PERGAMON

Computers and Structures 79 (2001) 2261–2276

Computers & Structures

www.elsevier.com/locate/compstruc

Neural networks in mechanics of structures and materials – new results and prospects of applications

Zenon Waszczyszyn [a,*], Leonard Ziemiański [b]

[a] *Institute of Computer Methods in Civil Engineering, Cracow University of Technology, Krakow, Poland*
[b] *Chair of Structural Mechanics, Rzeszów University of Technology, Rzeszów, Poland*

Abstract

Basic ideas of back-propagation neural networks (BPNNs) are presented in short. Then BPNN applications in analysis of the following problems are discussed: (1) bending analysis of elastoplastic beams, (2) elastoplastic plane stress problem, (3) estimation of fundamental vibration periods of real buildings, (4) detection of damage in a steel beam, (5) identification of loads applied to an elastoplastic beam. Regularization neural network is briefly discussed and its application to estimation of concrete fatigue durability it shown. A modified Hopfield network is used to the analysis of an elastic angular plate with unilateral constraints. In the end some conclusions and prospects of neurocomputing applications are pointed out. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Neural network; Applications; Mechanics; Structures; Materials

1. Introduction

In recent years the interest in transferring methods developed in one discipline to the analysis of problems in other disciplines has evidently increased. Such a co-disciplinary approach can give solutions which are difficult to achieve by methods specialized to the analysis of inward-disciplinary oriented problems. Development of computer hardware and software causes beyond of doubt formulation of new, nonstandard methods of data processing. From among these methods *soft computing* has to be mentioned as an approach related to the so-called *intelligent methods* and especially to 'biological' methods of information processing. Artificial neural networks (ANNs), evolutionary strategies (ES) and Fuzzy inference systems (FIS) make a background for 'intelligent computing'. ANNs and *neurocomputing* related to them have especially rich possibilities of applications.

ANNs simulate biological neural systems very loosely. They are massively parallel, they can process not only crisp but also noisy or incomplete data. ANNs can be used for the mapping of input to output data without known 'a priori' a relationship between those data. ANNs can be applied in optimum design, classification and prediction problems. ANNs advantages offer, in fact, complementary possibilities to standard, von Neuman type computers. In such a way hybrid systems with neural procedures open the door to new computational strategies.

On the base of above remarks it is clear why ANNs have been recently widely introduced to the analysis of increasing number of problems in science and technology. Of course, ANNs have also been applied in mechanics of structures and materials. It seems that paper [1] by Adeli and Yeh was the first to be published in 1989 in an archival journal. During the last ten years ANNs applications in mechanics grew so rapidly that now it would be difficult to point out those problems of mechanics in which ANNs have not been used yet. Only a rough reflection of actual situation in this field can be

[*] Corresponding author.
E-mail addresses: zenwa@twins.pk.edu.pl (Z. Waszczyszyn), ziele@prz.rzeszow.pl (L. Ziemiański).

found in general lectures at international conferences, e.g. Refs. [2,3] or recent books, e.g. Ref. [4].

In the paper some new results are discussed in short, related to research done at the Civil Engineering Departments of Cracow and Rzeszów Universities of Technology. Only selected problems are reported, corresponding to mechanics of structures and materials. From among the problems listed in Paez's classification [5] the main attention is paid to simulation and identification problems, defined as below:

A. *Simulation*: Known excitation variables and characteristics of mechanical system (MS), search response.
A′. *Inverse simulation*: Known response variables and characteristics of MS, search an excitation.
B. *Identification*: Known excitation and response, search characteristics of MS.

There is an extensive literature on ANNs and corresponding information can be easily found elsewhere, cf. references in Ref. [4]. In the paper the main attention is paid to the back-propagation neural networks (BPNNs) which are mostly used in the analysis of mechanics problems. That is why the basic fundamentals of BPNN are given in Section 2. Other types of NNs (regularization and Hopfield networks) are briefly discussed in Section 3.

Section 2 is devoted to discussion of problems analyzed by BPNNs by the authors' research groups. Each example is related to the formulation of problems and their neural analysis. The attention is turned to special selected problems, associated with BPNN applications, which are: (i) data selection and preprocessing, (ii) design of BPNNs, (iii) accuracy of neurocomputing. The problems are related to both data prepared by computational systems and those taken from experimental evidence.

Section 3 deals with the analysis of some problems by other ANNs. The achieved results enable us to point out promising prospects of neurocomputing applications in mechanics of structures and materials.

2. Applications of BPNNs

2.1. Back-propagation neural network

The main goal of BPNNs is mapping of input, i.e. vector $\boldsymbol{x} \in R^N$ into output, i.e. vector $\boldsymbol{y} \in R^M$. This can be written in short

$$\boldsymbol{x}_{N\times 1} \overset{\text{BPNN}}{\longrightarrow} \boldsymbol{y}_{M\times 1}, \tag{1a}$$

and in general

$$\boldsymbol{x}^{(p)} \to \boldsymbol{y}^{(p)}, \quad \text{for } p = 1,\dots,P, \tag{1b}$$

where p – number of pattern. The mapping is performed by a network composed of processing units (neurons) and connections between them. In Fig. 1a a single neuron i is shown. Input signals x_j are cumulated in the neuron summing block Σ and activated by function F to have only one output y_i:

$$y_i = F(v_i), \quad v_i = \sum_{j=1}^{N} w_{ij}x_j + b_i, \tag{2}$$

where v_i – activation potential, w_{ij} – weights of connections, b_i – threshold parameter.

From among various activation functions the sigmoid functions are commonly used:

(a) binary sigmoid (logistic function)

$$F(v) = \frac{1}{1+\exp(-\sigma v)} \in (0,1) \quad \text{for } \sigma > 0, \tag{3a}$$

(b) bipolar sigmoid

$$F(v) = \frac{1-\exp(-\sigma v)}{1+\exp(-\sigma v)} \in (-1,1) \quad \text{for } \sigma > 0. \tag{3b}$$

In Fig. 1b a standard, multilayer network is shown. The network is composed of the input, hidden and output layers, respectively. Each neuron is connected with all neurons of the previous and subsequent layers but their are no connections inside the layer. An example of the network, shown in Fig. 1b is of 4-3-3-2 structure, i.e. they are $N = 4$ inputs; H1 = 3, H2 = 3 are numbers of

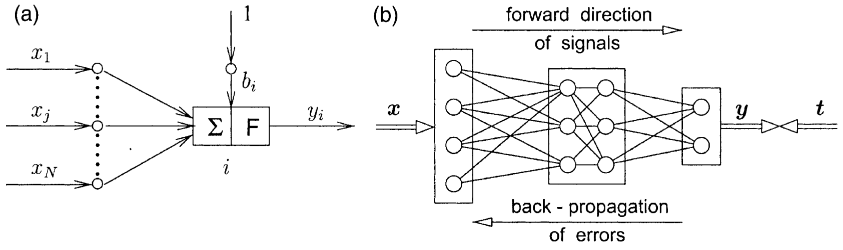

Fig. 1. (a) Single neuron i, (b) three-layer, forward, error back-propagation network.

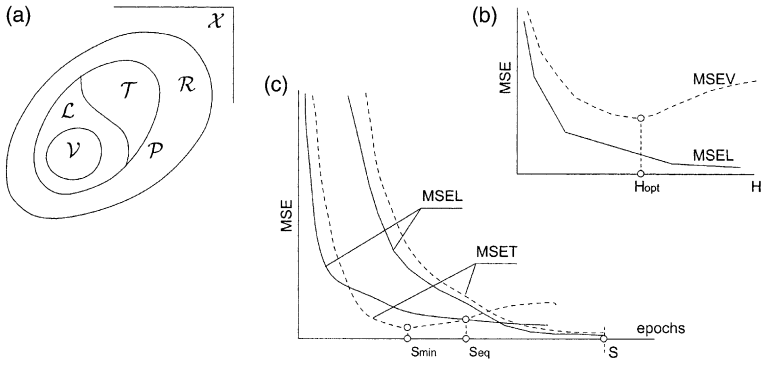

Fig. 2. (a) Different sets of patterns in the behavior space x, (b) cross-validation approach for estimating an optimal number of neurons H_{opt} in a hidden layer, (c) stopping criteria related to learning–testing relationships.

neurons in two hidden layers and the output layers has $M = 2$ outputs. The weights w_{ij}^l and biases b_i^l (where l is the number of layer) are called the *network parameters.*

The values of the network parameters are computed iteratively in the process of network training (learning). After training the network should be tested. That is why a set of patterns $\mathscr{P}$, composed of the pairs of known input and output vectors, is formulated (or selected from space $\mathscr{R}$ in which certain rules obey, cf. Fig. 2a):

$$\mathscr{P} = \{(\boldsymbol{x}, \boldsymbol{t})^{(p)} | p = 1, \ldots, P\} \subset \mathscr{R}, \tag{4}$$

where $\boldsymbol{x}^{(p)}$, $\boldsymbol{t}^{(p)}$ – input and target vectors for the pth pattern, P – number of patterns.

From the set $\mathscr{P}$ the *training and testing sets*, $\mathscr{L}$ and $\mathscr{T}$ respectively, are selected:

$$\begin{aligned} L &= \{(\boldsymbol{x}, \boldsymbol{t})^{(p)} | p = 1, \ldots, L\}, \\ T &= \{(\boldsymbol{x}, \boldsymbol{t})^{(p)} | p = 1, \ldots, T\}. \end{aligned} \tag{5}$$

The signals $x_i^{(p)}$ are transmitted in the forward direction, i.e. from the input to output – cf. Fig. 1b. After computation the output vector $\boldsymbol{y}^{(p)}$ can be compared with target vector $\boldsymbol{t}^{(p)}$ and *network error* is computed

$$E = \frac{1}{2} \sum_{p=1}^{P} \sum_{i=1}^{M} \left(t_i^{(p)} - y_i^{(p)} \right)^2. \tag{6}$$

Different measures of the network error can be defined, e.g. formula (6) corresponds to the mean-square-error. Other measure, frequently used is the mean-square-error

$$\text{MSE} = \frac{2E}{P}. \tag{7}$$

The network parameters are computed (the network is "tuned") using the formula

$$w_{ij}(s+1) = w_{ij}(s) + r_{ij}(s), \tag{8}$$

where s – the number of iteration step, r_{ij} – reinforcement of the network parameters (the bias can be formally treated as a weight $w_{i0} = b_i$ for signal $x_0 = 1$, cf. Fig. 1a).

The reinforcement is in fact the weight increment and it can be computed by a *learning rule.* The simplest rule corresponds to the gradient formula

$$r_{ij}(s) = \Delta w_{ij}(s) = -\eta \frac{\partial E}{\partial w_{ij}} \bigg|_s, \tag{9}$$

where η – learning rate. The correction of network parameters is performed in the backward direction, cf. Fig. 1b. The network trained in the manner described above is called *back-propagation neural network.*

One transmission of signals corresponding to all patterns and back-propagation of errors is called an *epoch.* In what follows the iteration step s corresponds to an epoch number. The iteration process is used according to different *stopping criteria*: (i) training and testing errors are nearly equal to each other, i.e. $\text{MSET}(s_{\text{eq}}) \approx \text{MSET}(s_{\text{eq}})$, (ii) the number of epoch S is prescribed, cf. Fig. 2c.

The gradient formula (9) is sensitive to the value of learning rate η and usually the corresponding training process is slowly convergent (a great number of epochs is needed). That is why various learning methods have been developed. Computer simulators usually offer a variety of learning methods. The authors of this paper applied successfully the learning method Rprop (Resilient-propagation) which was accessible in the SNNS simulator [6]. The Rprop is a locally adaptive method in which learning rates η_{ij} are associated with network parameters and the reinforcement $r_{ij}(s)$ takes into account the changes of signs of error gradients sgn $(g_{ij}(s) \cdot g_{ij}(s-1))$, where $g_{ij} = \partial E / \partial w_{ij}$.

Before training the BP network should be designed. One of the heuristic methods is related to the selection of *validation set* $\mathscr{V}$, cf. Fig. 2a, which is used to estimate BPNN architecture. In Fig. 2b an optimal number of neurons H_{opt} in a hidden layer is shown, corresponding to minimal error on the validation set MSEV.

The trained network is checked by means of testing patterns (set $\mathscr{T}$ in Fig. 2a) and the network has good generalization properties if it gives small errors of computations in the rule space $\mathscr{R}$, cf. Fig. 2a.

From among many problems related to BPNNs the most important are:

1. selection of learning and testing patterns, their principal components, then their preprocessing;
2. design of network architecture (number of hidden layers and neurons in them);
3. training of networks;
4. estimation of network generalization.

Those and other questions are discussed at length in literature but the answers are strongly problem-dependent and, besides the knowledge of the neural "kitchen", a deep understanding of analyzed models should be always appreciated.

2.2. Bending analysis of elastoplastic beams – neural simulation of physical relationship

Let us consider the simplest case of elastoplastic beam bending corresponding to linear geometric and equillibrium equations:

$$v' = \varphi(x), \quad \varphi' = -\kappa(x), \quad T' = -p(x), \quad M' = T(x), \tag{10}$$

where x – independent variable measured along the beam axis, v, φ – transverse displacement and rotation of the beam cross-section respectively, T, M – shear force and bending moment, respectively, $p(x)$ – load field, $(\cdot)' = \mathrm{d}(\cdot)/\mathrm{d}x$.

Eq. (10) have to be completed by physical relationship:

$$\kappa = \begin{cases} M/(EI) & \text{for elastic deformation,} \\ f(M) & \text{for elastoplastic deformation.} \end{cases} \tag{11}$$

In a general case the elastoplastic relation $\kappa = f(M)$ cannot be formulated explicitly. This relationship depends not only on material model $\sigma(\varepsilon)$ but also on shape of beam cross-section.

In case of rectangular cross-section $b \times h$ and linear strain-hardening the relationship $M(\kappa)$ can be formulated in the following dimensionless form:

$$m = \begin{cases} 2/(3k) & \text{for } |k| \leqslant 1, \\ 2/(3k) + (1-\chi)(1 - 1/(3k^2))\operatorname{sgn} k & \text{for } |k| \geqslant 1, \end{cases} \tag{12}$$

where

$$m = 4M/(bh\sigma_0^2), \quad k = \kappa h/(2\varepsilon_0), \quad \chi = E_\mathrm{p}/E. \tag{13}$$

The following material parameters are used in Eq. (12): σ_0, ε_0 – stress and strain at the yield point, E, E_p – elastic and plastic tangent stiffnesses, respectively.

Finite difference method (FDM) equations, corresponding to the finite difference scheme with a midpoint were used to the analysis of elastoplastic boundary-value problem (11) and (12), cf. [7].

Second term of formula (12) was employed to compute the patterns which were used for the training of BPNN, as shown schematically in Fig. 3.

In Ref. [8] the BP neural networks were used as a neural procedure. After a number of numerical experiments the BPNN structure 1-10-1 was adopted with the binary sigmoid activation function (3a). The network was trained for fixed values of the strain-hardening parameter $\chi = 0, 0.1$. In order to cover the range $k \in [1, 20]$ four BPNNs were used for each χ and the number of training patterns was $L = 288$. The Rprop learning method was applied in the frame of SNNS computer simulator [6]. The training was limited to $S = 2000$ epochs. The trained networks were tested on about $T = 500$ patterns and it was proved that the relative error for predicting k was below 1%.

The trained BPNNs were used in FDM equations. In this way a hybrid FDM/BPNN program was implemented. On the bench-mark tests it was proved that the accuracy of this program was very high.

As an example, a cantilever beam is considered. The beam data: $b \times h = 0.012 \times 0.025$ m^2, $le = 1.0$ m, $E = 2.1 \times 10^5$ MPa and $\sigma_0 = 210$ MPa are subjected to reverse loads. The results, shown in Fig. 4, correspond to the equilibrium paths $P(v_1; \chi)$ for $\chi = 0.1$, 1 and for the following load programs: $0 \rightarrow P_1 \rightarrow P_2 \rightarrow P_1$, where $P_1 = 420$ kN, $P_2 = -420$, -430, -560 kN. The hybrid simulation by program FDM/BPNN gives practically the same results as purely numerical program FDM.

2.3. Analysis of elastoplastic plane stress problem – neural simulation of return mapping algorithm

Analysis of elastoplastic constitutive equations in the finite element method (FEM) computer program is difficult from the numerical point of view. The analysis needs computation of the actual stress vector and consistent stiffness matrix at each Gauss point of plane finite elements. This can be made by the return mapping al-

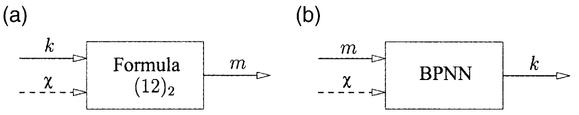

Fig. 3. (a) Application of second term of formula (12), to compute patterns for BPNN shown in (b).

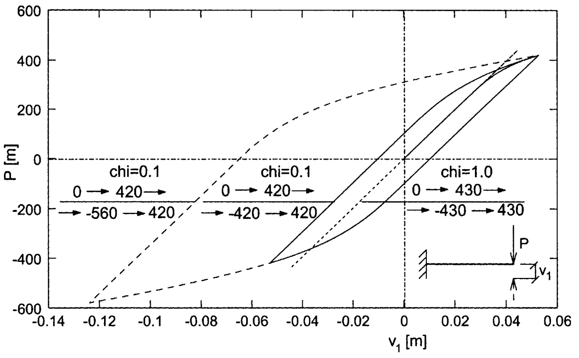

Fig. 4. Equilibrium paths $P(v_1;\chi)$ for reverse loads $0 \to P_1 \to P_2 \to P_1$ for elastoplastic beam ($\chi = 0.1$) and elastic beam ($\chi = 1$).

gorithm (RMA). It enables us to perform the mapping shown in Fig. 5:

$$\{\boldsymbol{\sigma}^A, \Delta\boldsymbol{\varepsilon}\} \to \{\boldsymbol{\sigma}^D, \boldsymbol{E}_D^{ep}\} \tag{14}$$

where $\boldsymbol{\sigma}^A$, $\boldsymbol{\sigma}^D \in R^3$ – stored and actual stress vectors, respectively, $\Delta\varepsilon \in R^3$ – increment of strain vector, $\boldsymbol{E}_D^{ep} \in R^3 \times R^3$ – consistent stiffness matrix.

In Ref. [9] BPNN was used as a simulator for RMA. Patterns for BPNN training and testing were computed on the base of numerical procedure sketched in Fig. 5b, applying the following assumptions: (i) the material is homogeneous and isotropic with von Mises yield surface and linear, isotropic strain-hardening, (ii) constitutive equations of classical flow theory are valid.

Patterns were formulated with respect to the following dimensionless variables:

$$\bar{\boldsymbol{\sigma}} = \boldsymbol{\sigma}/\sigma_e^B, \quad \Delta\bar{\boldsymbol{\varepsilon}} = E\,\Delta\boldsymbol{\varepsilon}/\sigma_e^B, \quad \Delta\bar{\lambda} = E\Delta\lambda, \quad \chi = H/E, \tag{15}$$

where σ_e^B – effective stress at the yield surface $F^B = 0$ in Fig. 5a, E, H – elasticity and strain-hardening moduli, respectively, $\Delta\lambda$ – increment of yielding parameter. After computation of $\boldsymbol{\sigma}^D$ and $\Delta\lambda_D$ the stiffness matrix $\boldsymbol{E}_D^{ep} = E \cdot \overline{\boldsymbol{E}}_D^{ep}$ can be computed by analytical formulae, cf. Ref. [9].

Assuming the strain-hardening parameter to be $\chi = 0.032$ about 1×10^6 patterns were computed. From them plastically active patterns were randomly selected, i.e. $L = 3349$ and $T = 9044$ for training and testing, respectively. BPNN of structure 6-40-20-4 with bipolar sigmoid activation function (3b) was trained. The Rprop learning method and SNNS computer simulator were used. After $S = 34000$ epochs the training and testing errors were MSE $< 10^{-4}$.

The efficiency of neural simulator was examined on 10000 patterns selected randomly. It was stated that BPNN needed about 40–50% of computational time consumed by numerical procedure to perform mapping (14).

BPNN was incorporated as a neural procedure into ANKA FE code [7]. In this was the hybrid program ANKA-H was implemented.

ANKA and ANKA-H programs were tested on a bench-mark problem (a perforated plate [9]) and then the programs were used to the analysis of a notched plate [9]. All the corresponding data are shown in Fig. 6a. The 8-node isoparametric finite elements with four Gauss points of reduced integration were used. The FE mesh is shown in Fig. 6b.

The results of computations are shown in Fig. 6c. The neural procedure was trained at $\chi = 0.032$ but its generalization properties are quite satisfactory for other values of the strain-hardening parameter $\chi \in [0, 0.07]$.

In order to compare the processor time used by ANKA and ANKA-H programs the computation was carried out for the strain-hardening parameter $\chi = 0.005$ and 65 steps $\Delta v_A = -0.02$ mm. In case of ANKA the processing time was 74 s at 145 global iterations. The processing time used by ANKA-H program was 38 s at 83 iterations. Then the computation was repeated for $\chi = 0.07$. The processing time was 78 s at 146 iterations for ANKA and the corresponding figures were 64 s, at 117 iterations if ANKA-H was explored.

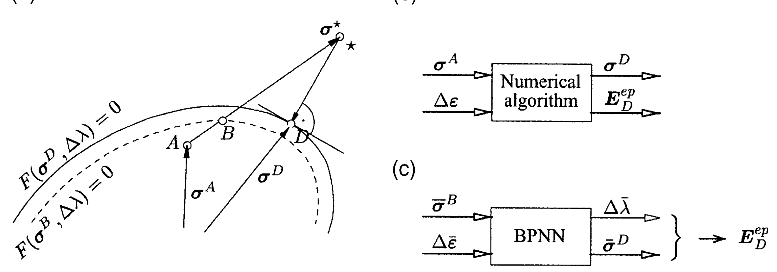

Fig. 5. (a) RMA, (b) scheme of numerical computation of patterns, (c) scheme of BP neural network.

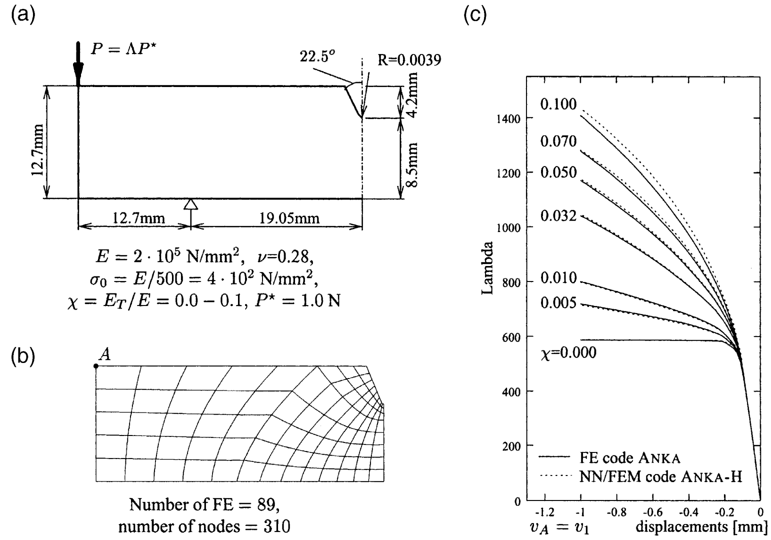

Fig. 6. (a) Geometry, load and material characteristics, (b) FE mesh of half a plate, (c) load paths $\Lambda(v_A;\chi)$ computed by ANKA and ANKA-H programs.

2.4. Estimation of fundamental vibration periods of prefabricated buildings – selection of principal components

The fundamental period of building vibrations is used in simple expert systems as a factor for the assessment of the building state. That is why simple empirical formulae were proposed in literature, cf. e.g. Ref. [10], but the estimations given by them are far from satisfactory. In order to improve the estimation ANNs can be used.

Simple BPNN neural networks were applied to estimate the fundamental natural periods of vibrations of typical Polish prefabricated, residential buildings. A group of 5-storey buildings consists of 13 objects, erected in various technology, cf. Table 1. In Fig. 7 the plan and vertical section of a building is shown.

Long term measurements of horizontal components of vibrations excited by firing of explosives in a nearly quarry were performed. FFT and spectral analysis were used to process the records corresponding to vibrations in x and y directions (transverse and longitudinal vibrations, respectively). In such a way the "measured" natural periods of vibrations, put together in Table 1, were deduced.

It was stated in Ref. [11] that the soil–structure interaction plays an important role in vibrations of medium height buildings. The interaction is roughly expressed by the coefficient of elastic uniform vertical deflection of soil basement C_z. This suggested the following empirical formula, for both transverse and longitudinal direction:

$$T_1 = 0.98/\sqrt[3]{C_z} \tag{16}$$

In column 8 of Table 1 the estimation by formula (16) is given.

In Ref. [10] it was also stated that the dimension of building plan b along the vibration direction, cf. Fig. 7, could be taken as a representative component. It seemed to be reasonable to consider also building stiffnesses as next candidates for the input vector components. The equivalent bending and shear stiffnesses s and r were adopted

$$s = E\sum_i I_i/a, \quad r = G\sum_i A_i/a, \tag{17}$$

where E, G – Young's and Kirchhoff's moduli, I_i, A_i – moment of inertia and area of the ith wall in the building plan, a – length of the building segment, Fig. 7.

Three input vectors $\boldsymbol{x}_{N\times 1}$ were considered and a single output y:

$$\begin{aligned}&(1)\ \boldsymbol{x}_{2\times 1} = \{\bar{C}_z, \bar{b}\},\\ &(2)\ \boldsymbol{x}_{3\times 1} = \{\bar{C}_z, \bar{b}, \bar{s}\},\\ &(3)\ \boldsymbol{x}_{4\times 1} = \{\bar{C}_z, \bar{b}, \bar{s}, \bar{r}\}, \quad y = T_1,\end{aligned} \tag{18}$$

where all the components x_j are scaled to the range [0.1, 0.9].

$P = 31$ patterns, put together in Table 1, were split into the training, validation and testing patterns, $L = 22$, $V = 4$, $T = 5$ patterns, respectively. BPNNs with a single layer, of structure 2-3-1, 3-4-1 and 4-4-1 and binary sigmoid activation functions were formulated after some numerical experiments. After $S = 1000$–2000 epochs

Table 1
Five-storey buildings

Building	Direction	Pattern number/set	T_1(s) measured	$e_{\mathrm{rel}} = \lvert 1 - T_{\mathrm{com}}/T_e \rvert \times 100\%$			
				Network 2-3-1	Network 3-4-1	Network 4-4-1	Empirical formula
1	2	3	4	5	6	7	8
DOMINO-68 (I)	Transverse	$1/l$	0.256	4.8	4.7	6.4	2.6
	Longitud.	$2/l$	0.230	5.5	1.0	0.6	11.6
DOMINO-86 (II)	Transverse	$3/l$	0.256	4.8	4.7	6.4	2.6
	Longitud.	$4/l$	0.230	5.5	1.0	0.6	11.6
WUF-T-67-S.A. /V	Transverse	$5/t$	0.253	5.9	5.5	8.1	5.0
	Longitud.	$6/l$	0.204	6.0	2.9	1.6	22.4
WUF-GT 84 (I)	Transverse	$7/l$	0.175	3.4	3.3	2.9	3.5
Seg. I	Longitud.	$8/v$	0.185	8.7	8.3	5.5	5.4
Seg. II	Transverse	$9/l$	0.180	0.5	0.7	1.4	9.3
	Longitud.	$10/t$	0.169	29.9	10.1	8.6	0.0
WUF-GT 84 (II)	Transverse	$11/l$	0.157	4.8	6.1	7.7	3.1
Seg. I	Longitud.	–	–	–	–	–	–
Seg. II	Transverse	$12/l$	0.180	8.6	7.2	4.4	10.7
	Longitud.	$13/l$	0.177	22.6	0.3	1.3	8.8
C/MBY/V (I)	Transverse	$14/v$	0.172	9.8	7.8	6.5	13.8
	Longitud.	$15/l$	0.192	10.8	9.8	1.5	26.9
C/MBY/V(II)	Transverse	$16/l$	0.185	6.8	5.8	2.9	1.9
	Longitud.	$17/t$	0.213	2.0	1.2	2.2	17.0
C/MBY/V (III)	Transverse	$18/l$	0.227	0.5	0.8	2.5	11.4
	Longitud.	$19/t$	0.233	0.3	1.1	1.0	13.9
BSK (I)	Transverse	$20/l$	0.155	7.8	5.0	1.3	1.5
Seg. I	Longitud.	$21/v$	0.233	8.7	0.2	1.3	53.5
Seg. II	Transverse	$22/l$	0.155	7.8	6.3	0.7	1.5
	Longitud.	$23/l$	0.233	9.5	2.0	1.1	53.5
BSK (II)	Transverse	–	–	–	–	–	–
Seg. I	Longitud.	–	–	–	–	–	–
Seg. II	Transverse	$24/l$	0.156	7.1	5.6	0.1	3.1
	Longitud.	$25/l$	0.233	9.5	2.0	1.1	53.5
WWP	Transverse	$26/l$	0.270	2.0	2.2	1.3	5.4
	Longitud.	$27/t$	0.294	17.6	12.1	12.0	14.7
WBL	Transverse	$28/l$	0.294	9.9	10.3	9.6	14.7
	Longitud.	$29/l$	0.263	0.5	1.4	1.0	2.6
WK-70	Transverse	$30/v$	0.256	3.4	3.2	3.9	0.0
	Longitud.	$31/l$	0.277	6.6	4.1	1.0	11.4

/set–l: training set; v: validation set; t: testing set.

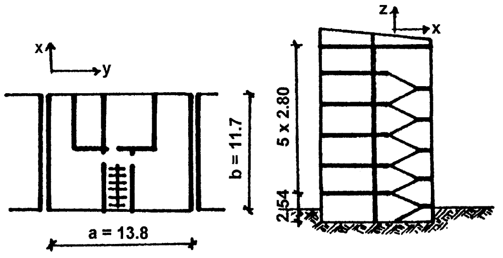

Fig. 7. Five-storey building WBL.

Rprop learning method gave errors (MSEL, MSEV) $<3.0 \times 10^{-4}$, cf. Ref. [12].

In Table 1 the relative error is listed

$$e_{\mathrm{rel}}^{(p)} = |1 - T_{\mathrm{comp}}(p)/T_{\mathrm{exp}}(p)| \times 100\%. \qquad (19)$$

In case of BPNN: 2-3-1 maximal error is $\max e_{\mathrm{rel}}^{(p)} \approx 30\%$ but the majority of patterns have errors smaller than 10%, cf. Fig. 8a. The accuracy is significantly improved in case of BPNN: 3-4-1 since $\max e_{\mathrm{rel}} \approx 12\%$. Adding the stiffness input $\bar{r}$ in the third term of Eq. (16) improves the errors $e_{\mathrm{rel}}^{(p)}$, cf. column 7 in Table 1 and Fig. 8b.

In Fig. 8 there are also shown estimations of fundamental vibration periods computed by the empirical formula (16).

2.5. Detection of damage in a steel beam – natural frequencies as input data

In Ref. [13] steel beams with an artificial crack (notch) were analyzed. BPNNs were used to identify the location and depth of notch. In Ref. [14] more precise tests and analysis of this problem were carried out.

In Fig. 9a the considered cantilever beam is shown. A set of beams with eight notch locations $a = 8, 16, \dots, 64$ cm and five depths $g = 2, 4, \dots, 10$ mm were tested. In this way $P = 40$ specimens were prepared for training and testing. Structure vibrations were measured simultaneously with the exciting force. Inertance frequency response function $H(f)$ was determined for each specimen. In Fig. 9c functions $H(f, g; a)$ are shown for the location of notch $a = 64$ cm, excitement applied to point 10 at the distance $b = 40$ cm and measurement acceleration at point 18, cf. Fig. 9b.

Subsequent natural frequencies f_n can be deduced from graphics shown in Fig. 9c. Increments of the first four frequencies Δf_n were computed:

$$\Delta f_n = 1 - f_n/f_n^0 \quad \text{for } n = 1, \dots, 4, \qquad (20)$$

where f_n^0, f_n – frequencies for lack of notch ($g = 0$) and at fixed notch depth g, respectively. After scaling the frequencies $\Delta\bar{f}_n \in [0.1, 0.9]$ they were used as input values and the outputs correspond to dimensionless location a/l and depth g/h:

$$x_{4\times1} = \{\Delta\bar{f}_n | n = 1, \dots, 4\}, \qquad y_{2\times1} = \{a/l, g/l\}. \qquad (21)$$

The neural identification problem was analyzed by the BPNN of structure 4-8-2 using binary sigmoid in each neuron. The Rprop learning method and SNNS computer simulator were applied. From among $P = 40$ patterns $T = 5$ randomly selected patterns were used for network testing. The same patterns were taken for cross-validation procedure for determining the number $H = 8$ neurons in the hidden layer, cf. Fig. 9d.

In Fig. 10a results of prediction of notch location is shown for the case of patterns taken from the FE analysis by ADINA computer code [15] (23 × 4 plane FEs were used). Results shown in Fig. 10b are quite close to those obtained on the basis of tests on laboratory specimens.

2.6. Identification of loads applied to an elastoplastic beam – one-level and cascade BPNNs

A simply supported beam, made of elastoplastic materials is subjected to uniform load defined by three

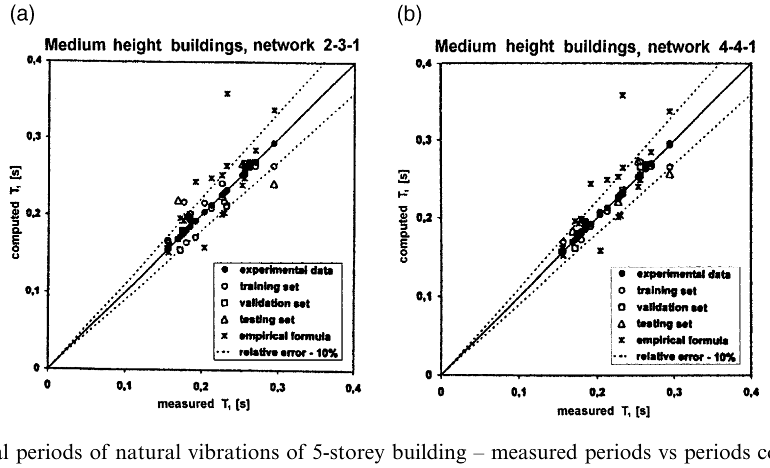

Fig. 8. Fundamental periods of natural vibrations of 5-storey building – measured periods vs periods computed by BPNNs.

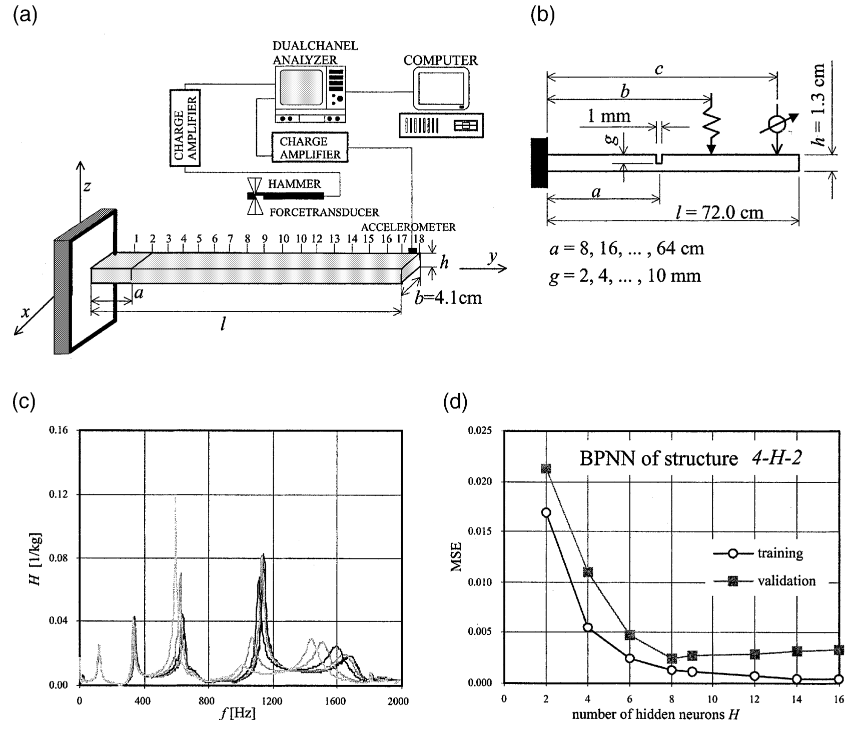

Fig. 9. (a) Tested beam and measuring system, (b) cantilever beam, (c) inertancy frequence response functions, (d) cross-validation for finding number of neurons H_{opt} in the hidden layer of BPNN.

parameters: (1) resultant Q, (2) load center location l_Q, (3) load width application w_q, cf. Fig. 11.

The identification problem corresponds to computation of the load parameters on the base of known first eigenfrequencies of the beam. The patterns for BPNNs were prepared by the FEM method using the ADINA computer code [15]. The plane stress was assumed and the mesh 6×30 of 8-node FEs with reduced integration was used.

The above formulated problem was analyzed in Ref. [16] assuming fixed width of load application w_q. The results discussed below were taken from Ref. [17], where more general assumptions were adopted and a new formulation of 'cascade' BPNN was proposed.

The training and testing sets were composed of $L = 440$ and $T = 90$ patterns, respectively. As in the previous applications the Rprop learning method and SNNS computer simulator were used. One hidden layer composed of H neurons and the binary sigmoid activation function in all neurons were assumed. The intensive cross-validation process led to determination of six first eigenfrequencies in the input vector of form of first term in Eq. (21) and $H = 10$ neurons in the hidden layer.

The neural analysis was performed by means of two types of BPNNs, shown in Fig. 12. *One-level* BPNN has the input vector $\boldsymbol{x}_f \in R^6$, corresponding to six increments of eigenvalue (20).

The *cascade* BPNN is composed of three BPNN l, where $l = \mathrm{I}, \mathrm{II}, \mathrm{III}$. The output of network BPNN I is used as input in BPNN II and then the network BPNN III utilizes outputs from the two previous BPNN I and BPNN II as components of the input vector $\boldsymbol{x}_{8\times 1} = \{\boldsymbol{x}_f, \overline{Q}, \bar{l}_Q\}$.

In Fig. 13 there are shown computed versus target results by two types of BPNNs. It is visible that identification of the load resultant values Q and the load center location l_Q are quite satisfactory by both networks. Identification of the load width application w_q is not so accurate but it can be slightly improved by the cascade BPNN (in Fig. 13 the coefficients r_L and r_T are written, corresponding to linear correlation of computed and target patterns taken from the training and testing sets, respectively).

3. Applications of other neural networks

3.1. Regularization neural network

Regularization neural networks (RNNs) are related to the regularization theory which is associated with the

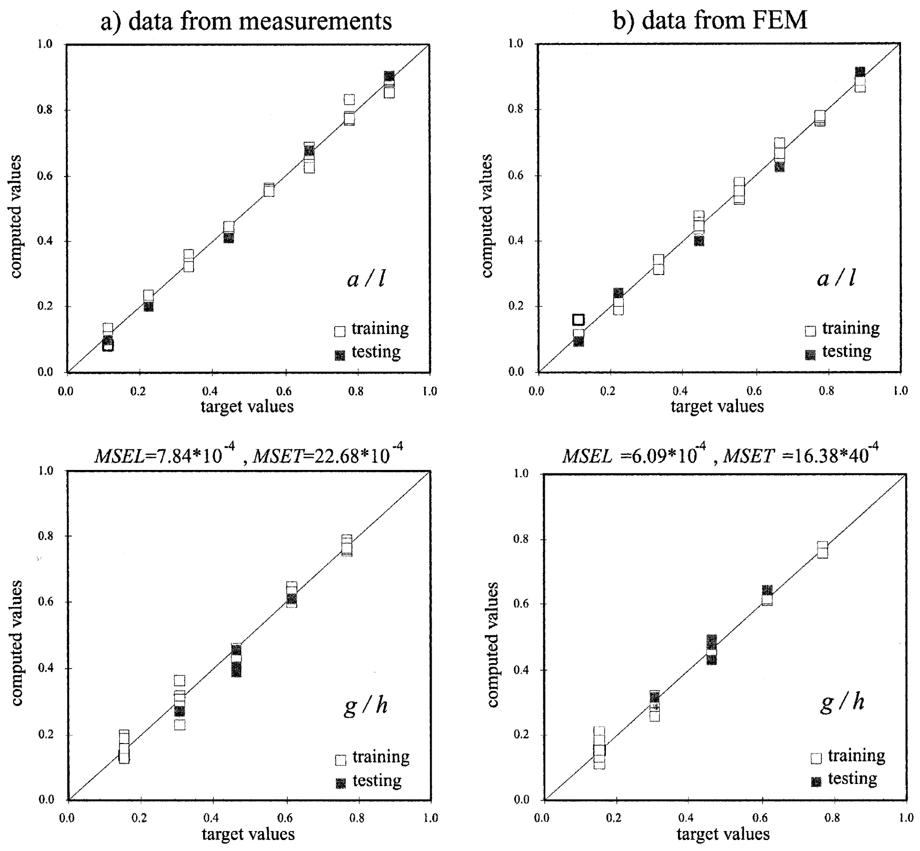

Fig. 10. Neural identification of notch location a/l and depth g/h for patterns taken from: (a) tests, (b) FEM.

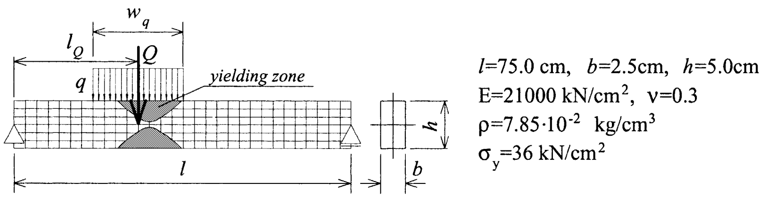

Fig. 11. Data of considered beam and load parameters.

analysis of ill-posed problems. RNNs can be used to noisy data in order to omit an over-fitted neural approximation. These questions were discussed by Haykin [18] and applications of RNNs to the analysis of measured data was recently used by Adeli and Wu [19].

The considered RNN is composed of a layer of the radial basis function (RBF) neurons, cf. Fig. 14.

The approximation function is assumed in the form:

$$y(\boldsymbol{x}) = \sum_{p=1}^{H} w_p G(\boldsymbol{x}, \boldsymbol{x}_p), \tag{22}$$

where Gaussian activation function is commonly used:

$$G(\boldsymbol{x}, \boldsymbol{x}_p) = \exp(-\beta\|\boldsymbol{x}, \boldsymbol{x}_p\|^2) = \exp\left[-\beta\sum_{j=1}^{N}(x_j - x_j^{(p)})^2\right] \quad \text{for } \beta > 0. \tag{23}$$

In Fig. 14b the unidimensional activation function is shown $G(x, x_c) = F(x; x_c) = \exp(-\beta|x - x_c|^2)$, where $\beta = 1/(2\sigma^2)$.

In Ref. [19] the number of Gaussian neurons was assumed to be equal the number of training patterns $H = L$. The weights w_p are computed from the following equation:

$$(\boldsymbol{G} + \lambda \boldsymbol{I})\boldsymbol{w} = \boldsymbol{t}, \tag{24}$$

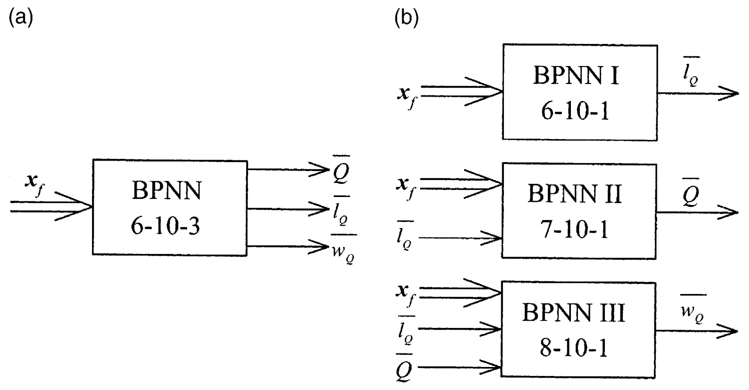

Fig. 12. (a) One-level BPNN, (b) three levels, cascade BPNN.

where

$$G_{L\times L} = \begin{bmatrix} G(x_1,x_1) & G(x_1,x_2) & \cdots & G(x_1,x_L) \\ G(x_2,x_1) & G(x_2,x_2) & \cdots & G(x_2,x_L) \\ \vdots & \vdots & & \vdots \\ G(x_L,x_1) & G(x_L,x_2) & \cdots & G(x_L,x_L) \end{bmatrix},$$

$$w_{L\times 1} = \{w_1, w_2, \ldots, w_L\}, \quad t_{L\times 1} = \{t_1, t_2, \ldots, t_L\}, \tag{25}$$

$t \in R^N$ – target vector, $\lambda > 1$ – regularization parameter. In what follows the root-mean-square-error is used:

$$\mathrm{RSME}(\beta; S) = \sqrt{\frac{1}{S}\sum_{p=1}^{S}[t_p - y(x_p)]^2}, \tag{26}$$

where $S = L$, V – number of patterns in the training and validation sets, respectively.

In Gaussian function (23) factor β has to be computed. The computation corresponds to the cross-validation procedure formulation as the minimum problem:

$$\min_{\beta} \mathrm{RMSE}(\beta, V) \rightarrow \beta_{\mathrm{opt}}. \tag{27}$$

Function RMSE $(\beta; V)$ is usually convex and the computation of β_{opt} can be easily performed numerically, e.g. by means of gradient formula. The above described idea of the RNN formulation was developed in Ref. [20], where more complicated RBFs were used. The parameters of those RBFs were there computed by the error back-propagation algorithm.

3.2. Estimation of concrete fatigue durability – application of RNN

Tests on concrete specimens subjected to cyclic compressive loads were considered in Ref. [21]. In case of experimental evidence related to more than 400 specimens both neural simulation and identification problems were put together in Ref. [22]. In what follows only the simulation problem is discussed.

The mapping of four variables x_j to one variable y was considered in Refs. [21,22]. These variables constitute the components of the input and output vectors:

$$x_{4\times 1} = \{f_c, f_{cN}/f_c, R, f\}, \quad y = \log N, \tag{28}$$

where f_c – concrete compression strength, f_{cN} – concrete dynamic strength for compression, $R = \sigma_{\min}/\sigma_{\max}$ – coefficient of cycle asymmetry, f – frequency, N – number of fatigue cycles.

Examples of tests are shown in Fig. 15, taken from Ref. [21]. As can be seen the results are very scattered (noisy).

In the case of considered problem the set of $P = 218$ patterns was randomly split into $L = 109$ and $V = 109$ patterns which were used for the training and validation, respectively. In Fig. 16a the relationship between the training and validation errors RMSE(β) is shown. The optimal $\beta_{\mathrm{opt}} = 0.95$ corresponds to min RMSE (β; V). In Fig. 16b values of $\log N$ estimated by the regularization NN versus measured $\log N$ are shown. In the same figure values of linear correlation coefficients r, r_L and r_V are written, corresponding to all patterns and to training and validation sets of patterns.

The achieved results of RNN estimation are very close to those obtained by a BP neural network. The multi-layer network of structure 4-5-4-1 was trained up to $S = 4000$ epochs and from the results discussed in Ref. [22] only two correlation coefficients are quoted: $r_L = 0.857$, $r_V = 0.826$.

The application of RNN is superior to the application of BPNN because in case of RNN there are no problems with the network design. Also the training process is much simpler and quicker than for BPNN.

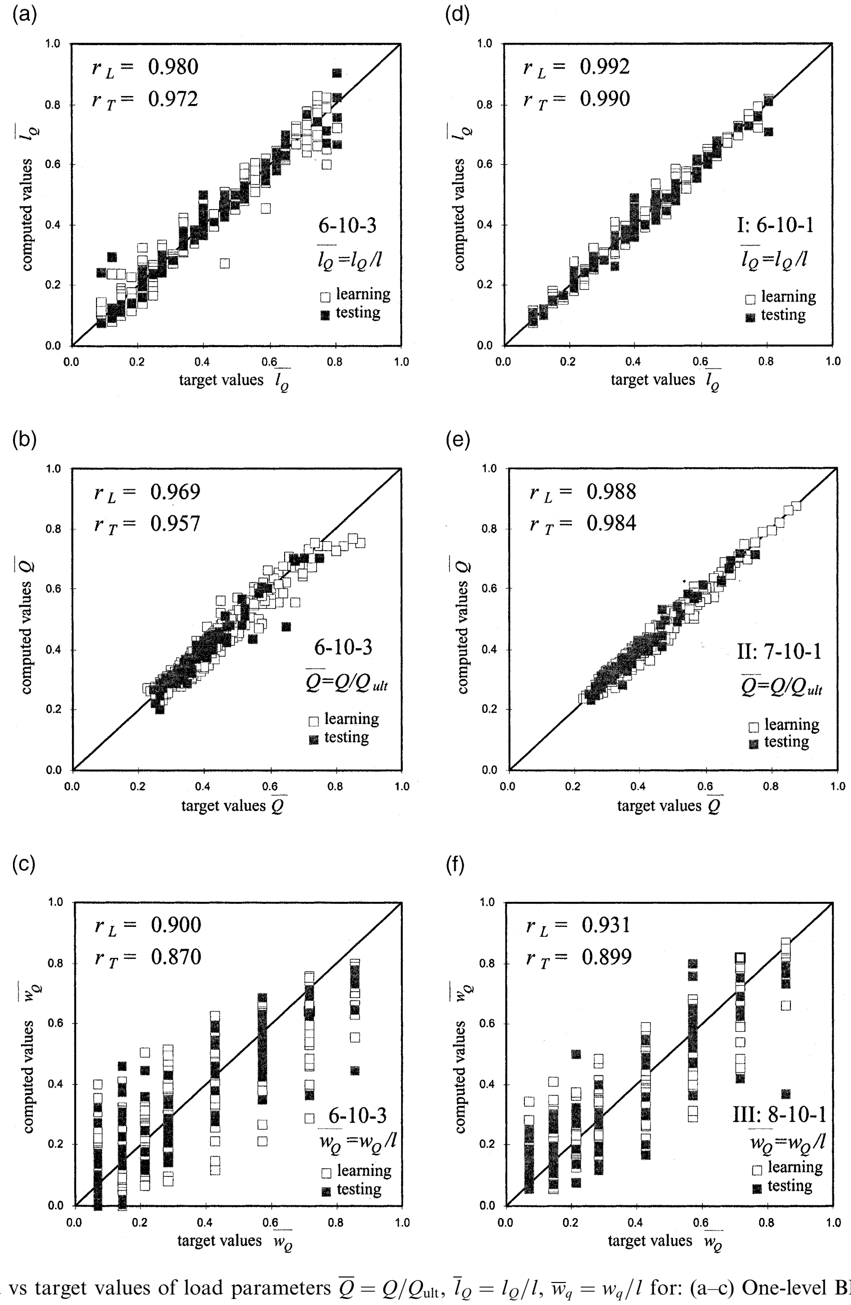

Fig. 13. Computed vs target values of load parameters $\overline{Q} = Q/Q_{\text{ult}}$, $\overline{l}_Q = l_Q/l$, $\overline{w}_q = w_q/l$ for: (a–c) One-level BPNN, (d–e) Cascade BPNN.

In Fig. 15 the neural prediction of the number of fatigue cycles was compared with estimations by an empirical formula derived in Ref. [21].

3.3. Hopfield network

The Hopfield neural network (HNN) is a *recurrent network*, i.e. a network with feed-back. The architecture of HNN is shown in Fig. 17. Only initial values of variables $x_j(0)$ are introduced into the network and then they are processed at fixed values of the network parameters $w_{ij} = w_{ji}$ and b_i for $i, j = 1, \dots, N$.

In the first phase HNN is trained by a special learning rule by means of the so-called *fundamental memory patterns* $\boldsymbol{x}_f^{(p)}$. In the *retrieval phase* a new pattern (usually noisy pattern $\boldsymbol{x}' = \boldsymbol{x}_f + \boldsymbol{\xi}$) is *recog-*

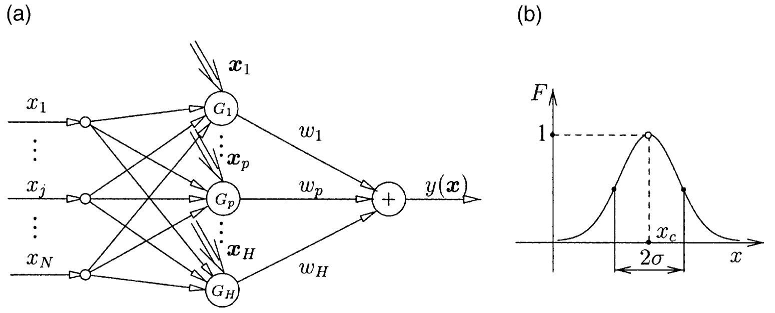

Fig. 14. (a) Regularization neural network, (b) Gaussian radial basis function.

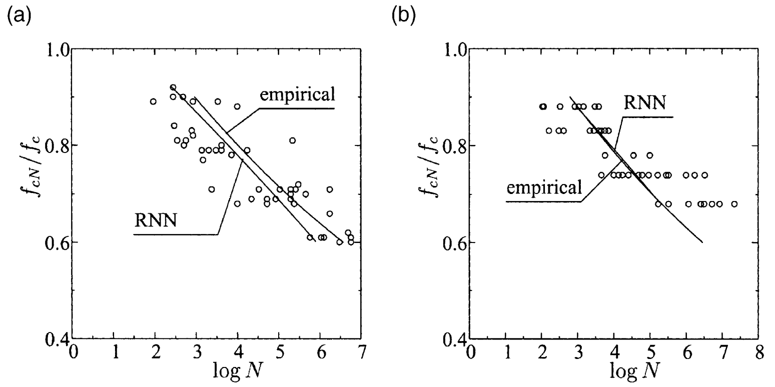

Fig. 15. Results of tests quoted in Ref. [21], performed by: (a) Antrim and Mc Laughlin (1959) for $f_c = 10.0$ MN/M^2, $R = 0.253$, $f = 10.0$ Hz, (b) Weigler and Freitag (1975) for $f_c = 28.0$ MN/M^2, $R = 0.025$, $f = 10.0$ Hz.

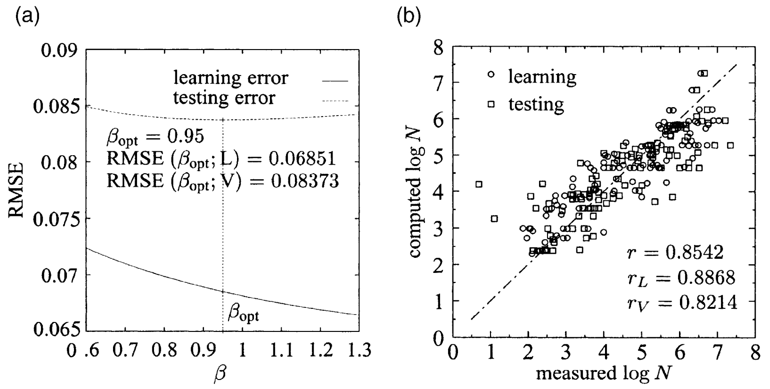

Fig. 16. (a) RMSE errors as functions of β parameter, (b) RNN estimation of fatigue cycle numbers vs results of laboratory tests.

nized, i.e. $\boldsymbol{x}' \rightarrow \boldsymbol{x}_f^{(r)}$ for a certain number of iteration steps.

The approach mentioned above is used in discrete HNNs in which the inputs x_j are binary or bipolar digits. The discrete HNN is usually designed for patterns recognition. The *continuous HNN* (also called Hopfield–Tank NN), i.e. with real numbers x_j can be used for the analysis of mathematical programming problems. That is why this version of HNN is discussed in what follows since it can be widely used in the analysis of mechanical problems.

HNNs are classified as dynamic networks, contrary to statical networks, e.g. networks with supervised learning (both input and output vectors are used in the training process). The retrieval phase of HNN can be treated as a dynamic process in which a stable equilibrium state of the network is searched. This can be analyzed by Lyapunov's method.

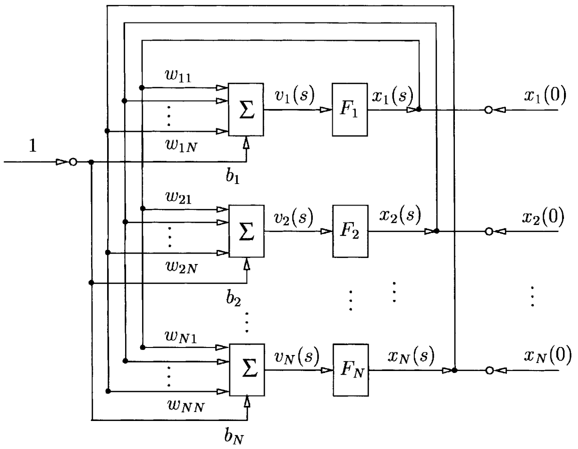

Fig. 17. Architecture of Hopfield neural network.

For HNN the *energy function* is defined:

$$E(\boldsymbol{x}(s)) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}x_i(s)x_j(s) - \sum_i b_i x_i(s) + \sum_i \int_0^{x_i} F_i^{-1}(x)\,\mathrm{d}x, \tag{29}$$

Function (29) is also valid for the dynamic, continuous in time, process, which is described by a set of ordinary equations:

$$\tau_i \frac{\mathrm{d}v_i}{\mathrm{d}t} = -\left.\frac{\partial E}{\partial x_i}\right|_{x_i(t)=F_i(v_i(t))} = -v_i + \sum_{j=1}^{N} w_{ij}x_j + b_i, \quad x_i = F_i(v_i(t)) \quad \text{for } i = 1,\dots,N, \tag{30}$$

where τ_i – time constants. Eq. (30) are called equations of HNN dynamics or the HNN *evolutionary equations*. The stable equilibrium state corresponds to the stopping criteria:

$$\min_t E(t) = 0 \Longleftrightarrow \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = 0. \tag{31}$$

HNN is an analogue to the circuit model which can be used to the analysis of various problems of mathematical programming in real time t.

3.4. Analysis of an elastic, plane stress problem with unilateral constraints – application of HNN

In the papers by Panagiotopoulos and his associates [4,23] it was proved that HNN can be used to the analysis of quadratic programming problems with unilateral constraints:

$$\min\left\{\tfrac{1}{2}\boldsymbol{q}^{\mathrm{T}}\boldsymbol{k}\boldsymbol{q} - \boldsymbol{p}\boldsymbol{q} | \boldsymbol{q} \geqslant 0\right\}, \tag{32}$$

where the FEM notation is used: $\boldsymbol{k}$ – stiffness matrix, $\boldsymbol{q}, \boldsymbol{p}$ – vectors of nodal displacements and equivalent load forces, respectively.

After substitution

$$w_{ij} = \begin{cases} -k_{ij} & \text{for } i \neq j, \\ -k_{ij}+1 & \text{for } i = i, \end{cases} \quad b_i = p_i,$$
$$x_i = q_i = F(v_i) = \begin{cases} v_i & \text{for } v_i > 0, \\ 0 & \text{for } v_i \leqslant 0, \end{cases} \tag{33}$$

and assumption: $\tau_i = 1$ for $i = 1,\dots,N$. Eq. (30) can be written in the form:

$$\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} = (-\boldsymbol{k}\boldsymbol{q} + \boldsymbol{p})|_t \equiv \boldsymbol{r}_t, \tag{34}$$

where $\boldsymbol{r}_t = \boldsymbol{r}(t)$ – vector of residual forces.

Eq. (34) was successfully used by Panagiotopoulos et al. [4] to the analysis of problems of fracture mechanics and plasticity. In Ref. [24] the Panagiotopoulos approach was improved. Eq. (34) corresponds in fact to the gradient methods of solution of linear algebraic equations. That is why more efficient equations of conjugate gradient method can be used:

$$\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} = \boldsymbol{r}_t - \beta_t \boldsymbol{r}_{t-1}, \quad \text{where} \quad \beta_t = \frac{\boldsymbol{r}_t^{\mathrm{T}}\boldsymbol{r}_t}{\boldsymbol{r}_{t-1}^{\mathrm{T}}\boldsymbol{r}_{t-1}}, \tag{35}$$

and a more general activation function was applied:

$$F(v_i) = \begin{cases} 0 & \text{for } v_i \leqslant a_i \text{ or } v_i \geqslant b_i, \\ v_i & \text{elsewhere.} \end{cases} \tag{36}$$

Relationship (36) enables us to consider supporting constraints with fissures.

In Ref. [24] an example of angle plate was analyzed, cf. Fig. 18. The plate was discretized by 17 FEs. Unilateral constraints were taken into account at the node no. 1 ($b_1 = 0.03$ mm) and nos. 70–88 ($a_{70} = \cdots = a_{88} = -0.02$ mm). After condensation of DOFs to nine vertical displacements v_1, v_{70}–v_{88} (Eq. (35)) were analyzed by the Runge–Kutta umbending formula of the 5th and 6th orders [25]. Assuming error $\max|v_i(t) - v_i(t-1)| < 1 \times 10^{-9}$ the load programs composed of the following steps: (I) $\Delta\Lambda_1 = 10$, $10 \times (-1)$, (II) $\Delta\Lambda_2 = 10 \times 1$, $-\Lambda_1$ were carried out. Those programs corresponded to closed loops with inverse load increments for loading and unloading parts.

Realization of 11 load steps in the frame of load program I gave the same results as for program II. It needed about 630 iteration steps. In case of lack of the constraint in node 1 (very high value of distance b in Fig. 18a) the number of iterations increased up to 4000 steps. This effect is in agreement with Panagiotopoulos conclusion: the increasing number of bilateral constraints increases the number of iterations (steps) related to numerical integration of evolutionary Eq. (35).

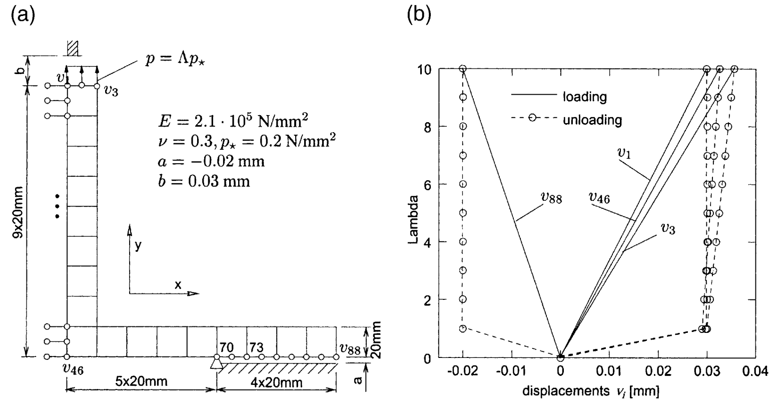

Fig. 18. (a) Data and FE mesh of angle plate, (b) vertical displacements v_i of selected nodes i.

Another conclusion of great practical importance is that for linear field equations the reaching of prescribed values of load parameter does not need increments of load. During numerical integration of Eq. (35) all boundary conditions related to unilateral constraints are fulfilled automatically.

4. Final remarks

4.1. Conclusions

On the basis of the achieved results the following general conclusions can be stated:

1. BPNNs can be efficiently applied to the implementation of hybrid programs in which neural procedures are used instead of numerical procedures.
2. Neural procedures seem to be numerically efficient in the analysis of constitutive equations.
3. ANNs suit well to processing experimental data, taken both from tests on laboratory specimens and measurements on real structures.
4. BPNNs can be used for selection of principal components in the input vector.
5. Data preprocessing to the spectral space opens the door to the analysis of various problems of identification and assessment of MS.
6. RNN is suitable to the analysis of problems with noisy data. RNNs are simpler for design and training than BPNNs.
7. HNN can be efficiently applied to the analysis of mathematical programming problems. This concerns also problems with unilateral constraints.

4.2. Prospects of ANNs applications

ANNs are a new tool for the analysis of many problems of mechanics which are difficult to solve by means of standard computational methods (first of all this concerns the FEM). ANNs seem to be very promising in the analysis of the following problems:

1. Simulations and identification problems for which there are not known mathematical models or given ‘a priori’ relationship between input and output data. This concerns also problems for which a mathematical model needs updating of parameters, cf. Ref. [26].
2. Analysis of problems for which only experimental evidence exists, usually with noisy or incomplete data.
3. Analysis of mathematical programming problems, also with unilateral constraints. This concerns especially contact problems and problems of fracture mechanics [23].
4. Hybrid strategies in which ANNs are a complementary part (e.g. neural procedures) of standard computational systems.
5. Assessment problems, closely related to expert systems in which ANNs enable us to increase accuracy of estimation and flexibility with respect to updating the stored knowledge.
6. Joining ANNs with other fields of soft computing is especially promising. Evolutionary methods can be efficiently used for learning and designing of ANNs. Fuzzy networks can be used for linguistic variables and in a much more efficient analysis than the one performed only by classical ANNs, cf. Ref. [4].

Acknowledgements

The authors wish to express their cordial gratitude to Doctors E. Pabisek and K. Kuźniar, Mrs. E. Han-Wiercińska and Mrs. K. Nuckowska, Messrs. B. Miller and R. Putanowicz, for the assistance of preparing the examples of neural network applications and the manuscript.

Research related to the problems, reported in this paper, was financially supported by the Polish Committee for Scientific Research, Grant nos. 8 T11F 022 14 and 7 T07A 003 13. The authors wish to express their grateful acknowledgement for this support.

References

[1] Adeli H, Yeh C. Peceptron learning in engineering design. Microcomput Civil Engng 1989;4:247–56.

[2] Waszczyszyn Z. Some recent and current problems of neurocomputing in civil and structural engineering. In: Topping BHV, editor. Advanced in computational structures technology. Edinburgh: Civil-Comp Press; 1996. p. 43–58.

[3] Waszczyszyn Z. Some new results in applications of backpropagation neural networks in structural and civil engineering. In: Topping BHV, editor. Advances in engineering computational technology: Civil-Comp Press; Edinburgh. 1998. p. 173–87.

[4] Waszczyszyn Z, editor. Neural networks in the analysis and design of structures. CISM Courses and Lectures – no. 404. New York: Springer; 1999.

[5] Paez ThL. Neural networks in mechanical system simulation, identification and assessment. Shock Vibr 1993;1:177–99.

[6] Zell A, editor. SNNS – Stuttgart neural network simulator. User's Manual, ver. 4.1, Institute for Parallel and Distributed High Performance Systems, Report no. 6/95, University of Stuttgart, 1995.

[7] Waszczyszyn Z, Cichoń C, Radwańska M. Stability of structures by finite element methods. Amsterdam: Elsevier; 1994.

[8] Mucha G, Waszczyszyn Z. Hybrid neural-network/computational program for bending analysis of elastoplastic-beams. Proc 13th Polish Conf Comp Meth Mech, Poznań, 1997. p. 949–54.

[9] Waszczyszyn Z, Pabisek E. Hybrid NN/FEM analysis of the elastoplastic plane stress problem. Comput Assisted Mech Engng Sci 1999;6:177–88.

[10] Ciesielski R, Kuźniar K, Maciąg E, Tatara T. Empirical formulae for fundamental natural periods of buildings with load bearing walls. Archives Civil Engng 1992;38:291–9.

[11] Maciąg E. Experimental evaluation of changes of dynamic properties of buildings on different grounds. Earthquake Engng Struct Dyn 1986;14:925–32.

[12] Kuźniar K, Waszczyszyn Z. BP neural identification of fundamental natural periods of prefabricated buildings. In: Rutkowski L, Tadeusiewicz R, editors. Proceedings of the Fourth Conference on Neural Networks and Their Applications, Zakopane-Częstochowa, 1999. p. 344–9.

[13] Ziemiański L, Łakota W. The use of neural networks for damage detection in clamped-free beams. Proc 13th Polish Conf Comput Meth, Poznań, 1997. p. 1447–54.

[14] Łakota W. Damage detection in beam structures (in Polish). Ofic Wyd Polit Rzeszowskiej, Rzeszów, 1999.

[15] Adina – In Manual. Adiana-Plot Manual. ADINA R&D Inc., 1992.

[16] Miller B, Piątkowski G, Ziemiański L. Beam yielding load identification by neural networks. Comput Assisted Mech Engng Sci 1999;6:449–67.

[17] Miller B, Piątkowski G, Ziemiański L. Identification of the yielding load of a simple supported beam. In: Rutkowski L, Tadeusiewics R, editors. Proceedings of the Fourth Conference on Neural Networks and Their Applications, Zakopane-Częstochowa, 1999. p. 363–8.

[18] Haykin S. Neural networks – a comprehensive foundation. New York: Macmillan; 1994.

[19] Adeli H, Wu M. Regularization neural network for construction cost estimation. J Constr Engng Mgmt 1998;124:18–24.

[20] Putanowicz R, Waszczyszyn Z. Neural network identification of concrete properties. Proceedings of the Fifth International Conference on Engineering Applications of Neural Networks, Warszawa, 1999. p. 286–91.

[21] Furtak K. Strength of concrete subjected to cyclic loads (in Polish). Actives Civil Engng 1984;30:677–98.

[22] Kaliszuk J, Urbańska A, Waszczyszyn Z, Furtak K. Neural analysis of concrete fatigue durability on the base of laboratory tests (in Polish). Proceedings of the 45th Polish Civil Engineering Conference, Krynica, 1999. v. 4, p. 22–34.

[23] Theocaris PS, Panagiotopoulos PD. Neural networks for computing in fracture mechanics – methods and prospects of applications. Comput Meth Appl Mech Engng 1993;106:213–28.

[24] Waszczyszyn Z, Pabisek E. Application of a Hopfield type neural network to the analysis of unilateral, elastic stress problem. In: Łakota W, Waszczyszyn Z, Ziemiański L, editors. Proc 14th Polish Conf Comput Meth Mech, Rzeszów, 1999. p. 383–4.

[25] Englen-Hüllges G, Uhling F. Numerical algorithms with C. Berlin: Springer; 1996.

[26] Ziemiański L, Miller B, Piątkowski G. Dynamic model updating by neural networks. Proceedings of the Fifth International Conference on Engineering Applications of Neural Networks, Warszawa, 1999. p. 177–82.