A new adaptive backpropagation algorithm based on Lyapunov stability theory for neural networks

Man, Zhihong; Wu, Henry; Liu, Sophie; Yu, Xinghuo

Man, Wu, H., Liu, S., & Yu, X. (2006). A new adaptive backpropagation algorithm based on Lyapunov stability theory for neural networks. IEEE Transaction on Neural Networks, 17(6), 1580–1591. https://doi.org/10.1109/TNN.2006.880360

Published Version: https://doi.org/10.1109/TNN.2006.880360

A New Adaptive Backpropagation Algorithm Based on Lyapunov Stability Theory for Neural Networks

Zhihong Man, Hong Ren Wu, Sophie Liu, and Xinghuo Yu, *Senior Member, IEEE*

***Abstract*—A new adaptive backpropagation (BP) algorithm based on Lyapunov stability theory for neural networks is developed in this paper. It is shown that the candidate of a Lyapunov function $V(k)$ of the tracking error between the output of a neural network and the desired reference signal is chosen first, and the weights of the neural network are then updated, from the output layer to the input layer, in the sense that $\Delta V(k) = V(k) - V(k-1) < 0$. The output tracking error can then asymptotically converge to zero according to Lyapunov stability theory. Unlike gradient-based BP training algorithms, the new Lyapunov adaptive BP algorithm in this paper is not used for searching the global minimum point along the cost-function surface in the weight space, but it is aimed at constructing an energy surface with a single global minimum point through the adaptive adjustment of the weights as the time goes to infinity. Although a neural network may have bounded input disturbances, the effects of the disturbances can be eliminated, and asymptotic error convergence can be obtained. The new Lyapunov adaptive BP algorithm is then applied to the design of an adaptive filter in the simulation example to show the fast error convergence and strong robustness with respect to large bounded input disturbances.**

***Index Terms*—Adaptive filtering, backpropagation (BP), convergence, feedforward neural networks, Lyapunov stability.**

I. Introduction

THE applications of neural networks have been receiving a great deal of attentions in many engineering disciplines. By properly choosing neural network structures and training the weights, engineers and researchers may use neural networks for digital signal processing, system modeling, automatic control, and others [1]–[9].

The conventional trainings of neural networks are mainly based on optimization theory. For example, a cost function is first defined, which may be the sum of squared errors or the mean squared error between the output of a neural network and the desired reference signal. The cost function is then minimized in the weight space, and a set of optimal weights may be obtained. The neural network with the optimal weights can then be used to perform some special tasks. In order for searching the optimal weights for neural networks, a number of algorithms have been developed. The gradient-based backpropagation (BP) training algorithms are probably the most popular ones [2].

It is well known that the gradient-based BP training algorithms may have a slow convergence in practice, and the searching for the global minimum point of a cost function may be trapped at local minima during gradient descent. Also, if a neural network has large bounded input disturbances, the global minimum point may not be found. Therefore, the fast error convergence and strong robustness of the neural network with the gradient-based BP algorithms may not be guaranteed.

In order to avoid the aforementioned problems, some sliding mode control-based adaptive training algorithms have been developed recently in [10]–[15]. These adaptive learning schemes have been used to train Adalines and multilayer neural networks with good convergence and robustness. In this paper, we propose a new adaptive BP algorithm based on Lyapunov stability theory [16]. The basic idea of the new Lyapunov adaptive BP algorithm is as follows: The candidate of a Lyapunov function $V(k)$ (an energy function) of the tracking error is chosen first. The weights of the neural network are then adaptively updated, from the output layer to the input layer, to make $\Delta V(k) = V(k) - V(k-1) < 0$. According to Lyapunov stability theory, the candidate of the Lyapunov function is a true Lyapunov function of the error dynamics of the considered neural network, and the tracking error can then asymptotically converge to zero as time goes to infinity.

Unlike the gradient-based BP training algorithms, the new Lyapunov adaptive BP algorithm in this paper is not used for searching the global minimum point along the cost-function surface in the weight space, but it is aimed at constructing an energy surface with a single global minimum point through the adaptive adjustment of the weights as time goes to infinity. The tracking error can then asymptotically converge to zero. Another important feature of the new Lyapunov adaptive BP algorithm is that, although neural networks may have bounded input disturbances, the effects of the disturbances can be eliminated through adaptively updating the weights according to Lyapunov stability theory.

The basic idea of the optimization using Lyapunov stability theory has been recently applied to the design of the finite-impulse-response (FIR) adaptive filters, the radial basis function-based adaptive filters, and adaptive control in [17]–[23]. However, in this paper, we will further explore how Lyapunov stability theory can be used to develop a new adaptive BP scheme for neural networks. The great potential of the new Lyapunov

Manuscript received May 18, 2004; revised March 14, 2005.

Z. Man is with the School of Computer Engineering, Nanyang Technological University, Singapore 639798, Singapore (e-mail: aszhman@ntu.edu.sg).

H. R. Wu is with the School of Electrical and Computer Engineering, RMIT University, Melbourne, VIC 3001, Australia (e-mail: henry.wu@rmit.edu.au).

S. Liu is with the School of Engineering, Physics and Physical Science, Oral Roberts University, Tulsa, OK 74171 USA (e-mail: sliu@oru.edu).

X. Yu is with the Faculty of Engineering, RMIT University, Melbourne, VIC 3001, Australia (e-mail: x.yu@rmit.edu.au).

Color versions of Figs. 3–9 are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2006.880360

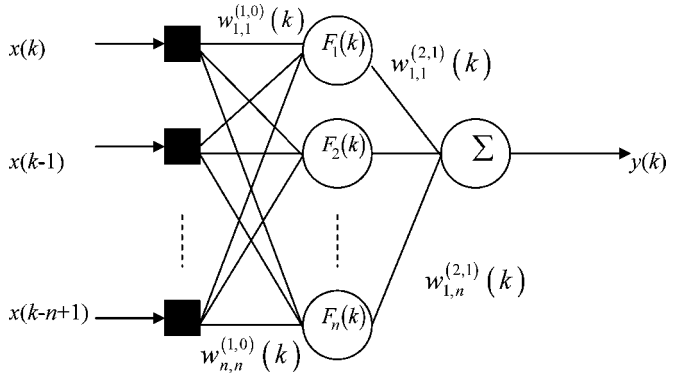


Fig. 1. Feedforward neural network with three layers.

adaptive BP algorithm will be seen from both the detailed theoretical analysis and the excellent simulation results in this paper.

The paper is organized as follows. In Section II, the new Lyapunov adaptive BP algorithm is formulated. In Section III, the convergence analyses of the Lyapunov adaptive BP algorithm with the modified weight update-laws to handle singularities are discussed. In Section IV, a simulation example for a neural network-based adaptive filter with the new Lyapunov adaptive BP algorithm is implemented to demonstrate the fast error convergence and strong robustness with respect to large bounded input disturbances.

II. Problem Formulation

The architecture of a standard feedforward neural network with three layers is presented in Fig. 1, where the output layer has a single linear node, the hidden layer has n nonlinear nodes, and the input data vector is $\boldsymbol{X}(k) = [x_1(k), x_2(k), \cdots, x_n(k)]^T = [x(k), x(k-1) \cdots, x(k-n+1)]^T$. The output of the jth nonlinear node in the hidden layer is

$$S_j(k) = F_j\left(\sum_{i=1}^{n} w_{j,i}^{(1,0)}(k) x_i(k)\right) \tag{2-1}$$

where the nonlinear function $F_j(\bullet)$ is of the form

$$F_j(\bullet) = \frac{1}{1+e^{-\alpha(\bullet)}}, \qquad \text{for} \quad j = 1, 2, \ldots, n \tag{2-2}$$

and α is a positive constant.

The output of the neural network can then be expressed as

$$\begin{aligned} y(k) &= \sum_{j=1}^{n} w_{1,j}^{(2,1)}(k) S_j(k) \\ &= \sum_{j=1}^{n} w_{1,j}^{(2,1)}(k) F_j\left(\sum_{i=1}^{n} w_{j,i}^{(1,0)}(k) x_i(k)\right). \end{aligned} \tag{2-3}$$

The idea of the new adaptive BP algorithm based on Lyapunov stability theory can be described as follows.

1) Define the tracking error $e(k)$

$$e(k) = y(k) - d(k)$$

where $d(k)$ is the desired reference signal for $y(k)$, the output of the neural network, to follow, and the point $e(k) = 0$ is assumed to be the equilibrium point of the system error dynamics.

2) Choose a candidate of Lyapunov function $V(k) = f(e(k))$, where $V(k) = f(e(k) = 0) = 0$, and $V(k) = f(e(k)) > 0$ for $e(k) \neq 0$.
3) Update the weights of the neural network at time instant k, from the output layer to the input layer, to make $\Delta V(k) = V(k) - V(k-1) < 0$.
4) Let $k = k+1$, and then go to step 2)

According to Lyapunov stability theory [16], if $V(k) > 0$ and $\Delta V(k) < 0$, $V(k)$ is a true Lyapunov function of the error dynamics of the neural network, and the output tracking error $e(k)$ can then asymptotically converge to zero as time goes to infinity.

In Theorem 2.1 and the Lemma 2.1, we discuss how the weights of the three layered neural network in Fig. 1 are updated, and how the error dynamics of the neural network behaves when using the new Lyapunov adaptive BP algorithm.

Theorem 2.1: Consider the feedforward neural network in Fig. 1. If the weights $w_{1,j}^{(2,1)}(k)$ and $w_{j,i}^{(1,0)}(k)$, from the output layer to the hidden layer, are updated in sequence as follow:

$$w_{1,j}^{(2,1)}(k) = \frac{\beta^{-\frac{k}{2}} e(k-1) + d(k)}{n S_j(k-1)} \tag{2-4}$$

and

$$w_{j,i}^{(1,0)}(k) = \frac{1}{n x_i(k)} G_j\left(\frac{\beta^{-\frac{k}{2}} e(k-1) + d(k)}{n w_{1,j}^{(2,1)}(k)}\right) \tag{2-5}$$

where

$$x_i(k) \neq 0 \quad S_j(k) \neq 0 \quad w_{1,j}^{(2,1)}(k) \neq 0$$

and

$$G_j(\bullet) = F_j^{-1}(\bullet) \tag{2-6}$$

the output tracking error $e(k)$ can then asymptotically converge to zero.

Proof: Choose the following candidate of Lyapunov function:

$$V(k) = \beta^k e^2(k) \tag{2-7}$$

where β is a positive constant and $\beta > 1$.

The difference between $V(k)$ and $V(k-1)$ is then given by

$$\begin{aligned} \Delta V(k) &= V(k) - V(k-1) \\ &= \beta^k e^2(k) - \beta^{k-1} e^2(k-1) \\ &= \beta^k \left(y(k) - d(k)\right)^2 - \beta^{k-1} e^2(k-1) \\ &= \beta^k \left(\sum_{j=1}^{n} w_{1,j}^{(2,1)}(k) S_j(k) - d(k)\right)^2 \\ &\quad - \beta^{k-1} e^2(k-1) \end{aligned} \tag{2-8}$$

Using (2-4) and (2-5) in (2-8), we have

$$\begin{aligned}\Delta V(k) &= \beta^k\Bigg(\sum_{j=1}^{n} w_{1,j}^{(2,1)}(k)F_j\Bigg(\sum_{i=1}^{n} w_{j,i}^{(1,0)}(k)x_i(k)\Bigg) \\ &\quad - d(k)\Bigg)^2 - \beta^{k-1}e^2(k-1) \\ &= \beta^k\Bigg(\sum_{j=1}^{n} w_{1,j}^{(2,1)}(k)F_j \\ &\quad \times\Bigg(\sum_{i=1}^{n}\Bigg(\frac{1}{nx_i(k)}G_j\Bigg(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{nw_{1,j}^{(2,1)}(k)}\Bigg)\Bigg) \\ &\quad \times x_i(k)\Bigg) - d(k)\Bigg)^2 - \beta^{k-1}e^2(k-1) \\ &= -(\beta^{k-1}-1)e^2(k-1) < 0. \end{aligned} \tag{2-9}$$

Therefore, according to Lyapunov stability theory, the tracking error $e(k)$ asymptotically converges to zero.

Remark 2.1: Like the conventional BP algorithms [1]–[3], the weights $w_{1j}^{(2,1)}(k)$ of the output layer are first updated using (2-4), and the values of $w_{1j}^{(2,1)}(k)$ are fixed. The weights $w_{ji}^{(1,0)}(k)$ are then updated using (2-5). It is seen that the update-law of $w_{ji}^{(1,0)}(k)$ in (2-5) plays an important role to guarantee $\Delta V(k) < 0$.

Lemma 2.1: Consider the feedforward neural network in Fig. 1 with the weight update-laws in (2-4) and (2-5). The error convergence is then specified as follows:

$$e(k) = \beta^{-\frac{(1+k)k}{4}}e(0), \qquad \text{for} \quad k = 1, 2, \ldots \tag{2-10}$$

Proof:

$$\begin{aligned} e(k) &= y(k) - d(k) \\ &= \sum_{j=1}^{n} w_{1,j}^{(2,1)}S_j(k) - d(k) \\ &= \sum_{j=1}^{n} w_{1,j}^{(2,1)}(k)F_j \\ &\quad \times\Bigg(\sum_{i=1}^{n}\Bigg(\frac{1}{nx_i(k)}G_j\Bigg(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{nw_{1,j}^{(2,1)}(k)}\Bigg)\Bigg)x_i(k)\Bigg) \\ &\quad - d(k) \\ &= \beta^{-\frac{k}{2}}e(k-1). \end{aligned} \tag{2-11}$$

It is noted that

$$\begin{aligned} e(1) &= \beta^{-\frac{1}{2}}e(0) \\ e(2) &= \beta^{-\frac{2}{2}}e(1) = \beta^{-\frac{1+2}{2}}e(0) \\ e(3) &= \beta^{-\frac{3}{2}}e(2) = \beta^{-\frac{1+2+3}{2}}e(0) \\ &\vdots \\ e(k) &= \beta^{-\frac{k}{2}}e(k-1) = \beta^{-\frac{1+2+3+\cdots+k}{2}}e(0) \\ &= \beta^{-\frac{(1+k)k}{4}}e(0). \end{aligned} \tag{2-12}$$

Remark 2.2: It is easy to see from Lemma 2.1 that the error convergence can be specified after choosing the value of parameter β in the Lyapunov function in (2-7), and update-laws in (2-4) and (2-5). Parameter β controls the error convergence rate, and the value of β must be chosen to be greater than 1 to guarantee the error convergence.

Remark 2-3: In many engineering areas, the input signal $x(k)$ to a system is of the form

$$x(k) = d(k) + n(k) \tag{2-13}$$

where $d(k)$ is the desired reference signal and $n(k)$ is a bounded disturbance. When we use the gradient-based BP algorithms to compute the weights of a neural network or a FIR adaptive filter, the optimal weights are closely related to the stochastic properties of the input signal $x(k)$. However, in the new Lyapunov adaptive BP algorithm in this paper, we do not need to know the stochastic properties of $x(k)$. What we need to do is to develop the parameter update-laws using the measurements of $x(k)$ and the error $e(k)$ to update the weights of a neural network in the sense that $\Delta V(k) < 0$, which guarantees that the tracking error $e(k)$ asymptotically converges to zero. Therefore, the Lyapunov adaptive BP algorithm in this paper is independent of the stochastic properties of the input signal $x(k)$. This feature is very important and useful for practical application.

Remark 2-4: It is also noted that the parameter update-laws in (2-4) and (2-5) can guarantee that the Lyapunov function has a single global minimum point as the time scale k goes to infinity. However, the gradient-based algorithms with the associated parameter update-laws may not have such a good property. This is because a gradient-based adaptive algorithm is to search for the global minimum point of a *cost-function surface*, and such a searching may not find the global minimum point when the cost function has local minima. Therefore, it is believed that the new Lyapunov adaptive BP algorithm in this paper has provided a new way to further improve dynamic properties of neural networks.

Remark 2-5: The bounded property of the adjustable weights $w_{1,j}^{(2,1)}(k)$ and $w_{j,i}^{(1,0)}(k)$ using the update-laws in (2-4) and (2-5) are briefly described as follows. When the time scale k is large enough, $w_{1,j}^{(2,1)}(k)$ in (2-4) can be approximated as follows:

$$w_{1,j}^{(2,1)}(k) \approx \frac{d(k)}{nS_j(k-1)}. \tag{2-14}$$

$S_j(k)$ can then be expressed as

$$\begin{aligned} S_j(k) &= F_j\Bigg(\sum_{i=1}^{n} w_{j,i}^{(1,0)}(k)x_i(k)\Bigg) \\ &\approx F_j\Bigg(\sum_{i=1}^{n}\Bigg(\frac{1}{nx_i(k)}G_j\Bigg(\frac{d(k)}{nw_{1,j}^{(2,1)}(k)}\Bigg)\Bigg)x_i(k)\Bigg) \\ &= F_j\Bigg(G_j\Bigg(\frac{d(k)}{nw_{1,j}^{(2,1)}(k)}\Bigg)\Bigg) = \frac{d(k)}{nw_{1,j}^{(2,1)}(k)}. \end{aligned} \tag{2-15}$$

Using (2-15) in (2-14), we have

$$w_{1,j}^{(2,1)}(k) \approx \frac{d(k)}{d(k-1)}w_{1,j}^{(2,1)}(k-1). \tag{2-16}$$

If the initial value of $w_{1,j}^{(2,1)}(k)$ is $w_{1,j}^{(2,1)}(0)(\neq 0)$, the solution of (2-16) is of the form

$$w_{1,j}^{(2,1)}(k) = \frac{d(k)}{d(0)} w_{1,j}^{(2,1)}(0). \tag{2-17}$$

Therefore, for bounded nonzero $d(k)$ $(k = 0, 1, \ldots)$ and bounded $w_{1,j}^{(2,1)}(0)$ $(\neq 0)$, the norm of $w_{1,j}^{(2,1)}(k)$ is upper bounded.

In addition, for large value of the time scale k

$$\begin{aligned} w_{j,i}^{(1,0)}(k) &\approx \frac{1}{n x_i(k)} G_j\left(\frac{d(k)}{n w_{1,j}^{(2,1)}(k)}\right) \\ &= \frac{1}{n x_i(k)} G_j\left(\frac{d(0)}{n w_{1,j}^{(2,1)}(0)}\right). \end{aligned} \tag{2-18}$$

It is noted that $G_j(d(0)/nw_{1,j}^{(2,1)}(0))$ is a constant determined by the initial values of $d(k)$ and $w_{1,j}^{(2,1)}(k)$. Therefore, for nonzero $x_i(k)$, whose norm is lower and upper bounded, the norm of $w_{j,i}^{(1,0)}(k)$ is upper bounded.

III. Lapunov Adaptive BP Algorithm With Modified Update-Laws to Handle Singularities

Theorem 2.1 and Lemma 2.1 have described the fundamentals of the new Lyapunov adaptive BP algorithm for neural network in Fig. 1, with weight update-laws in (2-4) and (2-5), where all $x_i(k)$, $S_j(k-1)$, and $w_{1,j}^{(2,1)}(k)$ are assumed to be nonzero. However, in practice, some $S_j(k-1)$ may be near zero, and the corresponding $w_{1,j}^{(2,1)}(k)$ may have a very large value. In order to avoid such a situation in practice, the update-law of $w_{1,j}^{(2,1)}(k)$ in (2-4), when $S_j(k-1)$ is near zero for $j = l$, may be modified as follows:

$$w_{1,j}^{(2,1)}(k) = \begin{cases} \frac{\beta^{-\frac{k}{2}} e(k-1) + d(k)}{(n-1) S_j(k-1)}, & \text{for } j \neq l \\ \frac{\beta^{-\frac{k}{2}} e(k-1) + d(k)}{S_l(k-1) + \lambda_l}, & \text{for } j = l \end{cases} \tag{3-1}$$

where λ_l is a small positive number.

Also, at time instant k: 1) some $x_i(k)$ may be near or equal to zero; 2) some $w_{1,j}^{(2,1)}(k)$ may be near or equal to zero; and 3) both some $x_i(k)$ and some $w_{1,j}^{(2,1)}(k)$ may be near or equal to zero and so on. Therefore, it is necessary to properly modify the update-law of $w_{j,i}^{(1,0)}(k)$ in (2-5) to avoid the singularities. In the following, we first modify the parameter update-law in (2-5) to handle the singularity in the case 3), and then explore the corresponding error convergence region in Theorem 3.1. The modifications of parameter update-law of $w_{j,i}^{(1,0)}(k)$ and error convergence for cases 1) and 2) are briefly summarized in Lemmas 3.1 and 3.2.

First, we assume that both $x_i(k)$ and $w_{1,j}^{(2,1)}(k)$ may be near or equal to zero at time instant k, for $i = i1$, and $j = j1$, respectively. We then modify the update-law for $w_{j,i}^{(1,0)}(k)$ in (2-5) as follows:

$$w_{j,i}^{(1,0)}(k) = \begin{cases} \dfrac{G_j\left(\frac{\beta^{-\frac{k}{2}} e(k-1) + d(k)}{(n-1) w_{1,j}^{(2,1)}(k)}\right)}{(n-1) x_i(k)}, & \text{for } i \neq i1 \text{ and } j \neq j1 \\ \dfrac{G_j\left(\frac{\beta^{-\frac{k}{2}} e(k-1) + d(k)}{(n-1) w_{1,j}^{(2,1)}(k)}\right)}{x_{i1}(k) + \lambda_{i1}}, & \text{for } i = i1 \text{ and } j \neq j1 \\ \dfrac{G_{j1}\left(\frac{\beta^{-\frac{k}{2}} e(k-1) + d(k)}{w_{1,j1}^{(2,1)}(k) + \lambda_{j1}}\right)}{(n-1) x_i(k)}, & \text{for } i \neq i1 \text{ and } j = j1 \\ \dfrac{G_{j1}\left(\frac{\beta^{-\frac{k}{2}} e(k-1) + d(k)}{w_{1,j1}^{(2,1)}(k) + \lambda_{j1}}\right)}{x_{i1}(k) + \lambda_{i1}}, & \text{for } i = i1 \text{ and } j = j1 \end{cases} \tag{3-2}$$

where λ_{i1} and λ_{j1} are small positive numbers.

Theorem 3.1: Consider the neural network in Fig. 1. If $w_{1,j}^{(2,1)}(k)$ and $w_{j,i}^{(1,0)}(k)$ are updated using (3-1) and (3-2), respectively, the tracking error $e(k)$ will then converge to the ball, centered at the system origin, with radius

$$r = \sqrt{\beta} \left| \gamma(k) + d(k) \left(\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k) + \lambda_{j1}} \right) \right| \tag{3-3}$$

where

$$\begin{aligned} \gamma(k) = \frac{\alpha \alpha_3}{4} \frac{x_{i1}(k)}{x_{i1}(k) + \lambda_{i1}} \sum_{\substack{j=1 \\ j \neq j1}}^{n} w_{1j}^{(2,1)}(k) \sum_{\substack{i=1 \\ i \neq i1}}^{n} w_{j,i}^{(1,0)}(k) x_i(k) \\ + \frac{\alpha \alpha_4}{4} \frac{w_{1,j1}^{(2,1)}(k) x_{i1}(k)}{x_{i1}(k) + \lambda_{i1}} \sum_{\substack{i=1 \\ i \neq i1}}^{n} w_{j1,i}^{(1,0)}(k) x_i(k) \end{aligned} \tag{3-4}$$

with $0 < \alpha_3 \leq 1$ and $0 < \alpha_4 \leq 1$.

Proof: Choosing the candidate of Lyapunov function $V(k) = \beta^k e^2(k)$, we can then express $\Delta V(k)$ as follows:

$$\begin{aligned} \Delta V(k) = \beta^k \left(\sum_{j=1}^{n} w_{1,j}^{(2,1)}(k) F_j \left(\sum_{i=1}^{n} w_{j,i}^{(1,0)}(k) x_i(k) \right) - d(k) \right)^2 \\ - \beta^{k-1} e^2(k-1). \end{aligned} \tag{3-5}$$

If, at time instant k, $w_{1,j}^{(2,1)}(k)$ and $x_i(k)$ are near or equal to zero, for $j = j1$ and $i = i1$, respectively, $\sum_{i=1}^{n} w_{j,i}^{(1,0)}(k) x_i(k)$ may be written as follows:

$$\sum_{i=1}^{n} w_{j,i}^{(1,0)}(k) x_i(k) = \sum_{\substack{i=1 \\ i \neq i1}}^{n} w_{j,i}^{(1,0)}(k) x_i(k) + w_{j,i1}^{(1,0)}(k) x_{i1}(k), \quad j \neq j1 \tag{3-6}$$

$$\sum_{i=1}^{n} w_{j1,i}^{(1,0)}(k) x_i(k) = \sum_{\substack{i=1 \\ i \neq i1}}^{n} w_{j1,i}^{(1,0)}(k) x_i(k) + w_{j1,i1}^{(1,0)}(k) x_{i1}(k), \quad j = j1. \tag{3-7}$$

Therefore, using the update-law (3-2) in (3-6) and (3-7), we can write (3-5) as shown in (3-8) at the bottom of the page.

According to the geometry of the nonlinear function $F_j(\bullet)$ (see *Remark* 3.1), $F_j(G_j(\bullet)+(x_{i1}(k)/(x_{i1}(k)+\lambda_{i1}))G_j(\bullet))$ in (3-8) can be expressed as

$$\begin{aligned}
&F_j\left(G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right.\\
&\quad\left.+\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right)\\
&\quad=F_j\left(G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right)\\
&\qquad+\alpha_3\left(F_j\left(\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right)\right.\\
&\qquad\left.-F_j(0)\right) \qquad (3\text{-}9)
\end{aligned}$$

with $0<\alpha_3\le 1$.

Also, $F_j((x_{i1}(k)/(x_{i1}(k)+\lambda_{i1}))G_j(\bullet))$ in (3-9) can be approximated by (see *Remark* 3.1)

$$\begin{aligned}
&F_j\left(\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right)\\
&\quad\approx F_j(0)+\frac{\alpha}{4}\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\\
&\quad=F_j(0)+\frac{\alpha}{4}\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}\sum_{\substack{i=1\\ i\neq i1\\ j\neq j1}}^{n}w_{j,i}^{(1,0)}(k)x_i(k). \qquad (3\text{-}10)
\end{aligned}$$

Therefore, (3-9) becomes

$$\begin{aligned}
&F_j\left(G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right.\\
&\quad\left.+\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right)\\
&\quad=F_j\left(G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right)\\
&\qquad+\alpha_3\left(F_j\left(\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right)\right.\\
&\qquad\left.-F_j(0)\right)\\
&\quad=F_j\left(G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right)\\
&\qquad+\frac{\alpha\alpha_3}{4}\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}\sum_{\substack{i=1\\ i\neq i1\\ j\neq j1}}^{n}w_{j,i}^{(1,0)}(k)x_i(k)\\
&\quad=\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}+\frac{\alpha\alpha_3}{4}\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}\\
&\qquad\times\sum_{\substack{i=1\\ i\neq i1}}^{n}w_{j,i}^{(1,0)}(k)x_i(k)\quad\text{for}\quad j\neq j1. \qquad (3\text{-}11)
\end{aligned}$$

Similarly, $F_{j1}(\bullet)$ in (3-8) can be written as

$$\begin{aligned}
&F_{j1}\left(G_{j1}\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\right)\right.\\
&\quad\left.+\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_{j1}\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\right)\right)\\
&\quad=F_{j1}\left(G_{j1}\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\right)\right)\\
&\qquad+\alpha_4\left(F_{j1}\left(\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_{j1}\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\right)\right)\right.\\
&\qquad\left.-F_{j1}(0)\right)\\
&\quad=F_{j1}\left(G_{j1}\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\right)\right)\\
&\qquad+\frac{\alpha\alpha_4}{4}\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}\sum_{\substack{i=1\\ i\neq i1}}^{n}w_{j1,i}^{(1,0)}(k)x_i(k)\\
&\quad=\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}+\frac{\alpha\alpha_4}{4}\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}\\
&\qquad\times\sum_{\substack{i=1\\ i\neq i1}}^{n}w_{j1,i}^{(1,0)}(k)x_i(k) \qquad (3\text{-}12)
\end{aligned}$$

with $0<\alpha_4\le 1$.

$$\begin{aligned}
\Delta V(k)=\beta^k&\left(\sum_{\substack{j=1\\ j\neq j1}}^{n}w_{1j}^{(2,1)}(k)F_j\left(G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)+\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_j\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}\right)\right)+w_{1,j1}^{(2,1)}(k)F_{j1}\right.\\
&\left.\times\left(G_{j1}\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\right)+\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}G_{j1}\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\right)\right)-d(k)\right)^2-\beta^{k-1}e^2(k-1) \qquad (3\text{-}8)
\end{aligned}$$

Then, using (3-11) and (3-12) in (3-8), we have (3-13) as shown at the bottom of the page.

Considering the definition $\gamma(k)$ in (3-4), we can write (3-13) as

$$\begin{aligned}\Delta V(k)=&\beta^k\Bigg(\beta^{-\frac{k}{2}}e(k-1)+\gamma(k)\\&+\frac{w_{1,j1}^{(2,1)}(k)\Big(\beta^{-\frac{k}{2}}e(k-1)+d(k)\Big)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\Bigg)^2\\&-\beta^{k-1}e^2(k-1)\\=&-\Bigg(\beta^{k-1}-1-2\left(\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{J1}}\right)\\&-\left(\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{J1}}\right)^2\Bigg)e^2(k-1)\\&+2\Bigg(d(k)\left(1+\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{J1}}\right)\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{J1}}\\&+\gamma(k)\left(1+\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{J1}}\right)\Bigg)\beta^{\frac{k}{2}}e(k-1)\\&+\left(\gamma(K)+d(k)\left(\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{J1}}\right)\right)^2\beta^k. \quad (3\text{-}14)\end{aligned}$$

Also, considering the facts that $\beta^{k-1} \gg 1$ when k is large enough, and $(w_{1,j1}^{(2,1)}(k)/(w_{1,j1}^{(2,1)}(k)+\lambda_{J1})) \ll 1$ when $w_{1,j1}^{(2,1)}(k)$ is near or equal to zero, (3-14) can be approximated as follows:

$$\begin{aligned}\Delta V(k)\approx&-\beta^{k-1}e^2(k-1)+2\left(d(k)+\gamma(k)\right)\beta^{\frac{k}{2}}e(k-1)\\&+\left(\gamma(k)+d(k)\left(\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{J1}}\right)\right)^2\beta^k. \quad (3\text{-}15)\end{aligned}$$

Two roots of $\Delta V(k)=0$ are then obtained as shown in (3-16) at the bottom of the page. When the value of k is large enough, (3-16) can be approximated by

$$r_{1,2}=\pm\sqrt{\beta}\left|\gamma(k)+d(k)\left(\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\right)\right|. \quad (3\text{-}17)$$

Therefore, the error will converge to the ball, centered at the system origin, with the radius in (3-3).

Remark 3.1: When we discuss the error convergence region in Theorem 3.1, the following properties of the nonlinear function $F_j(\bullet)$ in (2-2) have been used:

- $$F_j(x+\Delta x)=F_j(x)+\kappa\left(F_j(\Delta x)-F_j(0)\right), \quad \text{with} \quad 0<\kappa\le 1; \quad (3\text{-}18)$$
- $$F_j(\Delta x)\approx F_j(0)+F_j'(0)\Delta x, \quad \text{if} \quad \Delta x \ll 1. \quad (3\text{-}19)$$

Remark 3.2: Using the similar methods, we can easily modify the update-law in (2-5) and discuss the error convergence for singularity cases: 1) and 2). The results are briefly summarized in the following lemmas.

$$\begin{aligned}\Delta V(k)=&\beta^k\Bigg(\sum_{\substack{j=1\\ j\neq j1}}^{n}w_{1,j}^{(2,1)}(k)\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{(n-1)w_{1,j}^{(2,1)}(k)}+\frac{\alpha\alpha_3}{4}\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}\sum_{\substack{i=1\\ i\neq i1}}^{n}w_{j,i}^{(1,0)}(k)x_i(k)\right)\\&+w_{1,j1}^{(2,1)}(k)\left(\frac{\beta^{-\frac{k}{2}}e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}+\frac{\alpha\alpha_4}{4}\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}\sum_{\substack{i=1\\ i\neq i1}}^{n}w_{j1,i}^{(1,0)}(k)x_i(k)\right)-d(k)\Bigg)^2-\beta^{k-1}e^2(k-1)\\=&\beta^k\Bigg(\beta^{-\frac{k}{2}}e(k-1)+\frac{\alpha\alpha_3}{4}\frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}\sum_{\substack{j=1\\ j\neq j1}}^{n}w_{1,j}^{(2,1)}(k)\sum_{\substack{i=1\\ i\neq i1}}^{n}w_{j,i}^{(1,0)}(k)x_i(k)+\frac{w_{1,j1}^{(2,1)}(k)\Big(\beta^{-\frac{k}{2}}e(k-1)+d(k)\Big)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}}\\&+\frac{\alpha\alpha_4}{4}\frac{w_{1,j1}^{(2,1)}(k)x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}}\sum_{\substack{i=1\\ i\neq i1}}^{n}w_{j1,i}^{(1,0)}(k)x_i(k)\Bigg)^2-\beta^{k-1}e^2(k-1) \quad (3\text{-}13)\end{aligned}$$

$$r_{1,2}=\frac{\left(d(k)+\gamma(k)\right)\beta}{\beta^{\frac{k}{2}}}\pm\beta\sqrt{\frac{\left(d(k)+\gamma(k)\right)^2}{\beta^k}+\frac{\left(d(k)\left(\frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{J1}}\right)+\gamma(k)\right)^2}{\beta}} \quad (3\text{-}16)$$

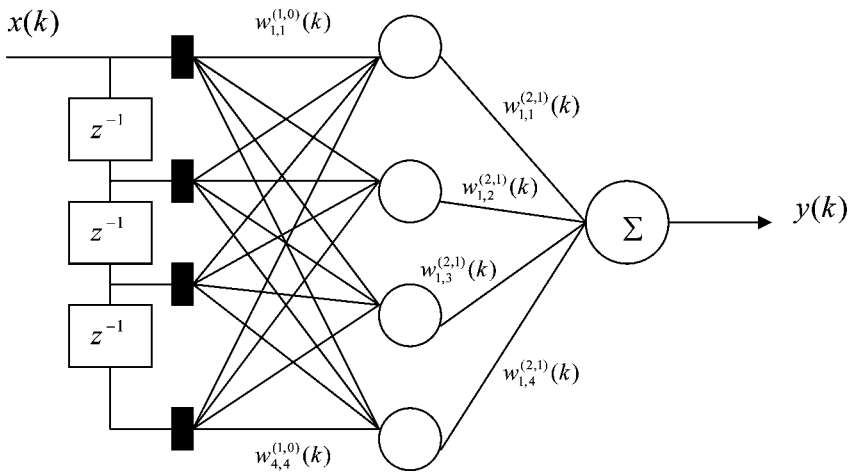


Fig. 2. Three-layered feedforward neural network.

Lemma 3.1: Consider the neural network in Fig. 1. If $x_i(k)$, for $i = i1$, is near or equal to zero at time instant k, the update-law of $w_{j,i}^{(1,0)}(k)$ in (2-5) can be modified as follows:

$$w_{i,j}^{(1,0)}(k) = \begin{cases} \frac{1}{(n-1)x_i(k)} G_j \left(\frac{\beta^{-\frac{k}{2}} e(k-1)+d(k)}{w_{1,j}^{(2,1)}(k)} \right), & \text{for } i \neq i1 \\ \frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}} G_j \left(\frac{\beta^{-\frac{k}{2}} e(k-1)+d(k)}{w_{1,j}^{(2,1)}(k)} \right), & \text{for } i = i1 \end{cases} \tag{3-20}$$

and the tracking error can then converge to the ball, centered at the system origin, with radius

$$r = \frac{\kappa\alpha\sqrt{\beta}}{4} \left| \sum_{j=1}^{n} w_{1,j}^{(2,1)}(k) G_j \left(\frac{\beta^{-\frac{k}{2}} e(k-1)+d(k)}{n w_{1,j}^{(2,1)}(k)} \right) \times \frac{x_{i1}(k)}{x_{i1}(k)+\lambda_{i1}} \right|. \tag{3-21}$$

Lemma 3.2: Consider the neural network in Fig. 1. If $w_{1,j}^{(2,1)}(k)$, for $j = j1$, is near or equal to zero at time instant k, the update-law of $w_{j,i}^{(1,0)}(k)$ in (2-5) can be modified as follows:

$$w_{i,j}^{(1,0)}(k) = \frac{1}{n x_i(k)} G_j \left(\left. \frac{\beta^{-\frac{k}{2}} e(k-1)+d(k)}{(n-1) w_{1,j}^{(2,1)}(k)} \right|_{j \neq j1} + \frac{\beta^{-\frac{k}{2}} e(k-1)+d(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}} \right) \tag{3-22}$$

and the tracking error can then converge to the ball, centered at the system origin, with radius

$$r = \sqrt{\beta} \left| d(k) \frac{w_{1,j1}^{(2,1)}(k)}{w_{1,j1}^{(2,1)}(k)+\lambda_{j1}} \right|. \tag{3-23}$$

Remark 3.3: In fact, it is not convenient to use the modified parameter update-laws discussed in the above to handle singularities. For practical application, we may use the following modified update-laws to compute $w_{i,j}^{(2,1)}(k)$ and $w_{j,i}^{(1,0)}(k)$ to handle all singularities:

$$w_{1,j}^{(2,1)}(k) = \frac{\beta^{-\frac{k}{2}} e(k-1)+d(k)}{n S_j(k-1)+\lambda_1} \tag{3-24}$$

$$w_{j,i}^{(1,0)}(k) = \frac{1}{n x_i(k)+\lambda_2} G_j \left(\frac{\beta^{-\frac{k}{2}} e(k-1)+d(k)}{n w_{1,j}^{(2,1)}(k)+\lambda_3} \right) \tag{3-25}$$

where λ_1, λ_2, and λ_3 are small positive numbers.

It is easy to see that (3-24) is the approximation of (3-1) when $S_j(k-1)$ is near zero, and (3-25) is the approximation of (3-2), (3-20), and (3-22) when the system got singularities in cases 1)–3). It will also be seen from the following simulation section that, if λ_1, λ_2, and λ_3 in (3-24) and (3-25) are small enough, the good tracking performance and robustness with respect to bounded input disturbances can be obtained. In addition, the authors have estimated in the simulations that the computational complexity of the proposed new Lyapunov stability-based BP algorithm and the one of the conventional BP algorithms in [1]–[9] are at the same level, and the new algorithm can be easily implemented for onlinear learning.

Remark 3.4: It is noted that the analysis and design of the proposed new adaptive BP algorithm in Sections II and III are carried out for the neural networks with log-sigmoid nonlinear activation function. However, the algorithm can also be implemented for neural networks with tan-sigmoid nonlinear function.

IV. Simulation Example

In this section, we consider a three-layered neural adaptive filter, as seen in Fig. 2, for the purpose of evaluating the proposed new Lyapunov adaptive BP algorithm.

The desired reference signal $d(k)$ and the input signal $x(k)$ are given in Fig. 3 with $x(k) = d(k) + 0.1^{*}\text{randn}(1)$. The weights $w_{1,j}^{(2,1)}(k)$ and $w_{j,i}^{(1,0)}(k)$ are initialized randomly with $w_{1,j}^{(2,1)}(0) = 0.1^{*}\text{randn}(1)$ and $w_{j,i}^{(1,0)}(0) = 0.1^{*}\text{randn}(1)$. The value of parameter β is chosen as 1.2. Although, theoretically, it is all right for β to be greater than one in order to guarantee

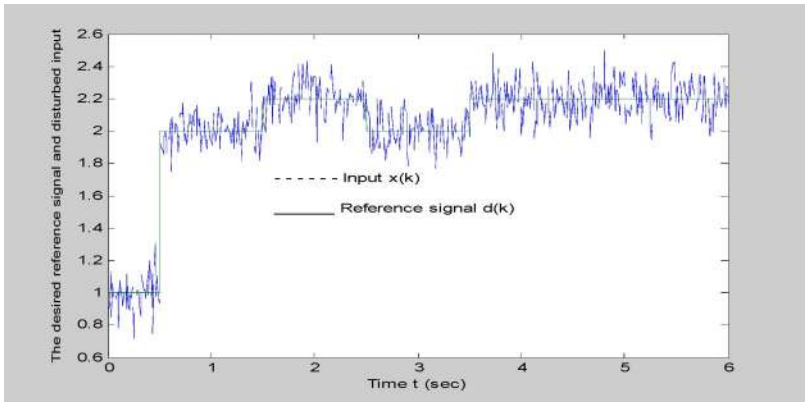
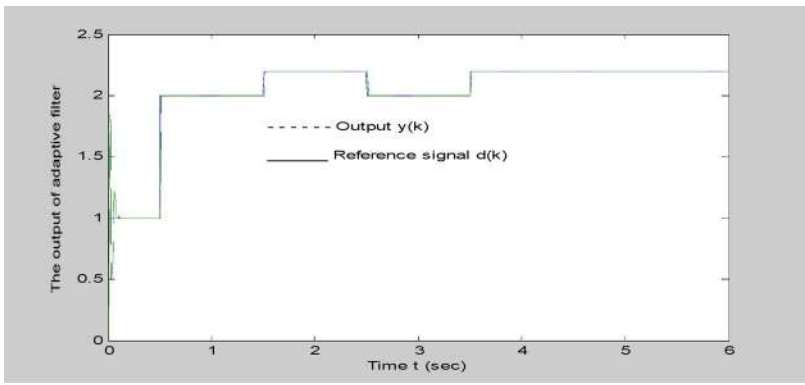
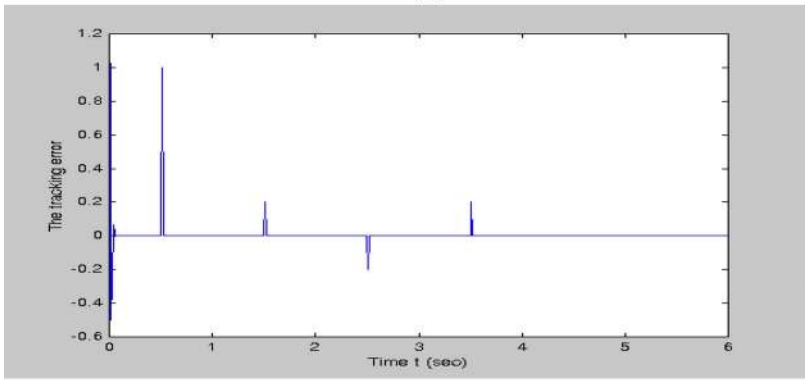


Fig. 3. Desired reference signal $d(k)$ and the input signal $x(k)$.



(a)



(b)

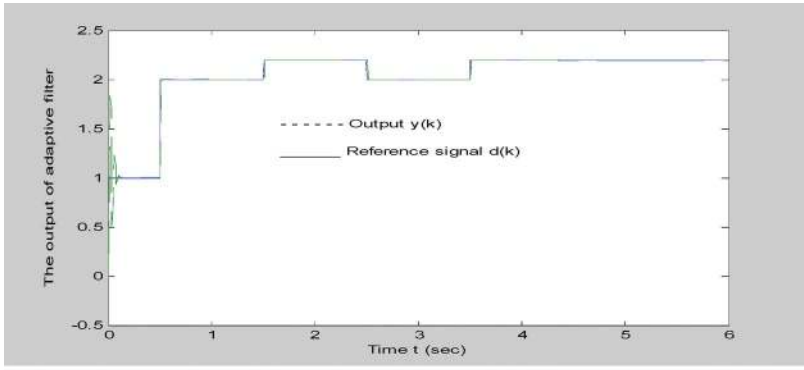
Fig. 4. (a) Output tracking of the neural adaptive filter and (b) tracking error of the neural adaptive filter using Lyapunov BP algorithm.

the asymptotic error convergence, if β is closed to 1, the error convergence is very slow. If β is much greater than 1, the error convergence will be very fast. This will require that the designed adaptive filter should have a wide frequency band or a very fast response.

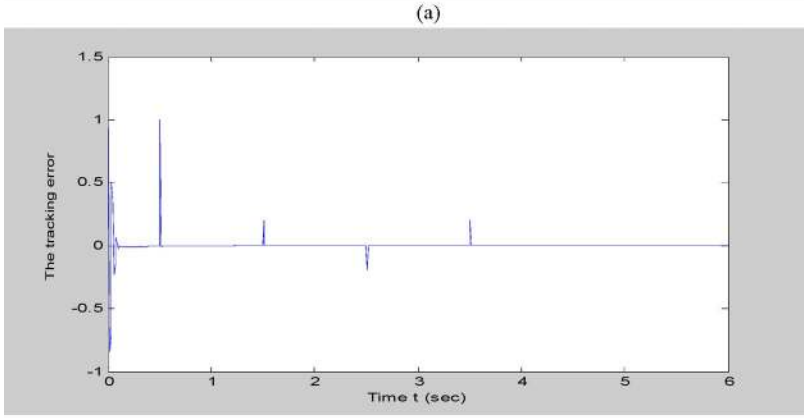
Fig. 4 shows the output of the neural adaptive filter and the tracking error using the new Lyapunov BP training algorithm, where the weights are updated according to the fundamental update-laws in (2-4) and (2-5). It is seen that the effects of the large bounded input noise have been eliminated, and the output of the neural adaptive filter can track the desired reference signal very well.

Fig. 5 shows the output and the tracking error with the weights updated using the modified update-laws in (3-24) and (3-25) with $\lambda_1 = 0.01$, $\lambda_2 = 0.01$, and $\lambda_3 = 0.01$. It is seen that the tracking performance is also very good. Fig. 6 shows the output and the tracking error with the weights updated using the modified update-laws in (3-24) and (3-25) with $\lambda_1 = 0.03$, $\lambda_2 = 0.03$, and $\lambda_3 = 0.03$. Because the values of λ_1, λ_2, and λ_3 are relatively large, the steady-state tracking error exists. Therefore, the values of λ_1, λ_2, and λ_3 must be chosen properly in practice.

For further comparisons, Fig. 7 shows the output of the neural adaptive filter and the tracking error using the conventional gradient descent BP training algorithm with step size $\lambda = 0.25$. Obviously, the robustness property with respect to the bounded input noise is no good, and the tracking performance between the filter output and the desired reference signal is very poor.

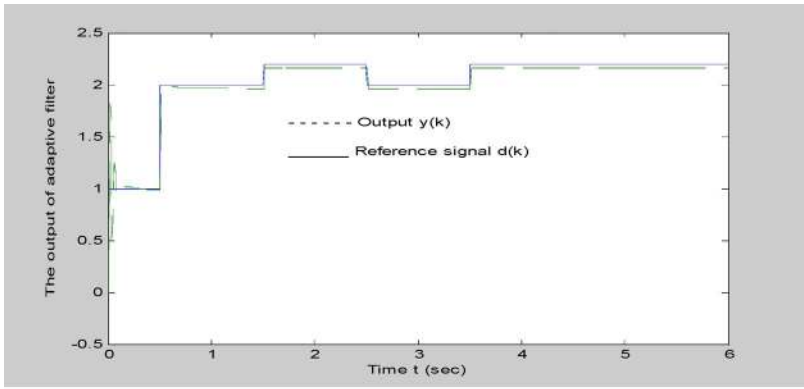


(a)

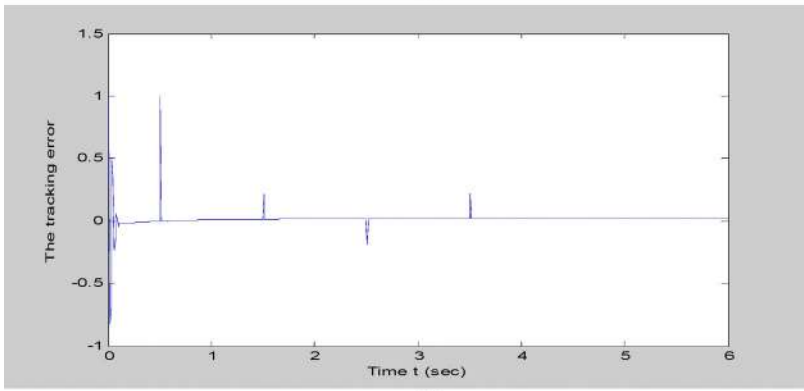


(b)

Fig. 5. (a) Output tracking of the neural adaptive filter and (b) tracking error of the neural adaptive filter using Lyapunov BP algorithm with the modified update-laws in (3-24) and (3-25) ($\lambda_1 = 0.01$, $\lambda_2 = 0.01$, and $\lambda_3 = 0.01$).

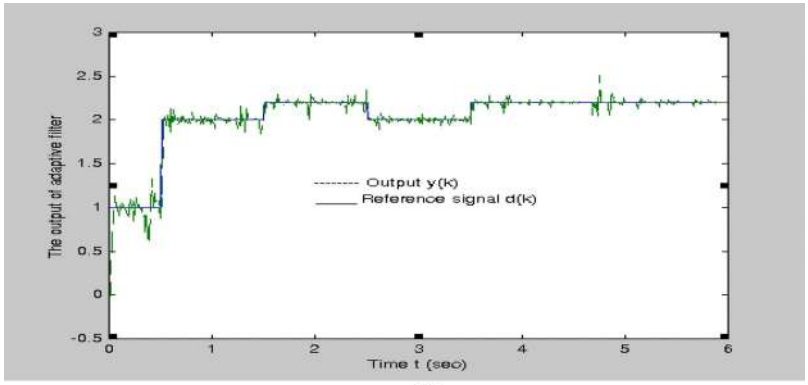


(a)

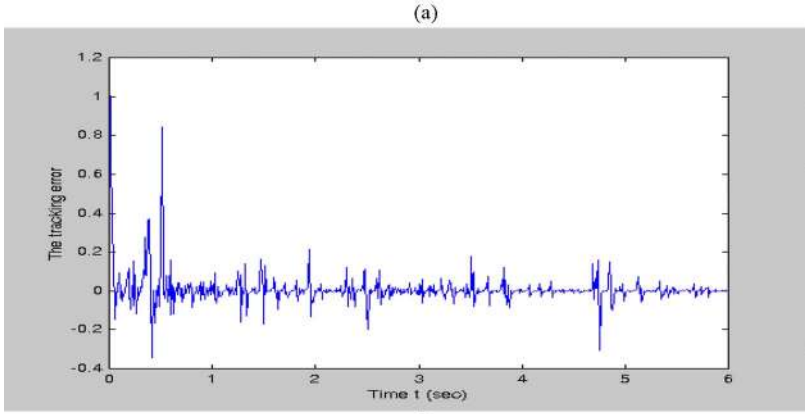


(b)

Fig. 6. (a) Output tracking of the neural adaptive filter and (b) tracking error of the neural adaptive filter using Lyapunov BP algorithm with the modified update-laws in (3-24) and (3-25) ($\lambda_1 = 0.03$, $\lambda_2 = 0.03$, and $\lambda_3 = 0.03$).

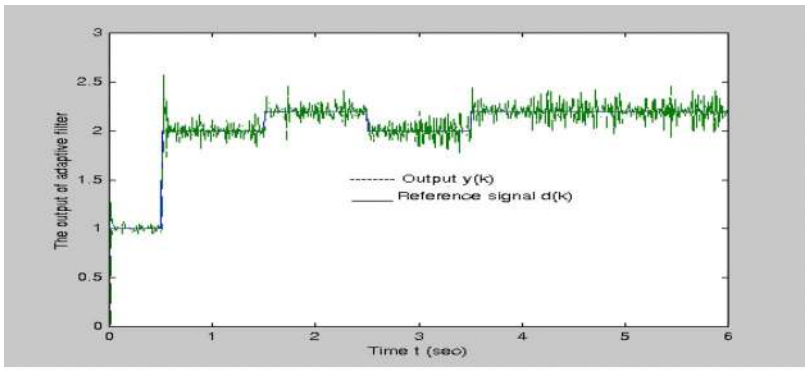


(a)

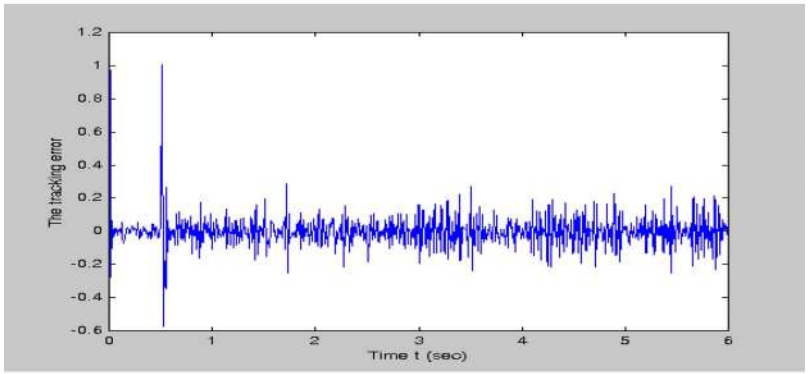


(b)

Fig. 7. (a) Output tracking of the neural adaptive filter and (b) tracking error of the neural adaptive filter using the gradient descent BP algorithm.



(a)



(b)

Fig. 8. (a) Output tracking of a FIR adaptive filter and (b) tracking error of a FIR adaptive filter using the normalized LMS algorithm.

Also, Fig. 8 shows the simulation results of a FIR adaptive filter $y(k) = \sum_{i=0}^{3} w_i(k)x(k-i)$ trained using the normalized least mean squares (LMS) searching scheme. Fig. 9 shows the simulations of the same FIR adaptive filter trained using

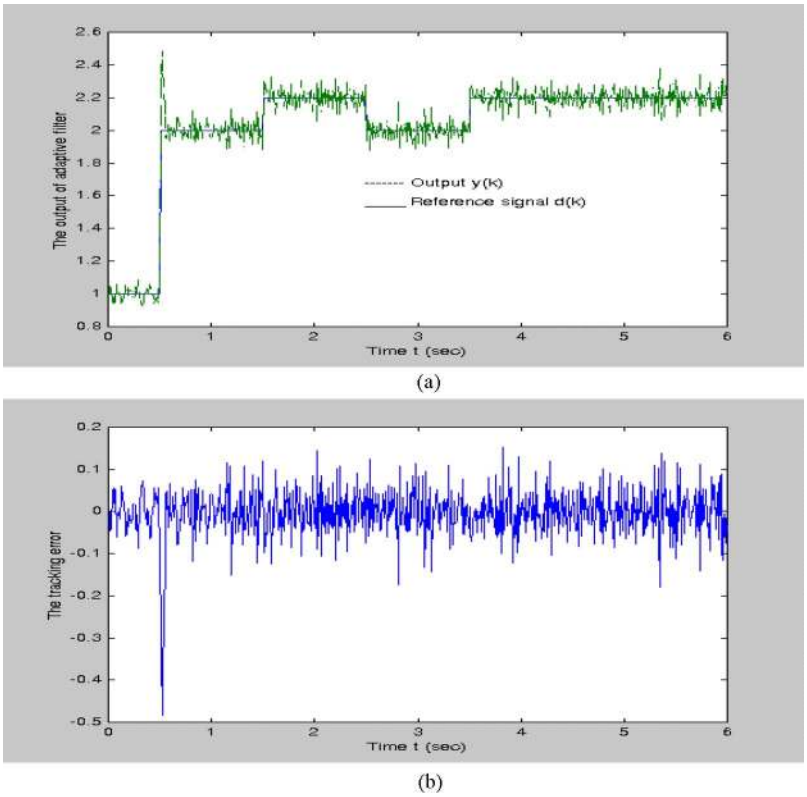


Fig. 9. (a) Output tracking of a FIR adaptive filter and (b) tracking error of a FIR adaptive filter using the RLS algorithm.

the recursive least squares (RLS) scheme. It is clearly seen that the tracking performance and robustness with respect to the bounded noise are very poor.

Therefore, the neural adaptive filter with the new Lyapunov BP algorithm has demonstrated a very good performance with the fast error convergence the strong robustness property with respect to the large bounded input disturbances.

V. Conclusion

In this paper, we have developed a new adaptive BP algorithm using Lyapunov stability theory for three-layered feedforward neural networks. The error convergences have been explored in detail. The new Lyapunov adaptive BP algorithm has been applied to the design of a neural adaptive filter in the simulation section. The excellent robustness property with respect to large bounded input disturbances and fast error convergence have been shown in the simulation example. The proposed new adaptive BP algorithm can be easily expended to general multilayered neural networks. The research on the optimization using Lyapunov stability theory in this paper is just at its initial stage, and many further work in this area are under authors' investigation.

Acknowledgment

The authors would like to thank the Editor-in-Chief, the Associate Editor, and the anonymous reviewers for their valuable comments and suggestions on this paper.

References

[1] J. C. Principe, N. R. Euliano, and W. C. Lefebvre, *Neural and Adaptive Systems*. New York: Wiley, 2000.

[2] J. Hertz, A. Krough, and R. G. Palmer, *Introduction to the Theory of Neural Computation*. Reading, MA: Addison-Wesley, 1991.

[3] S. Haykin, *Neural Network: A Comprehensive Foundation*. Piscataway, NJ: IEEE Press, 1994.

[4] E. A. Wan, "Temporal backpropagation for FIR neural networks," in *Proc. Int. Joint Conf. Neural Netw.*, 1990, vol. 1, pp. 577–580.

[5] F. Beaufays and E. Wan, "Relating real-time backpropagation and backpropagation through time: an application of flow graph interreciprocity," *Neural Comput.*, vol. 6, pp. 296–306, 1994.

[6] K. S. Narendra and K. Parthasarathy, "Gradient methods for the optimization of dynamical systems containing neural networks," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 252–262, Mar. 1991.

[7] R. Battiti, "First and second order methods for learning: between steepest descent and Newton's method," *Neural Comput.*, vol. 4, pp. 141–166, 1992.

[8] G. W. Ng, *Application of Neural Networks to Adaptive Control of Nonlinear Systems*, ser. Control Systems Centre Series. New York: Wiley, 1997.

[9] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamics systems using neural networks," *IEEE Trans. Neural Netw.*, vol. 1, no. 1, pp. 4–27, Mar. 1990.

[10] H. Sira-Ramirez and E. Colina-Morles, "A sliding mode strategy for adaptive learning in adalines," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 42, no. 12, pp. 1001–1012, Dec. 1995.

[11] G. G. Parma, B. R. Menezes, and A. P. Braga, "Sliding mode algorithm for training multilayer artificial neural networks," *Electron. Lett.*, vol. 34, no. 1, pp. 97–98, 1998.

[12] J. A. K. Suykens, J. Vandewalle, and B. L. R. De Moor, "NLp theory: checking and improving stability of recurrent neural networks for nonlinear modeling," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2682–2691, Nov. 1997.

[13] O. Kaynak, K. Erbatur, and M. Ertugrul, "The fusion of computationally intelligent methodologies and sliding-mode control—a survey," *IEEE Trans. Ind. Electron.*, vol. 48, no. 1, pp. 4–17, Feb. 2001.

[14] E. N. Poznyak, E. N. Sanchez, and W. Yu, *Differential Neural Networks for Robust Nonlinear Control-Identification, State Estimation and Trajectory Tracking*. Singapore: World Scientific, 2001.

[15] J. T. Spooner, M. Maggiore, R. Ordonez, and K. M. Passino, *Stable Adaptive Control and Estimation for Nonlinear Systems: Neural and Fuzzy Approximator Techniques*. New York: Wiley, 2002.

[16] J.-J. E. Slotine and W. Li, *Applied Nonlinear Control*. Englewood Cliffs, NJ: Prentice-Hall, 1991.
[17] Z. Man, H. R. Wu, W. Lai, and T. Nguyen, "Design of adaptive filters using Lyapunov stability theory," in *Proc. 6th IEEE Int. Workshop Intell. Signal Process. Commun. Syst.*, 1998, pp. 304–308.
[18] K. P. Seng, Z. Man, and H. R. Wu, "Lyapunov theory-based radial basis function networks for adaptive filtering," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 49, no. 8, pp. 1215–1220, Aug. 2002.
[19] T. Hayakawa, W. M. Haddad, N. Hovakimyan, and V. Chellaboina, "Neural network adaptive control for nonlinear nonnegative dynamical systems," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 399–413, Mar. 2005.
[20] J. Zhang, S. S. Ge, and T. H. Lee, "Output feedback control of a class of discrete MIMO nonlinear systems with triangular form inputs," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1491–1503, Nov. 2005.
[21] F. Abdollahi, H. A. Talebi, and R. V. Patel, "A stable neural network-based observer with application to flexible-joint manipulators," *IEEE Trans. Neural Netw.*, vol. 17, no. 1, pp. 118–129, Jan. 2006.
[22] C. Wang and D. J. Hill, "Learning from neural control," *IEEE Trans. Neural Netw.*, vol. 17, no. 1, pp. 130–146, Jan. 2006.
[23] S. N. Huang, K. K. Tan, and T. H. Lee, "Nonlinear adaptive control of interconnected systems using neural networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 1, pp. 243–246, Jan. 2006.

Zhihong Man received the B.E. degree from Shanghai Jiaotong University, Shanghai, China, the M.S. degree from Chinese Academy of Sciences, Beijing, China, and the Ph.D. degree from The University of Melbourne, Melbourne, Australia, all in electrical and electronic engineering, in 1982, 1996, and 1993, respectively.

From 1994 to 1996, he was a Lecturer in the Department of Computer and Communication Engineering, Edith Cowan University, Perth, Australia. From 1996 to 2001, he was Lecturer and then Senior Lecturer in the School of Engineering, The University of Tasmania, Australia. In 2001, he was Visiting Senior Fellow in the School of Computer Engineering, Nanyang Technological University (NTU), Singapore. Since January 2002, he has been Associate Professor of Computer Engineering at NTU. His research interests are in neural networks, fuzzy systems, time varying systems, nonlinear systems, signal processing, robotics, intelligent control, and transmission control protocol (TCP) congestion control. He has published more than 120 journal and conference papers in these areas.

Dr. Man received the NTU Best Teacher Award in 2004. In addition, he was the Most Popular Lecturer in the School of Computer Engineering at NTU, from 2002 to 2006.

Hong Ren Wu received the B.E. and M. E. degrees from the University of Science and Technology, Beijing, China, in 1982 and 1985, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Wollongong, Wollongong, Australia, in 1990.

From 1982 to 1985, he worked as a Teaching Fellow in the Department of Industrial Automation, University of Science and Technology, Beijing, China. From 1990 to 2005, he was with the School of Computer Science and Software Engineering, Monash University, Victoria, Australia, where he was Lecturer, Senior Lecturer, and then Associate Professor. Since February 2005, he has been with the School of Electrical and Computer Engineering, RMIT University, Victoria, Australia, where he currently holds the positions of Professor of Visual Communication Engineering and the Discipline Head of Software and Networks. Since 1987 he has authored and coauthored over 150 papers in refereed international journals and conferences as well as commercial-inconfidence R&D reports in areas of digital signal processing and fast algorithms, image processing, video coding and compression, circuit and systems, automatic control, and digital signal processing (DSP) education. He has also participated in, and headed, numerous research projects and industrial contracts in signal processing and digital video coding, compression, and transmission. His current research interests include DSP and fast algorithms, image processing, audio/image/video coding and compression, variable bit rate video coding and transmission for ATM networks, multimedia signal processing and communications, DSP industrial applications, and fast digital signal processors, hardware and systems.

Sophie Liu received the B.S. degree from Sichun University, Chendu, China, the M.E. degree from Xidian University, Shaanxi, China, and the Ph.D. degree from National University of Singapore, Singapore, all in electrical and electronic engineering, in 1982, 1991, and 1996, respectively.

From 1996 to 2000, she was a Lecturer in Temesak Polytechnic of Singapore, Singapore. From 2001 to 2002, she was an Associate Professor in the Department of Engineering and Physics, Oral Roberts University, Tulsa, OK. From 2002 to 2004, she was an Assistant Professor in the School of Computer Engineering, Nanyang Technological University, Singapore. In August 2004, she joined in the Department of Engineering and Physics, Oral Roberts University again, as an Associate Professor. Her current research interests are in image processing, video processing, digital signal processing, and filter design and modeling.

Xinghuo Yu (M'92–SM'97) received the B.Eng. and M.Eng. degrees from the University of Science and Technology of China, Beijing, China, in 1982 and 1984, respectively, and the Ph.D. degree from South-East University, Nanjing, China, in 1988.

He is now with Royal Melbourne Institute of Technology University, Melbourne, Australia, where he is the Associate Dean (Research and Innovation) of Science, Engineering and Technology Portfolio, and the Professor of Information Systems Engineering. He has held a Visiting Professor position in City University of Hong Kong, Hong Kong, and Bogazici University, Istanbul, Turkey, as well as a Guest Professor position in six Chinese universities. His research interests include variable structure and nonlinear control, complex, and intelligent systems. He has published over 300 refereed papers in technical journals, books, and conference proceedings as well as coedited nine research books.

Prof. Yu has served as the Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS PART I (2001–2004) and IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS (2005–Present) and several other scholarly journals. He has been on the program committees of more than 50 international conferences, and cochaired several international conferences. He was the sole recipient of the 1995 Central Queensland University (CQU) Vice Chancellor's Award for Research. He is a Fellow of Institution of Engineers Australia. He was made Emeritus Professor of CQU Australia in recognition of his significant contributions to the university.