Synthesis of neural network controllers for objects with non-linearity of the constraint type

S V Feofilov and D L Khapkin
Tula State University, 92, Lenin prospekt, Tula, 300012, Russia
E-mail: dima-hapkin@ya.ru

Abstract. The paper discusses the use of artificial neural networks of direct propagation to control objects with nonlinearities such as saturation and a rigid mechanical stop. The multilayer structure of a neural network is considered using the ReLU activation function and the Dropout layer, which allows working with the indicated nonlinearities. The possibility of training a neural network to control piecewise linear objects of a general type is demonstrated. Using the method of generalized inverse neurocontrol, a controller is synthesized that provides an acceptable quality of control in the presence of such nonlinearities. A specific example shows the possibility of creating a direct neuroemulator that repeats the dynamics of nonlinearities such as saturation and a rigid mechanical stop. The work of the neuroregulator in the examples is compared with the PID controller configured in Matlab. All theoretical conclusions are confirmed by numerical modeling of real technical systems with neural network control.

1. Introduction
Currently, one of the urgent problems is the synthesis of quality controllers for automatic systems. In this case, the control objects are usually nonlinear. This complicates the synthesis procedure due to the lack of regular methods. Moreover, the mathematical model may be incomplete or inaccurate. In these complicated cases the use of neural network controllers can be a solution. Artificial neural networks (ANNs) allow to create a digital model based on the collected statistical information about an object. Due to their features, ANNs can reproduce and control both linear and nonlinear control objects. Below we will consider the problem of creating and training a neural network controller for systems with constraints in the control object. Such objects are very common in engineering practice. For example, systems with mechanical stops (servo drives, electromagnets, etc.), or with saturation (by current, magnetic induction, etc.). The training of an ANN for its use as a controller is proposed using the method of generalized inverse control [1]. This will be considered on simple examples of objects with constraints. Previously, the authors considered the application of the generalized inverse control method for linear objects [2].

2. Mathematical model considered objects
Two types of elements with restraints can be distinguished in real objects. An element with a saturation-type constraint is described by a first-order differential equation (figure 1) and an element with a hard mechanical stop type constraint with a second-order differential equation (figure 2).
The element of the first type is described as follows

$$\dot{x} = \begin{cases} \frac{(ku - x)}{T}, & \text{if } |x| < D \text{ or } |x| = D \text{ and } (ku - x)T \cdot \text{sign} x \leq 0; \\ 0, & \text{if } |x| = D \text{ and } (ku - x)T \cdot \text{sign} x > 0. \end{cases}$$

The element of the second type is set

$$\dot{x}_1 = x_2,$$

$$\dot{x}_2 = \begin{cases} \frac{ku - x_1}{T} - \frac{2\xi x_2}{T}, & \text{if } |x_1| < D \text{ or } |x_1| = D \text{ and } (ku - x_1) \cdot \text{sign} x_1 \leq 0; \\ 0, & \text{if } |x_1| = D \text{ and } (ku - x_1) \cdot \text{sign} x_1 > 0. \end{cases}$$

It is assumed that the impact against the stop is completely inelastic, i.e., at each moment $t^*$ of entry to the limiter

$$x_1(t^* + 0) = x_1(t^* - 0),$$
$$x_2(t^* + 0) = 0.$$

The descent from the limiter is continuous.

As can be seen, an object containing limiters is a nonlinear system of a special type, the dynamics of which is described by differential equations with discontinuous right-hand sides, and phase trajectories can also be discontinuous. Such objects have a number of specific features, due to which the synthesis of controllers for them is a difficult task.

3. The structure of the ANN used

To work with such objects, it is proposed to use the structure (figure 3) of a multilayer neural network of direct propagation (multilayer perceptron, MLP) [3, 4].

When choosing an MLP structure to obtain a direct (figure 4) and inverse neuroemulator (figure 5), current and delayed data should be supplied to the input of the neural network [5, 6].

![Figure 3. General structure of a multilayer forward propagation neural network.](image)
Usually, an ANN is implemented on digital devices, and it is more convenient to consider an ANN as a discrete system for generating delayed data. Thus, the input of the ANN is the current data and the data from previous cycles. It follows from this that it is required to form data taking into account the required amount of previous data.

To train a neural network, first of all, it is necessary to collect data about the object, i.e., to form a training sample. For this purpose, it is suggested to feed the input of the object with a signal consisting of a harmonic part, then changing to a step signal and ending with a random signal. This allows to cover all possible modes of operation. In the process of this procedure, the signal parameters were changed randomly within acceptable limits. For a harmonic signal—frequency and amplitude. For step and random signals only amplitude. Further, from the received data, the training data for a neural network with the chosen number of delay lines is formed.

ReLU was chosen as the activation function [7, 8]. Its graph is shown in figure 6.

The ReLU function is described by the following equation: $g(x) = \max\{0, x\}$.

The advantage of this activation function is that it has no limits on the linear side, does not lose information about the magnitude of the value in the forward pass, and does not decrease the gradient values in the backward pass by the back-propagation method [9]. For this reason, ReLU is used in multilayer neural networks to train a large number of layers.

Also, when training an inverse neural controller, there are often learning difficulties associated with overfitting the network and hitting local minima. To overcome the problem of overfitting, a dropout layer [10] is used, which throws the outputs of some neurons during training, chosen randomly for each training example. The use of this layer allows to achieve a similar effect as the ensemble of neural networks [11, 12]. That is, one large neural network inside contains “subnets” that can perform the same functions and their results are averaged, thereby improving the final quality of the network [13]. Also, during training, not all weights are adjusted for a particular training example, which also prevents the network from learning the training sample. Thus, it allows to reduce the probability of overtraining appearance.

During training it is proposed to use stochastic gradient descent [3] with the Adam optimization method [14] to overcome some local minima.
4. Examples
As examples, consider an element with a saturation constraint and an element with a hard mechanical stop constraint. The parameters are shown in Table 1.

| Parameter | Element with a saturation-type restriction | Element with a hard mechanical stop type constraint |
|-----------|--------------------------------------------|---------------------------------------------------|
| Time sample ($T_s$) | 0.01 s | 0.01 s |
| $k$ | 1 | 1 |
| $T$ | 0.1 | 0.1 |
| $\xi$ | — | 0.5 |
| $D$ | ±0.5 | ±0.5 |

4.1. Example No. 1 with a direct neural emulator. Element with a saturation-type constraint
The structure of the neural network for the direct neural emulator is shown in Table 2.

The input layer receives one current and two previous control values, three previous element output values. The results of the trained network in comparison with the element with a saturation-type constraint are shown in Figure 7 and 8.

| Name layer | Number of neurons |
|-----------|------------------|
| 1 Input | 6 |
| 2 Linear (fully connected) + ReLU | 10 |
| 3 Output | 1 |

Figure 7. Response of an element with a saturation-type constraint and a direct neuroemulator to a harmonic signal, at which the constraint are not reached.

Figure 8. Response of an element with a saturation-type constraint and a direct neuroemulator to a harmonic signal, at which constraints are reached.
It can be seen that the neural network with the selected structure quite accurately repeats a simple nonlinear object.

4.2. Example No. 2 with a direct neural emulator. Element with a hard mechanical stop type constraint

The structure of the neural network in the direct neural emulator is the same as in example No. 1. The results of the trained network in comparison with the object, which is an element with a hard mechanical stop type constraint, are shown in figure 9 and 10.

![Figure 9](image)

**Figure 9.** Response of an element with a hard mechanical stop type constraint and a direct neuroemulator to a harmonic signal, in which the constraints are not reached.

![Figure 10](image)

**Figure 10.** Response of an element with a hard mechanical stop type constraint and a direct neuroemulator to a harmonic signal, at which the constraints are reached.

It can be seen that the training results are also good.

4.3. Example No. 3 with an inverse neural emulator. An element with a saturation-type constraint

The structure of the inverse neural network for controlling an element with a saturation-type constraint is presented in table 3.

| Table 3. Structure of the neural network for the neural controller with a saturation-type constraint element. |
|--------------------------------------------------|
| **Name layer** | **Number of neurons** |
| Input             | 6                      |
| Linear (Fully connected) + ReLU | 100                  |
| Linear (Fully connected) + ReLU | 40                   |
| Output            | 1                      |
For training, the current and three previous values of the object’s output, two previous values of the object’s input are fed to the input of the neural network.

The result of the obtained neural network controller in comparison with the PID controller synthesized using Matlab tools for the system with the object, which is an element with saturation-type restriction, is shown in figure 11 and 12.

**Figure 11.** Response of systems with neural network controller and PID controller for element with saturation-type constraint to harmonic signal with amplitude 0.3, at which the constraints are not reached.

**Figure 12.** Response of the system with neural network controller and PID controller for the element with saturation-type constraint to the harmonic signal of amplitude 1, at which the constraints are reached.

It can be seen that the resulting neural controller can successfully control an element with a saturation-type constraint well.

4.4. **Example No. 4 with an inverse neuroemulator. Element with a hard mechanical stop type constraint**

The neural network structure in an inverse neural emulator for controlling an element with a mechanical stop type constraint is descended in table 4.

For training, the current and four previous values of the object’s output are fed to the input of the neural network.

The result of the obtained neural network controller in comparison with the PID controller synthesized using Matlab tools for the system with the object, which is an element with saturation-type constraint, is shown in figure 13 and 14.

The resulting neural controller can adequately control an element with a mechanical stop type constraint.
Table 4. Description of the structure of the inverse neuroemulator for an element with a hard mechanical stop type constraint.

| Name layer | Number of neurons |
|------------|-------------------|
| 1 Input    | 5                 |
| 2 Linear (fully connected) + ReLU | 500         |
| 3 Dropout  | —                 |
| 4 Linear (fully connected) + ReLU | 50          |
| 5 Output   | 1                 |

Figure 13. Response of the system with neural network controller and PID re controller for an element with a hard mechanical stop type constraint to a harmonic signal with an amplitude of 0.4, at which the constraints are not reached.

Figure 14. Response of the system with neural network controller and PID controller for an element with a hard mechanical stop type constraint to a harmonic signal with an amplitude of 0.6, at which the constraints are reached.

5. Conclusions
The results of the study show that the neural network can be used as an object emulator and as a controller in control systems for nonlinear objects with saturation and hard mechanical stop constraints. Thus, there are good prospects of using neural controllers for arbitrary piecewise-linear objects of any order. At the same time, the main difficulties are the lack of regular methods for choosing the optimal neural network structure and the correct formation of the training sample.

Acknowledgments
The research was carried out under the grant of the Russian Foundation for Basic Research No. 20-38-90213.
References

[1] Psaltis D et al 1988 A multilayered neural network controller IEEE Control Systems Magazine 8 17–21
[2] Feofilov S V and Khapkin D L 2020 Basic synthesis of neural network controllers for following control systems News of the Tula state university. Technical sciences 11 256–61
[3] Goodfellow I, Bengio Y and Courville A 2016 Deep Learning (Cambridge, MA: MIT Press)
[4] Rumelhart D E and McClelland J L 1986 Parallel distributed processing: explorations in the microstructure of cognition (Cambridge, MA: MIT Press)
[5] Waibel A, Hanazawa T, Hinton G, Shikano K and Lang K J 1989 Phoneme recognition using time-delay neural networks Acoustics, Speech and Signal Processing IEEE Transactions 37 328–39
[6] Lang K J, Waibel A H and Hinton G E 1990 A Time-Delay Neural Network Architecture for Isolated Word Recognition Neural Networks 3 23–43
[7] Nair V and Hinton G 2010 Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair Proceedings of ICML 27 807–14
[8] Glorot X, Bordes A and Bengio Y 2011 Deep Sparse Rectifier Neural Networks Journal of Machine Learning Research 15 315–23
[9] Werbos P J 1974 Beyond regression: New tools for prediction and analysis in the behavioral sciences. Ph.D. thesis (Harvard University, Cambridge, MA)
[10] Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R 2014 Dropout: A simple way to prevent neural networks from overfitting Journal of Machine Learning Research 15 1929–58
[11] Szegedy C et al 2015 Going deeper with convolutions CVPR 1–9
[12] Huang G et al 2017 Snapshot Ensembles: Train 1, get M for free URL https://www.researchgate.net/publication/315765130_Snapshot_Ensembles_Train_1_get_M_for_free
[13] Breiman L 1994 Bagging predictors Machine Learning 24 123–40
[14] Kingma D P and Ba J 2014 Adam: A Method for Stochastic Optimization URL https://arxiv.org/pdf/1412.6980.pdf