Dynamic Algorithms of Multilayered Neural Networks Training in Generalized Training Plant

Efimov D.V., Zakharenkova T.A., Terekhov V.A., Tyukin I.Yu.
The Department of Automatics and Control Processes
St. Petersburg State Electrical Engineering University "LETI"
Phone: 234-64-35, e-mail: efde@mail.rcom.ru

Abstract - *The new modifications of multilayered neurak networks training algorithms in a generalized training plant structure are introduced. The first modification for algorithm " on error backpropagation algorithm through time " is introduced and its sufficient conditions of a training procedure stability are obtained. Other modification for speed gradient algorithm of an error backpropagation is obtained , where measurement state vector in a training procedure instead of a parametrical model of plant is used.*

1. INTRODUCTION

It is known that a connection of a neurocontroller with a control plant (CP) may be represented as generalized training plant (GTP) (fig. 1) [1,2].

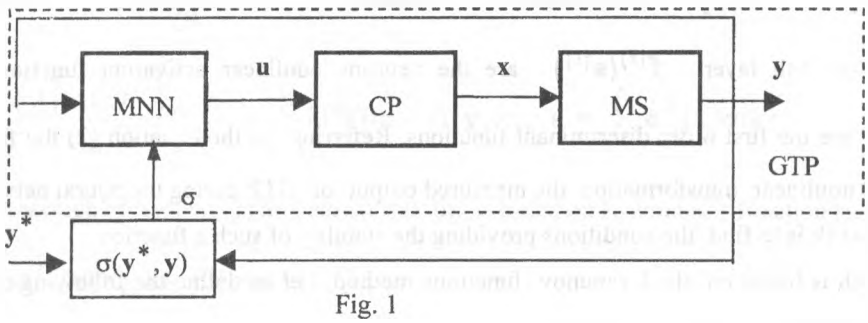

Fig. 1

There MS is a measuring system, which translates the statement vector $\mathbf{x}$ to the measured vector $\mathbf{y} = \mathbf{h}(\mathbf{x}(t))$, where $\mathbf{h}(\cdot)$ is a vector function; $\mathbf{y}^*$ is a desired function, $\boldsymbol{\sigma} = \boldsymbol{\sigma}(\mathbf{y}^*, \mathbf{y})$ is a generalized error vector, which is used for network training. The architecture of MNN is defined by $N^K_{n_0, n_1, \ldots n_K}$ symbol, where K is an amount of network layers, n_0 is a number of inputs, n_i, $i \in \overline{1, K}$ is a count of the artificial neurons or basic processor elements (BPE) in the i-th layer. The problem how to find the control function $\mathbf{u}(\mathbf{y}^*, \mathbf{y})$, which provides a minimum of function Q, is reduced in this case to the neural network training problem.

The base algorithm of multilayered neural network training is the algorithm of an error backpropagation (BP). Its direct use for synthesis of neural network training algorithms in problems of dynamic plants control by virtue of a series of the reasons is hampered [1]. However it forms the basis for synthesis of multilayered neural network training algorithms modifications in a structure of GTP.

In this paper we suggest the training algorithm for MNN within GTP (fig. 1), which is based on error backpropagation algorithm through time (BPTT) [3]. Also the algorithm resulted from back propagation error method and speed-gradient method (SG) [4] and was later called speed back propagation error algorithm (SBP) are introduced. Using that algorithm one can take into account the dynamics of the network training process and to cover a wide spectrum of problems. Unfortunately the algorithm lacks an adequate control plant model, it also takes a long time of computation and the necessity first to make the prognose of signal in the plant output. Modification of the algorithm

SBP, free from enumerated shortages is offered below.

2. BPTT ALGORITHM

We make an algorithm step estimation, which provides training process stability for functional $Q = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(k)\boldsymbol{\sigma}(k)$ at discrete-time point $k\Delta t$, $\Delta t = \text{const}$; $k = 1,2, \ldots$, and also a training error is defined by us as $\boldsymbol{\sigma}(k) = \mathbf{y}^*(k) - \mathbf{y}(k)$.

Let a plant be defined by the following equation:

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k)); \\ \mathbf{y}(k) = \mathbf{h}(\mathbf{x}(k)). \end{cases} \quad (1)$$

The control vector on fig. 1 is found by MNN during its training:

$$\mathbf{u} = \mathbf{q}^{(K)} = \mathbf{f}^{(K)}(\mathbf{W}^{(K)}\mathbf{f}^{(K-1)}(\mathbf{W}^{(K-1)}\mathbf{f}^{(K-2)}(\ldots\mathbf{W}^{(\ell)}\mathbf{f}^{(\ell-1)}(\mathbf{W}^{(\ell-1)}\mathbf{f}^{(\ell-2)} \times (\cdots\mathbf{W}^{(2)}\mathbf{f}^{(1)}(\mathbf{W}^{(1)}\mathbf{y}(k-1))\cdots)\cdots))) = \mathbf{F}(\mathbf{y}), \quad (2)$$

where $\mathbf{W}^{(\ell)} = \left[\mathbf{w}_1^{(\ell)} : \ldots : \mathbf{w}_{n_\ell}^{(\ell)}\right]^T$ is the weight-factor matrix for layer $\ell \in \overline{1,K}$, $\mathbf{w}_i^{(\ell)}$ is a weight-factor vector for the i-th BPE in the l-th layer, $\mathbf{f}^{(\ell)}(\mathbf{s}^{(\ell)})$ are the neurons nonlinear activation functions in the l-th layer, $\mathbf{s}^{(\ell)} = \mathbf{w}^{(\ell)}\mathbf{q}^{(\ell-1)}$ are the first order discriminant functions. Referring to the equation (2) the function carries out a complex dynamical nonlinear transformation the measured output of GTP during the neural network training process. The subject of this work is to find the conditions providing the stability of such a function.

The research is based on the Lyapunov functions method. Let us define the following training algorithm for MNN in a GTO structure:

$$\delta\mathbf{w}_i^{(\ell)}(n) = -\gamma \sum_{j=1}^{N} \frac{\partial \boldsymbol{\sigma}^T(j)\boldsymbol{\sigma}(j)}{\partial \mathbf{u}(1)} \frac{\partial \mathbf{u}(1)}{\partial \mathbf{w}_i^{(\ell)}}, \quad (3)$$

where γ is an algorithm step, n is a number of steps. The partial derivative from equation (3) may be calculated by following formula:

$$\frac{\partial \boldsymbol{\sigma}^T(j)\boldsymbol{\sigma}(j)}{\partial \mathbf{u}(1)} = \frac{\partial \boldsymbol{\sigma}^T(j)\boldsymbol{\sigma}(j)}{\partial \mathbf{y}(j)} \frac{\partial \mathbf{y}(j)}{\partial \mathbf{x}(j)} \left(\frac{\partial \mathbf{f}(\mathbf{x}(j-1), \mathbf{u}(j-1))}{\partial \mathbf{u}(j-1)} \frac{\partial \mathbf{u}(j-1)}{\partial \mathbf{y}(j-1)} + \frac{\partial \mathbf{f}(\mathbf{x}(j-1), \mathbf{u}(j-1))}{\partial \mathbf{x}(j-1)} \frac{\partial \mathbf{x}(j-1)}{\mathbf{y}(j-1)} \right) \cdots \frac{\partial \mathbf{x}(2)}{\partial \mathbf{u}(1)}. \quad (4)$$

The parity such as (4) is used in error backpropagation algorithm (BP). However, this "error backpropagation algorithm" differs from BP algorithm, because the described algorithm is not passed from layer to layer, it's passed from state to state, i. e. from one time moment to next. That is why this algorithm is called "error backpropagation through time".

As any recurrent algorithm the gradient procedure (3) may be stable or unstable depending on the algorithm parameters and, first of all, on the algorithm step γ, when the criterion is the function $Q = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(k)\boldsymbol{\sigma}(k)$. Besides this in practice one requires to provide the MNN training process stability only if the desired function, network architecture and the initial state are changed. In the square training criterion $Q = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(k)\boldsymbol{\sigma}(k)$ case a sufficient conditions follow from the theorem.

Theorem. *Let the functions* $\mathbf{f}, \mathbf{h} \in \mathbf{C}^1$ *in system (1) and there exist: a) partial derivatives* $\dfrac{\partial \mathbf{x}(j)}{\partial \mathbf{y}(j-1)}$

b) vector function $\delta\mathbf{y}(\mathbf{W}, k)$ *which may be represented in form (5) and also indicate an increment of the vector* $\mathbf{y}(\mathbf{W}, k)$ *followed a variation of weight-factor matrix* $\mathbf{W}$.

$$\delta\mathbf{y}(\mathbf{W}, k) = \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \frac{\partial \mathbf{y}(\mathbf{W}, k)}{\partial w_{i,j}^{(\ell)}} \delta w_{i,j}^{(\ell)} + \mathbf{e}(\delta\ \mathbf{W}, k) \tag{5}$$

Then the gradient procedure (3) convergence for square criterion Q *if the following conditions are satisfied:*

1) there exists $\lambda(\mathbf{W}, \mathbf{y}, k) > 0$ *such as*

$$\delta\mathbf{y}(\mathbf{W}, k)^T \delta\mathbf{y}(\mathbf{W}, k) \le \lambda(\mathbf{W}, \mathbf{y}, i) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \left(\delta w_{i,j}^{(\ell)}\right)^2;$$

2) $\gamma \in \left(\dfrac{1 - \sqrt{1 - 8\tilde{\lambda}\varepsilon\alpha^{-1}}}{2\tilde{\lambda}}, \dfrac{1 + \sqrt{1 - 8\tilde{\lambda}\varepsilon\alpha^{-1}}}{2\tilde{\lambda}} \right)$ *in the case* $1 - 8\tilde{\lambda}\varepsilon\alpha^{-1} \ge 0$ *and* $\gamma = 0$ *in the case* $1 - 8\tilde{\lambda}\varepsilon\alpha^{-1} < 0$,

where $\alpha = \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \left(\dfrac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right)^2$, $\tilde{\lambda} = \sum_{k=1}^{N} \lambda(\mathbf{W}(n-1), \mathbf{y}, k)$, $\varepsilon = \sum_{k=1}^{N} \mathbf{e}^T(k)\boldsymbol{\sigma}(k)$.

Proof. The research of system (3) movement stability is based on Lyapunov's function analysis:

$$V(n) = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(\mathbf{W}(n), k)\boldsymbol{\sigma}(\mathbf{W}(n), k) - Q_{\min} \tag{6}$$

where $Q_{\min}$ is minimum value of a functional Q at neural network architecture $N_{n_0, n_1, \ldots n_K}^{K}$ and activation functions $f(\cdot)$.

Let us consider an increment $\Delta V(n)$:

$$\Delta V(n) = V(n) - V(n-1) = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(\mathbf{W}(n), k)\boldsymbol{\sigma}(\mathbf{W}(n), k) - \boldsymbol{\sigma}^T(\mathbf{W}(n-1), k), \boldsymbol{\sigma}(\mathbf{W}(n-1), k). \tag{7}$$

The generalized error $\boldsymbol{\sigma}(\mathbf{W}(n), i)$ in (7) is represented in the following kind:

$$\boldsymbol{\sigma}(\mathbf{W}(n), k) = \mathbf{y}^{(*)}(k) - \mathbf{y}(n, k) = \mathbf{y}^{(*)}(k) - (\mathbf{y}(\mathbf{W}(n-1), k) + \delta\mathbf{y}(\mathbf{W}(n-1), k)), \tag{8}$$

where $\delta\mathbf{y}(\mathbf{W}(n-1), k)$ is a vector function of the $\mathbf{y}(\mathbf{W}(n-1), k)$ vector increment which is determined by the weight-factors variation. Then the Lyapunov's function (6) increment ΔV referred to (7), (8) is discovered as:

$$\begin{aligned} \Delta V(n) = \sum_{k=1}^{N} \Big((\mathbf{y}^{(*)}(k) - (\mathbf{y}(\mathbf{W}(n-1), k) + \delta\mathbf{y}(\mathbf{W}(n-1), k))^T (\mathbf{y}^{(*)}(k) - -(\mathbf{y}(\mathbf{W}(n-1), k) + \delta\mathbf{y}(\mathbf{W}(n-1), k)) \Big) - \\ - \sum_{k=1}^{N} (\mathbf{y}^{(*)}(k) - \mathbf{y}(\mathbf{W}(n-1), k))^T (\mathbf{y}^{(*)}(i) - \mathbf{y}(\mathbf{W}(n-1), k)). \end{aligned} \tag{9}$$

Simple transformations of equation (9) result its in the following form:

$$\Delta V(n) = \sum_{k=1}^{N} \delta\mathbf{y}(\mathbf{W}(n-1), k)(2(\mathbf{y}(\mathbf{W}(n-1), k)) - \mathbf{y}^{(*)}(\mathbf{W}(n-1), k)) + \delta\mathbf{y}(\mathbf{W}(n-1), k)). \tag{10}$$

It is necessary to emphasize that the increment $\delta\mathbf{y}(\mathbf{W}(n-1), k)$ in equation (10) may be represent in the kind of following sum:

$$\delta\mathbf{y}(\mathbf{W}(n-1),k) = \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}} \frac{\partial \mathbf{y}(\mathbf{W}(n-1),k)}{\partial w_{i,j}^{(\ell)}} \delta w_{i,j}^{(\ell)}(n-1) + \mathbf{e}(\delta\mathbf{W},\mathbf{y},k), \tag{11}$$

where $\delta w_{i,j}^{(\ell)} = (-\gamma(n-1))\dfrac{\partial Q}{\partial w_{i,j}^{(\ell)}}$. Then let us write down the increment ΔV of the chosen Lyapunov's function (6) refer to (10), (11):

$$\Delta V(n) = \left(\sum_{k=1}^{N} \delta\mathbf{y}(\mathbf{W}(n-1),k)^T \delta\mathbf{y}(\mathbf{W}(n-1),k) + 2\mathbf{e}(\delta\mathbf{W},\mathbf{y},k)^T \boldsymbol{\sigma}(\mathbf{W},\mathbf{y},k)\right) + \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}} \left(\delta w_{i,j}^{(\ell)}\right)^2 (-\gamma(n-1))^{-1} \le$$
$$\le \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}} \left(\delta w_{i,j}^{(\ell)}\right)^2 \left((-\gamma(n-1))^{-1} + \sum_{k=1}^{N}\lambda(\mathbf{W},\mathbf{y},k)\right) + 2\sum_{k=1}^{N}\mathbf{e}(\delta\mathbf{W},\mathbf{y},k)^T\boldsymbol{\sigma}. \tag{12}$$

Let $\alpha = \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\left(\dfrac{\partial Q}{\partial w_{i,j}^{(\ell)}}\right)^2 = \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\left(\delta w_{i,j}^{(\ell)}\right)^2(\gamma(n-1))^{-2}$, $\tilde{\lambda} = \sum_{k=1}^{N}\lambda(\mathbf{W}(n-1),\mathbf{y},k)$, $\varepsilon = \sum_{k=1}^{N}\mathbf{e}^T(k)\boldsymbol{\sigma}(k)$.

Then the equation (12) can be rewritten as $(-\gamma)^2\tilde{\lambda}\alpha - \gamma\alpha + 2\varepsilon \le 0$. If γ is expressed we receive that:

$\gamma \in \left(\dfrac{1-\sqrt{1-8\tilde{\lambda}\varepsilon\alpha^{-1}}}{2\tilde{\lambda}}, \dfrac{1+\sqrt{1-8\tilde{\lambda}\varepsilon\alpha^{-1}}}{2\tilde{\lambda}}\right)$ in the case $1-8\tilde{\lambda}\varepsilon\alpha^{-1} \ge 0$ and $\gamma = 0$ in the case $1-8\tilde{\lambda}\varepsilon\alpha^{-1} < 0$.

The theorem is proved.

It can be shown that the function $\lambda(\mathbf{W},\mathbf{y},k)$ which meets the theorem condition *1)* under condition *2)* always exists. To obtain this it will be enough to consider the variation rate of a left bound γ parameters (to refer to the condition *2)*) for small increment $\delta w_{i,j}^{(\ell)}$. It is necessary to emphasize, that in a general case the stability by Lyapunov's method and, hence, the negative determinacy of Lyapunov function do not guarantee the stability of the gradient procedure (3). However, when we choose a suitable Lyapunov's function V, which equals the MNN training functional Q accurate to any constant (as it is shown in the proof), the function V negative determinacy provides non-growing functional Q variations.

It is following from the theorem condition *1)* that the function may be defined by the following inequality:

$$\delta\mathbf{y}(\mathbf{W},k)^T\delta\mathbf{y}(\mathbf{W},k)\left(\sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\left(\delta w_{i,j}^{(\ell)}\right)^2\right)^{-1} \le \lambda(\mathbf{W},\mathbf{y},k) \tag{13}$$

Let symbol $\mathbf{J}(h)$ is:

$$\mathbf{J}(h) = \left(\prod_{\nu=3}^{h}\frac{\partial\mathbf{y}(\nu)}{\partial\mathbf{y}(\nu-1)}\right)\frac{\partial\mathbf{y}(2)}{\partial\mathbf{x}}\frac{\partial\mathbf{x}(2)}{\partial\mathbf{q}^{(K)}(1)} \tag{14}$$

When we discover the $\delta\mathbf{y}(\mathbf{W},k)$ increment in (13) refer to (14) we'll receive:

$$\left(\sum_{h=2}^{k}\mathbf{J}(h)\sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\frac{\partial\mathbf{q}^{(K)}(1)}{\partial w_{i,j}^{(\ell)}}\delta w_{i,j}^{(\ell)} + \mathbf{e}(\delta\mathbf{W},\mathbf{y},h)\right)^T \times \left(\sum_{h=2}^{k}\mathbf{J}(h)\sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\frac{\partial\mathbf{q}^{(K)}(1)}{\partial w_{i,j}^{(\ell)}}\delta w_{i,j}^{(\ell)} + \mathbf{e}(\delta\mathbf{W},\mathbf{y},h)\right)\left(\sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\left(\delta w_{i,j}^{(\ell)}\right)^2\right)^{-1} \le \lambda(\mathbf{W},\mathbf{y},k). \tag{15}$$

The inequality (15) allows to estimate bounds of the γ parameters "stable" area. In particular, if we neglect the function $\mathbf{e}(\delta\mathbf{W},\mathbf{y},h)$ values because of the increments $\delta w_{i,j}^{(\ell)}$ are rather small (as far as the function $\mathbf{e}(\delta\mathbf{W},\mathbf{y},h)$ is $o(\delta\mathbf{W})$) we'll obtain the following step γ estimation:

$$\gamma \le \sum_{k=2}^{N} \left(\left(\sum_{h=2}^{k} \mathbf{J}(h) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \frac{\partial \mathbf{q}^{(K)}(1)}{\partial w_{i,j}^{(\ell)}} \frac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right)^T \times \left(\sum_{h=2}^{k} \mathbf{J}(h) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \frac{\partial \mathbf{q}^{(K)}(1)}{\partial w_{i,j}^{(\ell)}} \frac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right) \right)^{-1} \left(\sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \left(\frac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right)^2 \right). \quad (16)$$

We should like to emphasize that the theorem conditions are sufficient for the gradient procedure (3). Also their observation provides roughness of the network training algorithm. Besides this function $\lambda(\mathbf{W}, \mathbf{y}, k)$ can be used for estimation of the MNN architecture and single neuron activation function influence the BPTT procedure stability.

The obtained expressions may be used for *adaptive* algorithm of MNN training within GTP because the function $\gamma(n)$ is corrected (varied) during the training process referring plant state, the amounts of MNN layers, neurons in the layer, the network initial conditions, the current weight-factor value and to the activation function in covert form. In other words this function depends on the plant's dynamics, the MNN architecture and state.

Let us consider an example.

Let a plant is defined by the following equation system:

$$\begin{cases} x_1(k+1) = x_1(k) + (-x_2(k) - x_3(k) + u_1(k))\Delta t \\ x_2(k+1) = x_2(k) + (x_1(k) + 0.3x_2(k) + u_2(k))\Delta t \\ x_3(k+1) = x_3(k) + (0.38x_1(k) - 4.5x_2(k) + x_1(k)x_3(k) + u_3(k))\Delta t, \end{cases}$$

where Δt is a discretization step. Let all components of the state vector are measured. Hence, the Jacobian $\frac{\partial \mathbf{x}(j)}{\partial \mathbf{y}(j-1)}$ used in BPTT algorithm exists:

$$\frac{\partial \mathbf{x}(j)}{\partial \mathbf{x}(j-1)} = \begin{pmatrix} 1 & -\Delta t & -\Delta t \\ \Delta t & 0.3\Delta t + 1 & 0 \\ (0.38 + x_3)\Delta t & -4.5\Delta t & x_1 \Delta t + 1 \end{pmatrix} + \begin{pmatrix} \Delta t & 0 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \Delta t \end{pmatrix} \frac{\partial \mathbf{u}(j-1)}{\partial \mathbf{x}(j-1)}$$

A reference function is set as $\mathbf{y}^*(t) = col(e^{-t}, e^{-t}, e^{-t})$ function.

To simulat the process we chose some MNN with a $N^3_{3,50,50,3}$ architecture and the sigmoidal activation functions $f(s) = \frac{1 - e^{-2s}}{1 + e^{-2s}}$ The conditions of training simulation was: N = 30 trajectory points, Δt =0.01c descretization step, $x_1(1) = x_2(1) = x_3(1) = 1$ initial conditions. The estimation of γ parameter is chosen as in the inequality (6). Let us mark:

$$\mathbf{D}(\ell) = \prod_{\ell_1 = K}^{\ell+1} \mathbf{w}^{(\ell_1)}; \quad (17)$$

$$d_{j,i}^{(\ell)} = \delta(j,i) \max(f_i'^{(\ell)}) \left| q_j^{(\ell-1)} \right|, \quad (18)$$

where $\delta(j,i)$ is Kronecer's symbol, $j = \overline{1, n_{\ell-1}}$, $i = \overline{1, n_\ell}$. After the partial derivations $\frac{\partial Q}{\partial w_{i,j}^{(\ell)}}$ were discovered in (16) we receive refer to (17), (18):

$$\gamma \le \sum_{k=2}^{N} \left(\left(\sum_{h=2}^{k} \mathbf{J}(h) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \left| \mathbf{D}(\ell) \mathbf{d}_i^{(\ell)} \right| \left(\mathbf{D}(\ell) \mathbf{d}_i^{(\ell)} \right)^T \boldsymbol{\sigma} \right)^T \times \left(\sum_{h=2}^{k} \mathbf{J}(h) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \left| \mathbf{D}(\ell) \mathbf{d}_i^{(\ell)} \right| \times \left(\mathbf{D}(\ell) \mathbf{d}_i^{(\ell)} \right)^T \boldsymbol{\sigma} \right) \right)^{-1} \times \left(\sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \left(\frac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right)^2 \right). \quad (19)$$

These parameter γ estimations satisfy the theorem conditions when the variations $\delta w_{i,j}^{(\ell)}$ are small. Besides it can be shown that the weight-factors $\left|w_{i,j}^{(\ell)}\right| > w_{i,j\,\text{bound}}^{(\ell)}$ exist for MNN with sigmoidal activation functions and they provide a decrement of the criterion $Q(n)$ value.

The computer simulation showed that neural network training procedure has an undiverging reference. In fig. 2 a, b accordingly you can see the graphs which illustrates the variation of the algorithm step $\gamma(n)$ and training functional $Q(n)$ in the cases when estimation (19) is used and when $\gamma = \text{const} = 0.0015$.

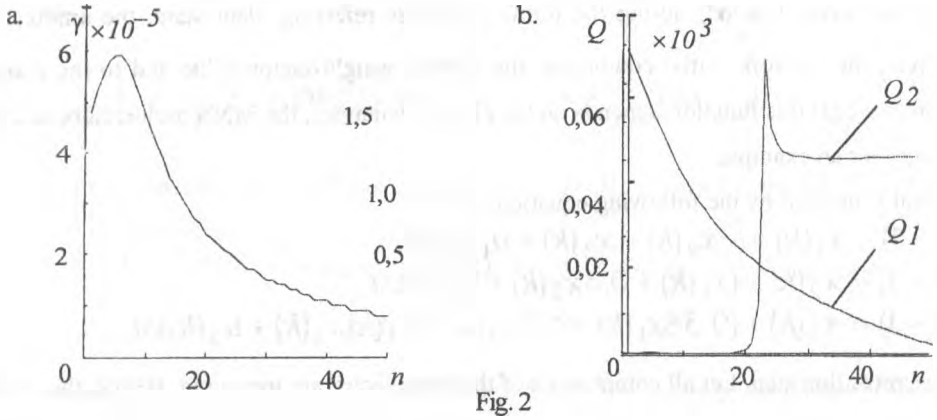

Fig. 2

This graphs allow to conclude that the computer simulation does not to contradict the theoretical items of the work.

3. SBP ALGORITHM

Let the dynamics of control plant is described by equations:

$$\mathbf{x} = \mathbf{f}(\mathbf{x}, \boldsymbol{\theta}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \quad \mathbf{x} \in R^n, \quad \mathbf{u} \in R^m. \tag{20}$$

The behaviour of control plant (20) depends on unknown parameters values $\boldsymbol{\theta} \in \Omega_{\boldsymbol{\theta}}$. The goal of control presented as desirable dynamic characteristics of control plant based on a reference model described by a system of linear differential equations:

$$\mathbf{x}^{\mathcal{M}} = \mathbf{A}^{\mathcal{M}}\mathbf{x}^{\mathcal{M}} + \mathbf{B}^{\mathcal{M}}\mathbf{r}, \tag{21}$$

where $\mathbf{x}^{\mathcal{M}} \in R^n$ is a state vector of a reference model, $\mathbf{A}^M, \mathbf{B}^M$ are model matrixes chosen so that a response of state vector $\mathbf{x}^{\mathcal{M}}$ on assigning action $\mathbf{r}$ carried a desirable character.

Let a control vector $\mathbf{u} = \mathbf{q}^{(K)}$ be the output of the last K th layer of a network. It is required to derive network training algorithm, which would change the vector of adjusted parameters $\mathbf{w}_i^{(\ell)}$ (where $\ell = \overline{1, K}$ - number of MNN layer, and $i = \overline{1, n_\ell}$ - number in a layer) to ensure reaching control goal at any vector of unknown parameters $\boldsymbol{\theta} \in \Omega_{\boldsymbol{\theta}}$.

A speed-gradient algorithm in the combined form [5] for set-up vectors $\mathbf{w}_i^{(\ell)}$ can be written as follows:

$$\frac{d\left(\mathbf{w}_i^{(\ell)} + \psi\right)}{dt} = -\Gamma \nabla_{\mathbf{w}} \omega(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t), \quad i = \overline{1, n_\ell}, \tag{22}$$

where $\omega(\,)$ is function, continuously differentiable on components of a vector $\mathbf{w}_i^{(\ell)}$; $\Gamma = \Gamma^T > 0$ is $((n_\ell + 1) \times (n_\ell + 1))$ gain matrix; ψ is a pceudo-gradient function which meets to condition $\| \psi\, \omega(\cdot) \| \geq 0$. The stability theorems for speed-gradient systems (20),(22) can be found in [5].

For quasistationary plant the function $\omega(\cdot)$ is determined as follows:

$$\omega(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t) = \left[\mathbf{f}(\mathbf{x}, \boldsymbol{\theta}) + \mathbf{g}(\mathbf{x})\mathbf{u}(\mathbf{w}_i^{(\ell)})\right]^T \nabla_{\mathbf{x}} J(\mathbf{x}; t) + \mathbf{x}^{\mathcal{M}^T} \nabla_{\mathbf{x}^{\mathcal{M}}} J(\mathbf{x}; t). \tag{23}$$

Instead of (22) it is possible to use SG algorithm in its finite form:

$$\mathbf{w}_i^{(\ell)} = -\psi(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t),\ i = \overline{1, n_\ell}\,.$$

The typical form of a finite algorithm is linear

$$\mathbf{w}_i^{(\ell)} = -\Gamma \nabla_{\mathbf{w}} \omega(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t),\ i = \overline{1, n_\ell}, \tag{24}$$

Let MNN training function be as follows:

$$J(\mathbf{x}, \mathbf{x}^{\mathcal{M}}) = 0.5\left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right)^T \left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right).$$

Then

$$\omega(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t) = \left[\mathbf{x}^{\mathcal{M}} - \mathbf{f}(\mathbf{x}, \boldsymbol{\theta}) - \mathbf{g}(\mathbf{x})\mathbf{u}(\mathbf{w}_i^{(\ell)})\right]^T \left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right). \tag{25}$$

In this case equations of adjusting weight factors $\mathbf{w}_i^{(\ell)}$ in MNN layers $\ell = \overline{1, K}$ by virtue of (25) will take the dependence of:

$$\frac{d\mathbf{w}_i^{(\ell)}}{dt} = \Gamma \left(\frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{w}_i^{(\ell)}} \right) \mathbf{g}(\mathbf{x})^T \left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right),\ i = \overline{1, n_\ell}\,, \tag{26}$$

or

$$\mathbf{w}_i^{(\ell)} = \Gamma \left(\frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{w}_i^{(\ell)}} \right) \mathbf{g}(\mathbf{x})^T \left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right),\ i = \overline{1, n_\ell}\,, \tag{27}$$

The evaluation of derivatives $\partial\, \mathbf{q}^{(K)} \big/ \partial\, \mathbf{w}_i^{(\ell)}$ in algorithms (26) - (27) is produced in the correspondence with a back propagation error technique:

$$\frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{w}_i^{(\ell)}} = \frac{\partial\, \mathbf{q}_i^{(\ell)}}{\partial\, \mathbf{w}_i^{(\ell)}} \frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{q}_i^{(\ell)}} = \mathbf{f}'(\mathbf{s}_i^{(\ell)}) \mathbf{q}^{(\ell-1)} \frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{q}_i^{(\ell)}};$$

$$\frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{q}_i^{(\ell)}} = \mathbf{W}^{(\ell+1)} \mathbf{f}'(\mathbf{s}^{(\ell+1)}) \frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{q}^{(\ell+1)}}.$$

where $\mathbf{s}^{(\ell)} = \mathbf{W}^{(\ell)} \mathbf{q}^{(\ell-1)}$ is first order discriminant function and $\mathbf{f}'(\mathbf{s}^{(\ell)}) = \left. \dfrac{d\mathbf{f}(\mathbf{s})}{d\mathbf{s}} \right|_{\mathbf{s}=\mathbf{s}^{(\ell)}}$ is $(n_\ell \times n_\ell)$ diagonal matrix , $\mathbf{W}^{(\ell)}$ is weight matrix of ℓ th layer.

If the matrix function $\mathbf{g}(\mathbf{x})$ does not depend on a state vector $\mathbf{x}$, i.e. $\mathbf{g}(\mathbf{x}) = \mathbf{B}$, that is taking into account an arbitrary choice of a matrix Γ, substituted to the algorithm (26) - (27) all nonzero elements of a matrix $\mathbf{B}$ can be reduced by unity. It allows to expand the class of parametrical indeterminacy of plant, and assume the matrix $\mathbf{B}$

depending on the vector of unknown parameters $\boldsymbol{\theta}$.

The obtained no autonomous algorithms of on-line network training in a neural net control system structure (26) - (27) are modifications of algorithm SBP. However algorithms (26) - (27) are deprived of the listed above shortages of algorithm SBP and use a measurement information for network training, that adds some adaptive properties in neural net control systems .

Let's illustrate an overall performance of the considered network training algorithms on forced brusselator's model excitation of oscillations example [6]:

$$\begin{cases} \dot{x}_1 = a - (b+1)x_1 + x_1^2 x_2 + u; \\ \dot{x}_2 = bx_1 - x_1^2 x_2, \end{cases}$$

The value of a parameter a is unknown and belongs to area of admissible values $\Omega_{\boldsymbol{\theta}}$, specified by interval $0.1 \le a \le 1$.

We shall choose the MNN parameters as follows: numbers of layers $K = 2$, numbers of neurons in hidden layer $n_1 = 3$, numbers of neurons in output layer $n_2 = 1$.

Based on the control plant equations, algorithms (26) and (27) can be written as follows:

$$\frac{d\mathbf{w}_i^{(\ell)}}{dt} = -\gamma \left(\frac{\partial q^{(K)}}{\partial \mathbf{w}_i^{(\ell)}} \right) \left(x_1 - x_1^{\mathcal{M}} \right), \; i = \overline{1, n_\ell},$$

$$\mathbf{w}_i^{(\ell)} = -\gamma \left(\frac{\partial q^{(K)}}{\partial \mathbf{w}_i^{(\ell)}} \right) \left(x_1 - x_1^{\mathcal{M}} \right), \; i = \overline{1, n_\ell},$$

where $\Gamma = \gamma = 0.1$ is a gain factor.

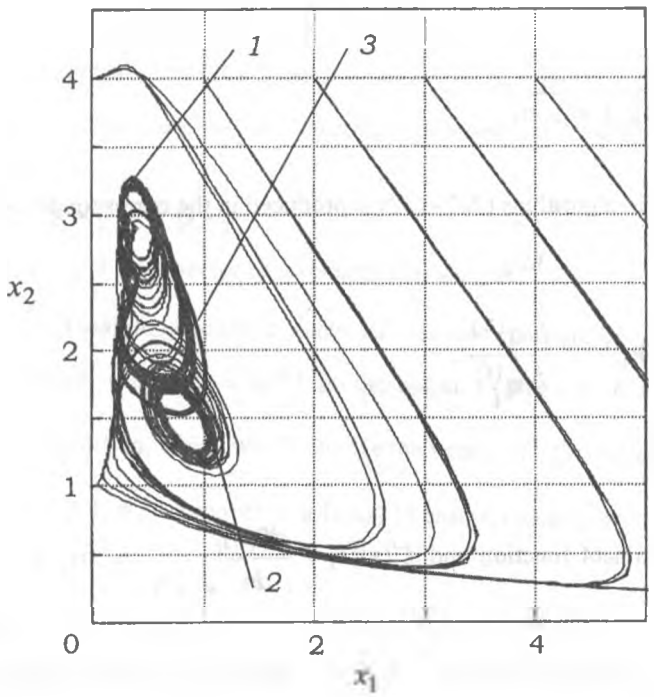

Fig. 3. Phase portraits: *1* - forced brusselator chaotic attractor, *2* - control system with MNN trained on algorithm (26), *3* - control system with MNN trained on algorithm (27)

The following parameters values are chosen for simulation: $a = 0.4 + 0.05\sin(0.81t)$ and $b = 1.2$. At these parameters values in a model phase space there is a chaotic attractor designated as *1* in fig. 3. With the purpose of excitation of oscillations we shall choose reference motion as $x_1^M = 0.8 + 0.5\sin(0.81t)$. The phase portraits of neural net control system, trained on algorithms (26) and (27) are reduced on fig. 3 and are designated as *2* and *3* accordingly.

4. CONCLUSION

In this paper we receive the step estimations for the gradient procedure (3) which provide a training process stability when MNN is included in to the dynamical system (1). Such estimations allow to make a synthesis of an adaptive neural network training algorithms. If use BPTT training technique we can beforehand estimate quality of the neurosystem for the final time interval. It should be emphasize the obtain result may be used in the case when we have a half-final time interval. Future researches will be directed toward an roughness, controlability and observiability of the control system with a neural network.

The introduction of dynamics in multilaered static neural networks training algorithm is added to it properties of a dynamic networks, but without use of feed-backs, as in algorithm BPTT. The process of network training on dynamic algorithm allows to unit in one processes of training and control in static MNN, that is essential at use it as the controller in dynamic systems.

The modifications of speed back propogation algorithm are presented. The efficiency of the proposed algorithms for some chaotic oscillation control problems has been demonstrated. The results of computer simulation demonstrate coincidence of trajectories of own plant motions and trajectories of control system on a significan time interval . It testifies about minimal interference in own plant dynamics and requeres the minimum controls for reaching a control goal.

5. REFERENCES

[1] Terekhov V.A., Efimov D.V., Tyukin I.Y., Antonov V.N. Neural net control systems. — St.Petersburg: Publishing House of St/ Petersburg University, 1999. 265 p.

[2] Terekhov V.A., Yakovlev V.B., Efimov D.V., Tyukin I.Y. The Control and Training Algorithms for Intelligent Systems with Multilayer Neural Networks// The 2-d internat. symposium "INTELS'96" thesis , June 1-4 1996. Vol.1. SPb. - Moscow: 1996. pp 11- 14.

[3] Werbos P.J. Backpropagation Through Time: what it does and how to do it? // Proc. of IEEE. 1990. Vol. 78. pp. 1550-1560.

[4] V.A. Terehov "Dynamic algorithms of multilayered neural networks training in control systems" , Izv. vuz. The theory and control systems, vol. 3, 1996, pp. 70-79.

[5] A.L. Fradkov "Adaptive Control in Complex Systems", Moscow: Nauka, 1990. (Russian).

[6] Fradkov A.L., Pogromsky A.Yu. Introduction to control of oscillations and chaos // World scientific series on nonlinear science / Ed. Leon O. Chua. Ser. A. 1998. Vol. 35. P. 391.