Computer Assisted Mechanics and Engineering Sciences, 7: 781–793, 2000.

Dynamic model updating using neural networks

Leonard Ziemiański and Bartosz Miller
Department of Structural Mechanics, Rzeszów University of Technology
ul. W. Pola 2, PL-35959 Rzeszów, Poland

(Received November 3, 1999)

The paper presents an application of Artificial Neural Networks for updating a mathematical model of the structure based on dynamic parameters. Neural networks which predict the value of selected stiffness or concentrated masses on the basis of Frequency Response Function (FRF) have been built. Two types of neural networks have been used for this task: multi-layer feed-forward (MLFF) networks with different learning algorithms and networks with radial basis function (RBF). Preceding the update, the FRF is compressed in order to reduce the number of input values necessary for updating the model.

1. Introduction

Accurate and representative computer models are necessary to predict the dynamic characteristics of the structure under the study. A dynamic model of a structure (dynamic model may be built with the finite element method or discrete modelling) is verified by building a physical model (usually a laboratory model is built), testing, and comparison of responses from the dynamic model (analytical) and physical model (experimental). There are often discrepancies between these results. For that reason the dynamic model must be modified until a good agreement between the responses of these models is achieved. The process of modifying the dynamic model is called updating [2, 6]. The main task in the model updating is to determine coefficient matrices $\mathbf{M}$, $\mathbf{C}$, $\mathbf{K}$ on the basis of the response of the structure. In this paper a dynamic model updating technique using neural networks is developed and demonstrated. Neural networks have been applied successfully to many diverse areas of application. They are particularly applicable to problems in which plenty of examples are available but it is difficult to specify an explicit algorithm, such as character recognition or time series prediction, etc. [3, 9]. To solve this problem multi-layer feed-forward (MLFF) neural networks and neural networks with radial basis function (RBF) are used [9].

In recent years a significant amount of work on generating and testing different updating methods has been done [2, 4, 5, 6, 7]. The resulting algorithms may be split into several categories based on whether they work in the frequency or modal domains and whether they adjust the mass and stiffness matrices directly or make parametric changes to the model:

- direct techniques based on modal data,
- iterative techniques based on modal data,
- iterative techniques based on frequency domain data.

Direct methods require mode shape information and they reproduce the measured data exactly. The main drawback of these methods is that the responses of all degrees of freedom are needed. It has been shown that the direct methods are not appropriate for model updating as the results obtained are not physically meaningful. The iterative methods improve the correlation between the measured data and the analytical model by using penalty function. In penalty function mode shape and eigenvalue or FRF data directly without extracting the natural frequencies and mode shapes can be involved. The penalty functions are generally non-linear of the parameters and so iterative procedure is required.

2. Problem description

A simple model of a 12-storey frame consisting of twelve concentrated masses coupled by springs is considered. The model has 12 degrees of freedom (see Fig. 1).

The model updating procedure consists of the following steps: 1) generation of a set of training data vectors based on the dynamic model, 2) training of the neural network with the training data, 3) exposition of expected data to obtain set of changes, 4) application of the changes to the original model in order to generate a new model, 5) repetition of the previous steps if necessary. This procedure leads to an updated model that can be updated again using the same method [4]. The most important question in the procedure is: "What kind of input data must be used to train the network?" In this paper Frequency Response Functions (FRF) are used as input data vectors. The Frequency Response Function is defined as the ratio of the Fourier transforms of the response and the excitation force time domain signals [1]. The FRF is usually obtained by auto-correlation and cross-correlation functions of time signals. An example of Frequency Response Function is shown in Fig. 2.

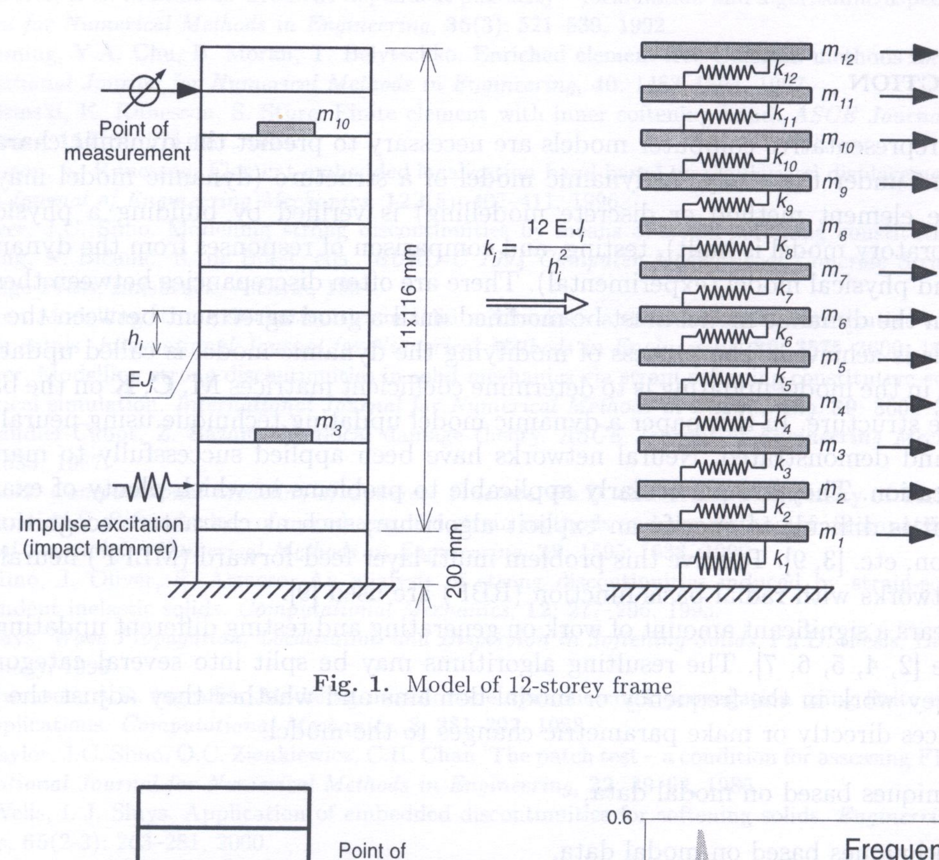

Fig. 1. Model of 12-storey frame

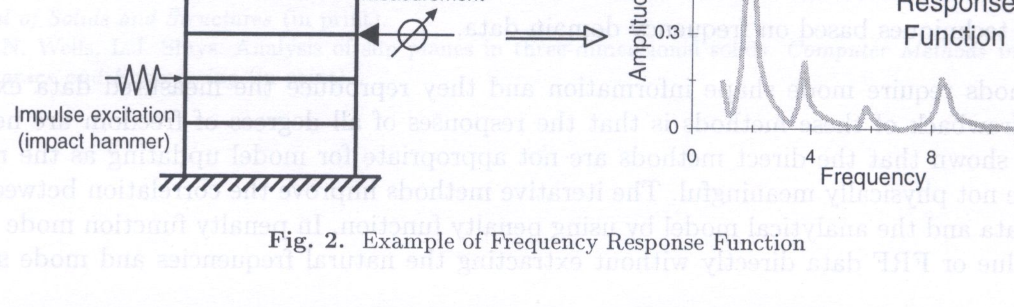

Fig. 2. Example of Frequency Response Function

However, the FRFs of even small models contain too many points to use them all as an input of a neural network. When the input vector has a high dimension, the number of learning patterns required for adequate network generalisation is also very high. This behaviour is known as the *curse of dimensionality* (for detailed discussion on that matter see [3]). Therefore it is necessary to reduce the number of inputs of the neural network. Data reduction could be achieved in many ways. The simplest method is discarding many of FRF data points. Another way is to work in the modal domain, using modal analysis to obtain natural frequencies and mode shapes of the structure [2, 4]. In this paper the input data have been reduced by the technique of compression by a neural network [8] or by calculating geometrical characteristics of selected FRF bands. In the previously presented updating algorithm two separate stages can be distinguished:

- extraction of model dynamical features from FRF using the so called Characteristic Replicator (ANN designed for data compression) or calculating geometrical characteristics of five selected FRF bands,
- building an ANN for the mathematical model updating based on a compressed FRF or geometrical characteristics of FRF bands.

This algorithm has been tested on 12-storey frame model for two problems:

- updating k_1 stiffness (1^{st} storey stiffness),
- updating two masses (on the 3^{rd} and 10^{th} storey – see Fig. 1).

3. Updating of k_1 stiffness

3.1. Characteristic Replicator

The model of a 12-storey frame is considered (see Fig. 1). An assumption that the only parameter which is updated is the stiffness of the first storey has been made. All other parameters are invariable. Learning and testing patterns have been calculated by numerical simulation for changes of k_1 stiffness in the range of 50%÷150% (altogether 102 different values of k_1). Additionally, the location of impulse excitation and the location of FRF measurements are shifting (adequately on the 1^{st}, 3^{rd}, 8^{th} and 10^{th} storey or on the 1^{st}, 5^{th}, 8^{th} and 10^{th} storey). This approach lets calculate 1632 different FRFs, 153 out of which are randomly selected as testing patterns, the remaining ones (1479) are the learning patterns.

The calculated FRFs contain 50 points. In order to separate the dynamic features from the FRF a Characteristic Replicator has been built. Different networks of architecture 50-h-50 (where h is the number of neurons in a hidden layer: 10, 8, 6, 5 or 4) have been learned to replicate at the output the vector FRF data points given at the input. During the replication in a hidden layer, which contains fewer neurons than the number of inputs, the compression of characteristic is performed. For example 50-8-50 net compresses information given at the input from 50 to 8 values, and then decompresses it back to 50 values. To obtain a network outputting a compressed FRF the output layer of the learned 50-h-50 network has been removed. The task of the output layer has been taken over by so far hidden layer. Networks 50-h give on the output FRF condensed into 10, 8, 6, 5 or 4 values.

The learning process of the 50-h-50 networks consisted of 2500 epochs. Figure 3 presents the results of FRF decompression by different networks for randomly chosen learning and testing patterns. Networks are able to decompress a FRF with accurate precision after compressing it in a hidden layer even to 4 or 5 values. It proves that it is possible to compress a FRF calculated for the mathematical model without a loss of the important information and a network used for model updating can base on condensed characteristics; there is no need to input all the 50 data point.

FRFs compressed by Characteristic Replicator are used as input vectors for neural networks updating model. These networks have one or two hidden layers consisting of various numbers of

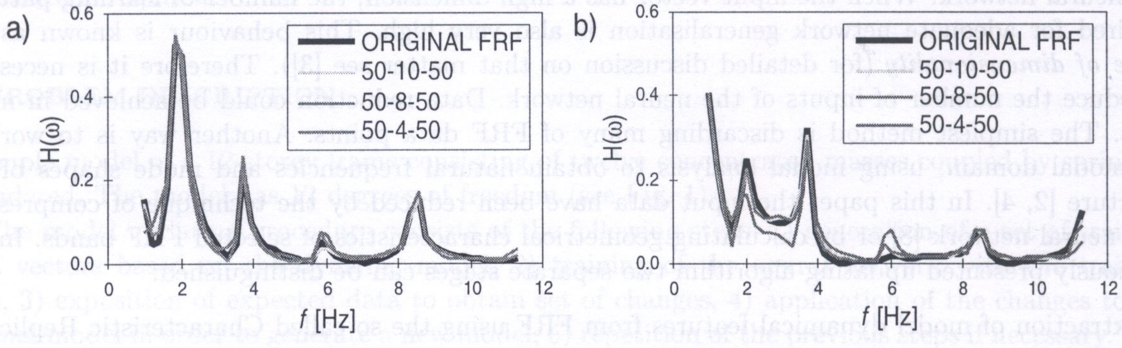

Fig. 3. Frequency Response Function of the model after compression and decompression by 50-h-50 networks: a) learning pattern, b) testing pattern.

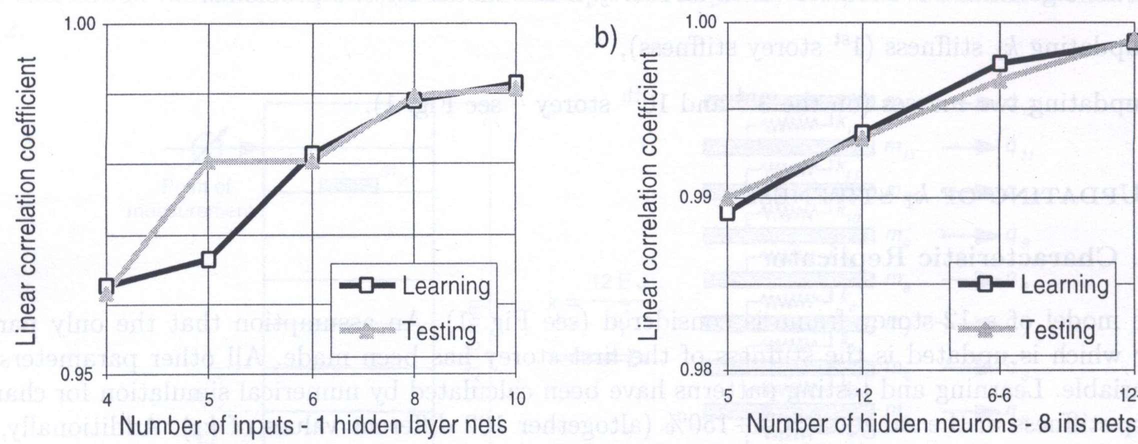

Fig. 4. Results from networks with different number of inputs and hidden neurons

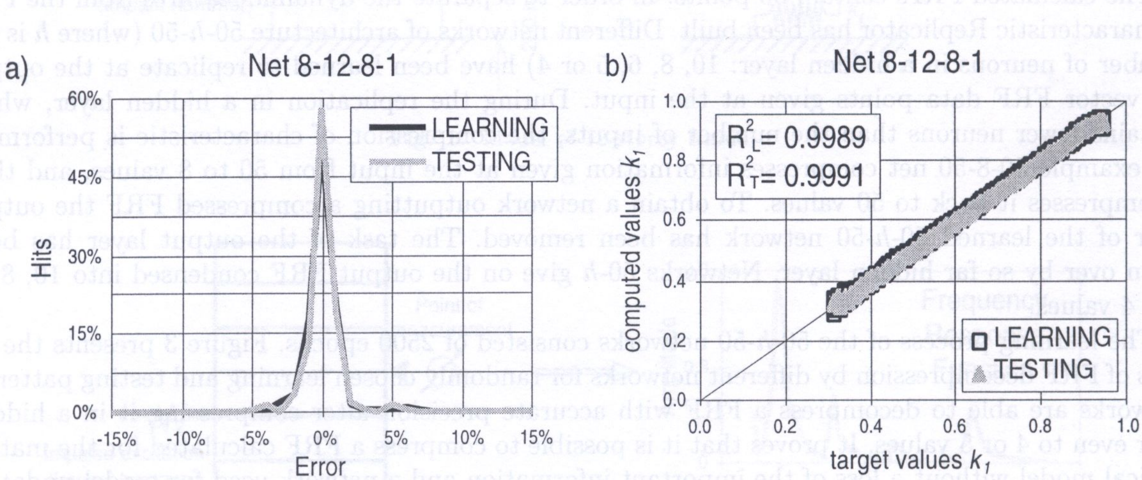

Fig. 5. Results from neural network updating of k_1 stiffness: a–b) 8-12-8-1

neurons. The output of the network equals the value of first storey k_1 stiffness. The learning process consists of 4000 epochs.

Figure 4a shows the relation between the number of network inputs and obtained results accuracy. Networks fed with FRF condensed to 4, 5 or 6 values predict k_1 stiffness with significantly worse precision than networks fed with FRF condensed to 8 or 10 values. Because the difference between the results from networks with 8 and 10 inputs are not large the decision to apply 8-input network has been made. Figure 4b shows the relation between the number of hidden neurons in 8-input networks and the obtained results accuracy. The best results are obtained from the network with 12 neurons in first hidden layer and 8 in the second one.

The detailed results for selected 8-12-8-1 network are shown in Fig. 5. Figure 5a shows the relation between the error of k_1 prediction and the percentage of occurrences of this value of error. In Fig. 5b the horizontal axis is the k_1 parameter determination error, the vertical axis is the percentage of the error occurrences.

The Characteristic Replicator has also been learned with FRF expressed in log scale or after 2500 epochs of learning it has been additionally learned with testing patterns (100 epochs of additional learning). The Characteristic Replicator has also been learned all over again using another learning algorithm and a grater number of learning epochs. Compressed FRF obtained from the new Characteristic Replicator were used to learn and test the network updating mathematical model. The detailed results from all tested networks are shown in Table 1.

Table 1. Collected results from MLFF networks updating k_1 stiffness using Characteristic Replicator

No.	Network architecture	Correlation coefficient R^2		Standard deviation	
		Learning	Testing	Learning	Testing
1	10-4-1	0.99160	0.99059	0.03322	0.03939
2	10-7-1	0.99530	0.99399	0.02593	0.03012
3	10-12-1	0.99812	0.99779	0.01675	0.01737
4	10-6-6-1	0.99816	0.99791	0.01691	0.01916
5	10-12-8-1	0.99924	0.99871	0.01076	0.01144
6	8-5-1	0.98908	0.98986	0.03777	0.04252
7	8-5-1 [1)]	0.98908	0.99011	0.03795	0.04172
8	8-12-1	0.99363	0.99342	0.03113	0.03147
9	8-6-6-1	0.99761	0.99676	0.02066	0.02547
10	8-12-8-1	0.99894	0.99909	0.01256	0.01217
11	8-6-6-1 [1)]	0.99762	0.99667	0.02062	0.02569
12	8-6-6-1 [2)]	0.99598	0.99548	0.02122	0.01959
13	6-4-1	0.98145	0.98036	0.04905	0.05034
14	5-3-1	0.96646	0.98052	0.07012	0.05624
15	4-3-1	0.96281	0.96180	0.06853	0.09144
16	4-6-1	0.98469	0.98529	0.05296	0.04841
17	4-6-6-1	0.98705	0.98146	0.04952	0.05010

[1)] CR additionally learned with the testing patterns
[2)] CR learned with FRF expressed in log scale

3.2. Using geometrical characteristics of selected frequency bands

The division of patterns into learning and testing ones was the same as for the networks based on compressed FRF. The use of Characteristic Replicator has been given up and the FRF consisting of 50 points has been divided into bands. Five of them have been taken to the further calculation, the remaining ones have been given up. The selected bands are shown in Figs. 6a and b.

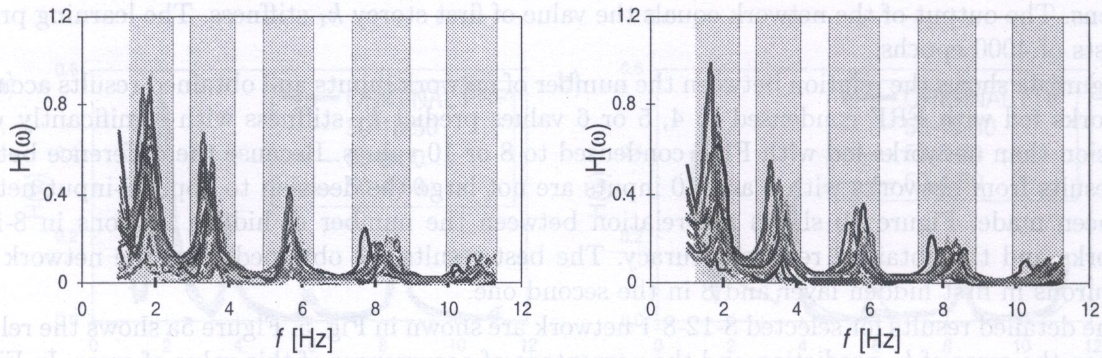

Fig. 6. Bands selected from the FRF: a) learning patterns, b) testing patterns

Considering each band as a flat figure some geometrical characteristics have been calculated:

- area A,

$$A = \iint_{\Omega} \mathrm{d}\Omega, \tag{1}$$

- two moments of inertia I_x and I_y,

$$I_x = \iint_{\Omega} y^2 \,\mathrm{d}\Omega, \qquad I_y = \iint_{\Omega} x^2 \,\mathrm{d}\Omega, \tag{2}$$

- co-ordinates of centre of gravity X_s and Y_s.

$$X_s = \frac{\iint_{\Omega} x \,\mathrm{d}\Omega}{\iint_{\Omega} \mathrm{d}\Omega}, \qquad X_s = \frac{\iint_{\Omega} y \,\mathrm{d}\Omega}{\iint_{\Omega} \mathrm{d}\Omega}. \tag{3}$$

Co-ordinate system for one selected band is shown in Fig. 7.

Calculated geometrical characteristics of selected bands were used as inputs to the nets updating k_1 stiffness. Detailed results from selected networks basing on geometrical characteristics of FRF bands as inputs are shown in Fig. 8.

In Fig. 8d there are shown results from the network, which had on input areas A of selected five bands and additionally five eigenfrequencies f within these bands. The values of eigenfrequencies were taken from FRF. The results from the networks with 10, 20 or 25 inputs are of the same quality. Basing on this the decision of application of 10-input networks has been made. In these networks each of five FRF bands is described by 2 parameters: area A and X_s co-ordinate of centre of gravity.

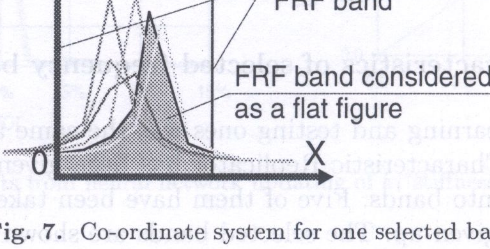

Fig. 7. Co-ordinate system for one selected band

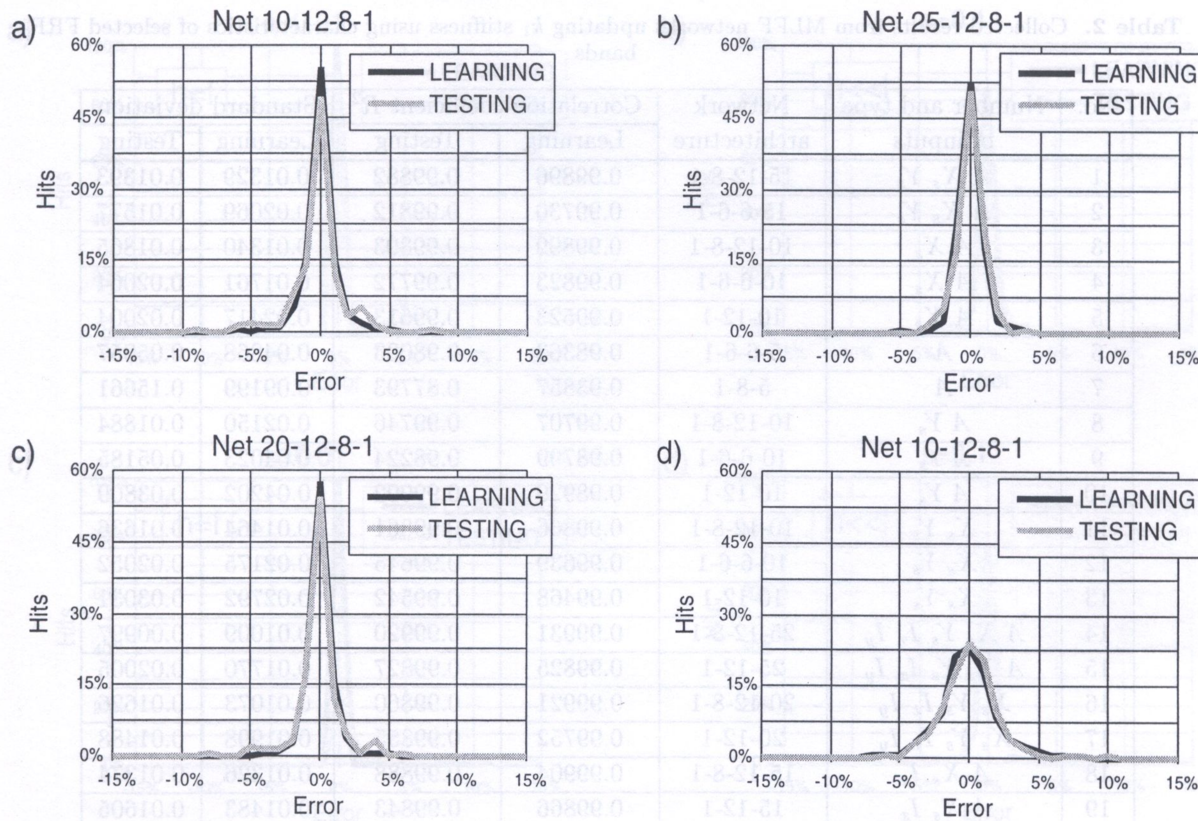

Fig. 8. Results from neural network updating of k_1 stiffness: a) A and X_s in input, b) A, X_s, Y_s, I_x and I_y on input, c) X_s, Y_s, I_x and I_y on input, d) A and f on input.

The detailed results from all tested networks are shown in Table 2. In some cases instead of X_s co-ordinate of the centre of gravity calculated in local co-ordinate system, different for each FRF band, there was used X_s^G calculated in global co-ordinate system. The results from the networks basing on geometrical characteristics are of the same quality as the results from the networks basing on condensed FRF.

3.3. Using networks with Radial Basis Function

Networks with radial basis function have been also tried out [10]. The networks have based on geometrical characteristics of selected FRF bands. The calculations were made with different values of σ (0.5, 0.02, 0.01, 0.001 and 0.0005). In Fig. 9a one can see the relation between σ and correlation coefficient R^2 between real and predicted values. The optimum value of σ is 0.01. Figure 9b shows the detailed results from the network with σ=0.01.

This network has had a number of hidden neurons equal to the number of learning patterns. The division of patterns into learning and training sets was the same as for the MLFF networks, so the number of hidden neurons in this network was 1479. In Figs. 10a and 10b one can see the comparison of the results from this network with the network with number of hidden neurons less than the number of learning patterns. Next a new division of patterns into learning and training set has been made. In order to do it every second pattern from previous learning set was moved to testing set. The new number of learning set was 739, the testing set consisted of 892 patterns. New networks with number of hidden neurons equal to or less than the number of learning patterns were learned. The results are shown in Figs. 10c and 10d.

As one can notice the generalisation of network with the number of hidden neurons less than the number of learning patterns is much better than the generalisation of the network with the number

Table 2. Collected results from MLFF networks updating k_1 stiffness using characteristics of selected FRF bands

No.	Number and type of inputs	Network architecture	Correlation coefficient R^2 Learning	Correlation coefficient R^2 Testing	Standard deviation Learning	Standard deviation Testing
1	$A\ X_s\ Y_s$	15-12-8-1	0.99896	0.99882	0.01329	0.01393
2	$A\ X_s\ Y_s$	15-6-6-1	0.99730	0.99812	0.02069	0.01577
3	$A\ X_s$	10-12-8-1	0.99899	0.99803	0.01340	0.01865
4	$A\ X_s$	10-6-6-1	0.99823	0.99772	0.01761	0.02064
5	$A\ X_s$	10-12-1	0.99523	0.99613	0.02417	0.02004
6	A	5-6-6-1	0.98363	0.98033	0.04868	0.05057
7	A	5-8-1	0.93857	0.87793	0.09199	0.15661
8	$A\ Y_s$	10-12-8-1	0.99707	0.99746	0.02150	0.01884
9	$A\ Y_s$	10-6-6-1	0.98799	0.98224	0.04023	0.05185
10	$A\ Y_s$	10-12-1	0.98929	0.99092	0.04202	0.03809
11	$X_s\ Y_s$	10-12-8-1	0.99866	0.99861	0.01464	0.01636
12	$X_s\ Y_s$	10-6-6-1	0.99639	0.99645	0.02175	0.02052
13	$X_s\ Y_s$	10-12-1	0.99468	0.99542	0.02792	0.03031
14	$A\ X_s\ Y_s\ I_x\ I_y$	25-12-8-1	0.99931	0.99920	0.01009	0.00997
15	$A\ X_s\ Y_s\ I_x\ I_y$	25-12-1	0.99825	0.99827	0.01770	0.02005
16	$X_s\ Y_s\ I_x\ I_y$	20-12-8-1	0.99921	0.99860	0.01073	0.01626
17	$X_s\ Y_s\ I_x\ I_y$	20-12-1	0.99752	0.99855	0.01998	0.01488
18	$A\ X_s\ I_x$	15-12-8-1	0.99904	0.99888	0.01326	0.01374
19	$A\ X_s\ I_x$	15-12-1	0.99866	0.99843	0.01483	0.01605
20	X_s	5-5-5-1	0.94911	0.94909	0.07472	0.05560
21	X_s	5-6-6-1	0.94184	0.93859	0.08049	0.06235
22	X_s^G	5-6-6-1	0.91453	0.94680	0.09196	0.05784
23	$A\ X_s^G$	10-12-1	0.98979	0.99167	0.04313	0.03890
24	$A\ X_s^G$	10-6-6-1	0.99248	0.99367	0.03854	0.02968
25	$A\ X_s^G$	10-12-8-1	0.99458	0.99635	0.03204	0.02246
26	$A\ f$	10-12-1	0.98745	0.98397	0.04161	0.04210
27	$A\ f$	10-6-6-1	0.98801	0.98349	0.04126	0.05258
28	$A\ f$	10-12-8-1	0.99543	0.99565	0.02699	0.02773
29	f	5-6-6-1	0.97273	0.97646	0.05127	0.04617
30	f	5-8-1	0.94481	0.95173	0.07387	0.06842

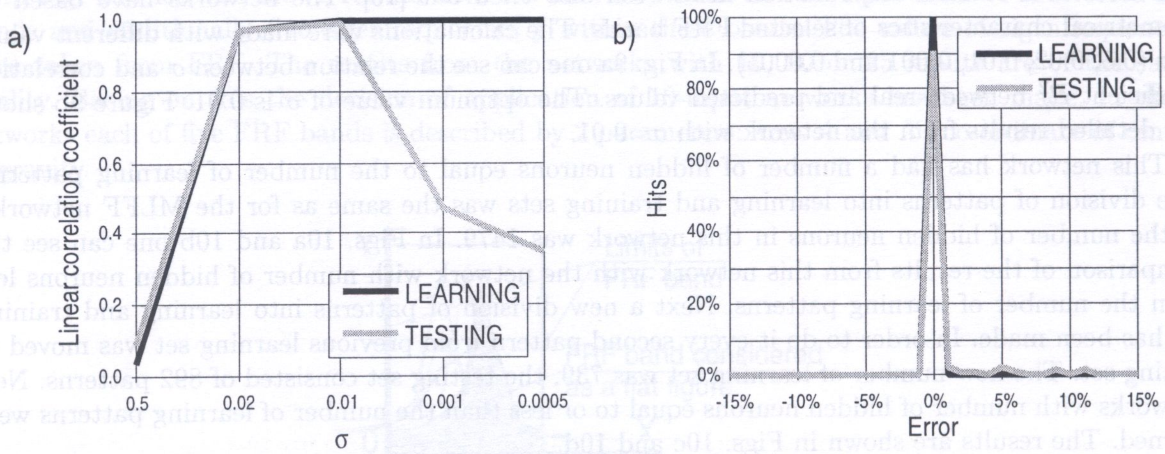

Fig. 9. Correlation coefficient R^2 for RBF networks with different parameters

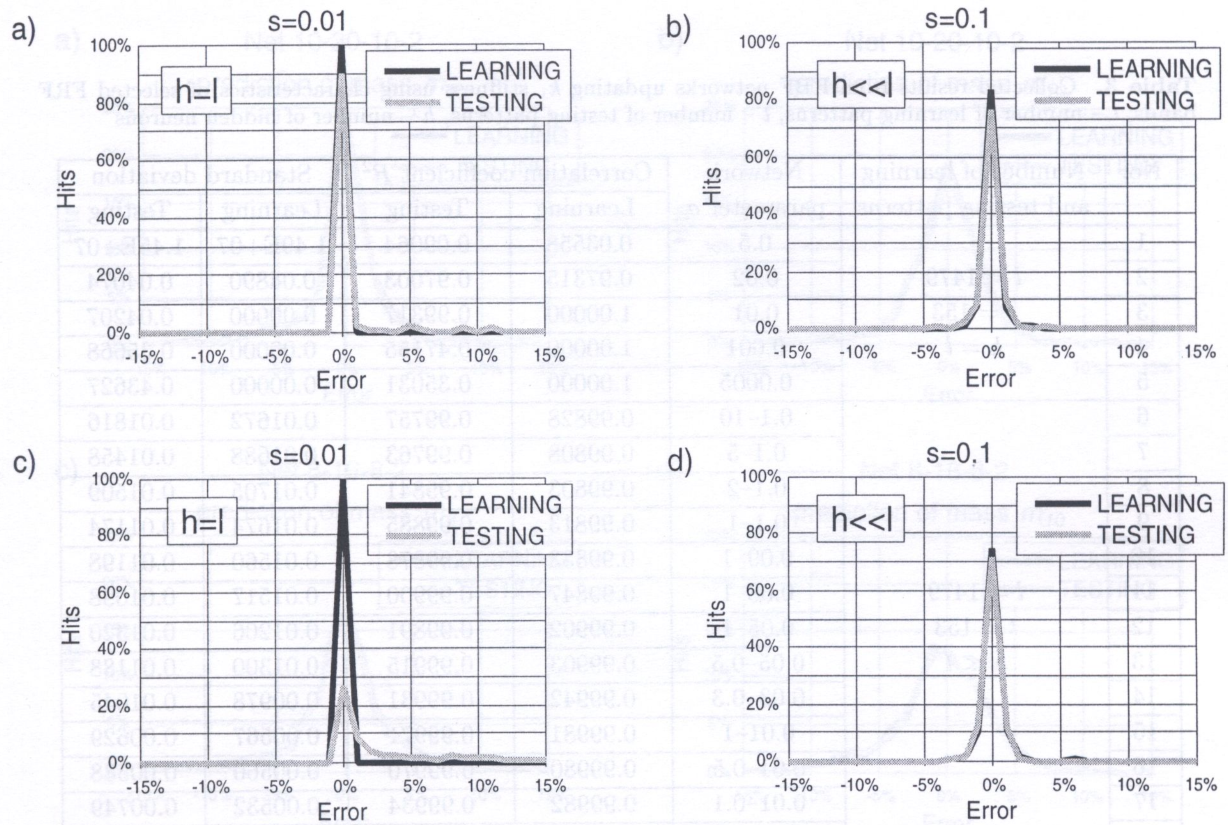

Fig. 10. Results from RBF network updating of k_1 stiffness

of hidden neurons equal to the number of learning patterns. Table 3 presents the results from all considered RBF networks.

4. Updating of m_3 and m_{10} masses

Next the updating of two parameters of the model simultaneously has been made. These parameters are two masses m_3 and m_{10} (see Fig. 1). Learning patterns have been calculated by numerical simulation for m_3 and m_{10} masses changing in the range of 50–150% (altogether 2601 different values). Testing patterns have been calculated in the same way but with different starting values of both masses. The number of patterns in testing set is 2601.

FRF compression is done by Characteristic Replicator of a structure 50-h-50. One- (50-10-50 and 50-8-50) and three-hidden layer networks (50-20-6-20-50) are considered. Figure 11 shows FRF compressed and then decompressed by 50-h-50 networks for the patterns with the biggest error.

The output layer of the learned 50-h-50 network is removed and the task of the output layer is taken over by so far the only or the last hidden layer. 50-h network outputs a FRF condensed to 10, 8 or 6 values. Data obtained from Characteristic Replicator is used as input vectors for the networks predicting selected parameters: masses m_3 and m_{10}. The results from these networks are shown in Fig. 12 and Table 4.

Figures 13b and 13c show the value of the error of masses prediction in relation to both masses. The surfaces are very flat, which means that there are no areas of mass values where network has difficulties with achieving precise prediction. The only areas with higher error are the surfaces boundaries for minimal values of masses, where the error of each network prediction is higher. Figure 13a shows the value of Characteristic Replicator error. This error is transferred to the end networks predicting the values of both masses.

Table 3. Collected results from RBF networks updating k_1 stiffness using characteristics of selected FRF bands; l – number of learning patterns, t – number of testing patterns, h – number of hidden neurons

No.	Number of learning and testing patterns	Network parameter σ	Correlation coefficient R^2		Standard deviation	
			Learning	Testing	Learning	Testing
1		0.5	0.03558	0.09064	1.49E+07	1.45E+07
2	$l = 1479$	0.02	0.97315	0.97003	0.04890	0.04074
3	$t = 153$	0.01	1.00000	0.99317	0.00000	0.04207
4	$h = l$	0.001	1.00000	0.47155	0.00000	0.35668
5		0.0005	1.00000	0.35031	0.00000	0.43627
6		0.1–10	0.99828	0.99757	0.01672	0.01816
7		0.1–5	0.99808	0.99763	0.01688	0.01458
8		0.1–2	0.99803	0.99841	0.01705	0.01509
9		0.1–1	0.99813	0.99885	0.01674	0.01174
10		0.09–1	0.99833	0.99878	0.01560	0.01198
11	$l = 1479$	0.08–1	0.99847	0.99900	0.01517	0.01098
12	$t = 153$	0.05–1	0.99902	0.99891	0.01206	0.01320
13	$h \ll l$	0.05–0.5	0.99903	0.99915	0.01300	0.01188
14		0.03–0.3	0.99942	0.99931	0.00978	0.01045
15		0.01–1	0.99981	0.99971	0.00567	0.00629
16		0.01–0.5	0.99980	0.99970	0.00560	0.00588
17		0.01–0.1	0.99982	0.99934	0.00532	0.00749
18		0.4–4	0.99233	0.99375	0.03965	0.03285
19		0.3–5	0.99446	0.99276	0.03171	0.03459
20		1	0.00848	0.00575	1.40E+06	1.39E+06
21	$l = 739, t = 892$	0.5	0.00095	0.00000	8.09E+02	7.81E+02
22	$h = l$	0.01	1.00000	0.72443	0.00000	0.23871
23		0.001	1.00000	0.09437	0.00000	0.50194
24	$l = 739, t = 892$	0.01–1	0.99961	0.99959	0.00747	0.00783
25	$h \ll l$	0.01–0.1	0.99961	0.99956	0.00729	0.00759

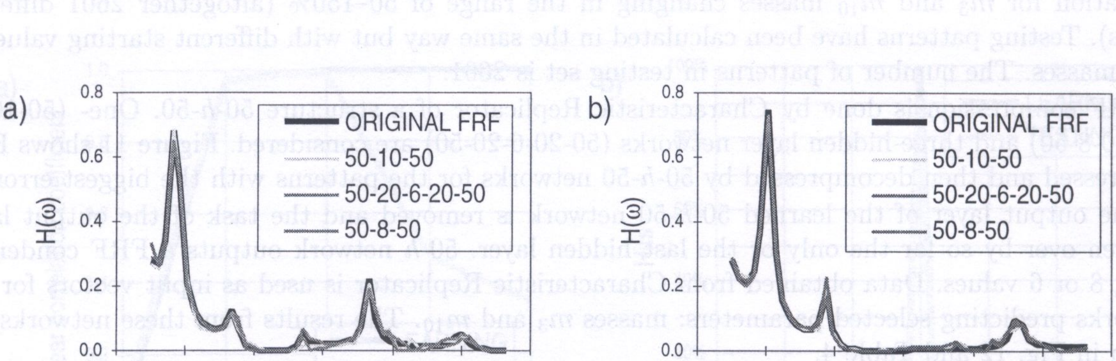

Fig. 11. Frequency Response Function of the model after compression and decompression by 50-h-50 networks: a) learning pattern, b) testing pattern.

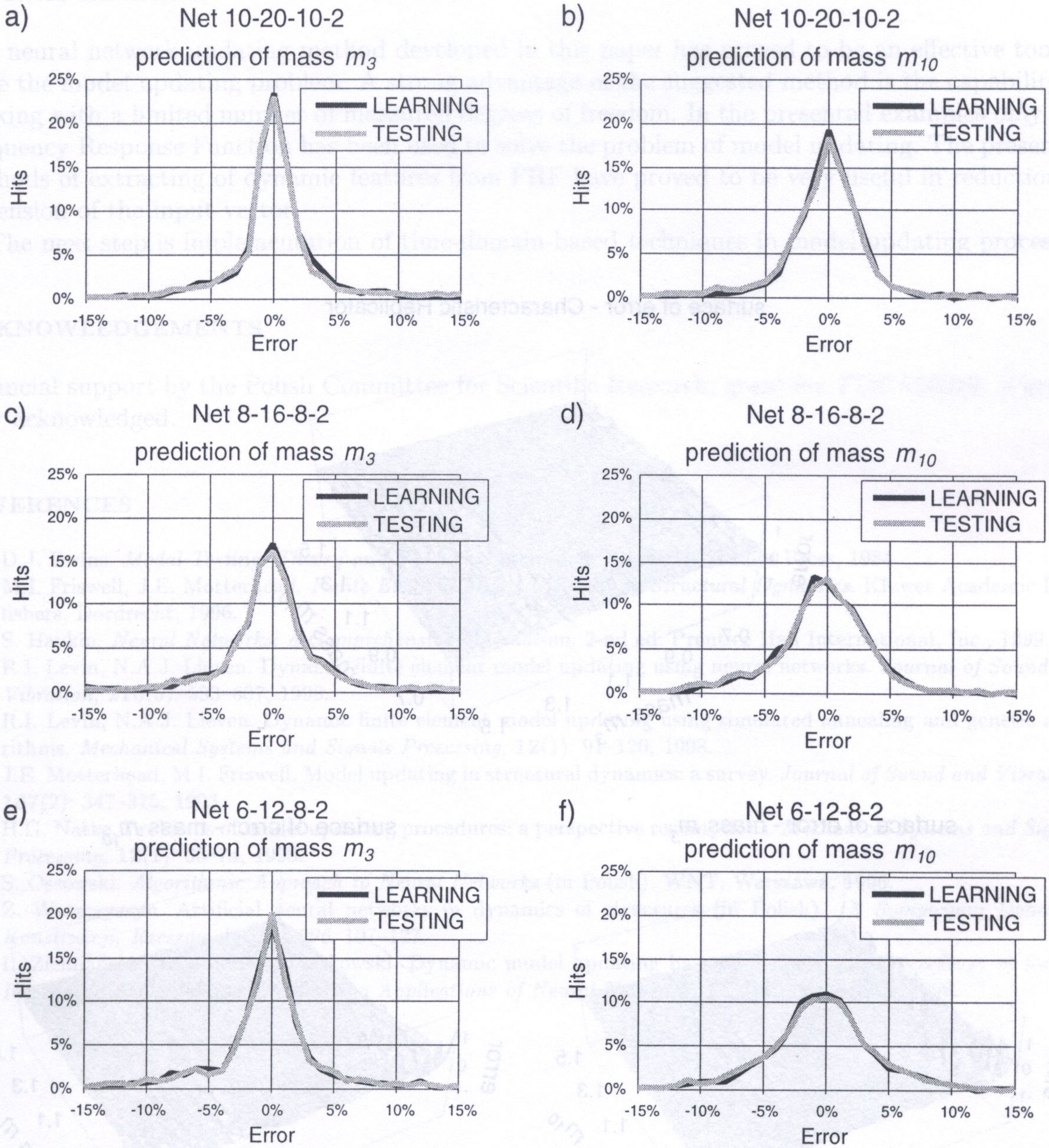

Fig. 12. Results from neural network updating of m_3 and m_{10} masses a–b) 10-20-10-2, c–d) 8-16-8-2, e–f) 6-12-8-2

Table 4. Collected results from RBF networks updating m_3 and m_{10} using Characteristic Replicator

No.	Network architecture	Mass m_1				Mass m_2			
		R^2		Standard deviation		R^2		Standard deviation	
		Learning	Testing	Learning	Testing	Learning	Testing	Learning	Testing
1	10-20-10-2	0.99585	0.99286	0.06786	0.07392	0.99547	0.99323	0.07323	0.08168
2	8-16-8-2	0.99276	0.98685	0.09317	0.09027	0.99103	0.98873	0.09605	0.07982
3	8-16-8-2 [1)]	0.99183	0.98945	0.10358	0.10868	0.98709	0.98779	0.09696	0.08670
4	6-12-8-2	0.99288	0.99019	0.08939	0.08734	0.99007	0.98536	0.09939	0.09489
5	10-8-6-2	0.98938	0.98728	0.10462	0.09849	0.98894	0.98838	0.11386	0.08703

[1)] network with learning set substituted with testing set

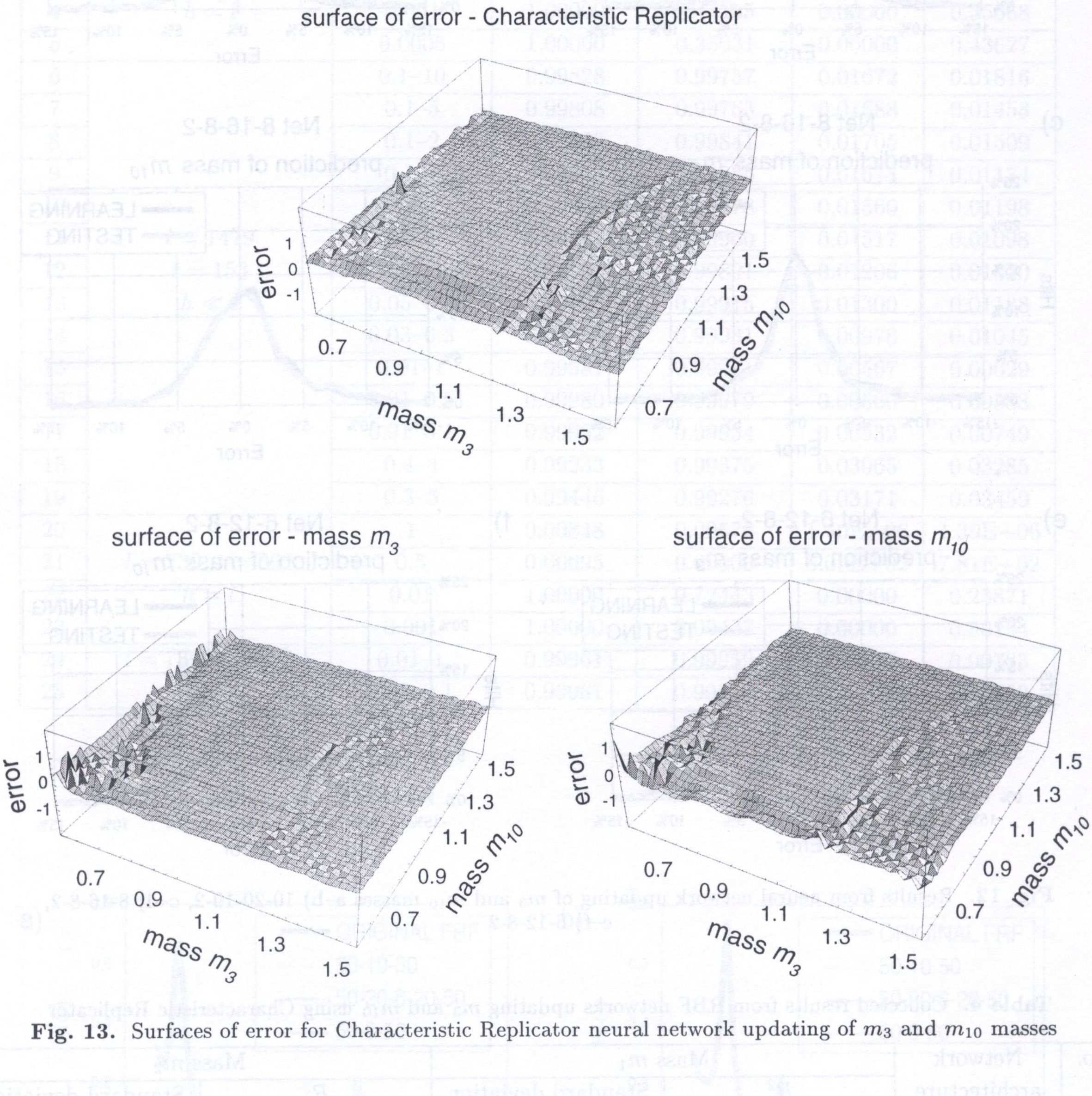

Fig. 13. Surfaces of error for Characteristic Replicator neural network updating of m_3 and m_{10} masses

5. Final remarks

The neural network updating method developed in this paper has proved to be an effective tool to solve the model updating problem. A strong advantage of the suggested method is the capability of working with a limited number of measured degrees of freedom. In the presented examples only one Frequency Response Function has been used to solve the problem of model updating. The presented methods of extracting of dynamic features from FRF have proved to be very useful in reduction of dimension of the input vector.

The next step is implementation of time-domain-based techniques in model updating process.

Acknowledgements

Financial support by the Polish Committee for Scientific Research, grant no. 7T07A00313, is gratefully acknowledged.

References

[1] D.J. Ewins. *Modal Testing: Theory and Practice.* Letchworth: Research Studies Press, 1984.

[2] M.I. Friswell, J.E. Motterhead. *Finite Element Model Updating in Structural Dynamics.* Kluwer Academic Publishers, Dordrecht, 1996.

[3] S. Haykin. *Neural Networks: a Comprehensive Foundation*, 2-nd ed. Prentice Hall International, Inc., 1999.

[4] R.I. Levin, N.A.J. Lieven. Dynamic finite element model updating using neural networks. *Journal of Sound and Vibration*, **210**(5): 593–607, 1998.

[5] R.I. Levin, N.A.J. Lieven. Dynamic finite element model updating using simulated annealing and genetic algorithms. *Mechanical Systems and Signals Processing*, **12**(1): 91–120, 1998.

[6] J.E. Motterhead, M.I. Friswell. Model updating in structural dynamics: a survey. *Journal of Sound and Vibration*, **167**(2): 347–375, 1993.

[7] H.G. Natke. Problems of model updating procedures: a perspective resumptions. *Mechanical Systems and Signals Processing*, **12**(1): 65–74, 1998.

[8] S. Ossowski. *Algorithmic Approach to Neural Networks* (in Polish). WNT, Warszawa, 1996.

[9] Z. Waszczyszyn. Artificial neural networks in dynamics of structures (in Polish). *IX Sympozjum Dynamiki Konstrukcji, Rzeszów-Jawor 1996*, 107–127.

[10] L. Ziemiański, B. Miller, G. Piątkowski. Dynamic model updating by neural networks. *Proceedings of the 5th International Conference Engineering Applications of Neural Networks*, 177–182, Warszawa, 1999.