JOURNAL OF OPTIMIZATION, DIFFERENTIAL EQUATIONS AND THEIR APPLICATIONS (JODEA)
Volume **31**, Issue **2**, December **2023**, pp. 24–49, DOI 10.15421/142309

ISSN (print) 2617–0108
ISSN (on-line) 2663–6824

SYSTEMS OF SINGULAR DIFFERENTIAL EQUATIONS AS THE BASIS FOR NEURAL NETWORK MODELING OF CHAOTIC PROCESSES

Vasiliy Ye. Belozyorov*, Oleksandr A. Inkin†

Abstract. Currently, systems of neural ordinary differential equations (ODEs) have become widespread for modeling various dynamic processes. However, in forecasting tasks, priority remains with the classical neural network approach to building a model. This is due to the fact that by choosing the neural network architecture, a more accurate approximation of the trajectories of a dynamic system can be achieved. It is known that the accuracy of the mentioned approximation significantly depends on the settings of the neural network parameters and their initial values. In this regard, the main idea of the article is that the initial values of the neural network parameters are taken to be equal to the parameters of the neural ODE system obtained by modeling the same process, which will then be simulated using a neural network. Subsequently, the singular ODE system was used to adjust the parameters of the LSTM (Long Short Term Memory) neural network. The results obtained were used to model the process of epilepsy.

Key words: time series, system of differential equations, compact region of attraction, neural network.

2010 Mathematics Subject Classification: 34A34, 34D20, 37D45, 93A30.

Communicated by Prof. V. Kapustyan

1. Introduction

Let

$$x_0 = x(t_0), x_1 = x(t_1), ..., x_N = x(t_N) \tag{1.1}$$

be a finite sequence (time series) of numerical values of some scalar dynamical variable $x(t)$ measured with the constant time step Δt in the moments $t_i = t_0 + i\Delta t$; $x_i = x(t_i)$; $i = 0, 1, ..., N$ (thus, $\Delta t = t_N/N$) [5, 6, 9, 11, 12, 16, 21].

The choice of equations for a model that describes the dynamics of certain processes is a difficult task. Experiments show that the most logical approach to constructing models that describe the dynamics of the passage of electrical

*Department of Applied Mathematics, Oles Honchar Dnipro National University, 72, Gagarin's Avenue,49010, Dnipro, Ukraine, belozvye2017@gmail.com

†Department of Applied Mathematics, Oles Honchar Dnipro National University, 72, Gagarin's Avenue, 49010, Dnipro, Ukraine, inkin.work@gmail.com

signals through certain objects is based on the use of well-known physical laws (for example Ohm, Maxwell, Joule-Lenz, the law of conservation of energy), in which the interaction between measured quantities is described with using quadratic functions.

In addition, in rapidly oscillating processes there is a sharp change in the sign of the derivative. It is this characteristic that most often determines chaotic processes. Therefore, we believe that a sufficiently informative model of chaos can be described by differential equations, on the right sides of which there are rational functions with quadratic functions in the numerator and periodic functions that take sufficiently small non-zero values in the denominator. Such ODE systems are called singular [9].

In order to construct the mentioned system of differential equations using a known time series (1.1), it is necessary to know its dimension. The last characteristic (dimension n of the embedding space) and the optimal time delay τ (at which time t must be shifted to obtain a new variable $y(t_i) = x(t_i + \tau)$) can be determined using recurrent qualitative analysis (RQA) methods [21]. (Note that the number τ must be such that $t_i + \tau \in \{t_0, t_1, ..., t_N\}; i \in \{0, ..., N-(n-1)\cdot\tau\}$.)

Having parameters n and τ, we can assume that to model a process described by time series (1.1), a certain system of differential equations has already built. (In what follows, we will assume that a recurrent neural network (RNN), which is a discrete analogue of the mentioned ODE system, was also constructed [4, 6, 11, 16, 21].)

Below we will focus on two areas of research, which can be formulated in the following questions.

1. If a neural network models a certain dynamic process, then how to guarantee the stability or boundedness of solutions of the system of differential equations describing a continuous analog of the aforementioned neural network?

2. In the theory of bifurcations, the following result is well known: in any determinate system, chaotic processes arise as a result of bifurcations of limit cycles or homoclinic orbits [17, 18]. Therefore, how to design the architecture of neural ODEs system so that the resulting architecture would generate a limit cycle? (It is now known that most types of chaos in systems of differential equations begin with bifurcations of limit cycles [3, 7, 8].)

The final sections of the article are devoted to the development of an algorithm for determining the parameters of ODE systems for a known time series. The essence of this algorithm is that it uses a special structure of neural ODEs (antisymmetric neural ODEs), with which it is possible to generate a limit cycle [6, 10, 13]. After this, by selecting weight coefficients, we obtain such bifurcations of the indicated cycle that lead to the modeling of a real chaotic process. Subsequently, the found weighting coefficients are used as initial data for adjusting the parameters of the LSTM neural network [1].

2. Mathematical preliminaries

By $\mathbf{x} = (x_1, ..., x_n)^T$ it denotes an arbitrary vector of real space $\mathbb{R}^n$. Consider the real system of ordinary autonomous differential equations

$$\begin{cases} \dot{x}_1(t) = \dfrac{f_1(x_1(t), ..., x_n(t))}{1 - \vartheta \cdot u_1(x_1(t), ..., x_n(t))}, \\ . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; , \\ \dot{x}_n(t) = \dfrac{f_n(x_1(t), ..., x_n(t))}{1 - \vartheta \cdot u_n(x_1(t), ..., x_n(t))} \end{cases} \tag{2.1}$$

of order n with the vector of initial data $\mathbf{x}^T(0) = (x_{10}, ..., x_{n0})$. Here $f_i(x_1, ..., x_n)$, $u_i(x_1, ..., x_n)$; $i = 1, ..., n$, are continuous functions of their arguments, and for functions $u_i(x_1, ..., x_n)$ the condition

$$\Omega = \max_{1 \le i \le n} \sup_{\|\mathbf{x}\| \to \infty} (|u_i(x_1, ..., x_n)|) < \infty$$

is satisfied. In addition, ϑ is a real parameter such that $0 \le |\vartheta| < 1/\Omega$.

Definition 2.1. System (2.1) will be called singular.

Let $A = (a_{ij}), B_1, ..., B_n \in \mathbb{R}^{n \times n}$ be real matrices. In addition, let the matrices $B_1 = (b_{ij}^{(1)}), ..., B_n = (b_{ij}^{(n)})$ be symmetrical; $i, j = 1, ..., n$. Let us consider one special case of system (2.1):

$$\begin{cases} \dot{x}_1(t) = \dfrac{\sum\limits_{j=1}^{n} a_{1j} x_j(t) + \mathbf{x}^T(t) B_1 \mathbf{x}(t)}{1 - \vartheta \cdot u_1(x_1(t), ..., x_n(t))}, \\ . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; , \\ \dot{x}_n(t) = \dfrac{\sum\limits_{j=1}^{n} a_{nj} x_j(t) + \mathbf{x}^T(t) B_n \mathbf{x}(t)}{1 - \vartheta \cdot u_n(x_1, ..., x_n(t))}. \end{cases} \tag{2.2}$$

Below we recall some of the results obtained in [7,9].

Consider the system of ordinary autonomous quadratic differential equations

$$\begin{cases} \dot{x}_1(t) = \sum\limits_{j=1}^{n} a_{1j} x_j(t) + \mathbf{x}^T(t) B_1 \mathbf{x}(t), \\ . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; . \; , \\ \dot{x}_n(t) = \sum\limits_{j=1}^{n} a_{nj} x_j(t) + \mathbf{x}^T(t) B_n \mathbf{x}(t). \end{cases} \tag{2.3}$$

Assume that the region of attraction for the solutions of system (2.3) is a ball

$$\mathbb{B} \equiv (x_1 + \gamma_1)^2 + ... + (x_n + \gamma_n)^2 - R^2 \le 0$$

of radius R with center at point $(-\gamma_1, ..., -\gamma_n)^T$.

Let also the elements of matrices $B_1, ..., B_n$ satisfy the following three groups of restrictions:
C_n^1 one-term restrictions

$$b_{ii}^{(i)} x_i^3 \equiv 0; i = 1, ..., n; \tag{2.4}$$

$2C_n^2$ two-term restrictions

$$b_{jj}^{(i)} x_i x_j^2 + b_{ij}^{(j)} x_i x_j^2 \equiv 0; i \neq j; i, j = 1, ..., n; \tag{2.5}$$

C_n^3 three-term restrictions

$$b_{jk}^{(i)} x_i x_j x_k + b_{ik}^{(j)} x_i x_j x_k + b_{ij}^{(k)} x_i x_j x_k \equiv 0; i \neq j \neq k; i, j, k = 1, ..., n. \tag{2.6}$$

As shown in [5], for small values of n system (2.3), taking into account restrictions (2.4), (2.5), and (2.6), has the following form:

$n = 3$

$$\begin{cases} \dot{x}(t) = a_{11}x + \cdots + a_{13}z + b_{12}xy + b_{13}xz + b_{22}y^2 + b_{23}yz + b_{33}z^2, \\ \dot{y}(t) = a_{21}x + \cdots + a_{23}z - b_{12}x^2 - b_{22}xy + c_{13}xz + c_{23}yz + c_{33}z^2, \\ \dot{z}(t) = a_{31}x + \cdots + a_{33}z - b_{13}x^2 - (b_{23} + c_{13})xy - b_{33}xz - c_{23}y^2 - c_{33}yz; \end{cases} \tag{2.7}$$

$n = 4$

$$\begin{cases} \dot{x}(t) = a_{11}x + \cdots + a_{14}u + b_{12}xy + b_{13}xz + b_{14}xu + b_{22}y^2 \\ \qquad +b_{23}yz + b_{24}yu + b_{33}z^2 + b_{34}zu + b_{44}u^2, \\ \dot{y}(t) = a_{21}x + \cdots + a_{24}u - b_{12}x^2 - b_{22}xy + c_{13}xz + c_{14}xu \\ \qquad +c_{23}yz + c_{24}yu + c_{33}z^2 + c_{34}zu + c_{44}u^2, \\ \dot{z}(t) = a_{31}x + \cdots + a_{34}u - b_{13}x^2 - (b_{23} + c_{13})xy - b_{33}xz \\ \qquad +d_{14}xu - c_{23}y^2 - c_{33}yz + d_{24}yu + d_{34}zu + d_{44}u^2, \\ \dot{u}(t) = a_{41}x + \cdots + a_{44}u - b_{14}x^2 - (b_{24} + c_{14})xy - (b_{34} + d_{14})xz \\ \qquad -b_{44}xu - c_{24}y^2 - (c_{34} + d_{24})yz - c_{44}yu - d_{34}z^2 - d_{44}zu. \end{cases} \tag{2.8}$$

Note that equations (2.7) – (2.8) are presented in this detailed form solely for the convenience of users. In the case of arbitrary n, the system that satisfies the conditions (2.4) – (2.6) looks like this:

$$\dot{\mathbf{x}}(t) = (A + B(\mathbf{x}) - B^T(\mathbf{x})) \cdot \mathbf{x}. \tag{2.9}$$

Here

$$B(\mathbf{x}) = \begin{pmatrix} 0 & b_{12}^1 x_1 + b_{22}^1 x_2 & b_{13}^1 x_1 + b_{23}^1 x_2 + b_{33}^1 x_3 & \dots & \sum\limits_{i=1}^{n} b_{in}^1 x_i \\ 0 & 0 & b_{13}^2 x_1 + b_{23}^2 x_2 + b_{33}^2 x_3 & \dots & \sum\limits_{i=1}^{n} b_{in}^2 x_i \\ 0 & 0 & 0 & \dots & \sum\limits_{i=1}^{n} b_{in}^3 x_i \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & \sum\limits_{i=1}^{n} b_{in}^{n-1} x_i \\ 0 & 0 & 0 & \dots & 0 \end{pmatrix},$$

$b_{ij}^k \in \mathbb{R}; i,j,k \in \{1,...,n\}$. (It is clear that $\mathbf{x}^T \cdot (B(\mathbf{x}) - B^T(\mathbf{x})) \cdot \mathbf{x} \equiv 0$.)

The method for finding the radius R of sphere $\mathbb{B}$ and its center $(-\gamma_1, ..., -\gamma_n)^T$ is presented in [5].

Below we will use the following well-known result:

Theorem 2.1. *(**LaSalle's Theorem** [14]). Let $\mathbb{H} \subset \mathbb{R}^n$ be a compact set that is positively invariant with respect to (2.2). Let $V : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function such that $\dot{V}(\mathbf{x}) \leq 0$ (or $\dot{V}(\mathbf{x}) \geq 0$) in $\mathbb{H}$. Let $\mathbb{E}$ be the set of all points in $\mathbb{H}$ where $\dot{V}(\mathbf{x}) = 0$. Let $\mathbb{M}$ be the largest invariant set in $\mathbb{E}$. Then every solution starting in $\mathbb{H}$ approaches $\mathbb{M}$ as $t \to +\infty$.*

Let $s_1, ..., s_n$ be unknown real constants. Let us construct from matrices A and $B_1, ..., B_n$ of system (2.2) the following matrix:

$$F(s_1, ..., s_n) := (A^T + A)/2 + s_1 B_1 + ... + s_n B_n.$$

Introduce also the following function $V : \mathbb{R}^n \to \mathbb{R}$:

$$V(x_1, ..., x_n) = \frac{1}{2}\sum_{i=1}^{n} \int (1 - \vartheta \cdot u_i(x_1, ..., x_n))(x_i + s_i)\, dx_i,$$

where $\forall i$ $(1 - \vartheta \cdot u_i(x_1, ..., x_n)) > 0$. (The indefinite integral symbol is used here.)

It is obvious that

$$\dot{V}_t = \frac{1}{2}\sum_{i=1}^{n}(1 - \vartheta \cdot u_i(x_1, ..., x_n))(x_i + s_i)\dot{x}_i$$
$$= \mathbf{x}^T(\frac{1}{2}(A + A^T) + \sum_{i=1}^{n} s_i B_i)\mathbf{x} + L(\mathbf{x}), \quad (2.10)$$

(Here the derivative $\dot{V}_t$ is defined by virtue of the equations (2.2); $L(\mathbf{x})$ is a cubic function of variables $x_1, ..., x_n$ without quadratic terms.)

Let R be a positive constant. We define the set $\mathbb{B}_R \subset \mathbb{R}^n$ as follows:

$$\mathbb{B}_R := \{(x_1, ..., x_n) \in \mathbb{R}^n | V(x_1, ..., x_n) - R^2 \leq 0\}. \quad (2.11)$$

Introduce the following sets:

$$\mathbb{D}_- := \{(x_1, ..., x_n) \in \mathbb{R}^n | \dot{V}_t(x_1, ..., x_n) \leq 0\}, \quad (2.12)$$

$$\mathbb{D}_+ := \{(x_1, ..., x_n) \in \mathbb{R}^n | \dot{V}_t(x_1, ..., x_n) \geq 0\}, \quad (2.13)$$

and

$$\mathbb{L} := \{(x_1, ..., x_n) \in \mathbb{R}^n | \dot{V}_t(x_1, ..., x_n) = 0\}. \quad (2.14)$$

Theorem 2.2. *Let's assume that for system (2.2) the following conditions:*

1) the matrices B_i satisfy the restriction $\mathbf{x}^T(B(\mathbf{x}) - B^T(\mathbf{x}))\mathbf{x} \equiv 0$;

2) there are real constants $s_1^*, ..., s_n^*$ such that the matrix $F(s_1^*, ..., s_n^*)$ is negative definite, are satisfied.

Then there exists the compact region of attraction $\mathbb{H} = \mathbb{D}_+ \neq \emptyset$ for trajectories of system (2.2).

Proof. Condition 1) guarantees that the function $\dot{V}_t(x_1, ..., x_n)$ contains only linear and quadratic terms and does not contain cubic terms.

It remains only to clarify condition 2). So, let there exist numbers $s_1^*, ..., s_n^*$ such that the matrix $F(s_1^*, ..., s_n^*)$ is negative definite.

Proof of condition 2) split into two parts.

2a) The function $V(x_1, ..., x_n)$ for $s_i = 0$ positive definite and in this case $\lim\limits_{\|\mathbf{x}\|\to\infty} V(x_1, ..., x_n) = \infty$. Thus, by virtue of the construction of the function $V(x_1, ..., x_n)$, the sets $\mathbb{B}_R$ and $\mathbb{L}$ are compact. Therefore, we can choose R such that $\mathbb{B}_R \cap \mathbb{D}_- \neq \emptyset$ and $\mathbb{L} \subset \mathbb{B}_R$. Then, in the region $\mathbb{B}_R \cap \mathbb{D}_-$, we can assert that $V(x_1(t), ..., x_n(t))$ is a decreasing function of t. Since $V(x_1(t), ..., x_n(t))$ is continuous on the compact set $\mathbb{B}_R$, it is bounded from below on $\mathbb{B}_R$. Therefore, $V(x_1(t), ..., x_n(t))$ has a finite limit as $t \to \infty$. Then, according to Theorem (2.1), we can assume that $\mathbb{H} = \mathbb{B}_R \cap \mathbb{D}_-$ and $\mathbb{H}$ is a the compact region of attraction for trajectories of system (2.2).

2b) Now we choose the radius R so large that the set $\mathbb{B}_R \cap \mathbb{D}_+ = \mathbb{D}_+ \neq \emptyset$. (Note that, by virtue of 2), the set $\mathbb{D}_+$ is compact. Therefore, we have $\mathbb{L} \subset \mathbb{B}_R$.) Then, in the domain $\mathbb{B}_R \cap \mathbb{D}_+$ the function $V(x_1(t), ..., x_n(t))$ is an increasing function of t. Since the function $V(x_1(t), ..., x_n(t))$ is continuous on the compact set $\mathbb{D}_+$, it is bounded from above on $\mathbb{D}_+$ and has a finite limit as $t \to \infty$.

Thus, from items 2a) and 2b) it follows that, regardless of the starting point $\mathbf{x}^T(0) \in \mathbb{R}^n$, the trajectory $V(x_1(t)), ..., x_n(t))$ will be attracted to the boundary $\dot{V}_t(x_1, ..., x_n) = 0$ (this is $\mathbb{L}$) of the compact set $\mathbb{D}_+$. This means that there exists an attractor belonging to the region $\mathbb{D}_+$. (An equilibrium point can act as such attractor.) $\square$

2.1. Model design

This article is a continuation of work [9]. Therefore, the motives leading to certain results are based on the assumptions introduced in article [9].

The main object of study in this work will be the electroencephalograms (EEGs) of the brain of patients suffering from epilepsy. Now, we will consider EEGs obtained for healthy and sick patients (see Fig.2.1). (The main features of the processes Fig.2.1 are described in [20].)

Naturally, when modeling the real process of epilepsy, it is impossible to take into account all these features. However, we will try to at least establish the trend accompanying such processes.

The latest considerations allow us to use system (2.2) for modeling the processes represented on encephalograms, in which $cos(...)$ is used as functions $u_i(...)$; $i = 1, ..., n$. The final appearance of this system is as follows:

$$\begin{cases} \dot{x}_1(t) = \dfrac{\displaystyle\sum_{j=1}^{n} a_{1j}x_j(t) + \mathbf{x}^T(t)B_1\mathbf{x}(t)}{1 - \vartheta \cdot \cos(x_1(t))}, \\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ , \\ \dot{x}_n(t) = \dfrac{\displaystyle\sum_{j=1}^{n} a_{nj}x_j(t) + \mathbf{x}^T(t)B_n\mathbf{x}(t)}{1 - \vartheta \cdot \cos(x_n(t))}, \end{cases} \tag{2.15}$$

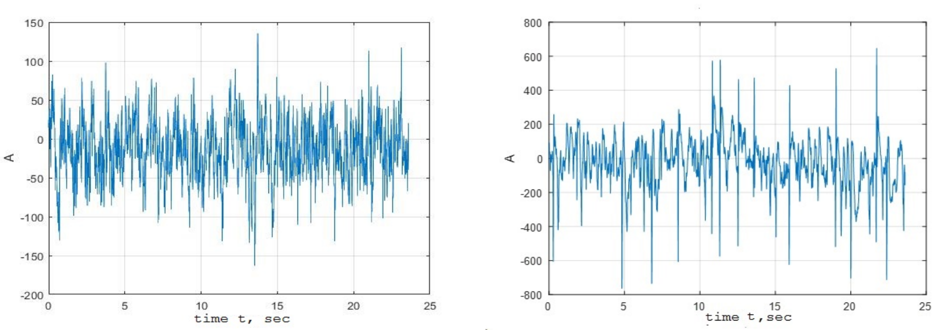
where ϑ is a real parameter such that $0 \leq |\vartheta| < 1/\Omega = 1/1 = 1$. (Note that the simplest case of system (2.15), in which $B_1 = ... = B_n = 0$ was investigated in [9].)

In what follows, instead of the system (2.15), we will sometimes consider the system

$$\begin{cases} \dot{x}_1(t) = \dfrac{a_{10} + \displaystyle\sum_{j=1}^{n} a_{1j}x_j(t) + \mathbf{x}^T(t)B_1\mathbf{x}(t)}{1 - \vartheta \cdot \cos(x_1(t))}, \\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ , \\ \dot{x}_n(t) = \dfrac{a_{n0} + \displaystyle\sum_{j=1}^{n} a_{nj}x_j(t) + \mathbf{x}^T(t)B_n\mathbf{x}(t)}{1 - \vartheta \cdot \cos(x_n(t))}, \end{cases} \tag{2.16}$$

where $a_{10}, ..., a_{n0} \in \mathbb{R}$.

We assume that $x_1 = \phi_1, ..., x_n = \phi_n$ is a real solution to the system of



(a1) (a2)

Fig. 2.1. The electroencephalogram taken from a certain point in the cerebral cortex: (a1) a healthy patient, (a2) a patient with an epileptic disease (see [20]).

algebraic equations

$$a_{10}+\sum_{j=1}^{n}a_{1j}x_j+\mathbf{x}^TB_1\mathbf{x}=0,...,a_{n0}+\sum_{j=1}^{n}a_{nj}x_j+\mathbf{x}^TB_n\mathbf{x}=0.$$

Let us introduce new variables $y_1,...,y_n$ into system (2.16) using the formulas: $x_1=y_1+\phi_1,...,x_n=y_n+\phi_n$. Then we have

$$\begin{cases} \dot{y}_1(t)=\dfrac{\sum\limits_{j=1}^{n}c_{1j}y_j(t)+\mathbf{y}^T(t)B_1\mathbf{y}(t)}{1-\vartheta\cdot\cos(y_1(t)+\phi_1)}, \\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ , \\ \dot{y}_n(t)=\dfrac{\sum\limits_{j=1}^{n}c_{nj}y_j(t)+\mathbf{y}^T(t)B_n\mathbf{y}(t)}{1-\vartheta\cdot\cos(y_n(t)+\phi_n)}, \end{cases} \tag{2.17}$$

where $c_{ij}\in\mathbb{R}; i,j=1,...,n$. Then the statements of Theorems 2.1 and 2.2 can be applied to system (2.17) (and therefore (2.16)).

3. Two-stage neural network modeling procedure

It is known that one of the most common methods for adjusting the weighting coefficients of a neural network is the steepest descent method. This method is a type of gradient descent, which means it descends the error surface, continuously adjusting the weights towards the minimum. The error surface of a complex network is highly rugged and consists of hills, valleys, folds and ravines in high-dimensional space. A network may fall into a local minimum (shallow valley) when there is a much deeper minimum nearby. At the local minimum point, all directions lead upward, and the network is unable to escape from it. The main difficulty in training neural networks is precisely the methods for exiting local minima: each time you exit a local minimum, the next local minimum is again searched until it is no longer possible to find a way out of it.

In this regard, the following two-stage modeling method suggests finding an initial point in the space of weighting coefficients (parameters) at which the error function would be as close as possible to the local minimum point. Subsequently, the found point is used as a starting point for adjusting the parameters of some recurrent neural network (the LSTM neural network) using the back propagation method.

3.1. First stage

At this stage, we will try to solve the problem of parametric identification of system (2.16).

Let us write the equations of system (2.16) in the following form

$$\dot{x}_i(t) = \frac{a_{i0} + a_{i1}x_i + \cdots + a_{in}x_n + \mathbf{x}^T B_i \mathbf{x}}{1 - \vartheta \cdot \cos(x_i)} = \phi_i(x_1, ..., x_n); i = 1, ..., n. \quad (3.1)$$

Now we rewrite the equations of system (3.1) as follows

$$\dot{x}_i(t) = a_{i0} + a_{i1}x_i + \cdots + a_{in}x_n + \mathbf{x}^T B_i \mathbf{x} + \dot{x}_i \vartheta \cdot \cos(x_i) = \psi_i(x_1, ..., x_n); i = 1, ..., n. \quad (3.2)$$

From the point of view of the theory of differential equations, systems (3.1) and (3.2) describe the same dynamics. However, from the point of view of approximation theory (determining the coefficients $a_{i0}, ..., a_{in}, B_i, \vartheta$ from the known values of the functions $x_i(t), i = 1, ..., n$), these are different problems for systems (3.1) and (3.2).

Indeed, in case of system ((3.1)) it is necessary to minimize by $a_{10}, ..., B_n, \vartheta$ the loss function $\sum_{i=1}^{n} |\dot{x}_i - \phi_i(x_1, ..., x_n, a_{10}, ..., B_n, \vartheta)|$, and in case of system (3.2) it is necessary to minimize by $a_{10}, ..., B_n, \vartheta$ the loss function $\sum_{i=1}^{n} |\dot{x}_i - \psi_i(x_1, ..., x_n, a_{10}, ..., B_n, \vartheta)|$, where the equations (3.1) are rational and the equations (3.2) are linear.

It is clear that in the case of system (3.2), the approximation problem will be simpler than in the case of system (3.1). That is why we chose system (3.2) for solving the approximation problem. (It should be remembered that the approximation results for system (3.2) may be worse than for system (3.1).)

In the study of dynamic processes, as a rule, only a few variables describing the process are available for direct measurement. The remaining variables (the so-called hidden variables) are inaccessible to observation. This raises the problem of reconstructing these unobserved variables from known observable variables. The first step towards solving this problem is to establish the minimum number of all variables (measured and hidden) on which the dynamic process depends.

In article [9] it was shown that for time series obtained using EEG, the dimension of the embedding space is $n = 5$. This means that in addition to the measured variable, it is also necessary to recover 4 hidden variables. Therefore, in the following we will demonstrate the modeling procedure only for a $5D$ system.

In addition, we will assume that the non-diagonal elements of matrix $A = \{a_{ij}\}; i, j = 1, ..., n; i \neq j$, form an antisymmetric matrix, and the elements of matrices $B_1, ..., B_n$ satisfy the condition $\mathbf{x}^T(B(\mathbf{x}) - B^T(\mathbf{x}))\mathbf{x} \equiv 0$ (see Theorem 2.2). (Transferring the algorithm to an arbitrary n is not difficult.)

3.2. Algorithm for quadratic model

In order for system (3.1) to be stable, we introduce into it the diffusion parameter $\mu > 0$ [10, 13]. Then, we have

$$\dot{\mathbf{x}}(t) = T \cdot (\mathbf{a}_0 + (A - \mu I) \cdot \mathbf{x} + K(\mathbf{x}) \cdot \mathbf{x}), \quad (3.3)$$

where $\mathbf{x} = (x_1, ..., x_n)^T$,

$$K(\mathbf{x}) = \begin{pmatrix} k_{11}(\mathbf{x}) & b_{12}x_2 & b_{13}x_3 & \cdots & b_{1n}x_n \\ b_{21}x_1 & k_{22}(\mathbf{x}) & b_{23}x_3 & \cdots & b_{2n}x_n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{n1}x_1 & b_{n2}x_2 & b_{n3}x_3 & \cdots & k_{nn}(\mathbf{x}) \end{pmatrix},$$

$$k_{ii}(\mathbf{x}) = -\sum_{j=1, j\neq i}^{n} (b_{ij}x_j); i = 1, ..., n.$$

This system is a complicated version of system

$$\dot{\mathbf{x}}(t) = \sigma[T \cdot ((\mathbf{W} - \mu I)\mathbf{x}(t) + \mathbf{V}\mathbf{u}(t) + \mathbf{b})], \tag{3.4}$$

where $\mathbf{x}$ is the hidden state, $\mathbf{u}$ is the input, and σ is the activation function. (Thus, we can assume that system (3.3) is system (3.4) closed by nonlinear state feedback $\mathbf{u} = K(\mathbf{x})\mathbf{x}$, where V is the identity matrix.)

In the case $n = 5$, system (3.3) takes the following form:

$$\begin{cases} \dot{x}(t) = \dfrac{a_{10} + (a_{11} - \mu)x + a_{12}y + a_{13}z + a_{14}u + a_{15}v}{1 - \vartheta \cdot \cos(x)} \\ \qquad + \dfrac{b_{22}y^2 + b_{33}z^2 + b_{44}u^2 + b_{55}v^2 - c_{11}yx - d_{11}zx - e_{11}ux - f_{11}vx}{1 - \vartheta \cdot \cos(x)}, \\ \dot{y}(t) = \dfrac{a_{20} - a_{12}x + (a_{22} - \mu)y + a_{23}z + a_{24}u + a_{25}v}{1 - \vartheta \cdot \cos(y)} \\ \qquad + \dfrac{c_{11}x^2 + c_{33}z^2 + c_{44}u^2 + c_{55}v^2 - b_{22}xy - d_{22}zy - e_{22}uy - f_{22}vy}{1 - \vartheta \cdot \cos(y)}, \\ \dot{z}(t) = \dfrac{a_{30} - a_{13}x - a_{23}y + (a_{33} - \mu)z + a_{34}u + a_{35}v}{1 - \vartheta \cdot \cos(z)} \\ \qquad + \dfrac{d_{11}x^2 + d_{22}y^2 + d_{44}u^2 + d_{55}v^2 - b_{33}xz - c_{33}yz - e_{33}uz - f_{33}vz}{1 - \vartheta \cdot \cos(z)}, \\ \dot{u}(t) = \dfrac{a_{40} - a_{14}x - a_{24}y - a_{34}z + (a_{44} - \mu)u + a_{45}v}{1 - \vartheta \cdot \cos(u)} \\ \qquad + \dfrac{e_{11}x^2 + e_{22}y^2 + e_{33}z^2 + e_{55}v^2 - b_{44}xu - c_{44}yu - d_{44}zu - f_{44}vu}{1 - \vartheta \cdot \cos(u)}, \\ \dot{v}(t) = \dfrac{a_{50} - a_{15}x - a_{25}y - a_{35}z - a_{45}u + (a_{55} - \mu)v}{1 - \vartheta \cdot \cos(v)} \\ \qquad + \dfrac{f_{11}x^2 + f_{22}y^2 + f_{33}z^2 + f_{44}u^2 - b_{55}xv - c_{55}yv - d_{55}zv - e_{55}uv}{1 - \vartheta \cdot \cos(v)}, \end{cases} \tag{3.5}$$

where the parameter $\vartheta (0 \leq |\vartheta| < 1)$ is assigned. (In total in system (3.5) we have $5 + 15 + 20 = 40$ unknown parameters.)

To find the coefficients of system (3.5), the following algorithm is proposed.

1. Fix parameters ϑ and μ that exclude the appearance of singularities in the iterative process. Let, for example, be $\vartheta = 0.95, \nu = 0.1$. Additionally, we choose the diffusion parameter $\mu = 0.00$.

2. Based on the known time series $\mathbf{x}(t) = \{x_0, x_1, ..., x_N\}$, determine the dimension of the embedding space n and the delay time τ.

3. Based on the known n (here $n = 5$) and τ (here $\tau = 1$) , construct five time series

$$\mathbf{x}(t) = (x_0, x_1, x_2, ..., x_L)^T, \mathbf{y}(t) = \mathbf{x}(t+\tau) = (y_0, y_1, y_2, ..., y_L)^T$$
$$\mathbf{z}(t) = \mathbf{x}(t+2\tau) = (z_0, z_1, z_2, ..., z_L)^T, \mathbf{u}(t) = \mathbf{x}(t+3\tau) = (u_0, u_1, u_2, ..., u_L)^T$$
$$\mathbf{v}(t) = \mathbf{x}(t+4\tau) = (v_0, v_1, v_2, ..., v_L)^T$$

   that are given on the same time interval $T_L \leq t_0 + (n-1)\tau \leq T$ in equally spaced $L \leq N$ nodes: $0, \Delta t, ..., k\Delta t, ..., L\Delta t = T_L \leq T$. Thus, $\Delta t = T_L/L$.

4. Fix a learning selections

$$x_0, x_1, ..., x_L; y_0, y_1, ..., y_L; z_0, z_1, ..., z_L; u_0, u_1, ..., u_L; v_0, v_1, ..., v_L,$$

   where $L \geq 40$.

5. Construct the columns of numerical derivatives D_x, D_y, D_z, D_u, D_v, where

$$D_x = \begin{pmatrix} D_{x1} \\ \vdots \\ D_{x_L} \end{pmatrix} = \frac{1}{\Delta t} \begin{pmatrix} x_1 - x_0 \\ \vdots \\ x_L - x_{L-1} \end{pmatrix} \in \mathbb{R}^L,$$

$$..., D_v = \begin{pmatrix} D_{v1} \\ \vdots \\ D_{v_L} \end{pmatrix} = \frac{1}{\Delta t} \begin{pmatrix} v_1 - v_0 \\ \vdots \\ v_L - v_{L-1} \end{pmatrix} \in \mathbb{R}^L,$$

$$\mathbf{D} = \begin{pmatrix} D_x \\ \vdots \\ D_v \end{pmatrix} \in \mathbb{R}^{5L}.$$

6. Calculate the disturbances introduced by the diffusion parameter

$$R_x = \begin{pmatrix} \dfrac{x_0}{1 - \vartheta \cdot \cos(x_0)} \\ \vdots \\ \dfrac{x_{L-1}}{1 - \vartheta \cdot \cos(x_{L-1})} \end{pmatrix} \in \mathbb{R}^L, ..., R_v = \begin{pmatrix} \dfrac{v_0}{1 - \vartheta \cdot \cos(v_0)} \\ \vdots \\ \dfrac{v_{L-1}}{1 - \vartheta \cdot \cos(v_{L-1})} \end{pmatrix} \in \mathbb{R}^L,$$

$$\mathbf{R} = \begin{pmatrix} R_x \\ \vdots \\ R_v \end{pmatrix} \in \mathbb{R}^{5L}.$$

7. Introduce the designations:

$$\mathbf{0} = (0, ..., 0)^T, \mathbf{1} = (1, ..., 1)^T \in \mathbb{R}^L, E_L \in \mathbb{R}^{L\times L} \text{ is the identity matrix,}$$

$$\mathbf{x} = (x_0, ..., x_{L-1})^T \in \mathbb{R}^L, ..., \mathbf{v} = (v_0, ..., v_{L-1})^T \in \mathbb{R}^L;$$

$$\mathbf{x}\odot\mathbf{x} = (x_0^2, ..., x_{L-1}^2)^T, \mathbf{y}\odot\mathbf{y} = (y_0^2, ..., y_{L-1}^2)^T, ...,$$

$$\mathbf{u}\odot\mathbf{u} = (u_0^2, ..., u_{L-1}^2)^T, \mathbf{v}\odot\mathbf{v} = (v_0^2, ..., v_{L-1}^2)^T,$$

$$\mathbf{x}\odot\mathbf{y} = (x_0y_0, ..., x_{L-1}y_{L-1})^T, ..., \mathbf{x}\odot\mathbf{v} = (x_0v_0, ..., x_{L-1}v_{L-1})^T, ...$$

$$\mathbf{v}\odot\mathbf{x} = (v_0x_0, ..., v_{L-1}x_{L-1})^T, ..., \mathbf{v}\odot\mathbf{u} = (v_0u_0, ..., v_{L-1}u_{L-1})^T;$$

$$\cos(\mathbf{x}) = \mathrm{diag}(\cos(x_0), ..., \cos(x_{L-1})),$$

$$..., \cos(\mathbf{v}) = \mathrm{diag}(\cos(v_0), ..., \cos(v_{L-1})),$$

$$T_1 = \mathrm{diag}(E_L - \vartheta\cdot\cos(\mathbf{x}))^{-1}, ..., T_5 = \mathrm{diag}(E_L - \vartheta\cdot\cos(\mathbf{v}))^{-1} \in \mathbb{R}^{L\times L},$$

$$T = \begin{pmatrix} T_1 & \dots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \dots & T_5 \end{pmatrix} \in \mathbb{R}^{5L\times 5L}.$$

$$J_1 = \begin{pmatrix} \mathbf{1} & \mathbf{x} & \mathbf{y} & \mathbf{z} & \mathbf{u} & \mathbf{v} \\ \mathbf{0} & \mathbf{0} & -\mathbf{x} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{x} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{x} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{x} \end{pmatrix} \in \mathbb{R}^{5L\times 6},$$

$$J_2 = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{y} & \mathbf{z} & \mathbf{u} & \mathbf{v} \\ \mathbf{0} & \mathbf{0} & -\mathbf{y} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{y} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{y} \end{pmatrix} \in \mathbb{R}^{5L\times 5},$$

$$J_3 = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{z} & \mathbf{u} & \mathbf{v} \\ \mathbf{0} & \mathbf{0} & -\mathbf{z} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{z} \end{pmatrix} \in \mathbb{R}^{5L\times 4},$$

$$J_4 = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{u} & \mathbf{v} \\ \mathbf{0} & \mathbf{0} & -\mathbf{u} \end{pmatrix} \in \mathbb{R}^{5L\times 3}, J_5 = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{v} \end{pmatrix} \in \mathbb{R}^{5L\times 2},$$

$$J_6 = \begin{pmatrix} \mathbf{y}\odot\mathbf{y} & \mathbf{z}\odot\mathbf{z} & \mathbf{u}\odot\mathbf{u} & \mathbf{v}\odot\mathbf{v} \\ -\mathbf{x}\odot\mathbf{y} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{x}\odot\mathbf{z} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{x}\odot\mathbf{u} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{x}\odot\mathbf{v} \end{pmatrix} \in \mathbb{R}^{5L\times 4}.$$

$$J_7 = \begin{pmatrix} -\mathbf{y}\odot\mathbf{x} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{x}\odot\mathbf{x} & \mathbf{z}\odot\mathbf{z} & \mathbf{u}\odot\mathbf{u} & \mathbf{v}\odot\mathbf{v} \\ \mathbf{0} & -\mathbf{y}\odot\mathbf{z} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{y}\odot\mathbf{u} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{y}\odot\mathbf{v} \end{pmatrix} \in \mathbb{R}^{5L\times 4}.$$

$$J_8 = \begin{pmatrix} -\mathbf{z}\odot\mathbf{x} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{z}\odot\mathbf{y} & \mathbf{0} & \mathbf{0} \\ \mathbf{x}\odot\mathbf{x} & \mathbf{y}\odot\mathbf{y} & \mathbf{u}\odot\mathbf{u} & \mathbf{v}\odot\mathbf{v} \\ \mathbf{0} & \mathbf{0} & -\mathbf{z}\odot\mathbf{u} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{z}\odot\mathbf{v} \end{pmatrix} \in \mathbb{R}^{5L\times 4}.$$

$$J_9 = \begin{pmatrix} -\mathbf{u}\odot\mathbf{x} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{u}\odot\mathbf{y} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{u}\odot\mathbf{z} & \mathbf{0} \\ \mathbf{x}\odot\mathbf{x} & \mathbf{y}\odot\mathbf{y} & \mathbf{z}\odot\mathbf{z} & \mathbf{v}\odot\mathbf{v} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{u}\odot\mathbf{v} \end{pmatrix} \in \mathbb{R}^{5L\times 4}.$$

$$J_{10} = \begin{pmatrix} -\mathbf{v}\odot\mathbf{x} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{v}\odot\mathbf{y} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{v}\odot\mathbf{z} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{v}\odot\mathbf{u} \\ \mathbf{x}\odot\mathbf{x} & \mathbf{y}\odot\mathbf{y} & \mathbf{z}\odot\mathbf{z} & \mathbf{u}\odot\mathbf{u} \end{pmatrix} \in \mathbb{R}^{5L\times 4}.$$

8. Construct Jacobi matrix:

$$\mathbf{H} = T\cdot(J_1, J_6, J_2, J_7, J_3, J_8, J_4, J_9, J_5, J_{10}) \in \mathbb{R}^{5L\times 40}.$$

9. Using the least squares method (see [12]), compute the vector: :

$$\mathbf{P} := (\mathbf{H}^T\cdot\mathbf{H} + \nu I)^{-1}\cdot\mathbf{H}^T\cdot(\mathbf{D} + \mu\mathbf{R}) =$$

$$(a_{10}, a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, b_{22}, b_{33}, b_{44}, b_{55}, a_{20}, a_{22}, a_{23}, a_{24}, a_{25},$$

$$c_{11}, c_{33}, c_{44}, c_{55}, a_{30}, a_{33}, a_{34}, a_{35}, d_{11}, d_{22}, d_{44}, d_{55},$$

$$a_{40}, a_{44}, a_{45}, e_{11}, e_{22}, e_{33}, e_{55}, a_{50}, a_{55}, f_{11}, f_{22}, f_{33}, f_{44})^T \in \mathbb{R}^{40}.$$

(Here $I \in \mathbb{R}^{40\times 40}$ is the identity matrix.)

10. Solve system (3.5) the weight coefficients of which are the coordinates of vector $\mathbf{P}$. If the solution of system (3.5) is unstable, then increase the parameter μ (for example, $\mu = 0.01$) and go to step 9, and repeat the algorithm. (After a few iterations, the solution to system (3.5) will become bounded.)

    If the use of diffusion parameter μ is undesirable ($\mu = 0$), then in system (3.5) should be assigned $a_{10} = ... = a_{50} = a_{11} = ... = a_{55} = 0$. In this case, the vector $\mathbf{P} \in \mathbb{R}^{30}$ and for any of its coordinates the system (3.5) has a bounded solution. (In step 9 of the algorithm, we have $\mathbf{P} \in \mathbb{R}^{30}$ and $\mu = 0$.)

3.3. Algorithm for linear model

In this case, items 1 – 6 are the same as in the quadratic model.

In item 7 matrices $J_6, J_7, J_8, J_9, J_{10}$ are not calculated.

Items 8, 9, and 10 are rewritten as follows:

8. Construct Jacobi matrix:

$$\mathbf{H} = T \cdot (J_1, J_2, J_3, J_4, J_5) \in \mathbb{R}^{5L \times 20}.$$

9. Compute the vector:

$$\mathbf{P} := (\mathbf{H}^T \cdot \mathbf{H} + \nu I)^{-1} \cdot \mathbf{H}^T \cdot (\mathbf{D} + \mu \mathbf{R}) =$$

$$(a_{10}, a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, a_{20}, a_{22}, a_{23}, a_{24}, a_{25}, a_{30}, a_{33}, a_{34}, a_{35},$$
$$a_{40}, a_{44}, a_{45}, a_{50}, a_{55})^T \in \mathbb{R}^{20}.$$

(Here $I \in \mathbb{R}^{20 \times 20}$ is the identity matrix.)

10. Solve system (3.5) the weight coefficients of which are the coordinates of vector $\mathbf{P}$ and all twenty coefficients $b_{22}, ..., e_{55}$ at nonlinear terms are equal to zero. In the future, the actions of item 10 of the quadratic model algorithm are repeated. (If $\mu = 0$, then in the linear model we should put $a_{10} = ... = a_{50} = a_{11} = ... = a_{55} = 0$ and $\mathbf{P} \in \mathbb{R}^{10}$. In addition, in the Jacobian matrix $\mathbf{H}$ block J_5 is absent and each of blocks $J_1 - J_4$ has 2 columns less. In this case, we have $\mathbf{H} \in \mathbb{R}^{5L \times 10}$.)

The final step is to estimate the parameters of vector $\mathbf{P}$, which will be used for further calculations or predictions based on the input time series. This algorithm allows you to adjust model parameters based on input data and improve their suitability for analysis or prediction.

3.4. Second stage: using the LSTM method

The function $\sigma(x)$ (hyperbolic tangent) satisfies the inequality $\forall x \in \mathbb{R}\ 0 \le |\sigma(x)| < 1$. Therefore, from the boundedness of solutions of system (3.3) with initial conditions $\mathbf{x}_0$ it follows the boundedness of solutions of system (3.4) with initial conditions $\sigma(\mathbf{x}_0)$ [14]. Consequently, the weight matrices of system (3.3) can be taken as the initial weight matrices W and V for system (3.4).

In order to use the LSTM method it is necessary to insert under the sign σ in the system (3.4) equations (3.5) with known coefficients $a_{10}, ..., f_{44}$ (for the quadratic model) or $a_{10}, ..., a_{55}$ (for the linear model) as initial data.

On the basis of the parameter vector $\mathbf{P}$, the antisymmetric matrix W and the rectangular matrix V are formed. They represent the connections between the input and hidden layers of neurons and are key components of the LSTM structure.

The obtained matrices W and V are transformed into weight matrices of the LSTM model (input weight) taking into account the architectural features of the LSTM and their dimensions. On the basis of the weight matrices, the architecture of the LSTM neural network is created in Matlab, including the definition of the number of layers, the number of neurons in each layer, activation functions and other parameters that determine the behavior of the network.

We have

$$W = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ -a_{12} & a_{22} & a_{23} & a_{24} & a_{25} \\ -a_{13} & -a_{23} & a_{33} & a_{34} & a_{35} \\ -a_{14} & -a_{24} & -a_{34} & a_{44} & a_{45} \\ -a_{15} & -a_{25} & -a_{35} & -a_{45} & a_{55} \end{pmatrix} \in \mathbb{R}^{5\times 5}, \mathbf{b} = \begin{pmatrix} a_{10} \\ a_{20} \\ a_{30} \\ a_{40} \\ a_{50} \end{pmatrix} \in \mathbb{R}^5$$

$$\mathbf{u} = (x^2, y^2, z^2, u^2, v^2, xy, xz, xu, xv, yz, yu, yv, zu, zv, uv)^T \in \mathbb{R}^{15}$$

$$V = \left(\begin{array}{ccccc|} 0 & b_{22} & b_{33} & b_{44} & b_{55} \\ c_{11} & 0 & c_{33} & c_{44} & c_{55} \\ d_{11} & d_{22} & 0 & d_{44} & d_{55} \\ e_{11} & e_{22} & e_{33} & 0 & e_{55} \\ f_{11} & f_{22} & f_{33} & f_{44} & 0 \end{array} \right. \rightarrow$$

$$\left| \begin{array}{cccccccccc} -c_{11} & -d_{11} & -e_{11} & -f_{11} & 0 & 0 & 0 & 0 & 0 & 0 \\ -b_{22} & 0 & 0 & 0 & -d_{22} & -e_{22} & -f_{22} & 0 & 0 & 0 \\ 0 & -b_{33} & 0 & 0 & -c_{33} & 0 & 0 & -e_{33} & -f_{33} & 0 \\ 0 & 0 & -b_{44} & 0 & 0 & -c_{44} & 0 & -d_{44} & 0 & -f_{44} \\ 0 & 0 & 0 & -b_{55} & 0 & 0 & -c_{55} & 0 & -d_{55} & -e_{55} \end{array} \right) \in \mathbb{R}^{5\times 15}.$$

(For linear model $V = 0$!)

After initializing the weight matrices and building the LSTM network, you can start training the network on the input data or use it for various tasks such as time series prediction or data analysis.

Further, in this study, an algorithm for determining and forming weighting coefficients, as well as a neural network for achieving forecasting goals, was developed and implemented. The development was carried out in the MATLAB 2020a environment. Below is a block diagram (see Fig.3.1), which shows the main process of the developed algorithm and neural network.

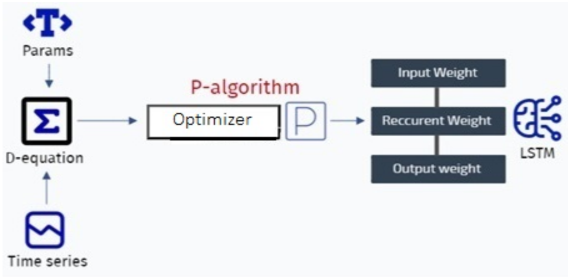


Fig. 3.1. Block diagram of the general EEG data processing algorithm (see ([15])

In particular, attention was focused on the development of a neural network using the Long Short Time Memory (LSTM) layer. LSTM is a powerful tool for processing serial data, and it shows the most accurate forecasting results after proper training of input parameters because it has the ability to consider and analyze long-term dependencies in serial data. The LSTM layer is able to store and use information from previous time steps, allowing the neural network to effectively model and predict complex sequences.

LSTMs are a type of recurrent neural networks (RNNs) designed to model sequential data. This architecture was specifically designed to solve the gradient vanishing problem that often occurs in conventional RNNs. The main characteristics of LSTMs are the ability to store and use information from previous time steps, supervised forgetting, and the assignment of weights to control the flow of information.

LSTM consists of the following main components:

1. Cell state (Cell State) is the main memory of LSTM. It allows a neural network to store and transfer information over many time steps. This memory is controlled by interface weights that determine which information should be forgotten or retained.

2. Input layer (Input Gate) - this input decides what information should be updated in the cellular state. It is activated by a weighted multiplication of the input data and the previous state.

3. Output layer (Output Gate) - determines what information should be output from the cellular state. It is also governed by weighting factors and the internal state of the model.

4. Forgetting layer (Forget Gate) - allows LSTM to decide what information should be forgotten from the cell state based on the current input data and the previous state.

5. Internal weights (Internal Weights) - internal weight coefficients that allow the model to interact and calculate the new state of the cell based on the input data and the previous state.

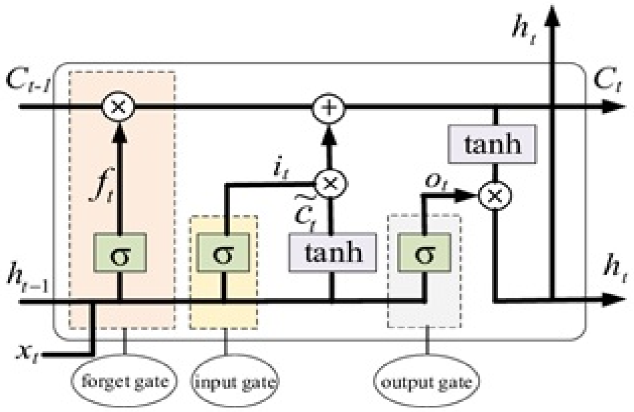


Fig. 3.2. Block diagram of the LSTM layer (see [1, 2])

The main idea behind LSTM is that it can effectively handle and model long-term dependencies in sequential data due to its ability to control the flow of internal cell information. This architecture has found wide application in areas where it is important to model complex sequences, such as language analysis, machine learning, and many other areas (see Fig.3.2).

4. Real applications and numerical analysis of the obtained results

The above algorithm generally describes the steps involved in processing an input time series and is designed to help researchers and practitioners analyze and model complex systems using time series data. In particular, the algorithm provides step-by-step instructions for finding coefficients in a special system, and the system itself is a set of differential equations that can be used to model a wide range of physical, biological, and social phenomena.

Solving the corresponding systems allows you to find the values of unknown parameters that accurately describe the dynamics of the modeled system. Once the coefficients are found, they can be used to predict the behavior of the system over time. For example, the coefficients found in our study can be set by the input weights of the neural network for predicting EEG behavior. Similarly, if the system is a model of a biological process such as the spread of a disease, unraveling the system can help predict the future number of infected individuals given the current state of the population. In general, solving a system allows you to gain insight into the underlying dynamics of complex systems and make predictions about their behavior, which can be useful in many industries.

The initial stages of the described algorithm determine significantly influential parameters of the system state, such as singularities, errors, nesting dimensions, and delays. Based on the known time series $x_0, x_1, ..., x_N$, the dimension of the embedding space and the delay time are determined. This can be done using the delay method, which involves constructing a set of time-delayed copies of the original time series and using them to reconstruct the underlying attractor that defines any chaotic system [17, 18].

In general, in the work, a user interface was implemented using MATLAB2020a tools, which provides a functional opportunity to process the input time step with three algorithms of a similar nature (as described above), but with different system parameters and, accordingly, their essential structured difference. Below is a comparative result of the work of each algorithm, which ended with the original parameter matrix and the solution of the simulated system.

4.1. Modeling epilepsy based on EEG data

Consider the implementation of the above algorithm on real data, taking into account the primary processing of the signal from the encephalograph cap by the primary noise filter [19]. For this, software tools were used for automatic data initialization in the system of multiple space, which means a time series is formed from each electrode of the EEG cap. The input series is divided into a series of trajectories with a predetermined displacement ($\tau = 10$), which was indicated above (Fig.4.1,4.2).

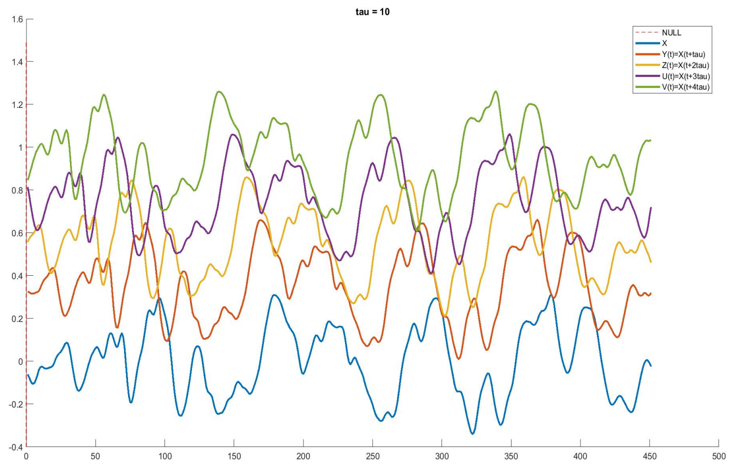


Fig. 4.1. EEG time series of a sick patient with shift $\tau = 10$

We will present the reconstruction (Fig.4.3,4.4) of both sequences and run the algorithm for their processing to obtain the parameters of the system that models the specified sequences with the defined control of the process of propagation of trajectories.

The methodology, which is developed on the basis of MATLAB2020a tools, was tested on two patients with pre-processing of the data to determine the weights of the neural network through the EEG behavior modeling algorithm and the singularities that were added to this process. Note that the developed

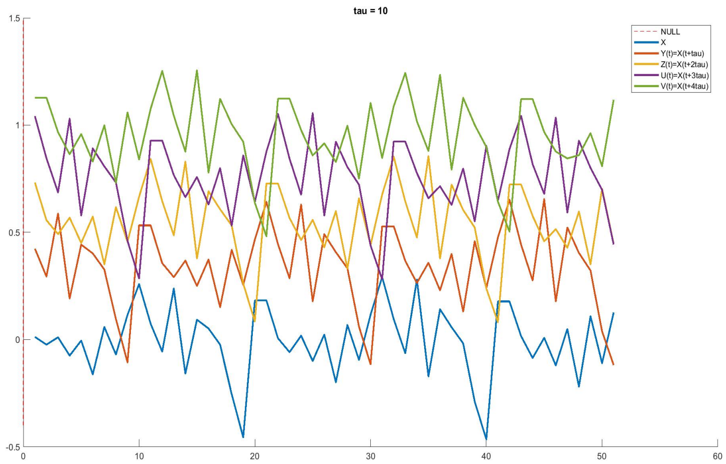


Fig. 4.2. EEG time series of a healthy patient with shift $\tau = 10$

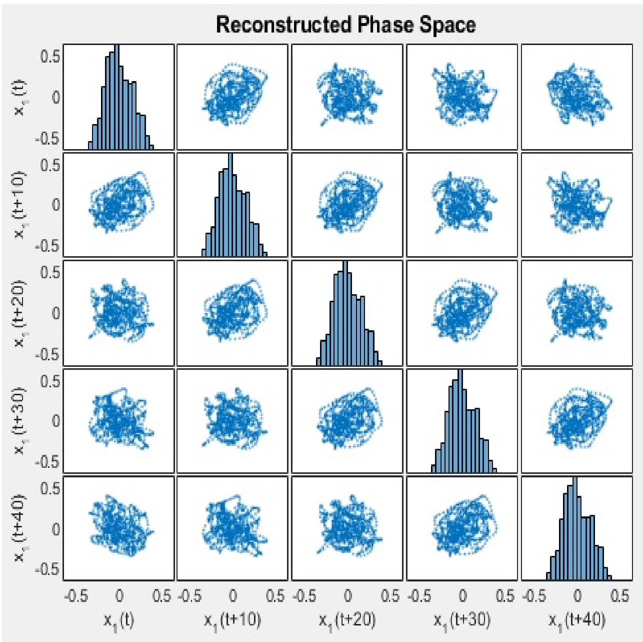


Fig. 4.3. Phase space of a sick patient

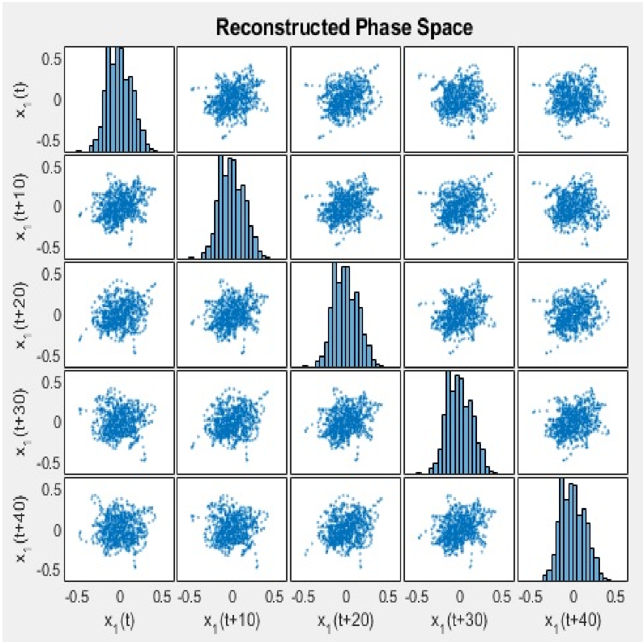
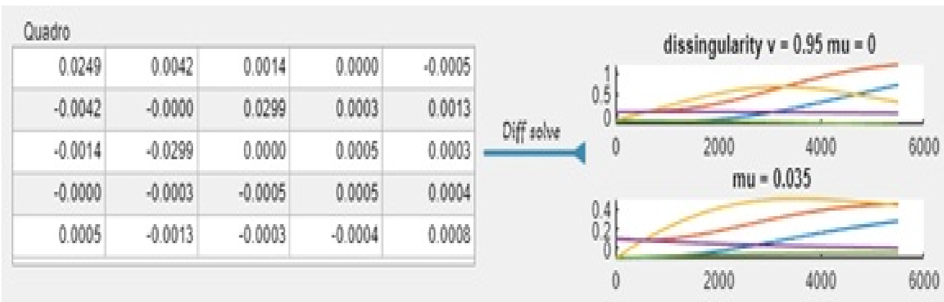


Fig. 4.4. Phase space of a healthy patient

algorithms have a characteristic difference in the presence of quadratic elements and the diffusion parameter. Therefore, three different cases were analyzed: a quadratic algorithm, a quadratic algorithm with a diffusion parameter, and a linear algorithm.

A neural network with defined weights is evaluated using the mean square error (MSE) method. The results of forecasting in relation to real data and the forecast of unknown values for the future 30 steps are displayed graphically (Fig.4.5 -4.7). This allows us to evaluate the effectiveness and accuracy of the model in forecasting based on the training data provided for training the LSTM neural network.



0.0249	0.0042	0.0014	0.0000	-0.0005
-0.0042	-0.0000	0.0299	0.0003	0.0013
-0.0014	-0.0299	0.0000	0.0005	0.0003
-0.0000	-0.0003	-0.0005	0.0005	0.0004
0.0005	-0.0013	-0.0003	-0.0004	0.0008

Fig. 4.5. The weight matrix of the quadratic algorithm with the diffusion parameter ($\mu = 0, \mu = 0.035$) of a sick patient with the result of data modeling; $MSE = 0.0041193$

Quadro without J5

0	0.0009	-0.0007	0.0001	0.0003
-0.0009	0	0.0019	-0.0008	-0.0000
0.0007	-0.0019	0	0.0016	-0.0003
-0.0001	0.0008	-0.0016	0	0.0006
-0.0003	0.0000	0.0003	0.0377	0

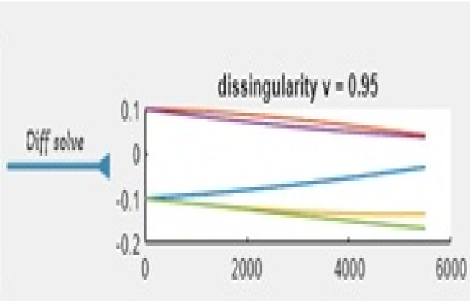


Fig. 4.6. The weight matrix of the quadratic algorithm of the sick patient with the result of data modeling; $MSE = 0.00511702$

Linea

-5.1346	4.1701	1.3474	0.0526	-0.4772
-4.1701	3.1699	0.2625	0.5892	0.0577
-1.3474	-0.2625	-0.1051	0.2941	1.3425
-0.0526	-0.5892	-0.2941	-3.2856	4.1560
0.4772	-0.0577	-1.3425	-4.1560	4.9664

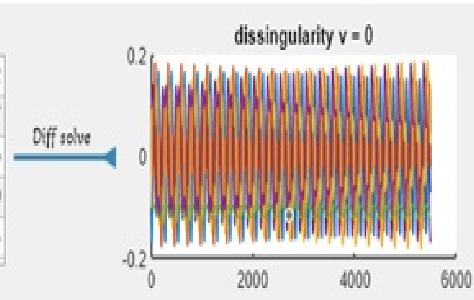
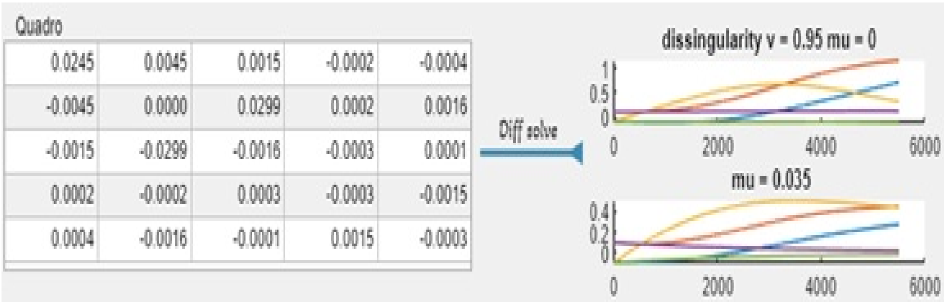


Fig. 4.7. The weight matrix of the patient-to-patient linear algorithm with the result of data modeling; $MSE = 0.0030853$

We note the appearance of nonlinear effects in modeling by the quadratic algorithm with diffusion, which provides the possibility of further adjustment of the system to obtain a more similar solution. At the same time, we note the effective operation of the linear algorithm, which repeats the input trajectory, but with falling into a periodic process, which distinguishes the simulation result from the real scenario. According to the values of neural network training errors, we observe the smallest deviations precisely in the linear algorithm, and the worst is the quadratic algorithm without the diffusion parameter.

Let's try to repeat these actions based on the data of a healthy patient (Fig.4.8 - 4.10).



0.0245	0.0045	0.0015	-0.0002	-0.0004
-0.0045	0.0000	0.0299	0.0002	0.0016
-0.0015	-0.0299	-0.0016	-0.0003	0.0001
0.0002	-0.0002	0.0003	-0.0003	-0.0015
0.0004	-0.0016	-0.0001	0.0015	-0.0003

Fig. 4.8. The weight matrix of the quadratic algorithm with the diffusion parameter ($\mu = 0, \mu = 0.035$) for a healthy patient with the result of data modeling; $MSE = 0.017608$

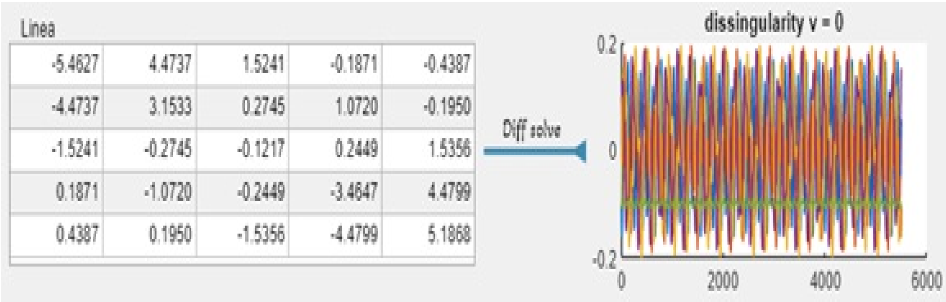
0	0.0011	-0.0001	0.0002	-0.0001
-0.0011	0	0.0018	-0.0003	0.0001
0.0001	-0.0018	0	0.0019	-0.0002
-0.0002	0.0003	-0.0019	0	0.0012
0.0001	-0.0001	0.0002	-0.0056	0

Fig. 4.9. The weight matrix of the quadratic algorithm for a healthy patient with the result of data modeling; $MSE = 0.017553$

These results demonstrate a significant difference in simulation results compared to sick patients, with an order of magnitude higher error. Such results demonstrate differences in input amplitudes and embedding dimensions that directly affect the training outcome and allow classification of healthy patients with more chaotic behavior of brain signals.

The next step is to present the results of the neural network and compare the prediction results.

In all the following figures, the blue line is the line obtained from the results of


Linea

-5.4627	4.4737	1.5241	-0.1871	-0.4387
-4.4737	3.1533	0.2745	1.0720	-0.1950
-1.5241	-0.2745	-0.1217	0.2449	1.5356
0.1871	-1.0720	-0.2449	-3.4647	4.4799
0.4387	0.1950	-1.5356	-4.4799	5.1868

Fig. 4.10. The weight matrix of the linear algorithm for a healthy patient with the result of data modeling; $MSE = 0.018206$

EEG measurements, the red line is obtained as a result of adjusting the weighting coefficients of the neural network model, and the yellow line is the prediction line.

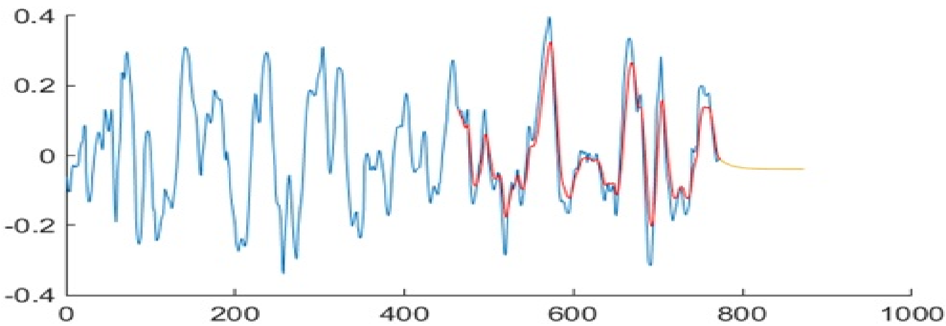


Fig. 4.11. The result of LSTM prediction of a patient's neural network with weighting coefficients of the quadratic algorithm and a diffusion parameter

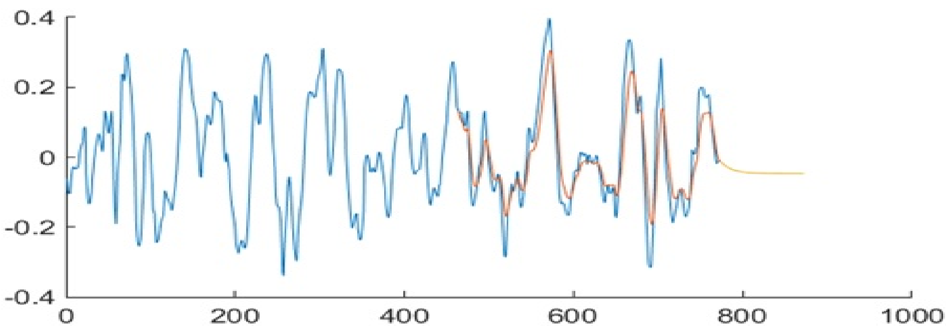


Fig. 4.12. The result of LSTM prediction of a patient's neural network with weighting coefficients of the quadratic algorithm

These results allow us to draw conclusions about the sufficiently effective result of neural network training using pre-processing algorithms and to determine a more optimal approach to the classification of EEG data. However, further use of

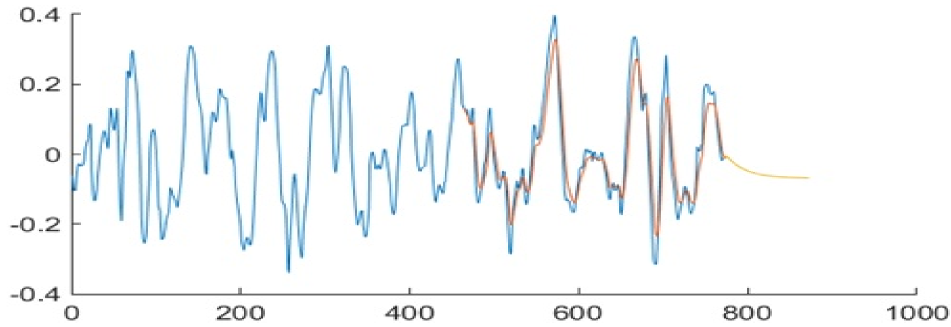


Fig. 4.13. The result of LSTM prediction of the patient's neural network with the weighting coefficients of the linear algorithm

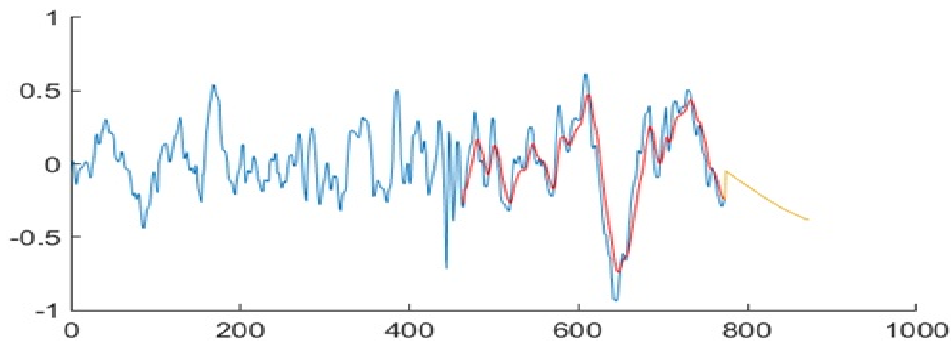


Fig. 4.14. Prediction result of LSTM neural network of a healthy patient with quadratic algorithm weights and diffusion parameter

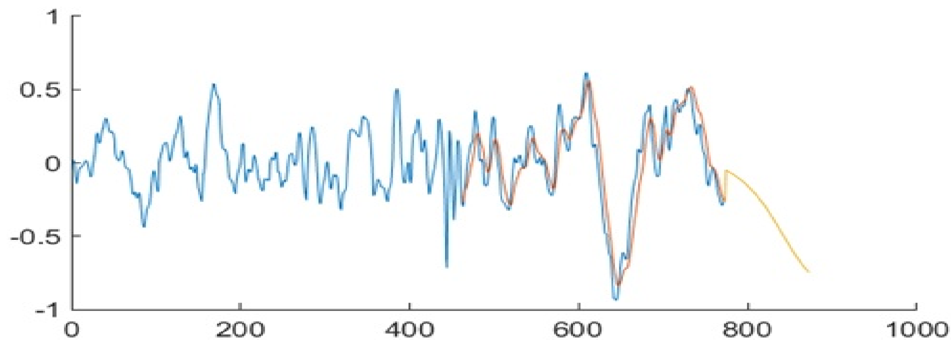


Fig. 4.15. LSTM neural network prediction result for a healthy patient with quadratic algorithm weights

the neural network to predict future values is possible for very short time intervals (about 10 steps), after which either a steady-state mode of the system is observed (in other words, there is no signal), or a mode of constant monotonicity, which excludes the occurrence of further chaos (see Fig.4.11 - 4.16).

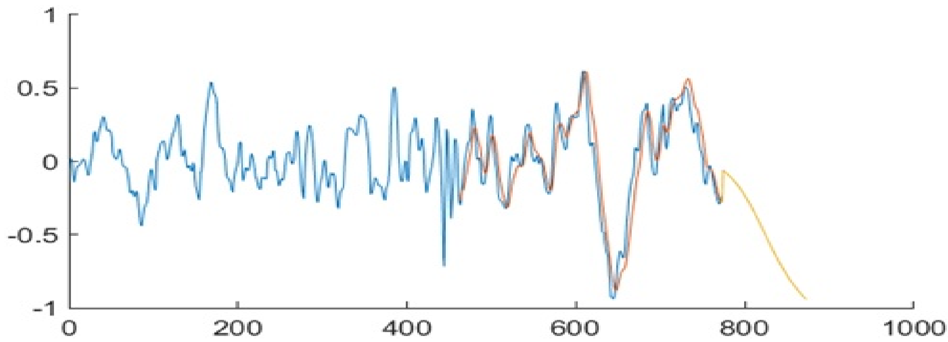


Fig. 4.16. LSTM neural network prediction result for a healthy patient with linear algorithm weights

References

1. U. R. Acharya, S. V. Sree, G. Swapna, R. J. Martis, J. S. Suri, *Automated EEG analysis of epilepsy: A review*, Knowledge-Based Systems, **45**, (2013), 147–165.
2. B. Ay, O. Yildirim, M. Talo, U. B. Baloglu, G. Aydin, S. D. Puthankattil, U. R. Acharya, *Automated depression detection using deep representation and sequence learning with EEG signals*, Journal of Medical Systems, **43**(7), (2019), 1–24.
3. V. Ye. Belozyorov, *Universal approach to the problem of emergence of chaos in autonomous dynamical systems*, Nonlinear Dynamics, **95**(1), (2019), 579–595.
4. V. Ye. Belozyorov, D. V. Dantsev, *Stability of neural ordinary differential equations with power nonlinearities*, Journal of Optimization, Differential Equations and Their Applications (JODEA), **28**(2)(2020), 21–46.
5. V. Ye. Belozyorov, D. V. Dantsev, Y. V. Koshel, *On the equivalence of real dynamic process and its models*, Journal of Optimization, Differential Equations and Their Applications (JODEA), **29**(2)(2021), 76–91.
6. V. Ye. Belozyorov, D. V. Dantsev, *Modeling of chaotic processes by means of antisymmetric neural ODEs*, Journal of Optimization, Differential Equations and Their Applications (JODEA), **30**(1)(2022), 1–41.
7. V. Ye. Belozyorov, S. A. Volkova, *Odd and even functions in the design problem of new chaotic attractors*, International Journal of Bifurcation and Chaos, **32**, (2022), Article ID 2250218, 26 pages
8. V. Ye. Belozyorov, S. A. Volkova, *Discrete processes and chaos in systems of ordinary differential equations*, Journal of Optimization, Differential Equations and Their Applications (JODEA), **30**(2)(2022), 62–78.
9. V. Ye. Belozyorov, S. A. Volkova, V. G. Zaytsev, *Singular differential equations and their applications for modeling strongly oscillating processes*, Journal of Optimization, Differential Equations and Their Applications (JODEA), **31**(1)(2023), 22–52.
10. B. Chang, M. Chen, E. Haber, E. D. Chi, *Antisymmetric RNN: A Dynamical System Viev on Recurrent Neural Networks*, In conference ICLR, May 6 - May 9, New Orleans, Louisiana, USA, (2019), 1–15.
11. R. T. Q. Chen, Y. Rubanova, J. Bettencourt, D. Duvenaud, *Neural ordinary differential equations*, arXiv preprint arXiv:1806.07366v5[cs.LG], (2019), 1–18.

12. S. Haykin, *Neural Networks. A Comprehensive Foundation*, Second Edition, Pearson Education, Prentice Hall, 2005.
13. E. Haber, L. Ruthotto, *Stable Architectures for Deep Neural Networks*, arXiv preprint arXiv: 1705.03341v1[cs.LG], (2019), 1–23.
14. H. K. Khalil, *Nonlinear Systems – 2nd Edition*, Prentice Hall, New-Jersy, 1996.
15. J. S. Kumar, P. Bhuvaneswari, *Analysis of electroencephalography (EEG) signals and its categorization*, A Study Procedia Engineering, **38**, (2012), 2525–2536.
16. Q. Li, *Dynamical Systems and Machine Learning*, Summer School, Peking University, 2020.
17. N. A. Magnitskii, *Universal theory of dynamical chaos in nonlinear dissipative systems of differential equations*, Commun. Nonlinear Sci. Numer. Simul., **13**, (2008), 416–433.
18. N. A. Magnitskii, *Universality of transition to chaos in all kinds of nonlinear differential equations*, Nonlinearity, Bifurcation and Chaos – Theory and Applications, Chapter 6. Intech, (2012), 133–174.
19. National Center for Biotechnology Information, *National Library of Medicine at the U.S. National Institutes of Health (NIH)*, USA, https://www.ncbi.nlm.nih.gov.
20. Toolboxes for Complex Systems, *Potsdam Institute for Climate Impact Research (PIK)*, Germany, http://tocsy.pik-potsdam.de.
21. C. L. Webber, N. Marwan (eds), *Reccurence Quantification Analysis. Theory and Best Practice*, Springer, NY, Dordrecht, Heidelberg, London, 2015.

Received 27.09.2023