Nonlinear Dynamics and Neuroscience

Izvestiya Vysshikh Uchebnykh Zavedeniy. Applied Nonlinear Dynamics. 2025;33(4)

Article DOI: 10.18500/0869-6632-003166

Recurrent neural network consisting of FitzHugh–Nagumo systems: characteristics required for training

N. I. Semenova

Saratov State University, Russia
E-mail: ✉semenovani@info.sgu.ru

Received 25.10.2024, accepted 19.02.2025, available online 7.03.2025, published 31.07.2025

Abstract. The *purpose* of this study is to determine the feasibility and features of training a recurrent neural network consisting of FitzHugh–Nagumo systems with delayed feedback to predict an impulse (spike signal). *Methods.* The network under consideration consisted of $N = 60$ FitzHugh–Nagumo systems with different lag times. During training, the problem of which neuron should be activated and with what strength of lagged feedback was solved. The network was trained using gradient descent from different initial conditions. In the process of research, it was found that the use of standard recurrent network training characteristics such as mean squared error or mean absolute error was not applicable to this task, so an alternative method for computing the loss function was proposed. *Results.* The proposed combined loss function is the sum of MSE error and interspike interval error, and therefore has the following advantages: 1 — includes the information about spike periodicity and interspike intervals, 2 — responds adequately to the absence of a network output signal, 3 — takes into account small amplitude fluctuations in addition to impulse dynamics, which allows predicting complex quasi-periodic signals. It has been shown that gradient descent can be used for the problem, but several initial conditions must be used because of the nonlinearity of the loss function. The more initial conditions, the more accurate the result. *Conclusion.* The problem of predicting a pulse (spike) signal using a self-closed recurrent neural network consisting of FitzHugh–Nagumo systems with delayed feedback has been successfully solved. It was clearly shown what features should be taken into account during loss function calculation and how the gradient descent should be realized.

Keywords: neural networks, recurrent neural networks, FitzHugh–Nagumo system, time-delayed feedback.

Acknowledgements. This work was supported by the Russian President scholarship SP-749.2022.5.

For citation: Semenova NI. Recurrent neural network consisting of FitzHugh–Nagumo systems: characteristics required for training. Izvestiya VUZ. Applied Nonlinear Dynamics. 2025;33(4):590–604. DOI: 10.18500/0869-6632-003166

Introduction

In recent years, artificial neural networks (ANN) have been increasingly introduced into our daily lives, as well as into various fields of science, industry, medicine, etc. The emergence of increasingly complex tasks requires the creation of new network topologies, optimization of the learning process of neural networks, the use of special activation functions and other changes. A deeper understanding of the work of biological neural networks has led to the emergence of fundamentally new types of ANN [1], as well as the emergence of deep neural networks of varying complexity (for example, VGGNet, ResNet, GPipe, BERT, GPT-3).

Currently, training a single high-performance model requires a large amount of hardware and energy resources, which leads to an increase in the carbon footprint [2]. As a separate research area, we can single out the direction of hardware (analog) neural networks [3, 4]. The names ‘analog neural network"and ‘hardware neural network"can be found in foreign literature. This direction implies the creation of neural networks in hardware, when the network neurons themselves and the connection between them are implemented at the physical level, which allows for a significant increase in speed and energy efficiency [5, 6]. Currently, hardware ANNs based on the operation of lasers [7, 8], memristors [5, 9], spin-transfer oscillators [10], photoelectronic chips [6], etc. show the greatest efficiency.

Although the ANN structure was originally built on a simplified understanding of how the brain works, ANNs are fundamentally different in structure, neural computing, and learning rules compared to a biological neural network. This observation led to the emergence of spike neural networks (SNN), which are often referred to as the third generation of neural networks, promising to be a breakthrough in eliminating the bottlenecks of conventional ANNs. Spike neural networks (SNN) are a special class of ANNs where neural models interact through sequences of spikes of varying duration and number [11, 12]. In terms of energy consumption, using SNN on neuromorphic equipment such as TrueNorth [13], Loihi [14], SpiNNaker [15], NeuroGrid [2], etc., is a promising approach [1]. At the same time, SNNS can encode temporal information in their signals and therefore also need other and biologically more plausible rules when learning and using [12, 16].

Despite significant achievements, the performance of SNN on reference datasets such as MNIST [17], CIFAR-10 [18] or Fashion-MNIST [19] is often lower than that of conventional ANN [1, 20]. However, this gap in some tasks is compensated by the speed and energy efficiency of [16, 20, 21]. Nevertheless, SNCs are still quite difficult to train, mainly due to their complex spike dynamics of neurons and the undifferentiability of their activation functions. This can be partly explained by the fact that the images in these datasets were obtained using traditional sensors rather than event cameras. However, the SNN has other significant disadvantages [20]. At the moment, in order to realize the full potential of the SNN, several fundamentally important tasks need to be solved [1]:

- Training. There are two main approaches to learning SNN: (i) learning SNN directly based on gradient descent or unsupervised learning with STDP (spike-timing-dependent plasticity); (ii) converting a pre-trained ANN into a SNN model. The first approach has a problem in calculating the gradient due to the undifferentiated spike signal. In addition, the gradient descent-trained SNN is limited to small architectures and provides poor performance on large datasets such as ImageNet. The second approach increases computational complexity due to the large number of time steps, although these SNN achieve accuracy comparable to ANN due to the similarity between SNN and recurrent neural networks and are trained using time-reversal error propagation (BPTT) [1].

- The architecture of the SNN. While most of the existing work on the SNN focused on the problem of image classification and used available ANN architectures such as VGG or Resnet, the development of a fundamentally new structure specifically for the SNN is an important and one of the primary tasks [1].

Thus, SNN often uses the architecture of deep networks as a basis, which have already shown their suitability for solving individual classification problems, and then this structure adapts to the introduction of neurons with spike properties [16]. At the same time, an important aspect of the SNN is the temporal dynamics, which is more typical for recurrent neural networks. Therefore, in this paper, a fundamentally different approach is proposed, when models of biological neurons will be taken as neurons, and for such a network the possibility of learning and the necessary characteristics will be shown.

In this paper, models of biological neurons based on the example of the FitzHugh-Nagumo system [22, 23] will be used as components of ANN. This will show the applicability of various ANN training methods, which will make it possible to bring the system closer to a real biological system, evaluate the features of implementing biological neuron models into an artificial network, and further refine learning algorithms. In our previous work [24], we showed how the FitzHugh-Nagumo systems can be embedded in a deep network and used to recognize the MNIST [25] handwritten digit database. However, this work did not take into account the features of temporal dynamics. This article discusses another problem, when the learning task itself and its solution are based on the features of spike dynamics of partial systems.

Pulse signals consisting of a sequence of pulses (spikes) generated by a single FitzHugh-Nagumo system under certain parameters will be used for training. A neural network consisting of FitzHugh-Nagumo systems will be trained to further predict this signal. Thus, the task of learning both a recurrent network and a spike neural network, in which memory of previous states is present, is solved simultaneously. For the FitzHugh-Nagumo systems, delayed feedback will act as «memory». The network will use several FitzHugh-Nagumo systems with different delay times, and which neuron and with what force of delayed feedback needs to be activated will be decided during network training.

The choice of FitzHugh-Nagumo systems with delayed feedback is due to the fact that, under certain connection parameters, it can exhibit periodic, quasi-periodic, and even chaotic dynamics [26]. Thus, it allows you to expand the possibilities of generating the network output signal. In this work, several types of periodic pulse signals were used for prediction. In the future, it is planned to expand the research and consider how the network will cope with more complex signals.

The learning process of recurrent neural networks, as a rule, consists in selecting connection matrices between neurons in such a way as to reduce the error (loss function) between the predicted signal and the previously known one. The standard deviation (MSE) or mean absolute error (MAPE) is often used as such an error. However, in this work it was shown that these values are not suitable for the task, since they do not take into account the pulse nature of the signals. Based on this, a combined loss function was proposed, which includes the MSE and the spike interval error.

Gradient descent was used to train the neural network based on the proposed combined loss function. It has been shown that the proposed method works well and makes it possible to predict complex quasi-periodic signals.

1. The system under study

The topology of the system under study is schematically similar to the structure of a recurrent network with a reservoir (echo network). The main component of the network is the so-called reservoir, consisting of models of biological neurons that are set using delayed feedback FitzHugh-Nagumo models. All the parameters of the systems in the reservoir are identical, except for the delay time. The system under consideration is schematically shown in Fig. 1.

In this work, the network will be trained to solve the problem of predicting the input pulse (spike) signal. The input signal supplied to the network is primarily used to set the initial data set for delayed feedback. After «memorization» of all the necessary data from the input signal, the network becomes self-closed, and the input signal is needed only for reconciliation with the output (predicted) signal of the network during the learning process. In the case of a self-closed system, the input signal u^t at time t is used as the output signal of the same network at the previous time z^{t-1}.

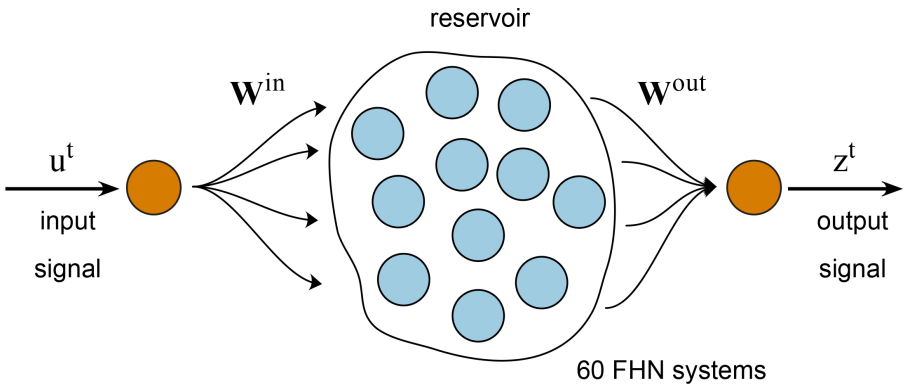

Fig. 1. Schematic representation of considered system (color online)

The neurons are in the reservoir. Each FHN model in the reservoir is defined using the following system of ordinary differential equations:

$$\begin{aligned} \varepsilon\dot{x} &= x - x^3/3 - y + \gamma(x_\tau - x), \\ \dot{y} &= x + a, \end{aligned} \tag{1}$$

where $x = x(t)$ and $y = y(t)$ are fast (activator) and slow-changing (inhibitor) system variables that set the system state at time t, parameters ε and a — these are the parameters of individual PHC systems, the values of ε affect the time scale, and the parameter a is the control parameter, while the values of $|a| < 1$ correspond to the oscillatory mode of operation of the system with spike dynamics, and the values of $|a| > 1$ correspond to the excitable mode in which The system has a stable equilibrium state of the focus type, and spike dynamics can occur only due to external influences, for example, with coherent resonance [27]. In this paper, the excitable mode will be considered, and the values of the parameters $a = 1.01$ and $\varepsilon = 0.05$ will not change throughout the article.

The main fundamentally important component of the system (1) is delayed feedback, implemented by the last term of the first equation: $\gamma(x_\tau - x)$. Here γ is the feedback force, $x_\tau = x(t - \tau)$ determines the value of the variable x at time $(t - \tau)$, and the parameter τ is the delay time. This method of setting delayed feedback is quite standard for the model under

consideration. It was originally proposed by K. Pyragas in 1992 [28] and is often used to model lagged networks [29–31].

In our previous work [26], we considered one FHN system (1) with different parameters of positive and negative feedback. It has been shown that delayed feedback in the excitable mode can lead to fluctuations at certain values of the feedback force γ and the delay time τ. At certain parameter values, one LFN system with delayed feedback can exhibit periodic, quasi-periodic, and even chaotic dynamics. From the point of view of spike dynamics, the time of interspike intervals can be controlled using the delay time and feedback force. Based on the results obtained in [26], we can conclude that the choice of the delay time $\tau \in (0.6]$ and the feedback force $\gamma \in [-0.1, 0.1]$ allows us to obtain a large number of different modes. For this reason, in the present work, there are $N = 60$ PHN neurons inside the reservoir with different delay times from the range $\tau_j \in [0.1, 6]$ with a step of $\Delta\tau = 0.1$. Thus, in the reservoir, all FHN systems are set using the same parameters, with the exception of the delay time τ. The input signal u^t is used to set the initial sequences of values of x_τ for delayed feedback.

Network training. The main purpose of network training is to determine which neurons are important for generating the output signal and what feedback strength they should have. Both problems can be solved using the connection matrices $\mathbf{W}^{\text{in}}$ and $\mathbf{W}^{\text{out}}$, which define the connection between the input artificial neuron (input signal) and the output artificial neuron (output signal) at the current time t. These neurons are shown in orange in Fig. 1. Since there are $N = 60$ neurons in the reservoir, the size of both matrices is $(1 \times N)$. From the point of view of learning, matrices can be set in two ways, and in the case of a self-closed system, these methods mathematically lead to the same result.

- The input matrix is set so that all values of the matrix are equal to one $W_i^{\text{in}} = 1$, and the values of the matrix $\mathbf{W}^{\text{out}}$ are set to zero for neurons that are not important for the formation of the output signal, otherwise — equal to γ, while setting the value of the feedback force for the output signal of the network, which is used as an input at the next moment in time. If all the elements in the output matrix of the connection except the jth are zero, then you can understand the neuron with which delay time is important for generating the output signal.
- The input coupling matrix is given as $\mathbf{W}^{\text{in}} = \gamma \cdot \mathbf{I}$, where $\mathbf{I}$ is a matrix filled with units, and the specific values of γ are selected during training, but are the same for all W_i^{in}. The output connection matrix $\mathbf{W}^{\text{out}}$ is a binary matrix in which the influence of unnecessary reservoir neurons corresponds to zero values, and those neurons that are important for the formation of the output signal correspond to single values in the output matrix. Thus, if only one jth element in the output matrix is equal to one, then it is possible to determine which neuron with which delay time it corresponds.

In this paper, we will use the second method. Then the output signal of the network is given as

$$z^t = z(t) = \sum_{j=1}^{N} W_j^{\text{out}} x_j(t). \tag{2}$$

Thus, the purpose of the training is: a) selecting the value γ responsible for forming the matrix $\mathbf{W}^{\text{in}}$; b) selecting the matrix $\mathbf{W}^{\text{out}}$ so that one of its elements is equal to 1, and the rest — zero.

As can be seen from Fig. 1, the input signal u^t is present in the system. This signal is used first to set the initial values of the lagging signal for all FHN systems, and then to create a self-closed network. In the first case, the input signal is used to specify an array of initial values of x_τ. For each jth FCN system, the length of such an array will be different, since each system has

its own delay time τ_j. After all the data for delayed connection is received, the system becomes self-closed and disconnected from the training input signal. Further in the learning process, it will be used only for reconciliation with the output signal of the z^t network and the selection of connection matrices. In a self-closed system, the input signal of the network u^t continues to be entered into arrays of the delayed signal, but this is no longer a training signal, but the output signal of the same network at the previous time z^{t-1}. During the learning process, the value of the delayed feedback force γ is selected, therefore, in a self-closed system, the input signal is not explicitly included in the equation (1), but the following equality holds:

$$\gamma x_{\tau,j} = W_j^{\text{in}} \cdot u(t-\tau) = W_j^{\text{in}} \cdot u^{t-\tau}. \tag{3}$$

In this work, the neurons inside the reservoir itself do not affect each other, but the input signal is important for each of the neurons. In future work, it is planned to add a connection inside the reservoir.

2. Loss function and required characteristics

2.1. Standard deviation (MSE). The most commonly used characteristic required to estimate the discrepancy between the predicted and target signals is the mean squared error (MSE), which is calculated as follows:

$$\text{MSE}(u,z) = \sum_{t=1}^{T} (z^t - u^{t+1})^2 / T, \tag{4}$$

where z^t is the predicted signal at time t, that is, the output signal of the network at time t, and u^{t+1} is the training (target) signal of the network at the next time. T — the total number of points in the implementation. Due to the fact that the network is trained to predict the signal, it is necessary to compare the output signal of the network z^t with the input (target) u^{t+1}. In the learning process, the target signal is used to compare u, rather than the output of a self-closed network.

In Fig. 2, *a* an example of a network input signal is given, which was set as an implementation of a single FCN system with parameters $\tau = 2$, $\gamma = 0.03$.

To understand the discrepancy with the desired implementation when changing the delay time and feedback strength in Fig. 2, *b* shows the MSE values obtained when comparing the target signal with known parameters and the output signal of the network, depending on the neuron with which delay time was selected for comparison and what feedback force γ should be used when setting the input matrix $\mathbf{W}^{\text{in}}$. Along the horizontal axis of Fig. 2, *b* the delay time τ has been postponed, but the number of the neuron from the reservoir $j = \tau/\Delta\tau$ can be matched to it. Thus, for a perfect match to the desired parameters, it turns out that in the output matrix $\mathbf{W}^{\text{out}}$ all elements must be equal to zero, except for $j = 2/0.1 = 20$, which is equal to 1. The input matrix $\mathbf{W}^{\text{in}}$ should be filled with the values $\gamma = 0.03$.

In Fig. 2, *b* the green circle highlights the correct values of τ and γ, which were used to obtain the target signal. Based on the color gamut, it can be seen that MSE changes non-linearly when the parameters γ and τ are varied. A large dark purple area with a fairly small error corresponds to the absence of fluctuations in the output signal, and the MSE value close to 0 (black rectangle) can only be obtained if you get into the parameters that exactly match the input signal. A spike output signal can also be obtained in the blue areas of the MSE, however, due to

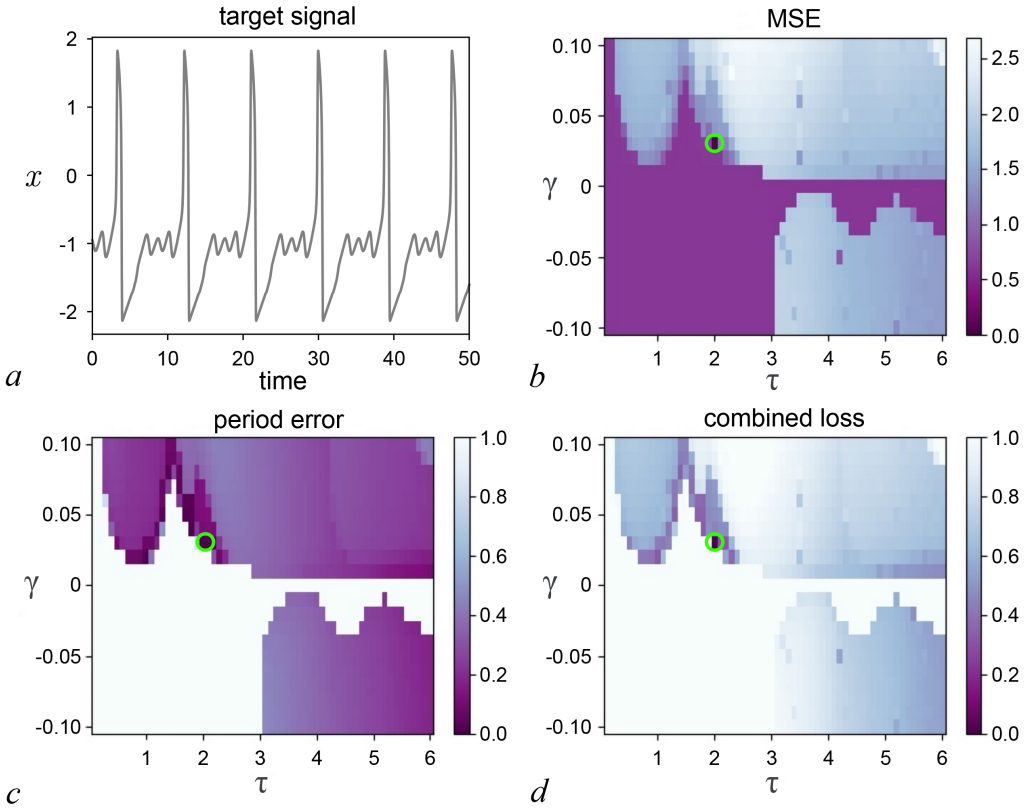

Fig. 2. The input signal to be predicted (a), obtained with parameters $\tau = 2$, $\gamma = 0.03$. For this signal, the mean square error (b), period error (c), and combined loss function (d) were calculated with variation of the coupling matrices that correspond to the predicted signals with some parameters τ and γ. For ease of evaluation and comparison, these characteristics are presented on the planes of these parameters (color online)

a phase failure, the MSE characteristic will not work correctly in this case, even if the difference in periods is minimal. Thus, the use of the MSE characteristic in its pure form is not applicable to the existing task, since in the general case for an unknown signal there is no guarantee that this signal can be set using the selected delay time range in the reservoir, and searching for a similar signal using the selected characteristic is impossible. Another disadvantage of this value is that in the absence of fluctuations (lower left corner in Fig. 2, b) this characteristic is not zero. When using automatic methods for searching for minimum values, in most cases, the minimum MSE with a fixed τ will fall precisely on areas with complete absence of fluctuations.

2.2. Estimation of interspike intervals. In Fig. 2, a shows a signal in which, in addition to the spike component, there are also quasi-periodic fluctuations near the equilibrium state. However, sometimes only the spike component in the signal is fundamentally important, especially the interspike intervals. As it was shown in [26], the spike interval can be controlled using the delay time and the strength of the delayed coupling.

The average spike interval was calculated as follows. Based on the waveform in Fig. 2, an indicator of a spike can be considered the moment when the sequence of values of the x signal ceases to be increasing and becomes decreasing, while in order not to take into account small fluctuations near the equilibrium state, an additional condition is imposed that these values must be strictly positive. Next, the average time is calculated between each such event, which in the future will be called the average spike interval. For the implementation shown in Fig. 2, a, the

spike interval was approximately 8.88.

In Fig. 2, *c* shows the spike interval error for the target signal u and the network output signal z, calculated as follows:

$$\xi(u,z) = \frac{|P_z - P_u|}{P_z + P_u}, \tag{5}$$

where P_z is the average spike interval of the network output signal, P_u is the same value for the target signal. The interspike intervals were calculated starting from time 100 to eliminate the period of oscillation for the network output signal. Calculating the value (5) is similar to calculating the SMAPE error. The choice of this value is due to the fact that this value itself varies in the range from 0 to 1, and the value 1 will correspond either to the complete absence of spikes in the network output signal, or if the interspike intervals are many times different. Values close to 0 will best match between P_u and P_z. It is assumed that the target signal u already contains some spike dynamics, therefore $P_u \neq 0$. However, there is a certain range of parameters γ and τ (lower left quarter in Fig. 2, *b*), for which there is a complete absence of fluctuations. If the error were calculated using the standard method using only P_u in the denominator, the error would be 1. The difference between the values of P_u and P_z several times would also look unrepresentative, since then the error does not have an upper and lower limit. Using the P_z value in the denominator would solve the latter problem, but then the absence of spikes in the output signal would lead to the presence of 0 in the denominator.

In Fig. 2, *c* the maximum error corresponds to white. It can be seen from the figure that in the considered range of parameter values there are several parameter values with a minimum spike interval error. Comparing Fig. 2, *b, c*, it can be seen that this characteristic is better suited for evaluating the behavior of the system, although it only takes into account the spike characteristics of the signals.

2.3. Combined loss function. Usually, the purpose of training neural networks is to select connection matrices to reduce the difference (error function) between the target signal and the output signal of the network. The search for optimal coupling matrices based on the available loss function can be implemented using various methods. The most commonly used is gradient descent.

The root-mean-square error or the average absolute error is often used as an error function in prediction problems. However, as shown above, this error in its pure form is not applicable to the existing task. Therefore, the following combined value will be used to calculate the loss function, which takes into account both interspike intervals and the total error:

$$L(u,z) = \frac{\mathrm{MSE}(u,z)}{\max(u) - \min(u)} + \xi(u,z). \tag{6}$$

The presence of the denominator in the first term is due to the need to normalize the MSE values, which do not have an upper limit. When the parameters τ and γ are varied for one isolated PHN system, the overall spread of the values of x and y remains almost unchanged. Therefore, normalization to the spread of values in the target signal will be sufficient to keep the first term in the range from 0 to 1. If there are no fluctuations in the network signal, then the first term will be 0, and the second term will be 1, so the combined loss function will be 1. If the target signal and the network signal are as close as possible, then both terms, and hence the loss function, will tend to 0.

The combined loss function is shown in Fig. 2, *d*. It can be seen from the graphs that the new error function is still not monotonous, but at the same time there are more options for

searching for similar modes of operation, which allows you to find similar modes even if there is no necessary delay time in the reservoir.

3. Network training using gradient descent

Gradient descent is the most common method of training neural networks. Its meaning is to automatically search for the minimum value of the loss function and the values of the coupling matrices that correspond to it [32]. The purpose of neural network training in this work is to predict the input spike signal and automatically search for τ and γ that may correspond to it. This task involves finding the minimum value of the combined loss function (6) and for which neurons from the reservoir, at what feedback strength γ this minimum is achieved.

The meaning of gradient descent is to gradually descend to the minimum value of the loss function by changing the coupling matrices with a certain step. Depending on the specific method, this step can either be constant or vary depending on the magnitude of the derivative of the loss function. The peculiarity of gradient descent is that in the case of a non-convex loss function, the final result will strongly depend on the initial conditions, since gradient descent can «get» to a local minimum and never get to a global minimum. For this reason, it makes sense to start gradient descent from several initial conditions.

Within the framework of the problem being solved, the application of gradient descent was as follows. In the matrix $\mathbf{W}^{\text{out}}$, one of the values alternated from 0 to 1. Then, for each such configuration, the values in the matrix $\mathbf{W}^{\text{in}}$ were changed. Gradient descent was used for the combined loss function calculated between the target signal and the output signal of the network. The initial conditions for the matrix were set as $\mathbf{W}^{\text{in}} = \gamma_0 \mathbf{I}$, where γ_0 is a certain constant, some values of which will be discussed later. After performing gradient descent, it was possible to determine at what values $\mathbf{W}^{\text{in}}$ the minimum combined loss function could be achieved. After that, another neuron was selected in the matrix $\mathbf{W}^{\text{out}}$, and the procedure was repeated. After iterating through all N neurons, the pair of matrices $\mathbf{W}^{\text{in}}$ and $\mathbf{W}^{\text{out}}$ was selected, which corresponded to the minimum value of the combined loss function.

In Fig. 3, *a* are graphs for the input signal used in the previous sections, obtained with the parameters $\tau = 2$ and $\gamma = 0.03$. For this signal, the values of the combined loss function were obtained depending on the coupling matrices, which corresponded to some values of τ and γ. For convenience, the values of the loss function are shown depending on the selected values of τ and γ in the first graph of Fig. 3, *a*. In this graph, the orange dots show the minimum values found using gradient descent, starting with four initial conditions $\gamma_0 = \pm 0.1, \pm 1$ with a constant step of 0.01. The gradient descent procedure was performed for each of the N reservoir neurons, that is, for each jth neuron, which corresponds to its delay time $\tau = j \cdot \Delta\tau$. Therefore, in Fig. 3, *a* N orange dots. Next, of all the N minimum values of the loss function found, the most minimal can be found, and this will correspond to the network output signal that is most similar to the input signal. For $\tau = 2$ and $\gamma = 0.03$ with the help of gradient descent from four initial conditions, it was possible to perfectly get into the desired parameters. In the second graph, Fig. 3, *a* shows the target signal in gray, and the predicted signal in orange, indicating the selected values of the parameters τ and γ that correspond to it.

Due to the non-linearity of the loss function and the presence of regions with the same values of the combined loss function, in some cases it is necessary to increase the number of initial conditions for gradient descent. To do this, eight initial conditions were considered $\gamma_0 = \pm 0.1$, ± 0.25, ± 0.5, and ± 1. The data obtained in this case is shown as green dots on the left graph in Fig. 3, *a* and the green dotted line on the right graph of Fig. 3, *a*. Since starting from four

initial conditions was already enough for the selected parameters, it was expected that for eight initial conditions it was also possible to accurately get into the desired parameters. However, this does not always work. On the other lines of Fig. 3, *b, c* the results for the input signal with other parameters are shown. The figure shows that in some cases, an increase in the number of initial conditions leads to a significant increase in accuracy. In the left graphs, Fig. 3 the current minimum value of the loss function is marked with a black circle.

In Fig. 3 the values of the parameters that can theoretically be reached using gradient descent with a selected step were considered. To preserve generality, the cases of $\tau = 3.05$ and $\tau = 4.05$, that is, such lag times, which definitely cannot be in the reservoir. In Fig. 4 shows the results of applying both gradient descents in this case. The graphs were obtained in the same way as for Fig. 3. It can be seen from the graphs that the global minimum of the combined loss function, marked with a black circle on the left graphs, has not been reached. However, judging by the time implementations, the coupling matrices selected during gradient descent and the corresponding values of τ and γ were sufficient to specify similar signals. Increasing the number

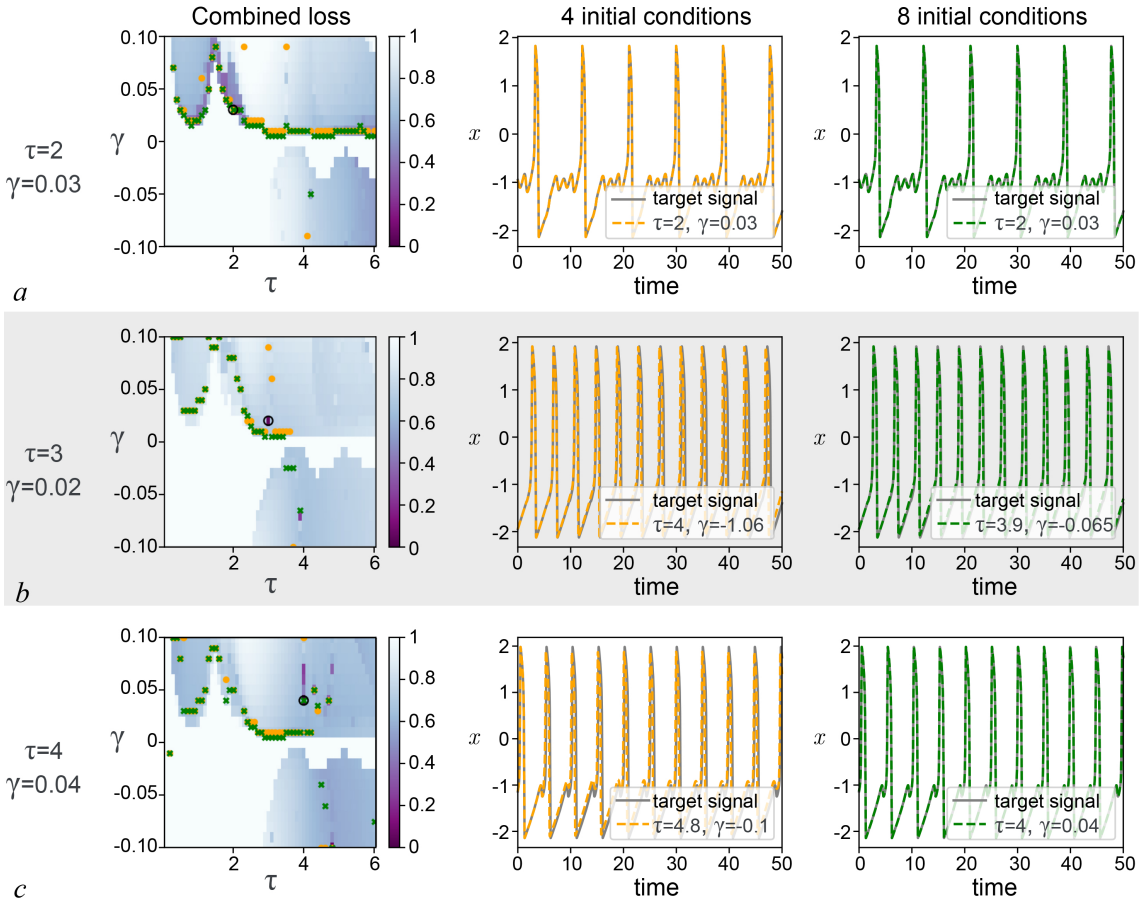

Fig. 3. The use of gradient descent for three input signals obtained with parameters: *a* — $\tau = 2$, $\gamma = 0.03$, *b* — $\tau = 3$, $\gamma = 0.02$, *c* — $\tau = 4$, $\gamma = 0.04$. The graphs on the left demonstrate the values of the combined loss function on the plane of the corresponding parameters τ and γ. The values themselves were obtained by enumeration with a step of 0.01. In the same figure, the orange and green dots show the results of applying gradient descent with 4 and 8 initial conditions, respectively. The middle graphs show the target signal (gray line) and the predicted signal (orange dotted line) obtained using gradient descent with 4 initial conditions, and the legends indicate the parameters τ and γ, reconstructed from the connection matrices of the trained neural network. The right graphs are obtained in a similar way, but using gradient descent from 8 initial conditions (color online)

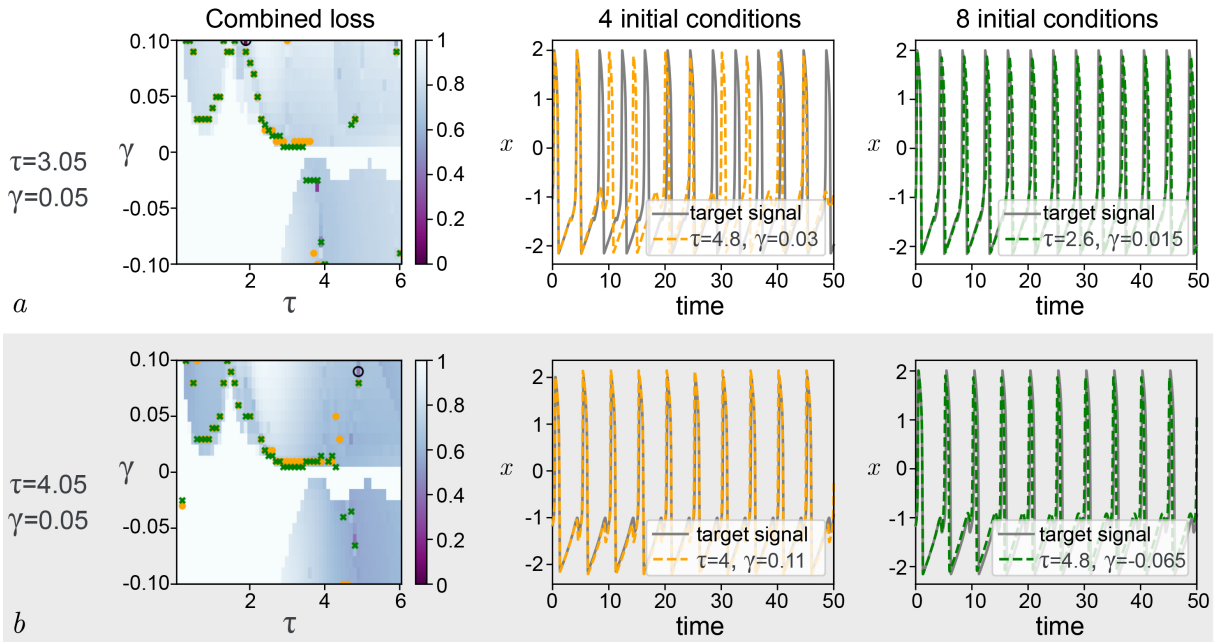

Fig. 4. The use of gradient descent for three input signals obtained with parameters: a — $\tau = 3.05$, $\gamma = 0.05$, b — $\tau = 4.05$, $\gamma = 0.05$. The rest of the construction and structure of the figure is similar to Fig. 3 (color online)

of initial conditions leads to a significant increase in accuracy in this case.

One of the problems with using gradient descent is the presence of areas where the combined loss function does not change. When entering such an area, the gradient descent algorithm stops, and further selection of values becomes impossible without starting from other initial conditions. Therefore, the principle works here: more initial conditions — more precisely, the result. Another problem is the gradient descent step. In this case, in Fig. 3, 4 choosing the step 0.01 was enough to achieve the necessary minima, however, choosing too large a step may lead to the algorithm not reaching the minimum, and too small a step — to a large number of unnecessary calculations. In future work, it is planned to consider the applicability of other learning algorithms, in particular, the use of gradient descent with adaptive step.

Conclusion

In this paper, we solved the problem of predicting a complex quasi-periodic spike signal using a recurrent neural network with reservoir models of biological neurons using the example of FitzHugh-Nagumo systems with delayed feedback. A training methodology has been proposed that combines standard methods for training artificial neural networks and how it can be adapted to FitzHugh-Nagumo systems. The general idea is that $N = 60$ FitzHugh-Nagumo systems with different delay times $\boldsymbol{\tau}_j$ were used as reservoir neurons. In the learning process, it is selected which of the neurons needs to be activated and with which feedback force. This paper suggests how this information can be embedded in the $\mathbf{W}^{\text{in}}$ and $\mathbf{W}^{\text{out}}$ connection matrices standard for artificial neural networks.

To train the network, a combined loss function was used, which includes both the spike interval error and the total MSE error. Next, gradient descent was applied to the resulting loss function, which allowed us to find the minimum values of the loss function, which connection matrices $\mathbf{W}^{\text{in}}$ and $\mathbf{W}^{\text{out}}$ correspond to them, and from this the optimal delay time $\boldsymbol{\tau}$ and the

feedback force γ.

The proposed methods have been used for several types of complex spike signals. Input signals with known parameters τ and γ were considered both for cases when there is a neuron in the reservoir with an ideally suitable delay time (Fig.3) and for cases when there is no suitable delay time in the reservoir (Fig.4). In both cases, the proposed technique helped to find the most similar signals. In this paper, it was shown that gradient descent can be successfully applied to the task, however, due to the noticeable nonlinearity of the loss function, it is necessary to use several initial conditions of gradient descent. The more initial conditions, the higher the accuracy. Improvements in the learning methodology can be achieved by using stochastic gradient descent, adaptive step gradient descent, or possibly using other network learning techniques. The comparison and application of these methods will be the purpose of the research in the following works.

References

1. Yamazaki K, Vo-Ho V-K, Bulsara D, Le N. Spiking neural networks and their applications: A review. Brain Sci. 2022;12(7):863. DOI: 10.3390/brainsci12070863.
2. Benjamin BV, Gao P, McQuinn E, Choudhary S, Chandrasekaran AR, Bussat JM, Alvarez-Icaza R, Arthur JV, Merolla PA, Boahen K. Neurogrid: A Mixed-Analog-Digital Multichip System for Large-Scale Neural Simulations. Proceedings of the IEEE. 2014;102(5):699–716. DOI: 10.1109/JPROC.2014.2313565.
3. Schuman CD, Potok TE, Patton RM, Birdwell JD, Dean ME, Rose GS, Plank JS. A survey of neuromorphic computing and neural networks in hardware. arXiv:1705.06963. arXiv Preprint, 2017. DOI: 10.48550/arXiv.1705.06963.
4. Ghimire D, Kil D, Kim S. A survey on efficient convolutional neural networks and hardware acceleration. Electronics. 2022;11(6):945. DOI: 10.3390/electronics11060945.
5. Aguirre F, Sebastian A, Le Gallo M, Song W, Wang T, Yang JJ, Lu W, Chang M-F, Ielmini D, Yang Y, Mehonic A, Kenyon A, Villena MA, Roldán JB, Wu Y, Hsu H-H, Raghavan N, Suñé J, Miranda E, Eltawil A, Setti G, Smagulova K, Salama KN, Krestinskaya O, Yan X, Ang K-W, Jain S, Li S, Alharbi O, Pazos S, Lanza M. Hardware implementation of memristor-based artificial neural networks. Nat. Commun.. 2024;15(1): 1974. DOI: 10.1038/s41467-024-45670-9.
6. Chen Y, Nazhamaiti M, Xu H, Meng Y, Zhou T, Li G, Fan J, Wei Q, Wu J, Qiao F, Fang L, Dai Q. All-analog photoelectronic chip for high-speed vision tasks. Nature. 2023;623(7985): 48–57. DOI: 10.1038/s41586-023-06558-8.
7. *Brunner D, Soriano MC, Mirasso CR, Fischer I.* Parallel photonic information processing at gigabyte per second data rates using transient states. Nat. Commun. 2013;4:1364. DOI: 10.1038/ ncomms2368.
8. McMahon PL. The physics of optical computing Nat. Rev. Phys. 2023;5(12):717–734. DOI: 10.1038/s42254-023-00645-5.
9. Tuma T, Pantazi A, Le Gallo M, Sebastian A, Eleftheriou E. Stochastic phase-change neurons. Nature Nanotechnology. 2016;11:693–699. DOI: 10.1038/nnano.2016.70.
10. Torrejon J, Riou M, Araujo FA, Tsunegi S, Khalsa G, Querlioz D, Bortolotti P, Cros V, Yakushiji K, Fukushima A, Kubota H, Yuasa S, Stiles MD, Grollier J. Neuromorphic computing with nanoscale spintronic oscillators. Nature. 2017;547(7664):428–431. DOI: 10.1038/nature23011.
11. Ponulak F, Kasinski A. Introduction to spiking neural networks: Information processing, learning and applications. Acta Neurobiologiae Experimentalis. 2011;71(4):409—433. DOI: 10.55782/ane-2011-1862.

12. Ghosh-Dastidar S, Adeli H. Spiking neural networks. International Journal of Neural Systems. 2009;19(4):295–308. DOI: 10.1142/S0129065709002002.
13. Merolla PA, Arthur JV, Alvarez-Icaza R, Cassidy AS, Sawada J, Akopyan F, Jackson BL, Imam N, Guo C, Nakamura Yu, Brezzo B, Vo I, Esser SK, Appuswamy R, Taba B, Amir A, Flickner MD, Risk WP, Manohar R, Modha DS. A million spiking-neuron integrated circuit with a scalable communication network and interface. Science. 2014;345(6197):668–673. DOI: 10.1126/science. 1254642.
14. Davies M, Srinivasa N, Lin T-H, Chinya G, Cao Y, Choday SH, Dimou G, Joshi P, Imam J, Jain S, Liao Y, Lin C-K, Lines A, Liu R, Mathaikutty D, McCoy S, Paul A, Tse J, Venkataramanan G, Weng Y-H, Wild A, Yang Y, Wang H. Loihi: A Neuromorphic manycore processor with On-Chip Learning. IEEE Micro. 2018;38(1):82–99. DOI: 10.1109/MM.2018.112130359.
15. Furber SB, Galluppi F, Temple S, Plana LA. The SpiNNaker Project. Proceedings of the IEEE. 2014;102(5):652–665. DOI: 10.1109/JPROC.2014.2304638.
16. Tavanaei A, Ghodrati M, Kheradpisheh SR, Masquelier T, Maida A. Deep learning in spiking neural networks. Neural Networks. 2019;111:47–63. DOI: 10.1016/j.neunet.2018.12.002.
17. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 1998;86(11):2278–2324. DOI: 10.1109/5.726791.
18. Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images. Technical Report. Toronto: University of Toronto; 2009. Available from: http://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf.
19. Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: A novel image Dataset for benchmarking machine learning algorithms. arXiv:1708.07747. arXiv Preprint, 2017. DOI: 10.48550/arXiv.1708.07747.
20. Nunes JD, Carvalho M, Carneiro D, Cardoso JS. Spiking Neural Networks: A survey. IEEE Access. 2022;10:60738–60764. DOI: 10.1109/ACCESS.2022.3179968.
21. Han B, Roy K. Deep Spiking Neural Network: Energy Efficiency Through Time Based Coding. In: Vedaldi A, Bischof H, Brox T, Frahm JM, editors. Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science. Vol. 12355. Cham: Springer; 2020. P. 388–404. DOI: 10.1007/978-3-030-58607-2_23.
22. FitzHugh R. Impulses and physiological states in theoretical models of nerve membrane. Biophys. J. 1961;1(6):445–466. DOI: 10.1016/S0006-3495(61)86902-6.
23. Nagumo J, Arimoto S, Yoshizawa S. An active pulse transmission line simulating nerve axon. Proceedings of the IRE. 1962;50(10):2061–2070. DOI: 10.1109/JRPROC.1962.288235.
24. Bogatenko T, Sergeev K, Slepnev A, Kurths J, Semenova N. Symbiosis of an artificial neural network and models of biological neurons: Training and testing. Chaos. 2023;33(7):073122. DOI: 10.1063/5.0152703.
25. LeCun Y, Cortes C, Burges CJC. The MNIST database of handwritten digits [Electronic resource] Available from: http://yann.lecun.com/exdb/mnist/.
26. Semenov VV, Bukh AV, Semenova N. Delay-induced self-oscillation excitation in the Fitzhugh-Nagumo model: Regular and chaotic dynamics. Chaos, Solitons & Fractals. 2023; 172:113524. DOI: 10.1016/j.chaos.2023.113524.
27. Pikovsky AS, Kurths J. Coherence resonance in a noise-driven excitable system. Phys. Rev. Lett. 1997;78(5):775–778. DOI: 10.1103/PhysRevLett.78.775.
28. Pyragas K. Continuous control of chaos by self-controlling feedback. Physics Letters A. 1992; 170(6):421–428. DOI: 10.1016/0375-9601(92)90745-8.
29. Schöll E, Hiller G, Hövel P, Dahlem MA. Time-delayed feedback in neurosystems. Phil. Trans. R. Soc. A. 2009;367(1891):1079–1096. DOI: 10.1098/rsta.2008.0258.

30. Just W, Pelster A Schanz M, Schöll E. Delayed complex systems: an overview. Phil. Trans. R. Soc. A. 2010;368(1911):303–304. DOI: 10.1098/rsta.2009.0243.
31. Masoliver M, Malik N, Schöll E, Zakharova A. Coherence resonance in a network of FitzHugh-Nagumo systems: Interplay of noise, time-delay, and topology. Chaos. 2017;27(10): 101102. DOI: 10.1063/1.5003237.
32. Ruder S. An overview of gradient descent optimization algorithms. arXiv:1609.04747. arXiv Preprint, 2017. DOI: 10.48550/arXiv.1609.04747.