MODIFIED NEURAL NETWORK STRUCTURE

L. Körösi, J. Paulusová

Institute of Robotics and Cybernetics,

Slovak University of Technology, Faculty of Electrical Engineering and Information Technology

Abstract

The artificial neural network (ANN) belongs to dynamically developed soft computing methods. These advanced computational methods are widely applied in several modern scientific and industrial fields due to their unique properties. Artificial neural networks have the ability of modeling of nonlinear systems and their generalization. The aim of the proposed paper is to present a modified ANN structure based on hidden neuron modifications for modeling of nonlinear processes. The modifications of multilayer perceptron (MLP) were realized and validated in Matlab.

1 Introduction

An ANN is an information processing system that is inspired by the human nervous system. The key element of this system is the structure of the information processing system. It is composed of a large number of interconnected processing neurons working together to solve specific problems. An ANN is designed for a specific application, such as nonlinear process modeling, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. These synaptic connections are called weights.

Perceptron designed by Frank Rosenblatt is considered as a fundamental element of artificial neural networks. The MLP is typically a three layer neural network which has the following characteristics: learning, adaptation, generalization, robustness, associative storage and others. The first layer consists of linear neurons. These neurons are responsible for preprocessing of the input signals to the hidden layer. Hidden layer (or layers) consists of nonlinear neurons. These neurons proceed with the nonlinear transformation of input signals. The nonlinear activation functions are the same for all hidden neurons. The output layer consists of linear neurons for linear processing of data from hidden layer(s). For the purposes of modeling of nonlinear systems, three-layer MLP is sufficient.

The most frequently used method for designing the ANN structure are the trial and error methods. The ANN structure and learning proposal for nonlinear process modeling includes the choice of number of input neurons, number of hidden neurons, activation function types, training method and parameters for the training algorithm. The number of output neurons is given by the process output. The aim of the proposed paper is to present a modified ANN structure based on hidden neuron activation function modifications for modeling of nonlinear processes.

2 Modified ANN structure

An example of MLP with 2 neurons in input layer, 3 neurons in hidden layer and 1 neuron in output layer is shown in Fig. 1. In the input and output layers are linear activation functions, but in some cases the output layer may also use a non-linear activation function. Nonlinear (sigmoidal) activation functions are used for modeling and identification in the hidden layer. Sigmoidal function is defined as a monotonically rising, smooth and bounded function. Sigmoidal activation functions allow achieving high neuron sensitivity for small signals, for higher signal levels the sensitivity decreases. Neural network is able to process signals in a sufficient range of dynamics without congestion on arrival of too large signals and vice-versa, that the arrival of very small signals ensures its sufficient sensitivity. In addition, that the activation functions provide nonlinear behavior of ANN, they are several times differentiable and therefore they may be used for a number of training methods. The most common methods for training of MLP are the back propagation method, the conjugate gradient method, quasi-Newton methods or the method of Levenberg-Marquardt. Differentiability of activation functions allows the use of ANN to predict the behavior of the process in different control algorithms, where the known model is used in the control signal optimization.

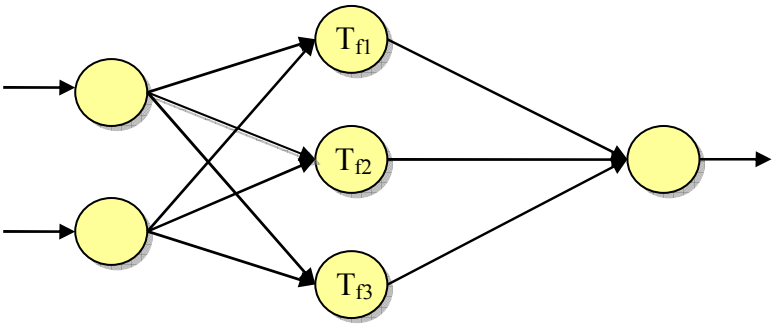

Figure 1: Example of MLP structure (2-3-1)

The standard method of construction a MLP is using homogeneous activation functions in hidden layer ($T_{f1} = T_{f2} = \ldots = T_{fn}$). The modified method is using non-homogeneous activation functions which ensure more sensibility of neurons (ANN). In the proposed paper the combinations of *tansig* and *logsig* activation functions were tested.

Transfer characteristic of *tansig* activation function used in Matlab is shown in Fig. 2. The output of the neuron is calculated as $tansig(n) = 2/(1+\exp(-2*n))-1$.

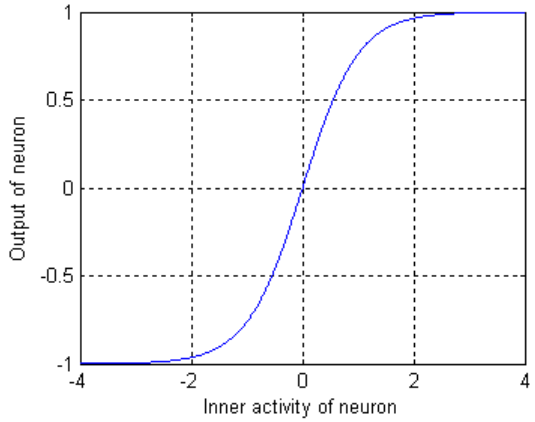

Figure 2: Transfer characteristic of *tansig* activation function

Transfer characteristic of *tansig* activation function is shown in Fig. 3. The output of the neuron is calculated as $logsig(n) = 1 / (1 + \exp(-n))$.

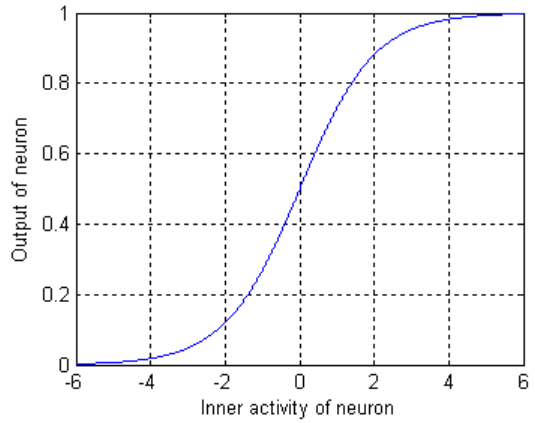

Figure 3: Transfer characteristic of *logsig* activation function

3 Case study

To verify the modified ANN accuracy the following nonlinear system was considered:

$$\ddot{y}(t) + 0.06\dot{y}(t)\left[y^2(t) + 1\right] + 0.2y(t) - 0.4u(t) = 0 \tag{1}$$

The recommended sampling time is between 1/12 and 1/6 of the rise time T_n (Fig. 4). The sample time can be chosen from interval <0.5, 1>. In this paper T_s = 0.7sec was used.

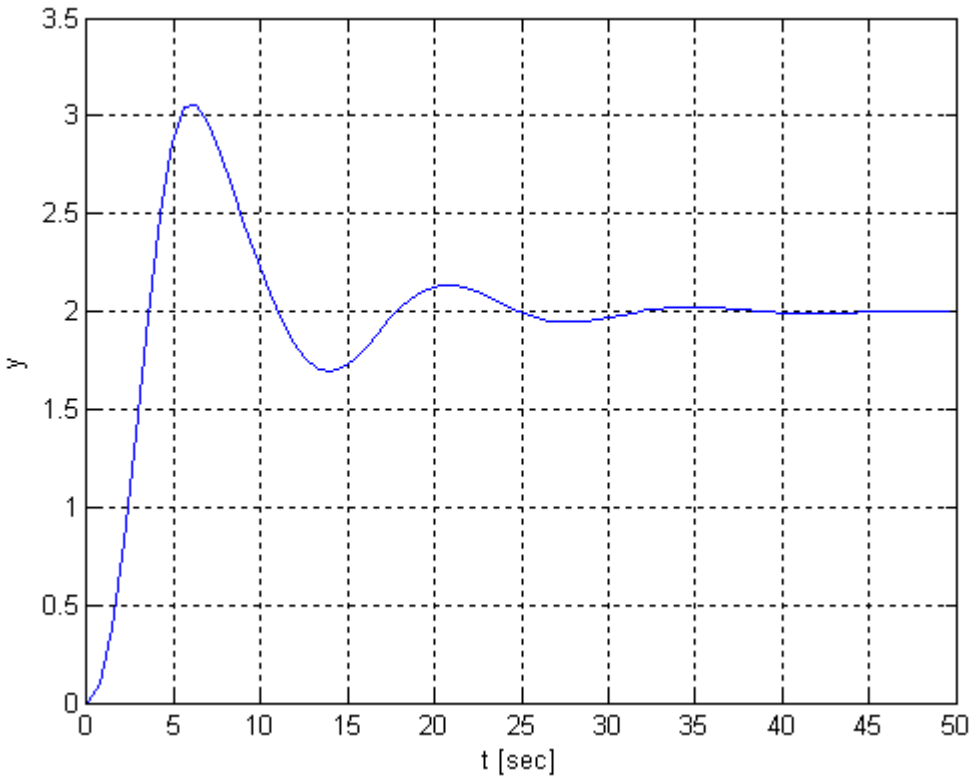

Figure 4: Step response of the nonlinear model for input signal $u(k) = 1$

The inputs are in the range from 1 to 2.5 due to output settling time. Using input variables outside this range considerably increases settling time for simulation purposes. The training and testing data are shown in Fig. 4, 5, 6 and 7. The following Neural Network (NN) ARX structures were tested:

- NN1: 2 inputs ($u(k)$, $y(k)$) and 1 output ($y(k+1)$) with 4, 5 and 6 hidden neurons.
- NN2: 4 inputs ($u(k)$, $u(k-1)$, $y(k)$, $y(k-1)$) and 1 output ($y(k+1)$) with 8, 9 and 10 hidden neurons.
- NN3: 6 inputs ($u(k)$, $u(k-1)$, $u(k-2)$, $y(k)$, $y(k-1)$, $y(k-2)$) and 1 output ($y(k+1)$) with 12, 13 and 14 hidden neurons.

The initial weights were generated randomly from the interval (-1, 1). For each modified and unmodified ANN five different initialization were considered to compare their accuracy. Activation functions of hidden neurons were changed gradually from activation functions *logsig* to activation function *tansig*. Each MLP was trained 5000 epochs. As a learning algorithm the back propagation algorithm was used. During ANN training local errors were recorded to compare the final training error after 5000 epochs with the possible "global" minimum which occurs during learning process.

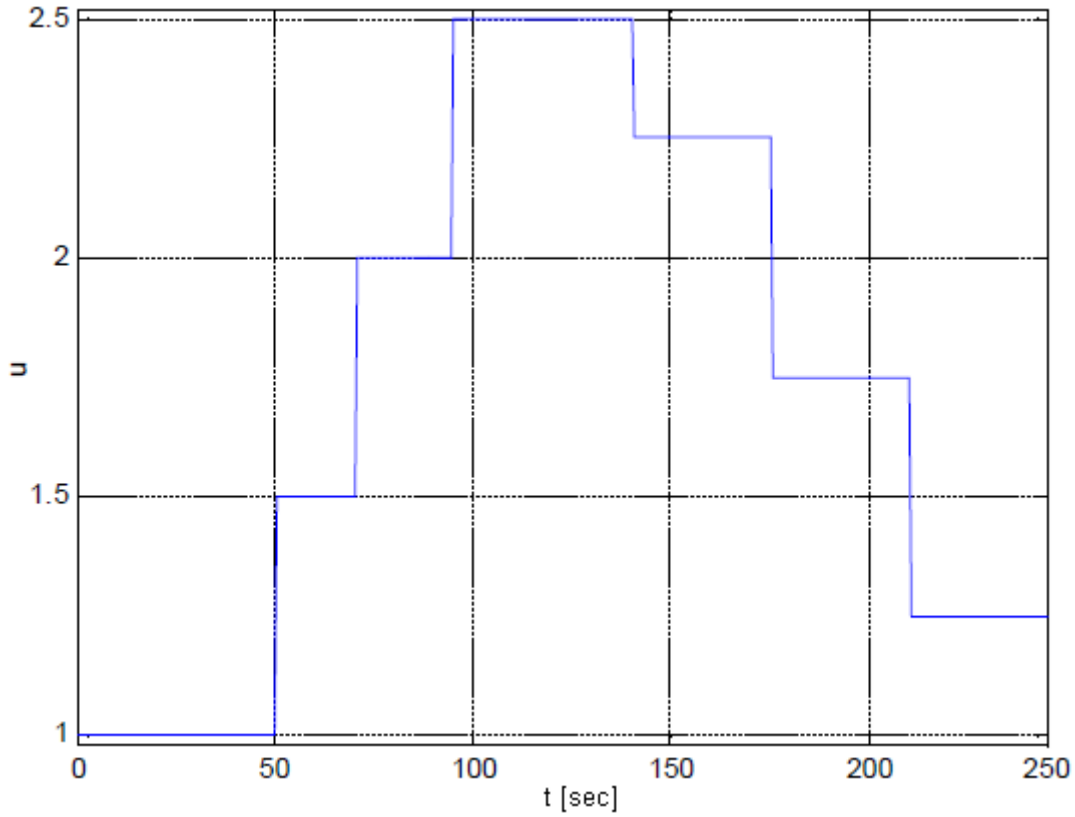

Figure 5: Time response of the input signal (training data)

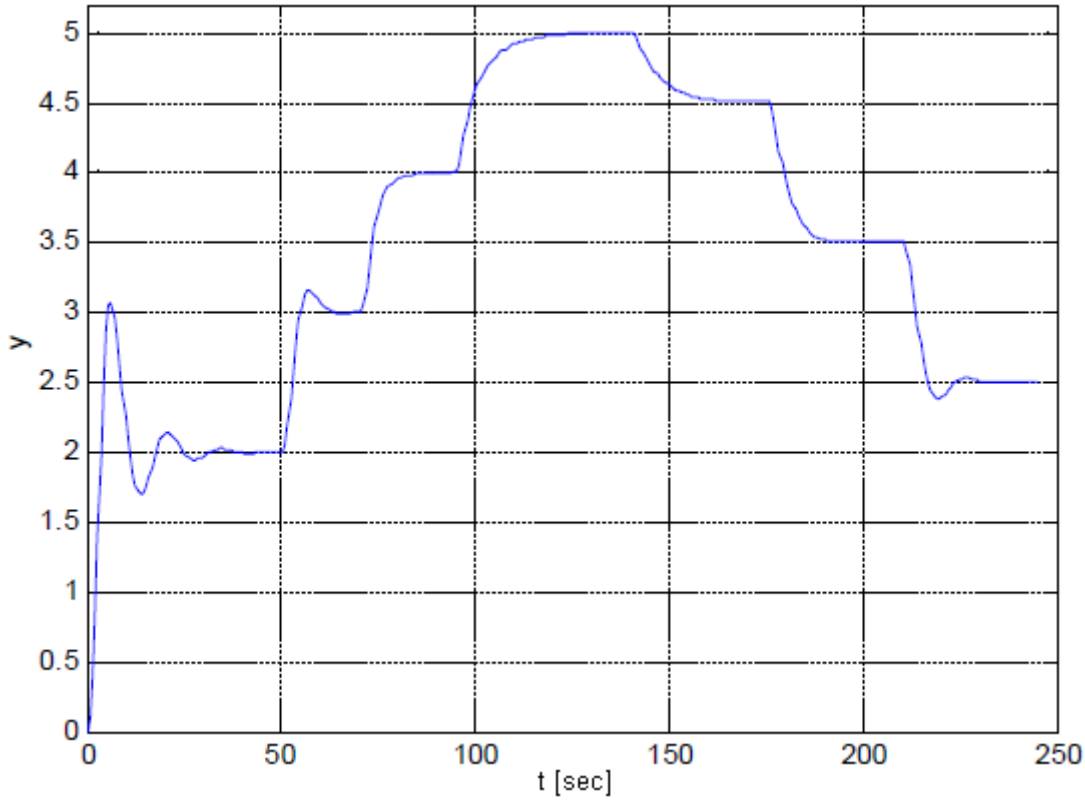

Figure 6: Time response of the output signal (training data)

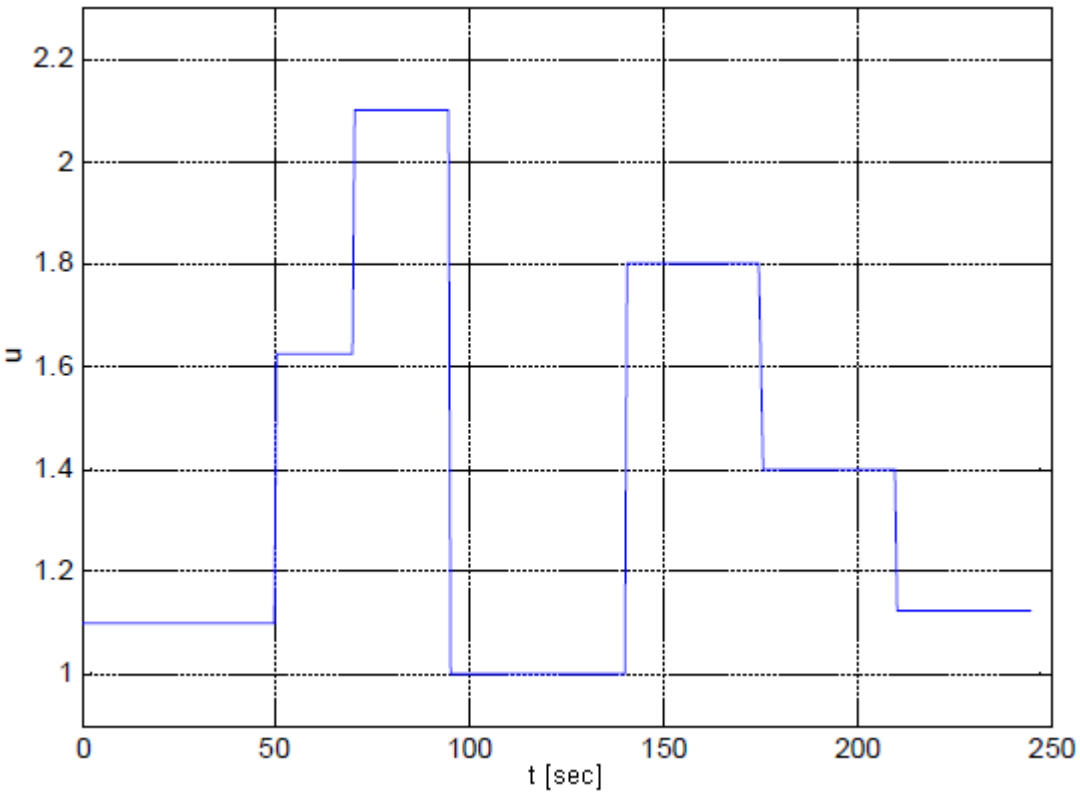

Figure 7: Time response of the input signal (testing data)

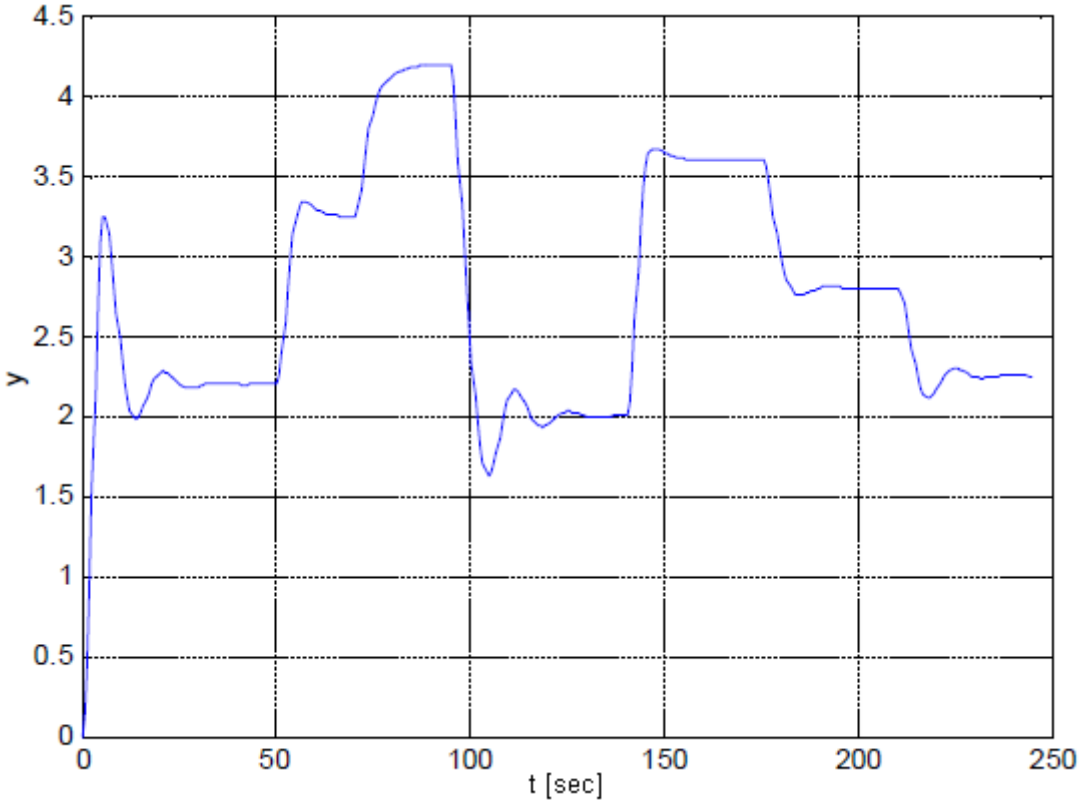

Figure 8: Time response of the output signal (testing data)

The detailed training (E_{trn} – training error) and testing (E_{tst} – testing error) results are shown in tables 1, 2, 3, 4, 5 and 6 for the NN1 structure with 4, 5 and 6 hidden neurons. The combination of activation

functions (Act. fn.) is listed in the first column (L_L_L_L means 4 *logsig* activation functions and T_L_L_L means 1 *tansig* and 3 *logsig* activation functions, etc.). Each table contains the final training or testing errors after 5000 epochs and the local error if it was smaller than the final error. The best results are marked as bold. The "-" indicates, that any modeling improvement was achieved during training algorithm.

Table 1: Modeling results for training data set - NN1 with 4 hidden neurons

Act. fn.	E_{trn}	epochs	E_{trn}	epochs	E_{trn}	epochs	E_{trn}	Epochs
L_L_L_L	2,0304	128	1,0248	1530	-	-	0,9762	5000
T_L_L_L	1,9881	41	1,3028	454	-	-	0,9437	5000
T_T_L_L	3,2508	26	1,0053	415	**0,8443**	1391	1,1680	5000
T_T_T_L	3,0997	29	1,0182	421	0,8577	1459	1,3664	5000
T_T_T_T	2,7930	33	1,0209	463	0,8743	1767	1,4845	5000

Table 2: Modeling results for testing data set - NN1 with 4 hidden neurons

Act. fn.	E_{tst}	epochs	E_{tst}	epochs	E_{tst}	epochs	E_{tst}	epochs
L_L_L_L	2,4027	128	1,1544	1530	-	-	1,0838	5000
T_L_L_L	2,2890	41	1,3250	454	-	-	1,0735	5000
T_T_L_L	3,7488	26	1,2241	415	**1,0278**	1391	1,3678	5000
T_T_T_L	3,5978	29	1,2406	421	1,0433	1459	1,5447	5000
T_T_T_T	3,3251	33	1,2090	463	1,0570	1767	1,6441	5000

Table 3: Modeling results for training data set - NN1 with 5 hidden neurons

Act. fn.	E_{trn}	epochs	E_{trn}	epochs	E_{trn}	epochs	E_{trn}	epochs
L_L_L_L_L	1,6373	215	1,0563	724	-	-	-	5000
T_L_L_L_L	-	-	-	-	-	-	0,8784	5000
T_T_L_L_L	0,8363	2886	-	-	-	-	0,9748	5000
T_T_T_L_L	1,6679	75	1,1254	242	**0,7918**	4982	**0,7918**	5000
T_T_T_T_L	2,2174	53	1,1095	400	0,8146	3607	0,9207	5000
T_T_T_T_T	1,8874	40	1,1450	258	0,8285	2661	0,9794	5000

Table 4: Modeling results for testing data set - NN1 with 5 hidden neurons

Act. fn.	E_{tst}	epochs	E_{tst}	epochs	E_{tst}	epochs	E_{tst}	epochs
L_L_L_L_L	1,9254	215	1,2122	724	-	-	1,0399	5000
T_L_L_L_L	-	-	-	-	-	-	1,0352	5000
T_T_L_L_L	1,0121	2886	-	-	-	-	1,1454	5000
T_T_T_L_L	1,9536	75	1,3254	242	**0,9929**	4982	**0,9933**	5000
T_T_T_T_L	2,5346	53	1,3498	400	1,0147	3607	1,1003	5000
T_T_T_T_T	2,2311	40	1,2408	258	1,0145	2661	1,1259	5000

Table 5: Modeling results for training data set - NN1 with 6 hidden neurons

Act. fn.	E_{trn}	epochs	E_{trn}	epochs	E_{trn}	epochs	E_{trn}	epochs
L_L_L_L_L_L	1,9627	175	1,0137	1190	-	-	1,1513	5000
T_L_L_L_L_L	1,1114	268	0,9056	1782	-	-	0,9131	5000
T_T_L_L_L_L	0,8475	1634	0,8549	3265	-	-	0,8934	5000
T_T_T_L_L_L	1,7557	75	1,0706	296	-	-	**0,7792**	5000
T_T_T_T_L_L	2,2659	56	1,0771	545	-	-	0,8637	5000
T_T_T_T_T_L	1,8485	43	1,1275	284	0,8236	2999	0,8965	5000
T_T_T_T_T_T	2,5750	30	0,8395	2266	-	-	1,0817	5000

Table 6: Modeling results for training data set - NN1 with 6 hidden neurons

Act. fn.	E_{tst}	epochs	E_{tst}	epochs	E_{tst}	epochs	E_{tst}	epochs
L_L_L_L_L_L	2.3087	175	1.1791	1190	-	-	1.1085	5000
T_L_L_L_L_L	1.3825	268	1.0558	1782	-	-	1.0137	5000
T_T_L_L_L_L	1.0266	1634	1.0066	3265	-	-	1.0272	5000
T_T_T_L_L_L	2.0559	75	1.2898	296	-	-	**0.9743**	5000
T_T_T_T_L_L	2.6417	56	1.2937	545	-	-	1.0133	5000
T_T_T_T_T_L	2.1799	43	1.2187	284	1.0061	2999	1.0587	5000
T_T_T_T_T_T	3.0626	30	1.0245	2266	-	-	1.1988	5000

Summarizing all simulation results for NN1, NN2 and NN3 with different number of hidden neurons can be declared, that the modified structures have smaller training and testing errors for modeling of nonlinear processes. Even one change of activation function in hidden layer with homogeneous neurons can achieved better results. Of course there are cases where the classical MLP has better modeling results because the training algorithm depends also on the initialization of weights.

4 Conclusion

The proposed paper presented a modified neural network structure based on non-homogeneous activation functions in hidden layer. In general the new approach has better results than the homogeneous distribution of transfer functions for MLP. Higher degree of modeling precision can result higher precision in process behavior prediction and control quality. The next improvement of the proposed method is the steady error evaluation during training algorithm to find the best ANN structure with generalization property.

ACKNOWLEDGMENT

This paper was supported by the Slovak Scientific Grant Agency VEGA, Grant no. 1/2256/12.

References

[1] BEŇUŠKOVÁ, Ľ. *Umelé neurónové siete*. [online], 2001, Available on the Internet: http://ii.fmph.uniba.sk/~benus/books/UNS_revised.pdf

[2] SINČÁK P., ANDREJKOVÁ G. *Neurónové siete inžiniersky prístup*. Available on the Internet:

http://www.ai-cit.sk/source/publications/books/NS1/html/index.html

http://www.ai-cit.sk/source/publications/books/NS2/html/index.html

[3] TUCKOVÁ, J. *Úvod do teorie a aplikací umelých neuronových sítí*. Praha: ČVUT, 2003. 103p. ISBN 80-01-02800-3.

[4] KVASNIČKA, V. - BEŇUŠKOVÁ, Ľ. - POSPÍCHAL, J. - FARKAŠ, I. – TIŇO, P. - KRÁĽ , A. *Úvod do teórie neurónových sietí* [online], 2010. Available on the Internet:

http://ics.upjs.sk/~novotnyr/home/skola/neuronove_siete/nn_kvasnicka/Uvod%20do%20NS.pdf

[5] VOLNÁ, E. *Neurónové sítě 1* [online].Ostrava, 2008, Available on the Internet:

http://www1.osu.cz/~volna/Neuronove_site_1_informace.pdf

[6] HAGAN, M. - MENHAJ, M. *Training Feedforward Networks with the Marquardt Algorithm*. In Ieee Transaction on Neural Networks. vol. 5, no. 6, p. 989-990

[7] MathWorks: *Neural Network Toolbox*, Available on the Internet:

http://www.mathworks.com/

Ladislav Körösi, Jana Paulusová

Institute of Robotics and Cybernetics,
Faculty of Electrical Engineering and Information Technology,
Slovak University of Technology,
Ilkovičova 3,
812 19 Bratislava, Slovak Republic
e-mail: ladislav.korosi@stuba.sk, jana.paulusova@stuba.sk