochastic structural analysis [1].

In principle, brute force MCS is the most suitable and easily implemented method to solve the aforementioned problems. Despite its generality, MCS has been used mostly as a means of verifying the accuracy of approximate and less costly procedures due to its usually high computational cost even if structural analysis is accelerated by advanced solution techniques and/or Neumann series expansion methods [2–4]. To alleviate this drawback advanced variance reduction-based simulation methods have been proposed in the context of reliability analysis, in order to reduce the number of MCS required for an accurate prediction of the probability of failure, such as adaptive sampling, importance sampling, line sampling and subset simulation [5–8]. In addition to the aforementioned methodologies, meta-models, such as artificial neural networks (ANN) have been successfully implemented in the framework of reliability analysis leading to cost-efficient yet acceptable predictions of the probability of failure [9–11]. This is due to the associative memory properties featured by these artificial intelligence devices that allow them to become efficient surrogates to the numerical solver of the mechanical model which is repeatedly invoked in the MCS. Following these early approaches, ANN approximations of the limit-state function were proposed in [12,13] combined either with MC or with first and second order

* Corresponding author. Tel.: +30 2107724158.
E-mail address: vpapado@central.ntua.gr (V. Papadopoulos).

http://dx.doi.org/10.1016/j.engstruct.2014.11.044
0141-0296/

reliability methods (FORM, SORM) for handling the uncertainties. Similar approaches were presented in [11,14–16] where ANN based response surface methods were implemented in order to estimate the reliability integral over the failure domain. Comparative studies of ANN-based reliability analysis with corresponding polynomial approximations of the response surface as well as FORM and SORM methods are presented in [9,11,17,18]. In those papers, the ability of ANN to predict efficiently and accurately enough the reliability of large and complex structural systems is demonstrated. Furthermore it is shown that the advantage of using the ANN approaches is that they adapt efficiently the input/output (I/O) relations allowing for more accurate mappings of the basic random variables than in the corresponding response surface polynomial approximations of the failure functions, especially in complex failure domains [9].

Despite the fact that ANN have been successfully applied in reliability analysis in an increasing number of publications in which uncertainties are described as random variables, similar applications for the cases where uncertainties are described as random fields are very limited. The main difficulty associated with the use of ANN in the context of stochastic finite element analysis (SFEM) is the large number of random variables induced by stochastic processes describing the uncertain system parameters. This feature of SFEM is making ANN inefficient for treating realistic problems due to the large dimensionality of the input vector. In an effort to alleviate this drawback, ANN were used in [19,20] in the context of Karhunen–Loève (KL) expansion for modeling uncertain material properties of one-dimensional random fields. This way a reduced input vector size for the ANN architecture was achieved by considering the random variables that define each eigenvector of the KL decomposition as the training data set. This ANN-based SFEM methodology was used for computing the response statistics (moments and pdf) of simple statically determinate stochastic beams by providing robust ANN estimates of the structural response in the framework of MCS.

In the present study, an alternative to the aforementioned ANN-based SFEM is proposed which is based on the spectral representation (SR) method [21,22] for the description of uncertain system properties. The basic reason for choosing this alternative approach is that it inherits some distinct advantages of SR method over the KL [1,23], namely its optimal convergence in cases that involve relatively short correlation lengths. An additional advantage of this approach, with respect to the KL-based ANN method is that it avoids the computationally expensive solution of the Fredholm integral equation in cases that involve autocorrelation structures for which analytical solutions of the integral equation are not available. In addition, the proposed methodology provides with a useful extension of ANN-based SFEM methods to multi-dimensional stochastic spaces and is formulated in the framework of the multi-dimensional spectral representation method for the modeling of the involved random quantities. A similar to [19] dimensionality reduction of the stochastic variables is achieved by considering directly the random phase angles of the truncated series expansion as the input vector for the ANN training, instead of the random variables describing the uncertain input parameters. The proposed methodology is applied to Gaussian as well as to non-Gaussian random fields in a straight forward manner. Since ANN require a training phase for tuning their parameters, some conventional Monte Carlo simulations are needed at the initial phase of the procedure. For this purpose, the LHS technique [24] is applied in order to effectively span the random space of the phase angles which, according to the spectral representation, are uniformly distributed in the range $[0, 2\pi]$. Thus, the effectiveness of the proposed ANN methodology is further enhanced by the aforementioned uniformly spanned training space. Numerical results are provided demonstrating the efficiency as well as the applicability of the proposed methodology in a stochastic beam as well as a stochastic plane stress finite element system. In addition, the distinct advantages of the proposed approach over KL-based ANN method are showcased.

The remainder of this paper is organized as follows. In Section 2, an overview of the spectral representation method is made. Section 3 contains a brief description of the basics of artificial neural networks. The proposed method is over-viewed in Section 4. Finally, in Section 5 numerical data is presented demonstrating the computational merits of the proposed methodology in terms of the computational performance.

2. Representation of the random fields

In nature, most of the uncertain quantities appearing in engineering systems are non-Gaussian (e.g. material, geometric properties, seismic loads), the Gaussian assumption is often used due to the lack of relevant experimental data and for simple mathematical convenience. It must be noted that this assumption can be problematic in many cases, for example in the case where the Young's modulus is assumed to be a random variable following a Gaussian distribution. In this case negative values for the Young's modulus may occur which have no physical meaning. From the wide variety of methods developed for the simulation of Gaussian stochastic fields, two are most often used in applications: The spectral representation method [22] and the Karhunen–Loève (KL) expansion as a special case of orthogonal series expansion methods [25]. A comparison between these two methods was made in [21]. The results showed that the KL expansion is particularly suitable for the representation of strongly correlated stochastic fields with smooth autocovariance function where only a few terms, corresponding to the N larger eigenvalues, are required in order to capture most of the random fluctuation of the field. On the other hand the spectral representation method, which was preferred in this work, is mostly suitable for the representation of weakly correlated random fields in which a large number of terms in the series expansion is required in order to capture the random fluctuations. In [23] it was presented that if the stochastic field is homogeneous and the observation interval is infinite the KL expansion reduces to spectral representation. Especially, for the case of a finite long field defined in $[-a, a]$ where a is large, it was shown that the eigenvalue is given by the spectral density function.

2.1. An overview of spectral representation method

Spectral representation method expands the stochastic field as a series of trigonometric functions with random phase angles. The simulation formula for a truncated after N_1 terms, one-dimensional homogeneous random field $\widehat{f}(\mathbf{x}_1)$ reads

$$\widehat{f}(\mathbf{x}_1) = \sqrt{2}\sum_{i=1}^{N_1} A_i \cos(\kappa_{1i}\mathbf{x}_1 + \varphi_i) \tag{1}$$

where $\varphi_i (i = 1, \ldots, N_1)$ are independent random phase angles uniformly distributed in the range $[0, 2\pi]$; the frequencies are set to

$$\kappa_{1i} = i\Delta\kappa_1 = i\frac{\kappa_{1u}}{N_1} \quad \text{for } i = 1, \ldots, N_1 \tag{2}$$

where κ_{1u} is the upper cut-off wave number. The coefficients A_i are defined as follows:

$$A_0 = 0 \quad A_i = \sqrt{2S_{f_0}(\kappa_{1i})\Delta\kappa_1} \quad \text{for } i = 1, \ldots, N_1 \tag{3}$$

where S_{f_0} is the power spectral density function which is a real non-negative function of κ_1. The coefficient A_0 is chosen zero such that the temporal mean value averaged over the whole simulation time

$T_0 = 2\pi/\Delta\kappa_1$ of the generated stochastic field $\widehat{f}(\mathbf{x}_1)$ remains zero. The simulated stochastic process of Eq. (1) constitutes the spectral representation of the random field. It is shown by Shinozuka and Deodatis [21] that $\widehat{f}(\mathbf{x}_1)$ is asymptotically Gaussian as $N \to \infty$ and ergodic in the mean and in correlation due to the central limit theorem.

For the cases of two and three dimensional random fields, Eq. (1) takes the form:

$$\widehat{f}(\mathbf{x}_1,\mathbf{x}_2) = \sqrt{2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\left[A_{ij}\cos\left(\kappa_{1i}\mathbf{x}_1 + \kappa_{2j}\mathbf{x}_2 + \varphi_{ij}^{(1)}\right) + \tilde{A}_{ij}\cos(\kappa_{1i}\mathbf{x}_1 - \kappa_{2j}\mathbf{x}_2 + \varphi_{ij}^{(2)})\right] \tag{4}$$

and

$$\begin{aligned}\widehat{f}(\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3) = \sqrt{2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1}^{N_3}\Big[&A_{ijk}\cos\left(\kappa_{1i}\mathbf{x}_1 + \kappa_{2j}\mathbf{x}_2 + \kappa_{3k}\mathbf{x}_3 + \varphi_{ijk}^{(1)}\right)\\ &+\tilde{A}_{ijk}\cos\left(\kappa_{1i}\mathbf{x}_1 + \kappa_{2j}\mathbf{x}_2 - \kappa_{3k}\mathbf{x}_3 + \varphi_{ijk}^{(2)}\right)\\ &+\widehat{A}_{ijk}\cos\left(\kappa_{1i}\mathbf{x}_1 - \kappa_{2j}\mathbf{x}_2 + \kappa_{3k}\mathbf{x}_3 + \varphi_{ijk}^{(3)}\right)\\ &+\check{A}_{ijk}\cos\left(\kappa_{1i}\mathbf{x}_1 - \kappa_{2j}\mathbf{x}_2 - \kappa_{3k}\mathbf{x}_3 + \varphi_{ijk}^{(4)}\right)\Big]\end{aligned} \tag{5}$$

respectively, with

$$A_{ij} = \sqrt{2S_{f_0f_0}(\kappa_{1i},\kappa_{2j})\Delta\kappa_1\Delta\kappa_2} \tag{6}$$

$$\tilde{A}_{ij} = \sqrt{2S_{f_0f_0}(\kappa_{1i},-\kappa_{2j})\Delta\kappa_1\Delta\kappa_2} \tag{7}$$

$$A_{ijk} = \sqrt{2S_{f_0f_0f_0}(\kappa_{1i},\kappa_{2j},\kappa_{3k})\Delta\kappa_1\Delta\kappa_2\Delta\kappa_3} \tag{8}$$

$$\tilde{A}_{ijk} = \sqrt{2S_{f_0f_0f_0}(\kappa_{1i},\kappa_{2j},-\kappa_{3k})\Delta\kappa_1\Delta\kappa_2\Delta\kappa_3} \tag{9}$$

$$\widehat{A}_{ijk} = \sqrt{2S_{f_0f_0f_0}(\kappa_{1i},-\kappa_{2j},\kappa_{3k})\Delta\kappa_1\Delta\kappa_2\Delta\kappa_3} \tag{10}$$

$$\check{A}_{ijk} = \sqrt{2S_{f_0f_0f_0}(\kappa_{1i},-\kappa_{2j},-\kappa_{3k})\Delta\kappa_1\Delta\kappa_2\Delta\kappa_3} \tag{11}$$

$$\Delta\kappa_1 = \frac{\kappa_{1u}}{N_1};\quad \Delta\kappa_2 = \frac{\kappa_{2u}}{N_2};\quad \Delta\kappa_3 = \frac{\kappa_{3u}}{N_3} \tag{12}$$

$$\kappa_{1i} = i\Delta\kappa_1;\quad \kappa_{2j} = j\Delta\kappa_2;\quad \kappa_{3k} = k\Delta\kappa_3 \tag{13}$$

where $S_{f_0f_0}(\kappa_1,\kappa_2)$ and $S_{f_0f_0f_0}(\kappa_1,\kappa_2,\kappa_3)$ are the power spectral density functions for the two and three dimensional case, respectively while κ_{2u} and κ_{3u} are the upper cut-off wave numbers. In Eqs. (4) and (5) the independent phase angles φ are generated randomly in the range $[0, 2\pi]$ and the superscripts (1), (2), (3) and (4) are used to denoted the number of the term of the function which is summed and in which the corresponding phase angle is implemented. For the two and three-dimensional random fields cases the total number of random phase angles required is

$$N_{2D} = 2 \times N_1 \times N_2 \tag{14}$$

and

$$N_{3D} = 4 \times N_1 \times N_2 \times N_3 \tag{15}$$

respectively.

2.2. An overview of Karhunen–Loève expansion

In terms of discretization error, an efficient decomposition of a zero-mean Gaussian random field is the Karhunen–Loève (KL) expansion which approximates it by a linear combination of orthogonal deterministic functions with independent standard Gaussian random variables, where the orthogonal deterministic functions and their magnitude are the eigenfunctions and eigenvalues of the covariance function, respectively. The truncated after M terms KL expansion of a one-dimensional random field $f(\mathbf{x}_1)$ is written as:

$$\widehat{f}(\mathbf{x}_1) = \mu(\mathbf{x}_1) + \sum_{i=1}^{M}\sqrt{\lambda_i}\xi_i\phi_i(\mathbf{x}_1) \tag{16}$$

$\mathbf{x}_1$ being a position variable. In Eq. (16) $\mu(\mathbf{x}_1)$ is the mean value of the random field, λ_i and $\phi_i(\mathbf{x}_1)$ are the eigenvalues and eigenfunctions of its covariance function $C(\mathbf{a}_1,\mathbf{a}_2)$, where $\mathbf{a}_1,\mathbf{a}_2$ are two locations of the random field $\mathbf{x}_1$. By definition, $C(\mathbf{a}_1,\mathbf{a}_2)$ is bounded, symmetric and positive definite with the following spectral or eigen-decomposition:

$$\widehat{C}(\mathbf{a}_1,\mathbf{a}_2) = \sum_{i=1}^{M}\lambda_i\phi_i(\mathbf{a}_1)\phi_i(\mathbf{a}_2) \tag{17}$$

The eigenvalues and eigenfunctions of $\widehat{C}(\mathbf{a}_1,\mathbf{a}_2)$ may be calculated, in the range D of the random field $\widehat{f}(\mathbf{x}_1)$, from the solution of the homogeneous Fredholm integral equation of the second kind given by:

$$\int_D \widehat{C}(\mathbf{a}_1,\mathbf{a}_2)\phi_i(\mathbf{a}_1) = \lambda_i\phi_i(\mathbf{a}_2) \tag{18}$$

The parameter $\xi_i(\theta)$ in Eq. (16) is a set of uncorrelated Gaussian random variables which can be expressed as

$$\xi_i(\theta) = \frac{1}{\sqrt{\lambda_i}}\int_D[\widehat{f}(\mathbf{x}_1,\theta) - \mu(\mathbf{x}_1)]\phi_i(\mathbf{x}_1)d\mathbf{x}_1 \tag{19}$$

with mean and covariance function given by:

$$\begin{aligned}&E[\xi_i] = 0\\ &E[\xi_i\xi_j] = \delta_{ij}\end{aligned} \tag{20}$$

3. Artificial neural networks

Artificial neural networks (ANN) are information processing models configured for a specific application through a training process. Trained ANN provide with rapid mapping of a given input into the desired output quantities (similar to curve fitting procedure) and thereby can be used as meta-models and enhance the computational efficiency of a numerical analysis process. This major advantage of a trained ANN over a conventional numerical analysis procedures, under the provision that the predicted results fall within acceptable tolerances, is that results can be produced in a fraction of wall clock time, requiring orders of magnitude less computational effort than the conventional procedure [26–28].

The most crucial matter in the use of an ANN is the learning procedure which is necessary in order for the network to operate. There are two kinds of learning procedures, the supervised learning and the unsupervised learning. In the case of supervised learning the ANN is trained with pairs of input/output data, referred to as the training patterns $\mathcal{P} = \{(\mathbf{x}^{(p)},\mathbf{t}^{(p)})|p = 1,\ldots,P)\}$, while in the second case the ANN learns automatically from its environment. Neural network consists of a number of units (neurons) linked together and in the case of supervised learning attempts to create a desired mapping between the input and the output data of the training set. The artificial neuron (Fig. 1) processes the information from one training set ($\mathbf{x}^{(k)} = [\mathrm{x}_1,\mathrm{x}_2,\ldots,\mathrm{x}_N]$) and predicts an output vector $\mathrm{y}^{(k)}$, which depends on a vector of connections $\mathbf{w} = w_1, w_2, \ldots, w_N$, known as synaptic weights.

Basically, the learning procedure tries to adjust the synaptic weights in order to have a mapping that fits well the training set.

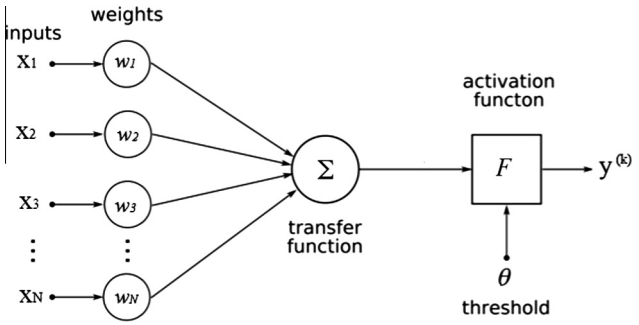

Fig. 1. Schematic plot of an artificial neuron.

The training procedure of an ANN can be considered as a general function optimization problem, with the adjustable parameters being the weights **w** of the network. The quality of this mapping, with respect to the training set, is measured with the aid of an error function. Without loss of generality let us assume that the network maps a given input vector $\mathbf{x} \in \mathbb{R}^N$ into the output vector $\mathbf{y} \in \mathbb{R}^L$. The error function is given by the formula

$$E_D = \frac{1}{2}\sum_{i=1}^{L}(t_i - y_i)^2 \tag{21}$$

where t_i and y_i are the target and computed values of the ith output, respectively. A learning algorithm tries to determine the values of the synaptic weights **w** in order to achieve the correct response for the input vector applied to the network by minimizing the value of E_D. The numerical minimization algorithm used for the training, generates a sequence of weight parameters **w** through an iterative procedure where the update formula can be written as follows:

$$w^{(t+1)} = w^{(t)} + \Delta\, w^{(t)} \tag{22}$$

The increment of the weight parameter $\Delta w^{(t)}$ is further decomposed into

$$\Delta w^{(t)} = a_t d^{(t)} \tag{23}$$

where d_t is a search direction vector and a_t is the step size along this direction.

The algorithms most frequently used in the ANN training are the steepest descent, the conjugate gradient and the Newton's methods with the following direction vectors:

- Steepest descent method: $d^{(t)} = -\nabla E(w^{(t)})$
- Conjugate gradient method: $d^{(t)} = -\nabla E(w^{(t)}) + \beta^{(t-1)} d^{(t-1)}$ where $\beta^{(t-1)}$ is defined: $\beta^{(t-1)} = \nabla E(w^{(t)}) \cdot \nabla E(w^{(t)}) \nabla E(w^{(t-1)}) \cdot \nabla E(w^{(t-1)})$
- Newton's method: $d^{(t)} = -[H(w^{(t)})]^{-1} \nabla E(w^{(t)})$

The learning algorithm used in this work is based on an adaptive version of the Manhattan-learning rule and developed by Riedmiller and Braun is the **R**esilient back**prop**agation abbreviated as Rprop [28]. The ANN learning process progresses iteratively, through a number of epochs. On each epoch the training patterns are submitted in turn to the network and the error is calculated by comparing the actual outputs with the corresponding target values. The weight updates can be written

$$\Delta w_{ij}^{(t)} = -\eta_{ij}^{(t)} sign\left(\frac{\partial E(w^{(t-1)})}{\partial w_{ij}}\right) \tag{24}$$

where

$$\eta_{ij} = \begin{cases} \min\left(\alpha \cdot \eta_{ij}^{(t-1)}, \eta_{max}\right), & \text{if } \frac{\partial E_t}{\partial w_{ij}} \cdot \frac{\partial E_{t-1}}{\partial w_{ij}} > 0 \\ \max\left(b \cdot \eta_{ij}^{(t-1)}, \eta_{min}\right), & \text{if } \frac{\partial E_t}{\partial w_{ij}} \cdot \frac{\partial E_{t-1}}{\partial w_{ij}} < 0 \\ \eta_{ij}^{(t-1)} & \text{otherwise} \end{cases} \tag{25}$$

where $a = 1.2, b = 0.5$, and η are step parameters in the formula for the increment of the design vector w which are called "locally adaptive" learning rates because they are based on weight specific information only such as the temporal behavior of the partial derivative of this weight. The learning rates are bounded by upper and lower limits in order to avoid oscillations and arithmetic underflow. In this work these bounds are set $\eta_{max} = 50$ and $\eta_{min} = 0.1$ [28].

One of the problems that occur during neural network training is the over-fitting. The error on the training set is driven to a very small value, but when new data is presented to the network the error is large. In order to prevent over-fitting we must improve network's generalization. This can be achieved by:

1. Stop the training early – before it has had time to learn the training data too well.
2. Retrain several neural networks.
3. Add some form of regularization term to the error function to encourage smoother network mappings.
4. Add noise to the training patterns to smear out the data points.

In the present work, several neural network were trained in order to ensure that a network with good generalization is found. In each neural network an early stopping method was used in order to further improve generalization. In this technique the available data is divided into three subsets. The first subset is the training set, which is used for computing the gradient and updating the network weights and biases. The second subset is the validation set. The error on the validation set is monitored during the training process. The validation error normally decreases during the initial phase of training, as does the training set error. However, when the network begins to over-fit the data, the error on the validation set typically begins to rise. When the validation error increases for a specified number of iterations the training is stopped, and the weights and biases at the minimum of the validation error are returned. The third subset is the testing data set which is used only for testing the final solution in order to confirm the actual predictive power of the network. This set of data is never used in the training process. Once a neural network is selected based on the validation set, the test set data is applied on the network and the error for this set is computed. This error is representative of the error which we can be expected from absolutely new data for the same problem. However, by partitioning the available data into three sets, we drastically reduce the number of samples which can be used for learning the model, and the results can depend on a particular random choice for the pair of (train, validation, test) sets. In most cases the input data are randomly divided so that 70% of the samples are assigned to the training set, 15% to the validation set and 15% to the testing set.

4. Neural network-based SFEM

As mentioned in the introduction, the use of ANN model as surrogates to the FEM model in the context of SFEM analysis is prohibitive due to the large dimensionality of the required input vector in the ANN training process. For example, in the case of a FEM model with 10^6 finite elements which utilizes the midpoint method for the random field mapping, an input vector of up to 10^6 random variables would be required in the ANN architecture. This number is impossible to be handled by any ANN. The basic idea of the proposed approximation is to achieve a dimensionality reduction of the aforementioned training space, by using as input vector directly the random phase angles $\boldsymbol{\varphi} = \{\varphi_1, \ldots, \varphi_n\},\ n = 1, \ldots, N$ of Eq. (1), where N is the number of intervals of the power spectrum discretization, while the structural response is used as the output vector. Since the number of the random phase angles depend on the discretization of the power spectrum in the wave

number domain the resulting input vector will be independent of the finite element discretization and the number of random variables used for the stochastic description of the uncertain parameter(s). This independence of the proposed method is not to be confused with the well-known dependence of the FE mesh to the SRM which is always present ($dx \leqslant 2\pi/(2Nd\kappa)$). Thus, it is expected that the dimensionality of input vector will be reduced by orders of magnitude, especially in cases where a large scale detailed FEM model is required to describe the structural behavior. A further improvement in the efficiency of the proposed method may be achieved by performing sampling of the random phase angles in the range $[0, 2\pi]$ with a LHS design, as described below.

It must be made clear however that a sufficiently discretized FEM mesh, both in terms of providing a "converged" solution as well as of having a mesh size of a fraction of the correlation length in order to accurately represent the fluctuations of the random input parameter, remains a prerequisite for an accurate SFEM analysis. However, this prerequisite is not yet correlated to the size of the input training vector of the ANN since the input consists of the random phase angles, which are mesh independent.

In order to demonstrate the proposed methodology the classical midpoint SFEM approach is considered where, a different random variable $f(x_i)$ must be assigned at the centroid of each element using the spectral representation method (Eq. (1)). Hence, the structure nodal displacements $\mathbf{u}$ are computed as:

$$\mathbf{u} = \mathbf{K}^{-1}(f(x_i))\mathbf{F}. \tag{26}$$

where the stiffness matrix of the structure $\mathbf{K}$ is clearly a function of the random variables $f(x_i)$, $\mathbf{F}$ being the nodal force vector. Conventional implementation of a ANN model as a surrogate to the FE model would use $f(x_i), i = 1, \ldots N_e$ as the ANN input, where N_e is the number of elements. However, Eq. (1) indicates that $f(x_i)$ depend on the random phase angles $\boldsymbol{\varphi}$. Therefore, Eq. (26) may be rewritten as:

$$\mathbf{u} = \mathbf{K}^{-1}(\boldsymbol{\varphi})\mathbf{F}. \tag{27}$$

Thus, the ANN may be designed having the reduced space of random phase angles $\boldsymbol{\varphi}$ as input vector instead of $f(x_i)$ replacing the dependence of $\mathbf{K}$ on $f(x_i)$ with a dependence on $\boldsymbol{\varphi}$.

4.1. Selection of training input data

The appropriate selection of the input training data is also an important factor for a successful training since the training set must include data over the entire range of the output space. Although the number of training patterns plays its own role in the accuracy of the predictions, the distribution of samples is of greater importance. The selection of the input training data is based on the requirement that the full range of possible results should be represented in the training procedure. In the present study the selection of the random phase angles of the truncated spectral representation as the input allows for an efficient implementation of LHS technique [24] due to the fact that LHS is proven to reduce the variance of statistical estimates in a Monte Carlo analysis [29].

A LHS design is constructed in such a way that each one of the random phase angles (dimensions) is divided into equal levels (bins) and that there is only one point (or sample) at each level. Thus, for our case the range $[0, 2\pi]$ is divided into equally spaced distances for the selection of suitable ANN training pairs covering efficiently the total random variables space. In Fig. 2 we see a schematic representation of the LHS for $N = 5$ realizations of 2 random variables. More specific, in Fig. 2(a) a stratified sampling is performed on the two random variables from each cumulative distribution function while in Fig. 2(b) the samples are randomly put together in order to generate the parameters to be entered in to the model for each simulation. The advantage of the LHS is that the random samples are generated from all ranges of possible values, in a way that no sub-domain is over-sampled.

4.2. Steps of proposed methodology

The basic steps of the proposed methodology are summarized below for the simplest case of one dimensional random fields. The generalization however to 2D and 3D fields is straightforward.

1. Choose the number of terms N for the truncated series of Eq. (1) which correspond to the number of independent phase angles φ_i used as input vector for the ANN training.
2. Choose an appropriate architecture of the ANN. The input layer consists of N neurons.
3. Use LHS technique in order to generate random phase angles uniformly distributed over $[0, 2\pi]$, as defined previously.
4. Generate a number of sample functions of the stochastic properties from Eq. (1) using the phase angles generated in the previous step.
5. Compute the response of the system for the generated sample functions with the finite element method, using the midpoint method, i.e by assigning a corresponding random variable at the centroid of each finite element of the discretized structure.
6. Use the computed response in order to obtain the input/output pair for the ANN training.
7. Train the neural network using the I/O pairs.
   If the error in the prediction of the ANN is smaller than a target value then go to the next step otherwise go to step 2 using a more refined LHS design [30,31] and/or refine the number of nodes in the ANN.
8. Proceed in the MCS using ANN.

A schematic description of the aforementioned methodology is presented in Fig. 3

The advantage of the proposed methodology is that the size of the input vector for the ANN training depends only on the number of terms of the truncated series expansion of the power spectrum. This number is orders of magnitude smaller than the number of random variables, describing the random process as the input vector, which makes feasible the ANN implementation in the context of SFEM. Two numerical examples are presented in the next section in order to demonstrate the efficiency of the proposed methodology. It must be mentioned that the computer platform used for the computational task was a Intel(R) Core(TM) i5-2430M CPU @ 2.40 GHz, with 6 GB RAM memory while the software used for the generation of the home-made code of the SFEM is MATLAB R2013b. For the neural networks the matlab toolbox was used in order to implement ANN in the framework of SFEM.

5. Numerical examples

5.1. 1D cantilever with Gaussian random field

Consider the statically determinate cantilever of length $L = 8$ m and section 0.25 m $\times$ 0.5 m shown in Fig. 4 with the nodes at the left edge of the cantilever fixed against translations and rotations. The cantilever is subjected to a uniformly distributed load $p = 50$ kN/m. and discretized with 200 beam elements.

The compliance (inverse of the elastic modulus) is assumed to vary randomly along its length according to the following expression:

$$\frac{1}{E(\mathbf{x})} = F_0(1 + f(\mathbf{x})) \tag{28}$$

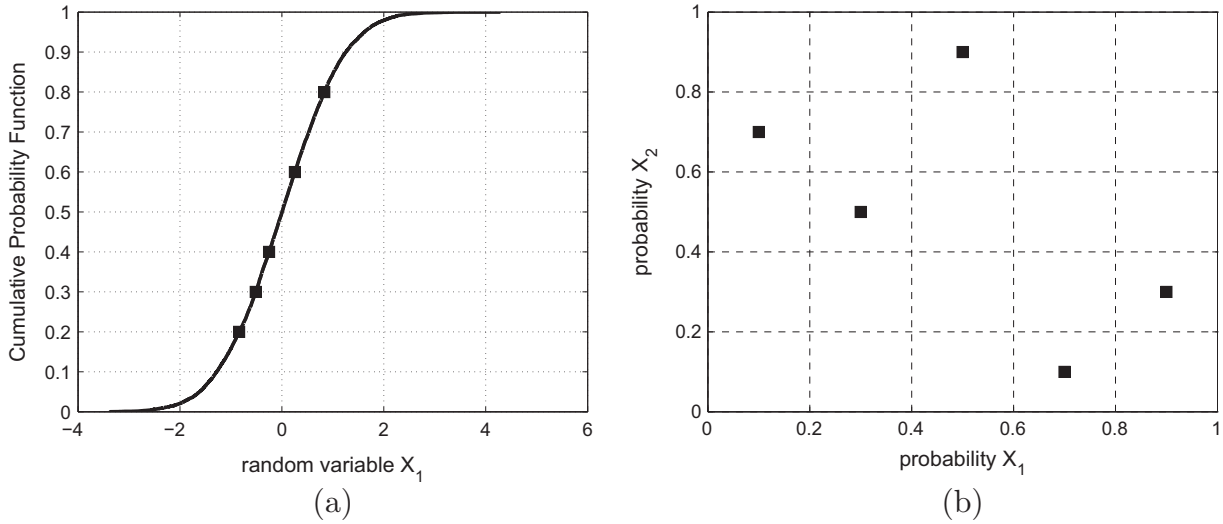

Fig. 2. The latin hypercube approach for sampling from probability distributions: (a) stratified sampling on each random variable from its cumulative distribution (b) samples are put together randomly in order to generate the parameters to be entered into the model for each simulation.

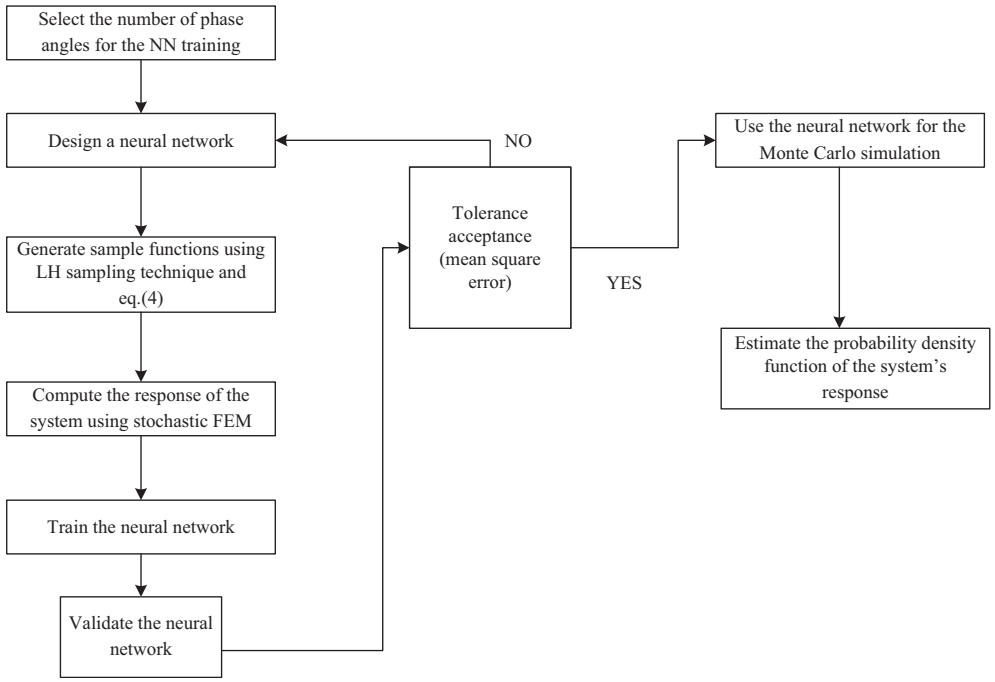

Fig. 3. A flow chart of the proposed methodology.

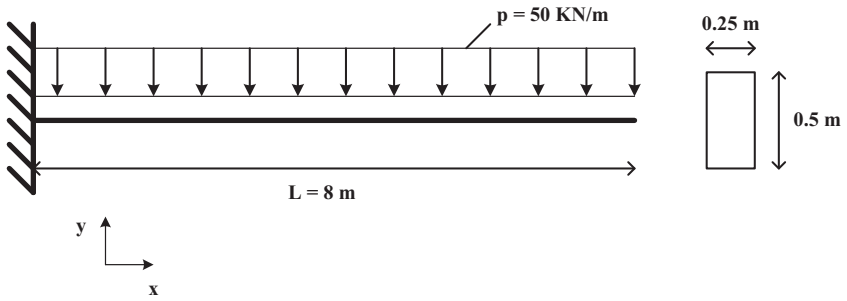

Fig. 4. Example 1: a cantilever beam fixed against translations and rotations in its left edge.

F_0 is the mean value of the compliance, and $f(\mathbf{x})$ is a zero-mean homogeneous stochastic field modeling the variation of the compliance around its mean value. In this example $F_0 = (2.1 \times 10^7 \text{ kN/m}^2)^{-1}$. Its covariance structure is assumed to be of exponential type:

$$C(\mathbf{x}_1, \mathbf{x}_2) = \sigma_f^2 e^{-\frac{|\Delta \mathbf{x}|}{b}} \tag{29}$$

where $\Delta \mathbf{x} = \mathbf{x}_2 - \mathbf{x}_1, b$ is the correlation length and σ_f is the standard deviation of the random field describing the compliance. In this example, $\sigma_f = 0.15$ and the value of $b = 0.4$ m is selected as a relatively short correlation length parameter representing the random field. The corresponding spectral density function is given by equation

$$S_f(\kappa) = \sigma_f^2 \cdot \frac{b}{\pi \cdot (1 + b^2 \kappa^2)} \tag{30}$$

Initially, brute force MCS with $N_{MCS} = 50,000$ samples was applied in order to obtain a reference solution. This number of MCS was selected in order to accurately estimate a failure probability in the order of 10^{-2}—10^{-3}. The mean value and the variance of the tip displacement for different number of terms in the SR method of Eq. (1) are gathered in Table 1. This Table also depicts the probability of the tip displacement to be larger (in absolute value) than a critical value $d_{cr} = -0.5236$ m. This probability was estimated to be $P_F = 5.2\%$ for $N = 10, P_F = 3.7\%$ for $N = 20, P_F = 1.98\%$ for $N = 50$ and $P_F = 2.0\%$ $N = 120$. As we can see from this Table an accurate enough description of the random field is achieved with only $N = 50$ terms in the SR. The value of d_{cr} used is located $2.5 \times \sigma_U$ far from the mean displacement response, where σ_U is the standard deviation of the response.

Next, the proposed methodology was implemented in order calculate the same response statistics together with the probability of failure as discussed previously.

5.1.1. Training and testing the neural networks

In a preliminary phase, the ANN architecture is selected; the topology includes N neurons for the input layer (same as the number of terms of the spectral representation), one hidden layer (the maximum number of neurons H_{max} of the hidden layer depends on the number of training samples and the size of the output vector) and 1 neuron for the output layer. The Rprop algorithm was selected for the updating of the weights. In order to further improve the efficiency of the ANN training the inputs/outputs pairs were scaled so the fall in the range $[-1, 1]$. For estimating the optimum number of neuron in the hidden layer different ANNs were trained for increasing number of neurons $(1 : H_{max})$, each re-initialized 10 times in order to check different starting values of the weights and thus avoiding phenomena of over-fitting the data. The best neural network ("optimum") in terms of the mean square error of the prediction (according to a criterion selected, e.g $mse \leqslant 10^{-2}$) is stored.

After the "optimum" ANN is found and stored, its generalization performance towards a set of test data is assessed (the test data contain 50 input/output pairs). The ANN's performance is calculated based on a Test Error Function (TEF) given by

$$TEF = \frac{1}{N} \sum_{i=1}^{N} \frac{|T_i - O_i|}{T_i} \times \% \tag{31}$$

where N is the total number of test patterns, T_i is the actual response value for the ith test pattern, and O_i is the ANN's estimation of the response for the ith test pattern.If the TEF is lower than a target value (e.g 0.1%) then the trained ANN can be used in the framework of MCS otherwise the procedure is repeated by changing the training parameters.

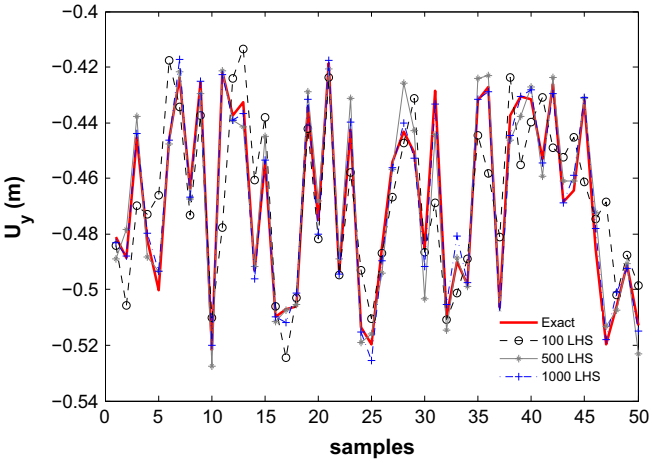

Fig. 5. Example 1: performace of the two trained ANN- Actual response vs ANN's estimation for 50 testing patterns.

5.1.2. Results for $N = 50$

The results of the proposed methodology for the case of $N = 50$ and 100, 500 and 1000 training samples are presented next. Fig. 5 depicts the results of testing the trained ANN which, is an indicator of their quality evaluation. In this figure the actual response value is plotted over the three trained ANN's estimation for 50 testing patterns. A quantification of these results is given in Table 2. In this Table it can be seen that very close predictions of the response statistics are obtained with the proposed methodology with respect to MCS method. The number of training samples used for training the ANN plays a critical role. Specifically, for the case where 100 LHS samples were used for the training, the relative error of the ANN-SFEM procedure, denoted with "$err^{U_y}\%$", with respect to MCS, in the prediction of the mean value of the vertical displacement of the tip of the cantilever is 5.14% while, by increasing the number to 500 and 1000 LHS samples this error reduces to 1.66% and 0.2136%, respectively. The same behavior is observed for the error in the prediction of the corresponding variance (also denoted as "$err^{\mathbf{Var}}\%$") where, for 100, 500 and 1000 training points, the error was found to be 36.27%, 27.86% and 0.8276%, respectively. This Table also depicts the number of neurons in the hidden layer required for an optimum ANN performance together with the

Table 1
Example 1: estimates of the mean value end variance of U_y with 50,000 MCS samples for $N = 20, N = 50$ and $N = 120$ number of terms in the SR method, along with the corresponding probabilities of failure.

FE analyses	d_{cr}	SR terms (N)	Mean $U_y(m)$	Variance	$P_F = P(U_y \leqslant d_{cr})$ (%)
50,000	−0.5236	10	−0.5102	10.89×10^{-4}	5.2
		20	−0.4855	8.02×10^{-4}	3.7
		50	−0.4683	7.29×10^{-4}	1.98
		120	−0.4682	7.25×10^{-4}	2.0

Table 2
Example 1: ANN-SFEM predictions of the mean value and variance of U_y for 100, 500 and 10,000 LHS training points of three different ANNs with 3, 10 and 21 neurons in the hidden layer.

ANN samples	Training sample size (FE analyses)	Number of neurons H	*TEF* (%)	Mean $U_y(m)$	$err^{(U_y)}(\%)$	Variance	$err^{(\mathrm{Var})}$ (%)
50,000	100	8	3.4752	−0.4923	5.14	9.88×10^{-4}	36.27
	500	15	1.241	−0.4760	1.66	5.23×10^{-4}	27.86
	1000	29	0.6853	−0.4692	0.21	7.19×10^{-4}	0.82

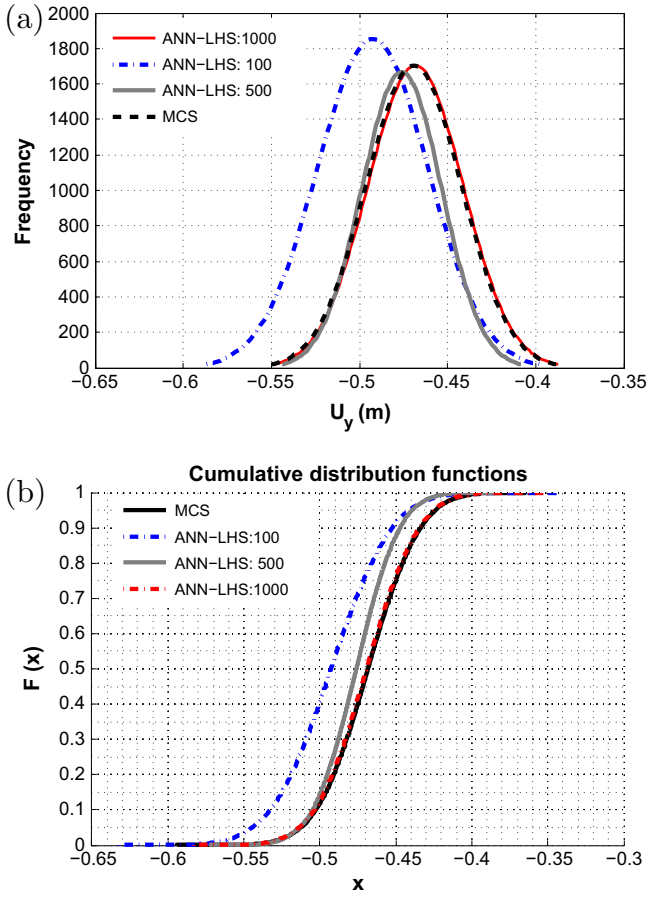

Fig. 6. Example 1 (a) histograms of the displacement of the cantilever tip obtained with MCS and ANN-based MCS with 50,000 simulations (b) cumulative distribution function (CDF) for the MCS and the ANN-MCS approach for 100, 500 and 1000 LHS points.

TEF. These are found to be $H = 8, H = 15$ and $H = 29$ for 100, 500 and 1000 training points with corresponding *TEF* errors 3.475%, 1.241% and 0.6853%, respectively.

Histograms of the displacements U_y for both brute force MCS and the proposed ANN-SFEM approach, are shown in Fig. 6(a) for the 100, 500 and 1000 training points, respectively, while Fig. 6(b) depicts the corresponding cumulative distribution functions (CDFs). As can be observed from this figure, for the case of

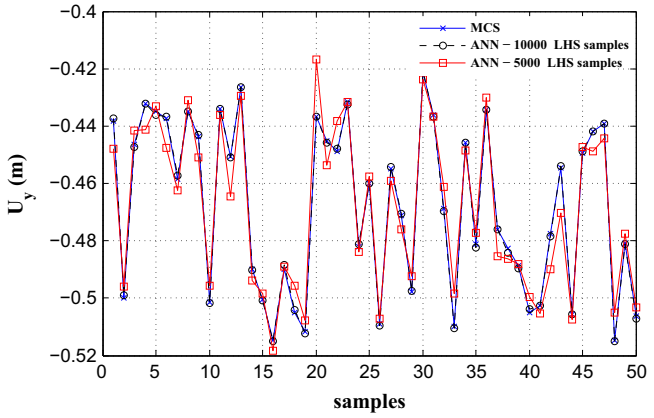

Fig. 7. Example 1: performace of the two trained ANN- Actual response vs ANN's estimation for 50 testing patterns.

100 and 500 training points the histograms and CDFs of the tip displacement obtained with the proposed methodology deviate from the one obtained by the conventional MCS, while for the case of 1000 training samples the results are almost identical to the one obtained with MCS. Table 3 presents the probability of failure estimated with ANN-SFEM with 100, 500 and 1000 training points. For the first case (100 points) was computed at 16% while for the other two cases (500 and 1000 points) it was found 2.5% and 1.99%. The corresponding errors were 700%, 25% and 0.5%, respectively.

A more clear view of the advantage of the proposed methodology over the standard MCS is presented through a comparison of the required computing time. For MCS and for the proposed methodology, the CPU times required are presented in Table 4. In this Table it can be seen that the CPU time required for MCS with 50,000 samples is 111.5 s, while for the same number of simulations with ANN-SFEM and 100 training samples is 0.23 s. For 500 and 1000 samples this time is 1.115 s and 2.23 s, respectively, which, again is 50 times less with respect to standard MCS, within acceptable tolerance (error of less than 0.5%). In this time we must add the time required for the estimation of the 50 training patterns (0.1115 s) plus the time required for the ANN training procedure (depending on the number of patterns used and the architecture of the ANN) which, in cases of realistic problems is significantly less compared to the time required for one MCS plus the time required for the simulation of the ANN samples after properly trained (less than 1 s). In this Table the reduction of the CPU time (%) of the ANN-SFEM method compared to the MCS is also presented.

Table 3
Example 1: estimation of the probability of failure P_F (probability the displacement of the tip of the cantilever exceeds a critical value $P(U_y \leqslant d_{cr})$ and its corresponding error with ANN-based MCS and 50,000 ANN samples.

ANN samples	MCS P_F	Training sample size (FE analyses)	ANN-MCS P_F (%)	Error (%)
50,000	2	100	16	700
		500	2.5	25
		1000	1.99	0.5

Table 4
Example 1: CPU time for MCS and ANN-SFEM – ANN training and testing.

Method	FE analyses	CPU time (s)	% reduction
MCS	50,000	111.5	–
	FE analyses (LHS: Train + test)	CPU time (s)	% reduction
ANN-SFEM	100 + 50	0.341	966.6
ANN-SFEM	500 + 50	1.23	958.86
ANN-SFEM	1000 + 50	2.34	949.21

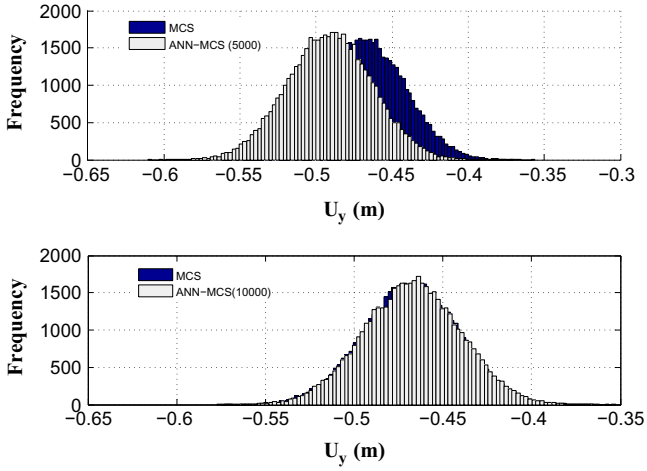

Fig. 8. Example 1: histograms of the displacement of the cantilever tip obtained with MCS and ANN-based MCS with 5000 and 10,000 training samples for 50,000 simulations.

5.1.3. Results for $N = 120$

Although, as shown previously, a number of $N = 50$ is sufficient for the discretization of the power spectrum for this problem, the results of the proposed methodology for the case of $N = 120$ are also presented in order to demonstrate the efficiency of the methodology in cases in which a more dense discretization is required. For this case, 5000 and 10,000 training samples are depicted next. In Fig. 7 we can see a schematic representation of the generalization performance of the ANN while Table 5 shows the results obtained with each trained ANN. For the case were 5000 LHS samples are used for the training the relative error of the ANN-SFEM procedure with respect to MCS, in the prediction of the mean value of the vertical displacement of the tip of the cantilever is 4.698% while, by increasing the number to 10,000 LHS samples this error reduces to 1.069%. The error in the prediction of the corresponding variance for 5000 and 10,000 training points, is found to be 9.241% and 1.931%, respectively. The number of neurons in the hidden layer is also presented in this Table together with the corresponding *TEF* errors.

Fig. 8 shows the histograms obtained with the proposed methodology together with the ones obtained with brute force MCS for 50,000 simulations while Fig. 9 depicts the corresponding CDFs. As can be observed from these figures, for the case of 5000 training points the histogram and CDF of the tip displacement obtained with the ANN-MCS significantly deviates from the ones obtained by the conventional MCS, while for the case of 10,000 training points the results obtained with the proposed methodology almost coincide with the one obtained with MCS. Table 6 presents the probability of failure estimated with ANN-SFEM with 5000 and 10,000 training points. For the first case (5000 points) was computed at 6% while for the other case (10,000 points) it was found 1.93% . The corresponding errors were 200% and 3.5%, respectively.

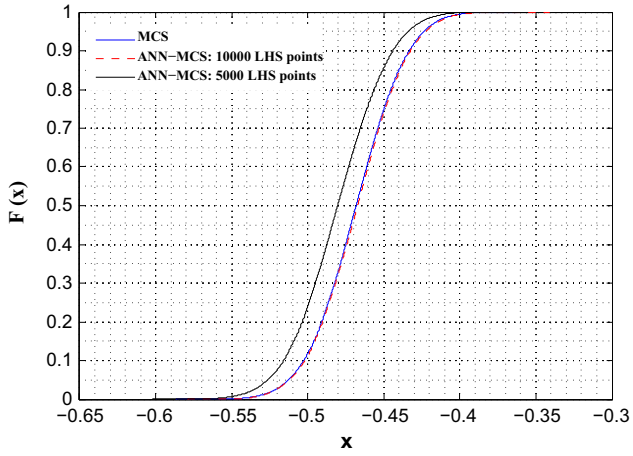

Fig. 9. Example 1: cumulative distribution function (CDF) for the MCS and the ANN-MCS approach for 5000 and 10,000 LHS points.

Table 7 shows the CPU time required with MCS and ANN-MCS method. Again, we see that a significant reduction in the computational effort required for the estimation of the first two moments of the response and for the probability of failure with the proposed methodology compared to brute force MCS. This reduction is of order 89.9% for 5000 training points and of order 79.9% for 10,000 training points.

5.1.4. Demonstration of distinct advantages of the proposed SR-based approach with respect to KL-based ANN methods

In order to demonstrate the advantage of this approach over KL-based relative methodologies, an exact analytic solution was utilized as reference solution for comparison purposes. This solution is based on the concept of variability response function (VRF), initially introduced by Shinozuka [32] for statically determinate beams and further developed by many researchers [33–38]. According to [34] an exact closed form expression for the response variance for the beam of Fig. 4 is given as

$$\mathrm{Var}[u(\mathbf{x})] = \int_{-\infty}^{\infty} \mathrm{VRF}(\mathbf{x}, \kappa) S_f(\kappa) d\kappa \tag{32}$$

where VRF is the variability response function which can be estimated numerically using a fast Monte Carlo simulation (FMCS) approach whose basic idea is to consider the stochastic field as a random sinusoid. The numerical estimation of the VRF through FMCS is extremely important as the closed-form analytic

Table 5
Example 1: ANN-SFEM predictions of the mean value and variance of U_y for 5000 and 10,000 LHS training points of two different ANNs with 39 and 71 neurons in the hidden layer.

ANN samples	Training sample size (FE analyses)	Number of neurons H	*TEF* (%)	Mean U_y (m)	$err^{(U_y)}$ (%)	Variance	$err^{(\mathrm{Var})}$ (%)
50,000	5000	39	3.025	−0.491	4.698	7.92×10^{-4}	9.241
	10,000	71	0.124	−0.473	1.069	7.39×10^{-4}	1.935

Table 6
Example 1: estimation of the probability of failure P_F (probability the displacement of the tip of the cantilever exceeds a critical value $P(U_y \leqslant d_{cr})$ and its corresponding error with ANN-based MCS and 50,000 ANN samples.

ANN samples	MCS P_F (%)	Training sample size (FE analyses)	ANN-MCS P_F (%)	Error (%)
50,000	2	5000	6	200
		10,000	1.93	3.5

Table 7
Example 1: CPU time for MCS and ANN-SFEM – ANN training and testing.

Method	FE analyses	CPU time (s)	% reduction
MCS	50,000	1115	–
	FE analyses (LHS: Train + test)	CPU time (s)	% reduction
ANN-SFEM	5000 + 50	112.615	89.9
ANN-SFEM	10,000 + 50	224.115	79.9

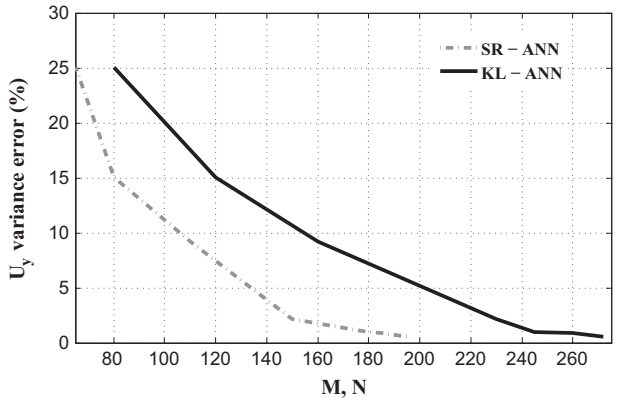

Fig. 10. Error in the estimation of the cantilever's tip displacement variance as a function of the number of terms M and N in the truncated KL and SR methods, respectively.

expressions involve modulating functions that are very difficult to establish. For statically determinant beams, an exact analytical expression for the VRF is

$$VRF(\mathbf{x},\kappa) = \left|\frac{F_0}{I}\int_0^x h(\mathbf{x},\xi)M(\xi)e^{i\kappa\xi}d\xi\right|^2 \tag{33}$$

In this formula S_f is the power spectral density function of the random field $f(\mathbf{x})$ and $h(\mathbf{x},\xi)$ and $M(\xi)$ are the beam's Green [34] and bending moment functions, respectively.

Fig. 10 depicts the error (%) convergence of the response variance at the tip of the beam ($L = 8$ m), as a function of the number of terms used in both the proposed SR-ANN methodology and the KL-ANN approach as this is described in [20]. The error is computed with respect to the reference solution of Eq. (32). From this figure it can be seen that the error is decaying for increasing number of terms in both SR-ANN and KL-ANN methods but the rate of the decay in the case of SR-ANN is faster than the KL-ANN method. For example, 100 terms are required for the proposed SR-ANN methodology to reach a target error of 10%, while the same error is reached with the KL-ANN approach with a 35% increased number of terms (150 terms). The same behavior is observed for lower orders of the error, where for example an error lower than 5% is achieved with 135 terms in the SR-ANN method and 200 terms in the KL-ANN. This difference between the number of terms required in the SR-ANN and the KL-ANN method in order to achieve a target error in the estimation of the response variance

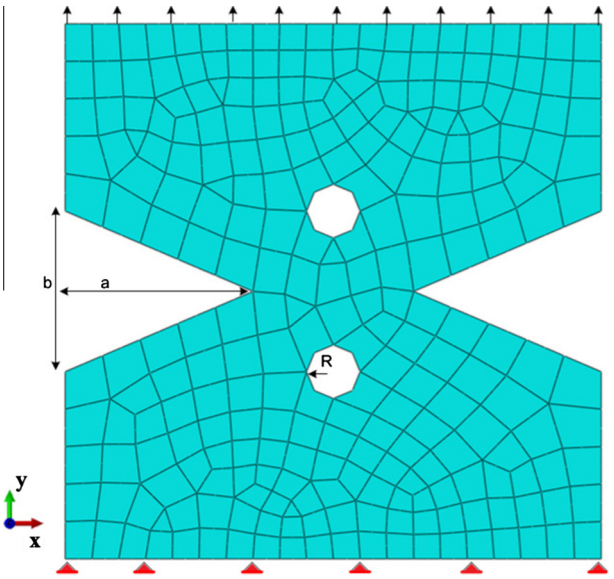

Fig. 11. Example 2: 2D plane stress plate.

Table 8
Example 1: comparison of the computational costs for the ANN training.

Method	CPU time (s)
SR-ANN	224.115
KL-ANN	302.55

is of crucial importance in the performance of the neural network, due to the fact that the higher the number of terms is, the higher the dimension of the input training vector and thus higher number of training samples are required in order to obtain the same quality in the training of the ANN. In fact, a 35% increase in the size of the input vector will result in an analog increase of the training samples (~35%) in order to maintain the same performance level of the ANN. This linear dependence between input size and training size has been evidenced in [39], together with a thorough explanation of this phenomenon. The advantage in computational efficiency of the proposed SR-ANN methodology over KL-ANN for the aforementioned distinct case of a relatively short correlation length ($b = 0.4$ m), is demonstrated in Table 8, which depicts the CPU time required for both approaches to reach the same target error of 10%. From this Table it can be seen that, following the increase in the training points required for the KL-ANN (15,000 points instead of 10,000), a 35% less computing time is required for the SR-ANN method, to achieve the same performance with the KL-ANN approach.

Finally, as mentioned previously, the eigen-solution of the covariance function in the KL-ANN method involves the solution of an integral equation, for which only a limited number of analytical eigen-solutions are available, only for simple geometries. For most covariance functions, the analytical solution of the integral equation for KL expansion is not tractable and needs to be evaluated numerically. In [23] it was shown that analytical KL gives significantly better results than numerical KL and thus more terms are needed for the numerical KL to represent the random process for a given accuracy. Consequently, the computational cost of the KL and consequently the KL-ANN method is further increased in cases where numerical methods are used for the solution of the Fredholm integral equation. For this example the additional CPU time required for the numerical solution of the

Fredholm equation and 150 terms is 79 s which is almost three times the total SR-ANN CPU time. However, it must be mentioned here that the influence of this additional cost might not be so pronounced in large-scale problems for which the cost of a single analysis is orders of magnitude higher than for the simple beam of this example.

5.2. Non-Gaussian random field

The proposed methodology can be applied also in cases involving strongly non-Gaussian stochastic fields. The two-dimensional perforated plate of Fig. 11 is considered for this case. The domain is a rectangle of length $L_x = 1$ m and width $L_y = 1$ m, with holes in the center of the domain of radius $R = 0.1$ m and two symmetric cut-offs at the middle, with dimensions $a = b = 0.33$ m. The domain is discretized with 400 plane stress quadrilateral elements. The model is subjected to a constant uniform tension load $p = 50$ kN/m along its boundary at its upper side.

The modulus of elasticity is considered to vary randomly along the x-direction according to the expression $E(x) = E_0 \cdot (1 + h(\mathbf{x}))$, with mean value equal to $E_0 = 2.1 \times 10^7$ kN/m^2. $h(\mathbf{x})$ is a zero-mean random field with standard deviation equal to 0.4 ($\sigma_h = 0.4$), defined as

$$h(\mathbf{x}) = F^{-1}\Phi[f(\mathbf{x})] \tag{34}$$

where Φ is the standard Gaussian cumulative distribution function and F is non-Gaussian "U-shaped" beta marginal cumulative distribution function given by:

$$F = \frac{\Gamma(C+D)}{\Gamma(C)\Gamma(D)(B-A)^{C+D-1}}(\mathbf{x}-A)^{C-1}(B-\mathbf{x})^{D-1} \tag{35}$$

The values of parameters are selected as follows: $A = -1.1$, $B = 1.7, C = 0.5, D = 0.5$. The transformation $F^{-1} \cdot \Phi$ is a memoryless translation since the value of the non-Gaussian random field $h(x)$ at an arbitrary point x depends only on the value of underlying Gaussian random field $f(\mathbf{x})$ at the same point. The resulting non-Gaussian field is called translation field [40]. It must be mentioned here that when simulating a translation field from its underlying Gaussian field, the resulting non-Gaussian field will not possess the specified correlation/power spectral density (a phenomenon referred to as correlation distortion). In order to generate realizations according to a prescribed non-Gaussian power spectrum, iterative approaches are typically used such as that proposed in [41]. In this work the underlying Gaussian field $f(\mathbf{x})$ is simulated according to the spectral density function of Eq. (30) using the spectral representation method with standard deviation equal to one ($\sigma_f = 1$) and correlation length equal to five ($b = 5$). For the generation of the Gaussian random field, $N = 128$ terms in the truncated spectral representation are selected in a way similar to the first example. For each realization,the y-displacement of the upper right node is monitored. Response statistics are used to estimate the mean value and the variance as well as histograms of U_y. First, a brute force MCS is performed with $N_{MCS} = 50,000$. The results are gathered in Table 9 and the histogram of the response is presented in Fig. 12.

Next, the proposed methodology is applied in order estimate the mean value and the variance of the response. Regarding the ANN topology in this case $N = 128$ neurons for the input layer (same as the number of terms of the spectral representation) are selected, one hidden layer and 1 neuron for the output layer. The results of the proposed methodology for different training sample sizes for the ANN are depicted in Table 10. Finally, the histograms computed using the proposed approach for various training sample sizes are plotted in Figs. 13, 14 and 15.

From Table 10 it can be seen that again very close predictions of the response statistics are obtained with the proposed methodology with respect to MCS method. As discussed before, the number of the training samples affects the performance of the ANN training. For example in the case where 1500 LHS points were used, the error in the prediction of the mean value of the displacement is almost 20% while the error in its variance is of order 19%. By increasing the number of training samples to 4000 the results obtained with the ANN are significantly improved and the errors of in the prediction of the mean value of the response and its variance are reduced to 9% and 7%, respectively. Finally, by using 10,000 training points the results we obtain are similar to the results obtained with brute force MCS; the errors of in the prediction of the mean value of the response and its variance are 0.0% and 2%, respectively. Observing Figs. 13,14 and 15 we can see that, for the case of 1500 training points the histogram of the displacement obtained with the proposed methodology deviates from the one

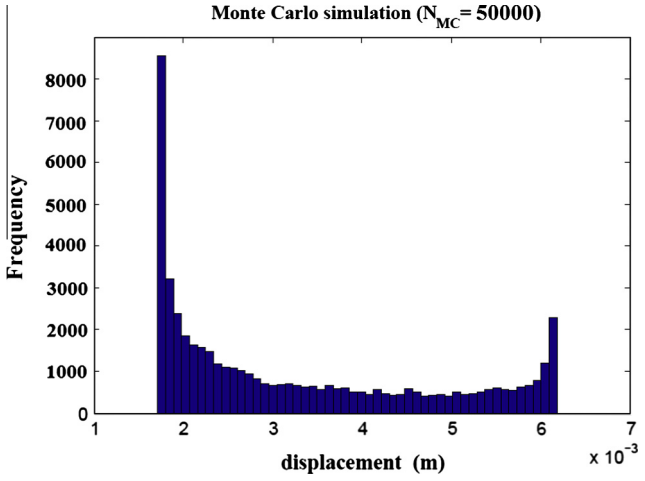

Fig. 12. Example 2: histogram of the displacement of the upper right node of the plate along the y-axis, obtained with 50,000 MCS.

Table 9
Example 2: MCS estimates of the first two moments of U_y.

FE analyses	Mean U_y (m)	Variance
10,000	0.0033	2.26×10^{-6}

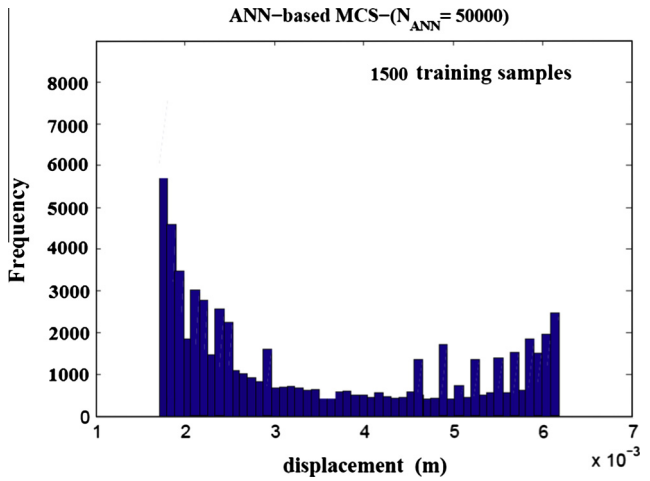

Fig. 13. Example 2: histogram of 50,000 predictions of the displacement of the upper right node of the plate along the y-axis, obtained with the ANN with 1500 training points.

Table 10
Example 2: ANN-SFEM predictions of the first two moments of U_y.

ANN samples	Training sample size (FE analyses)	Number of neurons in the hidden layer	Mean U_y (m)	$err^{(U_y)}$ (%)	Variance	$err^{(Var)}$ (%)
50,000	1500	29	0.0039	19.78	2.68×10^{-6}	18.58
	4000	42	0.0035	9.73	2.48×10^{-6}	7.21
	10,000	55	0.0033	0.0	2.32×10^{-6}	2.1

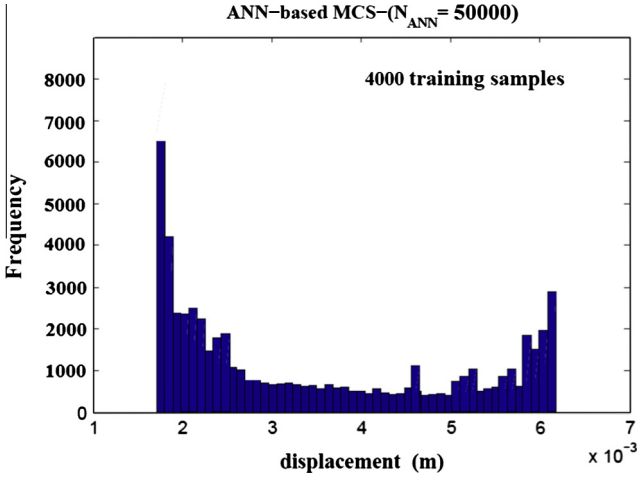

Fig. 14. Example 2: histogram of 50,000 predictions of the displacement of the upper right node of the plate along the y-axis, obtained with the ANN with 4000 training points.

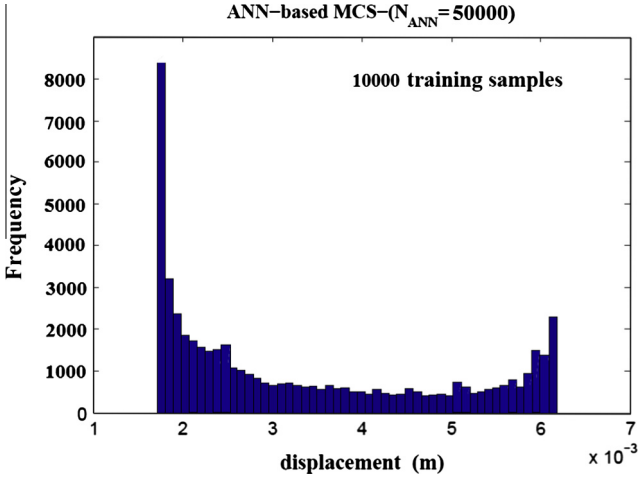

Fig. 15. Example 2: histogram of 50,000 predictions of the displacement of the upper right node of the plate along the y-axis, obtained with the ANN with 10,000 training points.

obtained by the conventional MC, while for the case of 10,000 training points the histogram is almost identical to the one obtained with MC. From these figures it is concluded that also in this strongly non-Gaussian case the proposed methodology converges to the results of the brute force MCS in a robust and efficient manner at a fraction of computational time. The CPU time required for the above calculations is depicted in Table 11. From this Table it can be seen that the CPU time required for MCS with 50,000 samples is 51,595 s while the CPU time required for the same number of simulations with ANN-SFEM with 1500, 4000 and 10,000 training samples is 1547.8, 4127.6 and 10,319 s, respectively. Again, in this time we must add the time required for the ANN testing (around 50 s) and the duration of the training procedure plus the time needed for the trained ANN to simulate the samples. Thus, a reduction around 80% was achieved for the more accurate ANN-SFEM case of 10,000 training points.

Table 11
Example 2: CPU time for MCS and ANN-SFEM – ANN training and testing.

Method	FE analyses	CPU time (s)	% reduction
MCS	50,000	51,595	–
	FE analyses (LHS: Train + test)	CPU time (s)	% reduction
ANN-SFEM	1500 + 50	1625.21	96.6
ANN-SFEM	4000 + 50	4201.32	91
ANN-SFEM	10,000 + 50	10,390	79.86

6. Conclusions

In this paper a neural network assisted stochastic finite element analysis of static linear models is proposed in the context of Monte Carlo simulation for assessing the spread of stochastic response under the presence of uncertain parameters described as random fields. The basic idea is to achieve a dimensionality reduction of the I/O ANN training space by using as input vector the random phase angles of the spectral representation method. A further improvement of the efficiency of the proposed approach is achieved by efficiently sampling the random phase angles in the range $[0, 2\pi]$ using a LHS technique. The presented test examples demonstrated the efficiency of the proposed SR-based ANN stochastic FEM and its distinct advantages with respect to other existing ANN-SFEM methods. Acceptable results were achieved at a fraction of computing time compared to the standard Monte Carlo simulation with finite element analysis. The method can be applied also in a straightforward manner to two and three-dimensional stochastic fields. Consequently, this method seems to be an efficient tool towards the direction of accurately assessing the effect of a random spread of design quantities in real world structures at an affordable computational cost. We can thus summarize that neural networks offer an alternative way of performing stochastic analysis of structural systems especially for problems where conventional methods cannot be efficiently applied. An extension of this method to non-linear and/or dynamic mechanics is considered a task for future work since these types of problems are most relevant in structural mechanics.

Acknowledgements

This work has been supported by the European Research Council Advanced Grant MASTER – Mastering the computational challenges in numerical modeling and optimum design of CNT reinforced composites (ERC-2011-ADG-20110209).

References

[1] Stefanou George. The stochastic finite element method: past, present and future. Comput Methods Appl Mech Eng 2009;198:1031–51.
[2] Charmpis DC, Papadrakakis M. Improving the computational efficiency in finite element analysis of shells with uncertain properties. Comput Methods Appl Mech Eng 2005;194:1447–78.

[3] Papadopoulos V, Papadrakakis M. Stochastic finite element-based reliability analysis of space frames. Probab Eng Mech 1998;1:53–65.
[4] Papadrakakis M, Papadopoulos V. Robust and efficient methods for stochastic finite element analysis using monte carlo simulation. Probab Eng Mech 1996;134:325–40.
[5] Au SK, Papadimitriou C, Beck JL. Reliability of uncertain dynamical systems with multiple design points. Struct Saf 1999;21:113–33.
[6] Au SK, Beck JL. Estimation of small failure probabilities in high dimensions by subset simulation. Probab Eng Mech 2001;16(4):263–77.
[7] Bucher CG. Adaptive sampling – an iterative fast Monte Carlo procedure. Struct Saf 1988;5(2):119–26.
[8] Melchers RE. Importance sampling in structural systems. Struct Saf 1989;6:3–10.
[9] Hurtado JE, Alvarez DA. Neural network-based reliability analysis : a comparative study. Comput Methods Appl Mech Eng 2001;191(1–2):113–32.
[10] Deng J. Structural reliability analysis for implicit performance function using radial basis function network. Int J Solids Struct 2006;43:3255–91.
[11] Bucher C, Most T. A comparison of approximate response functions in structural reliability analysis. Probab Eng Mech 2008;23(2–3):154–63.
[12] Lagaros ND, Garavelas ATh, Papadrakakis M. Innovative seismic design optimization with reliability constraints. Comput Methods Appl Mech Eng 2008;198(1):28–41.
[13] Mesbahi E, Pu Y. Application of ANN-based response surface method to prediction of ultimate strength of stiffened panels. J Struct Eng 2008;134(10):1649–56.
[14] Cheng J, Li QS. Reliability analysis of structures using artificial neural network based genetic algorithms. Comput Methods Appl Mech Eng 2008;197(45–48):3742–50.
[15] Lagaros ND, Papadrakakis M, Fragiadakis M, Stefanou G, Tsompanakis Y. Neural network aided stochastic computations and earthquake engineering. Comput Assist Mech Eng Sci 2007;14:251–7.
[16] Papadopoulos V, Giovanis DG, Lagaros ND, Papadrakakis M. Accelerated subset simulation with neural networks for reliability analysis. Comput Methods Appl Mech Eng 2012;223–224:70–80.
[17] Gomes HM, Awruch AM. Comparison of response surface and neural network with other methods for structural reliability analysis. Struct Saf 2004;26:49–67.
[18] Sudret B., Der Kiureghian A. Stochastic finite element methods and reliability; A State-of-the-Art Report. Report No. UCB/SEMM–2000/08; 2000.
[19] Hurtado JE. Neural networks in stochastic mechanics. Arch Comput Methods Eng 2001;8(3):303–42.
[20] Hurtado JE. Analysis of one-dimensional stochastic finite elements using neural networks. Arch Comput Methods Eng 2002;5:3–30.
[21] Shinozuka M, Deodatis G. Simulation of multi-dimensional Gaussian stochastic fields by spectral representation. Appl Mech Rev 1996;49:29–53.
[22] Shinozuka M, Deodatis G. Simulation of the stochastic process by spectral representation. Appl Mech Rev ASME 1991;44(4):29–53.
[23] Quek ST, Huang SP, Phoon KK. Convergence study of the truncated Karhunen–Loève expansion for simulation of stochastic processes. Int J Numer Methods Eng 2001;52:1029–43.
[24] Beckman RJ, McKay MD, Conover WJ. A comparison of three methods for selecting values of input variables in the analysis of output from computer code. Technometrics 1979;42:55–61.
[25] Zhang J, Ellingwood B. Orthogonal series expansions of random processes in reliability analysis. J Eng Mech (ASCE) 1994;120(12):2660–7.
[26] MacKay DJC. A practical Bayesian framework for backprop networks. Neural Comput. 1992;4:448–72.
[27] Papadrakakis M, Lagaros ND, Tsompanakis Y. Structural optimization using evolution strategies and neural networks. Comput Methods Appl Mech Eng 1998;156:309–33.
[28] Riedmiller M. Advanced supervised learning in multi-layer perceptrons from back-propagation to adaptive learning algorithms. Int J Comput Stand Inter 1994;16:265–78.
[29] Stein M. Large sample properties of simulations using Latin hypercube sampling. Technometrics 1987;29:143–51.
[30] Tong C. Refinement strategies for stratified sampling methods. Reliab Eng Syst Saf 2006;91:1257–65.
[31] Helton JC, Sallaberry CJ, Hora SC. Extension of Latin hypercube samples with correlated variables. Reliab Eng Syst Saf 2008;93:1047–59.
[32] Shinozuka M. Structural response variability. J Eng Mech (ASCE) 1987;113:825–42.
[33] Papadopoulos V, Deodatis G. Response variability of stochastic frame structures using evolutionary field theory. Comput Methods Appl Mech Eng 2006;195(9–12):1050–74.
[34] Deodatis G, Papadopoulos V, Papadrakakis M. Flexibility-based upper bounds on the response variability of simple beams. Comput Methods Appl Mech Eng 2005;194:1385–404.
[35] Wall FJ, Deodatis G. Variability response functions of stochastic plane stress/strain problems. J Eng Mech 1994;120(9):1963–82.
[36] Arwade SR, Deodatis G. Variability response functions for effective material properties. Probabilist Eng Mech 2010;26:174–81.
[37] Deodatis G, Graham L, Micaletti R. A hierarchy of upper bounds on the response of stochastic systems with large variation of their properties: random variable case. Probab Eng Mech 2003;18(4):349–64.
[38] Papadopoulos V, Papadrakakis M, Deodatis G. Analysis of mean response and response variability of stochastic finite element systems. Comput Methods Appl Mech Eng 2006;195(41–43):5454–71.
[39] Demuth HB, Hagan MT, Beale M. Neural network design. PWS Publishing Company; 1996.
[40] Grigoriu M. Simulation of stationary non-Gaussian translation processes. J Eng Mech (ASCE) 1998;124:121–6.
[41] Shields MD, Deodatis G, Bocchini P. A simple and efficient methodology to approximate a general non-Gaussian stationary stochastic process by a translation process. Probabilist Eng Mech 2011;26:511–9.