A NEW STRUCTURE FOR NONLINEAR SYSTEM IDENTIFICATION USING NEURAL NETWORKS

Hassan CHARAF and István VAJK

Department of Automation
Faculty of Electrical Engineering
Technical University of Budapest
H–1521 Budapest, Hungary
Fax:(36-1) 463 2871
Tel: (36-1) 463 2870
E-mail: Hassan@bme-at.aut.bme.hu, Vajk@bme-at.aut.bme.hu

Received: Sept. 17, 1998

Abstract

Most industrial systems are nonlinear. In these applications the conventional identification and control techniques are effectively used if the nonlinearity of the system is known. When the system contains unknown nonlinearities, however, the conventional techniques exhibit poor performance. To tackle this problem a neural network is proposed to use. The ability of neural networks to approximate nonlinear relationships makes them prime candidate for applications in nonlinear system identification. Simulation results show that if the conventional nonlinear system description is used the modelling error may be significant, but using the delta transformation this error can be reduced. This paper demonstrates the difference between the shift and delta model and verifies the effectiveness of the structure of the delta transformation. Simulation results demonstrate this difference.

Keywords: neural networks, delta transformation, delta operator, shift operator, identification.

1. Introduction

In the past three decades many identification and control design techniques have been established. These techniques are efficient for linear systems and for those nonlinear systems in which the nonlinearity is known. If the nonlinearity is unknown, however, then the task is very difficult. The ability of neural networks to learn any nonlinear mapping between input and output data makes them useful and efficient tools to solve this problem.

The Neural Networks (NN) are parametrised nonlinear functions. The parameters in the NN are its weights. Learning simply means parameter estimation. *It is well known that the fundamental properties of Neural Networks make them useful as approximators of nonlinear mapping.*

The Kolmogorov theorem gave insight into the capabilities of multilayered neural nets. As explained by LIPPMANN (LIPPMANN, 1987) this theorem states that any continuous function of N variables can be computed using only linear summations and nonlinear but continuously increasing functions of only one variable. Actually the theorem states that a three layered net with $N(2N+1)$ neurons using continuously increasing nonlinearities can approximate any continuous function of N variables. Unfortunately, the theorem does not indicate how the weights and the nonlinearities in the net should be selected.

2. System Identification

System modelling (i.e. its mathematical representation) and identification are *fundamental problems* in system theory, where it is often required to approximate the behaviour of a real system with an appropriate mathematical model given by a set of input-output data. The identification problem is to find relationships between past input-output data and future outputs. To identify nonlinear systems it is necessary to define nonlinear models, whose parameters have to be estimated to represent the system. One condition to obtain good identification results is that the input of the plant should be adequately 'rich' in order to capture the system dynamics accurately. For example, to identify the steady state gain an input signal of small frequency is required. On the other hand, the identification of the time constants requires another frequency region in the input signal. If the identification is accomplished only in a subspace of the possible inputs, the results produced by the network could be poor outside this subspace. The importance of the inputs used to train learning systems is widely appreciated. In the Neural Network literature the input and output data are called training data or training patterns. The main task of the identification is to determine the parameters of the assumed model (*Fig. 1*).

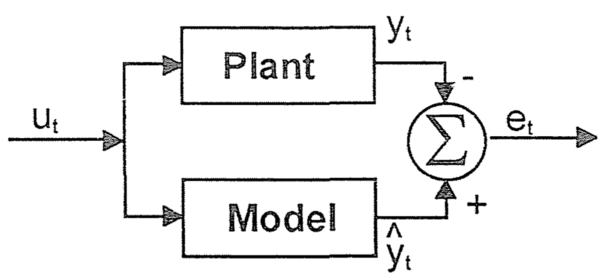

Fig. 1. The structure of identification

For linear time-invariant systems model structure selection and identification problem is well established and the literature abounds with many useful methods, algorithms and application studies (NARENDRA, 1990).

Eq. (1) represents a (SISO) linear time-invariant causal system structure:

$$y_{t+1} = \sum_{i=0}^{n-1} a_i y_{t-i} + \sum_{j=0}^{m-1} b_j u_{t-j} , \tag{1}$$

where a_i, b_j are the unknown parameters. Two identification models are often used (NARENDRA, 1990).

1. *Parallel model* (*Fig. 2*) for linear or linearized nonlinear systems:

$$\hat{y}_{t+1} = \sum_{i=0}^{n-1} \hat{a}_i \hat{y}_{t-i} + \sum_{j=0}^{m-1} \hat{b}_j u_{t-j} . \tag{2}$$

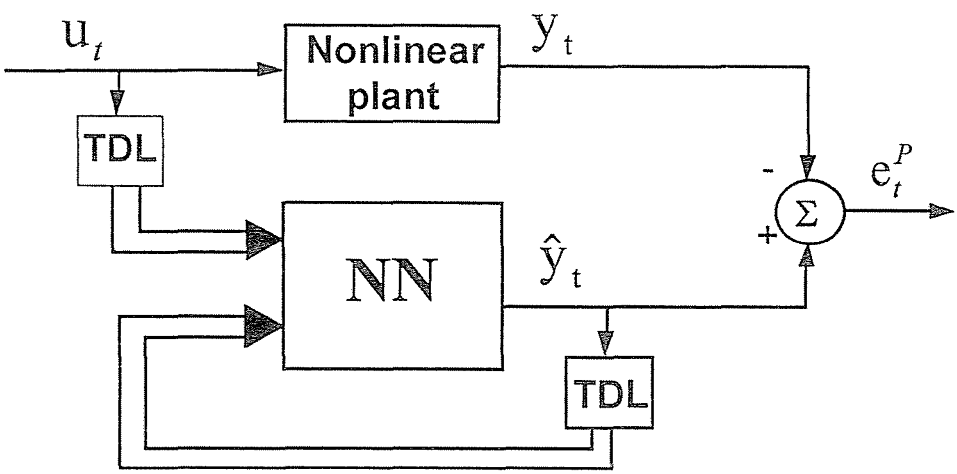

Fig. 2. Parallel identification model

Here the feedback is taken from the output of the estimated model.

2. *Series-parallel* model (*Fig. 3*) for linear or linearized nonlinear systems:

$$\hat{y}_{t+1} = \sum_{i=0}^{n-1} \hat{a}_i y_{t-i} + \sum_{j=0}^{m-1} \hat{b}_j u_{t-j} . \tag{3}$$

Here the feedback is taken from the plant output. The TDL in the figures denotes a tapped delay line (*Fig. 4*) where q^{-1} is the backward shift operator.

As the parallel model under training may cause a divergent result – during the training phase the series-parallel model, after the training the parallel model is used. Notice that in the series-parallel identification model

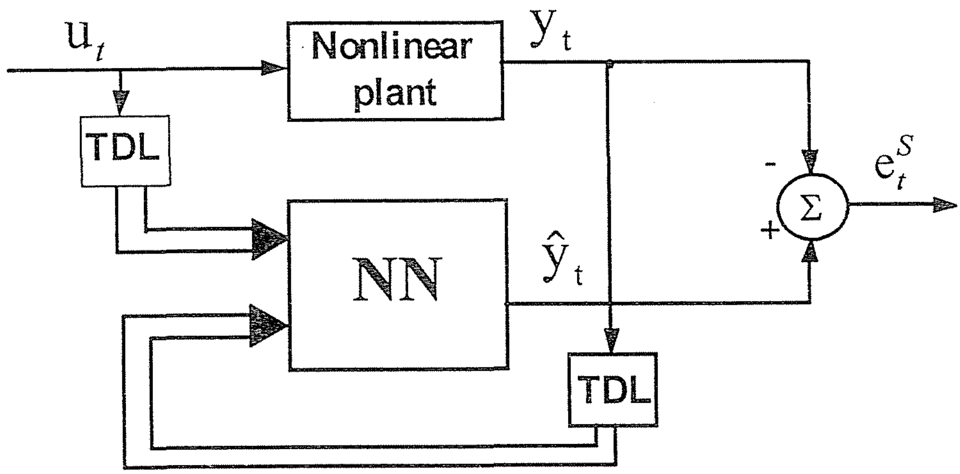

Fig. 3. Series-parallel identification model

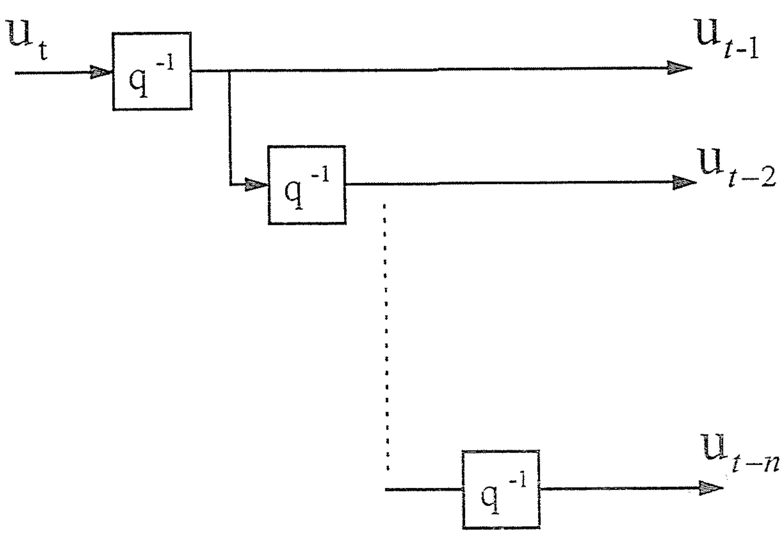

Fig. 4. The tapped delay line

a feedforward, while in the parallel identification model a recurrent neural network is used. The objective of the identification is to determine the a_i and b_i parameters that guarantee minimal error between the real output and its estimated value.

$$||\hat{y} - y|| < \varepsilon \, . \tag{4}$$

Most systems encountered in industry are nonlinear. To model nonlinear systems, nonlinear system descriptions have to be applied. Nonlinear Auto

Regressive Moving Average (NARMA) description has been shown to provide a very useful unified representation for a wide class of nonlinear systems:

$$y_{t+1} = f[y_t, \ldots, y_{t-n+1}, u_t, \ldots, u_{t-m+1}] \,. \tag{5}$$

Here $f[.]$ is a nonlinear function which is rarely known a priori and can be very complicated. Nevertheless, nonlinear system identification is a complex and difficult task.

The problem of identifying a model structure and its associated parameters can be related to the problem of learning a mapping between a known input and output space. In this paper a neural network is used to solve this problem.

An immediate problem is how large the variables n and m should be in *Eq.* (5). From a practical viewpoint they should be as small as possible to reduce the complexity of the network, and the number of the parameters. In this case the task of determining the parameters of the network is easier and the learning procedure is shorter. On the other hand, these variables should be large enough to model the significant dynamics of the nonlinear plant (WARWICK, 1992).

3. Neural Networks

The literature is *rich* in definitions of neural networks. A neural network is a set of *simple* elements (neurons) which are connected together organised into layers to form either one single layer or multiple ones. Each neuron has multiple inputs and one output. In each input there is a weight by which the input signal is multiplied. Each neuron has a self weight (bias). Each neuron sums all its weighted inputs and performs a nonlinear function operation on it. This nonlinear function is called the *activation function* of the neuron.

3.1. Neural Network Architecture

A typical multiple-layer feedforward neural network consists of an input, an output and one or more hidden layers. If a network contains some delayed outputs or delayed internal states as inputs then that network is a *recurrent* or *dynamic* network which has useful properties in dynamic system identification and control.

Key questions are: how many layers of hidden units should be used, and how many units are required in each layer? *What is the smallest possible number of neurons in a hidden layer for best possible operation?*

3.2. Neural Networks as Universal Continuous Mapping Approximators

Using Neural Networks is just a way of curve fitting to data. They have excellent approximation properties. Due to the Kolmogorov's theorem any continuous function f of several variables defined in space ($I^n = [0,1]^n$; $n \geq 2$) can be represented in the form :

$$f(x) = \sum_{j=1}^{2n+1} \chi_j \left(\sum_{i=1}^{n} \psi_{ij}(x_i) \right) ,$$

where χ_i, ψ_{ij} are continuous functions of one variable and ψ_{ij} are monotone functions which are not dependent on f.

A lot of theorems demonstrate the ability of a neural network to approximate nonlinear functions (HECHT–NIELSEN (1987), GALLANT and WHITE (1988), CYBENKO (1989), HORNIK et al (1989), FUNAHASHI (1989)). In (FUNAHASHI, 1989) the following theorem was proved:

Let $\varphi(x)$ be a nonconstant, bounded and monotone increasing continuous function. Let K be a compact subset (bounded closed subset) of R^n and $f(x_1, x_2, \ldots, x_n)$ be a real value continuous function on K. Then for an arbitrary $\varepsilon > 0$, there exists an integer N and real constants c_i, $\theta_i (i = 1, \ldots, N)$, $w_{ij} (i = 1, \ldots, N, j = 1, \ldots, n)$ so that:

$$\hat{f}(x_1, \ldots, x_n) = \sum_{i=1}^{N} c_i \varphi \left(\sum_{j=1}^{n} (w_{ij} x_j - \theta_i) \right) \tag{6}$$

satisfies :

$$\max_{x \in K} \left| f(x_1, \ldots, x_n) - \hat{f}(x_1, \ldots, x_n) \right| < \varepsilon . \tag{7}$$

In other words, for an arbitrary $\varepsilon > 0$, there exists a three-layer network (one-hidden layer) whose transfer functions for the hidden layer are $\varphi(x)$ and for input and output layers are linear and which has an input-output function $\hat{f}(x_1, \ldots, x_n)$ which satisfies *Eq.* (7). The theorem does not say that a single hidden layer is optimum in the sense of learning time or ease of implementation.

In general, one-hidden layer neural network with a nonlinear monotone increasing (e.g. sigmoidal) nonlinear hidden neuron transfer function can approximate any continuous function with an arbitrary accuracy. The transfer function is usually sigmoid, tangent hyperbolic or saturation.

3.3. Learning Algorithms

Learning for Neural Networks simply means parameter (weights) estimation. But the model is nonlinear in the parameters. In each neural network application learning is a *critical* question. The objective is to determine an adaptive algorithm or rule which adjusts the parameters (weights) of the network based on a given set of input-output pairs. The collected data are used as training data for the learning process of the neural network.

The problem of determining the network weights can be considered essentially as a nonlinear optimisation task. The simplest optimisation technique uses the objective function (cost function) to determine the search direction. It is well-known that *gradient search for the minimum is inefficient*, especially close to the minimum. It is better to use another search technique. The *quasi-Newton* (variable metric) and the *conjugate gradient* search techniques are very efficient solving this task.

The main feature of the quasi-Newton method is that it makes a sequence of progressive estimates of the inverse Hessian (second derivative of the cost function) matrix, based only on the first derivatives. The approximated matrix is updated in each iteration step, supposing that the function can be calculated at all points and the gradients can be determined analytically at each point or can be estimated from the differences of values of the function to be minimised.

The conjugate gradient algorithm generates a conjugate direction as a linear combination of the current gradient and the previous search direction. The current parameter vector is a linear combination of the previous parameter vector and the current conjugate direction (CHARAF et al, 1995).

3.4. The Problem of the Identification Based on the Shift Operator

Let us assume that a nonlinear system is defined by *Eq.* (5). A neural network is used to identify the nonlinear behaviour of the system. Under training the series-parallel model is used (*Fig. 3*). The neural network approximates function f (*Eq.* (5)) by $\hat{f}$. The training task is to minimise the square error between the real system output and the output of the network. The remained identification error is defined as follows:

$$e_{t+1}^{S} = \hat{f}\left[y_t, \ldots, y_{t-n+1}, u_t, \ldots, u_{t-m+1}\right] - f\left[y_t, \ldots, y_{t-n+1}, u_t, \ldots, u_{t-m+1}\right]. \tag{8}$$

After the training the parallel model is used. The output of the network (which contains e^S error) is fed back. The error of the network can be significant. This error (*Fig. 2*) is defined as follows:

$$e_{t+1}^{P} = \hat{f}\left[\hat{y}_t, \ldots, \hat{y}_{t-n+1}, u_t, \ldots, u_{t-m+1}\right] - f\left[y_t, \ldots, y_{t-n+1}, u_t, \ldots, u_{t-m+1}\right], \tag{9}$$

where

$$\hat{y}_{t+1} = \hat{f}\left[\hat{y}_t, \ldots, \hat{y}_{t-n+1}, u_t, \ldots, u_{t-m+1}\right] \tag{10}$$

supposing that the function f is continuous and differentiable around the working point. f can expanded to Taylor series. The network error is calculated as follows:

$$e^P_{t+1} \approx e^S_{t+1} + \left.\frac{\partial f}{\partial y_t}\right|_Y (\hat{y}_t - y_t) +$$

$$+ \left.\frac{\partial f}{\partial y_{t-1}}\right|_Y (\hat{y}_{t-1} - y_{t-1}) + \ldots + \left.\frac{\partial f}{\partial y_{t-n+1}}\right|_Y (\hat{y}_{t-n+1} - y_{t-n+1}) , \qquad (11)$$

where

$$Y = [y_t, \ldots, y_{t-n+1}, u_t, \ldots, u_{t-m+1}] .$$

The *Eq.* (11) can be written in other form:

$$e^P_{t+1} = e^S_{t+1} - \sum_{i=1}^{n} a_i \cdot e^P_{t-i+1} , \qquad (12)$$

where

$$a_i = - \left.\frac{\partial f}{\partial y_{t-i+1}}\right|_Y . \qquad (13)$$

Since in the steady state the following equalities are available:

$$e^S_t = e^S_{t-1} = \ldots = e^S \qquad (14)$$

and

$$e^P_t = e^P_{t-1} = \ldots = e^P , \qquad (15)$$

it can be shown that the network error in this case will be:

$$e^P = \frac{e^S}{1 + \sum_{i=1}^{n} a_i} . \qquad (16)$$

For example in the case of a first order system

$$e^P = \frac{e^S}{1 + a} . \qquad (17)$$

This means that if $a = -0.99$, then the error of the network in a given working point is 100 times bigger than the identification error remained after training (Example 1 in section 4.) In the case of a second order system:

$$y_t - 1.8561\, y_{t-1} + 0.8607\, y_{t-2} = 0.0024\, u_{t-1} + 0.0023\, u_{t-2} . \qquad (18)$$

According to *Eq.* (16) the network error is as follows:

$$e^P = \frac{e^S}{1 + a_1 + a_2} = \frac{e^S}{1 - 1.8561 + 0.8607} \approx 218 \cdot e^S . \qquad (19)$$

These examples demonstrate that the shift operator form has a lot of disadvantages. In case of higher order systems this error grows very fast. Numerical examples verify that in some cases this form is not useful. It is necessary to find another structure which guarantees the smaller error. The proposed structure uses the delta transformation.

3.5. Delta Transformation

The shift operator q is often used to describe discrete systems. The definition of the forward shift operator is

$$qx_t \equiv x_{t+1} \,. \tag{20}$$

Using this operator the discrete state space model of a system can be written as

$$\begin{aligned} qx_t &= F(x_t, u_t) \,, \\ y_t &= G(x_t, u_t) \,, \end{aligned} \tag{21}$$

where x_t, u_t, y_t are the state, the input and the output of the process, respectively.

Another equivalent description of the system can be obtained by using the delta operator. This operation is defined as

$$\delta x_t \equiv \frac{x_{t+1} - x_t}{h} = \frac{x(t \cdot h + h) - x(t \cdot h)}{h} \,, \tag{22}$$

where h is the sampling time (MIDDLETON et al, 1987). The relationship between the q and δ operators is a simple linear function:

$$\delta = \frac{q-1}{h} \,. \tag{23}$$

This guarantees the same flexibility in the modelling of dynamic systems as does the shift operator. Using the δ operator the discrete state model can be described by

$$\begin{aligned} \delta x_t &= F'(x_t, u_t) \,, \\ y_t &= G'(x_t, u_t) \,. \end{aligned} \tag{24}$$

One way to determine the discrete delta model form is to find a shift model form and then to substitute

$$q = 1 + h\delta \,. \tag{25}$$

Though this transformation method is technically correct, this is not the best way to derive the delta model. A better method is based on the selection of the state variables which are used in the continuous time state space equation.

To demonstrate the transformation we present the discrete description of a second order linear system. Consider the following continuous input-output model:

$$2\frac{d^2}{dt^2}y(t) + 3\frac{d}{dt}y(t) + y(t) = u(t) \,. \tag{26}$$

If we discretize this system assuming a zero order holding at the input and a sampling period of 0.1 sec, the following input-output model is obtained in the shift form:

$$(q^2 - 1.856\,q + 0.8607)y_t = (0.002379\,q + 0.002263)u_t\,. \tag{27}$$

The equivalent form of the above system in delta form is

$$(2.155\,\delta^2 + 3.101\,\delta + 1)y_t = (0.05125\,\delta + 1)u_t$$

or by eliminating the operator the obtained model is:

$$2.155\frac{y_t - 2y_{t-1} + y_{t-2}}{h^2} + 3.101\frac{y_{t-1} - y_{t-2}}{h} + y_{t-2} =$$

$$= 0.05125\frac{u_{t-1} - u_{t-2}}{h} + u_{t-2}\,. \tag{28}$$

We can see that the coefficients in the delta model show a close similarity to the corresponding coefficients of the continuous model. Another advantage of the delta model is that the numerical properties of the delta models are superior to those of shift operator model in practice. This fact will be presented in the next section. Here the nonlinear behaviour of the system is approximated by a neural network and the dynamics of the system is taken into consideration by a network containing only discrete integrators. This realisation is the special case of *Eq.* (24). *Fig. 5* shows the structure which is used for modelling the nonlinear system.

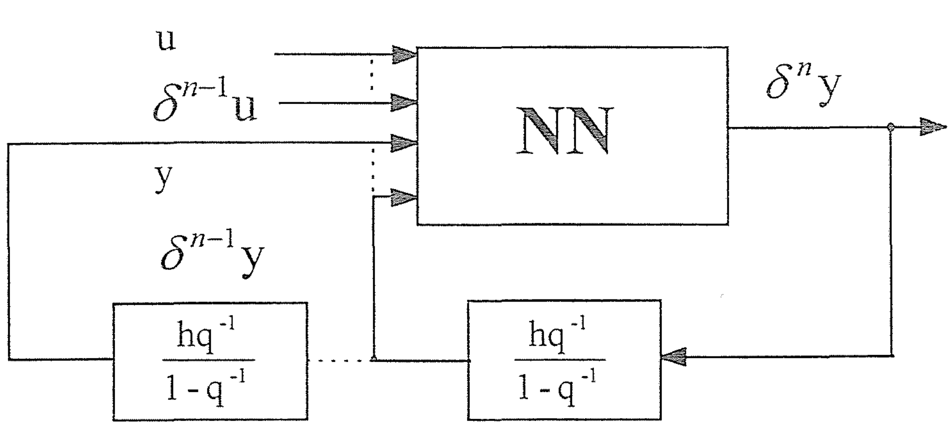

Fig. 5. The structure of delta operator

4. Numerical Results

To verify the feasibility of the proposed structure a number of examples has been studied by simulations. In this section the results of two simple examples are presented. In the first example a first order system, in the second one a second order system with deadzone nonlinearity is identified. The examples are to demonstrate the difference between the two structures.

EXAMPLE 1 A first order nonlinear model is given in *Fig. 6.*

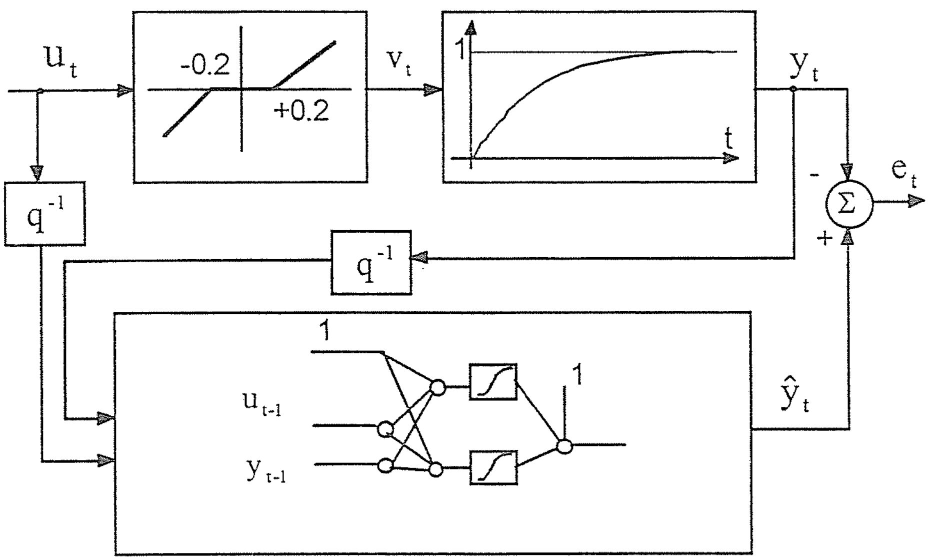

Fig. 6. The identification process

The model is described by the following equation:

$$y_t - 0.99\, y_{t-1} = 0.01\, v_{t-1}$$

with

$$v_t = \left\{ \begin{array}{ll} 0 & \text{if} \quad |u_t| < 0.2 \\ u_t - 0.2 \cdot \text{sign}\,(u_t) & \text{otherwise} \end{array} \right\} .$$

The nonlinearity represents a dead zone. The training of the model for this plant has been carried out using one hidden layer including two neurons in it. The transfer function of the hidden neurons is tangent hyperbolic. The transfer function of the output neuron is linear. The system equations above show that y_t will be a function of y_{t-1} and u_{t-1}. Using inputs in the $[-1; 1]$ interval at $-1, -0.8, \ldots, 1$ and assuming the same set for y_{t-1} a pattern of 121 values results in a surface of the y_t due to the system equations. The

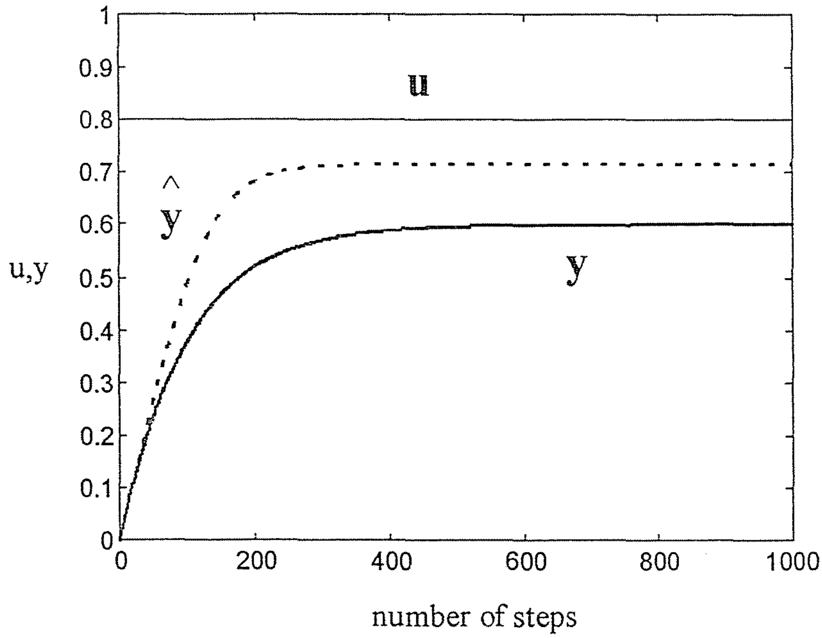

Fig. 7. The step response of the trained neural network using the shift operator

used input signal guarantees the persistent excitation. The training method is the quasi-Newton algorithm.

The simulation results of identification based on shift operator are shown in *Fig. 7.* The difference between the real and the estimated output is approximately $e^P = 0.12$. The identification error in this working point ($u = 0.8$, $y = 0.6$) is $e^S = 0.0012$. According to *Eq.* (17) the network error numerically is 100 times bigger than the identification error.

To illustrate the advantages of the delta transform model, now the same network with the same initial weights is trained using the delta operator model. This system is equivalent with a discretization of a continuous time system where the steady state gain is $K = 1$, the time constant is $T = 1$ sec, and the sampling time is $h = 0.01$ sec. A normalisation procedure is performed on the output values to scale the output interval to the $[-1, 1]$ interval.

The simulation results of identification based on the delta operator are shown in *Figs. 8, 9.* In *Fig. 8* the square impulse responses, in *Fig. 9* the sinusoidal responses of the trained network and the real system are shown. The used neural network learnt the given plant with an excellent accuracy. As a matter of fact the real system output and the output of the neural network almost completely cover each other. The parallel model is used to test the validity of the training. The dotted line is the output of the network and the solid line is the desired signal. *Figs. 10, 11* represent the

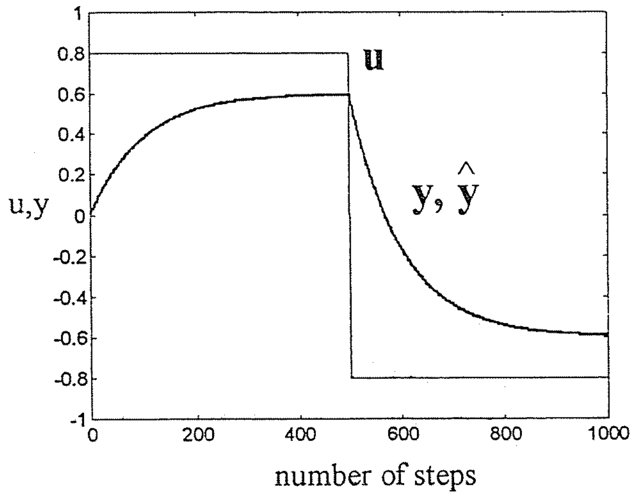

Fig. 8. Square pulse response using the delta operator

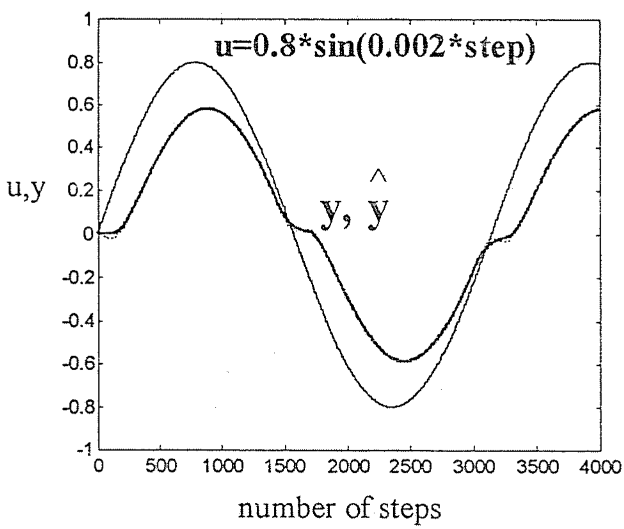

Fig. 9. Sinusoidal response using the delta operator

final output of the network and of each hidden neuron alone in 3-D.

EXAMPLE 2 Consider a second order nonlinear model described by the following:

$$y_t - 1.9825\, y_{t-1} + 0.9841\, y_{t-2} = 0.0007956\, v_{t-1} + 0.0007914\, v_{t-2}$$

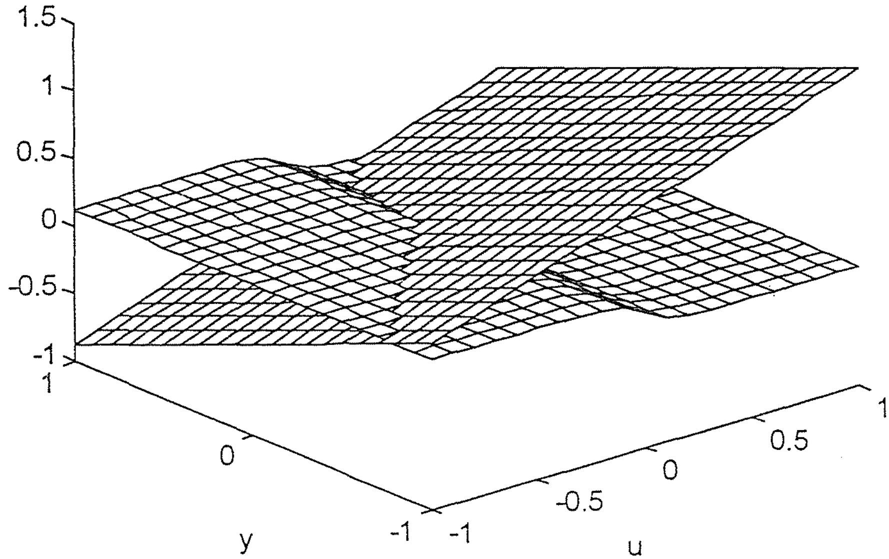

Fig. 10. The outputs of the hidden neurons.

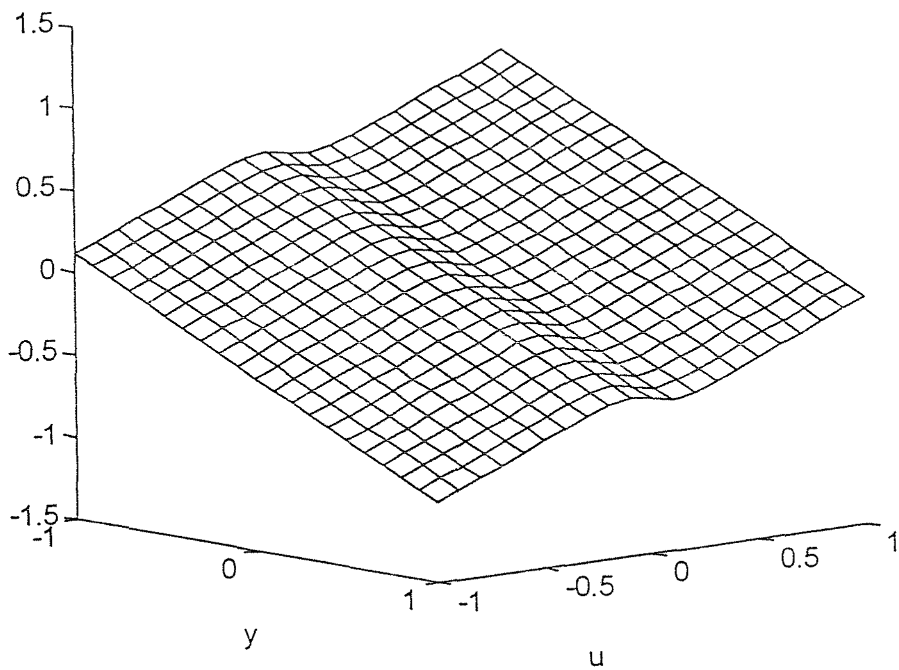

Fig. 11. The output of the network.

with

$$v_t = \begin{cases} 0 & \text{if } |u_t| < 0.2 \\ u_t - 0.2 \cdot \text{sign}(u_t), & \text{otherwise} \end{cases} .$$

The nonlinearity represents a dead zone. This system is equivalent with a discretized second order continuous time system, where the two poles are $s_1 = -0.04 + 0.196j$ 1/sec and $s_2 = -0.04 - 0.196j$ 1/sec, respectively, the steady state gain is $K = 1$ and the sampling time is $h = 0.2$ sec. The training of the model for this plant has been carried out using one hidden layer including six neurons in it. The activation function of the hidden neurons is tangent hyperbolic, while the activation function of the output neuron is linear. 1000 patterns have been used for training randomly between -1 and 1. The training method is the quasi-Newton algorithm. Using the delta transform as shown in section 3.5 better results are obtained.

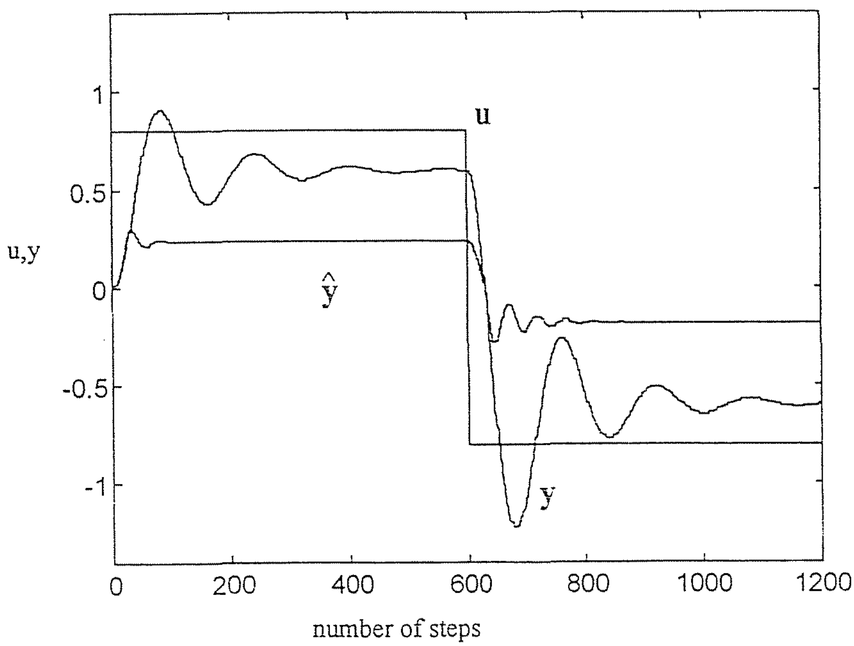

Fig. 12. Square impulse response using the shift operator

The simulation results of identification based on the shift operator are shown in *Fig. 12*. In *Fig. 12* the real system output, the estimated output and the input are shown. The difference between the real and estimated output is approximately 0.42. It is correct since the identification error in the used working point is 0.002 and according to *Eq.* (19) the network error is 218 times the identification error. The results of the identification based on the shift operator in this case are very bad.

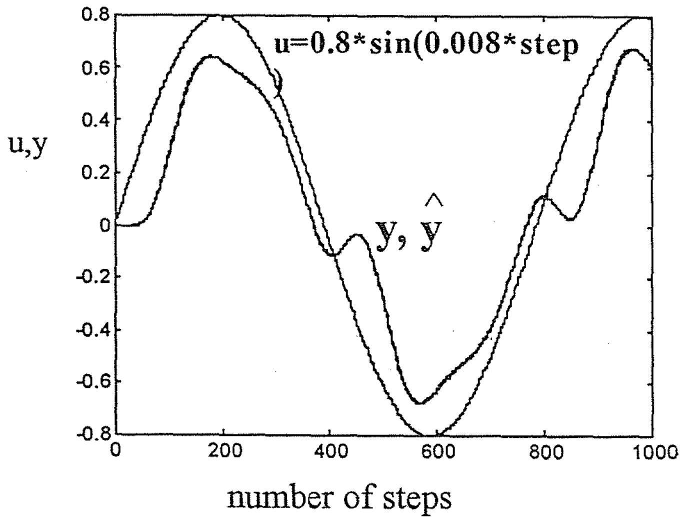

Fig. 13. The sinusoidal response using the delta operator

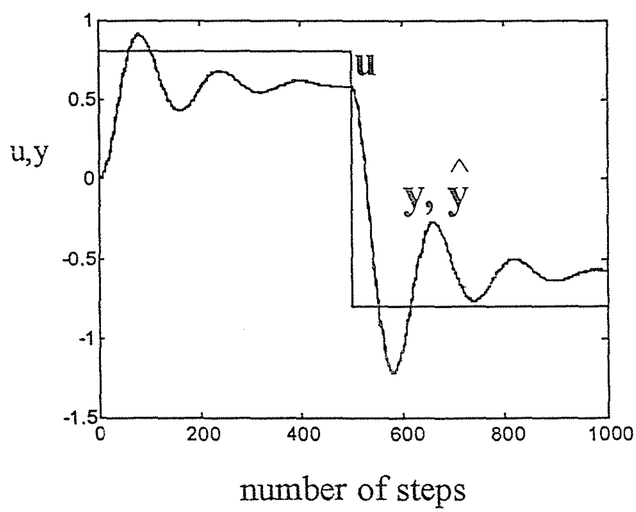

Fig. 14. The square impulse response using the delta operator

The simulation results of identification based on the delta operator are shown in *Figs. 13* and *14*. In the figures the real system output, the estimated output and the input signals are shown. The real system output and the output of the neural network almost completely cover each other. The results of identification based on delta operator are very good. In this case the used neural network learnt the nonlinear plant with an excellent accuracy. The results of the two examples verify the effectiveness of the proposed structure.

5. Conclusion

In this paper a new structure is proposed and described to solve nonlinear identification problems. Simulation studies are presented to demonstrate the effectiveness and feasibility of the approach. As it has been shown by examples using the delta transformation in an identification task makes the neural networks more effective tools. The shift operator form is not useful to some extent. Assuming the same environment (initial value, network size, etc.) the delta transformation model gives superior results. The examples shown above demonstrate the effectiveness of this thesis. The delta transformation structure produces the same results for another type of nonlinearity, as well.

The problem to design a controller which generates the *desired* control input is based on a good model of the process to be controlled. To have a good control behaviour it is necessary to have a model of good accuracy. The more accurate an identified model is, the better control can be achieved.

Acknowledgements

A part of the research was supported by the Hungarian National Research Fund (grant number T015776).

References

[1] ALEKSANDER, I. – MORTON, H. (1990): An Introduction to Neural Computing, Chapman and Hall.

[2] ATHERTON, D. P. (1975): Nonlinear Control Engineering Describing Function Analysis and Design. Van Nostrand Reinhold.

[3] CHARAF, H. – VAJK, I. (1995): Neural Networks Can be Trained Faster, *IFAC ACASP'95*, Budapest (to be published).

[4] CSÁKI, F. (1972): Modern Control Theories. Nonlinear Optimal and Adaptive Systems (Transl. by P. Szőke) Budapest, Akadémiai Kiadó (in Hungarian).

[5] CYBENKO, G. (1989): Approximation by Superpositions of a Sigmoidal Function. *Mathematics of Control, Signals and Systems*, Vol. 2, pp. 303–314.

[6] DEMUTH, H. – BEALE, M. (1993): Neural Network TOOLBOX for Use with MATLAB.

[7] FUNAHASHI, KEN-ICHI (1992): On the Approximate Realization of Continuous Mappings by Neural Networks. Artificial Neural Networks Concepts and Control Applications. pp. 375–384.

[8] FLETCHER, R. – POWELL, M. J. D. (1963): A Rapidly Convergent Descent Method for Minimization, *Computer Journal*, Vol. 6, p. 163.

[9] FLETCHER, R. (1970): A New Approach to Variable Metric Algorithms, *Computer Journal*, Vol. 13, p. 317.

[10] GALLANT, C. C. – WHITE, H. (1988): There Exists a Neural Network That Does Not Make Avoidable Mistakes. *IEEE International Conference on Neural Networks*, Vol. 1, pp. 657–664, San Diego, CA.

[11] HAYKIN, S. (1994): Neural Networks. A Comprehensive Foundation. Macmillan College Publishing Company.
[12] HECHT-NIELSEN, R. (1987): Kolmogorov's Mapping Neural Network Existance Theorem. *1-st IEEE International Confererence on Neural Networks*, Vol. 3, pp. 11–14, San Diego, CA.
[13] HORNIK, K. – STINCHCOMBE, M. – WHITE, H. (1989): Multilayer Feedforward Networks Are Universal Approximators. *Neural Networks*, Vol. 2, pp. 359–366.
[14] KOIVO, H. N. (1994): Artificial Neural Networks in Fault Diagnosis and Control, *Control Eng. Practice*, Vol. 2, No. 1, pp. 89–101.
[15] LIPPMANN, R. (1987): An Introduction to Computing with Neural Nets. *IEEE ASSP Magazine*, Vol. 4, pp. 4–22.
[16] MIDDLETON, R. H. – GOODWIN, G. C. (1987): Improved Finite Word Length Characteristics in Digital Control Using Delta Operators. *IEEE Trans. Auto. Control*, Vol. AC-31, No. 11, pp. 1015–1021.
[17] NARENDRA, K. S. – PARTHASARATHY, K. (1990): Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Transactions on Neural Networks*, Vol. 1, No. 1. pp. 4–27.
[18] SJÖBERG, J. – HJALMARSSON, L. – LJUNG, L. (1994): Neural Networks in System Identification. *SYSID' 94*, Copenhagen, Denmark, pp. 49–72.
[19] WARWICK, K. – IRWIN, G. W. – HUNT, K. J. (1992): Neural Networks for Control and Systems. *IEE Control Engineering Series* 46.