Proceedings of the 21st International Conference on Global Research and Education (Inter-Academia 2024)

Neural Network Based on Direct Inverse Control for Electro–Hydraulic Servo Drive

A. Winnicki* and B. Guś

Faculty of Mechatronics, Institute of Automatic Control and Robotics, Warsaw University of Technology, św. A. Boboli 8, 02-525 Warsaw, Poland

Doi: 10.12693/APhysPolA.146.406 *e-mail: arkadiusz.winnicki@pw.edu.pl

This paper describes the use of the nonlinear autoregressive with exogenous inputs neural network for the control of electro–hydraulic servo drive. The direct inverse controller was trained on a real object in an online way with the use of a programmable logic controller before starting work on a real object, and appropriate tests were performed in the MATLAB/Simulink environment in order to select the right structure of the neural network. Also, the paper includes various network learning algorithms: gradient descent, resilient backpropagation, and adaptive moment estimation, which belong to backpropagation algorithms. In order to compare the suitability of the direct inverse controller, the controller was implemented on a real electro–hudraulic test stand, and its performance was compared with the performance of a proportional–integral–derivative controller.

topics: artificial neural network (ANN), control engineering, direct inverse control, electro–hydraulic servo drive

1. Introduction

The it proportional–integral–derivative (PID) control algorithm is the most frequently used controller in industrial applications worldwide. The universality of PID controllers is due to the small number of parameters to manipulate, i.e., those related to proportional, integral, and derivative actions, which facilitate the operation of this type of controller. Additionally, there are various methods for selecting these parameters, such as the Ziegler–Nichols method. Unfortunately, for constant parameter values, the PID control algorithm operates correctly only at specific operating points.

Due to these limitations, alternative control techniques are being explored that can achieve satisfactory results even for a wider operating range of nonlinear systems. One such approach is *direct inverse control* (DIC) [1–4], which is based on *artificial neural networks* (ANN). The ability of neural networks to adapt to varying conditions gives them a significant advantage over PID controllers.

Neural networks can be trained using three techniques: supervised, unsupervised, and reinforcement learning. Supervised learning, or learning with a teacher, adjusts the network parameters to match reference values. Unsupervised learning does not involve a teacher to guide the process. Reinforcement learning focuses on determining the network's actions to maximize long-term rewards and is commonly used in direct inverse control.

This paper is structured as follows: Sect. 2 describes the basic concepts related to neural networks; Sect. 3 presents information about direct inverse control; Sect. 4 discusses the general assumptions and simulation results; Sect. 5 describes the laboratory test setup; Sect. 6 presents the control results of DIC implemented on an electro–hydraulic servo drive; and Sect. 7 provides conclusions from the implementation of direct inverse control.

2. Neural networks

Neural networks are composed of individual neurons (see Fig. 1) that form layers, enabling the solution of complex problems [5]. Typically, these layers include input, hidden, and output layers. This structure is characteristic of one of the most popular neural networks, namely multilayer perceptron (MLP).

The first layer of the neural network, i.e., the input layer, receives the input values, with the number of neurons corresponding to the number of input variables. Next comes the hidden layer, where the calculations are performed by multiplying the input

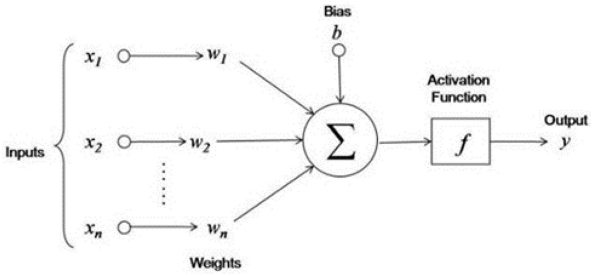

Fig. 1. Construction of a single neuron.

values by weights and adding a bias. The result is then passed through an activation function to produce the output for the following layer. The final layer, i.e., the output layer, generates the network's results, with the number of neurons in this layer depending on the number of output parameters.

The operation of MLP can be described as a function

$$y = f(x, \theta), \tag{1}$$

where y denotes outputs of neural networks, x — inputs of neural networks, θ — network weights, which were determined during the network training process.

Due to the presence of many layers of this type of network in the construction, this function can be presented as

$$f(x,\theta) = f^{(n)}(\dots f^2\left(f^1\left(x_1,\theta_1\right),\theta_2\right)\dots,\theta_n), \tag{2}$$

where n is the number of all network layers.

In the simplest case, when we deal with only a single perceptron, function (1) takes the form (see Fig. 1)

$$y = f\left(\sum_{i=1}^{n} w_i\, x_i + b\right), \tag{3}$$

where w_i is matrix of the neural network weights, b — bias, f — activation function.

We used the *nonlinear autoregressive with exogenous inputs* (NARX) network — a *recurrent neural network* (RNN) with bidirectional information flow. It computes outputs based on current inputs and previous outputs, creating at least one feedback loop (see Fig. 2).

3. Concept of direct inverse control

The concept behind DIC is to exert a direct influence on the system (see Fig. 3). This type of controller uses the inverse function of the equation describing the system's behavior [1–4]. The equation used to describe system behaviors is the following

$$y(t) = F\big\{y(t), y(t-1), y(t-2), \dots, y(t-k), u(t),$$
$$u(t-1), u(t-2), \dots, u(t-k)\big\}. \tag{4}$$

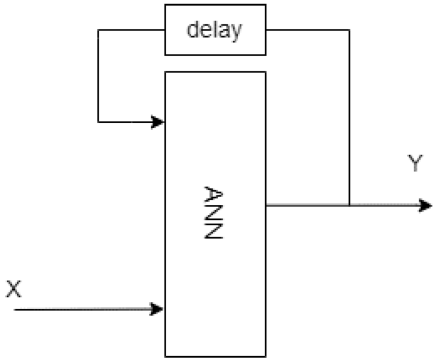

Fig. 2. The idea of RNN.

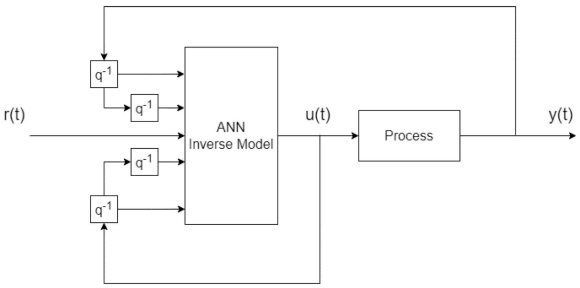

Fig. 3. The idea of DIC.

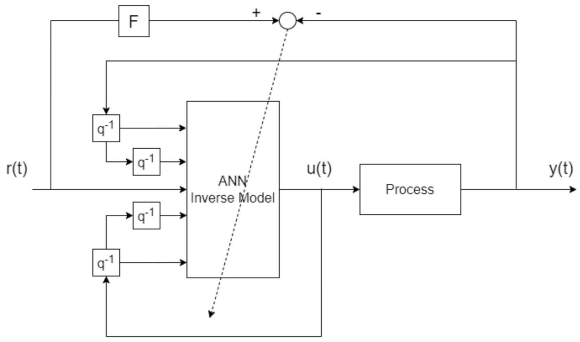

Fig. 4. The idea of training.

The mathematical model of the controller can be described as

$$u(t) = F^{-1}\big\{y(t), y(t-1), y(t-2), \dots, y(t-k), u(t),$$
$$u(t-1), u(t-2), \dots, u(t-k)\big\}. \tag{5}$$

The inverse dynamics (5), when combined with the system's dynamics (4), cancel each other out, so the input to the controller should match the output of the system.

The new weights of the neural network were calculated based on the difference between the reference signal and the process output (see Fig. 4).

In (4)–(5), F is a filter and will be discussed in Sect. 4.

4. General assumptions

Various neural controller structures were tested, differing in input signals, the number of hidden layer neurons, and learning methods. Each network had one hidden layer and was tested with input signal structures $\langle 0 : 2.0 \rangle$, $\langle 0 : 2.1 \rangle$, and $\langle 0 : 3.1 \rangle$. For each configuration, 5, 10, 15, and 20 hidden neurons were used. The activation function was tanh, with a linear function in the output layer. Three learning algorithms — *gradient descent* (GD), resilient backpropagation (RPROP), and adaptive moment estimation (ADAM) [6, 7] — were applied three times per configuration to minimize initial weight variation. Performance was measured by *mean squared error* (MSE), stopping when MSE dropped below 10^{-16}. Tables I–III list the learning parameters used in the simulations.

The assessment of the effectiveness of DIC operations was based on integral indicators of regulation quality

$$I_{\mathrm{IAE}} = \int_0^\infty \mathrm{d}t \; |e(t)|, \tag{6}$$

$$I_{\mathrm{ISE}} = \int_0^\infty \mathrm{d}t \; e^2(t), \tag{7}$$

$$I_{\mathrm{ITAE}} = \int_0^\infty \mathrm{d}t \; t\,|e(t)|. \tag{8}$$

In addition, a filter with a transmittance of $\frac{1}{0.1s+1}$ was applied before calculating the difference between the reference signal and the process output. This filter reduces the abrupt changes in the forcing signal, which affects the gradient value required to update the weights.

The training signal was a sinusoidal signal with an amplitude of 50 mm and a frequency of $\frac{\pi}{4}$ rad/s. The variation of this signal is smoother than that of a square signal, resulting in less abrupt changes in the weight values (see Fig. 5). Before being input to DIC, the signal values were scaled to the range $[-1, 1]$.

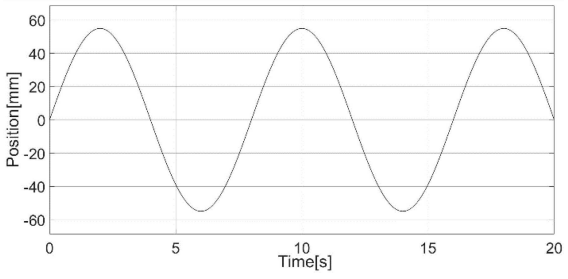

Fig. 5. Learning signal.

TABLE I

Learning parameters of GD algorithm. Here, learning rate is a scalar value that sets the step size for each iteration in the optimization process.

Parameter	Value
learning rate	5×10^{-5}

TABLE II

Learning parameters of RPROP, where: $\Delta_{\max}$ — single value that sets the maximum allowable change in weights during an update; $\Delta_{\min}$ — single value that establishes the minimum change in weights to prevent stagnation; η^+ — scalar value that determines how much the weight change should be increased when the previous update is successful; η^- — scalar value that indicates how much the weight change should be decreased if the previous update is not successful.

Parameter	Value
$\Delta_{\max}$	100
$\Delta_{\min}$	10^{-5}
η^+	1.0002
η^-	0.0002

TABLE III

Learning parameters of ADAM, where: learning rate — scalar value that sets the step size for each iteration in the optimization process; β_1 — scalar value representing the exponential decay rate for the first moment estimate, which helps to stabilize the optimization process; β_2 — scalar value that indicates the exponential decay rate for the second moment estimate, allowing the learning rate to adapt based on the variance of past gradients; ϵ — small constant added to prevent division by zero during weight updates, ensuring numerical stability.

Parameter	Value
learning rate	4×10^{-6}
β_1	0.9
β_2	0.999
ϵ	10^{-20}

On the other hand, the assessment of the regulation indicators (panels (a)–(c) in Fig. 6) will be based on the following signals:

- A sinusoidal wave with an amplitude of 35 mm and a frequency of $\frac{\pi}{4}$ rad/s (see Fig. 6a);
- A square signal with values oscillating between 0 and 30 mm, changing every second (see Fig. 6b);
- A complex square-wave signal with value changes occurring every second (see Fig. 6c).

5. The results of simulation

Based on the studies described in Sect. 4, the most appropriate structure was selected for each method (see Table IV). The following configurations were used in Table IV:

- DIC I — learning algorithm: gradient descent, structure $\langle 0 : 3.1 \rangle$, neurons in hidden layer: 20, time of reaching MSE: 285 s;
- DIC II — learning algorithm: resilient propagation, structure $\langle 0 : 3.1 \rangle$, neurons in hidden layer: 5, time of reaching MSE: 247 s;
- DIC III — learning algorithm: adaptive moment estimation, structure $\langle 0 : 2.1 \rangle$, neurons in hidden layer: 10, time of reaching MSE: 480 s.

From Table IV, we see that the control quality indicators for DIC controllers are generally higher than those for the PID controller, indicating that the PID controller achieves more precise control. Although neural controllers were trained only on sinusoidal signals, they achieved a control quality similar to that of the PID, especially for square signals. Among the neural controllers, the one trained with the RPROP method had the lowest regulation indices and the shortest time (247 s) of reaching the MSE target. Therefore, the best network configuration may be the one trained with RPROP, as shown in Table III. Figure 7 shows the system's responses to the sinusoidal input for DIC, as described in Table IV, compared to those for the PID controller.

Results of simulation. TABLE IV

	IAE	ISE	ITAE
Quality indicators for sin wave			
DIC I	13.33	27.45	53.3
DIC II	13.35	27.51	53.33
DIC III	12.99	26.08	51.99
PID	11.78	21.42	49.45
Quality indicators for square signal			
DIC I	11.53	223.1	18.75
DIC II	11.49	222.1	18.76
DIC III	11.54	225.4	18.86
PID	11.38	223.8	18.76
Quality indicators for complex-square signal			
DIC I	24.92	317.2	172.5
DIC II	26.07	326.8	171.2
DIC III	26.43	339.2	174.5
PID	24.55	311	156.6

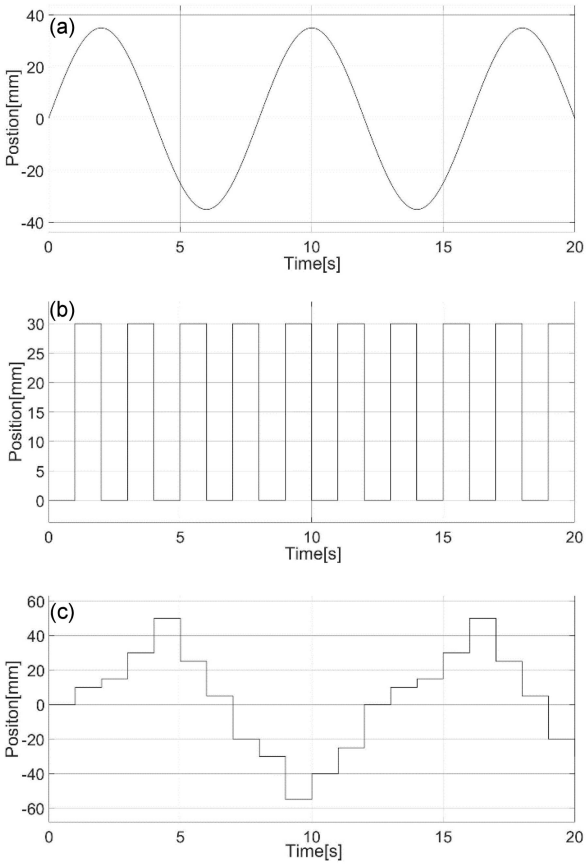

Fig. 6. (a) Sin wave, (b) square wave, (c) complex-square wave.

6. Laboratory station

The operation of the direct inverse controller was performed on electro–hydraulic test stand, whose structure is presented in Fig. 8. The laboratory system contains following main parts: hydraulic pump, pressure relief valve, electro–hydraulic servo valve, piston, linear position encoder, a *programmable logic controller* (PLC) and PC computer with MATLAB/Simulink software.

The laboratory test stand is shown in Fig. 9.

The system employs a double-acting actuator with double-side piston rod (#2). In order to stabilize the movement of the piston rod, the platform (4) is positioned on slideways within the range $\langle -50; +50 \rangle$ mm.

The position of the piston rod is changed by the servo valve (#1), controlled by the voltage signal in the range $\langle -10; +10 \rangle$ V. The position of the actuator piston rod is obtained by means of a magnetostrictive transducer (#3). The control algorithm and data transfer between the position transmitter and the servo valve is carried out using PLC (X20CP1586 from B&R). PLC was programmed using a PC computer with MATLAB/Simulink

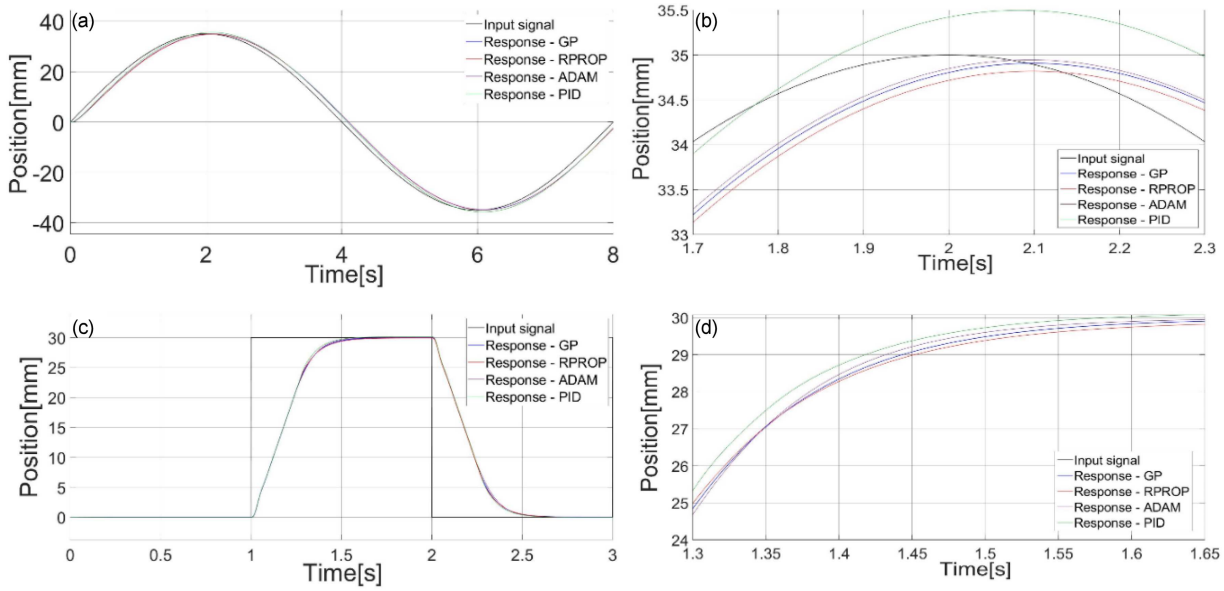

Fig. 7. (a) Sin-wave input, (b) zoom in 1.7–2.3 s: sin-wave input, (c) square-wave input wave, (d) zoom in 1.15–1.35 s: square-wave input.

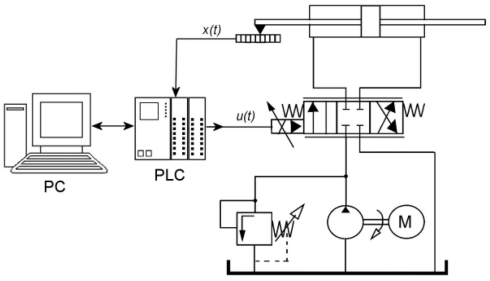

Fig. 8. Schematic diagram of the electro–hydraulic servo drive system with neural network direct inverse controller.

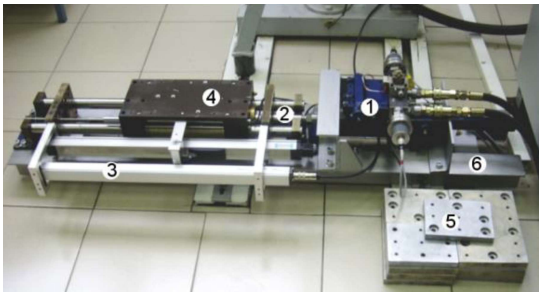

Fig. 9. Laboratory test stand: (1) servo valve, (2) piston, (3) position encoder, (4) load platform, (5) mass, (6) support.

software and Automation Studio (software from B&R for PLCs). A plug-in (Automation Studio Target for Simulink) from B&R was used to transfer the program from Simulink to Automation Studio.

TABLE V
Results of the experiment.

	IAE	ISE	ITAE
Quality indicators for sin wave			
DIC I	7.43	8.33	29.26
DIC II	7.29	8.00	29.01
DIC III	7.98	9.62	32.64
PID	5.43	4.51	22.97
Quality indicators for square signal			
DIC I	9.41	192.7	15.65
DIC II	9.66	194.7	16.29
DIC III	9.69	197.19	16.11
PID	9.45	195.9	15.65
Quality indicators for complex-square signal			
DIC I	20.18	291.25	130.26
DIC II	21.64	302.58	138.30
DIC III	21.52	298.08	139.01
PID	20.41	291.2	130.8

For the purpose of object modeling, the transmittance of the servo electro–hydraulic drive was determined by parametric identification. The object is usual described by the following transmittance

$$G(s) = \frac{\omega_0^2 k}{s\left(s^2 + 2\xi\omega_0 + \omega_0^2\right)}, \tag{9}$$

where k is amplification factor (87.761 mm/s), ξ — damping factor (0.37), ω_0 — natural vibration pulsation (93.742 rad/s).

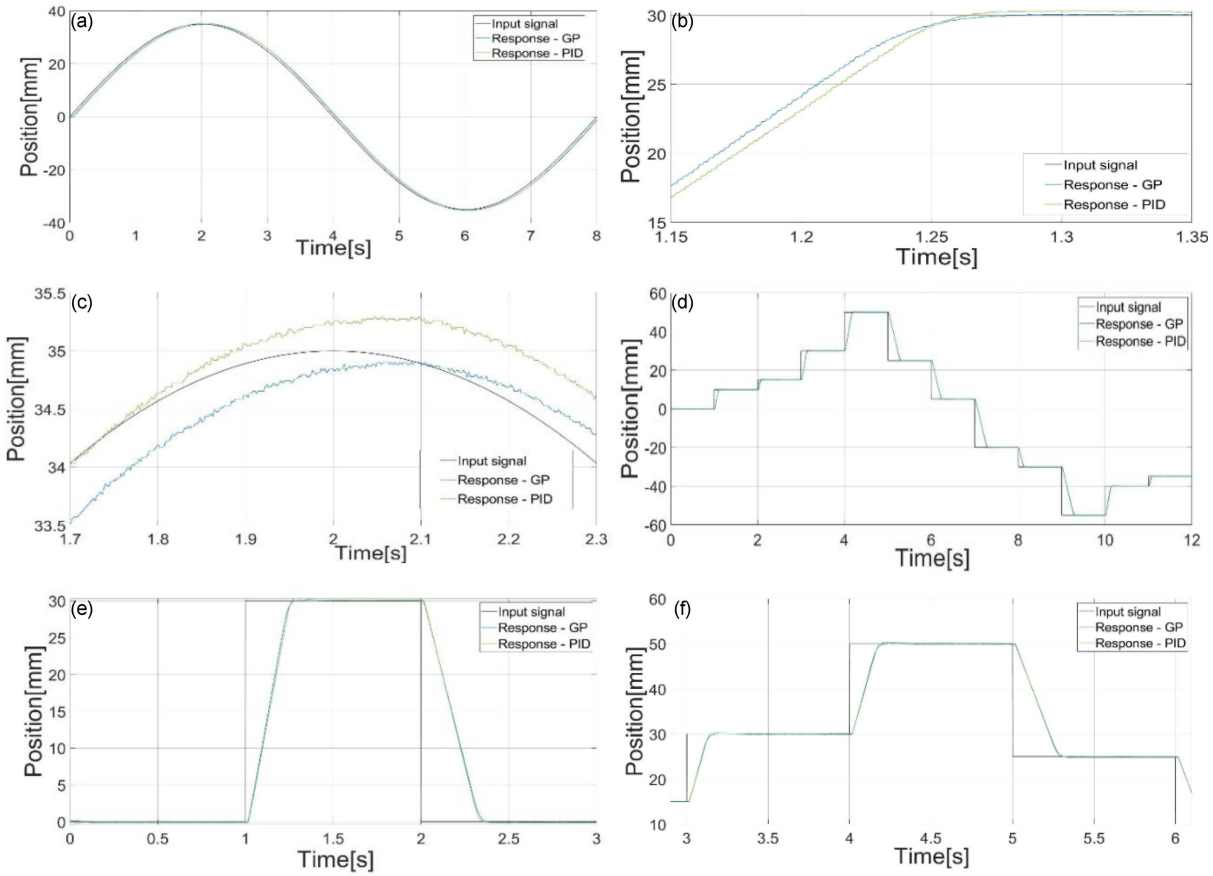

Fig. 10. (a) Sin-wave input, (b) zoom in 1.15–1.35 s: square-wave input, (c) zoom in 1.7–2.3 s: sin-wave input, (d) complex-square wave input, (e) square-wave input, (f) zoom in 2.9–6.1 s: complex-square wave input.

7. The results of the experiment on real object

This section presents the results of tests carried out on a real object [8, 9]. The structures of the artificial neural networks are the same as those presented in Sect. 4. The transmittance of the filter has been changed from $\frac{1}{0.1s+1}$ to $\frac{1}{0.05s+1}$, which is applied before calculating the difference between the set value and the measurement.

The new filter improved the quality indicators for both sinusoidal and square signals. However, the time to reach the target MSE error significantly increased when comparing Table IV to Table V, as the MSE value was fixed at 10^{-16} with a measuring resolution of 10^{-4}. Although the system operated with minimal error after approximately 400 s, the extended learning time may have contributed to slightly better quality indicators for square signals using neural controllers, compared to the PID controller (see Table V). Figure 10 shows the system responses using the GP algorithm, which achieved the most accurate control, alongside the PID controller, as described in Table V.

The following configurations were used in Table V:

- DIC I — learning algorithm: gradient descent, structure $\langle 0:3.1\rangle$, neurons in hidden layer: 20, time of reaching MSE: 3223 s;
- DIC II — learning algorithm: resilient propagation, structure $\langle 0:3.1\rangle$, neurons in hidden layer: 5, time of reaching MSE: 4644 s;
- DIC III — learning algorithm: adaptive moment estimation, structure $\langle 0:2.1\rangle$, neurons in hidden layer: 10, time of reaching MSE: 4381 s.

8. Conclusions

Based on this paper, we can conclude that direct inverse control can be successfully applied to control the electro–hydraulic servo drive. With an appropriate structure and learning duration, we can achieve even more precise control than with the most commonly used PID controller. The neural network controller utilized only one hidden layer, which was sufficient to control the electro–hydraulic

servo drive. To further advance the research, the number of layers in the neural network controller can be increased, and other activation functions can be implemented.

References

[1] S. Banerjee, A. Chandwani, A. Mallik, in: *2020 IEEE Int. Conf. on Power Electronics, Drives and Energy Systems (PEDES), Jaipur (India)*, 2020.

[2] M.T. Frye, R.S. Provence, in: *2014 IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC), San Diego (CA)*, 2014.

[3] G.C.M. De Abreu, R.L. Teixeira, J.F. Ribeiro, in: *Proc. 6th Brazilian Symp. on Neural Networks, Rio de Janeiro (Brazil)*, Vol. 1, 2000.

[4] A. Widaryanto, B. Kusumopotro, in: *2019 IEEE Int. Conf. on Innovative Research and Development (ICIRD), Jakarta (Indonesia)*, 2019.

[5] R. Tadusiewicz, Sieci Neuronowe, Akademicka Oficyna Wydawnicza RM, Warszaw 1993 (in Polish).

[6] P. Navneel, S. Rajeshni, P.L. Sunil, in: *2013 5th Int. Conf. on Computational Intelligence, Modelling and Simulation, Seoul, South Korea*, 2013, p. 29.

[7] S. Ruder, "An Overview of Gradient Descent Optimization Algorithms", 2016.

[8] B&R Industrial Automation, Automation Studio Target for Simulink, 2021.

[9] B&R Industrial Automation, Working with mappView, 2020.