SSNN toolbox for non-linear system identification

Marcel Luzar, Andrzej Czajkowski
Institute of Control and Computation Engineering, University of Zielona Góra,
ul. Podgórna 50, 65–246 Zielona Góra, Poland
E-mail: {M.Luzar,A.Czajkowski}@issi.uz.zgora.pl

Abstract.
The aim of this paper is to develop and design a State Space Neural Network toolbox for
a non-linear system identification with an artificial state-space neural networks, which can be
used in a model-based robust fault diagnosis and control. Such toolbox is implemented in
the MATLAB environment and it uses some of its predefined functions. It is designed in the
way that any non-linear multi-input multi-output system is identified and represented in the
classical state-space form. The novelty of the proposed approach is that the final result of
the identification process is the state, input and output matrices, not only the neural network
parameters. Moreover, the toolbox is equipped with the graphical user interface, which makes
it useful for the users not familiar with the neural networks theory.

1. Introduction
The identification of dynamic models out of experimental data has very often been motivated
and supported by the presumed ability to use the resulting models as a basis for model-based
Fault Diagnosis (FD) and control design [7, 15, 19, 21]. As such, FD and control design is
considered an important intended-application area for identified models. On the other hand,
model-based FD and control design is built upon the assumption that a reliable model of the
plant under consideration is available. Without model, there is no model-based FD and control
design. Thus, there is a need for good quality system identification methods for the control
design purposes.

In this paper, the system identification method which use the Artificial Neural Networks
(ANNs) is proposed. Such technique is known from many years [2, 6, 9, 13, 14]. It already
proven its usefulness in the tasks on linear systems identification [3] or prediction purposes [10].
However, for the non-linear ones, especially high-order and dynamic systems, it not always gives
the satisfactory results [8, 20]. Moreover, there is a need for a tool, which helps the researchers
to identify the system model based on the input/output measured data, without the knowledge
about the neural networks. There is some predefined toolboxes i.e. in MATLAB, however any
of them uses the state-space ANNs, which nowadays becomes more and more popular form of
the non-linear system representation [5, 4]. A good example that proves that the such toolbox is
needed is the GMDH Toolbox for neural network-based modelling, which was downloaded from the
Researchgate portal around 1300 times [12] and was implemented by the author of this
paper. Thus, in this paper, the implementation of the State-Space Neural Network (SSNN)
toolbox is presented, which overcomes above mentioned problems. It requires from the users
only basic experience with the ANNs and Matlab. It is equipped with some automatic methods.
and functions, which allows to clearly identify any Mult-Input Multi-Output (MIMO) non-linear dynamic system. Some of them based on the algorithms implemented in the NNSYSID toolbox, designed by Nørgaard [16]. Obviously, they were suitable modified to identify and linearize the system direct in the state-space form. Note that implementation of this toolbox is the first stage of realizing the grant Artificial neural networks for robust fault diagnosis and control of non-linear systems conducted by the author of this paper.

The paper is organised as follows. After short introduction, Section 2 presents the design assumptions of the toolbox. In Section 3, the theoretical background is given. Section 3 describes the implementation of the toolbox with explanation of its functionalities. The system identification results, which verify the proposed approach are given in the Section 5. Finally, Section 6 concludes the paper.

2. Design assumptions

Prior to implementation, the following assumptions are discussed with the interested researches and taken into account:

- possibility to identify any non-linear, dynamic MIMO system with SSNN,
- transformation of the SSNN parameters into state-space matrices,
- possibility to choose options for the neural network design in the professional mode such as:
  (i) neuron number in the output and hidden network layer,
  (ii) neural network training stopping criteria,
  (iii) the weight decay,
  (iv) backpropagation parameters,
- possibility to import training and validation data direct from the file or Matlab’s workspace,
- error handling,
- Graphical User Interface (GUI).

Moreover, the SSNNtool has to be equipped with the help files which allows to understand used techniques in details.

3. Theoretical background

3.1. Process model

As was mentioned in the introduction of this paper, in the toolbox for model identification, the dynamic ANN is implemented. Among many different dynamic ANNs there is a special class of network, called the State Space Neural Network (SSNN), which scheme is presented in Fig. 1. The hidden layer outputs are propagated back to the input layer through a bank of unit delays.

![Figure 1. The state-space neural network with one hidden layer](image)

The number of such delays defines the order of the system. In general, a user may decide how
many neurons are used to produce feedback. Let \( u(k) \in \mathbb{R}^n \) be the input vector, \( \bar{x}(k) \in \mathbb{R}^q \) - the output of the hidden layer at time \( k \), and \( \bar{y}(k) \in \mathbb{R}^m \) - the output vector. Then the state-space form of the neural model given in Fig. 1 is defined by the following equations

\[
\bar{x}(k+1) = \bar{g}(\bar{x}(k), u(k)) \\
\bar{y}(k) = C\bar{x}(k),
\]

(1)

where \( \bar{g}(\cdot) \) is a non-linear function characterizing the hidden layer, and \( C \) represents synaptic weights between hidden and output neurons.

Introducing the weight matrix between input and hidden layers \( W^u \) and the matrix of recurrent links \( W^x \), the previous formulae (1) can be rewritten as follows:

\[
\bar{x}(k+1) = h(W^x\bar{x}(k) + W^u u(k)) \\
\bar{y}(k) = C\bar{x}(k),
\]

(2)

where \( h(\cdot) \) denotes the activation function of the hidden neurons. In the most cases, when the hyperbolic tangent activation function is chosen, the modelling results are satisfactory. For the state space model the outputs of hidden neurons which constitute feedbacks are, in general, unknown during training. Thus, state space neural models can be trained only by minimizing the simulation error. If state measurements are available, the training can be carried out much easier using the series-parallel identification scheme, similarly as for the external dynamic approach (the feedforward network with tapped delay lines). In spite of this drawback, the state-space neural networks popularity was build due to its number of advantages, contrary to fully and partially recurrent networks ([17], and they are as follows:

- The determination of the states number (model order) may be done independently from the hidden neuron number. The obvious result of this feature is that the responsibility for defining the state of the network is on the neurons that propagate their outputs back to the input layer through delays. As a consequence, the output neurons are eliminated from the state definition [17]. In the recurrent networks, e.g. Williams-Zipser, Elman, locally recurrent networks, the model order is directly influenced by the number of the neurons, which significantly impedes the phase of modelling.
- Model states feed the network input (which makes them easily accessible). In case when state measurements are available at some time instants, this property can be very useful.
- State space neural models are useful in the fault diagnosis and tolerant control frameworks. State representation allows to approximate size of a fault effect and to handle different kind of faults including multiplicative and additive ones.

Mentioned above advantages of SSNN make models of this kind a very interesting and promising tool used to solve different engineering issues i.e. the fault diagnosis problem. Also the class of non-linear state space models is strongly evaluated in different scientific approaches as a nominal model.

3.2. Linearisation of the state space model

In the toolbox, to deal with the model non-linearity, the instantaneous linearisation is implemented. The primary idea is very simple. At each discrete time sample a linear model is extracted from the state space neural model. Model is linearised using Taylor series expansion around the current operating point \( (x, u) = (x(\tau), u(\tau)) \) and rejection of the non-linear components. The first order Taylor series expansion of the SSNN model is presented in the
following equations:

\[ h(x(k), u(k)) = h(x(\tau), u(\tau)) + \frac{\partial h}{\partial x}(x, u) \Delta x + \frac{\partial h}{\partial u}(x, u) \Delta u \]

\[ = h(x(\tau), u(\tau)) + h'W^x(x(k) - x(\tau)) + h'W^u(u(k) - u(\tau)). \]  

The following linear state space model is obtained:

\[ \bar{x}(k + 1) = A\bar{x}(k) + Bu(k) + D \bar{y} = C\bar{x}(k), \]  

is obtained, where \( A = h'W^x, B = h'W^u, D = x(\tau) - A\bar{x}(\tau - 1) - Bu(\tau - 1). \) The \( h' \) is the activation function first derivative. If the activation function is implemented in the form of hyperbolic tangent, this derivative can be simply calculated as follows:

\[ h' = 1 - \tanh^2. \]

This property allows the linearisation to be performed very quickly. This is crucial in the terms of computation burden especially when calculations need to be carried out in real-time.

4. SSNN Toolbox design

The entire implementation of the toolbox was done using the Matlab environment. The graphical view and functionality is inspired with the Neural Network toolbox, which is predefined in Matlab. Thus, the SSNN tool consist of two windows: the main window called SSNN Toolbox and the import data window. In the import data window presented in Fig. 2, there are three panels: the data source selection panel, the variable selection panel as well as the training input data, the validation input data, the training output data and, finally, the validation output data panel. The data can be imported either from an existing file with extension *.mat or from the Matlab workspace, respectively. After assigning the variables from the list as the relevant data (by the selection button Set), the import data window is closed. The main window presented
in Fig. 3 is equipped with the tools for choosing the neural network training parameters. First, user can select its experience in the NN field. When the Beginner option is selected, user can define only neuron number in the hidden and output layer. Other necessary parameters are set as default and their values are given in the help file. In the Advanced mode, user can define above mentioned parameters and moreover, weight decay, stopping criteria and backpropagation parameters.

![SSNN Toolbox main window](image1)

**Figure 3.** SSNN Toolbox main window

If the training data are imported, the SSNN training can be initiated by pressing the NN Train button. In the case, when data are not imported, or if are not correctly assigned as training/validation input/output data, the warning window is shown in Fig. 4.

![Warning message](image2)

**Figure 4.** Warning message

When the all NN parameters are set and the data is imported correctly, the NN training is started. During this process, number of iterations with associated modelling errors are displayed in the command window. When the NN training is finished, the number of figures are displayed automatically. In this figures, the results of the NN modelling are presented. If the SSNN is trained in the satisfaction way and it reflects the real system, the neural model can be linearized and presented in the classic state-space form described in Sec. 3.2 by pressing the Identify SS model button. The state-space matrices $A$, $B$, $C$ and $D$ obtained during linearization process are automatically saved in the current folder.

It is worth to note that the SSNN tool is equipped with the help files, which explains all used methods in details. Moreover, the SSNN theoretical background and linearization approach used
in the toolbox is presented. To open the help file user have to press the Help button in the Main window.

5. Illustrative example
To clearly exhibit the effectiveness of the proposed approach, two different scenarios of the system identification is performed. First, the neural network modelling of the multi-tank system is presented. Such laboratory equipment may be perceived as a Single Input Multiple Output (SIMO) non-linear system. It is chosen because its analytical model is known and given by the producer, thus it is easy to compare it with model obtained with the SSNN approach proposed in this paper. Second experiments regards the modelling of the wind turbine, which is highly non-linear MIMO system.

5.1. Multi-tank system identification
The multi-tank system presented in Fig. 5 is chosen for show the effectiveness of the developed toolbox in the system identification task. Experimental results obtained with the toolbox are then compared with the analytical model given by the manufacturer.

![Multi-tank system](image.png)

**Figure 5.** Multi-tank system

It consists of three separate tanks placed each above other and equipped with drain valves and level sensors based on a hydraulic pressure measurement. Each of them has a different cross-section in order to reflect system non-linearities. The lower bottom tank is a water reservoir for the system. A variable speed water pump is used to fill the upper tank. The water outflows the tanks due to gravity. The considered multi-tank system has been designed to operate with an external, PC-based digital controller. The control computer communicates with the level sensors, valves and a pump by a dedicated I/O board and the power interface. The I/O board is controlled by the real-time software, which operates in a Matlab/Simulink environment. The description of the parameters presented in Fig. 5 are given in Tab. 1.

For laminar outflow, where turbulence and acceleration of the liquid are commuted and inflow rate is constant \( q = \text{const} \), the flow equation is as follows:

\[
Q = \mu S \sqrt{2gH},
\]
Table 1. Specification of system parameters

| Signal  | Description                  | Units     | Operating range                  |
|---------|------------------------------|-----------|----------------------------------|
| q       | Flow rate of the fluid       | m³/s      | $0 \leq q \leq 1.5e^{-4}$        |
| $H_1$, $H_2$, $H_3$ | Level of the tanks | m         | $0.01 \leq H_1, H_2, H_3 \leq 0.35$ |
| $C_1$   | Open level of $C_1$ valve    | raw (%)   | $0 \leq C_1 \leq 1.0057e^{-4}$  |
| $C_2$   | Open level of $C_2$ valve    | raw (%)   | $0 \leq C_2 \leq 1.1963e^{-4}$  |
| $C_3$   | Open level of $C_3$ valve    | raw (%)   | $0 \leq C_3 \leq 9.8008e^{-5}$  |

where $Q$ is outflow rate from the tank, $\mu$ stands for orifice outflow coefficient, $S$ denotes the area of the orifice and $H$ is the level of the fluid measured from level 0.

![Figure 6. Laminar outflow from the tank](image)

The outflow rate of the system is described by:

$$
\frac{dV_1}{dt} = q - C_1 H_1^{\alpha_1},
\frac{dV_2}{dt} = C_1 H_1^{\alpha_1} - C_2 H_2^{\alpha_2},
\frac{dV_3}{dt} = C_2 H_2^{\alpha_2} - C_3 H_3^{\alpha_3},
$$

(7)

where $\alpha_i$ denotes the $i$-th tank dynamic flow rate coefficient.

The equation (7) can be transformed into:

$$
\frac{dH_1}{dt} = \frac{1}{\beta_1(H_1)} q - \frac{1}{\beta_1(H_1)} C_1 H_1^{\alpha_1},
\frac{dH_2}{dt} = \frac{1}{\beta_2(H_2)} C_1 H_1^{\alpha_1} - \frac{1}{\beta_2(H_2)} C_2 H_2^{\alpha_2},
\frac{dH_3}{dt} = \frac{1}{\beta_3(H_3)} C_2 H_2^{\alpha_2} - \frac{1}{\beta_3(H_3)} C_3 H_3^{\alpha_3},
$$

(8)

where $H_i$ is $i$-th tank fluid level and $\beta_i(H_i)$ stands for cross-sectional area of $i$-th tank at level $H_i$.

Cross-sectional areas for each of the tank are as follows:
\[ \beta_1(H_1) = aw - \text{constant cross-sectional area of the upper tank}, \]

\[ \beta_2(H_2) = cw + \frac{H_2}{H_{2\text{max}}}bw - \text{variable cross-sectional area for the middle tank (conical)}, \]

\[ \beta_3(H_3) = w\sqrt{R^2 - (R - H_3)^2} - \text{variable cross-sectional area of the lower tank (spherical)}, \]

where \( C_i \) denotes the resistance of output orifice of \( i \)-th tank.

Variable cross-sectional areas of the tanks introduce main nonlinearities and require particular approach.

The set of equations (8) is equivalent to:

\[ F_1(u, x_1) = \frac{1}{\beta_1(x_1)}u - \frac{1}{\beta_1(x_1)}C_1 x_1^{a_1}, \]

\[ F_2(x_1, x_2) = \frac{1}{\beta_2(x_2)}C_1 x_1^{a_1} - \frac{1}{\beta_2(x_2)}C_2 x_2^{a_2}, \]

\[ F_2(x_2, x_3) = \frac{1}{\beta_3(x_3)}C_2 x_2^{a_2} - \frac{1}{\beta_3(x_3)}C_3 x_3^{a_3}, \]

where \( x_i \) denotes the \( i \)-th tank fluid level and \( u \) is the control input.

Calculated mathematical model is a set of ordinary differential equations and it is ready to be compared with the non-linear SSNN model.

To obtain the SSNN model with the toolbox, 7500 samples of the training and validation data from the real system were collected. The control signal were chosen randomly from the following interval

\[ 0 \leq u_k \leq 0.0001 \]

and suitably rescaled.

The result of liquid level modelling in the first tank is presented in Fig. 7 with associated modelling errors. It is clear, that both models reflects real system behaviour with good quality. Despite this, based on modelling error presented in the second graph in Fig. 7 it is easy to see that model obtained with the SSNN toolbox reflects the real system better than analytical one. The same conclusion can be drawn by analysing Fig. 8 and Fig. 9, in which the result of liquid level modelling in second and third tank is depicted, respectively. Still, the SSNN model is better than analytical one which proves the usefulness of the proposed approach in i.e. fault detection and control.

### 5.2. Wind turbine model identification

The presented second example is based on the wind turbine description provided in [1] with suitable modifications. Note that an analytical model is not defined. The network feeding data is as follows:

\[ u_k = \begin{bmatrix} \beta_k \\ \rho_k \\ v_{wk} \end{bmatrix}, \quad y_k = \begin{bmatrix} \tau_k \\ \omega_r_k \\ \omega_g_k \\ \beta_k \end{bmatrix} \]

where \( \beta \) is the pitch angle, \( \rho \) stands for zero-torque speed, \( v_w \) is the wind speed, \( \tau \) denotes torsion angle, \( \omega_r \) and \( \omega_g \) is the rotor and generator velocity, respectively. The identification process was done with the Beginner mode, thus the neural network training parameters was set as default ones. The number of neurons was set to 13 and 6 in the hidden and output layer, respectively.

Graphs a), b), c) and d) in Fig. 10 presents the results of modelling for the rotor velocity, generator velocity and pitch angle, respectively. It is clear, that the neural model reflects real system satisfactorily.

Moreover, the histogram depicted in Fig. 11 presents the modelling errors.
6. Conclusions
The main objective of this paper was to design and implement the SSNN tool, which allows to clearly identify any non-linear dynamic system with the state-space neural networks. The
motivation for design such a tool was the success of previously implemented by author of this paper GMDHToolbox. It proves that there is a need for automatic tool which can be used without any expertise in the NN field. The effectiveness of designed toolbox was verified with two experiments scenarios. In first, the obtained SSNN model was compared with the analytical model. In second one, the non-linear SS model was obtained based on the data gathered from the high-order wind turbine model. In both cases, the SSNN models reflects the real system in a very good way. Moreover, the models may be linearized and described in the classic state-space form, with the state-space matrices, not with the weights parameters which can be found in the other approaches. Such a form can be used in the FD and control design.

Designed in this paper toolbox is the first stage of realizing the grant *Artificial neural networks for robust fault diagnosis and control of non-linear systems*. The next task of this research is to extend the functionality of the toolbox by adding the possibility to identify non-linear systems in the quasi-LPV form without linearization. Such form is widely used in modern fault-tolerant controls schemes [11, 18].

6.1. Acknowledgments
The work was supported by the National Science Centre of Poland under grants: UMO-2014/15/N/ST7/00749 and UMO-2012/07/N/ST7/03316.

References
[1] F.D. Bianchi, H. De Battista, and R.J. Mantz. *Wind turbine control systems: principles, modelling and gain scheduling design*. Advances in industrial control. Springer, 2007.
[2] S. Chen and P. Billings, S.and Grant. Non-linear system identification using neural networks. *International journal of control*, 51(6):1191–1214, 1990.
[3] S.R. Chu, R. Shoureshi, and M. Tenorio. Neural networks for system identification. *Control Systems Magazine, IEEE*, 10(3):31–35, 1990.
Figure 10. Results of the neural network modelling for: a) torsion angle $\tau$, b) rotor velocity $\omega_r$, c) generator velocity $\omega_g$, d) pitch angle $\beta$

[4] A. Czajkowski and K. Patan. Model predictive control of the two rotor aero-dynamical system using state space neural networks with delays. In Intelligent Systems in Technical and Medical Diagnostics, volume 230, pages 113–124. Springer Berlin Heidelberg, 2014.

[5] A. Czajkowski, K. Patan, and M. Szymański. Application of the state space neural network to the fault tolerant control system of the plc-controlled laboratory stand. Engineering Applications of Artificial Intelligence, 30:168–178, 2014.

[6] E. B. Kosmatopoulos, M. M. Polycarpou, M. Christodoulou, P. Ioannou, et al. High-order neural network structures for identification of dynamical systems. Neural Networks, IEEE Transactions on, 6(2):422–431, 1995.

[7] Dziekan L., Witczak M., and J. Korbicz. Active fault-tolerant control design for takagi-sugeno fuzzy systems. Bulletin of the Polish Academy of Sciences-Technical Sciences, 59(1):93–102, 2011.

[8] G. P. Liu. Nonlinear identification and control: a neural network approach. Springer Science & Business Media, 2012.

[9] M. Luzar, A. Czajkowski, M. Witczak, and M. Mrugalski. Actuators ans sensors fault diagnosis with dynamic, state-space neural networks. In Methods and Models in Automation and Robotics - MMAR 2012 : Proceedings of the 17th IEEE International Conference, pages 196–201, 2012.

[10] M. Luzar, L. Sobolewski, W. Miczulski, and J. Korbicz. Prediction of corrections for the polish time scale UTC(PL) using artificial neural networks. Bulletin of the Polish Academy of Sciences: Technical Sciences, 61(3):589–594, 2013.

[11] M. Luzar and Witczak. Robust MPC for a non-linear system - a neural network approach. Journal of Physics: Conference Series, 570(Article no: 032002), 2014.

[12] M. Luzar and M. Witczak. A GMDH toolbox for neural network-based modelling. https://www.researchgate.net/publication/235901341_A_GMDH_TOOLBOX_FOR_NEURAL_NETWORK-BASEDMODELLING, 2010. [Online; accessed October-2010].
Figure 11. Neural network modelling errors for: a) torsion angle $\tau$, b) rotor velocity $\omega_r$, c) generator velocity $\omega_g$, d) pitch angle $\beta$

[13] M. Luzar, M. Witczak, and C. Aubrun. Robust $H_\infty$ sensor fault diagnosis with neural networks. In *Intelligent Systems in Technical and Medical Diagnosis*. (Korbicz J., Kowal M., (Eds.)), pages 125–136. Springer, Berlin, 2013.

[14] M. Luzar, M. Witczak, and P. Witczak. Neural-network based robust FTC: Application to wind turbines. In *Artificial Intelligence and Soft Computing*. (Rutkowski L. et. all, (Eds.)), pages 97–108. Springer, Berlin, 2014.

[15] M. Luzar, M. Witczak, P. Witczak, and C. Auburn. Neural-network based robust predictive fault-tolerant control for multi-tank system. In *Proc. of the 13th European Control Conference (ECC 2014)*, pages 276–281, Strasbourg, 2014.

[16] M. Nørgaard, O. Ravn, N.K. Poulsen, and L.K. Hansen. *Neural Networks for Modelling and Control of Dynamic Systems*. Springer-Verlag, London, 2014.

[17] K. Patan. *Artificial neural networks for the modelling and fault diagnosis of technical processes*. Springer-Verlag, Berlin, 2008.

[18] T. Peni, B. Vanek, Z. Szabo, and J. Bokor. Supervisory fault tolerant control of the GTM UAV using LPV methods. *International Journal of Applied Mathematics and Computer Science*, 25(1):117–131, 2015.

[19] D. Rotondo, F. Nejjari, and V. Puig. Robust quasi-lpv model reference ftc of a quadrotor uav subject to actuator faults. *International Journal of Applied Mathematics and Computer Science*, 25(1):7–22, 2015.

[20] J. Suykens, J. Vandewalle, and B. de Moor. *Artificial neural networks for modelling and control of non-linear systems*. Springer Science & Business Media, 2012.

[21] M. Witczak, V. Puig, and S. Montes De Oca. A fault-tolerant control strategy for non-linear discrete-time systems: application to the twin-rotor system. *International Journal of Control*, 86(10):1788–1799, 2013.