Approximation of complex nonlinear functions by means of neural networks

Thomas Most*

Abstract

In this paper the applicability of neural networks for the approximation of several nonlinear problems is investigated. Neural networks are used to represent the limit state and implicit performance functions of complex systems. The obtained results are compared to these from classical response surface approaches. The advantage of neural networks especially for high-dimensional problems will be shown. Finally the successful application of neural networks for the identification of the numerical parameters of complex constitutive laws will be presented.

Keywords: Neural networks, response surfaces, reliability, identification

*Contact: Dipl.-Ing. Thomas Most, Institute of Structural Mechanics, Bauhaus-University Weimar, Marienstrasse 15, D-99423 Weimar, E-Mail: thomas.most@uni-weimar.de

1 Introduction

In engineering science the modeling and numerical analysis of complex systems and relations plays an important role. In order to realize such investigation, for example a stochastic analysis, in a finite computational time, approximation procedure have been developed. A very famous approach is the response surface method, where the relation between input and output quantities is represented for example by global polynomials or local interpolation schemes as Moving Least Squares introduced by Lancaster and Salkauskas (1981). In recent years artificial neural networks (ANN) have been applied as well for such purposes.

Artificial Neural Networks have been designed to model the processes in the human brain numerically. These mathematical models are used today mainly for classification problems as pattern recognition. In the recent years a large number of different neural network types have been developed, e.g. the multi-layer perceptron, the radial basis function network, networks with self-organizing maps and recurrent networks. A good overview is given e.g. in Hagan et al. (1996).

Neural networks (ANN) have been applied in several studies for stochastic analyses, e.g. in Papadrakakis et al. (1996), Hurtado and Alvarez (2001), Cabral and Katafygiotis (2001), Nie and Ellingwood (2004a), Gomes and Awruch (2004), Deng et al. (2005) and Schueremans and Van Gemert (2005). In these studies the structural uncertainties in material, geometry and loading have been modeled by a set of random variables. A reliability analysis has been performed either approximating the structural response quantities with neural networks and generating ANN based samples or by reproducing the limit state function by an ANN approximation and decide for the sampling sets upon failure without additional limit state function evaluations. The main advantage of ANN approximation compared to RSM is the applicability to higher dimensional problems, since RSM is limited to problems of lower dimension due to the more than linearly increasing number of coefficients. In Hurtado (2002) firstly a neural network approximation of the performance function of uncertain systems under the presence of random fields was presented, but this approach was applied only for simple one-dimensional systems.

A further application in engineering science is the identification of parameters of numerical models e.g. of constitutive laws. In Lehký and Novák (2004) the material parameters of a smeared crack model for concrete cracking were identified. In this study the relation between artificial load-displacement curves and the corresponding numerical parameters are used to train the network and the parameters are approximated directly from the experimental curves.

Both types of application are in the point of interest of the actual research activities at the Institute of Structural Mechanics of The Bauhaus-University. In this paper several examples will be presented, which clarify the power of neural networks. The numerical results will be compared to these obtained from classical response surface approaches.

2 Neural network approximation

The basic network includes nodes and connections which link the nodes. Each link is associated with a weight property which is the principal mechanism how a network stores information. Before a neural network can approximate complex coherences it has to be trained for the specific problem by adjusting these weights. The most widely used network type for approximation problems is the multi-layer perceptron which is also called feed-forward back-propagation network. This network type is used in this study and is shown in Fig. 1. The network consists of an input layer, several hidden layers and an output layer

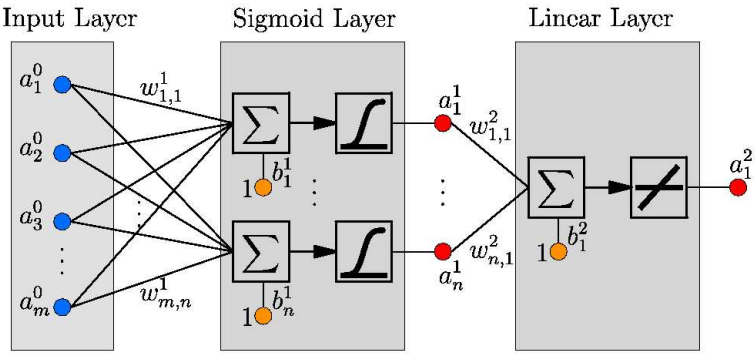

Figure 1: Neural network with feed-forward architecture and one hidden layer

and all nodes which are called neurons of one layer are connected with each neuron of the previous layer. This connection is feed-forward and no backward connection is allowed as in recurrent networks. At the neuron level a bias is added to the weighted sum of the inputs and the neuron transfer function is applied, which can be of linear and nonlinear type. The output of a single neuron reads

$$a_i^j = f_i^j(x) = f\left(\sum_{k=1}^{m} w_{k,i}^j a_k^{j-1} + b_i^j\right) \tag{1}$$

where m is the number of input impulses, i is the number of the current neuron in the layer j. $w_{k,i}^j$ is the synaptic weight factor for the connection of the neuron i, j with the neuron $k, j-1$. For the approximation of functions with minor discontinuities generally a combination of layers with sigmoid transfer functions $f(x) = 1/(1+e^{-x})$ and a linear output layer with $f(x) = x$ are used. Other transfer types are for example hard limit and sign function, which can represent strong discontinuities. A complete list of different transfer functions is given in Demuth and Beale (2002).

Three points have an important influence on the approximation quality of a neural network. The first one is the training of the network. The training for a feed-forward multi-layered network is called back propagation where the network operation is executed reversely for the training sets and the calculated input values are compared to the given values. Depending on a given learning rate the calculated error is corrected and the same procedure is repeated until the training values are reproduced optimally by the network. If no convergence is achieved, generally the training is terminated after a given number of training loops called epochs. Different learning algorithm have been developed for this

purpose which are described in Hagan et al. (1996). In this work the Scaled Conjugate Gradient Algorithm proposed by Møller (1993) is used since it has a fast and stable convergence. In the investigated examples the training is generally stopped after 2000 epochs and the best training set of three runs is used for each configuration.

The second important point for a sufficient network approximation is the design of the network architecture. Depending on the number of available training samples the number of neurons in the hidden layers has to be chosen in that way, that the so-called over-fitting is avoided. This phenomenon occurs, if the number of hidden nodes is to large for the number of training samples. Then the network can converge easier and fits well for the training data but it can not generalize well for other data. In Hagan et al. (1996) it is mentioned that the number of training samples m should be larger than the number of adjustable parameters

$$(n+2)M+1<m \tag{2}$$

where n is the number of input values and M is the number of hidden neurons for a network with single hidden layer. This leads to a much smaller number of required samples compared to RSM for high-dimensional problems if the number of hidden neurons is taken not to large. In Demuth and Beale (2002) two other approaches are discussed to avoid over-fitting for a large number of hidden neurons and a small number of training samples: regularization and early stopping. One automatic regularization method is called Bayesian training proposed by MacKay (1992) which handles the input values as random variables. This training method leads to a very good generalization of the data approximation but it is very slow for higher dimensions compared to Newton-based training approaches. The early stopping needs an additional data set, the control set, and stops the training if the error in the control set starts to increase, while the training error decreases. In general this approach does not avoid over-fitting completely and the disadvantage is that an additional number of data sets is needed compared to standard training. Because of the disadvantages of both alternative methods, here the standard way is used and the number of neurons is estimated following Eq. 2.

The final important influence on the approximation quality is the choice of appropriate training samples. In the most probabilistic analyses with neural networks e.g. in Papadrakakis et al. (1996) and Hurtado (2002) the training samples are generated by standard Monte Carlo Simulation. Especially for a small number of training samples this may lead to an insufficient covering of the stochastic space. In Lehký and Novák (2004) a stochastic training using Latin Hypercube Sampling was proposed, where the space is covered much better.

3 Numerical examples

3.1 Approximation of nonlinear limit state functions

In this example first a nonlinear limit state function given in Katsuki and Frangopol (1994) is investigated. This two-dimensional function is defined by a set of two linear and two quadratic functions:

$$\begin{aligned} g_1(\mathbf{x}) &= 0.1(x_1 - x_2)^2 - (x_1 + x_2)/\sqrt{2} + \beta_1 \\ g_2(\mathbf{x}) &= 0.1(x_1 - x_2)^2 + (x_1 + x_2)/\sqrt{2} + \beta_2 \\ g_3(\mathbf{x}) &= x_1 - x_2 + \sqrt{2}\beta_3 \\ g_4(\mathbf{x}) &= -x_1 + x_2 + \sqrt{2}\beta_4 \end{aligned} \tag{3}$$

whereby X_1 and X_2 are independent Gaussian random variables with zero mean and unit standard error. The reliability indices are given in Katsuki and Frangopol (1994) as $\beta_1 = \beta_2 = 4.0$ and $\beta_3 = \beta_4 = 4.5$.

The approximation of the limit state $g(\mathbf{x}) = 0$ is carried out by estimating the closest distance from the mean to $g(\mathbf{x}) = 0$ depending on the direction $1/\|\mathbf{X}\| \cdot [X_1 \quad X_2]^T$. In order to enable an application of this approximation concept for limit state functions which are unbounded in a certain region the inverse radius is used as approximation quantity similar to Nie and Ellingwood (2004b), which leads to a zero value for an unbounded direction and to the largest values for the points with the largest probability.

The support points are generated in order to be regular on this unit circle. In Fig. 2 the original and the approximated limit state functions are shown by using neural networks, a quadratic polynomial, weighted interpolation and MLS interpolation with exponential weight (MLS-G) and a scaled influence radius $D = 8/n$, where n is the number of support points. The figure clearly indicates, that with increasing number of support points the neural network and MLS approximations become very good estimates of the very complex function. By using the weighted interpolation the original function is approximated well, but with with remarkable oscillations between the support points. The application of the quadratic polynomial does not lead to an improved approximation if the number of support points increases.

In order to quantify the quality of the approximation the failure probability is calculated for each function by using directional sampling proposed by Bjerager (1988). In Fig. 3 the obtained results are shown in comparison to the reference solution $\hat{P}_{ref}(F) = 4.541 \cdot 10^{-5}$ obtained with exactly the same 1000 samples.

The figure indicates, that the neural network approximation gives very good results if at least 16 support points are used. The failure probability using the MLS interpolation is also estimated very well with a small number of samples. Due to the mentioned oscillations the weighted interpolation converges slower to the reference solution. As already shown in Fig. 2 the approximation with a global polynomial is not very sufficient for this example and it converges to a wrong failure probability.

This example shows that for very complex functions the neural network approximation needs a certain number of neurons and the corresponding support points to obtain a sufficient approximation. If only 8 support points are available only 2 neurons can be

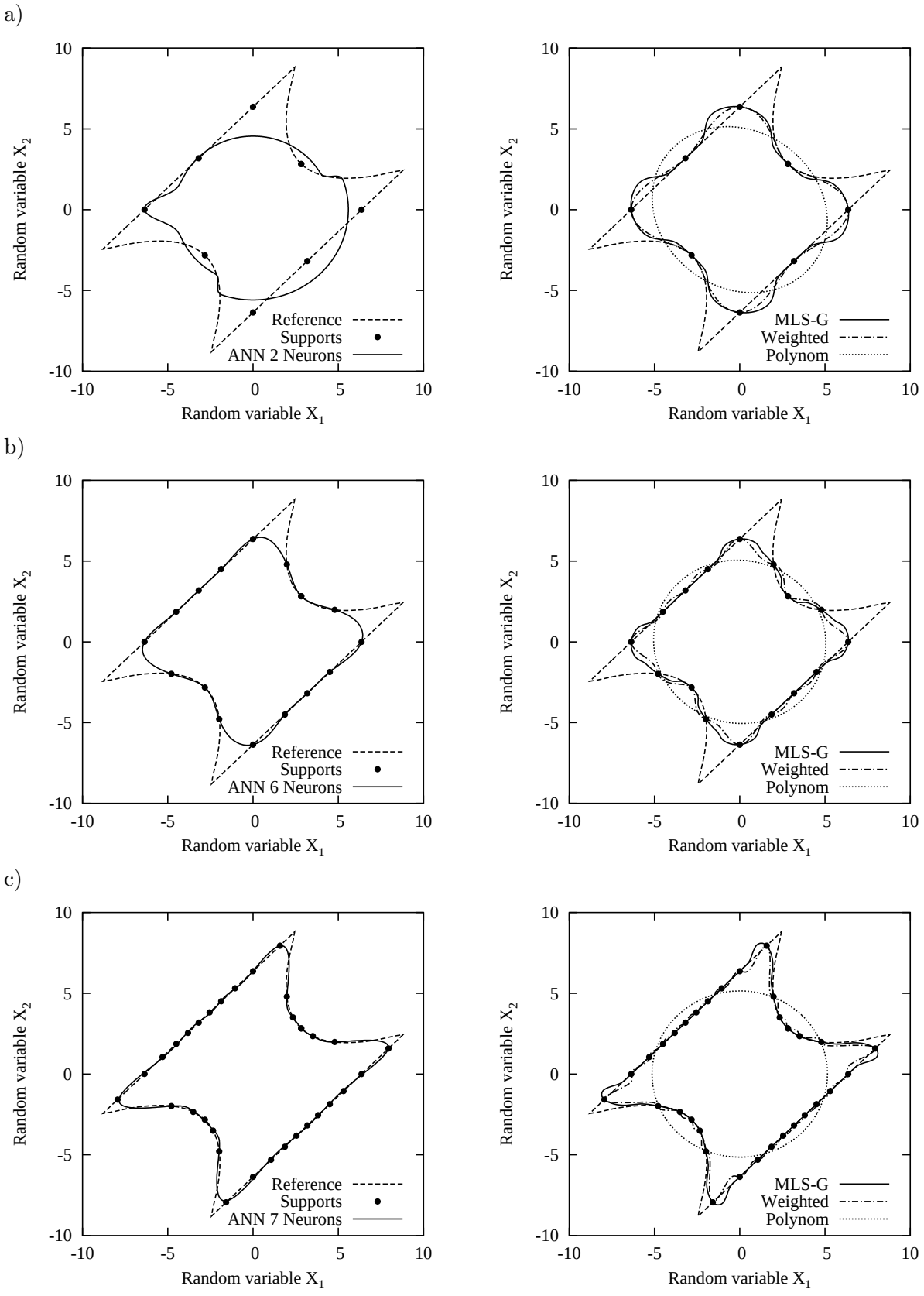

Figure 2: Limit state function of Katsuki and Frangopol (1994): original and approximations using a) 8, b) 16 and c) 32 regular support points

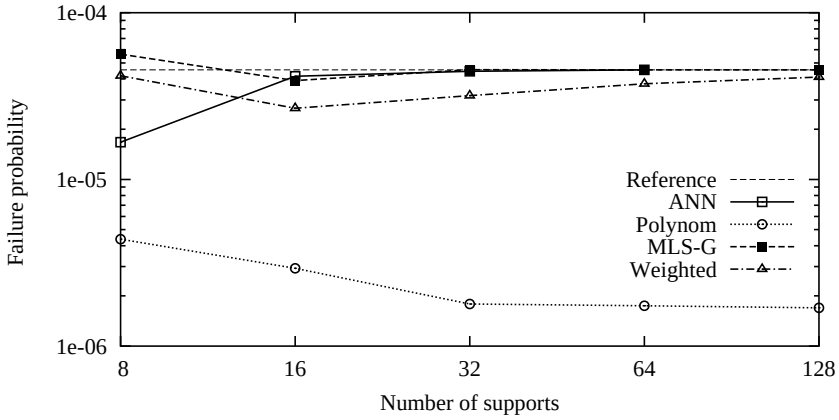

Figure 3: Estimated failure probabilities from 1000 directional samples using the approximated limit state function

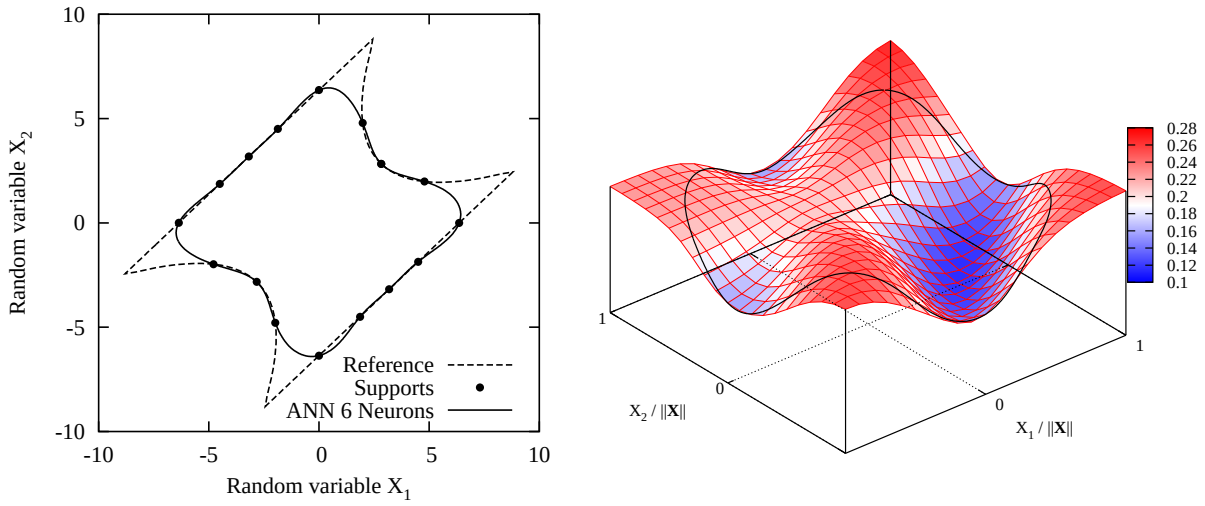

Figure 4: Approximated limit state function and approximated inverse radius

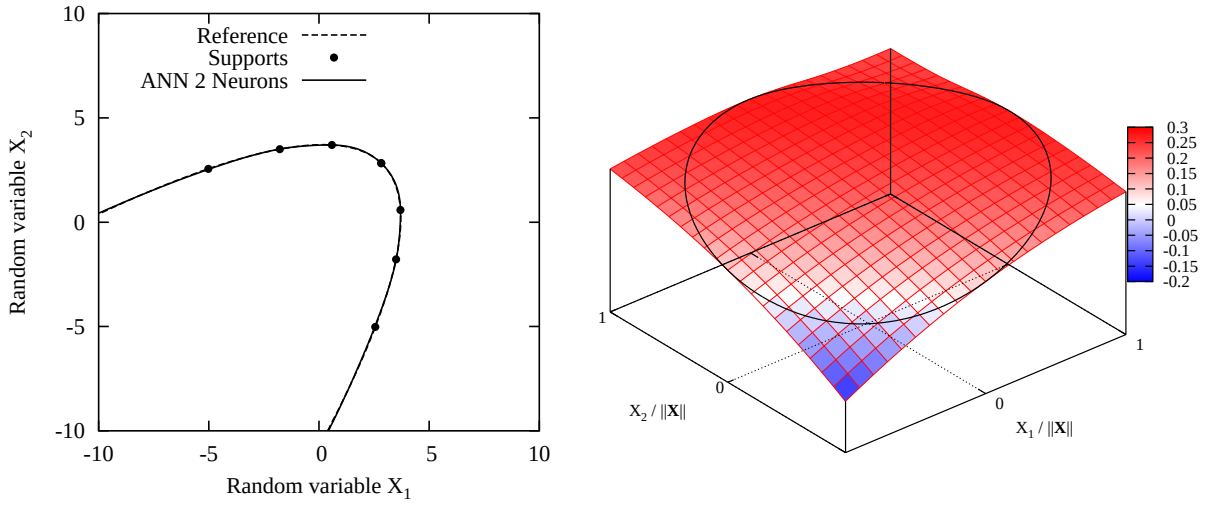

Figure 5: Approximated limit state function and inverse radius for a simpler function type

used in 2D, which is not enough for this function type. This is pointed out in Fig. 4, where an sufficiently approximated function is shown for 6 neurons. The black circle indicates the interesting approximated inverse radius depending on the direction.

For simpler limit state functions a small number of neurons gives very good results, which is demonstrated for a quadratic limit state function:

$$g(\mathbf{x}) = -0.1(x_1 - x_2)^2 - (x_1 + x_2)/\sqrt{2} + \beta_1 \tag{4}$$

In Fig. 5 the original and the approximated limit state function is shown, whereby an excellent agreement is reached with only 2 neurons and 10 support points. This is caused by the much simpler distance function shown additionally in Fig. 5. The failure probability for this function obtained again with 1000 directional samples reads $\hat{P}_{ref}(F) = 3.0439 \cdot 10^{-4}$ and the approximated value by using the 2 neurons and 10 support points is $\hat{P}_{approx}(F) = 3.0536 \cdot 10^{-4}$ agrees almost exactly with this solution. For this function type an unbounded region exists, thus the approximation works only sufficiently using the inverse radius as output quantity.

3.2 Crack growth in a plain concrete beam

3.2.1 Stochastic model and probabilistic analysis

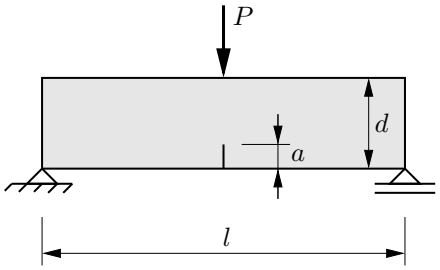

Figure 6: Three point bending beam with initial crack (Carpinteri et.al. 1986)

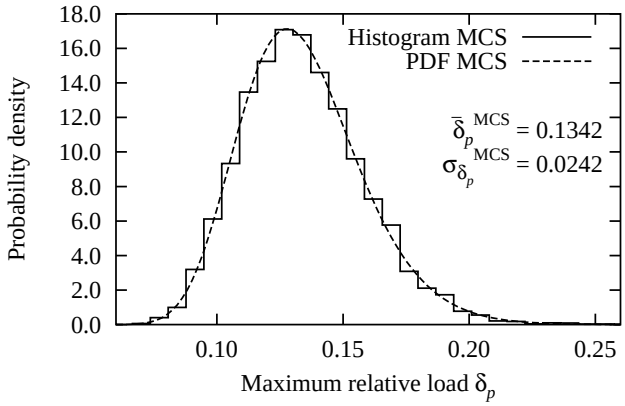

Figure 7: Histogram of the MCS samples and probability density function

In this example the three-point bending beam with initial crack shown in Fig. 6, which was analyzed deterministically in Carpinteri et al. (1986), is investigated by assuming random material properties. For this purpose a multi-parameter random field proposed by Most and Bucher (2005) with lognormal distribution is used to model the Young's modulus E, the tensile strength f_t and the fracture energy G_f as correlated parameters. The mean values of these three quantities are taken as $\bar{E} = 3.65 \cdot 10^{10} N/m^2$, $\bar{f}_t = 3.19 \cdot 10^6 N/m^2$ and $\bar{G}_f = 100 Nm/m^2$. The correlation length, the coefficients of variation (COV) and the parameter correlation coefficients are taken as $l_H = 0.6m$, $\text{COV}_E = \text{COV}_{f_t} = \text{COV}_{G_f^I} = 0.2$ and $\hat{\rho}_{12} = \hat{\rho}_{13} = \hat{\rho}_{23} = 0.8$, respectively. The other material and geometrical parameters are assumed to be deterministic as follows: Poisson's ratio

Approximation type	Sampling points		Approximation error ϵ		
			ϵ_{max}	$\bar{\epsilon}$	$\bar{\beta}$
Neural network 2 Neurons	LHS50	(10000)	0.2537	0.0106	0.0285
Neural network 6 Neurons	LHS200	(10000)	0.0903	0.0026	0.0100
Neural network 6 Neurons	LHS200W	(10000)	**0.0188**	**0.0019**	**0.0035**
RSM linear polynomial	LHS50	(10000)	0.2773	0.0251	0.1036
RSM linear polynomial	LHS200	(10000)	0.2792	0.0249	0.1037
RSM linear polynomial	LHS200W	(10000)	0.2082	0.0508	0.0604
RSM quadratic polynomial	LHS50	(10000)	0.1641	0.0201	0.0293
RSM quadratic polynomial	LHS200	(10000)	0.1347	0.0189	0.0307
RSM quadratic polynomial	LHS200W	(10000)	**0.1626**	**0.0195**	**0.0227**
RSM cubic polynomial	LHS50	(10000)	1.4873	0.0398	0.1603
RSM cubic polynomial	LHS200	(10000)	0.1768	0.0197	0.0319
RSM cubic polynomial	LHS200W	(10000)	0.1517	0.0189	0.0250
RSM MLS-G linear base	LHS50	(10000)	0.4728	0.0237	0.0492
RSM MLS-G linear base	LHS200	(10000)	0.1715	0.0202	0.0311
RSM MLS-G linear base	LHS200W	(10000)	0.2381	0.0237	0.0351
RSM MLS-G quadratic base	LHS50	(10000)	0.6601	0.0242	0.0623
RSM MLS-G quadratic base	LHS200	(10000)	0.1818	0.0195	0.0301
RSM MLS-G quadratic base	LHS200W	(10000)	0.1844	0.0212	0.0305

Table 1: Approximation errors of the maximum relative load using neural networks and response surfaces with different numbers of training points

$\nu = 0.1$, beam length $l = 0.6m$, beam height $d = 0.15m$, beam thickness $t = 1.0m$ and initial crack length $a_0 = 0.045m$.

The initial random field, which contains 4800 random variables (40×10 four-node finite elements, each having 4 integration points with 3 random parameters) is modeled with the largest 30 eigenvalues and belonging eigenvectors, which is equivalent to a representation of $Q = 95.84\%$.

First a stochastic analysis is carried out by calculating the load displacement curves of 10000 plain Monte Carlo samples. The cracking process is modeled by a coupled discretization with a meshless zone in the cracking area and standard finite elements in the undamaged part of the beam. More details about the crack growth algorithm can be found in Most (2005) and Most and Bucher (2005). The histogram of the maximum relative load determined from these MCS results is shown in Fig. 7. Additionally the probability density function (PDF) by assuming a lognormal distribution is shown using the indicated mean value and standard deviation obtained from the 10000 samples. The histogram and the PDF agree very well, thus the distribution type of the relative load is nearly lognormal.

3.2.2 Approximation of the maximum relative load

In the next step an approximation of the maximum relative load is performed using neural networks trained with 50, 200 and 200 wide-spanned LHS samples. The wide-spanned LHS training is obtained by stretching the original space of the random variable

distributions artificially to its double size.

The networks are defined using all 30 independent Gaussian random field variables as input values and one output value. Only 2 neurons are used together with the 50 training samples and 6 neurons for the 200 samples in order to fulfill Eq. 2. In Table 1 the calculated approximation errors are given. The approximation error is quantified by the maximum error ϵ_{max}, the mean error $\bar{\epsilon}$ and the mean error $\bar{\beta} = \bar{\epsilon}_{F<0.01 \cap F>0.99}$ which quantifies the error for all samples with distribution function values less than 1% and larger than 99%. Thus $\bar{\beta}$ gives an information about the approximation error for the regions with very low probability. The table indicates, that with increasing number $\bar{\epsilon}$, ϵ_{max} and $\bar{\beta}$ are reduced. By using the wide-spanned training ϵ_{max} and $\bar{\beta}$ are smaller than using standard LHS, which clarifies, that the approximation works well for the whole distribution space. In Fig. 8 this is shown more clearly. Due to the wide-spanned training the approximation of events with very low probability is much better than with the standard training. This is displayed in Fig. 9 in terms of the probability distribution function. The figures shows additionally the lognormal probability distribution functions using the statistical values of the 10000 MCS samples. The approximated probability distribution of 1000000 ANN samples with wide-spanned LHS200 training agrees very well with this function, thus the approximation of very small probabilities with the presented neural network concept seems to lead to very good results.

The excellent neural network results are compared to a response surface approximation using a global polynomial and the MLS interpolation. For this purpose only the random variables are used which have a correlation coefficient together with the response quantity larger than ten percent. Since only four random variables are assumed to be important a third order polynomial could be used for the approximation. The MLS approximation are carried out using the exponential weighting function with $D = 2.7$ for the linear base and $D = 4.7$ for the quadratic base. In Table 1 the errors of all response surface approximations are given, whereby the smallest approximation errors, which are obtained with the quadratic polynomial, are one order of magnitude larger than these of the neural network approximation.

3.2.3 Approximation of the random load displacement curves

In the final analysis of this example the neural network approximation of the complete nonlinear response is investigated. For this purpose the load values at ten fixed displacement values are used as output values while the 30 random variables are taken again as input variables of the network. 10 neurons are used in the hidden layer. As training samples the results of the 200 wide-spanned LHS samples are taken. In Fig. 10 the statistical load displacement curves obtained with the 10000 MCS samples and with the corresponding neural network approximation are shown. In Table 2 the maximum and the mean error of the approximated load for each displacement value are shown. These errors are normalized by the deterministic peak load in order to obtain comparable values. The table clearly indicates, that the approximation of the complete nonlinear response of the investigated structure leads to very good results. For the larger displacement values, where the structure is in the post-peak part of the load displacement curve, the maximum error increases, but the mean error stays in the same range. This means that although

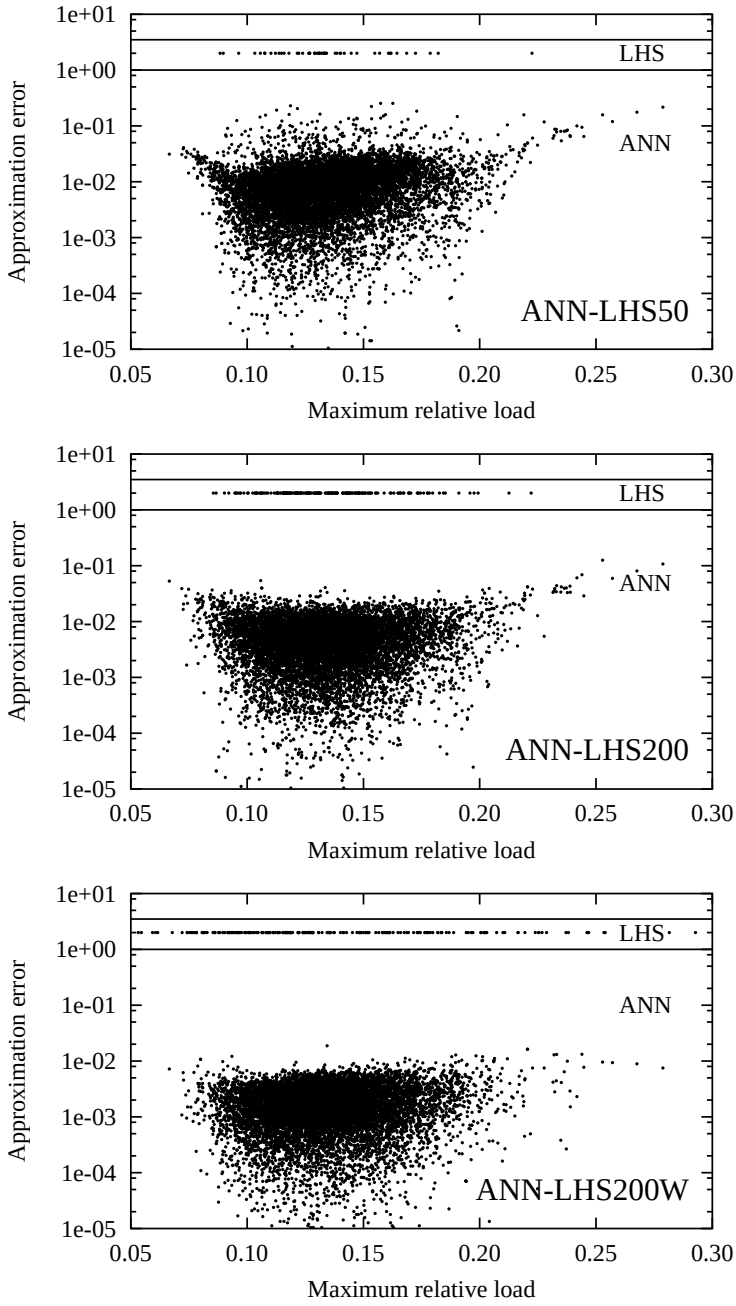

Figure 8: Approximation errors of the 10000 ANN samples using 50, 200 and 200 wide-spanned LHS training points with indicated range of the LHS samples

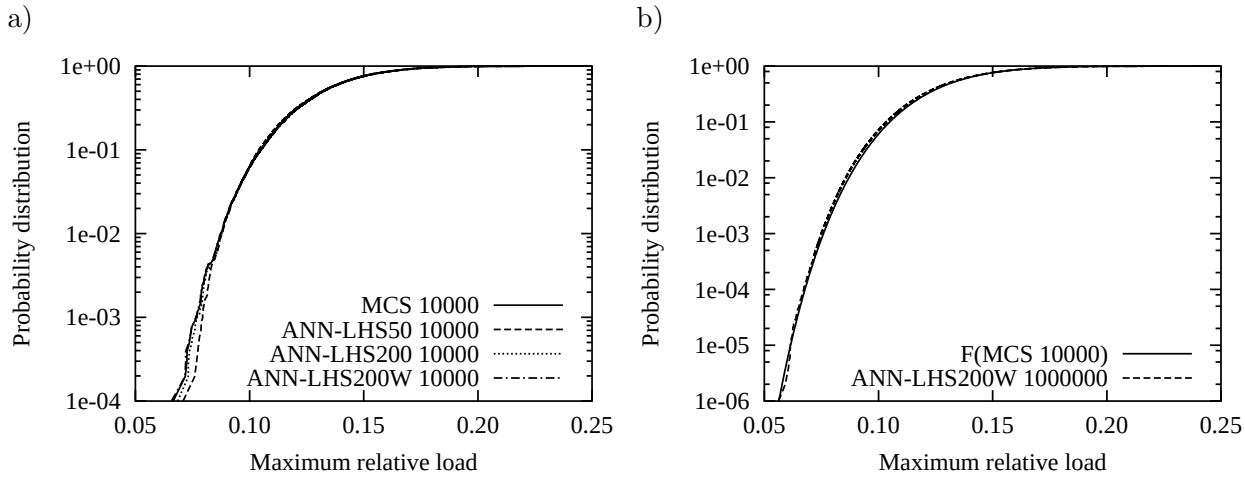

Figure 9: Probability distribution obtained by a) ANN approximation and using b) the statistical values and a lognormal distribution type

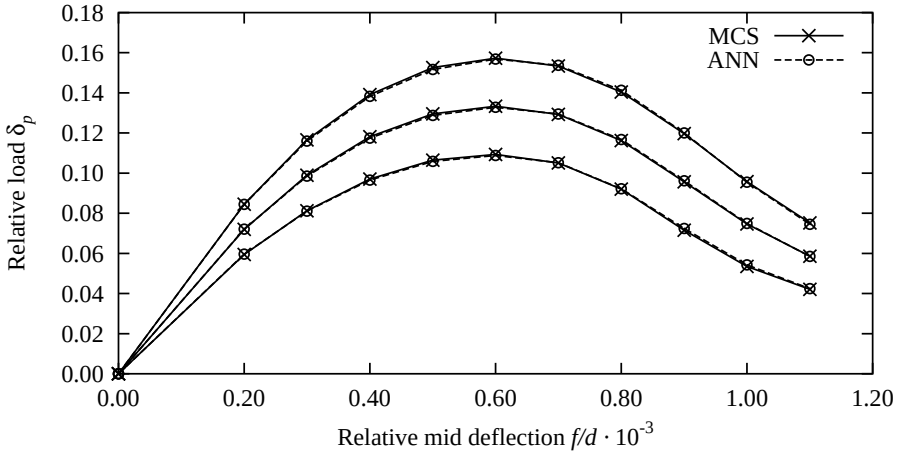

Figure 10: Statistical load displacement curves of the MCS samples and neural network approximation (mean values and mean values ± standard deviations)

the input parameters of the neural network describe only the initial state of the structure, the network approximation can represent the strongly nonlinear softening process due to cracking. A response surface approximation of this problem would require a much higher number of samples since most of the 30 random variables have a significant influence on at least some load values.

Relative mid deflection	Maximum error	Mean error
0.0002	0.02455	0.00227
0.0003	0.03133	0.00314
0.0004	0.03414	0.00555
0.0005	0.03457	0.00654
0.0006	0.04840	0.00516
0.0007	0.13592	0.00563
0.0008	0.14354	0.01044
0.0009	0.13899	0.01102
0.0010	0.12804	0.01145
0.0011	0.20828	0.00820

Table 2: Approximation errors of the relative external force values using the neural network

3.3 Identification of the parameters of a complex interface material model for concrete

Within this final example the neural network approach is used to identify the parameters of a material model for cohesive interfaces proposed by Carol et al. (1997). The model has twelve parameters: elastic stiffness and strength, each in normal and tangential direction, Mode-I and Mode-IIa fracture energy, friction coefficient, dilatancy stress and four shape parameters describing normal and tangential softening and the dilatancy behavior depending on the stress and the damage state.

In the first step the pure tensile softening is analyzed. For this purpose the normal stress curve used as reference is obtained with the following parameters: tensile strength $\chi_0 = 3 \cdot 10^6 N/m^2$, Mode-I fracture energy $G_f^I = 100 Nm/m^2$ and the shape parameter for tensile softening $\alpha_\chi = 0$. The corresponding softening curve depending on the irreversible crack displacements is shown in Fig. 11. All other parameters do not have an influence on the softening curve. Eleven regular points on this curve have been used as input values and the three parameters as output values for the neural network approximation. In Fig. 11 the curve from the identified parameters using 100 uniformly distributed LHS training samples and 15 hidden neurons is shown additionally. This curve shows a very good agreement with the reference solution, which clarifies that the neural network approximation works very well for this problem.

In the next step all twelve parameters together have been identified by means of a combined normal-shear-cracking test carried out by Hassanzadeh (1990). In Fig. 12 the measured curve for the normal and shear stress depending on the crack opening and sliding are shown. Additionally the curves obtained from the parameters identified by Carol et al. (1997) using an optimization strategy are displayed. The parameters are given in Table 3. The neural network approximation has been carried out using 150 uniformly distributed

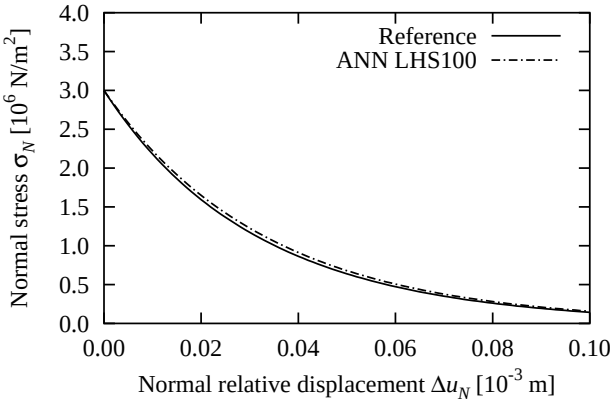

Figure 11: Tension softening curve from original and identified parameters

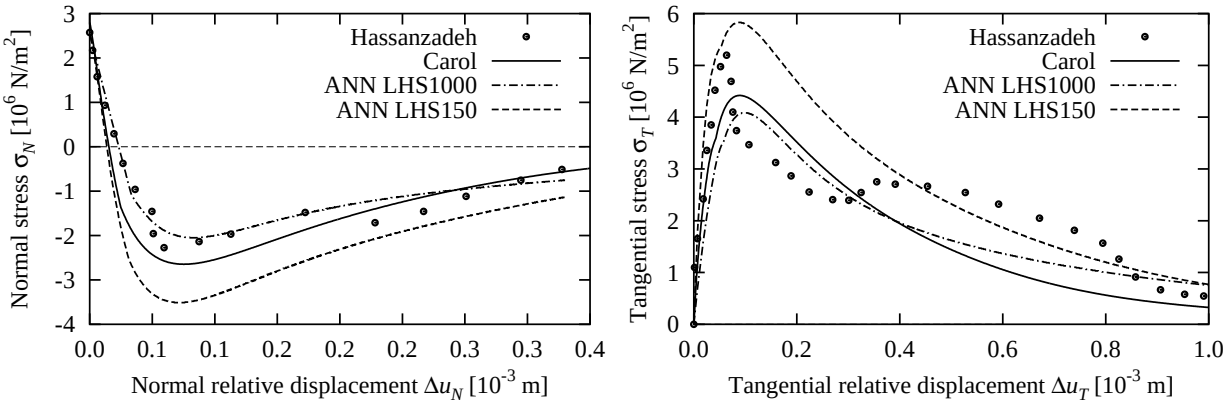

Figure 12: Normal and tangential stresses depending on the crack opening and sliding; Experimental and numerical curves with identified parameters

LHS samples with 4 hidden neurons and 1000 LHS samples with 29 hidden neurons. The training bounds and the identified parameters using 10 points from each curve are given in Table 3. Fig. 12 shows the curves belonging to these parameters. With 150 training samples the estimated parameters do not lead to a sufficient agreement of the normal and tangential stresses. If 1000 samples are used the agreement is much better and the resulting curves close to these obtained by Carol et al. (1997). But a very good agreement with the experimental curves could not be achieved, which might be caused on the one hand by the ill-conditioned experimental curve with softening and hardening, which can not represented by the material model. On the other hand the complexity of the problem, which needs much more neurons for a very good representation, might by another reason. With 4 neurons the minimum mean square error of the training was about 0.23 and with 29 neurons about 0.13, which shows that the training fitting improves with increasing number of samples and neurons, but for a very good fitting with a mean square error for example below 10^{-5} a much larger number of neurons and belonging training samples might be necessary. Although the neural network approximation leads to similar results as the application of classical optimization strategies.

			Carol	Training	ANN LHS150	ANN LHS1000
Normal stiffness	k_N	$[10^9 N/m^3]$	200	$100 - 500$	314	155
Tangential stiffness	k_T	$[10^9 N/m^3]$	200	$100 - 500$	301	102
Tensile strength	χ_0	$[10^6 N/m^2]$	2.8	$2.0 - 3.5$	2.5	2.6
Shear strength	c_0	$[10^6 N/m^2]$	7.0	$4.0 - 10.0$	8.0	9.9
Mode-I fract. energy	G_f^I	$[Nm/m^2]$	100	$50 - 300$	156	105
Mode-IIa fract. energy	G_f^{IIa}	$[Nm/m^2]$	1000	$500 - 2000$	1670	1590
Dilatancy stress	σ^{dil}	$[10^6 N/m^2]$	56	$20 - 100$	65	76
Friction coefficient	$\tan\phi$	[-]	0.9	$0.5 - 1.0$	0.83	0.87
Shape parameter	α_χ	[-]	0.0	$-1.0 - 1.0$	0.04	0.11
Shape parameter	α_c	[-]	1.5	$0.0 - 3.0$	1.33	2.16
Shape parameter	α_σ^{dil}	[-]	2.7	$0.0 - 6.0$	2.53	3.57
Shape parameter	α_c^{dil}	[-]	3.0	$0.0 - 6.0$	2.91	3.29

Table 3: Identified parameters and training bounds

4 Conclusions

The paper has shown that neural networks are a powerful tool for approximation purposes. For problems of lower dimension similar results as with the common response surface approach could be achieved. For higher-dimensional problems much better results could be obtained, since a smaller number of supports is necessary for a high order approximation. The applicability for identification problems could be shown, but for very complex models this application requires a large number of neurons and belonging training samples. Because of this reason optimization strategies might be more efficient for such tasks.

Acknowledgement

This research has been supported by the German Research Council (DFG) through Collaborative Research Center 524, which is gratefully acknowledged by the author.

References

Bjerager, P.: Probability Integration by Directional Simulation. In: *Journal of Engineering Mechanics* 114 (1988), S. 1288–1302

Cabral, S. V. S. ; Katafygiotis, L. S.: Neural network based response surface method and adaptive importance sampling for reliability analysis of large structural systems. In: Corotis, R.B. (Eds.) ; Schuëller, G.I. (Eds.) ; M., Shinozuka (Eds.): *Proc. 8th Intern. Conf. on Structural Safety and Reliability, Newport Beach, USA, June 17-22, 2001, CD-ROM.* Rotterdam : Balkema, 2001

Carol, I. ; Prat, P. C. ; López, C. M.: Normal/Shear Cracking Model: Application

to discrete crack analysis. In: *Journal of Engineering Mechanics ASCE* 123 (1997), S. 765–773

Carpinteri, A. ; Di Tommaso, A. ; Fanelli, M.: Influence of material parameters and geometry on cohesive crack propagation. In: Wittmann, F.H. (Eds.): *Fracture Toughness and Fracture Energy of Concrete - Proc. Int. Conf. on Fracture Mechanics of Concrete, Lausanne, Switzerland, October 1-3, 1985*. Amsterdam : Elsevier, 1986

Demuth, H. ; Beale, M.: *MATLAB Neural network toolbox*. 4. The Mathworks Inc., 2002

Deng, J. ; Desheng, G. ; Xibing, L. ; Zhong, Q. Y.: Structural reliability analysis for implicit performance functions using artificial neural networks. In: *Structural Safety* 27 (2005), S. 25–48

Gomes, H. M. ; Awruch, A. M.: Comparison of response surface and neural network with other methods for structural reliability analysis. In: *Structural Safety* 26 (2004), S. 49–67

Hagan, M. T. ; Demuth, H. B. ; Beale, M.: *Neural Network Design*. PWS Publishing Company, 1996

Hassanzadeh, M.: Determination of fracture zone properties in mixed mode I and mode II. In: *Engineering Fracture Mechanics* 35 (1990), S. 845–853

Hurtado, J. E.: Analysis of one-dimensional stochastic finite elements using neural networks. In: *Probabilistic Engineering Mechanics* 17 (2002), S. 35–44

Hurtado, J. E. ; Alvarez, D. A.: Neural-network-based reliability analysis: a comparative study. In: *Computer Methods in Applied Mechanics and Engineering* 191 (2001), S. 113–132

Katsuki, S. ; Frangopol, D. M.: Hyperspace Division Method for Structural Reliability. In: *Journal of Engineering Mechanics* 120 (1994), S. 2405–2427

Lancaster, P. ; Salkauskas, K.: Surface generated by moving least squares methods. In: *Mathematics of Computation* 37 (1981), S. 141–158

Lehký, D. ; Novák, D.: Identification of material model parameters using stochastic training of neural network. In: Walraven (Eds.) ; others (Eds.): *Proc. 5th Int. PhD Symposium in Civil Engineering, Delft, Netherlands, June 16-19, 2004*. Rotterdam : Balkema, 2004

MacKay, D. J. C.: Bayesian interpolation. In: *Neural Computation* 4 (1992), S. 415–447

Møller, M.: A scaled conjugate gradient algorithm for fast supervised learning. In: *Neural Networks* 6 (1993), S. 525–533

Most, Thomas: *Stochastic crack growth simulation in reinforced concrete structures by means of coupled finite element and meshless methods.* Germany, Bauhaus-University Weimar, PhD Thesis, 2005

Most, Thomas ; Bucher, Christian: Stochastic simulation of cracking in concrete structures using multi-parameter random fields. In: *International Journal of Reliability and Safety, submitted for publication* (2005)

Nie, J. ; Ellingwood, B. R.: A new directional simulation method for system reliability. Part II: application of neural networks. In: *Probabilistic Engineering Mechanics* 19 (2004), S. 437–447

Nie, J. ; Ellingwood, B. R.: A new directional simulation method for system reliability. Part II: application of neural networks. In: *Probabilistic Engineering Mechanics* 19 (2004), S. 437–447

Papadrakakis, M. ; Papadopoulos, V. ; Lagaros, N. D.: Structural reliability analysis of elastic-plastic structures using neural networks and Monte Carlo Simulation. In: *Computer Methods in Applied Mechanics and Engineering* 136 (1996), S. 145–163

Schueremans, L. ; Van Gemert, D.: Benefit of splines and neural networks in simulation based structural reliability analysis. In: *Structural Safety* 27 (2005), S. 246–261