Physics – Uspekhi **65** (10) 1020 – 1038 (2022)

REVIEWS OF TOPICAL PROBLEMS

PACS numbers: 07.05.Mh, **84.35.+i**, 87.19.L –

Nonlinear dynamics and machine learning of recurrent spiking neural networks

O V Maslennikov, M M Pugavko, D S Shchapin, V I Nekorkin

DOI: https://doi.org/10.3367/UFNe.2021.08.039042

Contents

Abstract. **Major achievements in designing and analyzing recurrent spiking neural networks intended for modeling functional brain networks are reviewed. Key terms and definitions employed in machine learning are introduced. The main approaches to the development and exploration of spiking and rate neural networks trained to perform specific cognitive functions are presented. State-of-the-art neuromorphic hardware systems simulating information processing by the brain are described. Concepts of nonlinear dynamics are discussed, which enable identification of the mechanisms used by neural networks to perform target tasks.**

Keywords: artificial neural networks, nonlinear dynamics, machine learning, spiking neurons, modeling of cognitive functions

1. Introduction

The use of nonlinear dynamic methods to study processes in neural structures of the brain has a fairly long history and goes back to the pioneering explorations by English electrophysiologists E Hodgkin and E Huxley of nerve impulse conduction. To date, significant progress has been made in this area. Nonlinear dynamic models of many types of neurons have been developed, and the dynamics of both various ensembles consisting of a relatively small number of interacting neurons and large neuromorphic systems have been studied [1–6].

In *Physics Uspekhi*, topics related to the application of nonlinear physics approaches to the study of dynamic processes in the brain and neural ensembles have been presented in a number of reviews [7–18]. Lately, modeling the cognitive and functional properties of the working brain have come to the forefront of neurodynamics [19–21]. In particular, growing interest in this topic is associated with the creation of artificial intelligence systems that reproduce the key properties of natural intelligence [22, 23]. To solve such problems, it is necessary to build new dynamic models that can reproduce, first, a complex hierarchical organization and, second, the plasticity of neuron structures, as their composition and connections between and within structures change depending on the presence or absence of information inputs.

To date, two approaches to dynamic modeling have been developed [24, 25]. In one of them, the so-called top-down approach, the models operate with brain activity modes — integral variables that imitate high-level processes in the brain [20]. In the other approach, bottom-up, to models of neural structures that can reproduce the higher functions of the brain, first, models of individual neurons are built based on a realistic description of connections between neurons and structures [25, 26].

It is clear that biologically relevant models of both approaches should be based on experimental data. In the experimental study of the brain by neurophysiologists, the activity of neurons is recorded in the state of rest of the subject or when the subject performs a certain task. A model based on experimental data can be developed in two ways. The first is data-driven modeling, when a dynamic system is reconstructed that produces time series quantitatively close to those recorded experimentally. The second way is modeling based on the behavioral problems under consideration, when

O V Maslennikov [(*)], **M M Pugavko, D S Shchapin, V I Nekorkin**
Federal Research Center Institute of Applied Physics,
Russian Academy of Sciences,
ul. Ul'yanova 46, 603950 Nizhny Novgorod, Russian Federation
E-mail: [(*)]olmaov@ipfran.ru

Received 1 June 2021, revised 13 August 2021
Uspekhi Fizicheskikh Nauk **192** (10) 1089 – 1109 (2022)
Translated by M Zh Shmatikov

data on the transformation of sensory inputs into certain motor outputs are recorded during the experiment, and the task is to restore a dynamic system that performs similar transformations of inputs into outputs [27].

State-of-the-art methods for studying the brain, positron emission tomography (PET), electroencephalography (EEG), magnetoencephalography (MEG), near infrared spectroscopy (nIRS), all of which feature a high time resolution, and magnetic resonance imaging (MRI), which has good spatial and satisfactory time resolution, have made it possible to form immense databases.

However, the direct use of experimental data in dynamic models is an extremely challenging problem. Now, as a way out of this controversial situation, computer technologies of machine learning have begun to be actively used to find hidden patterns of converting input signals into output ones [28–31]. In this area, in our opinion, the construction and training of models based on *recurrent spiking neural networks* are especially promising [32–36]. Models of this type have analogies with brain networks in two essential aspects. The *recurrent architecture* inherent in most parts of the brain is characterized by the presence of feedback loops, which allows the network not only to be directly influenced by input stimuli but also to respond to its own behavior in the past. *Spiking dynamics*, i.e., the ability to generate action potentials or spikes, is one of the most important properties of neurons, which underlies communication between neurons and the processing of large amounts of data. Training such a network yields a dynamic system, which can, on the one hand, be studied numerically using various input data and under various external conditions, and, on the other hand, be analyzed by nonlinear dynamics methods. The study of such neural networks is also of great interest for the currently emerging branch of microelectronics associated with the creation of specialized neurochips and neuroprocessors [37–39]. It is assumed that such devices, due to the principles of operation of the neural structures of the brain incorporated in them, will be effective in solving not only conventional problems of recognition and coding but also cognitive and behavioral tasks.

Our review is dedicated to this promising topic. Section 2 contains the basic concepts of machine learning, which are necessary for the subsequent presentation. Section 3 presents examples of modeling the functional and cognitive properties of brain structures using recurrent neural networks. Issues of modeling spiking networks are considered in Section 4. The main achievements in constructing artificial neuromorphic systems are described in Section 5. In the final Section 6, the main conclusions are presented and further promising areas are discussed.

2. Basics of machine learning

Machine learning is a vast area of modern science, and this review does not aim to cover all of its aspects [22, 40]. This section only discusses the key concepts and machine learning algorithms necessary for the further presentation of the main results of their application to the construction of recurrent spiking neural networks for modeling the functional properties of brain networks.

2.1 Three components of machine learning

The procedure for constructing an artificial neural network that models the functional properties of brain networks usually includes the following steps. First, the neurobiological phenomenon under study is formulated as a target function that transforms input stimuli into output responses in a certain way. Second, a basic artificial neural network is initialized, which takes into account the known anatomical aspects of the biological prototype. Third, the network is trained, i.e., its parameters are modified, after which the model becomes capable of generating output responses in a way qualitatively or even quantitatively similar to the generation of responses in a real system, based on input stimuli. The trained network is a dynamic system that can be studied by nonlinear dynamics methods, finding the population mechanisms of the performed function of converting input signals into output responses.

Thus, in developing any artificial neural network, it is required to determine three key components: (1) the target function or learning task; (2) the network architecture, i.e., the structure of connections between neurons; and (3) the learning algorithm, i.e., the rules by which network weight coefficients change.

2.1.1 Task of learning. Modern machine learning systems can be divided into three main classes according to the type of problem being solved: supervised learning, unsupervised learning, and reinforcement learning [41]. In this review, we mainly deal with issues related to the first class, but in this section we also outline the features of the other two classes.

Supervised learning. The supervised learning problem includes two sets of vectors: input $\mathbf{x}^{(i)}$, $i = 1, \ldots, M$ and target $\mathbf{z}_{\mathrm{targ}}^{(i)}$. Each vector of the first set is associated with a vector from the second set, i.e., the entire dataset consists of matched 'input–target output' pairs $\{\mathbf{x}^{(i)}, \mathbf{z}_{\mathrm{targ}}^{(i)}\}$. An artificial neural network maps inputs into outputs $F: \mathbf{x} \to \mathbf{z}$. In general terms, the learning task is to adjust the network parameters, which we denote as $\boldsymbol{\theta}$, in such a way that in response to some input stimulus $\mathbf{x}^{(i)}$, the network generates an output $\mathbf{z}^{(i)} = F(\mathbf{x}^{(i)}, \boldsymbol{\theta})$ that is as close as possible to the target signal $\mathbf{z}_{\mathrm{targ}}^{(i)}$, i.e., $\mathbf{z}^{(i)} \approx \mathbf{z}_{\mathrm{targ}}^{(i)}$. In other words, the learning process minimizes the value of the error function $E = 1/M \sum_i E(\mathbf{z}^{(i)}, \mathbf{z}_{\mathrm{targ}}^{(i)})$, where $E(\mathbf{z}^{(i)}, \mathbf{z}_{\mathrm{targ}}^{(i)})$ is a quantitative measure of the difference between the target signal $\mathbf{z}_{\mathrm{targ}}^{(i)}$ and the output $\mathbf{z}^{(i)}$. To this end, the initial data set is usually divided into two unequal sets: a larger one, the training set, and a smaller one, which includes the remaining data and is referred to as the validation set. Minimization of the neural network error is carried out using training set vectors, after which the quality of training, i.e., the magnitude of the error, is evaluated using the validation set data.

For example, in an image classification problem, the input is an image represented as a vector that contains information about the color of individual pixels, while the target output is a vector that specifies the class to which the imaged object belongs. In cognitive neuroscience problems, the target output may be a certain behavioral act performed by an animal under given conditions that determine the input. For example, in the perceptual decision-making problem, the experiment is set up in such a way that the test animal, which perceives images of randomly moving points from a monitor, should determine whether the ensemble of points is predominantly moving to the right or left by pressing one of two buttons. In terms of artificial neural networks, this problem can be formulated as follows: at each moment of time, a vector containing the coordinates of moving points is supplied to the input, and the target output is the sign of the average speed of the ensemble.

Reinforcement learning. In reinforcement learning problems, an agent interacts with the environment, the state of which is described by the variable $\mathbf{s}_t$, at time t. In natural experiments, a virtual labyrinth can also be chosen as a medium. The agent, interacting with the environment, receives an observation $\mathbf{o}_t$, performs an action a_t that changes the state of the environment to a new one $\mathbf{s}_{t+1}$, and receives a reward or reinforcement for this action, described by a scalar value r_t. For example, an artificial agent moves in a virtual maze, receives visual images in the form of pixel images as observations $\mathbf{o}_t$, performs actions a_t to move in the maze, and receives reinforcement if it manages to find a way out of the maze. The goal is to perform a sequence of appropriate actions, given past and present observations, in such a way as to maximize the total reinforcement $\sum_t r_t$. In many classical reinforcement-learning problems, observations $\mathbf{o}_t$ coincide with environment states $\mathbf{s}_t$, i.e., contain complete information about the environment.

Reinforcement learning theory is used in neuroscience to study decision-making problems. It is employed to analyze how the agent's behavior adapts over time to changing conditions and what neuronal mechanisms underlie certain value-based behavior. The use of artificial neural networks makes deep reinforcement-learning even more flexible and allows, in principle, simulation of most of the tasks set out in experiments on laboratory animals, since such animals almost always learn to perform certain commands through rewards [42, 43].

Unsupervised learning. In the case of unsupervised learning, the network only receives input stimuli $\mathbf{x}^{(i)}$, and the error function is set depending on the inputs and network parameters $E(\mathbf{z}^{(i)}, \boldsymbol{\theta})$. Unsupervised learning includes problems related to finding hidden structures and patterns in unlabeled data, for example, the clustering problem (grouping inputs into various clusters) or dimensionality reduction.

2.1.2 Architecture of networks. To date, numerous types of neural network architectures have been developed, including multilayer deep learning networks, convolutional networks, networks with long short-term memory, recurrent networks, and hybrid networks. Here, we describe two basic types of architecture: feedforward multilayer networks and recurrent networks, which are used in modeling the functional properties of brain networks. In the former, data are transmitted across neurons in one direction — from inputs to outputs, while in the latter, there is feedback, so the data can repeatedly return to the original neurons.

Feedforward networks. Multilayer feedforward networks, which are the most common type of network structure in the field of machine learning, consist of a sequence of layers, within each of which neurons are not connected to each other. The network usually has input and output layers, and one (or more) internal or deep layers. The input stimulus is processed sequentially starting from the input layer; the neurons of the previous layer are directly connected with the neurons of the next one. Thus, neurons are activated in a layer-by-layer way from the input layer to the output one. The basic type of feedforward network consists of N layers of neurons; the neurons of the lth layer receive inputs from the neurons of the $(l-1)$th layer and send outputs to the $(l+1)$th layer:

$$\mathbf{r}^{(1)} = \mathbf{x}\,, \tag{1}$$

$$\mathbf{r}^{(l+1)} = \sigma\left(\mathbf{W}^{(l)}\mathbf{r}^{(l)} + \mathbf{b}^{(l)}\right), \quad 1 < l < N\,, \tag{2}$$

$$\mathbf{z} = \mathbf{W}^{(N)}\mathbf{r}^{(N)} + \mathbf{b}^{(N)}\,, \tag{3}$$

where $\mathbf{x}$ is the input stimulus, $\mathbf{r}^{(l)}$ is the activity of neurons in the lth layer, $\mathbf{W}^{(l)}$ is the matrix of connections from the lth layer neurons to the neurons of the $(l+1)$th layer, and $\mathbf{z}$ is the output response. The activation function $\sigma(\ldots)$ converts the weighted sum of the inputs from the previous layer into the output response of the next layer. The output of the entire network $\mathbf{z}$ is read through the output connections $\mathbf{W}^{(N)}$. Parameters $\mathbf{b}^{(l)}$ and $\mathbf{b}^{(N)}$ are displacement vectors for neurons in the deep and output layers, respectively.

Recurrent networks. An important class of artificial neural networks is recurrent networks, which, due to the presence of feedback, can process data over time. In the basic model of a recurrent neural network, the activity of neurons $\mathbf{r}_t$ at the moment t is determined by external stimuli $\mathbf{x}_t$ input through the matrix of input connections $\mathbf{W}_x$ and the matrix of recurrent connections $\mathbf{W}_r$:

$$\mathbf{r}_t = \sigma\left(\mathbf{W}_x\mathbf{x}_t + \mathbf{W}_r\mathbf{r}_{t-1} + \mathbf{b}_r\right). \tag{4}$$

The output of the network $\mathbf{z}_t$ is determined through the matrix of output connections $\mathbf{W}_z$:

$$\mathbf{z}_t = \mathbf{W}_z\mathbf{r}_t + \mathbf{b}_z\,. \tag{5}$$

A recurrent neural network can be deployed in time and represented as a special form of a multi-layer feedforward network, in which the tth layer receives inputs from the $(t-1)$th layer and additionally from the input layer $\mathbf{x}_t$.

2.1.3 Learning algorithm. The main learning algorithm in the class of supervised learning problems is stochastic gradient descent. For a feedforward network, the error backpropagation algorithm can be described as follows. In the direct passage of the network, for each neuron, the input $\mathbf{x}$ is calculated, which is equal to the weighted sum of the outputs of the previous layer elements. Then, a nonlinear function is calculated, the value of which gives the output of this neuron. This procedure occurs layer by layer starting from the input layer and continues further, passing through deep layers, up to the output layer. Comparing the network output with the target value makes it possible to calculate the error function E. In the reverse pass, at each deep layer, the partial derivatives of the error function E with respect to the outputs of all elements in the layer are calculated. Each such derivative is a weighted sum of the partial derivatives of the error function with respect to the total inputs of the previous layer.

Thus, to adjust the weight coefficients, the learning algorithm calculates the error gradient vector, the components of which show the extent to which the error increases or decreases if one weight or another is slightly increased. The parameter vector $\boldsymbol{\theta}$ is then modified in the opposite direction of the error gradient $\partial E/\partial\boldsymbol{\theta}$. Since estimating the error over the entire training sample is a computationally expensive procedure, the error is often calculated based on a mini-sample: some small number $K \ll M$ of randomly selected elements $\{\mathbf{x}^{(k)}\}$ with indices k from the set $\mathbb{B} = \{k_1, \ldots, k_K\}$:

$$E_{\text{batch}} = \frac{1}{K}\sum_{k\in\mathbb{B}} E\left(\mathbf{z}^{(k)}, \mathbf{z}^{(k)}_{\text{targ}}\right). \tag{6}$$

To reduce the error, the modified parameters $\boldsymbol{\theta}$ are changed in the direction opposite to the gradient, by an amount proportional to the so-called learning rate η ($\eta > 0$):

$$\Delta\boldsymbol{\theta} = \eta \frac{\partial E}{\partial \boldsymbol{\theta}} \,. \tag{7}$$

Usually, the $\mathbf{W}$ and $\mathbf{b}$ parameters are modified,

$$\mathbf{W}^{\text{new}} = \mathbf{W} - \eta \nabla_{\mathbf{W}} E(\mathbf{W}, \mathbf{b}), \quad \mathbf{b}^{\text{new}} = \mathbf{b} - \eta \nabla_{\mathbf{b}} E(\mathbf{W}, \mathbf{b}) \,, \tag{8}$$

while other parameters, called hyperparameters, are initialized and do not change during the learning process. The key condition for calculating the error gradient is the differentiability of the activation functions $\sigma(\ldots)$ (see Eqns (2) and (4)).

2.2 Models of neurons

2.2.1 Static neurons. In conventional artificial neural networks, the basic element is a static neuron, i.e., an element without its own dynamics, which implements an activation function. Its input is a weighted sum of the outputs $z_i = \sum_j w_{ij} y_j$ of other neurons, and it performs a nonlinear transformation, yielding at the output the value $y_i = f(z_i)$. As a nonlinear function, we have various types of sigmoid, hyperbolic tangent $f(z) = \tanh z$, logistic function $f(z) = 1/[1 + \exp(-z)]$, or piecewise linear function ReLu (the so-called rectifying linear unit represented as $f(z) = \max(0, z)$). Suchlike neurons in recurrent networks are sometimes called rate neurons, since the value of the variable is interpreted as the average rate of the spikes, although the spikes themselves are not generated in the model [44–46]. A network of N such neurons can be described by continuous-time equations:

$$\tau \frac{\mathrm{d}x_i}{\mathrm{d}t} = -x_i + \sum_{j=1}^{N} J_{ij} \phi(x_j) \,, \quad i = 1, 2, \ldots, N \,, \tag{9}$$

where x_i are variables that describe the activity of the corresponding neuron, $\phi(x)$ is the average rate of the neuron, defined as a nonlinear function of its activity, the structure of connections is described by the adjacency matrix J_{ij}, and τ is the time constant. The dynamics in networks of static neurons manifest themselves in the case of precisely recurrent networks, since the presence of feedback indicates a dynamic system even in the absence of activity in individual nodes-neurons. In contrast, in multilayer deep learning networks consisting of static neurons, the process of layer-by-layer transmission of activation occurs beyond the time dimension. That is why, in tasks that require the processing of information presented in the form of packages alternating in time, recurrent neural networks are used. Problems whose solution is based on the use of recurrent networks of static neurons are reviewed in Section 3.

2.2.2 Dynamic neurons. Dynamic models of neurons include all systems in the form of differential equations or point mappings that describe a change in the activity of an individual neuron over time [14, 47]. These systems include models such as Hodgkin–Huxley, Morris–Lecar, FitzHugh–Nagumo, and Hindmarsh–Rose. However, to model the functional properties of brain networks, simplified spiking models, such as integrate-and-fire neurons, are most often used. This class of models, which makes it possible to describe the generation of action potentials, or spikes, includes a number of various modifications: first of all, the conventional leaky integrate-and-fire neuron [48], the quadratic [49, 50], exponential [51] integrate-and-fire neurons, the theta-neuron [52], and the adaptive exponential neuron [53]. A simplification common for this class of models is that they ignore the form of the action potential, and each spike is an event only determined by the moment of its occurrence. The dynamics of the membrane potential of a leaky integrate-and-fire type of neuron is described by a first-order differential equation:

$$\tau_{\mathrm{m}} \frac{\mathrm{d}v_i}{\mathrm{d}t} = v_0 - v_i + RI(t) \,, \quad i = 1, \ldots, N \,, \tag{10}$$

where v_i is a variable describing the neuron membrane voltage and τ_{m} is a time constant. When the variable reaches the threshold value of the potential v_{thr} at the moment t_{f} (assuming $\mathrm{d}v/\mathrm{d}t|_{t=t_{\mathrm{f}}} > 0$), a spike is generated, and the potential is reset to the resetting value v_0 (it is often assumed that $v_0 = 0$). The application of spiking models in recurrent neural networks is discussed in Section 4.

3. Recurrent neural networks in tasks of modeling the functional properties of brain networks

Recurrent neural networks, unlike feedforward networks, are characterized by the property that each neuron can receive input from any other neuron in the network. Thus, the activity of neurons at a given moment is influenced not only by an external stimulus but also by the state of the network during a whole time interval in the past. In computational neuroscience, recurrent networks are used to model the functional properties of brain networks associated with motor or cognitive processes. The capacity of such networks to perform as models of neural structures in the brain is justified by their anatomy and physiology, since the output connections of most cortical neurons are inputs for other cortical neurons, i.e., typical cortical populations are recurrent in their structure [54]. Moreover, the dynamics of artificial recurrent networks in many aspects reproduces the spontaneous activity observed in the brain [55].

Artificial recurrent neural networks are trained on specific tasks, designed based on experiments with animals conducted by neuroscientists. Such experiments usually study the properties of neuronal populations of the brain that underlie the execution of various cognitive or motor tasks. The task performed by the test animal in the course of the experiment is formalized as a target function that transforms the input stimuli into the required output responses. Then, an artificial neural network is set, the structure of which can be selected taking into account anatomical features. Further, using machine learning methods [40, 56], the parameters of this network are tuned to perform the target transformation of inputs into outputs.

This formulation of the problem represents a new methodology for obtaining network models, an alternative to the approach that prevailed for a long time, in which the construction of a mathematical model of a functional neural network was entirely based on the researcher's intuition. However, the artificial network obtained in the course of training is a 'black box' with an a priori unknown mechanism responsible for the fulfillment of its target tasks [57]. An analysis of a trained recurrent network, kind of its reverse engineering, can shed light on the mechanisms of a particular function at the level of both individual neurons and population activity.

It should be noted that the resulting neural network can be investigated using the methods of nonlinear dynamics and the

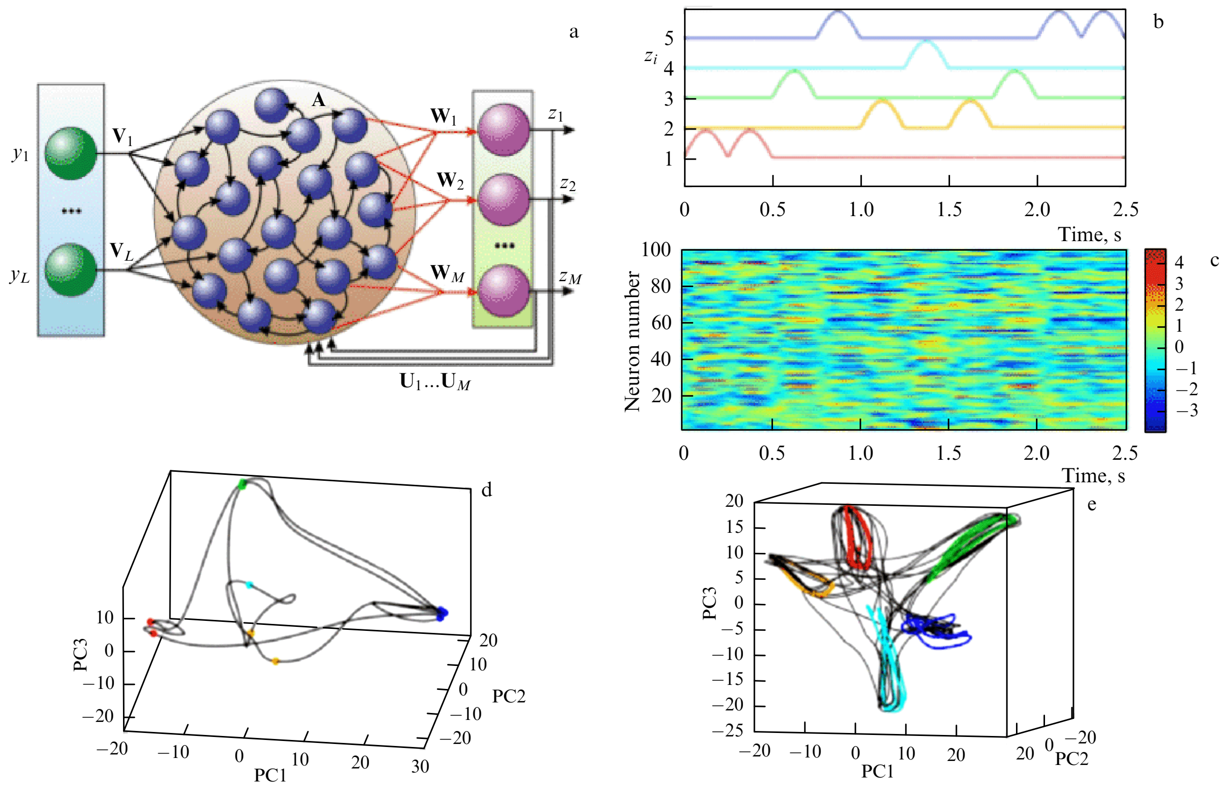

Figure 1. (Color online.) (a) Diagram of a trained recurrent neural network. (b) Example of target space-time pattern that the network generates as an output after successful training. (c) Corresponding dynamics of recurrent network neurons that underlie autonomous generation of the specified pattern. (d) Projection of the multidimensional activity of the neural network onto the space of three principal components. (e) Similar projection when the network converts short input stimuli into output responses on the corresponding output element. (Modified figure from [58].)

theory of complex networks [58–61]. In particular, special trajectories can be identified in a multidimensional phase space that correspond to the execution of a particular function, and the distribution of weights, clustering of connections, and presence of modules in the structure of a trained network can be analyzed [62]. In addition, for a better comparison of biological and artificial neural networks, the latter is often analyzed using methods similar to those applied in the analysis of experimental neurophysiological data. For example, dimensionality reduction methods, such as principal component analysis, are commonly used to identify low-dimensional subspaces in a multidimensional data space, in the projection onto which the variance of data is preserved to a greater extent. To identify cause-and-effect relationships between the structure of the network and its functions, both in the experiment and in the corresponding model, the activity of a certain group of neurons can be stimulated or suppressed, or connections between individual neurons or neural ensembles can be disrupted.

We consider first the problem associated with training a recurrent network of rate neurons to produce a target space-time pattern at the output. The corresponding system, schematically presented in Fig. 1a, consists of a layer of input neurons, the recurrent network itself, and a layer of output neurons. This model describes the process of generating motor and other rhythmically coordinated activities by neural networks of the motor and premotor cortex.

The system of rate neurons is described by the equation

$$\dot{\mathbf{x}} = -\mathbf{x} + \mathbf{A}\mathbf{r} + \mathbf{U}\mathbf{z} + \mathbf{V}\mathbf{y}\,, \qquad (11)$$

where $\mathbf{x} \in \mathbb{R}^N$ is the state vector of the neural network nodes, which determine the rates of their activity through the activation function $\sigma(x) = \tanh x$: $\mathbf{r} = [r_j] = [\sigma(x_j)]$. Matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ sets the structure of recurrent connections; matrix $\mathbf{V} \in \mathbb{R}^{N\times L}$ defines the weights of input links; and matrix $\mathbf{U} \in \mathbb{R}^{N\times M}$ determines the feedbacks (L, M are the dimensions of the input and output vectors, respectively). The input of the network is given by the vector $\mathbf{y} \in \mathbb{R}^L$, while the output of the network $\mathbf{z} \in \mathbb{R}^M$ is determined through the matrix of output connections $\mathbf{W} \in \mathbb{R}^{N\times M}$:

$$\mathbf{z} = \mathbf{W}\mathbf{r}\,. \qquad (12)$$

It is noteworthy that system (11) describes the dynamics of the network in continuous time, in contrast to the discrete form of Eqn (4). In addition, the recurrent neural network in Fig. 1a belongs to the class of so-called reservoir computing systems, which are characterized by the fact that, in contrast to the general case (4), the output weights $\mathbf{W}$ alone are trained in such systems. Due to the presence of feedback, a modification of the output connections leads to restructuring of the efficient topology at the level of the entire recurrent network.

It has been shown that a recurrent neural network of the form (11) can be trained to autonomously generate various output patterns in the absence of input signals ($\mathbf{y} = 0$). For example, Fig. 1b, c shows the target space-time activity of the output neurons and the corresponding dynamics of the neurons of the recurrent network trained to reproduce the given target pattern at the output. In a multidimensional phase space, after training, a stable limit cycle arises, the projection of which onto the space of three principal components is shown in Fig. 1d. It should be noted that the points where the

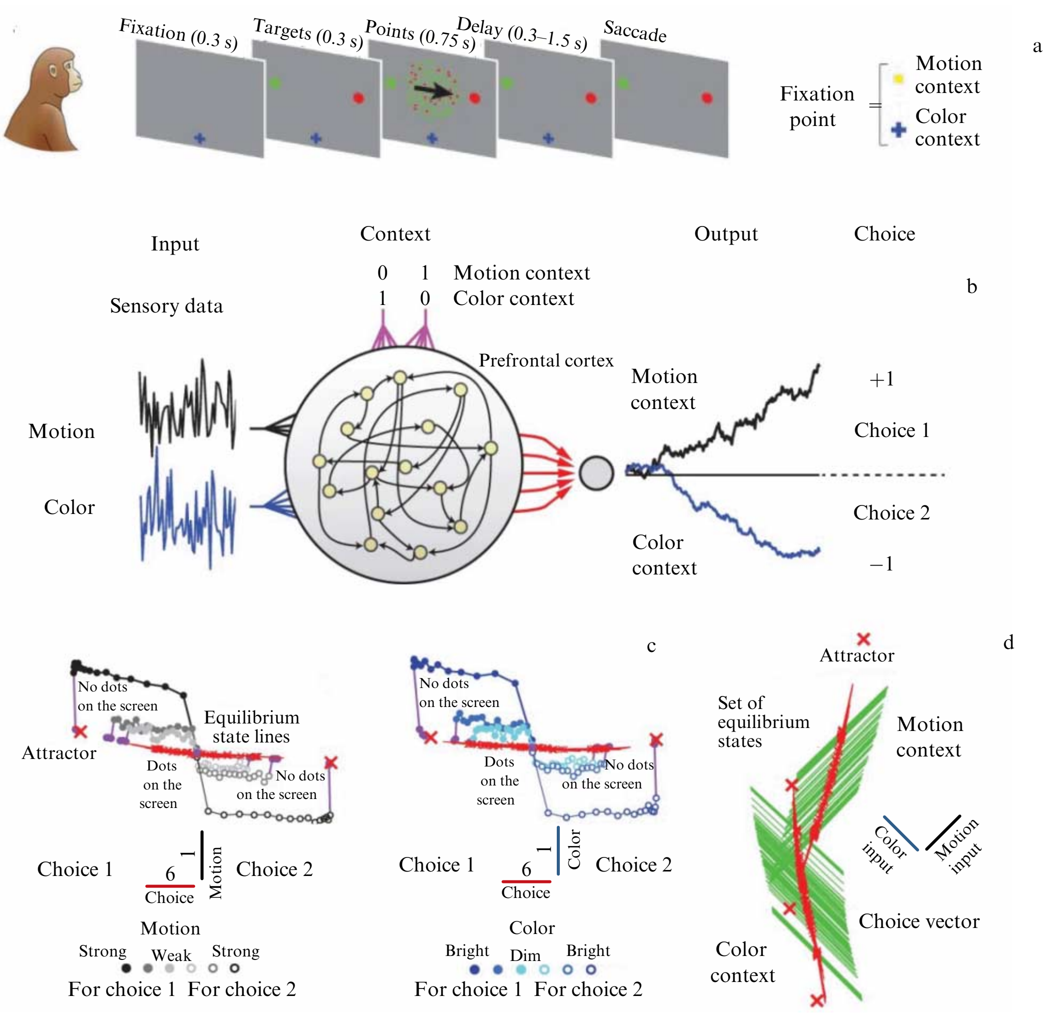

Figure 2. (Color online.) (a) Diagram of the screen with visual stimuli presented during the experiment. (b) Schematic of the model recurrent neural network. (c) Trajectories of the model system in two contexts and different intensities of input stimuli in the projection onto the choice axes, colors, and movement. (d) Attractors of a recurrent neural network. (Modified figure from [63].)

trajectory direction changes, which are highlighted in Fig. 1d in different colors, correspond to the activity peaks of the output neurons in Fig. 1b displayed in the same color.

In the presence of input signals, neural network (11) can perform various target transformations that model the simplest types of sensory-motor transmission of information. The use of a nonlinear dynamics approach also makes it possible in this case to establish the mechanisms of such transformations. For example, let the target problem be that the network, when subjected to a short action from one of the input neurons ($y_k > 0,\ y_j = 0,\ j \neq k$), generates a response with a given duration and shape on the corresponding output neuron alone ($z_k > 0$). It turns out then that the network exhibits irregular activity after training, just as before training; however, the arrival of input stimuli leads to the exit of the phase trajectory into metastable regions, the projections of which onto the space of three principal components are shown in different colors in Fig. 1e. The transient dynamics in these areas are quasi-regular, so the network functionally demonstrates target responses at the output.

Next, we consider examples of the use of recurrent neural networks for modeling cognitive functions studied in experiments. Study [63] describes an experiment in which a monkey learns to make a decision by analyzing a visual stimulus taking into consideration the so-called contextual signal. The animal observes dots randomly moving on the screen, some of which move in one direction (to the right or left), while the others move in the opposite direction. In addition, some of the dots are colored red, and the rest are green. From trial to trial, the color ratio and the predominant direction of movement change; moreover, in each trial, a contextual signal is presented that tells which task the animal should perform. If a yellow square appears, the direction of movement of most of the dots should be determined by saccadic (i.e., fast synchronous) eye movement in the corresponding direction; if a blue cross appears, then the dominant color should be determined in a similar way (Fig. 2a).

During the experiment, the activity of neurons in the prefrontal cortex of the brain is recorded, which demonstrate complex patterns of activity. An analysis of experimental data has shown that, although at the level of individual-neuron dynamics it is virtually impossible to separate the influence of input stimuli or decisions at the output, at the level of population activity, the influence of various aspects of input information and various solutions can be separated. It has been established that both characteristics of the visual stimulus (i.e., color and direction of movement) are reflected in the multidimensional activity of neurons, regardless of the

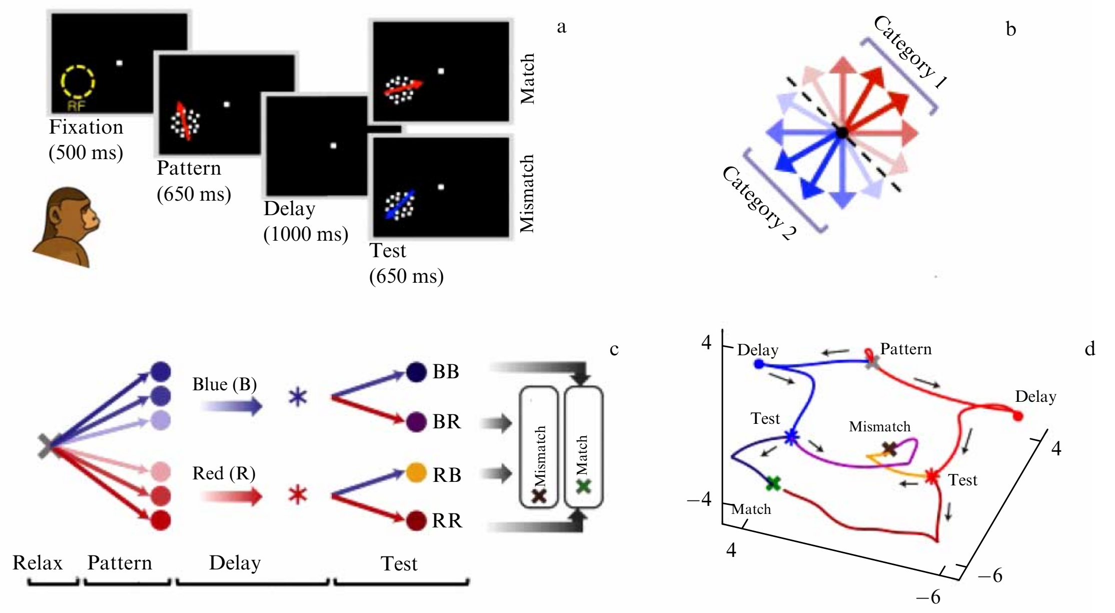

Figure 3. (Color online.) (a) Scheme of the screen with visual stimuli presented in the experiment on categorization of visual stimuli. (b) Diagram of the division of visual stimuli into two categories. (c) Stages of the experiment and the corresponding singular points of the phase space. (d) Singular points and trajectories of the phase space of the model system in perceiving two visual stimuli with a delay and making a decision about the match/mismatch of their categories. (Modified figure from [64].)

context. Thus, the communication of a contextual signal filters out irrelevant information not in earlier parts of the brain, as was previously thought, but in the neuronal populations of the cortex itself.

To identify the mechanism for solving the described problem at the level of population activity of the prefrontal cortex, the authors constructed a model in the form of a recurrent neural network (Fig. 2b) trained to perform the transformation of inputs into outputs, which is qualitatively similar to what is happening in the experiment. The network dynamics are described by a system of equations of the form (4) and (5). During each test, the model-network neurons receive two independent noisy signals as inputs, simulating sensory sensations about the predominant color and direction of movement of the cloud of dots, respectively. The network also receives a context input in the form of a 2D vector corresponding to the context signal in the experiment, which tells the network what specifically should be determined: color or direction of movement (Fig. 2b). After training, the network reports its decision in response to incoming stimuli, generating a binary response ($+1$ or -1 at the output). In the model network terms this implies that the output response supplies information about whether the average value of the signal over the test time from the input pointed to by the context input is positive or negative.

An analysis of the activity of the recurrent neural network after training revealed that in a multidimensional space it is possible to distinguish the axes of choice, color, and direction of movement. In the projection onto the three-dimensional subspace defined by these axes, the model dynamics qualitatively reproduce the key properties of the neurophysiological data from the experiment (Fig. 2c). First, in the process of receiving sensory input, a kind of accumulation of evidence in favor of one of two alternative solutions occurs. The population response trajectory moves along the choice axis. Second, the larger the values of the color and direction signals, the further the trajectories go from the choice axis along the corresponding axes. Third, the direction of the selection axes, colors, and direction of movement are invariant with respect to the contextual input signal. Fourth, trajectories associated with different contexts are spatially isolated from each other. This behavior of the system is determined by the special phase space trajectories that emerged in the course of training.

The authors have found that, in the phase space of a recurrent neural network, two lines of stable equilibrium states are formed, each of which is directed along the choice axis. The appearance of one of these two lines depends on the context, and the two of them never occur concurrently. Under the action of the input signal, the trajectory moves away from the corresponding line, and the stronger on average the input signal, the further it moves. For a fixed context, the movement of the trajectory after the end of the input stimulus occurs in the direction determined by the choice vector, which in turn is orthogonal to the direction of the irrelevant input. The directions of the lines of stable equilibrium states and the directions of the choice vectors are determined, respectively, by the right and left highest eigenvectors of the system linearized in the vicinity of the equilibrium states (Fig. 2c).

Thus, the use of a model recurrent neural network in this problem made it possible to identify the previously unknown mechanism of context-dependent decision making, in which the same input signal is processed in one context and ignored in another.

A recurrent neural network was constructed in [64] to model the problem of categorization of visual stimuli received with a delay. In prototype experiments [65, 66], monkeys were presented with visual stimuli in the following sequence: first, a fixation stimulus (for 500 ms), then a trial stimulus (for 650 ms), a delay period (1000 ms), and a test stimulus (650 ms) (Fig. 3a). The fixation stimulus is a yellow spot on the screen, and the trial and test stimuli are given by a set of dots that can move in a randomly chosen direction in the range from 0 to 360°. All stimuli used were divided into two categories using a preselected boundary (Fig. 3b). The test

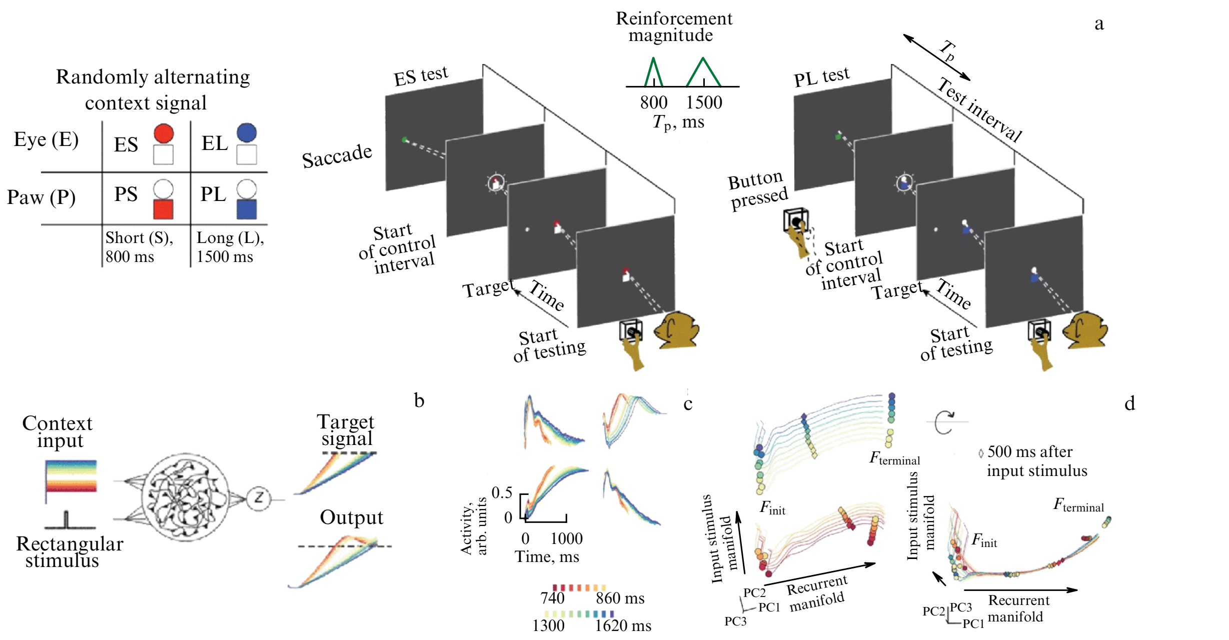

Figure 4. (Color online.) (a) Setup of experiment [67]. (b) Model structure in the form of a recurrent neural network. (c) Activity profiles of individual neurons after training. (d) Projections of activity onto space of the three principal components. (Modified image from [67]).

animals were trained to report the coincidence of the trial and test stimulus categories by pressing a lever.

In the model system [64], the recurrent network defined by system (4) and (5) receives signals from the layer of input neurons whose activity encodes the direction of movement of dots on the screen; a subset of neurons in the network sends connections to two output elements, the activity of which signals the match or mismatch of the categories of the first and second input stimuli. Analyzing the dynamics of the recurrent neural network after training, the authors of [64] revealed the mechanism of categorization of visual stimuli (Fig. 3c, d). At the beginning of the test, the trajectory is in a stable state of equilibrium (gray oblique cross); after the input stimulus is presented, the trajectory leaves its vicinity in the direction of one of the states that correspond to a particular category (red and blue circles) (Fig. 3d). Further, during the delay interval, a transition occurs to the neighborhood of stable or saddle points corresponding to two categories (red and blue stars). The action of the second stimulus brings the trajectory to the vicinity of one of the two equilibrium states that correspond to the match or mismatch of the stimulus categories. In qualitative terms, similar metastable states and transitions between them can be revealed based on the analysis of multidimensional neurophysiological data taken experimentally from cortex neurons of the monkeys being tested [64]. Thus, the recurrent neural network makes it possible to identify the features of population dynamics that reflect the entire chain of stages in the implementation of the described complex task.

Study [67] explored the mechanisms of neural activity that underlie the ability to control the speed of execution of motor and cognitive functions. During the experiment, the monkey was presented with visual stimuli: a circle in the center of the screen and a square below the circle (Fig. 4a). The test animal looked at the circle, while the square signaled that the button below the screen should be pressed. In each trial, one figure was painted blue or red, and the other was white. The colored figure indicated the target action: the circle corresponded to the requirement to make a saccadic eye movement, while the square communicated the requirement to press the button. The color indicated the desired time interval: red corresponded to 800 ms, and blue, to 1500 ms. The four different test conditions were designated as EL, ES, PL, PS (E and P stand for eye and paw, L and S for long and short intervals). After a delay period (varied in a range of 500–1500 ms), a target for saccadic eye movement appeared for a short time on the screen to the right or left of the center. After another delay, a short-term annular flash around the square and circle signaled the beginning of the control interval, during which the monkey had to perform the target action: saccadic fixation of the gaze or pressing a button. To receive a reward, the animal had to change the duration of this action in accordance with the target value indicated by color. Concurrently, the electrical activity of neurons in the medial frontal cortex was recorded.

The described setting of the experiment makes it possible to analyze the properties of neural activity that correspond to the control of the duration of time intervals during the performance of motor acts. Studying the population response of neurons, the authors have found that the average spiking frequency has a time profile, the width of which changes in accordance with the target duration of the control interval.

To find a mechanism that controls the duration of time intervals, the authors of [67] developed a model in the form of a recurrent neural network (4) and (5) consisting of 200 neurons (Fig. 4b). The network receives two scalar inputs: one is contextual, the value of which determines the duration of the target interval, and the other is a short rectangular stimulus that signals the beginning of the interval, during which the activity of the target output should monotonically increase until a certain threshold value is reached for a time equal to the target interval. An analysis of the activity of the trained network showed that the model qualitatively reproduces the key phenomena discovered in the experiment. At the level of individual neurons, their response profiles have complex and diverse shapes, and they change

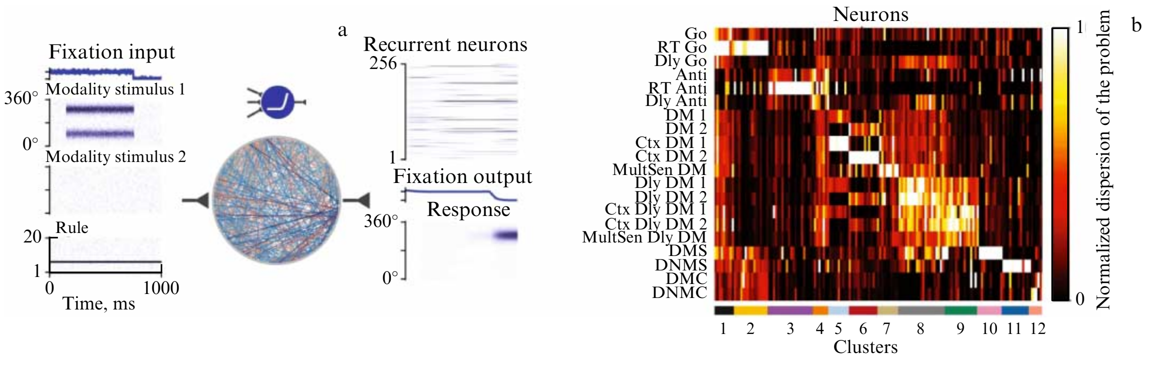

Figure 5. (Color online.) (a) Recurrent neural network performing 20 various cognitive tasks. (b) After training, clusters are distinguished in the network structure, each of which is specialized to perform one or more tasks. On the vertical axis is a list of tasks, on the horizontal axis are neurons arranged for better visualization of clusters. Different colors show the average activity of each neuron when the network performs a specific task. (Modified image from [68].)

their width in accordance with the target duration (Fig. 4c). Moreover, the speed of the population dynamics is the factor that determines the duration of the response produced at the output. The authors of [67] analyzed the properties of objects in the phase space of the network that correspond to such behavior (Fig. 4d). It has been found that two sets of stable equilibrium states are formed: F_{init}, corresponding to the beginning of the test, and F_{terminal}, corresponding to its end. The contextual stimulus initializes the network state in the vicinity of the equilibrium state F_{init}, then the input rectangular stimulus drives the trajectory out of its basin of attraction, and the trajectory relaxes to the vicinity of the stable equilibrium state F_{terminal} at a rate determined by the value of the contextual input stimulus, which affects the position of the points F_{init} and F_{terminal} along the direction called the input stimulus manifold, which is determined by the input weight vector. At the same time, the recurrent-network weights define the direction (the so-called recurrent manifold) along which the trajectories move between the initial and final states.

The approach presented has been successfully used to build recurrent neural networks that model other cognitive functions associated with working memory [27, 69, 70], compare the characteristics of input stimuli [71, 72], and make decisions depending on changes in the environment and contextual signals [67, 73–77]. As a rule, the artificial network for the given examples is trained on a single specific task, which makes it possible to identify the dynamic mechanisms for the implementation of a certain function. This formulation is a strong simplification compared to biological networks, which, in the process of functioning, concurrently perform a number of various tasks.

In [68], a recurrent neural network of the form (4) and (5) was built, which was trained to perform 20 various cognitive functions, such as working memory, decision-making after a time delay or in the presence of a contextual signal, and comparison and categorization of sensory stimuli. These functions are typical in solving problems performed by experimental animals with concurrent recording of the activity of brain neurons.

In solving each task, the network receives inputs of three types: a fixation signal, meaningful stimuli, and an input that defines the task (Fig. 5a). The fixation signal specifies the stages of task execution, i.e., determines when the network should 'perceive' information from the inputs (the fixation signal then takes a positive value), and when the network should generate a response at the output (when the fixation signal is reset to zero). The meaningful inputs are represented by stimuli of two modalities, each of which is given by a ring of input elements encoding a one-dimensional circular variable, such as direction of movement or color. The input that defines the task is given by a vector consisting of zeros and one unit, the position of which informs the network about which task it is trained for in this test or which task it should perform after training. The network outputs are a fixation output whose purpose is to repeat the path of the input fixation signal and a group of motor elements, the response of which determines the direction and intensity of the network response. Network training, in which all network weights (input, recurrent, and output) were tuned, was carried out with random switching of target cognitive tasks. An analysis of the network after training showed that groups of neurons are formed in it, clusters specialized to perform a certain class of cognitive tasks (Fig. 5b). In addition to the established functional specialization, the authors also discovered another principle for organizing the operation of the network: compositionality, i.e., the technique to solve a complex task by dividing it into simpler tasks that the network has already been trained in earlier.

Summing up, we note that the described recurrent neural networks that model the performance of cognitive functions make it possible to identify a number of important dynamic mechanisms underlying the problems solved by prototype neural populations. All examples presented are based on models of so-called rate neurons [78, 79]. This circumstance reflects the assumption that the coding of external signals and the transmission of information between individual neuron subsystems are carried out using a rate code. However, a large number of experimental facts indicate that rate coding is not the only way of coding which is possible in the nervous system, and that it is not only rate that plays a significant role. Also of importance are the specific moments when elementary dynamic events occur in the form of short impulses or spikes that emerge when the neuronal membrane voltage reaches the excitation threshold. Artificial neural networks, in which the very fact of generating action potentials by neurons is modeled, are called spiking networks. Section 4 presents various aspects of the use of specifically spiking neural networks in modeling the functional properties of brain networks.

4. Spiking neural networks

Despite the fact that in a number of neuroscience problems it has been possible to achieve results based on standard machine learning techniques (basic architecture in the form of recurrent neural networks and stochastic gradient descent as a learning method), further progress both in these areas and in the construction of next-generation neuromorphic systems of artificial intelligence requires taking into account biologically relevant aspects.

One of the factors that distinguishes the behavior of biological neurons from that of deep learning artificial neural networks is the ability to generate action potentials, or spikes. Almost a quarter of a century ago, in [80], spiking networks were characterized as third-generation artificial neural networks, in contrast to networks of the first generation (perceptrons based on McCulloch–Pitts neurons) and the second generation (deep networks based on sigmoid and other similar nonlinear activation functions), which includes most state-of-the-art machine learning systems [81].

It should be noted that there is currently no conclusive understanding of the role played by spikes in the implementation of the functions of the brain. Some arguments favor both the importance of the average spike frequency in information processing and the key role played by the timing of the emergence of individual spikes. Therefore, in modeling spiking neural networks, various hypotheses and views on the role of spike activity are common. In machine learning, one of the motivations for the use of spiking networks is energy efficiency, since the generation of sparse spike codes requires less energy than the processing of continuous variables in conventional deep neural networks.

It is worth noting that most deep neural networks in use today have static inputs and produce static outputs. Even in the case of video image processing by a convolutional neural network, it is divided into separate static frames and object recognition on each of them is carried out. In other words, such systems contain neither time as a physical variable, which is an argument of input and output signals, nor model neurons and synapses. In spiking neural networks, on the contrary, time is incorporated at the level of the dynamics of individual neurons, and input signals are processed in a natural way as a nonautonomous effect on the dynamic system.

The vast majority of studies in this area are associated with the conversion of deep neural networks into spiking ones [81–83], which involves scaling and normalization of the parameters of a nonspiking neural network trained to perform certain tasks so that the resulting spike network produces the same conversion of inputs to outputs. However, approaches are also being developed that are initially focused on the spiking dynamics of artificial neurons. Spiking neural networks are being developed both for solving standard machine learning problems and for building systems that model neuroscience problems [33–36]. Based on spiking neurons, multilayer feedforward networks have been constructed that are used to model low-level sensory systems, in particular, in vision [84, 85], smell, or tactile sensations [86]. Recurrent spiking neural networks have been used, for example, to study neural information processing associated with the formation of associative [87, 88] or working [89–91] memory. Recurrent spiking networks are also used to model and analyze the phenomena observed in the brain that arise from the complex dynamics of interconnected networks, such as network fluctuations [92, 93] or network multi-stability [94], and to model competition between neurons or neuron populations [95–97] and various processes that accompany decision-making [98].

To model the functional properties of neural spiking populations in terms of the top-down approach, various methods are used, among which reservoir computing and methods based on the neuroengineering framework may be singled out. These methods assume that the target behavior of the system is known a priori, and the recurrent network of spiking neurons is trained with various constraints on the level of biological detailing of the models. We now consider each of these approaches in more detail.

4.1 Reservoir computing

Reservoir computing has emerged as an alternative method to training recurrent networks, which makes it possible to avoid the difficulties associated with training networks [99–104]. In a typical implementation, a reservoir network consists of a fixed topology recurrent network (reservoir) and a set of output (readout) neurons. Typically, the latter only receive direct unidirectional connections from the reservoir, although, in some models, feedback from output neurons to the reservoir is also taken into account. However, the desired output from the network is obtained by training only the connections between the reservoir neurons and the readout neurons. This approach greatly simplifies training in reservoir networks.

A reservoir can be viewed as a structure that maps inputs to a multidimensional activity vector of neurons belonging to the network. Each component of this vector reflects the influence that specific neurons can have on the output units. The structure of connections within the reservoir is usually random and fixed. To obtain stable output data, it is not necessary to have stable internal states of the reservoir, since transient internal states can be converted by readout neurons into stable output signals due to the high dimensionality of the dynamic system. In addition, reservoir states and transitions between them do not require making adjustments for a specific task, implying that the same sufficiently large common reservoir can be used to solve many different problems.

The concept of reservoir computing was originally proposed in the context of nonspiking artificial neural networks [99–101]. Later, the method was adapted for spiking networks as part of the concept of so-called liquid state machines [102, 105]. Reservoir computing with spiking neurons has been successfully applied to problems such as spoken word recognition, space-and-time classification of spike patterns, motion prediction, motor pattern generation, and motion control [106–110]. To model functional properties, the FORCE method was used to extend application of spiking neural networks.

FORCE, the method of reduced and controlled first-order error [46], extends the reservoir computing approach by adjusting the low-rank component of the recurrent weight matrix that results from the feedback loop. The structure of the spiking network (Fig. 6a) trained by this method is similar to the reservoir computing systems described in Section 3, which are schematically shown in Fig. 1a. The difference is that the dynamics of neurons is given by equations that explicitly describe spiking events. Integrate-and-fire neurons (10) are often used as models of spiking neurons; models with an adaptive excitation threshold, Izhikevich and Courbage–Nekorkin models, and theta neurons can also be employed.

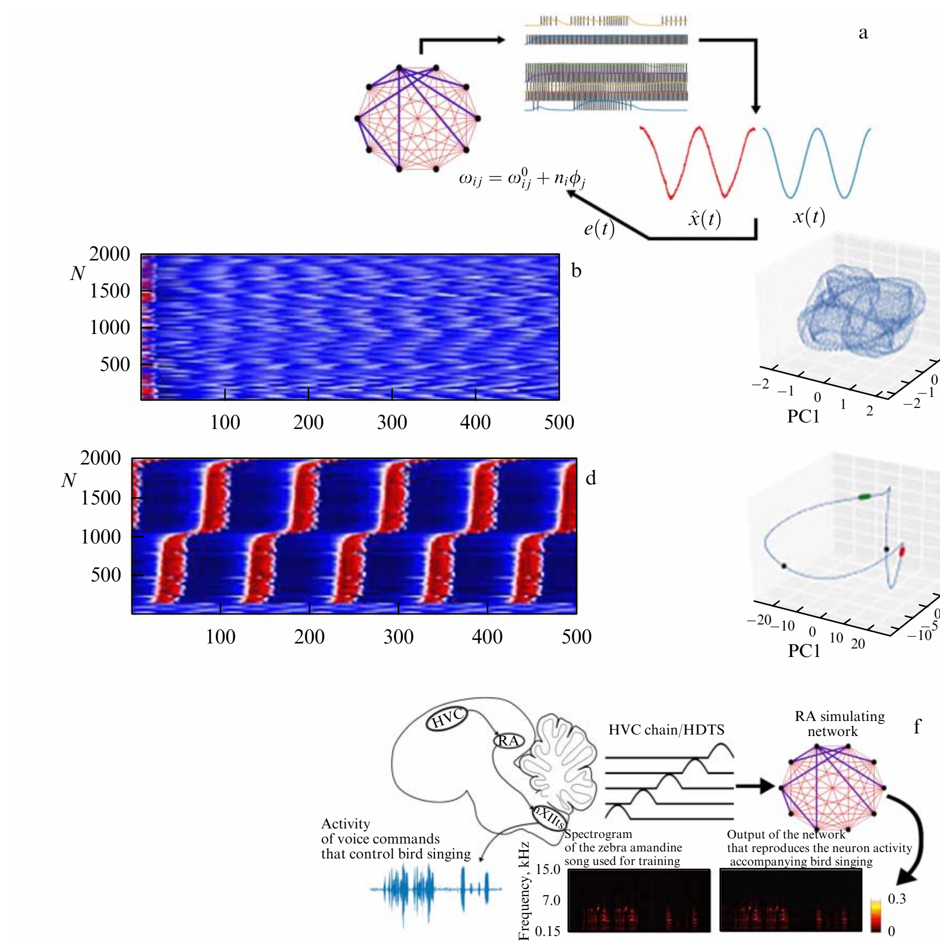
a
$\omega_{ij} = \omega^0_{ij} + n_i\phi_j$
$e(t)$
$\hat{x}(t)$
$x(t)$
b
N
PC1
PC2
PC3
c
d
e
HVC
RA
nXIIts
HVC chain/HDTS
RA simulating network
f
Activity of voice commands that control bird singing
Spectrogram of the zebra amandine song used for training
Output of the network that reproduces the neuron activity accompanying bird singing
Frequency, kHz
15.0
7.0
0.15
0.3
0

Figure 6. (Color online.) (a) In the FORCE method, a spike neural network is characterized, first, by fixed strong weights that induce chaotic network dynamics (blue). Second, a component is added to the weight matrix with elements ω_{ij}, the values of the elements of which are determined in the learning process (red). FORCE method requires output $\hat{x}(t)$ to be evaluated using target $x(t)$; $e(t)$ is the error signal. (b–e) Dynamics of a spiking neural network before and after learning the task of autonomous generation of a harmonic target signal: (b, c) spiking activity of a network of identical neurons before learning and its projection onto the space of three principal components; (d, e) network activity and projection onto space of the three principal components after learning the autonomous generation of a harmonic target signal. (f) Using a spiking neural network to store and reproduce a pattern corresponding to birdsong. Series of pulses built from the positive part of a sinusoidal oscillator with a period of 20 ms is used to simulate the excitation circuit of the output signal from projection neurons of the nucleus of arcopallium RA (Robust nucleus of the Arcopalliuin) (bird brain structure) in the HVC (High Vocal Center) (hyperstriatum bird brain structure). These neurons connect with neurons in RA and launch singing. The FORCE method was used to train a network of 1000 Izhikevich neurons to reproduce the spectrogram of a 5-second audio recording of a zebra finch singing; nXIIts, tracheosyringeal division of the twelfth nucleus, avian brain structure; HDTS (high-dimensional temporal signal) is the high-dimensional signal; HVC chains are stimuli generated in HVC. (Modified figure from [34, 59].)

It has been found that recurrent spiking networks can be a reservoir capable of learning tasks related to the autonomous and stimulus-induced generation of various space-time patterns. For example, a network of spiking neurons that exhibits irregular activity before training (Fig. 6b, c) by adjusting the output weights is able to maintain the reproduction of a harmonic signal at the output (Fig. 6d, e), forming a stable invariant curve in the phase space.

Figure 6f displays an example of a spiking neural network trained using the FORCE method to reproduce bird singing (in the form of a spectrogram) recorded from an adult zebra finch. The singing behavior of these birds is due to two main brain nuclei: HVC and RA. The neurons that project connections from RA to HVC form a synfire chain, and each neuron in RA fires only once at a certain time during the song performance. This synfire chain is transmitted to the RA module and activates it. Each neuron in RA generates bursts at several precisely defined times during song playback. The RA neurons then project connections to the lower motor nuclei, which stimulate the vocal muscles.

4.2 Neuroengineering framework

The neuroengineering framework consists of three principles used to describe neural computing: (1) representation, (2) transformation, and (3) dynamics.

The principle of representation is that the neural population activity encodes the dynamics of some vector variable. Coding is based on the fact that different neurons have different amplitude-frequency characteristics, i.e., the frequency of their spikes depends differently on the magnitude of the input current determined by the encoded variable. In addition, by measuring the spike activity of a neuronal population, the variable can be decoded using a linear transformation.

The transformation principle is that the weight vector, which can be used to decode a variable from the network spiking activity, can be configured in such a way that the neural population performs an arbitrary transformation (function) of the encoded variable.

The principle of dynamics is that the weights of connections in a recurrent spike neural network can be adjusted in

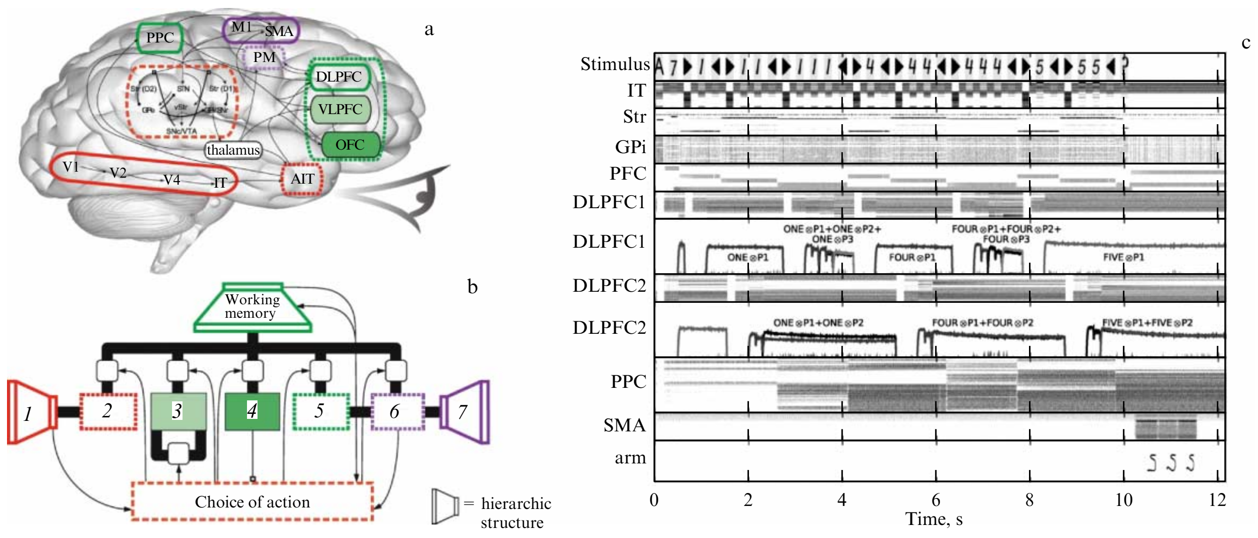

Figure 7. (Color online.) (a, b) Spaun's functional spike network with components and their anatomical prototypes: PPC — posterior parietal cortex, M1 — primary motor cortex, SMA — supplementary motor area, PM — premotor cortex, VLPFC — ventrolateral prefrontal cortex, OFC — orbitofrontal cortex, AIT — anterior inferior temporal cortex, Str — striatum, vStr — ventral striatum, STN — subthalamic nucleus, GPe — *globus pallidas externus*, GPi — *globus pallidus internus*, SNr — substantia nigra pars reticulata, SN — nigra pars compacta, VTA — ventral tegmental area, V2 — secondary visual cortex, V4 — extrastriatal visual cortex; *1* — visual entrance, *2* — encoding of information, *3* — conversion calculations, *4* — reinforcement evaluation, *5* — information decoding, *6* — motor processing, *7* — motor output. (c) Solution of the cognitive task associated with Raven's progressive matrices by the Spaun spike network [111]. Top line is visual stimuli applied to the system; bottom line is motor output imitating the handwritten response of the system; intermediate lines are the spiking activity of corresponding subsystems. (Modified figure from [112].)

such a way that its activity directly corresponds to a variable of a certain dynamic system.

For example, a spike network is presented in [112] that models a number of functional properties of the brain, based on a neuroengineering framework the essence of which is described below. The network, which consists of 2.5 million integrate-and-fire spike neurons, is organized as a system of interacting subnetworks (Fig. 7a, b), including three hierarchical networks (visual and motor networks and working memory) that compress input signals, and an action selection mechanism. Interaction among subsystems is maintained by spike neurons, which by their activity map the signals of the outside world onto the space of so-called semantic pointers. The latter are elements of a vector space, the dimension of which is determined by the degree of compression executed by the neural network. Compression is understood as a reduction in the size of subnets that process information flows. For example, in perceiving visual images, a sequential transmission of signals occurs from the primary visual cortex, in which contrast patterns in the observed image are recognized, down to the lower temporal cortex, which makes distinctions at the level of concepts related to individual elements of the observed image. The number of neurons in the subsystems participating in this process is successively reduced; thus, a compressive mapping of vectors from the image space to the low-dimensional space of concepts is carried out.

The authors used the described spiking network to solve a number of problems simulating cognitive functions, such as recognition of handwritten digits, their copying, addition, and pattern recognition in number series. One of the difficult tasks is the Raven test, in which two groups of images are presented. Each of the groups contains three logically related elements, and next a third group is presented which contains only two elements, and the missing third element must be reported based on the principles that have to be identified on the basis of the first two groups. For example, Fig. 7c shows the solution to the problem by the spiking network, in which the missing element of the pattern 1 11 111; 4 44 444; 5 55? should be determined.

Thus, the neuroengineering framework provides a general principle for training the weights of recurrent networks of spike neurons for the implementation of specific dynamic systems [113]. This problem is solved using weight matrices that encode and decode the dynamics of some vector variable into spiking activity (from spiking activity), and optimization through a multi-stage application of the least squares method. Effective application of the methodology is carried out in practice using the Nengo software package [114], which can be used to program neuromorphic devices [115].

5. Hardware neuromorphic systems

The term 'neuromorphic systems' refers to specialized hardware performing computations that mimic the principles of information processing in biological neural populations [116, 117]. Such computations are called neuromorphic. Currently, this concept is often applied in a narrower sense to spiking neural networks. From the perspective of information theory and computer calculations, hardware architectures based on spiking neurons perform functional transformations of inputs into Turing-computable outputs, i.e., can be implemented on a Turing machine. Unlike conventional computers, neuromorphic architectures are intended for implementing algorithms specially designed for spiking networks that perform parallelized computations that are sparse in space and time, due to which much less computer power is consumed [118].

The number of studies devoted to the development of neuromorphic systems using the principles on which biological neural networks function has increased significantly over recent decades; for example, a recent review [37] analyzed more than 3,000 such publications. Neuromorphic computing is now used in a variety of devices; this is due to the fact that the development of central processor systems with the von Neumann architecture is approaching its peak computing power [119]. It is assumed that the development of neuromorphic devices, in which the basic principles of the operation of neural networks of the brain, such as spike dynamics and synaptic plasticity, and which have a comparable number of neurons and a topology of connections close to reality, will result in an enhanced ability to self-organize, learn, and perform various classes of tasks.

It is noteworthy that today neuromorphic systems are implemented using various technologies, for example, optical (photonic), optoelectronic, and spin-electronic, but semiconductor technology is the most common, since it has been developing relatively longer and more actively than the other technologies and therefore is the most standardized for the creation of large integrated circuits of various types and applications. Application-specific integrated circuits (ASICs) that solve a specific class of problems have a unique architecture only characteristic of them; in this sense, neuromorphic systems belong to this type of semiconductor device, since they have a structure similar to that of living biological neural networks. Developers are naturally trying to implement all the properties of spiking neural networks, which, in principle, can boost the efficiency of calculations and reduce energy consumption when implemented using state-of-the-art microelectronic technologies [120].

However, it is apparent that the systems already developed by now do not yet feature a density of neuromorphic elements and connections or an energy efficiency comparable to those of brain neural networks. And this is not accidental, since a straightforward reproduction of the physicochemical structure of neurons and the architecture of biological neural networks in the semiconductor (and any other micro- and nanoelectronic) paradigm is impossible due to many limitations.

For example, biological neural networks have a three-dimensional (3D) architecture of connections located in space, while semiconductor crystals on which integrated circuits are placed have a planar structure with a limited area, and, if the number of circuit elements is very large (comparable to the number of neurons and connections between them in the human brain), their placement is a very challenging and not always feasible task. Despite the fact that attempts to develop semiconductors interconnected in three directions have been made since the 1970s [121], 3D semiconductor technologies so far only exist in single copies.

Another difficulty is the implementation of asynchronous multicast communication ('from one to many'), which often occurs in systems with a parallel architecture, including neuromorphic ones. In fact, such a connection should allow individual elements or subsystems of the whole system to operate (for example, generate spikes in the case of neuromorphic networks or perform calculations for conventional multiprocessor systems), without waiting for the execution of any tasks by other elements or subsystems. However, today, there is in general no asynchronous communication which would be suitable for any complex network implemented at the hardware-algorithmic level in continuous time (not sampled by clock counts). There are only a limited number of relevant studies [122], although in some neuromorphic systems, which will be discussed below, such as IBM's TrueNorth, such a connection at the level of interaction between subsystems has already been implemented.

Another key task in the construction of a neuromorphic system is the implementation of connections between elements, which are carried out in biological neural networks through a synaptic contact — a microscopic gap between two neurons. Synapses are known to be of two types: electrical and chemical (notably, the latter predominate in the human brain), and it is assumed that it is the various properties of chemical synapses, including synaptic plasticity, that enable the brain to learn and store information (have memory). Here, difficulties arise in creating artificial synapses with ideal synaptic properties, and developers again have to solve the problem of implementing chemical processes using semiconductor or any other technologies.

Problems, in turn, should be solved in developing the characteristics of artificial synapses, such as the dynamic range (the ratio between the maximum and minimum conductivity of synapses), the number of states (sampling by value), the linearity of the weight update (which enables changing the synaptic weight to the desired value during training), the ability to maintain the weight value (that provides long-term data storage), and the cyclic endurance of changing weight values (synaptic weights are repeatedly adjusted during training and, accordingly, the option to physically overwrite the state should be maintained). For instance, examples are reported in [123] of how relatively new ferroelectric field effect transistors (FeFETs) and organic electrochemical transistors (OECTs) are used to implement artificial synapses.

It should be noted that the implementation of various properties of biological neural networks in neuromorphic systems involves many limitations, due not only to the technological features of microelectronics but also to experimental data and the basic understanding of the properties of biological networks per se and their components. Thus, the implementation of a neuromorphic system is reduced to finding the optimal solution in attaining a balance of several factors: the biological relevance of the system, technological capabilities, and the goals pursued (i.e., what the constructed system should eventually 'be able' to do).

At present, it is not completely clear to what extent complete biological similarity leads to an improvement in the final capabilities of a neuromorphic system; nevertheless, it is apparent that, the higher the similarity degree, the significantly greater the technological costs. An important role in this issue is played by theoretical and numerical studies of the dynamics of the functioning of neuromorphic networks. Such studies enable making the optimal choice of fairly adequate models of neurons and synaptic contacts, which, when building a neuromorphic system, allow maintaining a balance between biological relevance and its final properties and capabilities. Nevertheless, even within the framework of modern ideas and technological capabilities, it has become obvious that, compared to conventional processor systems, neuromorphic networks are more efficient in solving problems of recognition (images, sounds, gas identification, etc.), forecasting of time series, and solving cognitive problems.

In Sections 5.1–5.6, we consider examples of several of the most popular large-scale projects in which neuromorphic systems have been implemented in the form of hardware devices.

5.1 Neurogrid and Braindrop neuromorphic systems

Researchers at Stanford University have created two classes of neuromorphic devices. The first is Neurogrid, a mixed analog-digital system that can simulate in real time networks consisting of several million spike neurons and several billion synaptic connections [124]. Neurogrid consists of 256×256 analog neurons produced on a 180-nm complementary metal–oxide–semiconductor (CMOS) structure to create the Neurocore core. To build a complete Neurogrid system, 16 Neurocore chips are placed on a board and arranged in a tree structure. An example of the practical application of the Neurogrid system is presented in [125], where an articulated robot with three degrees of freedom controlled by Neurogrid is described. A robot capable of performing arbitrary movements in 3D space controls a pen that touches a surface.

The second project, called Braindrop [126], has been developed, similar to Neurogrid, on the basis of a mixed analog-digital design. However, unlike Neurogrid, Braindrop is designed for programming at a high level of abstraction. Braindrop uses a neuroengineering framework as a theoretical basis for computations, which are defined as coupled nonlinear dynamic systems, while systems and equipment are synthesized using an automated procedure. Braindrop, which is manufactured based on 28-nm technology using fully-depleted silicon-on-insulator (FD SOI), combines 4096 neurons on a single Braindrop chip. Several Braindrop cores will be integrated to create a larger Brainstorm chip.

5.2 BrainScaleS and SpiNNaker neuromorphic systems

The BrainScaleS and SpiNNaker neuromorphic systems have been developed as part of the European Human Brain Project (HBP) [127]. BrainScaleS is a mixed analog-digital type neuromorphic system developed through the joint efforts of several research groups, including those from the University of Heidelberg and the Technical University of Dresden [128, 129]. The integration technology developed for BrainScaleS makes it possible to use a 20-cm plate for a large-scale neuromorphic system that includes up to 180,000 neurons and 40 million synapses. The BrainScaleS system is built by implementing a multitude of analog neural circuits and their synapses in a structure called the analog network core (ANC). The ANC is produced using a 180-nm CMOS structure to create a high input count analog neural network (HICANN). Three hundred fifty-two HICANN chips are placed on a single wafer.

The second generation of the BrainScaleS-2 system [130] is currently under development; it is based on a more complex neuron model, nonlinear transformations in dendrites, and other functions. Also added is the option to run the system in the mode of a multilayer network of feedforward propagation and build convolutional neural networks without spiking. This feature makes it possible to concurrently combine in BrainScaleS-2 both spiking neurons and multilayer networks of nonspiking neurons in one experiment.

As part of the HBP, researchers from the University of Manchester are developing the SpiNNaker digital neuromorphic system [131]. The SpiNNaker system simulates spiking neural networks in real time by connecting more than 1 million ARM processors, which enable simulation of 1 billion neurons with a biologically realistic number of connections (1,000–10,000 synapses per neuron) and with a time resolution of 1 ms. Eighteen ARM968 processors are integrated into a single-chip multiprocessor (SCMP) based on 130-nm CMOS. Sixteen processors are used for simulation; one processor is for administration, and another one is redundant in case of failure. The complete system, which has a two-dimensional toroidal structure of connections, consists of 216 CMPs.

The first version of SpiNNaker contains about 1% of the number of neurons and connections between them that exist in the human brain. The second generation, called SpiNNaker2, is designed to model a network with the number of neurons comparable to that contained in the whole brain [132]. To achieve this goal, it is planned to scale up the previous generation system so that SpiNNaker2 will have 144 ARM MF4 cores per CMP based on state-of-the-art 22FDX (22-nm FD SOI) technology. In addition, SpiNNaker2 will include new features such as dynamic power management, support of floating point computations, synchronous memory sharing with neighboring cores, multi-accumulation accelerators, and other numerical accelerators.

The SpiNNaker and BrainScaleS projects have been under development for more than 12 years, and there are numerous examples of their application. We only note a few of them. The spiking neural network implemented on the SpiNNaker neuromorphic chip (4 chips, 64 cores) performs as a noninvasive brain-computer interface that decodes imaginary movements based on electroencephalogram signals [133]. The SpiNNaker system can control the interaction of a mobile robot with the environment [134], as well as with a mobile robot by interpreting visual data [135]. The BrainScaleS system implements a spiking network that generates random samples with a Bayesian distribution, which are used for generative and discriminatory calculations on visual data [136]. A new generation of the BrainScaleS-2 neuromorphic system demonstrates the experimental possibility of creating a reward spiking network with STDP synapses that learns from the Pong video game [137].

5.3 TrueNorth system

The TrueNorth neuromorphic platform (Fig. 8) was developed by IBM as part of the SyNAPSE program of the Defense Advanced Research Projects Agency (DARPA) [138]. One chip of the TrueNorth system consists of 4096 neurosynaptic cores, each of which combines memory ('synapses'), processors ('neurons'), and connections ('axons'), manufactured using IBM's 45-nm SOI (silicon-on-insulator) technology [140]. Each core contains 256 leaky integrate-and-fire neurons which have 1024 axonal circuits of the input connection organized according to the static random access memory (SRAM) scheme [141]. One TrueNorth chip containing 5.4 billion transistors can simulate 1 million neurons and 256 million synapses integrated into a single structure. Several chips are combined into a larger NS16e platform with 16 chips, which made it possible to simulate 16 million neurons and 4 billion synapses.

The TrueNorth system has been under development for more than seven years, and many tasks have been tested using this system. For example, it performs classification tasks, demonstrating very high accuracy on eight sets of various data (including images of various infrastructure objects, road signs, house numbers, various sounds, and spoken English phonemes), and features an impressive image processing rate, from 1200 to 2600 fps, and energy consumption from 25 to 275 mW [142]. The TrueNorth system implements noise filtering for an image sensor with asynchronous detection of changes for moving objects (the so-called silicon retina, a new class of vision sensor that operates on the basis of the biological principles of vision); its filtering outperforms the conventional nearest-neighbor noise filter [143].

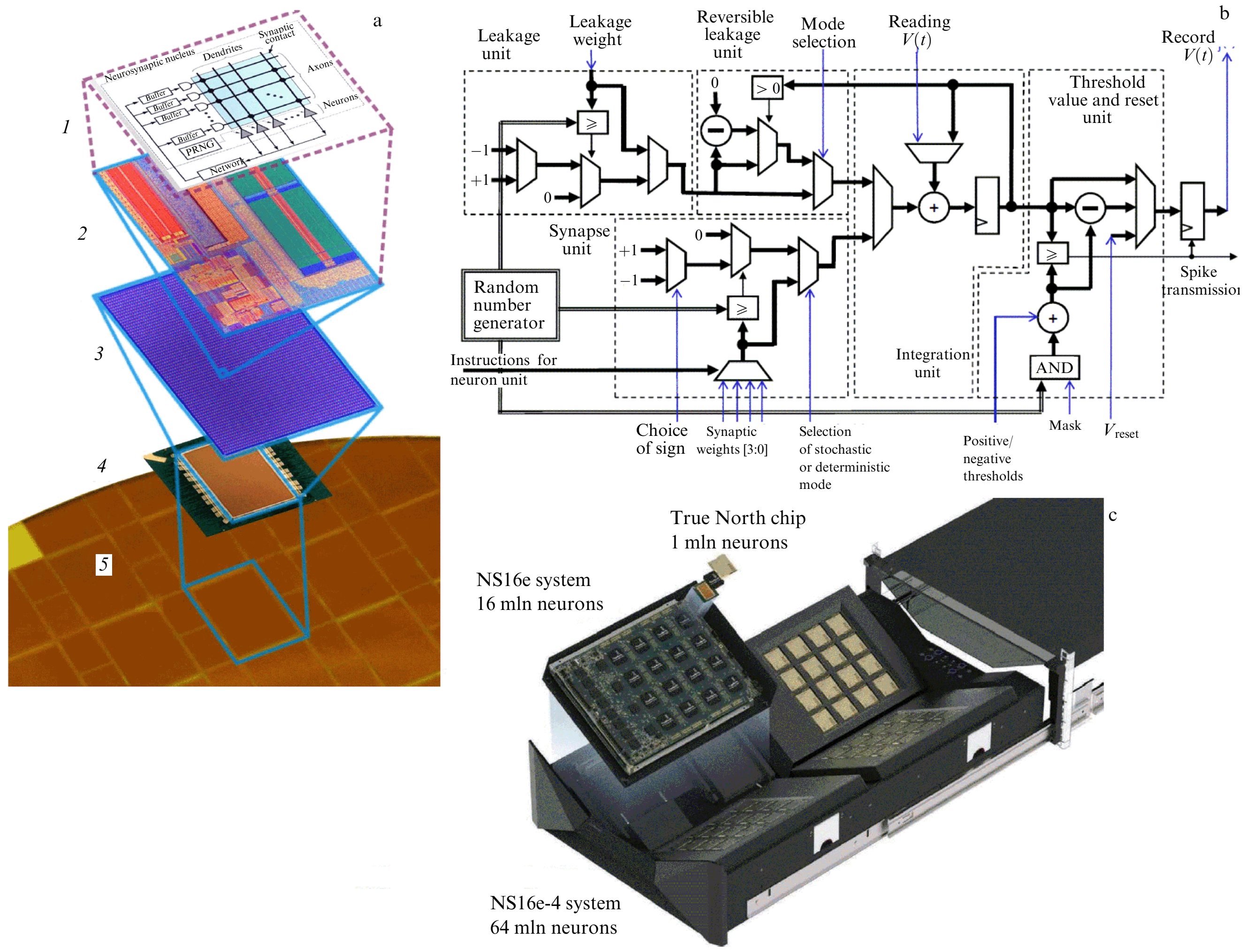

Figure 8. (Color online.) (a) TrueNorth system architecture. Neurosynaptic nucleus *1* is the main building unit containing 256 input axons, 256 neurons and 64,000 synaptic contacts (crossings). Core *2* consisting of connected neurons, synapses, and connections (axons and dendrites) occupies an area of 240×390 μm^2 on silicon. Combination of 4096 neurosynaptic nuclei into a two-dimensional array forms a TrueNorth chip *3*, which occupies an area of 4.3 cm^2 if a 28-nm CMOS process *4* is used. End result is an effective approximation of the cortical structure on silicon substrate *5*. PRNG is a pseudo-random number generator. (Adapted figure from [138].) (b) Block diagram of an accumulation-reset neuron with leakage. (Adapted figure from [138].) (c) TrueNorth-based system consisting of 64 million neurons [139].

5.4 Loihi system

Intel has developed Loihi, a digital neuromorphic research chip [144], which has a unique programmable on-chip training engine for spiking neural networks. Loihi is manufactured using Intel's 14-nm process. In addition to the 128 neuromorphic cores, the chip contains three Lakemont cores that implement more complex learning rules with control of neuromorphic cores. One hundred twenty-eight neuromorphic nuclei implement 130,000 artificial neurons (CUrrent BAse Intagrate and Fire, CUBA IF [86, 145]) and 130 million synapses. A detailed block diagram of such a neuron is displayed in Fig. 9. The Loihi design supports scaling up to 4096 cores per chip and 16,384 chips. All cores operate asynchronously, so when receiving signals, each can spend a different time processing them; however, it is only after all the cores have completed processing that the next iteration of the transmission of spikes and their processing begins. This property implies that, the more spikes generated throughout the system, the longer each iteration is processed.

According to Intel, this chip is completely deterministic, in contrast to other chips, i.e., transmission of all spikes between all neurons is guaranteed, and, under the same initial conditions, the computational process will be fully reproduced even if the system size is comparable in the number of neurons and synapses with the human brain. For the device under consideration, a programming language has been developed that enables setting various parameters of both the neurons themselves and synaptic connections, and the structure of connections between neurons.

The Loihi neuromorphic system shows various possibilities of its use for training spiking networks. For example, the Loihi system was used to implement a spiking network that classifies the video data of a road situation, which is necessary for solving problems involving autonomous driving. Implementation that uses this neuromorphic hardware features a maximum latency of 0.72 ms per sample and only consumes 310 mW [146]. The Loihi system was used in [147] to recognize images, handwritten digits, and other data, and its performance was compared with that of other neuromorphic systems. The Loihi system implemented a neural network for online learning and identification of odor samples under noise conditions, based on the architecture of the mammalian olfactory bulb, using synapses with STDP plasticity [148].

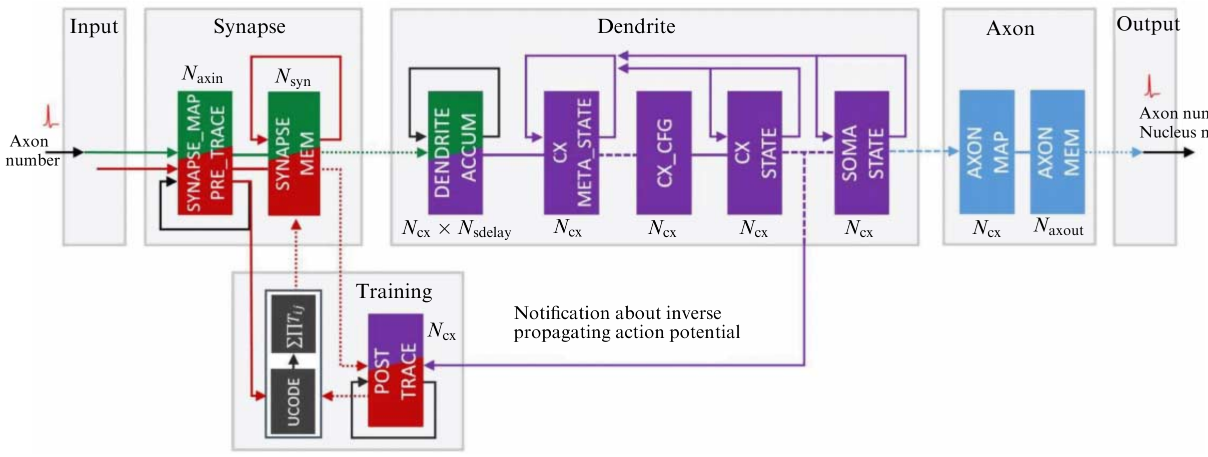

Figure 9. (Color online.) Structure of the CUBA IF neuron in the Loihi system. Colored rectangles correspond to memory units that store the connections, configuration, and dynamic state of all neurons. Color of the lines connecting the memory units and the color of the units themselves indicate the four main modes of operation: processing of input impulses (green), updating of individual units of a neuron (purple), generation of output pulses (blue), and updating of synapses (red). UCODE (Update CODE) black structure implements a customizable learning mechanism. SYNAPSE_MAP PRE-TRACE is a memory cell for describing synapses. SYNAPSE MEM is the state of presynaptic processing, DENDRITE ACCUM is a memory cell for describing dendrites, CX META_STATE is the initial state of the neuron components, CX_CFG is the configuration of the neuron components, CX STATE is the state of the neuron components, SOMA STATE is the soma state, AXON MAP is a memory cell for describing axons, AXON MEM is the state of axons, UCODE→ $\Sigma\Pi T_{ij}$ is the rules for updating learning variables T_{ij}, N_{axin} is the number of input axons, N_{axout} is the number of output axons, N_{syn} is the synaptic memory size, N_{cx} is the number of neuron components, and N_{sdelay} is the smallest synaptic delay.

5.5 ‘Altai’ system

Neuromorphic systems have also been developed in Russia. Novosibirsk-based Motiv LLC presented the Altai neuromorphic chip, which is being developed for use in robotic complexes and specialized systems for classification and recognition problems [149]. The chip itself is a scalable network of neurocores. Each core is a small isolated group of spiking neurons communicating with each other by means of a matrix of connections, which provides a large bandwidth of signals and excludes their mutual blocking. The cores are connected into a 2D network; shared buses and AER (Address Event Representation) relative routing are used for communication between them. In the Altai neuromorphic chip, similar to Loihi, asynchronous operation of the cores is used, but, inside the core, the computations occur in a synchronous way. A modified integrate-and-fire neuron model with a constant leakage value is used as a neuron model. Each neuron has a discrete set of allowed states (integer computations). The chip contains a 32 × 32 lattice of neurocores, i.e., 1024 cores, in which 262,144 neurons and 67,108,864 synapses are located [149]. The chip is made using 28-nm technology.

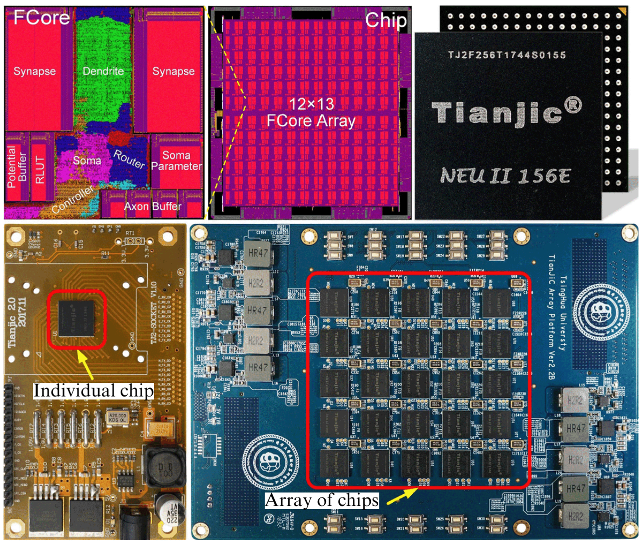

Figure 10. (Color online.) (a) Tianjic neuromorphic chip in a casing and an enlarged single-core circuit in which neural network components and control elements are color-coded. (b) Array of 25 combined chips [150].

5.6 Tianjic system

A group of researchers from China, Singapore, and the United States presented the Tianjic neuromorphic chip [150], which can be used to program two types of networks. The first is a network of rate neurons represented by activation functions of a weighted sum of inputs. The second type is spiking neural networks, in which the integrate-and-fire neuron is used as a model. Both types of network also contain models of dendrites and axons and synaptic connections between them. A single chip can support models of the two types together rather than of only one type. The chip, made using 28-nanometer technology, has a clock frequency of 300 MHz; the chip area is 14.44 mm^2, and its power consumption is 0.95 watts. One chip contains 156 cores, which house 40,000 neurons and 10^6 synapses. Chips can be combined with each other and provide interaction between neural networks located on different chips. Figure 10 shows an image of a single chip and an array of 25 connected chips.

Study [150] reports the results of using such a chip to control a two-wheeled bicycle that follows a person and processes voice commands, i.e., the chip processes and recognizes images, recognizes several voice commands, and also controls several motion parameters (the angular speed of rotation of the wheel, two angles, and transitional–motion speed), while the networks responsible for the corresponding control of a particular function have different connection topologies and are of a different type.

In general, the development of hardware neuromorphic systems mimicking the basic principles of brain operations is aimed at developing and implementing algorithms that

Table. Comparison of characteristics of neuromorphic systems.

Neuromorphic device	Graph of connections	Plasticity	Number of neurons/number of synapses per neuron	Model	Transmission of spike exchange
Neurogrid	Hierarchic (treelike), programmable	No	65,536 (on one chip)/256	Analog LiF	No
SpiNNaker	Hierarchic (treelike), programmable, 2D mesh	Programmable models	10^9/Programmable up to 1000	Programmable models	Possible
BrainScaleS	Hierarchic (treelike)	LTP, STP, STDP	4×10^6/224-14,336	ALIF and/or perceptron	Possible
TrueNorth	Programmable	No	10^6 (on one chip)/256	LIF	Possible
Loihi	Hierarchic (treelike), 2D mesh, address (configurable)	LTP, STP, STDP	130,000 (on one chip)/1000	CUBA IF	No
Altai	Programmable	—	262,144 (on one chip)/256	LIF	—
Tianjic	Programmable	Offline STDP training of weights	40,000 (on one chip)/250	LIF and/or perceptron	—

consume much less energy and are easier to scale to massive parallelization of computing architectures than standard deep learning neural networks and operate in real-time mode (see table).

6. Conclusion

Despite the progress in the experimental study of brain networks on various space and time scales and the development of mathematical modeling of nervous system activity, we are still far from a comprehensive understanding of how the elementary processes of transmission of short pulses between neurons are eventually combined into the implementation by the body of complex sensory, motor, and cognitive functions. One of the approaches, bottom-up is associated with finding a connectome, i.e., a detailed arrangement of connections between nerve cells, and the determination of modes of individual activity of the neurons that make up the network. It would seem that, by having an explicit matrix of connections between neurons and equations describing the dynamics of each of them, conclusions could be drawn about specific functional properties such a network exhibits. However, such a mathematical model with an immense array of variables and parameters turns out to be almost as difficult to analyze as its biological prototype. It can be integrated using supercomputer technologies, obtaining time series, but it is not always possible to answer the question about the mechanisms for performing a particular function based on these data.

An alternative approach, top-down, starts from the analysis of specific transformations of input stimuli into output responses performed by a neural network. Based on machine learning methods, the neural network, primarily its weight coefficients, is adjusted to perform similar transformations formulated as target functions. The resulting artificial neural network is a dynamic system that, on the one hand, takes into account certain aspects of the structure and activity of the prototype neural network, and, on the other hand, can perform the specified target functions. The study of this multidimensional dynamic system using nonlinear-dynamics methods and the theory of complex networks makes it possible to reveal the mechanisms for the implementation of the analyzed sensory, motor, or cognitive functions by the network. The examples of the use of recurrent neural networks presented in this review demonstrate the effectiveness of the described approach. However, the initial assumptions apparently limit the possibilities of a comprehensive understanding of the principles of encoding and processing of cognitive information by the brain.

First, the random recurrent topology of connections contrasts with highly organized and structured parts of the brain, and adding the identified structural properties to the model will most likely improve the performance of the artificial network. For example, in a number of studies, the structure of the model is based on the Dale principle, according to which about 80% of cortical neurons have excitatory connections and 20% have inhibitory connections. However, the complete connectome of connections between neurons has only been identified for the simplest organisms, such as the soil nematode *C. elegans*, while, for higher organisms, the existing tools fail as yet to enable determination of the exact topology with similar detail. On the other hand, in the nervous system of two individuals of the same species, connectomes are not identical to each other, which implies that the task still of importance is to identify common patterns in the structure of connections that surpass individual differences and enable performing complex nervous functions. Modeling using artificial neural networks may help to identify the desired characteristics, since, in the course of optimizing the parameters, the network is tuned to perform certain target tasks, and, using the found properties of the trained network connectome, the importance of a particular structure in solving specific tasks can be hypothesized.

Second, the learning methods in the examples given, based on controlled error minimization using stochastic gradient descent, are not biologically relevant. This observation does not imply that the trained network cannot have operational features down to the level of individual activity of individual neurons, which are characteristic of a biological prototype, since the model system after training nevertheless solves the target problem; however, the way in which it comes to the stage of executing the target function does not reflect the essence of the processes that occurs in biological systems. In this regard, of relevance are the tasks of applying the laws of plasticity of various types already established in the experiment, which are characterized by the fact that, unlike gradient descent, learning takes place online, i.e., in the current time, and locally. This implies that the intensity of synaptic connections is determined by the activity of adjacent neurons and the concentration of available neurotransmitters. In addition, natural learning occurs in the environment, and the target signal, as well as the error signal, is often not explicitly available to the cognitive agent. In such a situation, a solution is found by way of penalties and rewards received from outside, and an adequate method for describing this process in artificial neural networks is the concept of reinforcement learning.

Thus, advancing in the direction of ever greater detailing of the individual activity of neurons and the structure of connections between neurons, based at the same time on models that are effective for machine learning, studies based on this approach are able to identify the dynamic mechanisms of the performance of increasingly complex tasks by brain networks. The implementation of bioinspired plasticity properties in model systems will, in our opinion, make it possible to establish the mechanisms for the high adaptability and multitasking inherent in brain networks.

During the preparation of this review for publication, a number of papers have been published that are relevant to the topics reviewed. Among these publications, the following should be noted that are of importance for the respective sections of the review: [151–157] (Section 3), [158–160] (Section 4), and [161, 162] (Section 5).

This study was supported by the RF Ministry of Science and Higher Education as part of the state assignment to the Institute of Applied Physics of the Russian Academy of Sciences (project no. 00-30-2021-0011).

References

1. Gerstner W et al. *Neuronal Dynamics*: *From Single Neurons to Networks and Models of Cognition* (Cambridge: Cambridge Univ. Press, 2014)
2. Bassett D S, Zurn P, Gold J I *Nat. Rev. Neurosci.* **19** 566 (2018)
3. Breakspear M *Nat. Neurosci.* **20** 340 (2017)
4. Ashwin P, Coombes S, Nicks R *J. Math. Neurosci.* **6** 2 (2016)
5. Izhikevich E M *Dynamical Systems in Neuroscience*: *the Geometry of Excitability and Bursting* (Cambridge, MA: MIT Press, 2007)
6. Rabinovich M I et al. *Rev. Mod. Phys.* **78** 1213 (2006)
7. Hramov A E et al. *Phys. Usp.* **64** 584 (2021); *Usp. Fiz. Nauk* **191** 614 (2021)
8. Ivanitskii G R, Morozov A A *Phys. Usp.* **63** 1092 (2020); *Usp. Fiz. Nauk* **190** 1165 (2020)
9. Ivanitskii G R *Phys. Usp.* **61** 871 (2018); *Usp. Fiz. Nauk* **188** 965 (2018)
10. Doronina-Amitonova L M et al. *Phys. Usp.* **58** 345 (2015); *Usp. Fiz. Nauk* **185** 371 (2015)
11. Klinshov V V, Nekorkin V I *Phys. Usp.* **56** 1217 (2013); *Usp. Fiz. Nauk* **183** 1323 (2013)
12. Pavlov A N et al. *Phys. Usp.* **55** 845 (2012); *Usp. Fiz. Nauk* **182** 905 (2012)
13. Rabinovich M I, Muezzinoglu M K *Phys. Usp.* **53** 357 (2010); *Usp. Fiz. Nauk* **180** 371 (2010)
14. Nekorkin V I *Phys. Usp.* **51** 295 (2008); *Usp. Fiz. Nauk* **178** 313 (2008)
15. Bezruchko B P et al. *Phys. Usp.* **51** 304 (2008); *Usp. Fiz. Nauk* **178** 323 (2008)
16. Borisyuk G N et al. *Phys. Usp.* **45** 1073 (2002); *Usp. Fiz. Nauk* **172** 1189 (2002)
17. Abarbanel H D et al. *Phys. Usp.* **39** 337 (1996); *Usp. Fiz. Nauk* **166** 363 (1996)
18. Ivanitskii G R, Medvinskii A B, Tsyganov M A *Phys. Usp.* **37** 961 (1994); *Usp. Fiz. Nauk* **164** 1041 (1994)
19. Rabinovich M I, Zaks M A, Varona P *Phys. Rep.* **883** 1 (2020)
20. Rabinovich M I, Friston K J, Varona P *Principles of Brain Dynamics*: *Global State Interactions* (Cambridge, MA: MIT Press, 2012)
21. Rabinovich M I et al. *Phys. Life Rev.* **9** (1) 51 (2012)
22. Shumskii S A *Mashinnyi Intellekt. Ocherki po Teorii Mashinnogo Obucheniya i Iskusstvennogo Intellekta* (Machine Intelligence. Essays on the Theory of Machine Learning and Artificial Intelligence) (Moscow: RIOR, 2019)
23. Lake B M et al. *Behavioral Brain Sci.* **40** e253 (2017)
24. Vyas S et al. *Annu. Rev. Neurosci.* **43** 249 (2020)
25. Sussillo D *Current Opin. Neurobiol.* **25** 156 (2014)
26. Barak O *Current Opin. Neurobiol.* **46** 1 (2017)
27. Yang G R, Wang X-J *Neuron* **107** 1048 (2020)
28. Glaser J I et al. *Prog. Neurobiol.* **175** 126 (2019)
29. Marblestone A H, Wayne G, Kording K P *Front. Comput. Neurosci.* **10** 94 (2016)
30. Hassabis D et al. *Neuron* **95** 245 (2017)
31. Cichy R M, Kaiser D *Trends Cognitive Sci.* **23** 305 (2019)
32. Kiselev M *Impul'snye Neironnye Seti. Predstavlenie Informatsii, Obuchenie, Pamyat'* (Spiking Neural Networks: Information Representation, Learning, Memory) (Chisinau: Palmarium Acad. Publ., 2020)
33. Zenke F, Ganguli S *Neural Computat.* **30** 1514 (2018)
34. Nicola W, Clopath C *Nat. Commun.* **8** 2208 (2017)
35. Mozafari M et al. *Pattern Recognit.* **94** 87 (2019); arXiv:1804.00227
36. Bellec G et al. *Adv. Neural Inform. Proc. Syst.* **31** 787 (2018)
37. Schuman C D e al., arXiv:1705.06963
38. James C D et al. *Biol. Inspired Cognitive Arch.* **19** 49 (2017)
39. Chen Y et al. *Integration* **61** 49 (2018)
40. Schmidhuber J *Neural Networks* **61** 85 (2015)
41. Samarasinghe S *Neural Networks for Applied Sciences and Engineering*: *from Fundamentals to Complex Pattern Recognition* (Boca Raton, FL: CRC Press, 2016)
42. Botvinick M et al. *Neuron* **107** 603 (2020)
43. Botvinick M et al. *Trends Cognitive Sci.* **23** 408 (2019)
44. Sompolinsky H, Crisanti A, Sommers H J *Phys. Rev. Lett.* **61** 259 (1988)
45. Jaeger H, Haas H D *Science* **304** 78 (2004)
46. Sussillo D, Abbott L F *Neuron* **63** 544 (2009)
47. Dmitrichev A S et al. *Izv. Vyssh. Uchebn. Zaved. Priklad. Nelin. Dinamika* **26** (4) 5 (2018)
48. Abbott L F *Brain Res. Bull.* **50** 303 (1999)
49. Ermentrout B *Neural Comput.* **8** 979 (1996)
50. Latham P E et al. *J. Neurophysiol.* **83** 808 (2000)
51. Fourcaud-Trocmé N et al. *J. Neurosci.* **23** 11628 (2003)
52. Ermentrout G B, Kopell N *SIAM J. Appl. Math.* **46** 233 (1986)
53. Brette R, Gerstner W *J. Neurophysiol.* **94** 3637 (2005)
54. Douglas R J, Martin K A *J. Physiol.* **440** 735 (1991)
55. Van Vreeswijk C, Sompolinsky H *Science* **274** 1724 (1996)
56. LeCun Y, Bengio Y, Hinton G *Nature* **521** 436 (2015)
57. Sussillo D, Barak O *Neural Comput.* **25** 626 (2013)
58. Maslennikov O V, Nekorkin V I *Nonlin. Dyn.* **101** 1093 (2020)
59. Pugavko M M, Maslennikov O V, Nekorkin V I *Commun. Nonlin. Sci. Numer. Simulat.* **90** 105399 (2020)
60. Pugavko M M, Maslennikov O V, Nekorkin V I *Izv. Vyssh. Uchebn. Zaved. Priklad. Nelin. Dinamika* **28** (1) 77 (2020)
61. Maslennikov O V, Nekorkin V I *Chaos* **29** 103126 (2019)
62. Maslennikov O V, Nekorkin V I *Phys. Usp.* **60** 694 (2017); *Usp. Fiz. Nauk* **187** 745 (2017)
63. Mante V et al. *Nature* **503** 78 (2013)
64. Chaisangmongkon W et al. *Neuron* **93** 1504 (2017)
65. Freedman D J, Assad J A *Nature* **443** 85 (2006)
66. Swaminathan S K, Freedman D J *Nat. Neurosci.* **15** 315 (2012)
67. Wang J et al. *Nat. Neurosci.* **21** 102 (2018)
68. Yang G R et al. *Nat. Neurosci.* **22** 297 (2019)
69. Zhang X et al. *eLife* **8** e43191 (2019)
70. Romo R et al. *Nature* **399** 470 (1999)
71. Genovesio A, Tsujimoto S, Wise S P *Neuron* **63** 254 (2009)
72. Inagaki H K et al. *Nature* **566** 212 (2019)
73. Barraclough D J, Conroy M L, Lee D *Nat. Neurosci.* **7** 404 (2004)
74. Purcell B A, Kiani R *Proc. Natl. Acad. Sci. USA* **113** E4531 (2016)
75. Remington E D et al. *Neuron* **98** 1005 (2018)
76. Padoa-Schioppa C, Assad J A *Nature* **441** 223 (2006)

77. Munoz D P, Everling S *Nat. Rev. Neurosci.* **5** 218 (2004)
78. Wilson H R, Cowan J D *Biophys. J.* **12** (1) 1 (1972)
79. Dayan P, Abbott L F *Theoretical Neuroscience*: *Computational and Mathematical Modeling of Neural Systems* (Cambridge, MA: Massachusetts Institute of Technology Press, 2001)
80. Maass W *Neural Networks* **10** 1659 (1997)
81. Roy K, Jaiswal A, Panda P *Nature* **575** 607 (2019)
82. Tavanaei A et al. *Neural Networks* **111** 47 (2019)
83. Pfeiffer M, Pfeil T *Front. Neurosci.* **12** 774 (2018)
84. Perrinet L, Samuelides M, Thorpe S *IEEE Trans. Neural Networks* **15** 1164 (2004)
85. Escobar M-J *Int. J. Comput. Vision* **82** 284 (2009)
86. Brette R et al. *J. Comput. Neurosci.* **23** 349 (2007)
87. Gerstner W, van Hemmen J L *Network Comput. Neural Syst.* **3** (2) 139 (1992)
88. Sommer F T, Wennekers T *Neural Networks* **14** 825 (2001)
89. Amit D J, Mongillo G *Neural Comput.* **15** 565 (2003)
90. Mongillo G, Barak O, Tsodyks M *Science* **319** 1543 (2008)
91. Szatmáry B, Izhikevich E M *PLoS Comput. Biol.* **6** e1000879 (2010)
92. Buzsáki G *Rhythms of the Brain* (Oxford: Oxford Univ. Press, 2006)
93. Izhikevich E M, Edelman G M *Proc. Natl. Acad. Sci. USA* **105** 3593 (2008)
94. Ma J, Wu J *Neural Comput.* **19** 2124 (2007)
95. Oster M, Douglas R, Liu S-C *Neural Comput.* **21** 2437 (2009)
96. Jin D Z, Seung H S *Phys. Rev. E* **65** 051922 (2002)
97. Lumer E D *Neural Comput.* **12** 181 (2000)
98. Wang X-J *Neuron* **60** 215 (2008)
99. Atiya A F, Parlos A G *IEEE Trans. Neural Networks* **11** 697 (2000)
100. Dominey P F, Ramus F *Language Cognitive Proces.* **15** (1) 87 (2000)
101. Jaeger H, Technical GMD Report 148 (Boon: German National Research Center for Information Technology GMD, 2001) p. 13
102. Maass W, Natschläger T, Markram H *Neural Comput.* **14** 2531 (2002)
103. Schrauwen B, Verstraeten D, Van Campenhout J M, in *Proc. of the 15th European Symp. on Artificial Neural Networks, ESANN 2007, Bruges, Belgium, April 25–27, 2007*, p. 471
104. Lukoševičius M, Jaeger H *Comput. Sci. Rev.* **3** (3) 127 (2009)
105. Häusler S, Markram H, Maass W *Complexity* **8** (4) 39 (2003)
106. Joshi P, Maass W *Neural Comput.* **17** 1715 (2005)
107. Burgsteiner H, in *Novel Applications of Neural Networks in Engineering. Proc. of the 9th Intern. Conf. on Engineering Applications of Neural Networks, 24–26 August 2005, Lille, France* (Douai: École des Mines de Douai, 2005) p. 129
108. Kasiński A, Ponulak F *Int. J. Appl. Math. Comput. Sci.* **16** 101 (2006)
109. Verstraeten D et al. *Inform. Proces. Lett.* **95** 521 (2005)
110. Ponulak F, Kasiński A *Neural Comput.* **22** 467 (2010)
111. Raven J C, Court J H, Raven J *Manual for Raven's Progressive Matrices and Vocabulary Scales* (Oxford: Oxford Psychologists Press, 1998)
112. Eliasmith C et al. *Science* **338** 1202 (2012)
113. Voelker A R, Eliasmith C *Neural Comput.* **30** 569 (2018)
114. Bekolay T et al. *Front. Neuroinform.* **7** 48 (2014)
115. Boahen K *Comput. Sci. Eng.* **19** (2) 14 (2017)
116. Liu S-C et al. *Analog VLSI*: *Circuits and Principles* (Cambridge, MA: MIT Press, 2002)
117. Indiveri G et al. *Front. Neurosci.* **5** 73 (2011)
118. Tang J et al. *Adv. Mater.* **31** 1902761 (2019)
119. Kelly J E (III), Hamm S *Smart Machines*: *IBM's Watson and the Era of Cognitive Computing* (New York: Columbia Business School Publ., 2013)
120. Zheng N, Mazumder P *Learning in Energy-Efficient Neuromorphic Computing*: *Algorithm and Architecture Co-Design* (Hoboken, NJ: Wiley–IEEE Press, 2019)
121. Minvielle R, Bayoumi M, in *2013 4th Annual Intern. Conf. on Energy Aware Computing Systems and Applications, ICEAC* (Piscataway, NJ: IEEE, 2013) p. 125
122. Bhardwaj K, Nowick S M *IEEE Trans. Very Large Scale Integrat. VLSI Syst.* **27** (2) 350 (2019)
123. Kim M-K et al. *Science* **23** 101846 (2020)
124. Benjamin B V et al. *Proc. IEEE* **102** 699 (2014)
125. Menon S et al., in *Proc. of the 5th IEEE/RAS–EMBS Intern. Conf. on Biomedical Robotics and Biomechatronics* (Piscataway, NJ: IEEE, 2014) p. 181
126. Neckar A et al. *Proc. IEEE* **107** 144 (2019)
127. Markram H et al. *Procedia Comput. Sci.* **7** 39 (2011); in *Proc. of the 2nd European Future Technologies Conf. and Exhibition, FET 11, 4–6 May 2011, Budapest, Hungary* (Amsterdam: Elsevier, 2011) p. 11
128. Schemmel J et al., in *ISCAS 2010, 2010 IEEE Intern. Symp. on Circuits and Systems Nano-Bio Circuit Fabrics and Systems, May 30th–June 2nd, 2010, Paris, France* (Piscataway, NJ: IEEE, 2010) p. 1947
129. Scholze S et al. *Front. Neurosci.* **5** 117 (2011)
130. Schemmel J, in *Proc. of the 6th Annual Neuro Inspired Computational Elements, NICE, Conf., Hillsboro, Oregon, 2018*
131. Furber S, Brown A, in *Proc. of the Ninth Intern. Conf. on Application of Concurrency to System Design, 1–3 July 2009, Augsburg, Germany* (Eds S Edwards, R Lorenz, W Vogler) (Los Alamitos, CA: IEEE Computer Soc., 2009) p. 3
132. Höppner S, Mayr C, in *Proc. of the 6th Annual Neuro Inspired Computational Elements, NICE, Conf., Hillsboro, Oregon, 2018*
133. Tayeb Z, Erçelik E, Conradt J, in *Proc. of the 8th Intern. IEEE EMBS Conf. on Neural Engineering NER, May 25–28, 2017, Shanghai, China* (Piscataway, NJ: IEEE, 2017) p. 263
134. Galluppi F et al. in *2014 IEEE Intern. Conf. on Robotics and Automation, ICRA, 31 May – 5 June, 2014, Hong Kong, China* (Piscataway, NJ: IEEE, 2014) p. 2862
135. Denk C et al., in *Artificial Neural Networks and Machine Learning — ICANN 2013, 23rd Intern. Conf. on Artificial Neural Networks, Sofia, Bulgaria, September 10–13, 2013, Proc.* (Lecture Notes in Computer Science, Vol. 8131, Eds V Mladenov et al.) (Berlin: Springer, 2013) p. 467
136. Kungl A F et al. *Front. Neurosci.* **13** 1201 (2019)
137. Wunderlich T et al. *Front. Neurosci.* **13** 260 (2019)
138. Akopyan F et al. *IEEE Trans. Computer-Aided Design Integrated Circuits Syst.* **34** 1537 (2015)
139. DeBole M V et al. *Computer* **52** (5) 20 (2019)
140. Amir A et al., in *The 2013 Intern. Joint Conf. on Neural Networks IJCNN, 4–9 Aug. 2013* (Piscataway, NJ: IEEE, 2013) p. 1
141. Merolla P et al., in *2011 IEEE Custom Integrated Circuits Conf., CICC* (Piscataway, NJ: IEEE, 2011) p. 1
142. Esser S K et al. *Proc. Natl. Acad. Sci. USA* **113** 11441 (2016)
143. Padala V, Basu A, Orchard G *Front. Neurosci.* **12** 118 (2018)
144. Davies M et al. *IEEE Micro* **38** (1) 82 (2018)
145. Vogels T P, Abbott L F *J. Neurosci.* **25** 10786 (2005)
146. Viale A et al., arXiv:2107.00401
147. Rueckauer B et al., arXiv:2101.04261
148. Imam N, Cleland T A *Nat. Mach. Intell.* **2** 181 (2020)
149. Kangler V M, Panchenko K E, in *Bespilotnye Transportnye Sredstva s Elementami Iskusstvennogo Intellekta. Tretii Vserossiiskii Nauchno-Prakticheskii Seminar, 22–23 Sentyabrya 2015, Innopolis, Respublika Tatarstan, Rossiya. Trudy Seminara. Rossiiskaya Assotsiatsiya Iskusstvennogo Intellekta* (Proc. Third All-Russian Scientific and Practical Seminar "Unmanned Vehicles with Elements of Artificial Intelligence", September 22–23, 2015, Innopolis, Republic of Tatarstan, Russia) (Moscow: Pero, 2016) p. 169
150. Pei J et al. *Nature* **572** 106 (2019)
151. Yang G R, Molano-Mazón M *Current Opin. Neurobiol.* **70** 182 (2021)
152. Pulvermüller F et al. *Nat. Rev. Neurosci.* **22** 488 (2021)
153. Zhang X, Liu,S, Chen Z S *iScience* **24** 102919 (2021)
154. Goulas A, Damicelli F, Hilgetag C C *Neural Networks* **142** 608 (2021)
155. Maslennikov O V *Izv. Vyssh. Uchebn. Zaved. Priklad. Nelin. Dinamika* **29** 799 (2021)
156. Ehrlich D B et al. *eNeuro* **8** (1) 0427-20 (2021)
157. Chung S, Abbott L F *Current Opin. Neurobiol.* **70** 137 (2021)
158. Xue X et al. "Spiking recurrent neural networks represent task-relevant neural sequences in rule-dependent computation" *Cogn. Comput.* (2022) https://doi.org/10.1007/s12559-022-09994-2
159. Calaim N et al. *eLife* **11** e73276 (2022)
160. Pugavko M M, Maslennikov O V, Nekorkin V I *Radiophys. Quantum Electron.* **64** 736 (2022); *Izv. Vyssh. Uchebn. Zaved. Radiofiz.* **64** 817 (2021)
161. Schuman C D et al. *Nat. Comput. Sci.* **2** 10 (2022)
162. Ivanov D et al. *Front. Neurosci.* (2022) https://doi.org/10.3389/fnins.2022.959626