Journal of ELECTRICAL ENGINEERING, VOL. 54, NO. 7-8, 2003, 213–217

STATE ESTIMATION AND CONTROL OF NONLINEAR PROCESS USING NEURAL NETWORKS

Anna Jadlovská *

This paper considers the use of neural networks for non-linear state estimation, identification and control of non-linear processes. The non-linear identification is using feed-forward neural networks as a useful mathematical tool to build a model between the input and the output of a non-linear process. In this paper we consider the possibility of on-line state estimation of the actual parameters from an off-line trained neural state space model of the non-linear process using the gain matrix. This linearization technique is used in the algorithm of on-line tuning of the controller parameters based on pole placement control design for non-linear process.

K e y w o r d s: dynamic neural models, non-linear state estimation, gain matrix, non-linear control

1 INTRODUCTION

The purpose of this paper is to show how feed-forward neural network (Multi Layer Perceptron - MLP) can be used for modeling and control of the non-linear process. When the mathematical model of the process cannot be derived with an analytical method, then the only way is using the relationship between the input and output of the process. Fitting the model from the data is known as an identification of the process. For linear processes this technique is generally well known, [7].

For processes, which are complex or difficult to model the non-linear identification can use a feed-forward neural network - MLP as a useful mathematical tool to build the non-parametric model between the input and the output of the real non-linear process [2], [3] and [9]. We will consider in this paper the possibility of on-line state estimation of the actual parameters from an off-line trained neural model of the non- linear process using the gain matrix, introduced later, [5], [8].

This linearization technique is used to perform on-line tuning of the controller parameters based on pole placement control design in an observer-based control loop. The advantage using sample-by-sample linearization is that the controller parameters can be changed in response to process changes, [4]. In fact neural network method also becomes adaptive if training of the neural network state space model as an observer is continued on-line.

2 MODELING OF NONLINEAR PROCESS

In this part we will discuss some basic aspects of non-linear system identification using from among numerous neural networks structures only Multi-Layer Perceptron - MLP (Feed- Forward Neural Network) with respect to model based neural control, where the control law is based upon the neural model.

We know, that a discrete-time non-linear system can be written in the following general form

$$\begin{aligned} \mathbf{x}(k) &= \boldsymbol{f}\left(\mathbf{x}(k-1), \boldsymbol{u}(k-1), \boldsymbol{d}(k-1)\right) \\ \boldsymbol{y}(k) &= \boldsymbol{h}\mathbf{x}(k) \end{aligned} \tag{1}$$

in which k is the discrete sample number, $\boldsymbol{x}$ is a state vector, $\boldsymbol{u}$ is the input vector and $\boldsymbol{y}$ is the output, $\boldsymbol{d}$ is a vector of disturbances. We will furthermore include a vector Θ containing some set of parameters sufficient to describe the model at time k in the description. They are generally assumed to vary much slower than the states $\boldsymbol{x}$. Further $\boldsymbol{f}$ and $\boldsymbol{h}$ are given functions which may be non-linear, time-varying and multi-variable.

We wish to identify a non-linear (neural network) mapping $\boldsymbol{F}$

$$\begin{aligned} \hat{\boldsymbol{X}}(k) &= \boldsymbol{F}\left(\hat{\boldsymbol{X}}(k-1), \boldsymbol{U}(k-1), \boldsymbol{D}(k-1), \boldsymbol{E}(k-1)\right) \\ \hat{\boldsymbol{Y}}(k) &= \boldsymbol{H}\hat{\boldsymbol{X}}(k) \\ \boldsymbol{E}(k) &= \boldsymbol{Y}(k) - \hat{\boldsymbol{Y}}(k) \end{aligned} \tag{2}$$

based on samples of the system inputs and outputs $\{\boldsymbol{u}(k), \boldsymbol{y}(k)\}_{k=0}^{N}$ - the training set, such that the prediction error $\boldsymbol{E}$ defined by equations (2) becomes small. $\boldsymbol{F}$ is chosen to be a Multi-Layer Perceptron (MLP). We assume that the output mapping to be fixed, $\boldsymbol{H} = [\boldsymbol{I}\ \boldsymbol{0}]$, where $\boldsymbol{I}$ and $\boldsymbol{0}$ are identity and zero matrices of appropriate dimensions.

We will use in this paper Feed-Forward Neural Network MLP with a single hidden layer. This structure is shown in matrix notation in Fig. 1, [8].

Matrix $\boldsymbol{W}_1$ represents the input weights, matrix $\boldsymbol{W}_2$ represents the output weights, $\boldsymbol{F}_1$ represents a vector function containing the non-linear ($tanh$) neuron functions. The "$\boldsymbol{1}$" shown in Fig.1 together with the last column in $\boldsymbol{W}_1$ gives the offset in the network. The net

* Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 042 00 Košice, E- mail: Anna.Jadlovska@tuke.sk

ISSN 1335-3632

input is represented by vector $\boldsymbol{Z}_{in}$ and the net output is represented by vector $\hat{\boldsymbol{Z}}_{out}$. The mismatch between the desired output $\boldsymbol{Z}_{out}$ and the predicted output $\hat{\boldsymbol{Z}}_{out}$ is the prediction error $\boldsymbol{E}$.

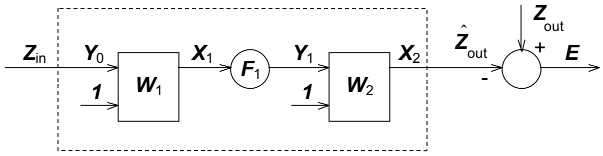

Fig. 1. Matrix block diagram of an MLP

The output from MLP can be written as:

$$\hat{\boldsymbol{Z}}_{out} = \boldsymbol{W}_2 \boldsymbol{F}_1 \left(\boldsymbol{W}_1 \left\{ \begin{matrix} \boldsymbol{Y}_0 \\ \boldsymbol{1} \end{matrix} \right\} \right) \tag{3}$$

From a trained MLP (by Back- Propagation Algorithm -first-order gradient method, Gauss-Newton algorithm — second-order gradient method) which has m_0 inputs and m_2 outputs can be found $m_2 \times m_0$ gain matrix $\boldsymbol{N}$ by differentiating with respect to the input vector of the network.

The gain matrix $\boldsymbol{N}$ can be calculated from (3)

$$\boldsymbol{N} = \frac{\mathrm{d}\hat{\boldsymbol{Z}}_{out}}{\mathrm{d}\hat{\boldsymbol{Z}}_{in}^{\top}} = \frac{\mathrm{d}\hat{\boldsymbol{Z}}_{out}}{\mathrm{d}\boldsymbol{Y}_1^{\top}} \frac{\mathrm{d}\boldsymbol{Y}_1}{\mathrm{d}\boldsymbol{X}_1^{\top}} \frac{\mathrm{d}\boldsymbol{X}_1}{\mathrm{d}\boldsymbol{Z}_{in}^{\top}} = \boldsymbol{W}_2\ \boldsymbol{F}_1'\left(\boldsymbol{X}_1\right) \boldsymbol{W}_1^{*} \tag{4}$$

where $\boldsymbol{W}_1^{*} \doteq \boldsymbol{W}_1$ (excluding last column). The above mentioned gain matrix $\boldsymbol{N}$ allows on-line estimation of the actual model parameters from off-line trained neural model - MLP of the non-linear process.

We know from linear identification that for a linear process with m inputs, n states and p outputs the following model is often used

$$\begin{aligned} \hat{\boldsymbol{x}}(k) &= \boldsymbol{\Phi}\hat{\boldsymbol{x}}(k-1) + \boldsymbol{\Gamma}\boldsymbol{u}(k-1) + \boldsymbol{K}\boldsymbol{e}(k-1) \\ \boldsymbol{y}(k) &= \boldsymbol{H}\hat{\boldsymbol{x}}(k) + \boldsymbol{e}(k) \end{aligned} \tag{5}$$

where $\hat{\boldsymbol{x}}$ is the state vector (order n), $\boldsymbol{u}$ is the input vector (order m), $\boldsymbol{y}$ is the output vector (order p) and $\boldsymbol{e}$ is the prediction error (order p). For the linear case $\boldsymbol{\Phi}, \boldsymbol{\Gamma}, \boldsymbol{K}$ and $\boldsymbol{H}$ are constant matrices of dimension $n\times n, n\times m, n\times p$ and $p\times n$ respectively. It is assumed that $\hat{\boldsymbol{x}}$ is not measurable (incomplete state information) and the model is named the Innovation State Space model. The matrix $\boldsymbol{H}$ is chosen to $\boldsymbol{H} = \{\boldsymbol{I}_{p,p}\boldsymbol{O}_{p,n-p}\}$, where $\boldsymbol{I}_{p,p}$ is $p\times p$ unity matrix and $\boldsymbol{O}_{p,n-p}$ is a $p\times(n-p)$ zero matrix. This means, that the first p elements of the state vector $\hat{\boldsymbol{x}}$ is filled with $\hat{\boldsymbol{y}}$. Given only matched pairs of input measurement $\boldsymbol{u}(k)$ and desired output measurement $\boldsymbol{y}(k)$, $k = 1, ..., N$, learning this model is equivalent to solving the Extended Kalman Filter problem, [5].

With the inspiration from this linear state space model (5) in this paper we will apply the idea using a dynamic neural network model, where no all inputs and outputs from the network are measurable. This type of model is the well-known Non-linear Innovation State Space model (NISS), [4], [10].

The non-linear Innovation State Space model (Kalman Predictor) can be defined by equations (6)

$$\begin{aligned} \hat{\boldsymbol{X}}(k) &= \boldsymbol{F}\left(\hat{\boldsymbol{X}}(k-1), \boldsymbol{U}(k-1), \boldsymbol{E}(k-1), \Theta\right) \\ \hat{\boldsymbol{Y}}(k) &= \boldsymbol{H}\hat{\boldsymbol{X}}(k) \\ \boldsymbol{Y}(k) &= \hat{\boldsymbol{Y}}(k) + \boldsymbol{E}(k) \end{aligned} \tag{6}$$

where $\boldsymbol{F}$ is non-linear vector function, $\boldsymbol{\Theta}$ represents vector of the parameters and $\boldsymbol{E}$ is the prediction error (order p), $\boldsymbol{U}$ is input vector (order m), $\boldsymbol{X}$ is state vector (order n) and $\boldsymbol{Y}$ is the output vector (order p).

The neural NISS model with input vector $\boldsymbol{Z}_{in}(k)$ and output vector $\boldsymbol{Z}_{out}(k)$ is shown in Fig. 2, which is a recurrent network where

$$\begin{aligned} \boldsymbol{Z}_{in}(k) &= \left\{ \begin{matrix} \hat{\boldsymbol{X}}(k-1) \\ \boldsymbol{U}(k-1) \\ \boldsymbol{E}(k-1) \end{matrix} \right\} \\ \boldsymbol{Z}_{out}(k) &= \hat{\boldsymbol{X}}(k) \end{aligned} \tag{7}$$

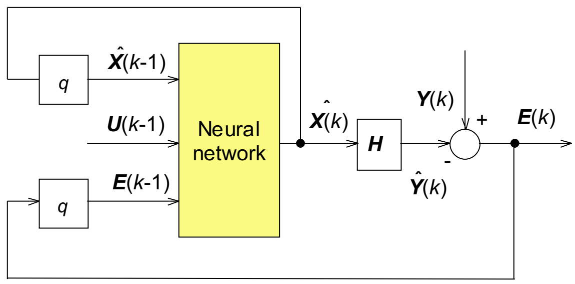

Fig. 2. Non-linear State Space Model (NISS)

Because neural state space NISS model in Fig. 2 contains feedback loops around MLP, we will apply for training this recurrent network a second order Recursive Prediction Error Method (RPEM) using Gauss-Newton search direction, [2], [4], [7].

After training the neural network MLP the actual gain matrix $\boldsymbol{N}(k)$ can be on-line estimated which is calculated by (4) and for NISS model by (8):

$$\begin{aligned} \boldsymbol{N}(k) &= \frac{\mathrm{d}\hat{\boldsymbol{X}}_{out}(k)}{\mathrm{d}\boldsymbol{Z}_{in}^{\top}(k)} = \\ &= \frac{\mathrm{d}\hat{\boldsymbol{X}}(k)}{\mathrm{d}\left\{\hat{\boldsymbol{X}}^{\top}(k-1)\ \boldsymbol{U}^{\top}(k-1)\ \boldsymbol{E}^{\top}(k-1)\right\}} = \\ &= \left\{ \frac{\partial\hat{\boldsymbol{X}}(k)}{\partial\hat{\boldsymbol{X}}^{\top}(k-1)}\ \frac{\partial\hat{\boldsymbol{X}}(k)}{\partial\boldsymbol{U}^{\top}(k-1)}\ \frac{\partial\hat{\boldsymbol{X}}(k)}{\partial\boldsymbol{E}^{\top}(k-1)} \right\} = \\ &= \left\{\hat{\boldsymbol{\Phi}}(k),\ \hat{\boldsymbol{\Gamma}}(k),\ \hat{\boldsymbol{K}}(k)\right\} \end{aligned} \tag{8}$$

For a non-linear process the actual linearized parameters $\hat{\boldsymbol{\Phi}}(k)$, $\hat{\boldsymbol{\Gamma}}(k)$, $\hat{\boldsymbol{K}}(k)$ are not constant matrices but depend on the actual values of the input, state and output vectors, [5].

3 NONLINEAR CONTROL USING PARAMETER STATE ESTIMATION

The model of non-linear process and the training method have been considered generally for the multivariable case in part 2. Next we will think about control design for non-linear MISO (Multi Inputs/Single Output) process using neural state space model NISS (6).

A trained neural NISS model representing a model of the non-linear process we use for on-line state estimation Of actual process parameters by gain matrix $\boldsymbol{N}$, [8], [10]. This linearization technique allows on-line tuning of the controller parameters using the pole-placement control strategy, which is well known from the linear control theory, [1]. In this paper the pole placement method as a control concept is formulated in a way which allows the neural model found in section 2 to be used as a state observer. An example of the control structure using estimation process parameters from NISS model is illustrated in Fig. 3.

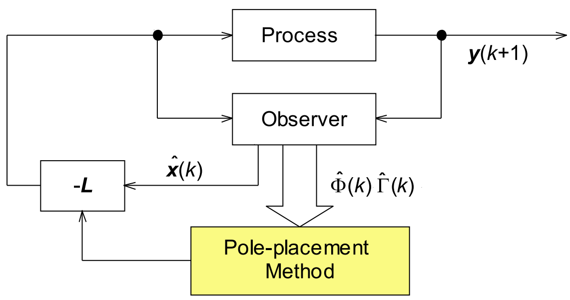

Fig. 3. The principle of pole placement control using a state observer

The aim is to design the feedback gain $\boldsymbol{L}$ so that the closed loop has a set of prescribed eigenvalues. Because the linearization model will change from sample to sample, it is necessary to recalculate $\boldsymbol{L}$ accordingly. If we have trained a simulation model - neural observer NISS we can design the control law. Since full state information is not available, the control law is chosen as a linear feedback from the state observer by equation (9)

$$\boldsymbol{u}(k) = -\boldsymbol{L}\hat{\mathbf{x}}(k) \tag{9}$$

where $\boldsymbol{L}$ is the state feedback matrix which places the eigenvalues of the closed loop system.

In order to include integral action in the controller we will augment the description with an integral state. The discrete-time equivalent of differentiation, $\Delta = 1 - q^{-1}$, is applied to the model, gives a differential model of the states

$$\begin{aligned} \Delta\hat{\mathbf{x}}(k) &= \hat{\boldsymbol{\Phi}}\Delta\hat{\mathbf{x}}(k-1) + \hat{\boldsymbol{\Gamma}}\Delta\boldsymbol{u}(k-1) \\ \Delta\boldsymbol{y}(k) &= \boldsymbol{H}\Delta\hat{\mathbf{x}}(k) \end{aligned} \tag{10}$$

$\Delta\boldsymbol{u}(k)$ is the input to the model, while $\boldsymbol{y}(k)$ is the desired output from the model. Then the augmented state space model is introduced in which the original state vector is supplemented with the integral output state

$$\boldsymbol{y}(k) = \boldsymbol{y}(k-1) + \Delta\boldsymbol{y}(k) \tag{11}$$

The augmented state vector is defined as:

$$\tilde{\mathbf{x}}(k) = \begin{pmatrix} \Delta\hat{\mathbf{x}}(k) \\ \boldsymbol{y}(k) \end{pmatrix} \tag{12}$$

including the integral state, is introduced giving

$$\begin{aligned} \tilde{\mathbf{x}}(k) &= \tilde{\boldsymbol{\Phi}}\tilde{\mathbf{x}}(k-1) + \tilde{\boldsymbol{\Gamma}}\Delta\boldsymbol{u}(k-1) = \\ &= \begin{pmatrix} \hat{\boldsymbol{\Phi}} & \boldsymbol{0} \\ \boldsymbol{H}\hat{\boldsymbol{\Phi}} & \boldsymbol{I} \end{pmatrix} \tilde{\mathbf{x}}(k-1) + \begin{pmatrix} \hat{\boldsymbol{\Gamma}} \\ \boldsymbol{H}\hat{\boldsymbol{\Gamma}} \end{pmatrix} \Delta\boldsymbol{u}(k-1) \\ \boldsymbol{y}(k) &= (\,\boldsymbol{0} \quad \boldsymbol{I}\,))\,\tilde{\mathbf{x}}(k) \end{aligned} \tag{13}$$

In equation (13) $\boldsymbol{0}$ and $\boldsymbol{I}$ are zero- and unity-matrices of suitable dimensions. If we use this equation $\Delta\boldsymbol{u}(k) = -\tilde{\boldsymbol{L}}\tilde{\mathbf{x}}(k)$ for calculating the control law of the state controller, then the characteristic polynomial of the augmented closed loop system is given by

$$P(z) = \det\left(\boldsymbol{I} - \left(\tilde{\boldsymbol{\Phi}} - \tilde{\boldsymbol{\Gamma}}\tilde{\boldsymbol{L}}\right)\right) \tag{14}$$

In the other words, if the desired closed loop poles are given as the roots of the polynomial $P_d(z)$, the control law $\tilde{\boldsymbol{L}}$ is calculated by solving $P(z) = P_d(z)$, *eg* by the Ackermann formula, [1].

The controller can be separated into two contributions, one concerning the differential states and one concerning the integral state

$$\begin{aligned} \Delta\boldsymbol{u}(k) &= -\tilde{\boldsymbol{L}}\tilde{\mathbf{x}}(k) = -\left(\,\mathbf{L}_{\Delta x} \quad \mathbf{L}_y\,\right)\begin{pmatrix} \Delta\mathbf{x}(k) \\ \boldsymbol{y}(k) \end{pmatrix} = \\ &= -\mathbf{L}_{\Delta x}\Delta\mathbf{x}(k) - \mathbf{L}_y\boldsymbol{y}(k) \end{aligned} \tag{15}$$

The final step in the regulator design is introducing an external reference $\boldsymbol{y}_r(k)$

$$\Delta\boldsymbol{u}(k) = -\mathbf{L}_{\Delta x}\Delta\mathbf{x}(k) - \mathbf{L}_y\left(\boldsymbol{y}(k) - \boldsymbol{y}_r(k)\right) \tag{16}$$

The implementation of the resulting controller, including a neural network model for parameter estimation, is shown in Fig. 4.

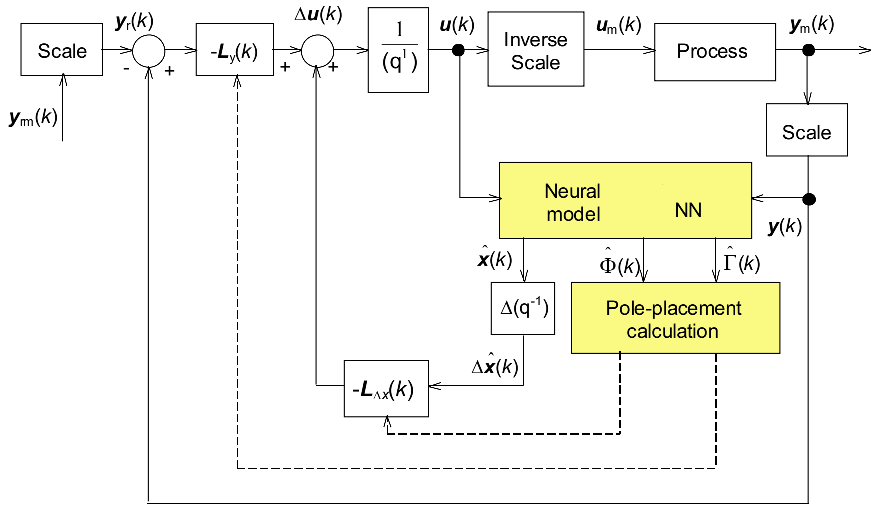

Fig. 4. Implementation of the state controller using trained neural network for parameter estimation

4 SIMULATION RESULTS

The idea and results of the estimation of the process parameters from an off-line trained neural NISS model as an observer and its using for tuning parameters of controller designed by pole-placement strategy (non-linear system control) are demonstrated for non-linear test process - two tanks system, [5].

The mathematical model of this system can be described as follows

$$\dot{h}_1(t) = \frac{1}{A}\left[q_{1,1}(t) - q_{2,1}(t)\right]$$
$$\dot{h}_2(t) = \frac{1}{A}\left[q_{1,2}(t) - q_{2,2}(t)\right] \qquad (17)$$

where

$h_1(t)$, $h_2(t)$ denotes the water levels (m) in the two tanks, which are in the interaction,

$q_{1,1}(t)$, $q_{1,2}(t)$ denotes the input flows $(\mathrm{m}^3\mathrm{s}^{-1})$ of the tanks,

$q_{2,1}(t)$, $q_{2,2}(t)$ denotes the outflows $(\mathrm{m}^3\mathrm{s}^{-1})$ of the tanks.

$-q_{2,1}(t) = q_{1,2}(t) = \alpha\, z_1(t)\sqrt{2g(h_1(t) - (h_2(t)}$,

$q_{2,2}(t) = \alpha\, z_2(t)\sqrt{2gh_2(t)}$

and where

$z_1(t)$ and $z_2(t)$ are the rises (m) of the output outlets of the two tanks.

The other parameters of two tanks are

the cross section area of the tanks $A = 1\,\mathrm{m}^2$,

the parameter of the tank output outlet $\alpha = 0.5$ m, the constant of gravity $g = 9,81\,\mathrm{ms}^{-2}$.

The inputs of the system are

the input flow of the first tank $u_1(t) = q_{1,1}(t)$,

the rise of output outlet of the second tank $u_2(t) = z_2(t)$,

the rise of output outlet of the first tank $z_1(t)$, $d(t) = z_1(t)$.

The output of the system is water level in the second tank $y(t) = h_2(t)$.

We consider NISS model with 6 inputs and 8 neurons in the hidden layer. The activation function in the hidden layer is "tanh" function and in the output layer is selected as a linear function. The actual gain matrix $\boldsymbol{N}(k)$ can be calculated by (4) and actual values of estimated parameters can be obtained from $\boldsymbol{N}(k)$ by (8).

Presentation of results of non-linear control designed by the pole placement method using on-line parameter estimation from an off-line trained neural state model is illustrated in Fig. 5 at the change of one parameter of the non-linear process (17).

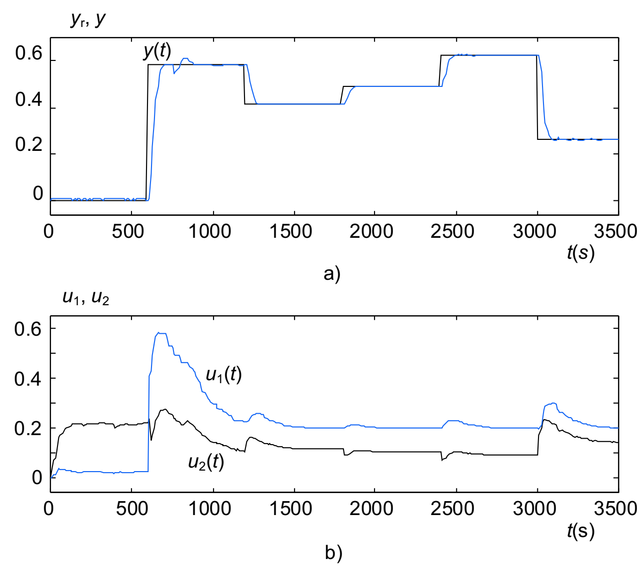

Fig. 5. State controller based on parameter estimation from neural NISS model: a) the output of the process $y(t)$, b)the control inputs $u_1(t), u_2(t)$

Simulation study shows that the non-linear observer-based control loop with the desired closed-loop poles gives a critically damped response of the output tracking. This example shows the real power of the neural modeling using the structure NISS model and the possibility to apply the method of pole- placement known from the linear control theory for control of non-linear MISO process.

5 CONCLUSION

In this paper the neural NISS model as a Kalman predictor is trained for a non-linear MISO process. After training, this NISS model can be used in a closed control loop for on-line state estimation of the process parameters, which allows tuning of the controller parameters by the pole-placement method. Practical simulations by languages Matlab/Simulink, Neural Toolbox and NNSID Toolbox illustrate that this control strategy using linearization technique by the gain matrix from neural state model produces excellent performance for control of non-linear MISO process. But this controller design can be applied only for a non-linear process which does not contain hard non-linearities.

Acknowledgements

The paper was worked out as part of the solution of the scientific project "Optimization of the Process Control Based on the Use of Informatics Methods", No 1/0373/03.

References

[1] ÅSTRÖM, K. J.—WITTENMARK, B.: Computer Controlled Systems, Theory and design, Prentice-Hall, second edition, 1990.

[2] CHEN, S.—BILLINGS, S.—GRANT, P.: Non-linear System Identification Using Neural Networks, International Journal of control **51** No. 6 (1990), 1191-1214.

[3] JADLOVSKÁ, A.: An Optimal Tracking Neuro-controller for Nonlinear Dynamic Systems, Control System Design, A Proceedings volume from IFAC Conference Bratislava, Slovak Republic, 18-20 June, 2000, Published for IFAC by Pergamon - an Imprint of Elseevier Science, 493-499, 2000, ISBN 00-080435467.

[4] JADLOVSKÁ, A.: Non-linear Control Using Parameter Estimation from Forward Neural Model, Journal of Electrical Engineering - Elektrotechnický časopis **53** No. 11-12, 324-327, Centre of IEE, FEI STU Bratislava ISSN 1335-3632.

[5] JADLOVSKÁ, A.: Modeling and Control of Dynamic Processes Using Neural Networks, FEI - TU Košice, Informatech Ltd., (2003), 173, ISBN 80-88941-22-9. (in Slovak)

[6] LEONTARITIS, I., J.: Input-output Parametric Models for Non-linear Systems, Part 1 and 2, International Journal of Control **41** No. 2 (1985), 303-344.

[7] LJUNG, L.: System Identification, Theory for User, Prentice Hall, first edition, 1987.

[8] NAJVÁREK, J.: Matlab and neural networks, FEI VUT, Brno, 1996.

[9] NARENDRA, K. S.—LEVIN, A. U.: Identification and Control of Dynamical Systems Using Neural Networks, IEEE Transaction on Neural Networks **1** No. 1 (1990), 4-27.

[10] NORGAARD, M.: NNSID Toolbox, Version 1.1, Technical Report, Department of Automation DTU, 1997.

[11] PERDUKOVÁ, D.—FEDOR, P.—FEDÁK, V.: Reference Model Robust Control of MIMO Systems with an Incomplete Acess to State Variables, Proc. of Int. Conference "Motion Control for Intelligent Automation" Vol. II, Perugia (1992), 183-187.

Anna Jadlovská (MSc, PhD), was born in Banská Bystrica, Slovakia, on October 29, 1960. She received MSc degree in technical cybernetics from the Faculty of Electrical Engineering of the Technical University Košice in 1984 and PhD degree in automation and control in 2001 at the same university. Her PhD thesis dealt with modeling and control of non-linear processes using neural networks. Since 1994 she has worked at the Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics at TU in Košice as Assistant Professor. Her main research and teaching activities include adaptive and optimal control, especially predictive control with constrains for non-linear processes with time-variant parameters using neural networks (Intelligent Control Design). She is an author of articles and contributions published in journals and international conference proceedings. She is also co-author of three monographs.