Input Disturbance Rejection Using Inverse Plant Model Generated Using Artificial Neural Network: A Case Study For DC Electric Motor Under Load

Yusuf Çağrı Öğüt
Graduate School of Science
Department of Computer Engineering
Hacettepe University
Ankara, Turkiye
yusuf.ogut@hacettepe.edu.tr

***Abstract*—In this paper, inverse and forward plant models are generated using ANN and their performances are evaluated through controllers and comparison.**

***Index Terms*—artificial neural networks, control, disturbance, inverse model**

I. Introduction

Artificial neural networks (ANNs) have found widespread use in numerous fields of science and engineering, with a growing emphasis on control systems design. Some common applications include the use of neurocontrollers, fine-tuning of existing controllers and system identification.

System identification is an important study in control systems design. Approaches for system identification mainly grouped in three categories: *White-box Approach*, *Gray-box Approach* and *Black-box Approach*.

In White-box Approach, detailed dynamic model of the system is created using methods such as Lagrangian. Using input/output data pairs which are collected from the real system are used as training data for adjusting model parameters with a cost function such as least-squares.

Gray-box Approach is used in cases where system dynamics are not known/unreachable, but designer can assign a transfer function regarding the need of model detail and observed system behaviour, such as overshoot. Created transfer function's parameters are estimated using collected input/output data pairs in sense of damping, natural frequency and damping factor.

In Black-box Approach, there is no baseline model for the plant, only input/output data pairs collected from the real plant. It is not achievable to create a convenient system model using classical analytical or statistical techniques. For this purpose, ANNs are very promising.

Disturbances and system uncertainties are inevitable aspects of real systems. There are many methodologies developed for suppressing them. Inverse plant models are widely used for this problem [3].

In this paper, an inverse system model is generated using ANN and this model is used for input disturbance rejection application on a DC Electric Motor under load.

ANN training script for the task is implemented using Matlab. Training data is generated using Simulink model of the system, which consists of a DC Electric Motor under load.

As an extra study, forward model of the system generated and compared with mathematical model.

II. Related Works

In recent years, ANNs found variety of application in control theory [1], [7], [8]. In [7], authors proposed a neurocontroller for an optimal tracking problem.

The applications of neural networks in system identification are extensive, with the ability of neural networks to adapt nonlinear systems, they are suitable for system identification problems. There are many studies in literature for the topic, in [1] and [9], authors investigated different aspects of neural network training for the purpose.

Input Disturbance Rejection (IDR) is a widely studied topic of control theory [5], [6]. In [5], authors provided a guideline for disturbance observers, in which using inverse plant model for the purpose is explained.

III. System Description

The system under consideration on this paper is a DC Electric Motor under load. Load consists two different inertias with damping, attached to motor shaft through a gearbox. System will be referred as *Plant* from now on. Following is the Simulink model of the plant:

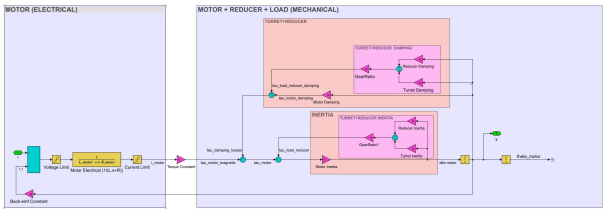

Fig. 1. Plant Model

Following subsection gives brief explanations for each model element.

A. Model Elements

1) DC Electric Motor and Load: DC Electric Motor in this study is a Brushless DC Motor (BLDC Motor). Specifications of the motors can be found in Fig. 2., where B_{load} is load damping, B_{motor} is motor damping, $B_{reducer}$ is reducer damping, $GearRatio$ is ratio of reducer's output to input, J_{load} is load inertia, J_{motor} is motor's rotor inertia, $J_{reducer}$ is reducer inertia, L_{motor} is motor phase's inductance, R_{motor} is motor phase's end-to-end resistance, k_e is motor's back-emf constant and k_t is motor's torque constant. Note that dampings are in unit of $N*m/(rad/s)$, inertias are in unit of $kg*m^2$, motor constant in unit of Nm/a and back-emf constant in unit of $V/(rad/sec)$.

Name	Status	Value	DataType
B_load		0.0047	double (auto)
B_motor		0.00025	double (auto)
B_reducer		0.001	double (auto)
GearRatio		1	double (auto)
J_load		0.0016118	double (auto)
J_motor		0.0002548	double (auto)
J_reducer		1E-5	double (auto)
L_motor		6E-6	double (auto)
R_motor		0.015	double (auto)
ke		0.039313695	double (auto)
kt		0.0378	double (auto)

Fig. 2. Plant Parameters

Dynamic modeling of BLDC motors can be done in a wide range of detail [2]. Fundamental methods use motor constants (torque constant, back-emf constant) in order to model electromagnetic, electrical and mechanical relations between stator and rotor. A more detailed mathematical model can be constructed by directly implementing electromagnetic equations. Nevertheless, using Simscape libraries of Mathworks enables the most precise models. This can be done using fundamental cyber-physical blocks such as resistors, inductors, resistance, inertia and damping. Also, there are direct models for BLDC motors and mechanical elements in Simscape libraries.
In this study, BLDC motor is modeled mathematically (Fig. 1.) using motor constants mentioned above. For the load side, inertias and dampings are modeled.

2) Controller: For simulations and performance comparison, a simple PI controller with output saturation is implemented. Proportional Gain is 1, Integral Gain is 0.12 and controller output is saturated at limits [-28,28]. Architecture can be seen Fig. 3.

IV. Training ANN for Inverse Plant Dynamics

ANNs come in variety of architectures and training methodologies. In this work, a feed-forward architecture

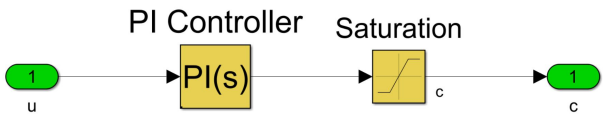

Fig. 3. Controller

ANN is trained using *error back propagation* with difference between output and estimated output as cost function. As to the different scaling of ANN inputs and outputs, a normalization is conducted as suggested in [4]. Mentioned ANN architecture is illustrated in Fig.4.

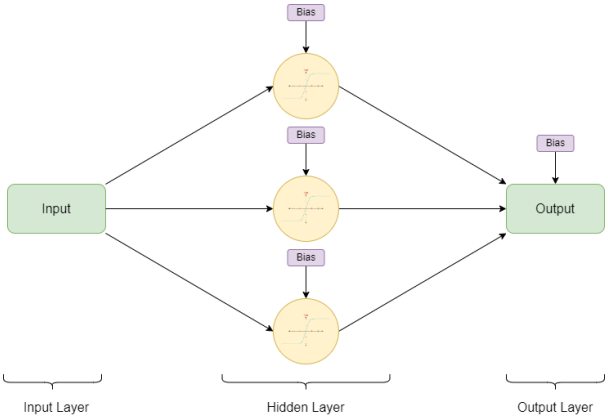

Fig. 4. ANN Architecture

As a generic calculation; inputs and output of the input layer is O_i^0, weights from input layer to hidden layer units are ${W_{ij}}^0$, sum for each hidden layer unit is S_i^0, outputs of hidden layer units are O_i^1, weights from hidden layer to output layer units are ${W_{ij}}^1$, sum at the output is S_i^1 and output of the network is O_i^2. Regarding this convention, following is a basic formulation:

$$O_i^0 = [i; Bias] \tag{1}$$

$$S_i^0 = O_i^0 * [{W_{ij}}^0] \tag{2}$$

$$O_i^1 = tanh(S_i^0) \tag{3}$$

$$O_i^2 = S_i^1 = [O_i^1; Bias] * {W_{ij}}^1 \tag{4}$$

Training data is a sine wave of $Amplitude(A) = 400$ and $Frequency(f) = 1rad/sec$ and applied to the plant as excitation. Collected plant output and controller output are used as training pair (Fig. 5.). Note that all training data are collected in 100 Hz.

For different training parameters (learning rate, number of epochs and number of hidden layer units), ANN exhibits different convergence behaviours. In this study, effects of changing these parameters are evaluated.

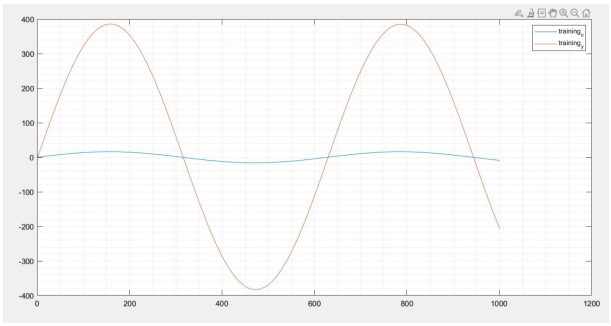

Fig. 5. Training data for inverse plant model training

Firstly, different learning rates (0.1, 0.05 and 0.02) are tried. Number of hidden layer units is 3 and number of epochs is 15. Result can be seen in Fig. 6.

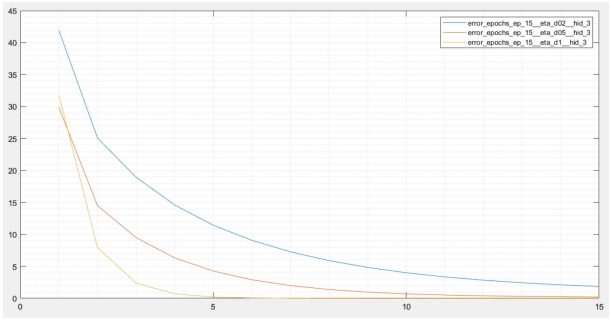

Fig. 6. Training with different learning rates

As it can be seen, epoch error decreases with increasing of learning rate. As epoch error of -0.035 is small enough, learning rate 0.1 chosen.
Following that, effect of changing number of epochs (10, 15 and 30) is investigated. Number of hidden layer units is 3 and learning rate is 0.1. Result can be seen in Fig. 7.

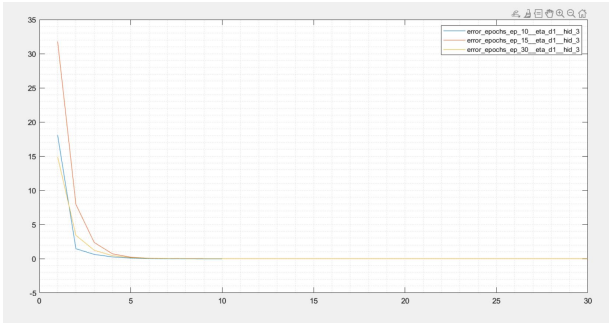

Fig. 7. Training with different epoch numbers

It can be seen that network converges for each 3 of number of epochs. This training is not computationally very costly, so number of epochs = 30 is chosen as slightly small number of error is achieved (-0.0022) at the end.

Lastly, effects of changing number of hidden layer units (3, 5 and 10) is investigated. Number of epochs is 30 and learning rate is 0.1. Result can be seen in Fig. 8.

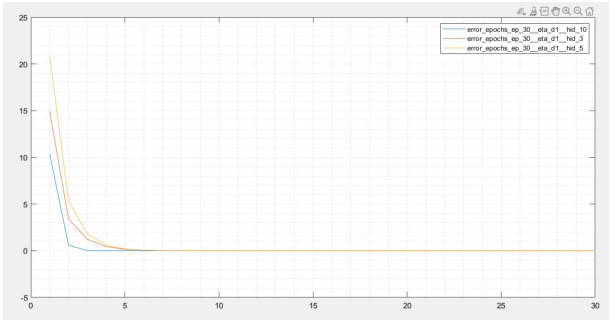

Fig. 8. Training with different number of hidden layer units

Error is small for each number of hidden layer units, but considering real time computational cost for larger number of hidden layer units, number of hidden layer neuron = 3 is chosen.

V. Integrating Inverse Plant For Input Disturbance Rejection Control

Input Disturbance Rejection is a control technique which is used for increasing regulator characteristics of a control system against disturbances which enter through control channel. Controller consists of inverse dynamic model of the plant and an *Observer* (which is a low-pass filter) [3]. Fig. 9. shows a generic architecture.

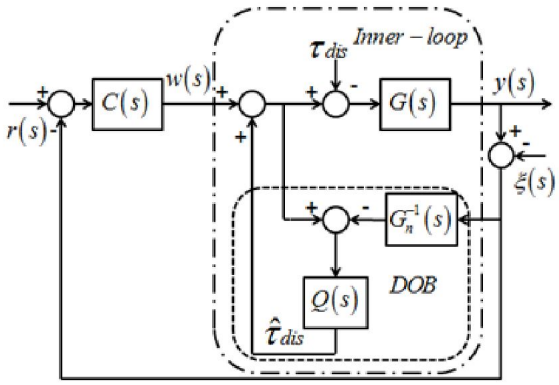

Fig. 9. Input Disturbance Rejection [3]

In Fig. 10., designed IDR control system for this paper can be seen.

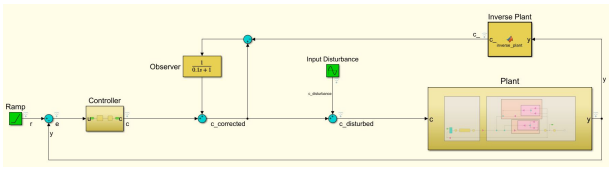

Fig. 10. Control System

Simulation duration is 10 seconds. A ramp input with $slope = 30$ is used as reference in order to investigate tracking characteristics of closed-loop system. A sinusoidal disturbance with $Amplitude = 1$ and $Frequency = 1rad/sec$ is used (Fig. 11.).

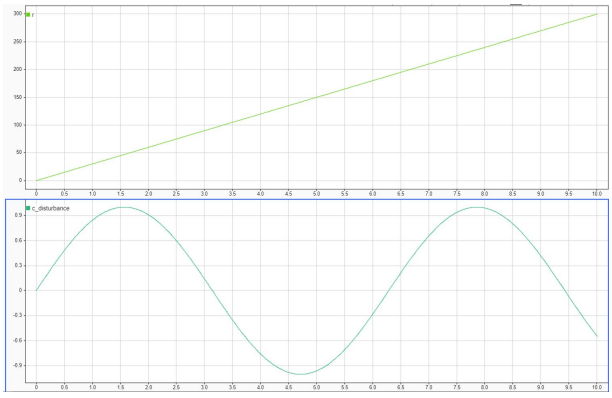

Fig. 11. Ramp reference input (r) (top) and sinusoidal disturbance (bottom)

First, closed-loop tracking performance without *IDR* and *disturbance* is investigated (Fig. 12.).

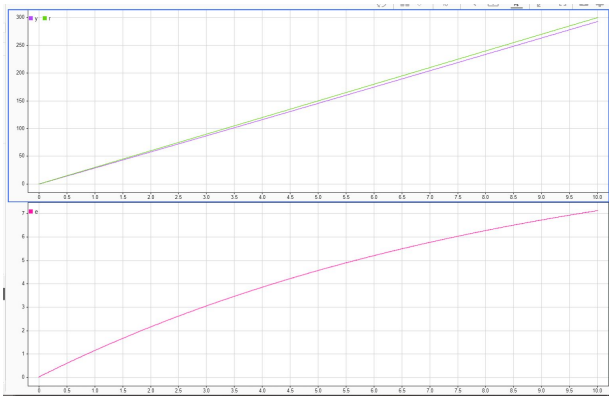

Fig. 12. Reference (r) and Output (y) (top) and error (bottom) without *diturbance* and *IDR*

Result of adding input disturbance to model investigated at Fig. 13.
As it can be seen, disturbance in control channel could not be compensated by controller and it is observed in *error*. To regulate this disturbance, *IDR* is added to model and results can be seen in Fig. 14.

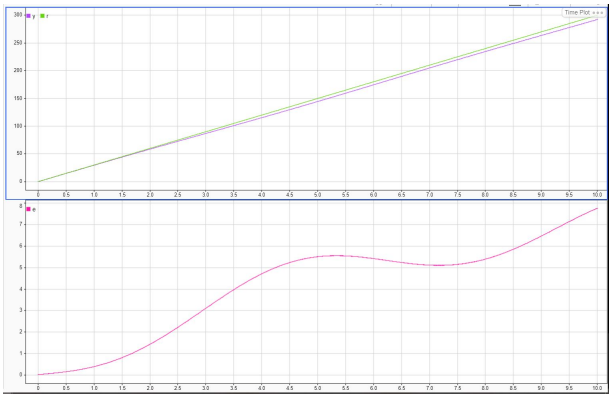

Fig. 13. Reference (r) and Output (y) (top) and error (bottom) with *diturbance* and without *IDR*

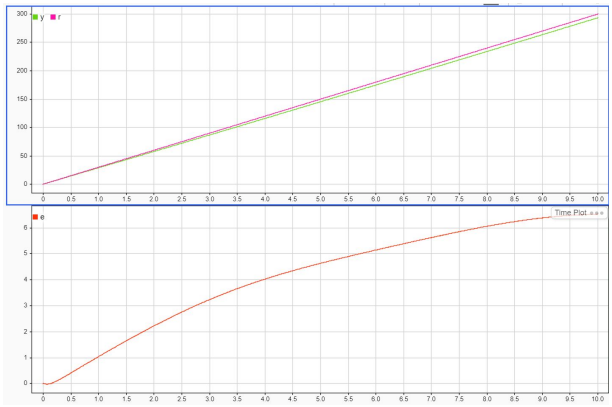

Fig. 14. Reference (r) and Output (y) (top) and error (bottom) with *diturbance* and *IDR*

It can be noticed that *IDR* regulated the disturbance from output.

VI. Training ANN for Forward Plant Dynamics For Comparison

As an extra study, a forward plant model is generated using ANN and techniques mentioned on *Section IV*.
Compared to study done for inverse plant dynamics, a more complex ANN training executed. For this training; configuration is 4 inputs, 10 hidden layer units and 1 output. Inputs are current time step controller output, one step previous controller output, one step previous plant output and two step previous plant output. Output is current time step output. ANN architecture can be seen in Fig. 15.
Training input is same with the one used for inverse plant (See Section IV). Training data can be seen in Fig. 16.
ANN is trained for 25 epochs and 0.01 learning rate. Epoch errors can be seen in Fig. 17.

As it can be seen, training error converges to a very small value, -0.0513.
To test the ANN-generated forward plant model, model is integrated to Simulink model in Fig. 9., updated architecture can be seen in Fig. 18.

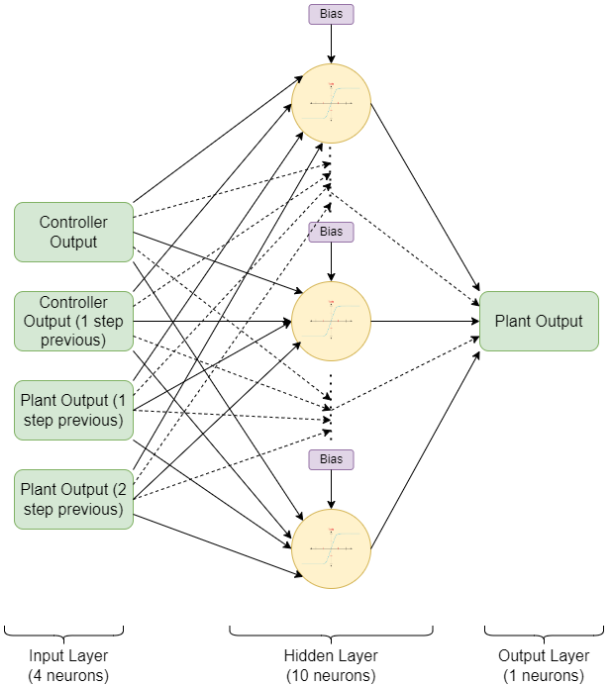

Fig. 15. ANN architecture for forward plant training

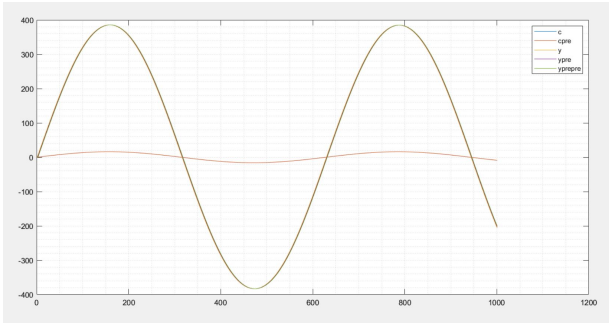

Fig. 16. Training data for forward plant

Forward Plant Model is tested with same ramp input used for inverse plant model (Fig. 10. (top)). Comparison between ANN-generated plant model and constructed mathematical model can be seen in Fig. 19.

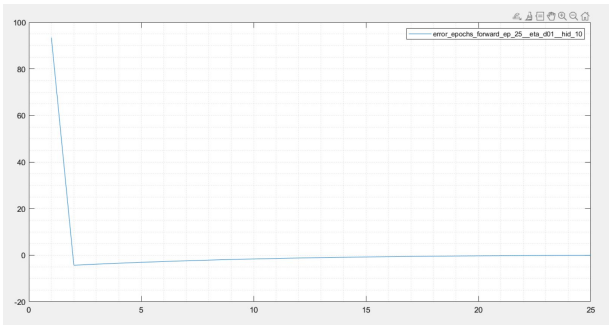

Fig. 17. Forward plant training

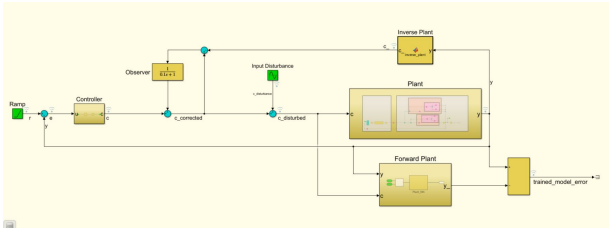

Fig. 18. Model extended with Forward Plant

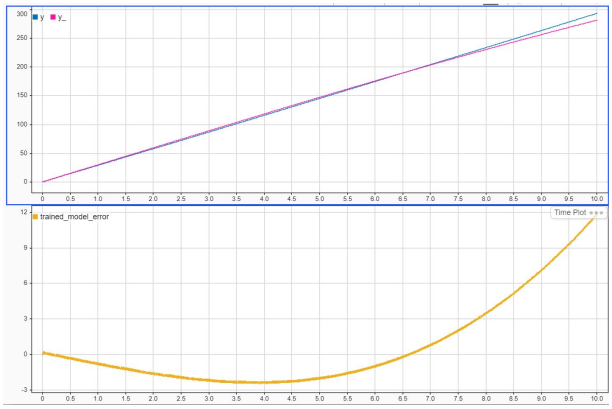

Fig. 19. Comparison between mathematical model and ANN-generated model, responses (top), difference (bottom)

As it can be seen, results are very close, as average error is 0.5925 with standard deviation 2.6047.

VII. Conclusion and Discussions

In this study, an inverse dynamic model of plant is obtained using ANN and this model is used for IDR Control for a DC Electric Motor under load. Plant model created in Simulink contained numerous nonlinearities, which would be hard to model analytically. Results are promising for case study which this paper deals.

Additionally, forward plant model is generated using ANN it's performance is compared with mathematical model's. Result is fair enough for using in simulations or model-based controllers.

VIII. Future Works

As further study, usability of ANN-generated forward and inverse plant models for different types of disturbances can be investigated. Also generated forward plant model can be used in model-based control applications.

References

[1] Efe, M. Ö. (1996). Identification and control of nonlinear dynamical systems using neural networks (Master's thesis, Fen Bilimleri Enstitüsü).

[2] P. Kazmierkowski, M. (2004). Electric motor drives: Modeling, analysis and control, R. Krishan, Prentice-Hall, Upper Saddle River, NJ, 2001, xxviii+ 626 pp. ISBN 0-13-0910147.

[3] Sarıyıldız, E., Ohnishi, K. (2014). A Guide to Design Disturbance Observer. Journal of Dynamic Systems, Measurement and Control. https://doi.org/10.1115/1.4025801

[4] M. Önder Efe, CMP 684 Lecture Notes

[5] Ohishi, K., Ohnishi, K., and Miyachi, K., 1983, "Torque-speed regulation of dc motor based on load torque estimation," in Proc. IEEJ IPEC-TOKYO, 2, pp. 1209– 1216

[6] S. Mukhopadhyay and K. S. Narendra, "Disturbance rejection in nonlinear systems using neural networks," [1992] Proceedings of the 31st IEEE Conference on Decision and Control, 1992, pp. 2696-2701 vol.3, doi: 10.1109/CDC.1992.371328.

[7] Young-Moon Park, Myeon-Song Choi and K. Y. Lee, "An optimal tracking neuro-controller for nonlinear dynamic systems," in IEEE Transactions on Neural Networks, vol. 7, no. 5, pp. 1099-1110, Sept. 1996, doi: 10.1109/72.536307

[8] Y. Yildiz, A. Sabanovic and K. Abidi, "Sliding-Mode Neuro-Controller for Uncertain Systems," in IEEE Transactions on Industrial Electronics, vol. 54, no. 3, pp. 1676-1685, June 2007, doi: 10.1109/TIE.2007.894719.

[9] Efe, M. Ö., Kaynak, O. (1999). A comparative study of neural network structures in identification of nonlinear systems. Mechatronics, 9(3), 287-300. https://doi.org/10.1016/S0957-4158(98)00047-6