IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 16, 2021, accepted January 6, 2022, date of publication January 18, 2022, date of current version January 21, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3143893

Neural Approximators for Variable-Order Fractional Calculus Operators (VO-FC)

BARTOSZ PUCHALSKI
Digital Technologies Center, Gdańsk University of Technology, 80-233 Gdańsk, Poland
Department of Intelligent Control and Decision Support Systems, Faculty of Electrical and Control Engineering, Gdańsk University of Technology, 80-233 Gdańsk, Poland

e-mail: bartosz.puchalski@pg.edu.pl

Financial support of these studies from Gdańsk University of Technology by the DEC-33/2020/IDUB/I.3.3 grant under the ARGENTUM – 'Excellence Initiative – Research University' program is gratefully acknowledged.

ABSTRACT The paper presents research on the approximation of variable-order fractional operators by recurrent neural networks. The research focuses on two basic variable-order fractional operators, i.e., integrator and differentiator. The study includes variations of the order of each fractional operator. The recurrent neural network architecture based on GRU (Gated Recurrent Unit) cells functioned as a neural approximation for selected fractional operators. The paper investigates the impact of the number of neurons in the hidden layer, treated as a hyperparameter, on the quality of modeling error. Training of the established recurrent neural network was performed on synthetic data sets. Data for training was prepared based on the modified Grünwald-Letnikov definition of variable-order fractional operators suitable for convenient numerical computing without memory effects. The research presented in this paper showed that recurrent network architecture based on GRU-type cells can satisfactorily approximate targeted simple yet functional variable-order fractional operators with minor modeling errors. In addition, the research also compares the presented solution with basic and recurrent neural networks that utilize Tapped Delay Lines (TDL) in their structure. The presented solution is a novel approach to the approximation of VO-FC operators. It has the advantage of automatic selection of neural approximator parameters by optimization based on data customized for specific requirements.

INDEX TERMS Approximation methods, fractional calculus, modeling, neural networks, recurrent neural networks.

I. INTRODUCTION

Fractional order calculus [1] has been known since the 17th century. This field of science concerns various methods that lead to a generalization of integration and differentiation operators to real or complex orders. In the last few decades, the interest in fractional-order calculus has grown in a very significant way in both the research and engineering fields, with special attention in modeling and simulation of physical phenomena as reported in [2] and [3]. Applications of the fractional order calculus can be found in areas such as astronomy [4], finances [5], time delay systems studies [6], fuzzy inference systems [7], nonlinear chaotic systems [8], unmanned aerial and ground vehicles [9], modeling of electrochemical capacitors [10], modeling and control of nuclear reactors [11], [12], and many others. Fractional operators used in the above exemplary research and engineering areas provide a specific way to generalize mathematical models by introducing an arbitrary order, for instance, denoted as α into the integro-differential fractional operator $\mathscr{D}^{\alpha}$, where $\alpha \in \mathbb{R}$ or in general case $\alpha \in \mathbb{C}$. In this paper, cases for which $\alpha \in \mathbb{R}$ will be considered.

By introducing an operator with such properties, in general, additional degrees of freedom are available, affecting the overall quality of models or algorithms that utilize them. The operator $\mathscr{D}^{\alpha}$ described above can be formalized in the following form [13]

$$ {}_{t_0}\mathscr{D}_t^{\alpha} f(t) = \begin{cases} d^{\alpha} f(t)/dt^{\alpha}, & \text{if } \alpha > 0, \\ f(t), & \text{if } \alpha = 0, \\ \int_{t_0}^{t} f(\tau) d\tau^{-\alpha} & \text{if } \alpha < 0. \end{cases} \tag{1} $$

A further extension of the capabilities of fractional order operators is to change their order α according to some

The associate editor coordinating the review of this manuscript and approving it for publication was Kathiravan Srinivasan.

dependent or independent variable, e.g. time t. In the case of the independent variable represented as a function of time, the order of the fractional operator is expressed as some known or unknown function, i.e. $\alpha = g(t)$ where $t \in \mathbb{R}$. Thus, in the context of (1), the variable order fractional calculus operator (VO-FC) expressed as ${}_{t_0}\mathscr{D}_t^{\alpha(t)}$ will be obtained.

There is no doubt that the introduction of variable order $\alpha(t)$ to the fractional operators extends their usefulness. In [3], the authors gave a comprehensive review related to applications of fractional order operators. The authors state that the most significant publications related to fractional operators of variable order belong to the following technical areas and real-world applications: transport processes in complex media [14], [15], control [16], [17], mechanics [18], elasticity, viscoelasticity [19], [20], model-order reduction of lumped parameter systems [21] and biomedical engineering [22]. Outside these application areas, most publications are in the purely scientific area of mathematics [23]–[25]. These publications are mainly in the fields of solution methods, along with definitions and properties.

An important issue related to the research on fractional-order operators is their practical implementation, i.e. through electric circuits or on modern digital platforms like PLC [26], industrial computers or FPGAs [27]. The synthesis of mathematical models or systems which include fractional-order operators is not problematic. At present, it boils down to the selection and application of appropriate definitions and numerical methods. Nowadays, there are many different software tools available like FOMCON [28], FOTF Toolbox [29], or CRONE [30] which support simulation studies in a beneficial and complex way.

When dealing with fractional systems that operate in on-line mode with unknown signals, for instance, in the broad control systems field, it is impossible to use classical definitions of fractional order operators. The unknown signal behavior in the field of control systems is mainly because these systems operate in tracking or disturbance rejection mode. The main reason and at the same time the fundamental disadvantage is that definition based fractional operators need an infinite history of signal samples based on which the output signals from, i.e. fractional controllers [12] or fractional observers [31], [32], etc. are evaluated. This requires the impossible requirement of using infinite memory in digital platforms which are currently acting as carriers of algorithms within many control loops. Therefore, there is a need for techniques and models that allow for approximations of fractional-order operators and systems that will reproduce required fractional operators characteristics with sufficient precision based on a finite history of samples. A typical approach uses continuous or discrete integer order linear dynamical systems that utilize a finite amount of samples for approximation purposes. Typical approximation solutions include: standard and refined Oustaloup filters, frequency response fitting approach, continued fraction-based approximations and many others described, for instance, in [13]. These solutions are typically applied for fixed order operators, and they are impractical for straight introduction of order variation. The disadvantage of these solutions is that it is necessary to recalculate all the parameters that are involved in the approximation model with respect to variations in the order $\alpha(t)$. Another way to introduce variable order while using integer operators is to utilize fuzzy modeling as presented in [25]. However, the latter approach also has some disadvantages in terms of the high complexity of the approximating model since it requires incorporation of multiple linear models and multiple fuzzy rules to ensure decent quality of the approximation.

Therefore, this paper proposes an original approach in the form of a neural network model of the VO-FC operator. The following properties characterize this approach: firstly it does not need infinite memory for sample storage, secondly, it is not computationally demanding, and finally neural networks can automatically learn VO-FC operators' characteristics from data through optimization. The data-driven approach to modeling the approximator, in this case, is very flexible because through the definition of appropriate training data and selecting an appropriate neural network architecture, it is possible to design approximation models with relatively small modeling errors and characteristics suited to the given requirements. Also, the following allow for the implementation of such neural approximated operators on digital platforms operating in real-time conditions, which do not have significant computing power, for instance, PLC, PAC, microcontrollers or FPGA boards.

Recurrent neural networks belong to a class of artificial neural networks that are characterized by the utilization of feedbacks embedded into their architecture. Feedbacks, depending on the chosen architecture, can take different forms. Nevertheless, the introduction of feedback into a neural network turns it into a dynamic model that represents an input-output mapping that considers the temporal sequences present in the data rather than just static ones. The combination of the main advantages of recurrent neural networks, i.e.: scalability and flexibility while modeling almost any dynamic system, automatic and optimal adaptability of the network to the presented data during supervised learning task and relatively low computational complexity of the trained network is the main reason for which they were used in this study to approximate VO-FC operators.

Previous research related to the usage of recurrent neural networks for modeling fractional dynamical systems of constant order presented in the paper [33] showed that recurrent neural networks could at the same time approximate the behavior of simple and complex fractional systems in a promising way. This paper extends the previous research towards modeling approximations of VO-FC operators based on recurrent neural networks.

The main contribution of the paper is represented by original research results that address:

- complex problem of approximation of VO-FC operators,
- utilization of recurrent neural network architectures to implement the approximator,

- comparison of results for basic neural networks using Tapped Delay Lines (TDL) and networks using modern GRU cells,
- propositions of supervised training of neural networks and their verification utilizing training and test data sets based on randomized sequences,
- modification of the Grünwald-Letnikov definition of the VO-FC operator, leading to less complex numerical implementation by introducing recursive evaluation of weights.

The paper is organized as follows. In section II the considered problem is described. In section III the problem statement and the methodology used in the study is presented. Section IV includes and discusses the quantitative and qualitative results that were obtained from the simulation studies. Finally, section V concludes the paper.

II. PROBLEM STATEMENT

A. VARIABLE ORDER FRACTIONAL OPERATORS

State-of-the-art summary of various VO-FC operators definitions can be found in [34]. It includes, among others, proposed definitions approaches such as: 1) Ross-Samko [35], [36], 2) Lorenzo-Hartley [37], 3) Coimbra and Valério-Sá da Costa [38], 4) Grünwald-Letnikov type A, B, C, and D formulations [25], [39]. The main factors characterizing the definitions of VO-FC are the variations in the appearance of the memory effects and the variations in the approach regarding changes of the order $\alpha(t)$ as reported in [40], [41]. Such diversity in definitions is most desirable. It allows for more flexibility in choosing an appropriate definition for the modeled physical phenomenon in which memory effects and specific phenomena with a change of order $\alpha(t)$ occur. Despite the wide range of definitions to choose from, according to [34], many of the novel VO-FC definitions do not satisfy the properties indicating that they can be considered as a fractional derivative. That being said, most definitions do not satisfy the criterion of existence of left inverse, which directly implies that the operators do not meet the following condition [34]

$$\mathscr{D}^{\alpha(t)}\mathscr{D}^{-\alpha(t)}f(t) = f(t) \tag{2}$$

Although, as previously mentioned, there are many definitions of VO-FC operators, it was decided in this research to use the definition based on the Grünwald-Letnikov approach proposed by [34] and given as

$$\mathscr{D}_f^{\alpha(t)}f(t) = \lim_{h\to 0^+} h^{-\alpha(t)} \sum_{k=0}^{\infty} \frac{(-\alpha(t))_k}{k!} f(t-kh) \tag{3}$$

where $(a)_k = a(a+1)(a+2)\ldots(a+k-1)$ denotes Pochhammer symbol.

Due to usage of a factorial operation in the denominator $(k!)$, the above definition may cause computational problems. Therefore, the definition (3) has been modified and replaced by formula (4) in which the weights expressed via the $(-\alpha(t))_k/k!)$ expression are determined recursively. The factor associated with the weights was first expanded by the expression $(-1)^{2k}$ yielding

$$\mathscr{D}_f^{\alpha(t)}f(t) = \lim_{h\to 0^+} h^{-\alpha(t)} \cdot \sum_{k=0}^{\infty} (-1)^k \frac{(-1)^k\,(-\alpha(t))_k}{k!} f(t-kh) \tag{4}$$

knowing that

$$\binom{\alpha}{k} = \frac{(-1)^k\,(-\alpha)_k}{k!} \tag{5}$$

and substituting (5) to (4) we get

$$\mathscr{D}_f^{\alpha(t)}f(t) = \lim_{h\to 0^+} h^{-\alpha(t)} \sum_{k=0}^{\infty} (-1)^k \binom{\alpha(t)}{k} f(t-kh) \tag{6}$$

Then, following [13], subsequent relationship was introduced to (6)

$$(1-z)^{\alpha(t)} = \sum_{k=0}^{\infty} (-1)^k \binom{\alpha(t)}{k} z^k = \sum_{k=0}^{\infty} w_k(t) z^k \tag{7}$$

finally resulting in

$$\mathscr{D}_f^{\alpha(t)}f(t) \approx h^{-\alpha(t)} \sum_{k=0}^{\infty} w_k(t) f(t-kh) \tag{8}$$

where $w_k(t)$ are the coefficients of $(1-z)^{\alpha(t)}$ that can be expressed recursively as

$$\begin{aligned} w_0(t) &= 1, \\ w_j(t) &= \left(1 - \frac{\alpha(t)+1}{j}\right) w_{j-1}, \quad \text{for } j = 1, 2, \ldots \end{aligned} \tag{9}$$

For the sake of clarity, in the rest of the paper, the relationships described by (8)-(9) will be abbreviated as operator $O1$.

It should also be mentioned that definition (3) transformed to the form (8)-(9) was chosen because 1) it satisfies condition (2), that is, it can be used for both integration and differentiation operations merely by changing the sign of the order of $\alpha(t)$ so that it is not necessary to use two separate definitions for fractional integration and differentiation, 2) does not have memory effects when changing the order of $\alpha(t)$ making it more challenging for a recurrent neural network to learn the approximation of a given VO-FC operator, 3) it is by itself a numerical procedure that can directly be used for the determination of variable-order fractional integrals and derivatives, and last but not least 4) it is straightforward in numerical implementation which improves the generation of training and testing data for the training process of recurrent neural network.

In order to verify the correctness of the calculations performed using the $O1$ operator, it was implemented and tested in the Python 3.9 programming language with NumPy library [42], [43]. The implemented operator was tested according to an example taken from [34] in which unit step

and ramp response of the system given by the following transfer function

$$F(s) = \frac{1}{0.1 + s^{\alpha(t)}} \tag{10}$$

was to be determined. During the verification, the FOMCON [28] toolbox dedicated for Matlab [44] software allowing for the calculation of responses of dynamic systems of fractional order was also used for comparison purposes. Figure 1 shows the results of the verification experiment of the $O1$ operator. The upper and middle part of the figure shows the responses of the system for different variants of fractional order α, while the lower part of the figure shows the time variation of $\alpha(t)$ assumed in this experiment. The $O1$ operator in this experiment was used to determine the unit step and ramp responses of the system (10) for both fixed and variable orders of α. As can be seen in the figure, the $O1$ operator correctly computes system responses for both variable and fixed orders of fractional order α. This fact is also confirmed by the data obtained from the simulation with the FOMCON toolbox.

B. NEURAL NETWORK ARCHITECTURE

Artificial neural networks are known as universal structures that can model complex relationships using a large set of simple mathematical operations processed through a large number of computational cells called neurons. Whether a neural network will be able to model the relationships that exist in the data depends mainly on the architecture of the network.

After expanding the relation (3), it can be seen that it represents a weighted sum that essentially is a moving average with an infinitely growing window. Since the definition of the Grünwald-Letnikov integro-differential operator utilizes previous samples of the signal on which it operates, it is reasonable to use a recurrent neural network architecture for approximation purposes. These type of network utilize feedbacks from inputs, hidden layers and outputs embedded in their structure to generate output. An essential element supporting the use of recurrent neural networks in this research is the universal approximation theorem expressed in the following form [45], [46]

Theorem 1: Any nonlinear dynamic system may be approximated by a recurrent neural network to any desired degree of accuracy and with no restrictions imposed on the compactness of the state space, provided that the network is equipped with an adequate number of hidden neurons.

This theorem was formalized in the work [47] where conditions for its applicability can also be found. The above-stated considerations lead to the formulation of assumptions considering a neural-based model of the VO-FC operator as follows:

- the model should be based on a recurrent neural network,
- the model should aim to minimize the number of neurons in the network and thus the feedbacks from inputs or outputs that are used to determine the resulting output signal from the network with a trade-off between modelling accuracy and the size of the neural network,
- although the Grünwald-Letnikov or $O1$ definition is linear in terms of the used input signal samples, the model should leverage the nonlinear activation functions of the neurons to compensate for modeling errors,
- the model should have at least two inputs responsible for supplying the input signal and the signal associated with order variations denoted as $\alpha(t)$, and at least a single output on which the operator subjected signal appears.

Considering the assumptions mentioned above, the general aim of this study is to approximate the $O1$ operator by a recurrent neural network to eliminate the fundamental disadvantages of definition driven fractional operators mentioned in section I. In order to simplify the training process, it was decided to divide the process according to integration and differentiation operations. Thus, in the presented research, two neural models were developed that approximate the fractional integral and differential operators concerning variations in the order $\alpha(t)$ of these operators.

III. RESEARCH METHOD

In this part of the paper, aspects related to: (A) the selection of the neural network architecture for approximating the VO-FC operator, (B) the procedure for generating the data necessary for the training and validation process of the neural network, (C) the selection of the hyperparameters and the training of the neural network, will be discussed.

A. NEURAL NETWORK

The preliminary research to investigate whether recurrent neural networks are suitable for approximation of fractional order systems was presented in [33]. In the mentioned paper recurrent neural networks based on neuron cells such as LSTM and GRU and the classical Elman recurrent network were examined. The mentioned networks were supposed to approximate the responses of both less and more complex fractional-order dynamic systems such as:

- first-order LTI system,
- second-order LTI system,
- nuclear reactor model,
- strongly nonlinear model.

The research presented in [33] showed that the studied recurrent neural networks with LSTM, GRU, and Elman cells were up-and-coming approximations of the selected dynamical fractional-order systems. In the research mentioned above, the best average performance was achieved by GRU cell type neural networks. For this reason, in the research presented in this paper, a GRU cell type recurrent network was used to approximate the VO-FC operators. The general architecture of the neural network with GRU cells is shown in Figure 2. The structure of a single GRU cell is defined as follows [48]:

$$r(t) = \sigma(W_{ir}x(t) + b_{ir} + W_{hr}h(t-1) + b_{hr})$$
$$z(t) = \sigma(W_{iz}x(t) + b_{iz} + W_{hz}h(t-1) + b_{hz})$$

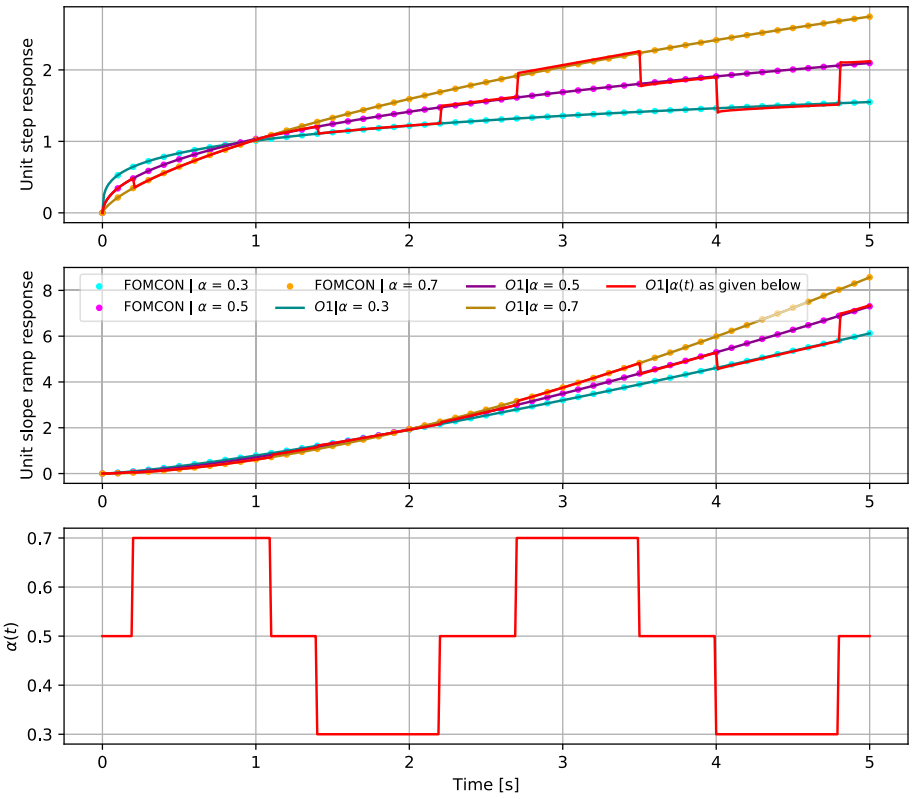

FIGURE 1. Verification of the behavior of the *O*1 operator with constant and varying orders $\alpha(t)$ for step and ramp response of the transfer function (10).

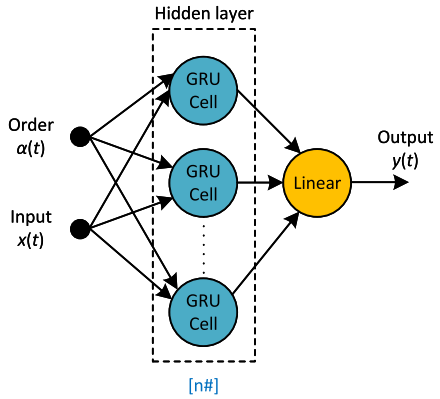

FIGURE 2. The general architecture of the utilized recurrent neural network with GRU cells. The parameters being changed during the study are indicated with blue text: n# - neuron number.

$$\begin{aligned} n(t) &= \tanh(W_{in}x(t) + b_{in} + r(t) * \\ &\quad * (W_{hh}h(t-1) + b_{hn})) \\ h(t) &= (1 - z(t)) * n(t) + z(t) * h(t-1) \end{aligned} \tag{11}$$

where $h(t)$ is the hidden state at time t, $x(t)$ is the input at time t, $h(t-1)$ is the hidden state at time $t-1$ or the initial hidden state at time 0, $r(t)$, $z(t)$, $n(t)$ are the reset, update and new gates at time t, respectively, σ is the sigmoid function, $*$ is the Hadamard product, W_{ir}, W_{iz}, W_{in} are learnable input-reset, input-update and input-new weights matrices, W_{hr}, W_{hz} are hidden-reset, and hidden-update weights matrices, b_{ir}, b_{iz}, b_{in} are learnable input-reset, input-update and input-new biases, b_{hr}, b_{hz}, b_{hn} are learnable hidden-reset, hidden-update, and hidden-new biases.

In general, the neural network consisted of a single hidden layer, which is in agreement with the previously presented Theorem 1, and a single output layer that applies a linear transformation to the incoming data described as

$$y(t) = h(t)A^T + b \tag{12}$$

where $y(t)$ is the output vector at time t, A is the linear transformation matrix, and b is the bias vector.

The network has two inputs, of which the first is the fractional-order $\alpha(t)$ of the neural operator, and the second is the input signal $x(t)$ subjected to fractional differentiation or integration operation. Both inputs depend on time t. The network also has a single output $y(t)$, which is the result of a fractional differentiation or integration operation of the input signal $x(t)$ depending on the varying order of $\alpha(t)$. In the case of mentioned network, the number of GRU cells in the hidden layer was analysed. More complex neural networks with two hidden layers, e.g., 20 neurons each in two hidden layers, were tested in initial experiments, but satisfactory modeling error was not achieved in these cases.

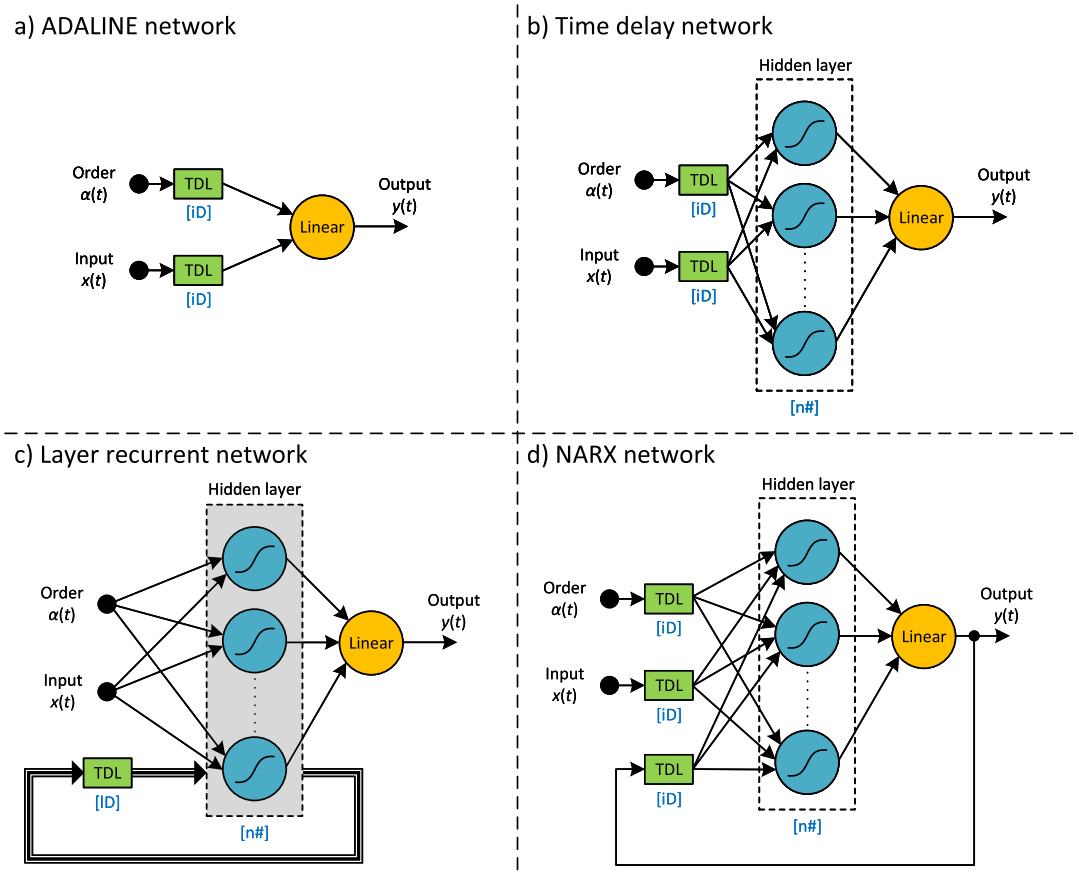

FIGURE 3. Simplified diagrams of the basic neural networks architectures with Tapped Delay Lines. The parameters that were changed during the study are marked with blue text. Bold line indicates the vector of signals from all neurons in the hidden layer.

Additionally, due to the nature of input signal samples processing by the Grünwald-Letnikov definition as mentioned in subsection II-B i.e., computation of a weighted average with an infinitely increasing window on the input signal samples, other basic neural network architectures that utilize Tapped Delay Lines (TDL) were also investigated. During conducted tests four basic neural network architectures were examined, namely:

1) ADALINE network,
2) Time delay neural network,
3) Layer recurrent network,
4) NARX network.

Simplified diagrams of the architectures of the examined basic neural networks are shown in Figure 3.

B. DATA

The recurrent neural networks used in this study was trained in a supervised manner from synthetically generated data. The data used in the training process reflected the operations performed by the VO-FC operators for a particular class of input signals with varying orders of the differentiation or integration operator. Data were divided into two categories that were directly related to the type of operator, which was to be approximated by a neural network, i.e. data reflecting integration and differentiation with a variable order. The plots of training data are shown in the Appendix in Figures 6 and 7. The training data were generated using an Amplitude Modulated Pseudo-Random Binary Sequence (APRBS) [49]. The input and order signals for the integration operation and the order signal for differentiation operation were not subjected to further transformations. In contrast, the input signal for the differentiation operation is an integrated version of the APRBS sequence. This change is because the differentiation of step functions will result in very high amplitude spikes, which is undesirable in training signals for recurrent neural networks. Parameters for training data are listed in Table 1.

In addition to the training data, test data were also generated. Test data was used to check if the trained neural network can generalize the integration and differentiation operations to signals that were not present in the training process. Plots of testing data are also shown in Appendix on Figures 8 and 9. The test data are also divided into integration and differentiation signals taking into account changes in the order of these operators. The main difference in the test data compared to the signals contained in the training data is that, in addition to APRBS sequences, they also contain sinusoidal

TABLE 1. Training data signals parameters.

Parameter	Integration input	Integration order	Differentiation input	Differentiation order
Signal class	APRBS	APRBS	Integrated APRBS	APRBS
Signal range	-1÷1	0÷1	n/a	-1÷0
Sampling time [sec.]	0.01			
Start time [sec.]	0			
End time [sec.]	10			
Signal count	50			
APRBS max. hold time [samples]	200			
APRBS min. hold time [samples]	100			

TABLE 2. Testing data signals parameters. Entries in parentheses apply for the differentiation operator.

Parameter	Input signal 1	Input signal 2	Input signal 3	Order signal
Class	APRBS (Integrated APRBS)	Sine wave	Sawtooth wave	APRBS
Signal range	-1÷1 (n/a)	-1÷1	-1÷1	0÷1 (-1÷0)
Frequency [rad/s]	n/a	0÷3	0÷3	n/a
Phase	n/a	$-\pi \div \pi$	$-\pi \div \pi$	n/a
Signal count	7	7	7	21
APRBS max. hold time [samples]	200	n/a	n/a	100
APRBS min. hold time [samples]	100	n/a	n/a	200
Sampling time [sec.]	0.01			
Start time [sec.]	0			
End time [sec.]	10			

TABLE 3. Parameters set during the basic neural operators training process. Hyperparameters marked with (*).

Parameter	Network			
	ADALINE	Time delay	Layer recurrent	NARX
Neurons in hidden layer [#n] (*)	n/a	1:10		
Input delays [iD] (*)	1:10	1:10	n/a	1:10
Layer delays [lD] (*)	n/a	n/a	1:10	n/a
Optimizer	Levenberg-Marquardt backpropagation			
Data Division	Random			
Trainer max epochs	1000			

and sawtooth signal sequences. The parameters of the test signals are summarized in the Table 2. Output signals for both training and test data were generated from the $O1$ definition of VO-FC operator described by the relationships (8)-(9).

C. TRAINING

Once a suitable neural network architecture is selected, and the appropriate training and testing data have been chosen, the next step is to train the neural VO-FC operators. In the first part of the study, the basic neural networks shown in Figure 3 were trained to approximate the neural operators. These networks were implemented and trained using the Deep Learning Toolbox package of Matlab R2021a software [50]. In this part, the behavior of four basic neural networks with embedded TDL was studied concerning changes in the number of:

- input delays [iL],
- neurons in the hidden layer [n#],
- delays from neurons of the hidden layer [lD].

The mentioned hyperparameters were changed depending on the network structure. A detailed list of the parameters and hyperparameters for the learning process of basic networks are given in Table 3.

Further research concerned training of the GRU cell-based neural network shown in Figure 2 to approximate the neural operators. The implementation and training of the GRU neural operators were carried out using Python 3.9 [42] with the aid of the PyTorch [48] and PyTorch Lightning [51] libraries. The parameters that were set during the GRU network training process are provided in Table 4. Parameters not listed in Tables 3 and 4 were set to default values according to the Matlab Deep Learning Toolbox, PyTorch and PyTorch Lightning libraries.

Previous research presented in the paper [33], studied the architectures of recurrent neural networks based on LSTM, GRU and Elman cells for approximation of fractional models with constant order. In the mentioned studies, the number of neurons in the hidden layer varied from 1 to 5. It was observed that greater number of neurons caused the common phenomenon of overfitting of the neural network, which decreased the ability to generalize the performance of neural operators on signals which did not occur in the training data set. In the current research, it was analyzed and determined that the proper range of the number of neurons in the hidden layer vary from 7 to 9. It should be recalled here that the choice of the number of neurons in the hidden layer was dictated by the assumptions presented in section II-B and in particular to minimize the complexity of the network, which directly reflects the number of neurons involved. It was also

TABLE 4. GRU neural operators training process parameters. Hyperparameters marked with (*).

Parameter	Value
Neurons in hidden layer (*)	7:9
Optimizer	AdamW
Learning rate	0.1
Use of AMSGrad algorithm	True
Weight decay	0.008
Data random split	35 training samples, 15 validation samples
Batch size for training	35
Batch size for validation	15
Trainer precision	32
Trainer max epochs	50000
Trainer gradient clip	0
Learning rate scheduler	CosineAnnealingLR, T_max = 50000

TABLE 5. Results of training sessions for basic neural networks. The results refer to the lowest recorded values of the loss function for the integration operator data set.

Network	Input delays	Neuron #	Layer delays	Testing loss
ADALINE	10	n/a	n/a	1.69E-01
Time delay	10	3	n/a	1.69E-01
Layer recurrent	n/a	10	9	8.44E-02
NARX	6	10	n/a	6.25E-02

observed that the modeling error is the lowest in this range of the number of neurons.

In order to achieve better generalization ability by the neural operators and reduce the risk of overfitting, the validation data were separated from the training data. Data division for basic neural networks using Matlab Deep Learning Toolbox was done randomly using the `dividerand` function with the default settings. When training neural networks with GRU cells, 35 samples were randomly selected for training data and 15 random samples for verification data. Additionally, in each training trial, data were reshuffled at each training epoch.

The optimal parameters of the neural networks, both basic and with GRU cells, were selected at the training epoch in which the lowest value of the loss function for validation data was achieved, the so-called sweet spot. The MSE function (squared L2 norm) given as (13) was used in the optimization process [48]

$$l\left(x_I, y_T\right) = \text{mean}\left\{l_1, \ldots, l_N\right\}^T, \quad l_n = \left(x_{I,n} - y_{T,n}\right)^2 \quad \text{for} \quad n = 1, \ldots, N \tag{13}$$

where x_I is the input sequence, y_T is the target sequence, and N is the batch size. This function was also used as a loss function in the results presented in Section IV.

IV. RESULTS

This section includes the results of training sessions that were carried out to determine the best neural models representing VO-FC operators. Table 5 shows results obtained during the training sessions for the basic neural networks. Table gathers the minimum values of the loss function for each of the tested networks, evaluated based on five independent runs. This minimum values were calculated for the test data set and for the hyperparameters reported in the table. Because the best results from the Table 5 were not satisfying (in contrast to the neural networks with GRU cells, these results were two orders of magnitude larger), no further studies with differentiation data were performed. Furthermore, the data set related to the differentiation operator is not well suited for comparing network performance. It contains spikes that can be misleading for quantitative interpretation due to

TABLE 6. Results of training sessions with statistics for neural models of variable order fractional integration operators.

Loss	Neuron #	Run1	Run2	Run3	Run4	Run5	Mean	Std	Min	Max
Training	7	5,58E-04	8,61E-04	9,41E-04	8,43E-04	1,06E-03	8,53E-04	1,86E-04	5,58E-04	1,06E-03
Validation		2,84E-02	2,17E-03	9,76E-03	2,73E-03	3,07E-02	1,48E-02	1,39E-02	2,17E-03	3,07E-02
Testing		1,14E-03	1,02E-03	1,18E-03	1,17E-03	9,43E-04	**1,09E-03**	1,05E-04	9,43E-04	1,18E-03
Training	8	5,85E-04	6,50E-04	7,23E-04	4,72E-04	6,55E-04	6,17E-04	9,48E-05	4,72E-04	7,23E-04
Validation		3,39E-02	2,61E-03	1,01E-02	3,31E-02	6,33E-03	1,72E-02	1,51E-02	2,61E-03	3,39E-02
Testing		1,09E-03	9,49E-04	4,06E-03	2,28E-03	8,80E-04	1,85E-03	1,36E-03	**8,80E-04**	4,06E-03
Training	9	8,59E-01	3,25E-04	6,48E-04	5,07E-04	8,26E-01	3,37E-01	4,61E-01	3,25E-04	8,59E-01
Validation		5,66E-01	3,96E-02	3,76E-02	3,01E-02	6,40E-01	2,63E-01	3,12E-01	3,01E-02	6,40E-01
Testing		2,90E-01	1,11E-03	6,41E-03	9,32E-04	2,86E-01	1,17E-01	1,56E-01	9,32E-04	2,90E-01

TABLE 7. Results of training sessions with statistics for neural models of variable order fractional differentiation operators.

Loss	Neuron #	Run1	Run2	Run3	Run4	Run5	Mean	Std	Min	Max
Training	7	1,59E-03	1,83E-03	1,66E-03	2,04E-03	1,86E-03	1,79E-03	1,76E-04	1,59E-03	2,04E-03
Validation		1,56E-02	6,84E-03	1,55E-02	1,08E-02	5,73E-03	1,09E-02	4,63E-03	5,73E-03	1,56E-02
Testing		2,15E+00	2,28E+00	2,17E+00	2,21E+00	2,12E+00	2,19E+00	6,16E-02	2,125E+00	2,28E+00
Training	8	1,13E-03	1,08E-03	8,91E-04	9,95E-04	1,31E-03	1,08E-03	1,57E-04	8,91E-04	1,31E-03
Validation		6,00E-02	6,32E-03	2,78E-02	3,11E-02	2,99E-03	2,56E-02	2,29E-02	2,99E-03	6,00E-02
Testing		2,16E+00	2,18E+00	2,13E+00	2,20E+00	2,14E+00	**2,16E+00**	2,94E-02	2,13E+00	2,20E+00
Training	9	1,22E-03	7,69E-04	1,16E-03	8,80E-04	1,32E-03	1,07E-03	2,34E-04	7,69E-04	1,32E-03
Validation		3,73E-03	1,90E-03	1,56E-03	1,54E-03	1,80E-02	5,34E-03	7,11E-03	1,54E-03	1,80E-02
Testing		2,19E+00	2,19E+00	2,16E+00	2,12E+00	2,17E+00	2,17E+00	2,90E-02	**2,123E+00**	2,19E+00

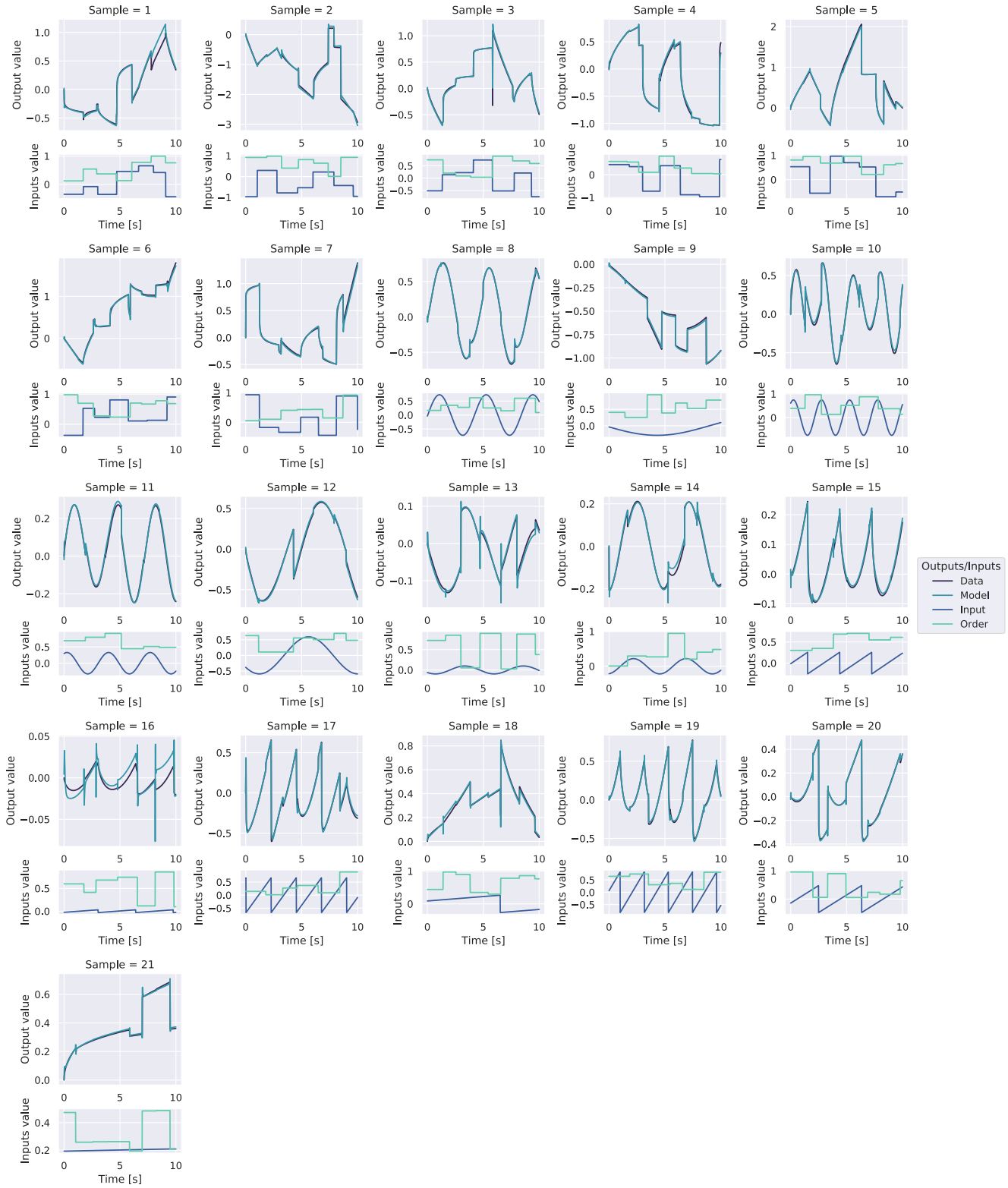

FIGURE 4. Responses to test signals for the best neural network for the integration operator. For each sample, the input signals are given in smaller plots.

the MSE objective function used in the study, which tends to amplify significant errors possibly caused by spikes.

Tables 6 and 7 contain results related to the loss functions for the training sessions for the GRU neural VO-FC

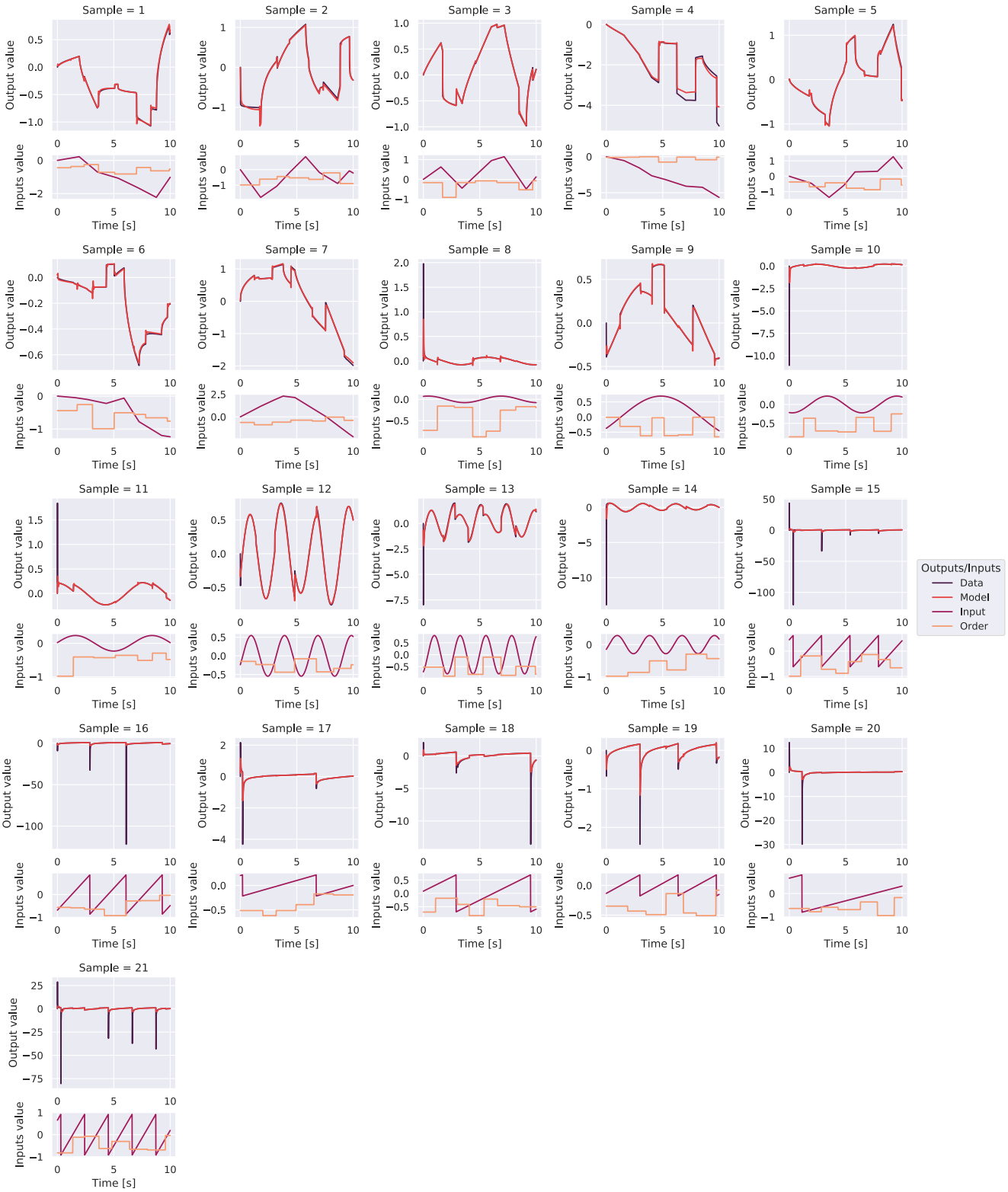

FIGURE 5. Responses to test signals for the best neural network for the differentiation operator. For each sample, the input signals are given in smaller plots.

integration and differentiation operators, respectively. In the tables, the best performance of the neural models for the test data (blue rows) in terms of the average loss function and the best training session loss function is highlighted

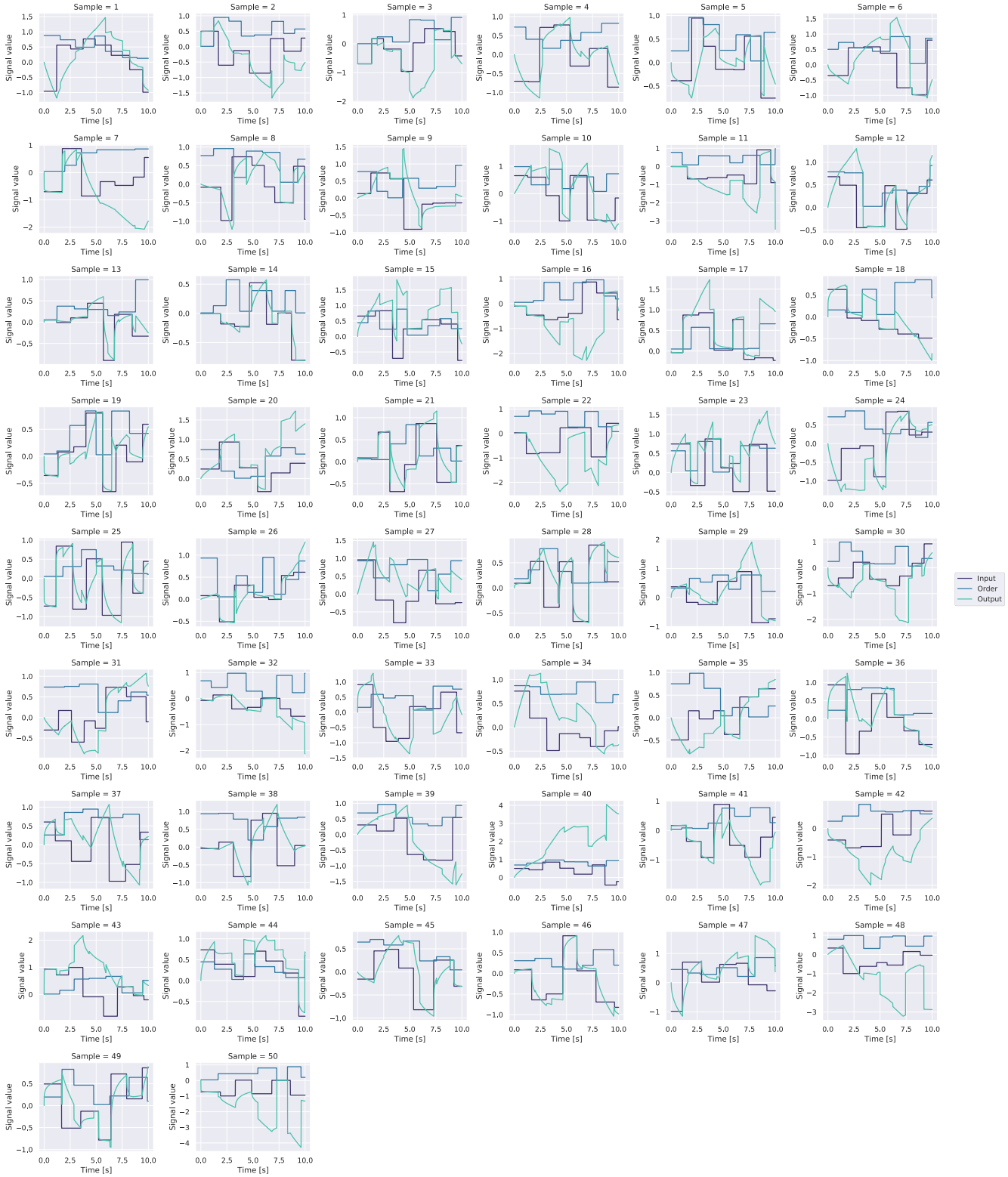

FIGURE 6. **Training data for integral operation.**

with bold and underline text. It can be seen that the best result for the integration operation in terms of mean and standard deviation was achieved for the network containing 7 GRU cells. For differentiation operation, the network

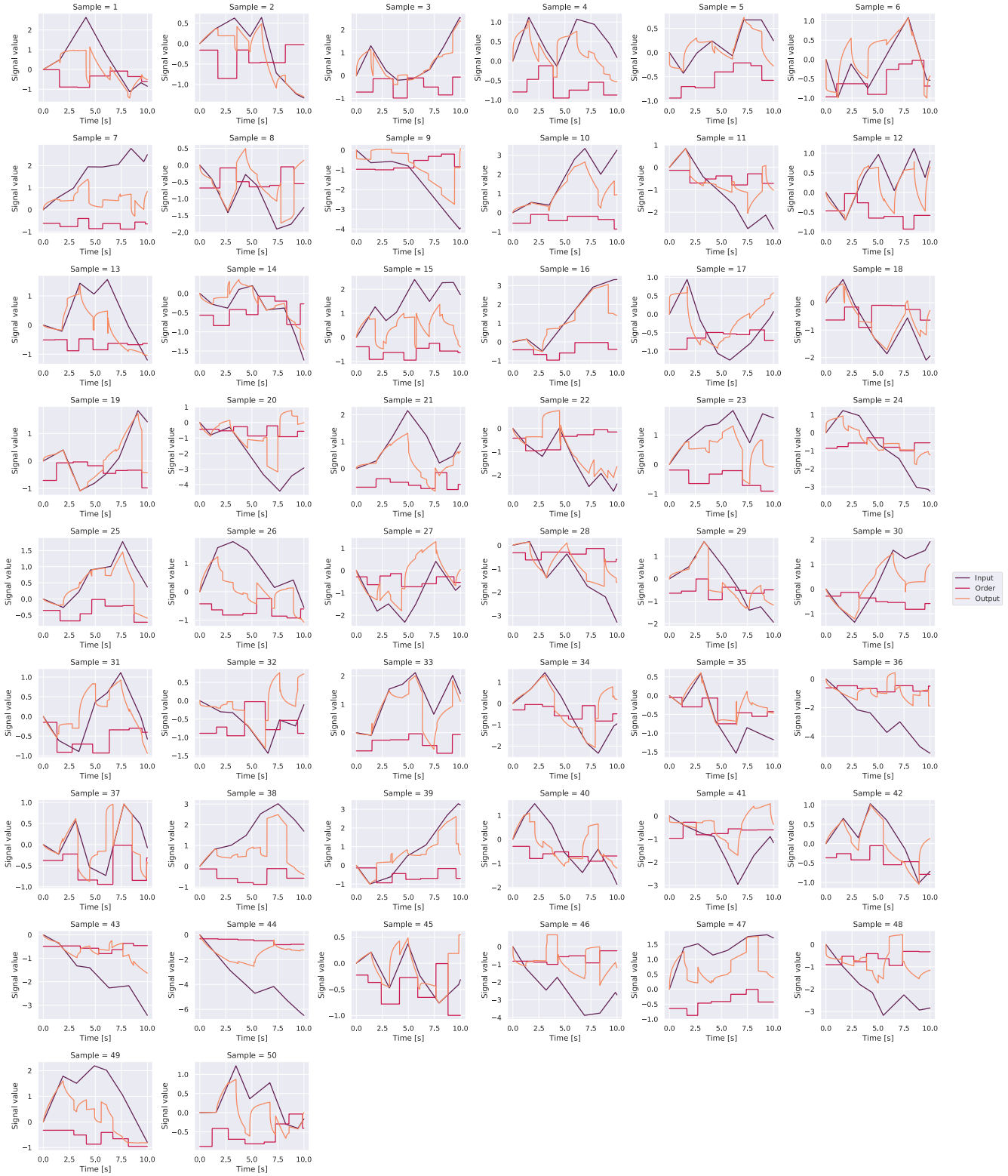

FIGURE 7. Training data for differentiation operation.

with 8 GRU cells achieved the best result for the mean. It should be noted here that the results were very similar for the differentiation operator in the mean testing loss category for each number of considered neurons. In general, the

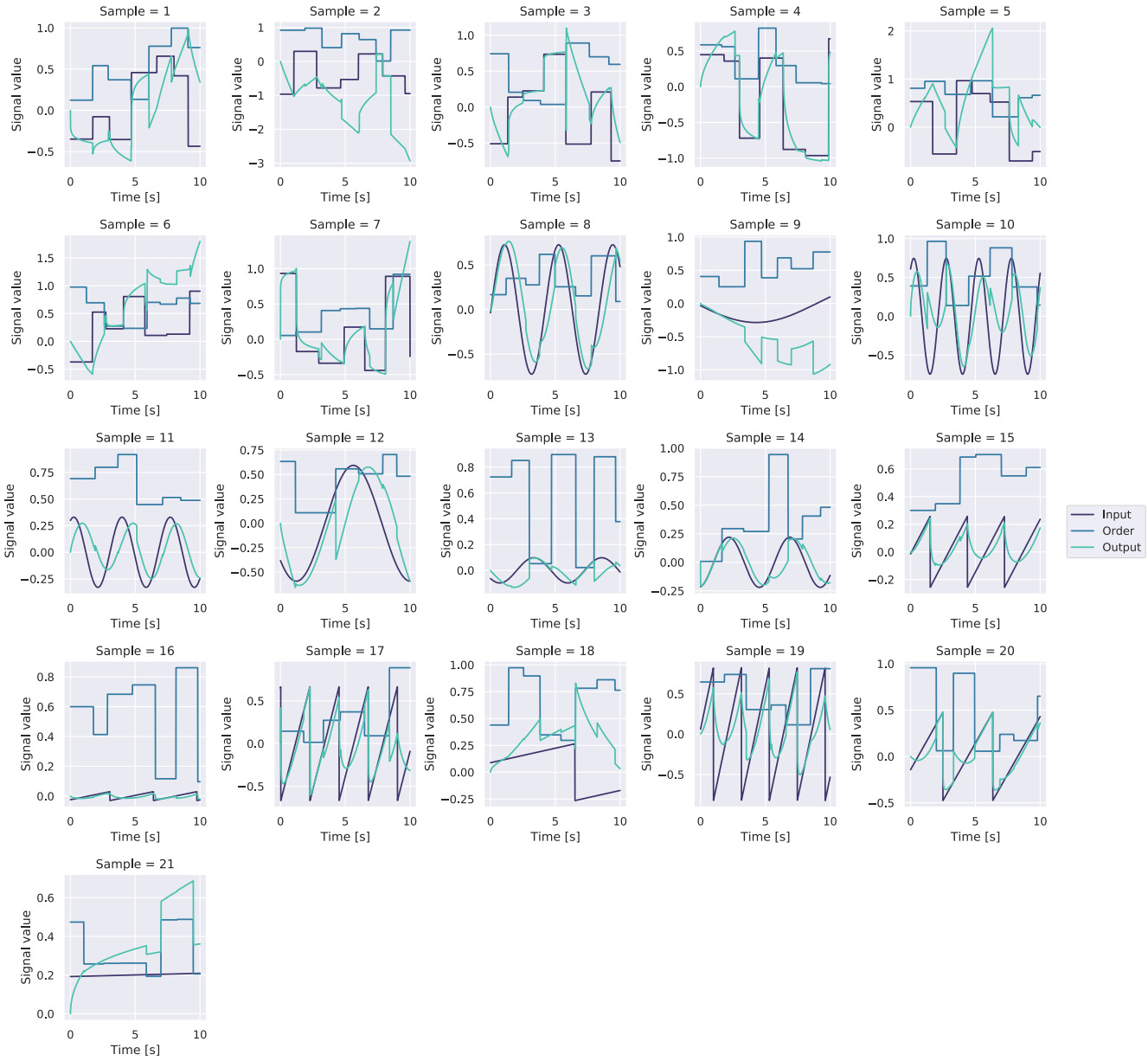

FIGURE 8. Testing data for integral operation.

network with 8 GRU cells for the integration operator and 9 GRU cells for the differentiation operator performed the best out of all trials. In this cases, the smallest recorded test loss values in all trials performed was observed as indicated with bold and underlined entries in Tables 6 and 7. In the context of the presented results, it should be noted that the overall goal of the paper was not to perform a detailed statistical analysis of the selected search region of the chosen hyperparameter (number of neurons in hidden layer). The goal was to find the overall best result in the context of the loss function, which resulted in the smallest modeling error of the corresponding neural operator with respect to the assumptions formulated in subsection II-B.

Qualitative results in the form of plots comparing the test data obtained using the $O1$ operator labelled "Data" and the neural model responses labelled as "Model" are shown in Figures 4 and 5 for the integration and differentiation operators, respectively. In the figures related to each sample, the top plots compare the behavior of the neural operator versus the $O1$ operator. In contrast the bottom plots include waveforms of the operators' inputs signals.

After examining all the samples in both Figures 4 and 5, it can be concluded that the modeling error between the data obtained with the $O1$ operator and the best trained neural networks is satisfactory. In addition, based on the presented plots, it can be concluded that the neural networks

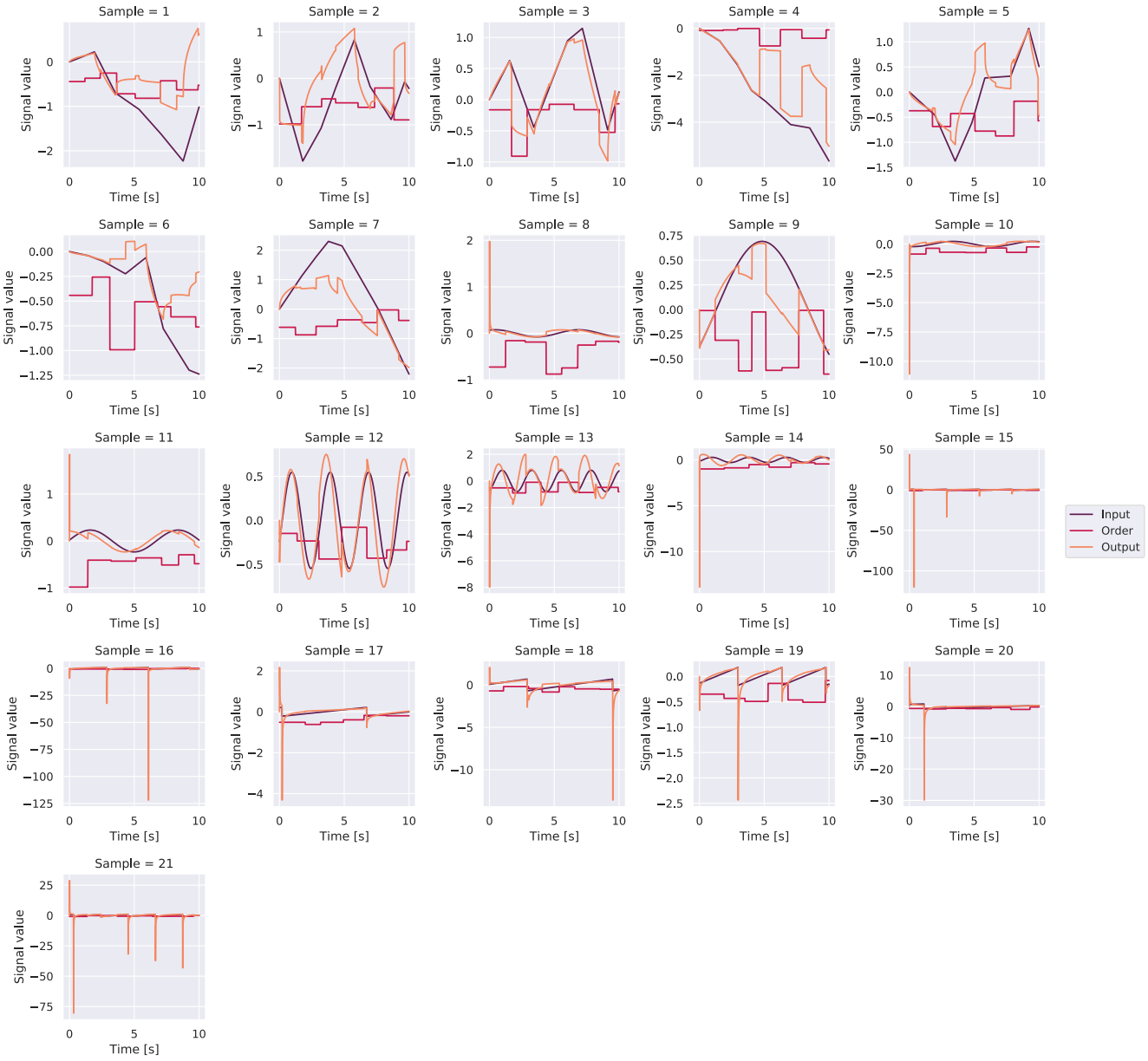

FIGURE 9. Testing data for differentiation operation.

approximate the function of the $O1$ operator without memory effects. Thus, the embedded recurrence in the network architecture positively affects the approximation properties while at the same time not causing additional memory effects during the transient of the input signal that is associated with the order of neural operator.

There are two cases where the neural operators do not fully approximate the behavior of the $O1$ operator. The first one occurs for sample no. 16 in Figure 4 for the integration operator. In this case, it can be seen that the neural approximation does not fully capture the amplitude of the reference signal obtained from the $O1$ operator. This sample is characterized by a very small amplitude of the sawtooth signal that appears at the input $x(t)$ of the neural operator. Although the signal amplitude is not fully approximated the overall dynamic character of the signal is in consistent with the data obtained from the $O1$ operator. This could be a consequence of inappropriate selection of training data, however, it requires further extended research.

The second one is related to spikes that occur mainly in the responses from the $O1$ operator in differentiation data presented in Figure 5. High amplitude spikes caused by differentiation of rapid changes in the input signal can be seen for samples 8 through 21. By analyzing the response of the neural

operator for the mentioned samples, it can be clearly seen that the trained neural network has problems reproducing those spikes. This can be considered a modeling error but nevertheless it is desirable. This effect can be thought of as the natural filtering tendency of a recurrent neural network model. Spike filtering positively impacts the implementation of neural operators on modern digital platforms, especially when considering numerical problems that may occur when the range of variables used for computation is exceeded due to occurring spikes. Also the lack of mapping of the spikes effects for the neural differentiation operator is noticeable in the context of the quantitative results, which are close to each other in both the mean and minimum loss function categories. The loss metric used in the paper in the form of the MSE function tends to miss minor errors and, in contrast, emphasize big ones, which in this case take the form of spikes. As mentioned earlier this tendency is strictly desirable, when transferring trained and ready-to-use neural operators to modern digital control platforms, to avoid computational issues.

Based on the results presented in this section, it can be concluded that the proposed GRU cell-based recurrent neural network architectures approximate VO-FC operators based on Grünwald-Letnikov definition without memory effects with excellent overall performance.

V. CONCLUSION

This paper discusses the process of modeling VO-FC integration and differentiation operators using modern recurrent neural networks architecture based on GRU cells. Neural networks were trained on synthetic data based on the variable order Grünwald-Letnikov definition, with no memory effects. The results presented in this paper show that the applied recurrent neural networks provide an excellent approximation of the Grünwald-Letnikov VO-FC operators.

The paper also includes the case of using basic neural networks to model VO-FC operators, but the use of such networks did not bring satisfactory results.

The developed neural operators can be easily implemented as a part of advanced control systems that use fractional control laws or fractional models in their structure. This approach overcomes an essential drawback of fractional systems, which requires limitless memory resources to store signal samples subjected to fractional-order integration or differentiation operations. The presented neural models and their adaptation for general fractional signal processing are undoubtedly a novel alternative to the currently used and found in the literature approximations of constant and variable-order fractional systems.

The next and natural step in the research related to the approximation of VO-FC operators by recurrent neural networks is their implementation on modern digital control platforms and verification of their proper operation in industrial-like conditions concerning real-time regime. In the future, research on the application of recurrent neural networks for the approximation of other VO-FC operators with weak and strong memory effects is planned. Furthermore, a study is planned to investigate the neural approximations of VO-FC operators with very low amplitudes of the input signals to inspect the issue related to weak modeling quality at low input signal amplitudes mentioned in Section IV.

APPENDIX TRAINING AND TESTING DATA

See Figures 6–9.

REFERENCES

[1] K. Oldham and J. Spanier, *The Fractional Calculus Theory and Applications of Differentiation and Integration to Arbitrary Order*. New York, NY, USA: Academic, 1974.

[2] J. T. Machado, V. Kiryakova, and F. Mainardi, "Recent history of fractional calculus," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 16, no. 3, pp. 1140–1153, 2011.

[3] S. Patnaik, J. P. Hollkamp, and F. Semperlotti, "Applications of variable-order fractional operators: A review," *Proc. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 476, no. 2234, Feb. 2020, Art. no. 20190498.

[4] R. Marazzato and A. C. Sparavigna, "Astronomical image processing based on fractional calculus: The AstroFracTool," Oct. 2009, *arXiv:0910.4637*.

[5] Z. Wang, X. Huang, and G. Shi, "Analysis of nonlinear dynamics and chaos in a fractional order financial system with time delay," *Comput. Math. Appl.*, vol. 62, no. 3, pp. 1531–1539, 2011.

[6] I. Birs, C. Muresan, I. Nascu, and C. Ionescu, "A survey of recent advances in fractional order control for time delay systems," *IEEE Access*, vol. 7, pp. 30951–30965, 2019.

[7] M. Mazandarani and X. Li, "Fractional fuzzy inference system: The new generation of fuzzy inference systems," *IEEE Access*, vol. 8, pp. 126066–126082, 2020.

[8] W. M. Ahmad and J. C. Sprott, "Chaos in fractional-order autonomous nonlinear systems," *Chaos, Solitons Fractals*, vol. 16, no. 2, pp. 339–351, Mar. 2003.

[9] R. Cajo, T. T. Mac, D. Plaza, C. Copot, R. De Keyser, and C. Ionescu, "A survey on fractional order control techniques for unmanned aerial and ground vehicles," *IEEE Access*, vol. 7, pp. 66864–66878, 2019.

[10] V. Martynyuk and M. Ortigueira, "Fractional model of an electrochemical capacitor," *Signal Process.*, vol. 107, pp. 355–360, Feb. 2015.

[11] T. K. Nowak, K. Duzinkiewicz, and R. Piotrowski, "Numerical solution of fractional neutron point kinetics model in nuclear reactor," *Arch. Control Sci.*, vol. 24, no. 2, pp. 129–154, Jun. 2014.

[12] B. Puchalski, T. A. Rutkowski, and K. Duzinkiewicz, "Fuzzy multi-regional fractional PID controller for pressurized water nuclear reactor," *ISA Trans.*, vol. 103, pp. 86–102, Aug. 2020.

[13] D. Xue, *Fractional-Order Control Systems: Fundamentals and Numerical Implementations*. Berlin, Germany: De Gruyter, Jun. 2017.

[14] S. Buonocore, M. Sen, and F. Semperlotti, "Occurrence of anomalous diffusion and non-local response in highly-scattering acoustic periodic media," *New J. Phys.*, vol. 21, no. 3, Mar. 2019, Art. no. 033011.

[15] J. P. Hollkamp and F. Semperlotti, "Application of fractional order operators to the simulation of ducts with acoustic black hole terminations," *J. Sound Vib.*, vol. 465, Jan. 2020, Art. no. 115035.

[16] P. Ostalczyk, "Stability analysis of a discrete-time system with a variable-, fractional-order controller," *Bull. Polish Acad. Sci., Tech. Sci.*, vol. 58, no. 4, pp. 613–619, Dec. 2010.

[17] P. Ostalczyk, "Variable-, fractional-order discrete PID controllers," in *Proc. 17th Int. Conf. Methods Models Autom. Robot. (MMAR)*, Aug. 2012, pp. 534–539.

[18] V. E. Tarasov, "Review of some promising fractional physical models," *Int. J. Modern Phys. B*, vol. 27, no. 9, Apr. 2013, Art. no. 1330005.

[19] F. Mainardi, *Fractional Calculus and Waves in Linear Viscoelasticity: An Introduction to Mathematical Models*. Singapore: World Scientific, 2010.

[20] S. Patnaik, M. Jokar, and F. Semperlotti, "Variable-order approach to nonlocal elasticity: Theoretical formulation, order identification via deep learning, and applications," *Comput. Mech.*, pp. 1–32, Sep. 2021.

[21] J. P. Hollkamp, M. Sen, and F. Semperlotti, "Model-order reduction of lumped parameter systems via fractional calculus," *J. Sound Vib.*, vol. 419, pp. 526–543, Apr. 2018.

[22] R. L. Magin, "Fractional calculus in bioengineering—Part 1," *Crit. Rev. Biomed. Eng.*, vol. 32, no. 1, pp. 1–104, 2004.

[23] M. D. Ortigueira, D. Valério, and J. T. Machado, "Variable order fractional systems," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 71, pp. 231–243, Jun. 2019.
[24] S. Samko, "Fractional integration and differentiation of variable order: An overview," *Nonlinear Dyn.*, vol. 71, no. 4, pp. 653–662, Mar. 2013.
[25] D. Valério and J. S. da Costa, "Variable-order fractional derivatives and their numerical approximations," *Signal Process.*, vol. 91, no. 3, pp. 470–483, Mar. 2011.
[26] B. Puchalski, T. A. Rutkowski, and K. Duzinkiewicz, "Implementation of the FOPID algorithm in the PLC controller—PWR thermal power control case study," in *Proc. 23rd Int. Conf. Methods Models Autom. Robot. (MMAR)*, Aug. 2018, pp. 229–234.
[27] M. F. Tolba, L. A. Said, A. H. Madian, and A. G. Radwan, "FPGA implementation of the fractional order integrator/differentiator: Two approaches and applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 4, pp. 1484–1495, Apr. 2019.
[28] A. Tepljakov, E. Petlenkov, and J. Belikov, *FOMCON Toolbox for Modeling, Design and Implementation of Fractional-Order Control Systems*. Berlin, Germany: De Gruyter, 2019, pp. 211–236.
[29] D. Xue, "FOTF toolbox for fractional-order control systems," in *Applications in Control*. Berlin, Germany: De Gruyter, Jan. 2019, pp. 237–266.
[30] A. Oustaloup, P. Melchior, P. Lanusse, O. Cois, and F. Dancla, "The CRONE toolbox for MATLAB," in *Proc. IEEE Int. Symp. Comput.-Aided Control Syst. Design*, Nov. 2002, pp. 190–195.
[31] I. N'Doye, T. M. Laleg-Kirati, M. Darouach, and H. Voos, "Adaptive observer for nonlinear fractional-order systems," *Int. J. Adapt. Control Signal Process.*, vol. 31, no. 3, pp. 314–331, Jun. 2017.
[32] N. Djeghali, S. Djennoune, M. Bettayeb, M. Ghanes, and J.-P. Barbot, "Observation and sliding mode observer for nonlinear fractional-order system with unknown input," *ISA Trans.*, vol. 63, pp. 1–10, Jul. 2016.
[33] B. Puchalski and T. A. Rutkowski, "Approximation of fractional order dynamic systems using Elman, GRU and LSTM neural networks," in *Artificial Intelligence and Soft Computing* (Lecture Notes in Computer Science), vol 12415, L. Rutkowski, R. Scherer, M. Korytkowski, W. Pedrycz, R. Tadeusiewicz, J. M. Zurada, Eds. Cham, Switzerland: Springer, 2020.
[34] M. D. Ortigueira and D. Valério, *Fractional Signals and Systems*. Berlin, Germany: De Gruyter, 2020.
[35] S. G. Samko and B. Ross, "Integration and differentiation to a variable fractional order," *Integral Transforms Special Functions*, vol. 1, no. 4, pp. 277–300, Dec. 1993.
[36] S. G. Samko, "Fractional integration and differentiation of variable order," *Anal. Math.*, vol. 21, no. 3, pp. 213–236, Sep. 1995.
[37] C. F. Lorenzo and T. T. Hartley, "Variable order and distributed order fractional operators," *Nonlinear Dyn.*, vol. 29, no. 1, pp. 57–98, 2002.
[38] C. F. M. Coimbra, "Mechanics with variable-order differential operators," *Annalen Physik*, vol. 12, no. 1112, pp. 692–703, Dec. 2003.
[39] W. Malesza, M. Macias, and D. Sierociuk, "Analytical solution of fractional variable order differential equations," *J. Comput. Appl. Math.*, vol. 348, pp. 214–236, Mar. 2019.
[40] H. G. Sun, W. Chen, H. Wei, and Y. Q. Chen, "A comparative study of constant-order and variable-order fractional models in characterizing memory property of systems," *Eur. Phys. J. Special Topics*, vol. 193, no. 1, pp. 185–192, Mar. 2011.
[41] D. Sierociuk, W. Malesza, and M. Macias, "Derivation, interpretation, and analog modelling of fractional variable order derivative definition," *Appl. Math. Model.*, vol. 39, no. 13, pp. 3876–3888, Jul. 2015.
[42] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA, USA: CreateSpace, 2009.
[43] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, and R. Kern, "Array programming with numpy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.
[44] MATLAB, *Version 9.10.0.1684407 (R2021a)*, MathWorks, Natick, MA, USA, 2021.
[45] S. Haykin, *Neural Networks and Learning Machines, 3/E*. London, U.K.: Pearson, 2010.
[46] J. T.-H. Lo, "System identification by recurrent multilayer perceptrons," in *Proc. World Congr. Neural Netw.*, vol. 4, Portland, OR, USA, 1993, pp. 589-600.
[47] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control Signals Syst.*, vol. 2, no. 4, pp. 303–314, Dec. 1989.
[48] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035.
[49] M. Deflorian and S. Zaglauer, "Design of experiments for nonlinear dynamic system identification," *IFAC Proc. Vol.*, vol. 44, no. 1, pp. 13179–13184, Jan. 2011.
[50] MATLAB, *Deep Learning Toolbox*. Natick, MA, USA: MathWorks, 2021.
[51] W. Falcon *et al.*, *PyTorch Lightning*, vol. 3. San Francisco, CA, USA: GitHub, 2019, p. 6. [Online]. Available: https://github.com/PyTorchLightning/pytorch-lightning

BARTOSZ PUCHALSKI was born in Gdańsk, in 1986. He received the M.Sc. degree in control engineering, the Engineering degree in electrical engineering, and the Ph.D. degree (Hons.) in control engineering from the Faculty of Electrical and Control Engineering, Gdańsk University of Technology, in 2011, 2013, and 2018, respectively. In 2012, he began to work as a Lecturer at the Gdańsk University of Technology. Since 2018, he has been employed at his Alma Mater as an Assistant Professor. His main research interests include the control, identification and modeling of dynamic systems, fractional order calculus, and recursive neural networks.