ELSEVIER

Comput. Methods Appl. Mech. Engrg. 136 (1996) 145–163

Computer methods in applied mechanics and engineering

Structural reliability analyis of elastic-plastic structures using neural networks and Monte Carlo simulation

Manolis Papadrakakis*, Vissarion Papadopoulos, Nikos D. Lagaros

Institute of Structural Analysis and Seismic Research, National Technical University of Athens, Athens 15773, Greece

Received 10 October 1995

Abstract

This paper examines the application of Neural Networks (NN) to the reliability analysis of complex structural systems in connection with Monte Carlo Simulation (MCS). The failure of the system is associated with the plastic collapse. The use of NN was motivated by the approximate concepts inherent in reliability analysis and the time consuming repeated analyses required for MCS. A Back Propagation algorithm is implemented for training the NN utilising available information generated from selected elasto-plastic analyses. The trained NN is then used to compute the critical load factor due to different sets of basic random variables leading to close prediction of the probability of failure. The use of MCS with Importance Sampling further improves the prediction of the probability of failure with Neural Networks.

1. Introduction

The theory and methods of structural reliability have developed significantly during the last twenty years and have been documented in an increasing number of publications. These advancements in structural reliability theory and the attainment of more accurate quantification of the uncertainties associated with structural loads and resistances have stimulated the interest in the probabilistic treatment of structures. The reliability of a structure or its probability of failure is an important factor in the design procedure since it investigates the probability of the structure to successfully complete its design requirements. Reliability analysis leads to safety measures that a design engineer has to take into account due to the aforementioned uncertainties. Although from a theoretical point of view the field has reached a stage where the developed methodologies are becoming widespread, from a computational point of view serious obstacles have been encountered in practical implementations.

First and second order reliability methods that have been developed to estimate structural reliability [1–5] lead to elegant formulations requiring prior knowledge of only the means and variances of the component random variables and the definition of a differentiable failure function. For small-scale problems these type of methods prove to be very efficient, but for large-scale problems and/or large numbers of random variables Monte Carlo Simulation (MCS) methods seem to be superior. In fact, simulation methods are the only methods available to treat practical reliability problems. The Basic MCS is simple to use, but for typical structural reliability problems the computational effort involved becomes excessive because of the enormous sample size and the CPU time required for each Monte Carlo run. To reduce the computational effort, more elaborate simulation methods, called variance

* Corresponding author.

PII S0045-7825(96)01011-0

reduction techniques, have been developed. Despite the improvement in the efficiency of the basic MCS variance reduction techniques, they still require disproportionate computational effort for treating practical reliability problems. This is the reason why very few successful numerical investigations are known in estimating the probability of failure and are mainly concerned with simple elastic frames and trusses [6–8].

The use of artificial intelligence techniques, such as Neural Networks (NN), to predict analysis outputs has been studied previously in the context of optimal design of structural systems [10–12], fracture mechanics [13] and adaptive mesh generation [14]. The principal advantage of a properly trained NN is that it requires a trivial computational effort to produce an acceptable approximate solution. Such approximations appear to be valuable in situations where the actual response computations are CPU intensive and a quick estimation is required. Structural loads and material properties can be considered as time dependent or independent, while the failure domain can be considered as time variant or invariant. In the present study a time invariant structural reliability analysis in conjunction with NN is performed. The use of NN was motivated by the approximate concepts inherent in reliability analysis and the time consuming repeated analyses required for MCS. The suitability of NN predictions is investigated in evaluating the probability of failure of real scale plane and space framed structures.

An NN is trained first utilising available information generated from selected elasto-plastic analyses. The limit state analysis data was processed to obtain input and output pairs which were used to produce a trained NN. The trained NN is then used to predict the critical load factor due to different sets of basic random variables. After the critical load factors are predicted, the probability of failure is calculated by means of MCS in order to produce acceptable results. The predicted values for the critical load factors should resemble closely, though not identically, to the corresponding values of the limit state analyses which are considered 'exact'. The NN type considered here is based on the feed-forward error back-propagation training algorithm [15]. The 'exact' limit state analysis required to train the NN are generated using a first order analysis approach in conjunction with efficient solution techniques developed in [16, 17], for plane and space frames, respectively. It appears that the use of a properly selected and trained NN can eliminate any limitation on the sample size used for MCS and on the dimensionality of the problem, due to the drastic reduction of the computing time required for the repeated analyses. Furthermore, the use of importance sampling leads, in most cases, to considerable improvement in the quality of the NN results.

2. Time invariant structural reliability analysis

The inherent probabilistic nature of design parameters, material properties and loading conditions involved in structural analysis is an important factor that influences structural safety. Reliability analysis leads to safety measures that a design engineer has to take into account due to the aforementioned uncertainties. The probability of failure can be determined using the relationship

$$p_f = p[R < S] = \int_{-\infty}^{\infty} F_R(s) f_S(s)\, \mathrm{d}s \tag{1}$$

where R denotes the structure's resistance and S the loading. The randomness of R and S can be described by known probability density functions $f_R(s)$ and $f_S(s)$ respectively, with $F_R(s)$ being the cumulative probability density function of S. By defining a performance (failure) function $G(R, S)$, Eq. (1) may be alternatively expressed as

$$p_f = p[G(R, S) \leq 0] = \int_{G \leq 0} f_R(r) f_S(s)\, \mathrm{d}r\, \mathrm{d}s \tag{2}$$

In the case of structural reliability analysis the performance function is denoted by $G(R, S) = R - S$. For large and complex structural systems, where R is not known analytically because of the large number of combinations of events that can lead to structural failure, reliability analysis requires a great amount of numerical and computational effort, while the integral of Eq. (2) can only be calculated by approximate

means. Several probabilistic methods have been developed in the past to calculate the integral of Eq. (2). Among them the so-called exact methods and the First Order Second Moment (FOSM) [1] are of significant importance. Exact methods require that the probability density functions of all component variables are known prior to the analysis. Monte Carlo Simulation (MCS) belongs to that category.

2.1. The Monte Carlo simulation

In reliability analysis the MCS is often employed when the analytical solution is not attainable and the failure domain can not be expressed or approximated by an analytical form. This is mainly the case in problems of complex nature with a large number of basic variables where all the other methods are not applicable. Although the mathematical formulation of the MCS is relatively simple and the method has the capability of handling practically every possible case regardless of its complexity, the computational effort involved in conventional MCS is excessive. It is for this reason that a lot of sampling techniques, also called variance reduction techniques, have been developed in order to improve the computational efficiency of the method by reducing the statistical error inherent in Monte Carlo methods. Among them Importance Sampling and Conditional Expectation are of particular interest because of their potential of being very efficient [7, 18–21].

Expressing the limit state function as $G(X)$, where $X=(X_1, X_2, \ldots, X_n)$ is the vector of the basic random variables, Eq. (2) may be rewritten as

$$p_f = \int_{G(x)\leq 0} f_x(X)\,\mathrm{d}X \tag{3}$$

where $f_x(X)$ is the joint probability and $G(X)$ is an irregular domain with highly non-linear boundaries. Following the law of large numbers, an unbiased estimator of the probability of failure is given by

$$\bar{p}_f = \frac{1}{N}\sum_{i=1}^{N} I(X_i) \tag{4}$$

where $I(X_i)$ is an indicator defined as

$$I(X_i) = \begin{cases} 1 & \text{if } G(X_i)\leq 0 \\ 0 & \text{if } G(X_i)>0 \end{cases} \tag{5}$$

Accordingly, N independent random samples of a specific probability density function of the vector X are prepared and the failure function is computed for each sample X_i. If $G(X_i)\leq 0$ a successful simulation is counted. The Monte Carlo estimate of the probability of failure p_f can then be expressed in terms of sample mean as

$$\bar{p}_f = \frac{N_H}{N} \tag{6}$$

where N_H is the number of successful simulations and N the total number of simulations.

2.2. Importance sampling

In order to improve the computational efficiency of the MCS, without deteriorating the accuracy of the solution, a number of variance reduction techniques has been proposed among which the Importance Sampling (IS) is generally recognised as the most efficient [5, 22]. Denoting by $Y=(Y_1, Y_2, \ldots, Y_n)$ a second random vector, with $g_y(Y)$ being its known joint probability density function, Eq. (3) may be rewritten as

$$p_f = \int_{G(x)\leq 0} \frac{f_x(X)}{g_y(X)}\, g_y(X)\,\mathrm{d}x \tag{7}$$

where $g_y(X)$ is the importance sampling function. An unbiased estimator of Eq. (7) is now given by

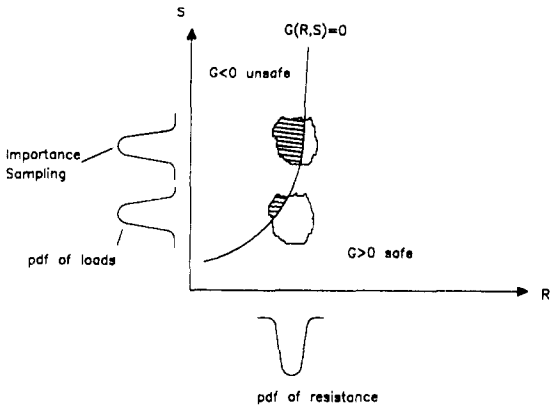


Fig. 1. Schematic representation of MCS and MCS with Importance Sampling.

$$\bar{p}_f = \frac{1}{N}\sum_{i=1}^{N} I(X_i)\frac{f_x(X_i)}{g_y(X_i)} \tag{8}$$

where X_i is generated according to $g_y(X_i)$. The Monte Carlo estimate of Eq. (8) may then be expressed in terms of sample mean as

$$\bar{p}_f = \sum_{i=1}^{N} \frac{S_i}{N} \tag{9}$$

where S_i is now a scale factor given by

$$S_i = \begin{cases} \dfrac{f_x(X_i)}{g_x(X_i)} & \text{if } G \leqslant 0 \\ 0 & \text{if } G > 0 \end{cases} \tag{10}$$

The selection of an appropriate important sampling density function is of critical importance for both the efficiency and the accuracy of the simulation. A successful choice of the Important Sampling function will reduce the variance of the estimation and accordingly will yield a reliable result in fewer steps while 'wrong' choice of the Importance Sampling function may lead to erroneous results. A schematic representation of MCS and MCS with Importance Sampling is shown in Fig. 1.

3. Limit elasto-plastic analysis

In this work the reliability analysis connected to a structural failure criterion of plane and space frames is examined. The failure criterion is considered to be the formation of a mechanism. The adopted incremental non-holonomic first order step-by-step limit analysis is based on the generalised plastic node concept proposed in [23, 24]. The non-linear yield surface is approximated by a multi-faceted surface, while the linear equilibrium equations at each load step are solved using the preconditioned conjugate gradient method [16, 17].

Under the assumption of concentrated plasticity all plastic deformations are confined to zero length plastic zones at the two ends of the member, leaving elastic the part of the member between the two plastic nodes. The materials are assumed to be elastic-perfectly plastic and the structural response is in the range of small displacements. The tangent elasto-plastic stiffness matrix used for the limit state analysis may be expressed as

$$K_{ep} = K_e - K_e\Phi\{\Phi^T K_e \Phi\}^{-1}\Phi^T K_e \tag{11}$$

in which K_{ep} is the elasto-plastic element stiffness matrix, K_e is the elastic element stiffness matrix, and

Φ is the gradient vector of the multi-faceted surface at the force point where a member end initiates the plastic behaviour. In this study the yield surface proposed in [23] is approximated by a piece-wise linear multi-faceted surface. For an efficient computer implementation a second internal yield surface of similar orientation (*homothetic*) and close to the first one is introduced in order to avoid unnecessary analysis steps [17].

The first-order step-by-step limit analysis adopted requires the computation of a number of successive linear solutions in which the overall stiffness matrix is slightly modified from one solution to the other. The total number of solutions corresponds to the total number of load increments required for the structure to become a mechanism. The change of stiffness from one step to the other is only due to the contribution of the elasto-plastic stiffness matrices of the elements with the newly formed or modified plastic nodes. These special features of the problem make the preconditioned conjugate gradient method (PCG) very attractive for the solution of the linear problem at each load increment. An important factor affecting the efficiency of the PCG is the preconditioning matrix. In this study two Cholesky type preconditioners are used for the PCG method. The first one is based on an incomplete Cholesky factorisation of the stiffness matrix [25] in which a rejection by magnitude factor controls the number of terms of the preconditioning matrix. Alternatively, a complete factored matrix is used as preconditioner and it is kept fixed for a number of steps before its reformulation [16, 17]. In addition to the PCG method, a direct solver based on a modified Cholesky factorisation is also employed to take into account the characteristic formulation aspects of the problem. Since the overall stiffness matrix changes gradually, with the successive formation of plastic nodes, the factorisation phase at each load increment is confined to the bottom right hand corner of the stiffness matrix, starting from the first node with a change in its stiffness value due to the plastic node formation at the end of one or more elements connected to that node.

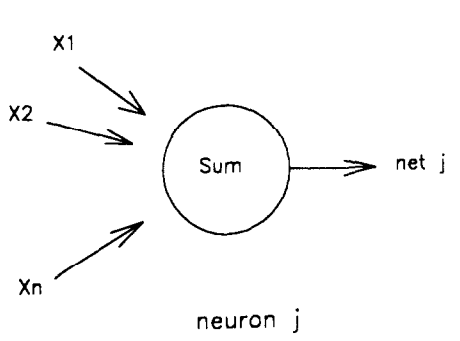
4. Application of neural networks

Only the basic ideas of NN will be discussed in this study. A more detailed introduction to NN may be found in [15, 26]. Neural net models of learning and the accumulation of expertise have found their way into practical applications in many areas. It appears that a number of computational structures technology applications, that are heavily dependent on extensive computer resources, have been investigated as demonstration of neural network capabilities [10, 27]. Reliability analysis of ultimate elastic plastic structural response using Monte Carlo Simulation is a highly intensive computational problem which makes conventional approaches incapable of treating real scale problems even in today's powerful computers. In the present study the use of NN was motivated by the approximation concepts inherent in reliability analysis. The idea here is to train a NN to provide computationally inexpensive estimates of analysis outputs required for the reliability analysis problem. The major advantage of a trained NN over the conventional process, under the provision that the predicted results fall within acceptable tolerances, is that results can be produced in a few clock cycles, representing orders of magnitude less computational effort than the conventional computational process.

4.1. Back Propagation learning algorithm

The basic model for a processing element is shown in Fig. 2. A neural network consists of multiple processing elements linked together. In a Back Propagation (BP) algorithm, learning is carried out when a set of input training patterns is propagated through a network consisting of an input layer, one or more hidden layers and an output layer as shown in Fig. 3. Each layer has its corresponding units (processing elements, neurons or nodes) and weight connections. A single training pattern is an i–o row vector of input–output values in the entire matrix of i–o training set.

The inputs x_i, $i = 1, 2, \ldots, n$ which are received by the input layer are analogous to the electro-chemical signals received by neurons in human brain. In the simplest model these input signals are multiplied by connection weights $w_{p,ij}$ and the effective input $\text{net}_{p,j}$ to elements is the weighted sum of the inputs



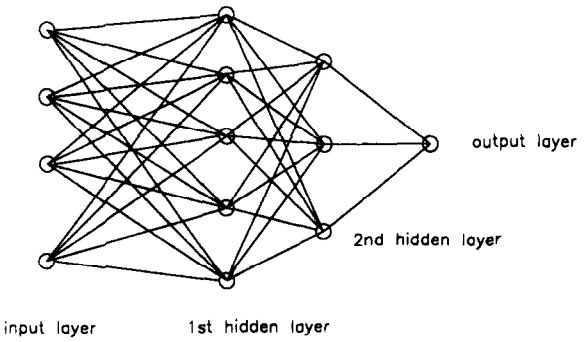


Fig. 2. Basic model for a processing element.

Fig. 3. A three fully connected NN configuration.

$$\text{net}_{p,j} = \sum_{i=1}^{n} w_{p,ij}\, \text{net}_{q,i} \tag{12}$$

where $w_{p,ij}$ is the connecting weight of the layer p from the i neuron in the q (source) layer to the j neuron in the p (target) layer, $\text{net}_{q,i}$ is the output produced at the i neuron of the layer q and $\text{net}_{p,j}$ is the output produced at the j neuron in the layer p, as shown in Fig. 4. Inputs x_i correspond to $\text{net}_{q,i}$ for the input layer.

In the biological system, a typical neuron may only produce an output signal if the incoming signal builds up to a certain level. This output is expressed in NN by

$$\text{out}_{p,j} = F(\text{net}_{p,j}) \tag{13}$$

where F is an activation function which produce the output at the j neuron in the p layer. The type of activation function that was used in the present study is the sigmoid function, given by the expression

$$F(\text{net}_{p,j}) = \frac{1}{1 + e^{-(\text{net}_{p,j} + b_{p,j})}} \tag{14}$$

where $b_{p,j}$ is a bias parameter used to modulate the element output. The principal advantage of the sigmoid function is its ability to handle both large and small input signals. The determination of the proper weight coefficients and bias parameters is embodied in the network learning process. The nodes are initialised arbitrarily with random weight and bias parameters.

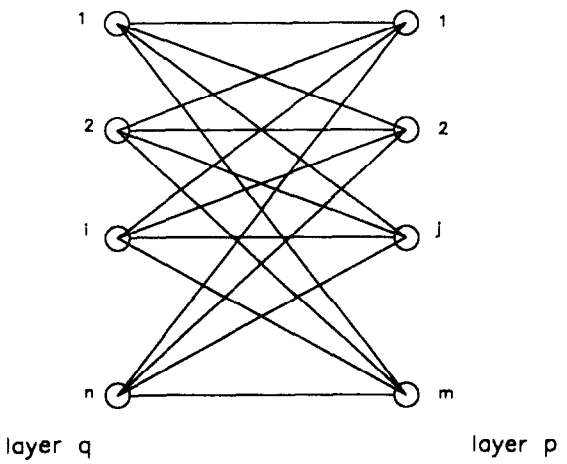


Fig. 4. Connection pattern between two layers.

At the output layer the computed output(s), otherwise known as the observed output(s), are subtracted from the desired or target output(s) to give the error signal

$$\mathrm{err}_{k,i} = \mathrm{tar}_{k,i} - \mathrm{out}_{k,i} \tag{15}$$

where $\mathrm{tar}_{k,i}$ and $\mathrm{out}_{k,i}$ are the target and the observed output(s) for the node i in the output layer k, respectively. This is called supervised learning. For the output layer the error signal, as given by Eq. (15), is multiplied by the derivative of the activation function, for the neuron in question, to obtain

$$\delta_{k,i} = \mathrm{d}F(\mathrm{net}_{k,i}) \cdot \mathrm{err}_{k,i} \tag{16}$$

while the derivative of the sigmoid function $\mathrm{d}F$ is given by

$$\mathrm{d}F(\mathrm{net}_{k,i}) = \mathrm{out}_{k,i} \cdot (1 - \mathrm{out}_{k,i}) \tag{17}$$

Subsequently, $\delta_{k,i}$ is used for the evaluation of the weight changes in the output layer k according to

$$\Delta w_{k,ji} = \eta \cdot \delta_{k,i} \cdot \mathrm{out}_{p,j} \tag{18}$$

where η denotes a learning rate coefficient usually selected between 0.01 and 0.9 and $\mathrm{out}_{p,j}$ denotes the output of the layer p immediately before the output layer. This learning rate coefficient is analogous to the step size parameter in the numerical optimisation algorithms.

The changes in the weights may alternatively be expressed according to [15] by

$$\Delta w_{k,ji}^{t+1} = \eta \cdot \delta_{k,i} \cdot \mathrm{out}_{p,j} + \alpha \cdot \Delta w_{k,ji}^{t} \tag{19}$$

where the superscript t denotes the cycle of the weight modification and α is the momentum term which controls the influence of the previous weight change. For the hidden layers the corresponding weight changes are given by

$$\delta_{p,j} = \mathrm{d}F(\mathrm{net}_{p,j}) \cdot \left(\sum_{i=1}^{n} \delta_{k,i} \cdot w_{k,ji} \right) \tag{20}$$

$$\Delta w_{p,lj}^{t+1} = \eta \cdot \delta_{p,j} \cdot \mathrm{out}_{r,l} + \alpha \cdot \Delta w_{p,lj}^{t} \tag{21}$$

where $\mathrm{out}_{r,l}$ denotes the output of the neuron l in the hidden layer r, $\Delta w_{p,lj}^{t}$ is the weight, changes between neuron l in the hidden layer r to neuron j in the hidden layer p.

After the evaluation of the weight changes the updated values of the weights given by $w_{p,ij}^{t+1} = w_{p,ij}^{t} + \Delta w_{p,ij}^{t+1}$, are used for the next training cycle. This process has to be repeated for all input training patterns until the desired level of error is obtained. The procedure used in this study is the single pattern training where all the weights are updated before next training pattern (training example) is processed.

4.2. The NN training

In our implementation the main objective is to investigate the ability of the NN to predict the collapse load by using the Back Propagation algorithm. This objective comprises the following tasks: (i) select the proper training set; (ii) find a suitable network architecture; (iii) determine the appropriate values of characteristic parameters, such as the learning rate and momentum term. The main limitation of an NN training algorithm is the fact that its efficiency depends on the correct learning rate, momentum term and network architecture. Unfortunately, there is little guidance on the selection of these parameters other than the experience which is based on a trial and error procedure. For the BP algorithm to provide good results the training set must include data over the entire range of the output space. The appropriate selection of input–output training data is one of the important factors in NN training. Although the number of training patterns may not be the only concern, the distribution of samples is of greater importance.

The output of the sigmoid function used in the conventional BP algorithm lies between 0 and 1. Thus, for Eq. (15) to produce meaningful results the output values of the training patterns should be

normalised within the same range. As the network is trained, the weights can become adjusted to very large values. This can force all or most of the neurons to operate at large output values in a region where the derivative of the activation function is very small. Since the correction of the weights depends on the derivative of the sigmoid function the network may come to virtual standstill. Initialising the weights to small random values would help to avoid this situation, however it is more appropriate to normalise the input patterns to be also between 0 and 1.

There are typically two types of networks, namely fully and patterned connected networks. In a fully connected network, as shown in Fig. 3, each unit in a layer is connected to all the units of the previous and the next layer. This type of network architecture is widely used. Alternatively, some local associativity between the units may be created or the number of connections may be reduced producing a patterned connected network. The number of neurons to be used in the hidden layers is not known in advance and usually is estimated by trial and error approach. At the first phase of learning it is convenient to gradually increase the number of hidden units and next, after achieving the desired convergence to try to remove some of them in order to find the minimal size of the network which performs the desired task [27].

The learning rate coefficient and the momentum term are two user defined BP parameters that effect the learning procedure of NN. The training is sensitive to the choice of these net parameters. The learning rate coefficient, employed during the adjustment of weights, is used to speed-up or slow-down the learning process. A bigger learning coefficient increases the weight changes, hence large steps are taken toward the global minimum of error level, while smaller learning coefficients increase the number of steps taken to reach the desired error level. If an error curve shows a downward trend but with poor convergence rate the learning rate coefficient is likely to be too high. Although these learning rate coefficients are usually taken to be constant for the whole net, local learning rate coefficients for each individual layer or unit may be applied as well.

In this work a fully connected network is used. The number of conventional step-by-step limit analysis calculations performed are in the range of 20 to 60, while 10 to 20 out of them are selected to give the pairs (inputs–outputs) for the NN training. This selection is based on the requirement that the full range of possible results should be represented in the training procedure. For the application of the NN simulation and for the selection of the suitable training pairs, the sample space for each random variable is divided into equally spaced distances. The central points within the intervals are used as inputs for the limit state analyses.

The basic NN configuration employed in this study is selected to have one hidden layer. Tests performed for more than one hidden layer showed no significant improvement in the obtained results. Based on this configuration various NN architectures are tested in order to find the most suitable in terms of the smallest prediction error. This is done either with a direct comparison of the predicted with the 'exact' results produced by the limit elasto-plastic analysis or by means of the Root Mean Square (RMS) error which is given by

$$e_{\mathrm{RMS}} = \sqrt{\frac{1}{N_P N_{\mathrm{out}}} \sum_{N_P} \sum_{i=1}^{N_{\mathrm{out}}} (\mathrm{tar}_i - \mathrm{out}_i)^2} \tag{22}$$

where N_P is the total number of i–o pairs in the training set and N_{out} is the number of output units. e_{RMS} gives a measure of the difference between predicted at each NN cycle and 'exact' values.

After the selection of the suitable NN architecture and the training procedure, the network is then used to produce predictions of the critical load factor corresponding to different values of the input random variables. The results are then processed by means of MCS or MCS with IS to calculate the probability of failure p_f.

4.3. NN based MCS for reliability analysis

In reliability analysis of elastoplastic structures using MCS the computed critical load factors are compared to the corresponding external loading leading to the computation of the probability of structural failure according to Eq. (5). By approximating the 'exact' solution with a NN prediction of

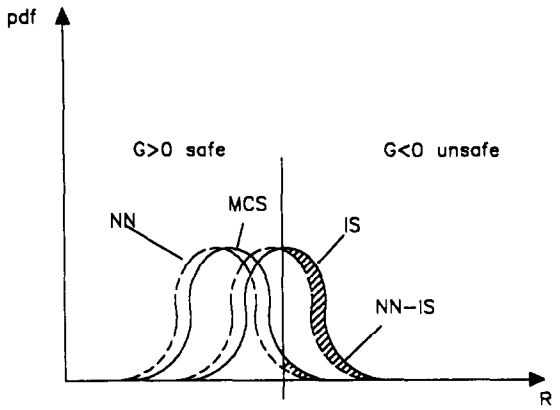


Fig. 5. Sensitivity of p_f prediction to different sample space of resistances.

the critical load factor, the accuracy of the predicted p_f depends not only on the accuracy of the NN prediction of the critical load factor but also on the sensitivity of p_f with regard to a slightly modified, due to the NN approximation, sample space of resistances. This sensitivity is represented in Fig. 5 by the ratio of the shaded area over the total area defined by the probability density distributions and the failure function on the unsafe side. It occurs that the error due to this sensitivity is always present but is more pronounced in low probability estimations where the shadowed area becomes significant compared to the total area that defines the low probability of failure. Thus, the use of Importance Sampling techniques is expected to be beneficial since the sampling is performed in an area of high probabilities. In this case, as the ratio of the shaded area over the total area is decreased the introduced error can be substantially reduced.

5. Numerical tests

Three test examples, one plane frame and two space frames, have been considered to illustrate the feasibility of the proposed methodology. The probability of failure is estimated using the Basic MCS and the MCS with Importance Sampling. Loading, yield stresses and plastic moduli are considered to be random variables. The loads acting on the structure follow a Log-normal probability density function, while random variables associated with material and section properties follow a normal probability density function. The failure condition is represented by the total collapse of the structure due to the successive formation of plastic nodes. Limit state analyses following the step-by-step approach described in Section 3 are considered 'exact' and are used to produce the training pairs required for the NN simulation. The step-by-step limit analysis has been also used to compute the 'exact' probability of failure with conventional Monte Carlo Simulation. The NN model used in this study is the back propagation algorithm NETS 2.01 [28] developed by NASA.

5.1. Example 1

The first test example is the five-storey plane frame shown in Fig. 6. A unit load is applied at the top storey of the frame. The dimensions and properties of the frame, the load–displacement curve until the formation of the collapse mechanism, as well as the plastic nodes at collapse, using the mean values of the basic random variables are also depicted in Fig. 6. Three test cases are considered for this example. In the first test case the yield stress is considered to be random variable with specified probability density function. In the second test case the plastic moduli of beams (Z_b) and columns (Z_c) are taken as additional but correlated random variables, requiring one random generation, while in the third test case the plastic moduli of beams (Z_b) and columns (Z_c) are considered to be independent random variables. Since Eq. (5) requires a comparison between the external loading and the critical load factor

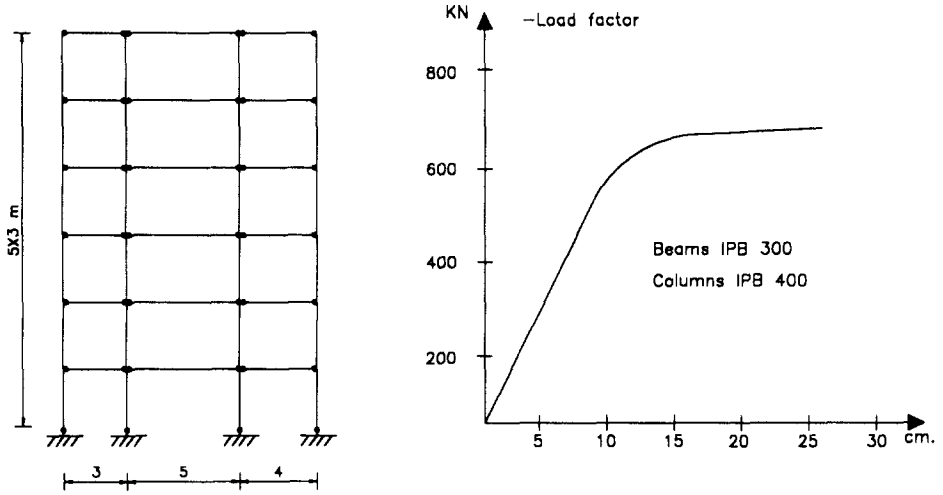


Fig. 6. Example 1. Five-story plane frame with data; loading; mode of failure and load–displacement curve.

representing the resistance of the structure, the external loading is also considered as a random variable for all test cases. When the external loading is less than the critical load factor, $I(x_i)$ of Eq. (5) takes the value of 1, otherwise it becomes 0. The type of probability density functions (PDF), mean values (μ) and standard deviations (σ) for all variables are presented in Table 1. A normal distribution is assumed for the Importance Sampling function $g_x(x)$ of the loading. The mean value of $g_x(x)$ is assumed to correspond to the failure load when all other random variables are kept to their mean values. The properties of the IS probability density functions are shown in Table 2 for all test cases.

Twenty values of the yield stress were used for test case one as input variables for the limit elastoplastic analysis. Ten of them were selected, together with their corresponding 'exact' critical load factors produced by the limit elastoplastic analysis, to be the training set. The remaining 'exact' pairs are used to test the accuracy of the NN prediction. For test cases 2 and 3 a similar procedure is adopted. In these cases 30 and 60 combinations of the 2 and 3 basic random variables were processed, while 13 and 19 of them were finally selected, together with their corresponding critical load factors, for training purposes, respectively.

Fig.7 demonstrates the performance of the NN configuration using different numbers of hidden units for the three test cases considered. It can be seen that the RMS error is reaching a plateau after a certain number of hidden units without any further improvement. In Table 3 three NN architectures with different number of hidden units are examined for each test case of this example in order to select the architecture that delivers good results as measured by the difference between predicted and 'exact'

Table 1
Example 1. Characteristics of random variables for Basic MCS

Random variables	PDF	μ	σ
σ_y (kN/cm^2)	N	24.0	2.4
Loads (kN)	Log-N	6.57	0.2
Z_b (cm^3)	N	1866.0	93.3
Z_c (cm^3)	N	2396.0	119.0

Table 2
Example 1. Characteristics of random variables for MCS-IS

Random variables	PDF	μ	σ
σ_y (kN/cm^2)	N	24.0	2.4
Loads	N	855.0	165.0

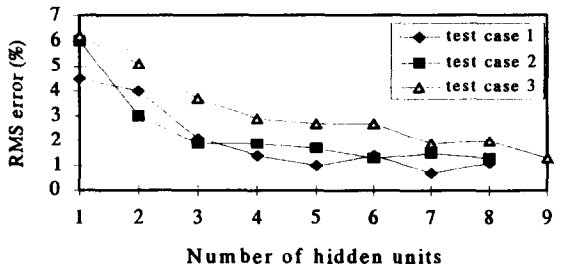


Fig. 7. Example 1. Performance of NN configuration using different number of hidden units.

Table 3
Example 1. Performance of varous NN architectures

Random variables	NN1				NN2				NN3			
	i–j–k	e_{RMS}	e_{MAX}	ε (%)	i–j–k	e_{RMS}	e_{MAX}	ε (%)	i–j–k	e_{RMS}	e_{MAX}	ε (%)
2	1–3–1	0.021	0.029	0.67	1–4–1	0.014	0.028	0.35	1–7–1	0.007	0.016	0.26
3	2–3–1	0.019	0.061	0.47	2–4–1	0.019	0.046	0.61	2–6–1	0.013	0.030	0.26
4	3–7–1	0.019	0.047	0.80	3–8–1	0.020	0.069	0.85	3–9–1	0.013	0.036	7.96

values of critical load factors. The symbols i–j–k correspond to the number of units at each layer (input–hidden–output) while ε gives the mean value of the error. The depicted results indicate that the selection of the best NN architecture can be based on the minimum achieved value of e_{RMS}. After the selection of the best NN architecture, the network was tested for different e_{RMS} tolerances, as shown in Table 4, in order to examine the existence of any overtraining effects before choosing the final trained NN. This table depicts the values of e_{RMS} and its corresponding error ε in the prediction. It can be observed that the lowest e_{RMS} produces the lowest error in the prediction. Hence, no overtraining effects are present in this study.

Once an acceptable trained NN in predicting the critical load factors is obtained, the probability of failure for each test case is estimated by means of NN based Monte Carlo Simulation using the Basic MCS and the MCS with IS. The results for various number of simulations are depicted in Tables 5–7 for

Table 4
Example 1. Performance of various training tolerances

Number of random variables	NN			
	i–j–k	e_{RMS}	e_{MAX}	ε (%)
2	1–7–1	0.024	0.047	0.65
	1–7–1	0.018	0.037	0.38
	1–7–1	0.011	0.025	0.29
	1–7–1	0.010	0.022	0.29
	1–7–1	0.008	0.017	0.26
3	2–6–1	0.028	0.094	0.78
	2–6–1	0.025	0.064	0.53
	2–6–1	0.021	0.045	0.49
	2–6–1	0.017	0.035	0.47
	2–6–1	0.016	0.032	0.43
	2–6–1	0.013	0.030	0.26
4	3–7–1	0.030	0.081	3.00
	3–7–1	0.022	0.065	2.10
	3–7–1	0.021	0.057	1.95
	3–7–1	0.020	0.051	1.80
	3–7–1	0.019	0.047	0.80

Table 5
Example 1—test case 1. 'Exact' and predicted values of p_f and the required CPU time ($i = 1$, $j = 7$, $k = 1$)

Number of simulations	'exact' Basic MCS p_f (%)	NN Basic MCS p_f (%)	'exact' MCS-IS p_f (%)	NN Basic-IS p_f (%)
50	6.00	2.00	8.18	6.72
100	7.00	5.00	7.45	6.28
300	8.00	5.33	7.88	6.72
500	8.20	5.40	8.61	7.30
1000	8.60	5.90	8.55	7.38
5000	8.48	5.98	8.46	7.40
10 000	8.36	5.93	8.44	7.40
50 000		6.04		7.38
100 000		6.04		7.38
CPU time in seconds				
Pattern selection	–	6	–	6
Training	–	4	–	4
Propagation	–	20	–	20
Total	31 320[a]	30	3132[b]	30

[a] Projected after 100 000 simulations.
[b] For 10 000 simulations.

Table 6
Example 1—test case 2. 'Exact' and predicted values of p_f and the required CPU time ($i = 2$, $j = 6$, $k = 1$)

Number of simulations	'exact' Basic MCS p_f (%)	NN Basic MCS p_f (%)	'exact' MCS-IS p_f (%)	NN Basic-IS p_f (%)
50	10.00	4.00	8.74	8.11
100	9.00	8.00	8.89	7.96
300	8.33	8.67	8.89	8.06
500	8.40	8.00	8.86	8.14
1000	8.79	8.00	8.73	8.01
5000	8.76	7.96	8.83	8.09
10 000	8.90	7.67	8.89	8.10
50 000		7.74		8.12
100 000		7.74		8.13
CPU time in seconds				
Pattern selection	–	10	–	10
Training	–	4	–	4
Propagation	–	27	–	27
Total	33 690[a]	41	3369.5[b]	41

[a] Projected after 100 000 simulations.
[b] For 10 000 simulations.

the three test cases, respectively. From these tables it can be observed that, in the case of basic MCS simulation, the maximum difference of the predicted probability of failure with respect to the 'exact' one is 30%, while the corresponding difference in the case of MCS with IS is around 10%. In Table 8 a comparison between 'exact' and predicted values of the critical load factor is shown, for five randomly selected simulations. The results indicate that the maximum error is only 1.5%. This accuracy however is not reflected, for this example, to the corresponding values of the probability of failure, where the error of the prediction is much larger as described above. This is because p_f shows a high sensitivity with respect to the slightly modified, due to the NN approximation, sample space of resistances.

Table 7
Example 1—test case 3. 'Exact' and predicted values of p_f and the required CPU time ($i = 3$, $j = 7$, $k = 1$)

Number of simulations	'exact' Basic MCS p_f (%)	NN Basic MCS p_f (%)	'exact' MCS-IS p_f (%)	NN Basic-IS p_f (%)
50	8.00	4.00	9.00	6.90
100	8.00	5.00	7.95	6.45
300	8.00	5.67	8.05	7.25
500	8.00	5.60	8.82	7.83
1000	8.30	5.70	8.73	7.73
5000	8.68	5.96	8.66	7.65
10 000	8.68	5.98	8.68	7.66
50 000		6.07		7.65
100 000		6.03		7.64
CPU time in seconds				
Pattern selection	–	22	–	22
Training	–	5	–	5
Propagation	–	33	–	33
Total	36 070[a]	60	3607[b]	60

[a] Projected after 1000 000 simulations.
[b] For 10 000 simulations.

Table 8
Example 1—test case 3. Performance of NN in calculating the collapse loads ($i = 3$, $j = 7$, $k = 1$)

Simulation run	'Exact' values	NN estimates
1	848.60	845.20
228	934.21	929.69
1994	879.10	884.77
4165	850.14	856.61
5000	874.92	882.27

5.2. Example 2

The second test example is the six storey space frame shown in Fig. 8. The loads consist of a 19 kN/m^2 gravity load and a basic load of 110 kN applied to each node in the front elevation in the negative z direction. The three test cases examined in Example 1, for different combinations of random variables, are also considered here. The type of probability density functions, mean values and standard deviations for all variables are presented in Tables 9 and 10 for Basic MCS and MCS-IS, respectively.

The performance of the basic NN configuration with one hidden layer, using different number of hidden units, is shown in Fig. 9, while Table 11 shows the efficiency of various NN architectures as

Table 9
Example 2. Characteristics of random variables for Basic MCS

Random variables	PDf	μ	σ		
σ_y (kN/cm^2)	N	24.0	2.4		
Loads ($\times 10^3$)	Log-N	6.4	0.2		
Random variables	PDf	μ_x	μ_y	σ_x	σ_y
Z_1 (cm^3)	N	1276.5	476.9	63.8	23.8
Z_2 (cm^3)	N	609.6	30.5	133.9	6.7
Z_3 (cm^3)	N	1222.5	61.1	573.6	28.7
Z_4 (cm^3)	N	2163.1	108.1	989.8	49.5
Z_5 (cm^3)	N	3048.0	152.4	1399.5	70.0

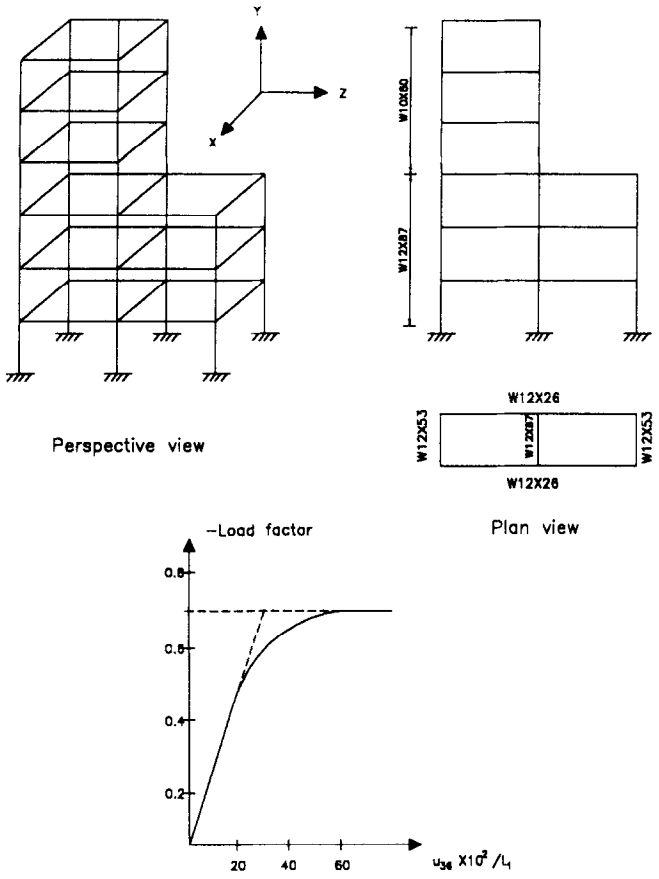


Fig. 8. Example 2. Six-story space frame and load–displacement curve.

Table 10
Example 2. Characteristics of random variables for MCS-IS

Random variables	PDF	μ	σ
σ_y (kN/cm^2)	N	24.0	2.4
Loads	N	0.723	0.160

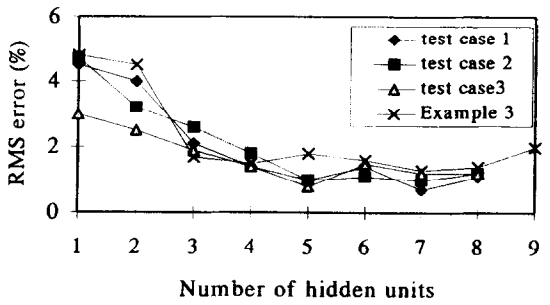


Fig. 9. Examples 2 and 3. Performance of NN configuration using different number of hidden units.

Table 11
Example 2. Performance of various NN architectures

Random variables	NN1				NN2				NN3			
	i–j–k	e_{RMS}	e_{MAX}	ε (%)	i–j–k	e_{RMS}	e_{MAX}	ε (%)	i–j–k	e_{RMS}	e_{MAX}	ε (%)
2	1–3–1	0.021	0.029	0.67	1–5–1	0.008	0.012	0.37	1–7–1	0.007	0.016	0.27
3	2–3–1	0.026	0.092	0.87	2–4–1	0.018	0.045	0.61	2–5–1	0.010	0.019	0.39
4	3–4–1	0.014	0.041	0.43	3–5–1	0.008	0.022	0.41	3–7–1	0.021	0.047	0.78

Table 12
Example 2—test case 1. 'Exact' and predicted values of p_f and the required CPU time ($i = 1$, $j = 7$, $k = 1$)

Number of simulations	'exact' Basic MCS p_f (%)	NN Basic MCS p_f (%)	'exact' MCS–IS p_f (%)	NN Basic–IS p_f (%)
50	2.00	2.00	3.64	3.64
100	4.00	5.00	4.70	4.70
300	5.00	5.33	5.04	5.00
500	5.20	5.40	5.40	5.38
1000	5.20	5.60	5.38	5.40
5000	5.42	5.82	5.38	5.38
10 000		5.76		5.42
50 000		5.84		5.41
100 000		5.85		5.41
CPU time in seconds				
Pattern selection selection	–	163	–	163
Training	–	4	–	4
Propagation	–	20	–	20
Total	816 322[a]	187	40 816[b]	187

[a] Projected after 100 000 simulations.
[b] For 5000 simulations.

Table 13
Example 2—test case 2. 'Exact' and predicted values of p_f and the required CPU time ($i = 2$, $j = 5$, $k = 1$)

Number of simulations	'exact' Basic MCS p_f (%)	NN Basic MCS p_f (%)	'exact' MCS–IS p_f (%)	NN Basic–IS p_f (%)
50	12.00	14.00	15.40	14.96
100	16.00	17.00	15.98	15.74
300	17.33	19.33	16.95	16.88
500	17.80	18.40	18.05	17.76
1000	17.60	17.50	18.15	18.08
5000	18.15	18.08	18.12	18.28
10 000		18.10		18.28
50 000		17.98		18.36
100 000		17.98		18.35
CPU time in seconds				
Pattern selection	–	261	–	261
Training	–	9	–	9
Propagation	–	26	–	26
Total	869 841[a]	296	43 492[b]	296

[a] Projected after 100,000 simulations.
[b] For 5,000 simulations.

Table 14
Example 2—test case 3. 'Exact' and predicted values of p_f and the required CPU time ($i = 3$, $j = 5$, $k = 1$)

Number of simulations	'exact' Basic MCS p_f (%)	NN Basic MCS p_f (%)	'exact' MCS-IS p_f (%)	NN Basic-IS p_f (%)
50	22.00	18.00	17.28	17.28
100	15.00	13.00	15.41	15.64
300	14.66	14.00	16.41	16.41
500	16.60	16.40	17.16	17.16
1000	17.20	17.40	16.86	17.00
5000	16.74	17.12	16.70	16.90
10 000		17.28		16.95
50 000		17.27		16.99
100 000		17.27		16.99
CPU time in seconds				
Pattern selection selection	–	554	–	554
Training	–	26	–	26
Propagation	–	33	–	33
Total time	923 360[a]	613	46 168[b]	613

[a] Projected after 100,000 simulations.
[b] For 5000 simulations.

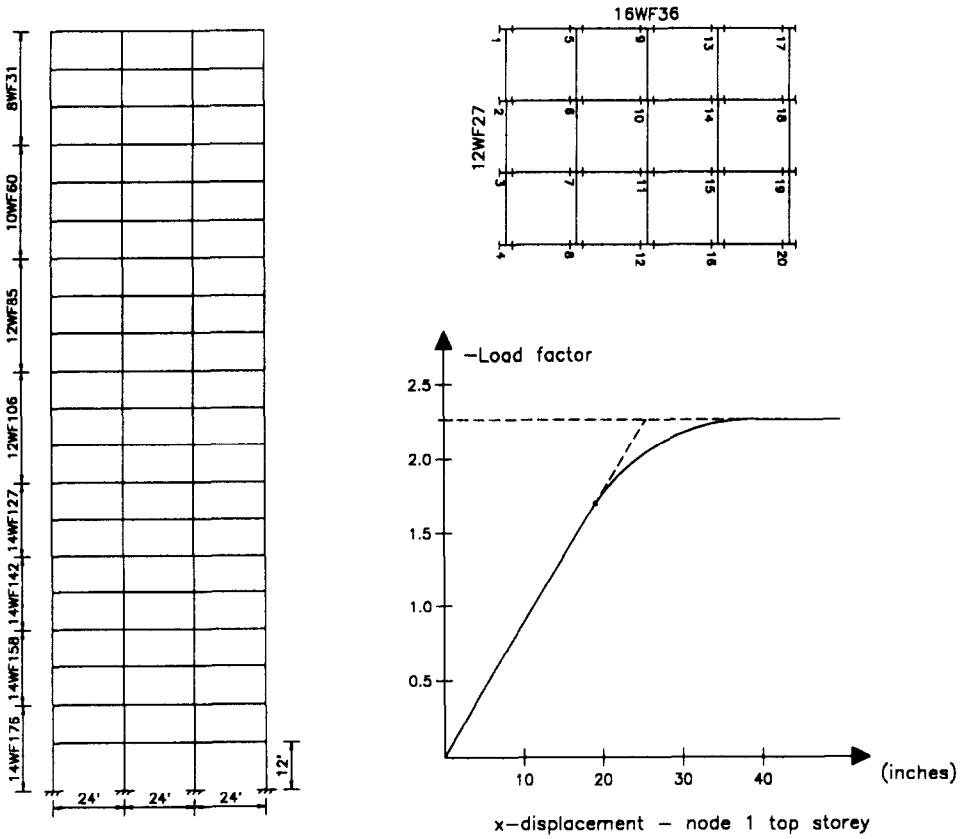


Fig. 10. Example 3. Twenty-story space frame and load–displacement.

measured by the difference between final predicted and 'exact' values of critical load factors. It can be seen that the selection of the best NN architecture can again be based on the minimum achieved value of e_{RMS}. The probabilities of failure as well as the CPU times required are shown in Tables 12–14. The difference between predicted and 'exact' values of the probability of failure for the three test cases considered, when Basic MCS is employed are 7%, 0.9%, 3%, while the corresponding differences when MCS-IS is used are 0.5%, 1.2%, 1.7%, respectively.

5.3. Example 3

The third test example is the twenty story space frame shown in Fig. 10. This example is selected in order to show, in a more realistic problem, the efficiency of the proposed approach both in terms of accuracy and computing effort in conjunction with advanced solution techniques. For this reason only test case 1 of previous examples is considered, while the type of probability density functions, mean values and standard deviations for all variables are presented in Table 15 for Basic MCS. The loads considered here are vertical forces equivalent to uniform load of 100 psf (4.788 kN/m^2) and a basic horizontal pressure of 20 psf (0.956 kN/m^2). The performance of different NN architectures with one hidden layer is shown in Fig. 9, while in Table 16 a comparison between 'exact' and predicted values of critical load factors is performed in five randomly selected simulation runs. The results indicate that a remarkable agreement is attained between 'exact' and predicted values of critical load factor. Very good agreement is also achieved, as shown in Table 17, between 'exact' and predicted values of the probability of failure. In fact the two sets of results are almost identical. It seems that the accuracy of the NN prediction depends also on the type and the scale of the structure as well as on the smoothness of the load displacement curve. Additionally, this table presents a comparison between different solution strategies for the 'exact' limit elastoplastic analysis. MCS I stands for the application of the direct Cholesky factorisation, while MCS II and MCS III correspond to the application of the PCG method. In MCS II the preconditioner is based on an incomplete factorisation of the stiffness matrix in which a rejection by magnitude factor ψ controls the number of terms retained in the preconditioning matrix. In the present tests ψ is taken equal to 10^{-2}. In the case of MCS III a complete factored matrix is used as preconditioner which is kept fixed for a number of steps of the incremental analysis before its reformulation. The number of steps at which the preconditioner is reformulated, is controlled by the number of CG iterations as described in [17]. A 60% and 30% reduction in CPU time is obtained, respectively, with the two versions of the preconditioning matrix as compared to the modified direct Cholesky solution.

Table 15
Example 3. Characteristics of random variables for Basic MCS

Random variables	PDF	μ	σ
σ_y (kN/cm^2)	N	24.0	2.4
Loads	Log-N	5.2	0.2

Table 16
Example 3. Performance of NN in calculating the collapse loads ($i = 1$, $j = 7$, $k = 1$)

Simulation run	'Exact values'	NN estimates
1	220.38	220.64
92	218.61	218.94
159	199.66	200.87
219	220.78	221.00
300	208.51	209.10

Table 17
Example 3. 'Exact' and predicted values of p_f and the required CPU time ($i = 1$, $j = 7$, $k = 1$)

Number of simulations	'exact' Basic MCS-I p_f (%)	'exact' Basic MCS-II p_f (%)	'exact' Basic MCS-III p_f (%)	NN Basic-MCS p_f (%)
50	7.99	7.99	7.99	7.99
100	7.99	7.99	7.99	7.99
300	8.33	8.33	8.33	8.33
500	8.82	8.82	8.82	8.80
1000	9.25	9.25	9.25	9.20
5000				9.08
10 000				9.04
50 000				9.13
100 000				9.12
CPU time in seconds				
Pattern selection	–	–	–	4010
Training	–	–	–	4
Propagation	–	–	–	20
Total	639 344[a]	390 000[a]	200 488[a]	4034

[a] Projected after 1000 simulations.

6. Conclusions

This paper presents an application of Neural Networks to the reliability analysis of complex structural systems in which failure of the system is due to plastic collapse. The approximate concepts that are inherent in reliability analysis and the time consuming requirements of repeated analyses involved in Monte Carlo Simulation motivated the use of Neural Networks.

The computational effort involved in the conventional Monte Carlo Simulation becomes excessive in large-scale problems because of the enormous sample size and the computing time required for each Monte Carlo run. The use of Neural Networks can practically eliminate any limitation on the scale of the problem and the sample size used for Monte Carlo Simulation provided that the predicted critical load factors, corresponding to different simulations, fall within acceptable tolerances.

A Back Propagation Neural Network algorithm is successfully used to produce approximate estimates of the critical load factors, regardless the size or the complexity of the problem, leading to very close predictions of the probability of failure. Moreover, for large and complex structural systems which resist a great percentage of the loading beyond their elastic state, the NN prediction appears to be more accurate. It was also deduced that, contrary to Neural Network applications in other fields of Computational Structural Mechanics, the present application showed a considerable robustness with regard to selection of training set and network architecture in predicting the of probability failure. Training samples, required to train the Neural Network, appear to be independent on the type of structure or the type of the required analysis.

The use of Monte Carlo Simulation with Importance Sampling leads to considerable improvement in Neural Network prediction of the probability of failure. This is due to the fact that using the Importance Sampling technique the sensitivity of p_f with regard to the modified sample space of critical load factors, displayed by Neural Network predictions, is reduced leading to more accurate estimates. The methodology presented could therefore be implemented for predicting accurately and at a fraction of computing time the probability of failure of large and complex structures.

Acknowledgments

The present work has been partially supported by the Research Programme EV5V-0278 and the HCM Network Programme CHRXCT 93-0390 of European Union. This support is gratefully acknowledged.

References

[1] M. Shinozuka, Basic analysis of structural safety, J. Struct. Engrg. ASCE 109 (1983) 721–739.
[2] H.O. Madsen, S. Krenk and N.C. Lind, Methods of Structural Safety (Prentice-Hall, Englewood Cliffs, NJ, 1986).
[3] P. Thoft-Chrinstensen, Reliability and optimisation of structural systems '88, Proceedings of the 2nd IFIP WG7.5, London, Springer-Verlag, 1988.
[4] G. Deodatis and M. Shinozuka, Weighted integral method II: Response variability and reliability, J. Engrg. Mech. ASCE 117 (1991) 1865–1877.
[5] P.D. Spanos and C.A. Brebbia, Computational stochastic mechanics, Computational Mechanics Publications, 1991.
[6] T.Y. Kam, R.B. Corotis and E.C. Rossow, Reliability of non-linear framed structures, J. Struct. Engrg. ASCE 109 (1983) 1585–1601.
[7] Y. Fujimoto, M. Iwata and Y. Zheng, Fitting-adaptive importance sampling reliability analysis, computational stochastic mechanics, in: P.D. Spanos and C.A. Brebbia, eds., Computational Mechanics Publications (1991) 15–26.
[8] J.E. Pulido, T.L. Jacobs and E.C. Prates De Lima, Structural reliability using Monte-Carlo simulation with variance reduction techniques on elastic-plastic structures, Comput. Struct. 43 (1992) 419–430.
[9] L. Berke and P. Hajela, Applications of artificial neural nets in structural mechanics, NASA 331-348 TM-102420, 1990.
[10] L. Berke, S.N. Patnaik and P.L.N. Murthy, Optimum design of aerospace structural components using neural networks. Comput. Struct. 48 (1993) 1001–1010.
[11] B. Fu and P. Hajela, Minimising distortion in truss structures—A Hopfield network solution, Comput. Syst. Engrg. 4 (1993) 69–74.
[12] R.J. Melosh, L. Berke and P.V. Marcal, Knowledge-based consultant for selecting a structural analysis strategy, NASA Conference Publication 2059, 1987.
[13] P.S. Theocaris and P.D. Panagiotopoulos, Neural networks for computing in fracture mechanics. Methods and prospects of applications, Comput. Methods Appl. Mech. Engrg. 106 (1993) 213–228.
[14] A.I. Khan, B.H.V. Topping and A. Bahreininejad, Parallel training of neural networks for finite element mesh generation, in: B.H.V. Topping and A.I. Khan, eds., Neural Networks and Combinatorial Optimisation in Civil and Structural Engineering (Civil-Comp Press, 1993) 81–94.
[15] D.E. Rummelhart and J.L. McClelland, Parallel Distributed Processing, Vol. 1: Foundations (The MIT Press, Cambridge, 1986).
[16] M. Papadrakakis and S.A. Karamanos, A simple and efficient solution method for the limit elastoplastic analysis of plane frames, J. Comput. Mech. 8 (1991) 235–248.
[17] M. Papadrakakis and V. Papadopoulos, A computationally efficient Method for the limit elastoplastic analysis of space frames, Comput. Mech. J. 16(2) (1995) 132–141.
[18] R.Y. Rubinstein, Simulation and the Monte Carlo Method (John Wiley & Sons, 1981).
[19] R.E. Melchers, Structural reliability: Analysis and Prediction (Ellis Horwood, John Wiley and Sons, Chishester, 1987).
[20] C.G. Bucher, Adaptive Sampling—An iterative fast Monte Carlo procedure, Structural Safety 5 (1988) 119–126.
[21] A. Karamchandani and C.A. Cornell, Adaptive hybrid conditionals expectation approaches for reliability estimation, Structural Safety 11 (1991) 59–74.
[22] G.I. Schueller and A.H-S. Ang, Advances in structural reliability, J. Nucl. Engrg. 39 (1991) 55–66.
[23] J.G. Orbinson, W. McGuire and J.F. Abel, Yield surface applications in non-linear steel frames analysis, Comput. Methods Appl. Mech. Engrg. 33 (1982) 557–573.
[24] Y. Ueda and T. Yao, The plastic node method: A new method of plastic analysis, Comput. Methods Appl. Mech. Engrg. 34 (1982) 1089–1104.
[25] N. Bitoulas and M. Papadrakakis, An optimised computer implementation of the incomplete Cholesky factorisation, Comput. Syst. Engrg. 5(3) (1994) 265–274.
[26] Y.H. Pao, Adaptive Pattern Recognition and Neural Networks (Addison-Wesley Publishing Co., 1989).
[27] Y. Hirose, K. Yamashita and S. Hijiya, Back-propagation algorithm which varies the number of hidden units, Neural Networks 4 (1991) 61–66.
[28] P.T. Baffes, Nets user's guide, version 2.01 NASA, Lyndon B. Johnson Space Centre, 1989.