Eng. & Technology, Vol.25, No.3, 2007

Artificial Neural Network Model for Predicting Nonlinear Response of Uniformly Loaded Fixed Plates

Dr. Ayad Amjad Abdul-Razzak*
Salim T. Yousif**
Received on: 5/5/2005
Accepted on: 6/6/2006

Abstract
An artificial neural network (ANN) model has been developed for the prediction of nonlinear response for plates with built-in edges and different sizes, thickness and uniform loads. The model is based on a six-layer neural network with back propagation learning algorithm. The learning data were performed using a nonlinear finite element program, the set of 1500x16 represent the deflection response of load. Incremental stages of the nonlinear finite element analysis was generated by using 25 schemes of built-in rectangular plates with different thickness and uniform distributed loads.
The neural network model has four input nodes representing the uniform distributed load, thickness, length of plate and length to width ratio, four hidden layers and sixteen output nodes representing the deflection response.
Regression analysis between finite element results and values predicted by the neural network model shows the least error. This approach helps in the reduction of the effort and time required determining the load-deflection response of plate as the FE methods usually deal with only a single problem for each run while ANN methods can solve simultaneously for a patch of problems.

Keywords: Artificial neural network, Elasto-plastic plate, Finite element, Nonlinear plate.

Notations
Ae: Element area
B : Strain-displacement matrix
B_b : Bending strain-displacement matrix
B_s : Transverse shear strain-displacement matrix
D : Elasticity matrix
D_b : Flexural (bending) rigidities
D_s : Shear rigidities
E : Young's modulus
J : Jacobian matrix
K : Stiffness matrix
K_b : Bending stiffness matrix
K_s : Transverse shear stiffness matrix
$[R]$: Transformation matrix
M_x, M_y, M_{xy} : Generalized stress components (moment)
N, R_i, S_I : Shape functions

https://doi.org/10.30684/etj.25.3.5
2412-0758/University of Technology-Iraq, Baghdad, Iraq

Q_x,Q_y : Generalized stress components (shear forces)
w, d_i : Displacements
NI : Number of neurons in the input layer.
NH : Number of neurons in the hidden layer.
NO : Number of neurons in the output layer.
x : Input vector.
hH : Input for the hidden layer.
hO : Input for the output layer.
yH : Output of the hidden layer.
y : Output of the network.
wji : Matrix $NH \times NI$ of synaptic weights connecting the input and hidden layers.
wkj : Matrix $NO \times NH$ of synaptic weights connecting the hidden and output layers.
b : Bias, or threshold vector.
i = [1 : *NI*] : A neuron in the input layer.
j = [1 : *NH*] : A neuron in the hidden layer.
k = [1 : *NO*] : A neuron in the output lay
γ_{xz}, γ_{yz} : Transverse shear strain components in the Cartesian coordinate system.
$\gamma_{\xi\zeta}$, $\gamma_{\eta\xi}$: Transverse shear strain components in the natural (local) coordinate system.
$\gamma'_{\xi\zeta}$,$\gamma'_{\eta\xi}$: Assumed transverse shear strain components in the natural (local) coordinate system.
ε_b : Bending strain tensor
ε_s : Transverse shear strain tensor
θ_{xi}, θ_{yi} : Rotations
ν : Poisson's ratio
$f_c^{'}$: Concrete cylinder compressive strength
$\{s\}$:Stress vector at sampling point.
$f(.)$: The nonlinear function performed by the neuron.

Introduction

Accurate modeling of the elasto-plastic plate has been attempted with a variety of numerical methods, such as the finite-element (FE), and the finite-difference approach. While accurate, these techniques are generally limited to a single analysis for a specific structure, and require long computation times when a number of simulations are run with different mesh properties. For this reason, the present work will explore the use of artificial neural network (ANN) modeling of the elasto-plastic plate in conjugation with the finite element techniques. The model is constructed through the use of the neural network design (NND) toolbox in MATLAB [1] from the MathWorks (Natick, MA).

Finite element formulation

Basic theory

The variation in the displacement and rotation fields over a Mindlin plate element are given by the following expression [2]

$$\left[W, q_x, q_y\right]^T = \sum_{i=1}^{n} N_i \, d_i \qquad (1)$$

The plate curvature-displacement and shear strain-displacement relations are then written as:

$$\varepsilon_b = \sum_{i=1}^{n} B_{bi} d_i$$

$$e_s = \sum_{i=1}^{n} B_{si} d_i \qquad (2)$$

The moment-curvature and shear force-shear strain relationships are given as:

$$[M_x, M_y, M_{xy}]^T = D_b \varepsilon_b \, , \; [Q_x, Q_y]^T = D_s \varepsilon_s \qquad (3)$$

The stiffness matrix (Kij) contributions from the bending and shear relations can be written as:

$$K^e_{bij} = \int_{Ae} B^T_{bi} D_b B_{bj} dA \quad ,$$

$$K^e_{sij} = \int_{Ae} B^T_{si} D_s B_{sj} dA \qquad (4)$$

Assumed transverse shear strain fields

Huang [2] used an artificial method for the elimination of shear locking by interpolating new shear strain fields from the strain values at the sampling points which are appropriately located in individual elements.

In the natural coordinate system, the transverse shear strain ($\gamma_{\xi\zeta}$) and ($\gamma_{\eta\xi}$) should tend towards zero for thin plate situations [2].

The assumed shear strain fields are chosen as:

$$\gamma'_{\xi\zeta} = \sum_{i=1}^{n} R_i(\xi,\eta)\, \gamma^i{}_{\xi\zeta} \quad ,$$

$$g'_{hz} = \sum_{i=1}^{n} S_i(x,h)\, g^i{}_{hz} \qquad (5)$$

The strain values at the sampling points are chosen as Dirac-delta functions of the following form:

$$\lambda^{13} = \sum_{i=1}^{n} \lambda_i^{13}\, \delta(\xi'_i - \xi)\, \delta(\eta'_i - \eta),$$

$$\lambda^{23} = \sum_{i=1}^{n} \lambda_i^{23}\, \delta(\xi'_i - \xi)\, \delta(\eta'_i - \eta) \qquad (6)$$

The shear strain can be expressed as:

$$\begin{bmatrix} g'_{xz} \\ g'_{hz} \end{bmatrix} = L \begin{bmatrix} g'_{xz} \\ g'_{yz} \end{bmatrix} \qquad (7)$$

in which

$$L = \frac{\partial z}{\partial z} \begin{bmatrix} \partial x/\partial x & \partial y/\partial x \\ \partial x/\partial h & \partial y/\partial h \end{bmatrix} = \frac{h}{2} J \qquad (8)$$

The assumed shear strain field for the 9- node Lagrangian element has been taken as a linear function of (ξ) direction for ($\gamma_{\xi\zeta}$) and a linear function of (η) direction for ($\gamma_{\eta\xi}$) see Fig. (1).

It is assumed that

$$\int_{-1}^{0} g_{xz}\, d_x = 0.0 \quad , \int_{0}^{1} g_{xz}\, d_x = 0.0 \, ,$$

$$\int_{-1}^{0} g_{hz}\, d_h = 0.0 \quad , \int_{0}^{1} g_{hz}\, d_h = 0.0 \qquad (9)$$

from which it can be concluded that

$$\xi = \pm a$$

where
$a = (1/3)^{1/2}$ (10)

which are the two Gauss points in the (ξ) direction.

Similarly, it can be concluded that

$\eta = \pm a$ where $a = (1/3)^{1/2}$ (11)

which are the two Gauss points in the (η) direction.

($\gamma_{\xi\zeta}$) is linear in (ξ) direction and quadratic in (η) direction and ($\gamma_{\eta\xi}$) is linear in (η) direction and quadratic in (ξ) direction, see Fig.(2), then

$$g'_{xz} = \sum_{i=1}^{3}\sum_{j=1}^{2} P_i(h)\,Q_j(x)\,g^{ij}_{xz} \quad ,$$

$$g'_{hx} = \sum_{i=1}^{3}\sum_{j=1}^{2} p_i(x)\,Q_j(h)\,g^{ij}_{hz} \qquad (12)$$

where

$$P_1(z)=\frac{z}{2b}\left(\frac{z}{b}+1\right),\ P_2(z)=1-\left(\frac{z}{b}\right)^2,$$

$$P_3(z)=\frac{z}{2b}\left(\frac{z}{b}-1\right)$$

$$Q_1(z)=\frac{1}{2}\left(1+\frac{z}{a}\right), Q_2(z)=\frac{1}{2}\left(1-\frac{z}{a}\right) \qquad (13)$$

The terms ($\gamma^{ij}_{\xi\zeta}$) and ($\gamma^{ij}_{\eta\xi}$) are the transverse shear strain evaluated from the displacement field at $a = (3)^{-1/2}$ and b =1

Neural network modeling
Overview of neural network approach

Neural networks are an information processing techniques based on the way biological nervous systems, such as the brain, process information. The fundamental concept of neural networks is the structure of the information processing system. Composed of a large number of highly interconnected processing elements or neurons, a neural network system uses the human-like technique of learning by example to solve problems. The neural network is configured for a specific application, such as data classification or pattern recognition, through a learning process called training. Just as in biological systems, learning involves adjustments to the synaptic connections that exist between the neurons.

Neural networks can differ on the way their neurons are connected; the specific kinds of computations their neurons do; the way they transmit patterns of activity throughout the network; and the way they learn including their learning fate. Neural networks are being applied to an increasing large number of real world problems. Their primary advantage is that they can solve problems that are too complex for conventional technologies; problems that do not have an algorithmic solution or for which an algorithmic solution is too complex to be defined.

The multi-layer perceptron is the most widely used type of neural network. It is both simple and based on solid mathematical grounds. Input quantities are processed through successive layers of ‘‘neurons’’. There is always an input layer, with a number of neurons equal to the number of variables of the problem, and an output layer, where the perceptron response is made available, with a number of neurons equal to the desired number of quantities computed from the inputs.

The layers in between are called ‘‘hidden’’ layers. With no hidden layer, the perceptron can only perform linear tasks. All problems, which can be solved by a perceptron can be solved with only one hidden layer, but it is sometimes more efficient to use two or more hidden layers. Each neuron of a layer other than the input layer computes first a linear combination of the outputs of the neurons of the previous layer, plus a bias. The coefficients of the linear combinations plus the biases are called the weights. Neurons in the hidden layer then compute a non-linear function of their input. Generally, the non-linear function is the sigmoid function. A sigmoid function is an S-shaped ‘‘squashing function’’, which maps a real value, which may be arbitrarily large in magnitude positive or negative to a real value, which lies within some narrow range. The result of this sigmoid function lies in the range 0–1. In the neural computation literature, the sigmoid is sometimes also referred to as the logistic function. According to the requirement of this function; the original data need to be scaled in to the range between 0 and 1. The criteria of convergence in training is based on the minimizing the root mean squared (RMS) error to a level, where a satisfactory agreement is found with the training set results of the network result. Once the networks are considered to be trained, testing data are presented to it and outputs are compared with the experimental or observed results.

In this study, a multi-layer feed-forward neural network is used. In a multi-layer feed-forward neural network, the artificial neurons are arranged in layers, and all the neurons in each layer have connections to all the neurons in the next layer. Associated with each connection between these artificial neurons, a weight value is defined to represent the connection weight. Fig.(3) shows architecture of a multi-layer feed-forward neural network with an input layer, an output layer, and foure hidden layers. The operation of the network consists of a forward pass through the network. A number of learning rules are available. The backpropagation learning algorithm is used in this study. Signals are received at the input layer, pass through the hidden layer, and reach the output layer. The learning process primarily involves the determining of connection weights and patterns of connections.

Applications of neural networks

Structural engineers started to use ANN in various applications. Chuang et al. [3] used back-propagation network to model the nonlinear relationship between the various input parameters associated with reinforced concrete columns and the actual ultimate capacity of the column.

Mathew et al. [4] proposed hybrid system to predict the failure pressure of masonry panels of various boundary conditions subjected to biaxial bending. The system combines both case-based reasoning technique and ANN.

Sanad and Saka [5] used ANN in predicting the ultimate shear strength of simply supported reinforced-concrete deep beams.

Pala et al [6] applied ANN approach for the soil-structure dynamic interaction analysis of a gravity dam.

Pathak and Arora [7] predicted responses of composite laminated plates under dynamic loading using neural network approach

Oreta [8] developed ANN model using past experimental data on shear failure of slender reinforced concrete beams without reinforcements.

Mathematical background

The Back Propagation (BP) neural network is a multi-layered, feed-forward NN. The BP neural network approximates the non-linear relationship between the input and the output by adjusting the weight values internally instead of giving the function expression explicitly. Further, the BP neural network can be generalized for the input that is not included in the training patterns

The back propagation algorithm is used to train the BPNN. This algorithm looks for minimum of error function in weight space using the method of gradient descent. The combination of weights that minimizes the error function is considered to be a solution to the learning problem. The algorithm [9] can be described in the following steps:

Input feedforward:

Once the input vector is presented to the input layer it can be calculated the input to the hidden layer as:

$$h_j^H = b_j + \sum_{i=1}^{NI} w_{ji} x_i \qquad (14)$$

Each neuron of the hidden layer takes its input h_j^H and uses it as the argument for a function and produces an output given by:

$$y_j^H = f(h_j^H) \qquad (15)$$

Now the input to the neurons of the output layer calculated as:

$$h_k^o = b_k + \sum_{j=1}^{NH} w_{kj} y_j^H \qquad (16)$$

and the network output is then given by:

$$y_k = f(h_k^o) \qquad (17)$$

where f represents the activation function

The error back-propagation learning algorithm:

An error vector can be defined as being the difference between the network output and the target output value:

$$e_k = t_k - y_k \qquad (18)$$

Based on the error vector the mean square error vector can be calculated as:

$$E = \frac{1}{NO} \sum_{k=1}^{NO} e_k^2 \qquad (19)$$

This is the cost function to be minimized during the learning process. The sum-squared error E is a function of all the variables of the network .

Using the chain rule it is possible to calculate the gradient of the error with respect to the weight matrix connecting the hidden layer to the output layer as follows:

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial e_k} \frac{\partial e_k}{\partial y_k} \frac{\partial y_k}{\partial h_k^O} \frac{\partial h_k^O}{\partial w_{kj}} \qquad (20)$$

Computing each term of this expression yields:

$$\frac{\partial E}{\partial e_k} = e_k \qquad \frac{\partial e_k}{\partial y_k} = -1$$

$$\frac{\partial y_k}{\partial h_k^O} = f_k^{'}\left(h_k^O\right) \qquad \frac{\partial h_k^O}{\partial w_{kj}} = y_j^H \qquad (21)$$

Combining the expressions above results in:

$$\frac{\partial E}{\partial w_{kj}} = -e_k f_k'(h_k^O) y_j^H \qquad (22)$$

The correction Δw_{kj} applied to the weight matrix connecting the hidden layer to the output layer is:

$$\Delta w_{kj} = -a \frac{\partial E}{\partial w_{kj}} \qquad (23)$$

where a is a constant known as the step-size or the learning rate.

To update the weights connecting the input layer to the hidden layer, the procedure above is to be repeated:

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial e_k}\frac{\partial e_k}{\partial y_k}\frac{\partial y_k}{\partial h_k^O}\frac{\partial h_k^O}{\partial y_j^H}\frac{\partial y_j^H}{\partial h_j^H}\frac{\partial h_j^H}{\partial w_{ji}} \qquad (24)$$

After calculating each of the terms above, the correction to the weight matrix is written as:

$$\Delta w_{ij} = -a\partial_j x_i \qquad (25)$$

where

$$\partial_j = f_j'(h_j^H)\sum_{k=1}^{NO}\partial_k w_{kj} \qquad (26)$$

Neural network design and training.

To train a back propagation type neural network with the results of the finite element analyses [2], network architecture was required. Four input variables representing the uniform distributed load, thickness, length of plate and length to width ratio constituted the network-input layer. The sixteen output variables represent the deflection of load incremental stages of finite element analysis.

The set of 1500x16 represents the deflection response of load. Incremental stages of finite element [2] analysis was generated by using 25 schemes of built-in rectangular plates (E_x=E_y=30000N/mm^2 and v=0.3) with different thickness and uniform distributed loads.

An ANN training input data file was formed comprised of 1500 rows and 4 columns, while the target file is formed of 1500 rows and sixteen columns. A network with four hidden layers was exclusively chosen for ANN models trained in this study.

To train ANN models, first the entire training data file was randomly divided into training and testing data sets. About 90 % of the data, 1350 patterns, were used to train the different network architectures where remaining 150 patterns were used for testing to verify the prediction ability of each trained ANN model. Since ANNs learn relations and approximate functional mapping limited by the extent of the training data, the best use of the trained ANN models can be achieved in interpolation.

Preprocessing of data by scaling was carried out to improve the training of the neural network. To avoid the slow rate of learning near the end points specifically of the output range due to the property of the sigmoid function which is asymptotic

to values 0 and 1, the input and output data were scaled between the interval 0.1 and 0.9. The linear scaling equation [8]:

$$y = (0.8/\Delta)x + (0.9 - 0.8x_{max}/\Delta) \quad (27)$$

A variable limited to minimum (x_{min}) and maximum (x_{max}) values given in Table 1 with $\Delta = x_{max} - x_{min}$ was used in this study.

Table (1). Range of input data.

	Load kN/m^2	Thickness (mm)	Length (m)	Aspect ratio
Maximum	28000	300	6.0	2.5
Minimum	70	100	2.0	1.0

The back-propagation learning algorithm (scaled conjugate gradient) was employed for learning in the MATLAB program [10]. Each training "epoch" of the network consisted of one pass over the entire 1350 training data sets. The 150 testing data sets were used to monitor the training progress for a total of 30,000 learning cycles (epochs). Six network architectures with four hidden layers were trained for predicting the deflection responses with four input nodes and sixteen output nodes. Overall, the MSEs (mean square errors) decreased as the networks grew in size with increasing number of neurons in the hidden layers. The error levels for both training and testing sets matched closely when the number of hidden nodes approached 40 in the 4-20-40-40-40-16 architecture (4 inputs, 20 nodes in first hidden layer, 40 nodes in other hidden layers and 16 output nodes, respectively)(see Fig. (3).)

Results and discussions:

The performance of the neural network and regression analysis is discussed in detail below. The mean square error of the network with number of epochs is shown in Fig.(4) and it is evident that for 30000 epochs, the network attains an accuracy of (0.0000121).

Figs.(5 and 6) show some of 150 test results of central point load-deflection relationships for different plate sizes, thickness and uniform distributed loads, the comparison between the nonlinear finite element and the neural network analysis are presented in Fig.(5), whereas Fig.(6) compares the linear part of the results with linear elastic analysis[11]. The average absolute errors for the 150 test results are between 0.05% and 2.13%. The figures show good agreement of ANN results with both finite element results and analytic result in its linear part.

The performance of a trained network can be measured to some extent by the errors on the training, validation and test sets, but it is often useful to investigate the network response in more detail. One option is to perform a regression analysis between the network response and the corresponding targets and finding a correlation coefficient. It is a measure of how well the variation in output is explained by the targets. If this

number is equal to 1, then there is a perfect correlation between targets and outputs.

Fig.(7) shows a plot of finite element maximum deflections against corresponding ANN prediction. A linear correlation can be observed and the correlation coefficient was found to be 1.0.

ANN methods can often obtain results in almost negligible time as compared to similar works using the FE methods. Moreover, FE methods usually deal with only a single problem for each run, while ANN methods can solve simultaneously for a patch of problems.

Conclusion:

The system described in this work assists the neural network prediction model for nonlinear response of plates with built-in edges and with different sizes, thicknesses and uniform distributed loads. Such model can aid with the implementation of concurrent engineering practice. It is concluded that the proposed network model is capable of predicting the load-deflection relationships of plates with least error, this approach helps in the reduction of the effort and time required in determining the nonlinear response of plates.

References

1. The math works,MATLAB v6.5, 24 prime way, Natick, MA 01760-1500,USA, 2002.

2. Hou-Cheng Huang, " Static and Dynamic Analyses of Plates and Shells", Springer-Verlag.UK., 1989.

3. Chuang, P. H., Anthony, T. C., and Wu, X., "Modeling the capacity of pin-ended slender reinforced concrete columns using neural networks", Journal of Structural Engineering Vol. 124, No. 7, July 1998, pp. 830-838.

4. Mathew, A., Kumar, B., and Pedrschi, R. F., "Analysis of masonry panel under biaxial bending using ANNs and CBR", Journal of Computing in Civil Engineering" Vol. 13 No. 3, July 1999, pp. 170-177.

5. Sanad, A., and Saksa, M. P., "Prediction of ultimate shear strength of reinforced-concrete deep beams using neural networks", Journal of Structural Engineering Vol. 127, No. 7. July 2001, PP. 818-828

6. Pala, M., Caglar, N., and Elmas, M., 'Soil structure dynamic interaction using artificial neural network" ,Turkish symposium on artificial intelligence and neural network. Vol. 1, No. 1, July 2003, PP 348-354.

7. Pathak, K. K., and Arora, A., "Dynamic response of composite laminated plates using artificial neural networks", IE(I) Journal-AS Vol. 85, May 2004.

8. Oreta, A. W. C., "Simulating size effect on shear strength of RC beams without stirrups using neural networks", Engineering Structures Vol. 26, 2004, pp. 681-691

9. Mauro J. Atalla, "Model updating using neural network", Ph.D. Thesis, Virginia Polytechnic Institute and State University,1996.

10. Howard, D., and Mark, B., "Neural network toolbox for use with MATLAB, User's Guide, Version 4", the Math works, Inc 2002.

11. Timoshenko S. S. and Woinowsky-Krieger, S.,"Theory of plates and shells", 2nd ed., McGraw. Hill, New Yourk, Ltd, 1959.

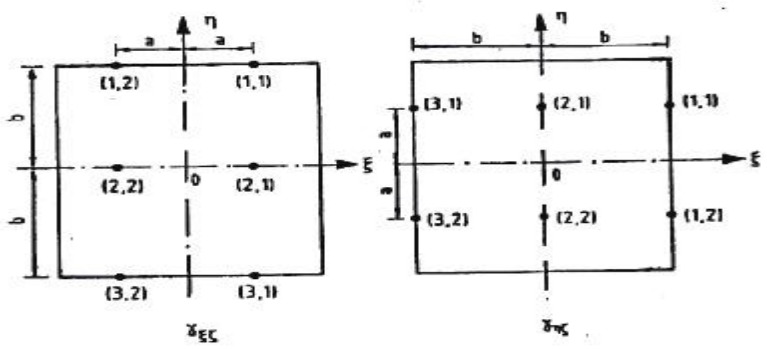


Fig. (1) Interpolation points (i, j) for the 9- Lagrangian element

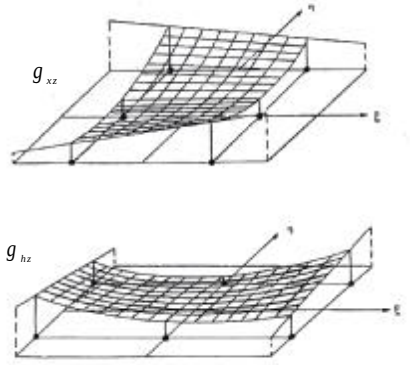
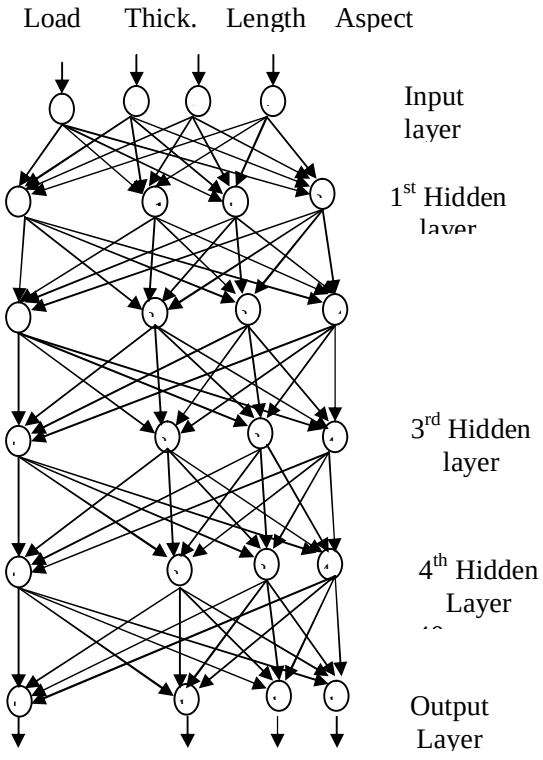


Fig. (2) Assumed shear strain fields for 9- node Lagrangian element.



(Deflection Response)

Fig. (3) Architecture of network

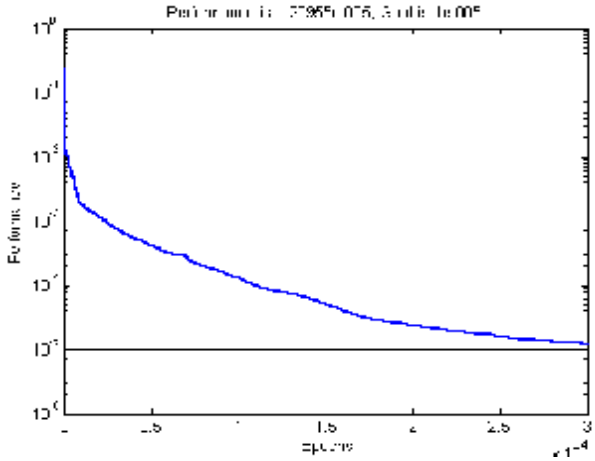


Fig. (4) Mean square error of the network (performance) with number of epochs

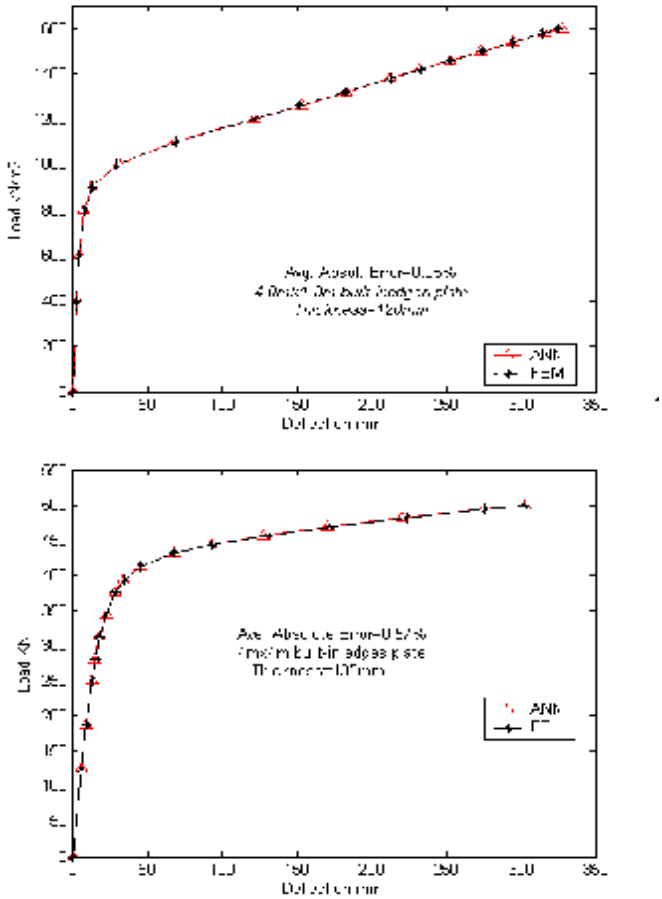


Fig. (5) Central point load-deflection relationships for nonlinear finite

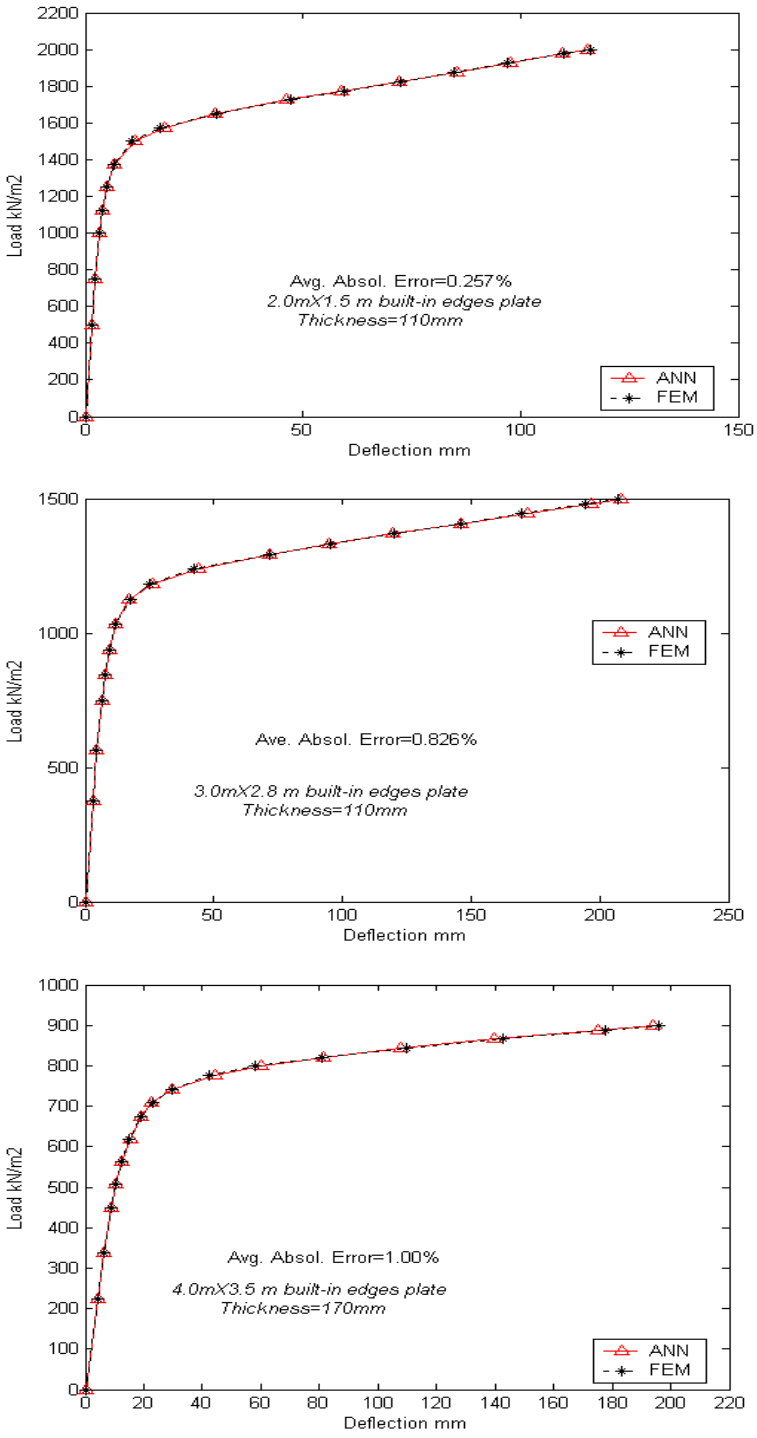


Fig. (5) continued

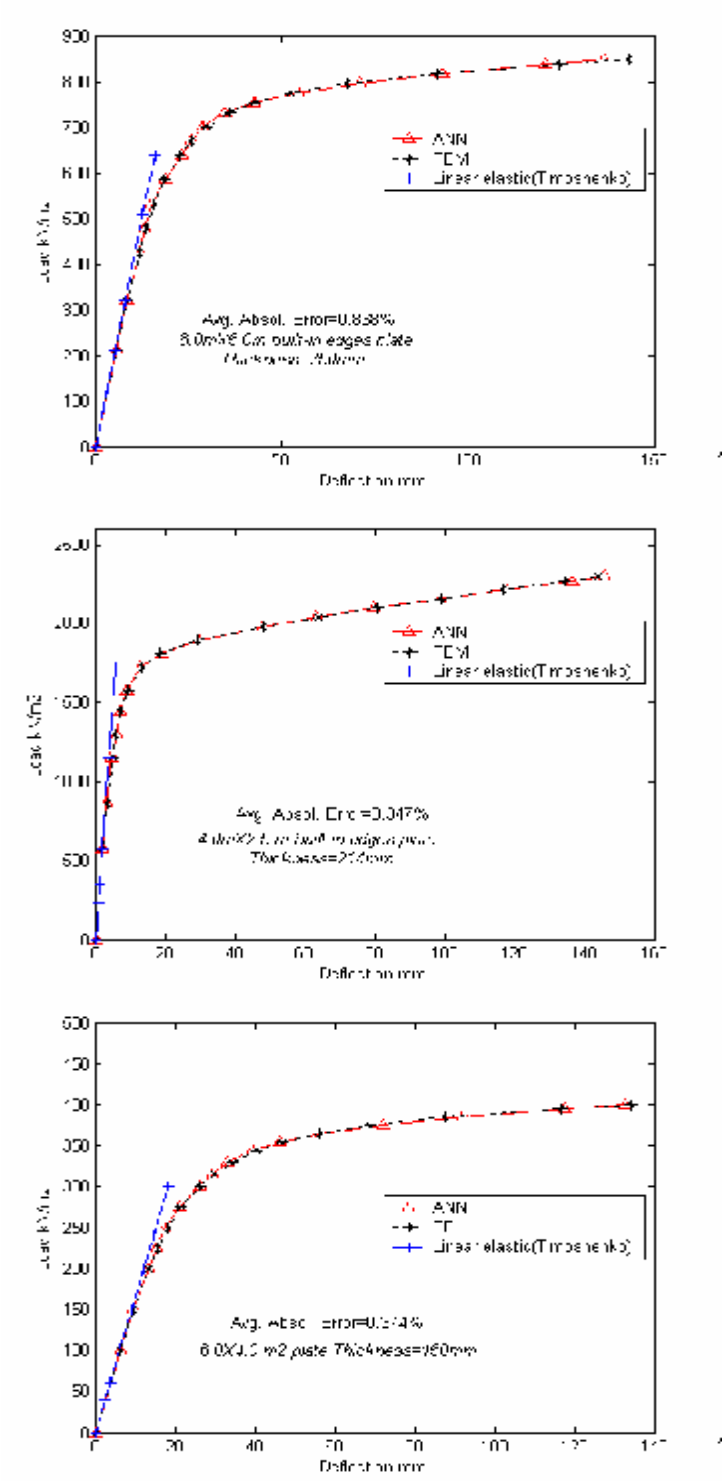


Fig. (6) Central point load-deflection relationships for nonlinear finite element, neural network and linear elastic analysis.

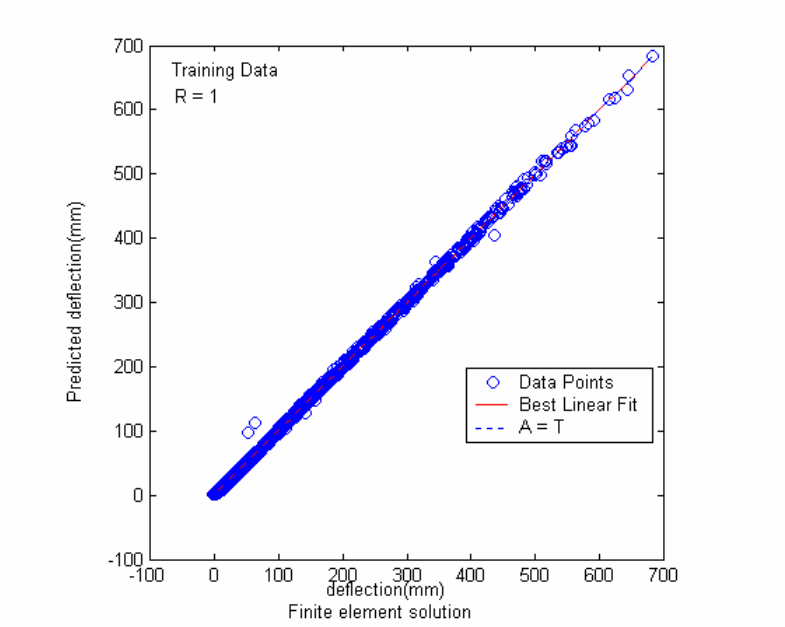


Fig. (7) Finite element maximum deflections and corresponding NN prediction