

Available online at www.sciencedirect.com**ScienceDirect**

Fuzzy Sets and Systems ●● (●●●●) ●●●—●●●

FUZZY
sets and systemswww.elsevier.com/locate/fss

An online gradient-based parameter identification algorithm for the neuro-fuzzy systems

Long Li ^{a,*}, Zuqiang Long ^b, Hao Ying ^c, Zhijun Qiao ^d^a College of Mathematics and Statistics, Hengyang Normal University, Hengyang, Hunan, China^b College of Physics and Electronic Engineering, Hengyang Normal University, Hengyang, Hunan, China^c Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI, USA^d Department of Mathematics, University of Texas Rio Grande Valley, Edinburg, TX 78539, USA

Received 23 January 2019; received in revised form 4 November 2020; accepted 5 November 2020

Abstract

Online gradient descent method has been widely applied for parameter learning in neuro-fuzzy systems. The success of the application relies on the convergence of the learning procedure. However, there barely have been convergence analyses on the online learning procedure for neuro-fuzzy systems. In this paper, an online gradient learning algorithm with adaptive learning rate is proposed to identify the parameters of the neuro-fuzzy systems representing the Mamdani fuzzy model with Gaussian fuzzy sets. We take the reciprocals of the variances of the Gaussian membership functions, rather than the variances themselves, as independent variables when computing the gradient with respect to the variance parameters. Subsequently, oscillation of the gradient value in the learning process can be avoided. Furthermore, some convergence results for this online learning scheme are studied. Finally, three numerical examples are provided to illustrate the performance of the proposed algorithm.

© 2020 Elsevier B.V. All rights reserved.

Keywords: Mamdani fuzzy model; Neuro-fuzzy systems; Online gradient learning algorithm; Adaptive learning rate; Convergence

1. Introduction

Neuro-fuzzy systems have been investigated widely, which combine the human-like reasoning style of fuzzy systems with the learning and connectionist structure of neural networks [9,10,15,21,24–26]. Due to the interpretability of fuzzy rules, the merits of universal approximation and the learning capability, neuro-fuzzy systems work well in many practical applications [3,5–7,11–13,20]. Actually, the learning algorithm is a fundamental element in the application of a neuro-fuzzy system. An important learning scheme is the parameter learning algorithm to fine-tune the membership functions and other parameters. Neuro-fuzzy systems need to perform parametric tuning optimally, e.g. tuning parameters of the membership functions and tuning the weights of the fuzzy rules using neural network learn-

* Corresponding author.

E-mail address: longli@hynu.edu.cn (L. Li).

<https://doi.org/10.1016/j.fss.2020.11.003>

0165-0114/© 2020 Elsevier B.V. All rights reserved.

ing techniques. Therefore, many algorithms have already been developed in recent years. Among these algorithms, gradient descent-based learning algorithm is one of the most frequently used techniques to adjust the parameters to match the desired output in neuro-fuzzy systems [1,4,13,14,18,22,23]. There are two essential training schemes for gradient descent learning: batch scheme and on-line scheme. The batch scheme corresponds to the standard gradient iteration procedure which updates the weights after all the training examples are processed. The exact gradient is used to determine the direction of the next update. It is a deterministic optimization algorithm. Differently, the online scheme is the procedure of updating network weights immediately after one training example is fed. The fed example may be randomly or circularly selected from the given training examples, but should keep periodic in the training set. The online scheme has been proven to be more effective and sometimes unique choice in application [28], especially when the given training examples are huge. Therefore, convergence of the online training procedure is a prerequisite of any successful application of neuro-fuzzy systems.

There have been many convergence results of the gradient based learning procedures, both batch and online schemes, for neural networks [29,31–33]. But there barely have been convergence analyses on learning procedures for the neuro-fuzzy systems. Among the few related works, the convergence of batch gradient learning scheme with constant learning rate for the neuro-fuzzy systems describing Takagi-Sugeno model are discussed in [16,17,30], and the convergence of an adaptive gradient algorithm are considered by using the Lyapunov stability theorem in [8], where the learning rates must be chosen to satisfy some strict conditions and a posterior assumption on the estimation of the higher order terms of the remainder of the Taylor series expansion for error function is required.

In this paper, we are focused on the online gradient learning algorithm for the neuro-fuzzy systems representing the Mamdani fuzzy model. An online gradient learning algorithm with adaptive learning rate (OGLA) is proposed for neuro-fuzzy systems. We take the reciprocals of the variances of the Gaussian membership functions, rather than the variances themselves, as independent variables when computing the gradient with respect to the width parameters. Hence, the differentiation with respect to the denominator which may result in oscillation in the learning process can be avoided. Moreover, a comprehensive study on the convergence of OGLA is made, indicating that the error function tends to a constant, the gradient of the error function goes to zero, and the weight sequence converges to a fixed point, respectively.

The rest of this paper is organized as follows. The neuro-fuzzy systems are described in the next section. Our proposed algorithm and its convergence analysis are presented in Section 3. Section 4 provides three supporting numerical examples. Some brief conclusions are drawn in Section 5.

2. The neuro-fuzzy systems

Assume a Mamdani fuzzy system with p inputs $\mathbf{x} = (x_1, x_2, \dots, x_p) \in \mathbb{R}^p$, a single output y , and n fuzzy if-then rules in the form

$$\text{Rule } i: \text{ IF } x_1 \text{ is } A_{1i} \text{ and } x_2 \text{ is } A_{2i} \text{ and } \dots \text{ and } x_p \text{ is } A_{pi} \text{ THEN } y \text{ is } B_i. \quad (1)$$

Each rule has its own unique consequent fuzzy set B_i characterized by a singleton membership function located at $y = u_i$, $i = 1, 2, \dots, n$, which is to be determined. A_{li} , ($l = 1, 2, \dots, p$, $i = 1, 2, \dots, n$) are labels of fuzzy sets and all the membership functions used are Gaussian membership functions defined as:

$$\mu_{A_{li}}(x) = e^{-\frac{(x-c_{li})^2}{\sigma_{li}^2}} \quad (2)$$

where c_{li} and σ_{li} are two parameters, mean and variance, respectively. The architecture of the neuro-fuzzy systems is shown in Fig. 1. It is a four-layer feedforward network with p input nodes and one output node.

The I/O relationship of the neuro-fuzzy systems is described by

$$y = \sum_{i=1}^n u_i h_i(\mathbf{x}) = \sum_{i=1}^n u_i \left(\prod_{l=1}^p \mu_{A_{li}}(x_l) \right) = \sum_{i=1}^n u_i e^{-\sum_{l=1}^p \frac{(x_l-c_{li})^2}{\sigma_{li}^2}} \quad (3)$$

where $h_i(\mathbf{x}) = \prod_{l=1}^p \mu_{A_{li}}(x_l)$. The learnable parameters in the neuro-fuzzy system are the premise parameters, c_{li} s and σ_{li} s, and the consequence parameters, u_i s.

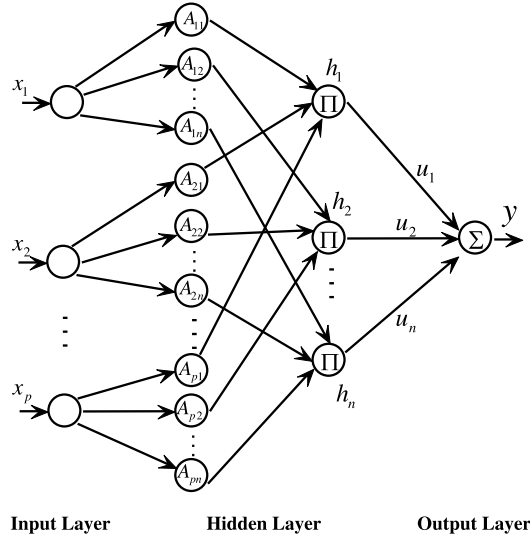


Fig. 1. Architecture of the neuro-fuzzy systems.

Suppose that $\{\mathbf{x}^j, o^j\}_{j=0}^{J-1} \subset \mathbb{R}^p \times \mathbb{R}$ is a given set of the training samples, where \mathbf{x}^j and o^j are the input and the corresponding desired output of the j -th sample, respectively. Let $\mathbf{u} = (u_1, u_2, \dots, u_n)$, $\mathbf{c}_i = (c_{1i}, c_{2i}, \dots, c_{pi})$ and $\boldsymbol{\sigma}_i = (\sigma_{1i}, \sigma_{2i}, \dots, \sigma_{pi})$, for $i = 1, 2, \dots, n$. To simplify the presentation, we write all the weight parameters in a compact form

$$\mathbf{w} = (\mathbf{u}, \mathbf{c}_1, \dots, \mathbf{c}_n, \boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n) \in \mathbb{R}^{(2p+1)n}.$$

The error function corresponding to all training samples is defined as

$$E(\mathbf{w}) = \frac{1}{2} \sum_{j=0}^{J-1} (y^j - o^j)^2 = \frac{1}{2} \sum_{j=0}^{J-1} (\mathbf{u} \cdot \mathbf{h}^j - o^j)^2 = \sum_{j=0}^{J-1} E_j(\mathbf{u} \cdot \mathbf{h}^j), \quad (4)$$

where $\mathbf{h}^j = (h_1(\mathbf{x}^j), h_2(\mathbf{x}^j), \dots, h_n(\mathbf{x}^j))^T$ and $E_j(t) = \frac{1}{2}(t - o^j)^2$. The aim of system learning is to update the parameters to minimize $E(\mathbf{w})$. Online gradient method is widely used to construct a parameter update procedure to optimize this problem. This method is a variation of the standard gradient method (also called batch learning scheme), where the parameters are updated after each training sample is processed. It is obvious that the gradient should be based on $E_j(\mathbf{u} \cdot \mathbf{h}^j)$ instead of $E(\mathbf{w})$ in (4). We can easily calculate the gradients of $E_j(\mathbf{u} \cdot \mathbf{h}^j)$ with respect to \mathbf{u} , \mathbf{c}_i and $\boldsymbol{\sigma}_i$ as follows:

$$\begin{aligned} \nabla_{E_j, \mathbf{u}}(\mathbf{u} \cdot \mathbf{h}^j) &= E'_j(\mathbf{u} \cdot \mathbf{h}^j) \mathbf{h}^j \\ \nabla_{E_j, \mathbf{c}_i}(\mathbf{u} \cdot \mathbf{h}^j) &= \left(\frac{\partial E_j(\mathbf{u} \cdot \mathbf{h}^j)}{\partial c_{1i}}, \frac{\partial E_j(\mathbf{u} \cdot \mathbf{h}^j)}{\partial c_{2i}}, \dots, \frac{\partial E_j(\mathbf{u} \cdot \mathbf{h}^j)}{\partial c_{pi}} \right)^T \\ \nabla_{E_j, \boldsymbol{\sigma}_i}(\mathbf{u} \cdot \mathbf{h}^j) &= \left(\frac{\partial E_j(\mathbf{u} \cdot \mathbf{h}^j)}{\partial \sigma_{1i}}, \frac{\partial E_j(\mathbf{u} \cdot \mathbf{h}^j)}{\partial \sigma_{2i}}, \frac{\partial E_j(\mathbf{u} \cdot \mathbf{h}^j)}{\partial \sigma_{pi}} \right)^T \end{aligned}$$

where

$$\frac{\partial E(\mathbf{u} \cdot \mathbf{h}^j)}{\partial c_{li}} = 2E'_j(\mathbf{u} \cdot \mathbf{h}^j) u_i h_i^j \left(\frac{x_l^j - c_{li}}{\sigma_{li}^2} \right) \quad (5)$$

and

$$\frac{\partial E(\mathbf{u} \cdot \mathbf{h}^j)}{\partial \sigma_{li}} = 2E'_j(\mathbf{u} \cdot \mathbf{h}^j) u_i h_i^j \frac{(x_l^j - c_{li})^2}{\sigma_{li}^3} \quad (6)$$

for $i = 1, 2, \dots, n$, $l = 1, 2, \dots, p$. Hence, started from an arbitrary initial guess \mathbf{w}^0 , the original online gradient learning algorithm (OGLA0) can be formulated as the following iteration procedure:

$$\begin{aligned} \mathbf{u}^{mJ+j+1} &= \mathbf{u}^{mJ+j} - \eta_m \nabla E_{j,\mathbf{u}}(\mathbf{u} \cdot \mathbf{h}^j) \\ \mathbf{c}_i^{mJ+j+1} &= \mathbf{c}_i^{mJ+j} - \eta_m \nabla E_{j,\mathbf{c}_i}(\mathbf{u} \cdot \mathbf{h}^j) \\ \sigma_i^{mJ+j+1} &= \sigma_i^{mJ+j} - \eta_m \nabla E_{j,\sigma_i}(\mathbf{u} \cdot \mathbf{h}^j) \end{aligned}$$

where for $m \in \mathbb{N}$; $i = 1, 2, \dots, n$; $j = 0, 1, \dots, J - 1$.

3. Our proposed algorithm and its convergence analysis

3.1. Proposed online gradient learning algorithm

From the expressions of partial derivatives (5) and (6), one can see that they include two factors, $\frac{1}{\sigma_{li}^2}$ and $\frac{1}{\sigma_{li}^3}$, respectively, which may result in oscillation of the parameter sequence in the iteration process due to small σ_{li} . The reason is discussed in our previous work (cf. [30]). This observation is also shown in Example 1 in the numerical examples section. Hence, we make a modification for the original online gradient learning algorithm (OGLA0) and set $b_{li} = \frac{1}{\sigma_{li}}$. In place of (2) we have, for $l = 1, 2, \dots, p$, $i = 1, 2, \dots, n$ and $j = 0, \dots, J - 1$,

$$\mu_{A_{li}}(x_l) = e^{-(x_l - c_{li})^2 b_{li}^2} \quad (7)$$

and

$$h_i(\mathbf{x}^j) = \prod_{l=1}^p \mu_{A_{li}}(x_l^j) = e^{-\|(\mathbf{x}^j - \mathbf{c}_i) \odot \mathbf{b}_i\|^2} \quad (8)$$

where $\|\cdot\|$ stands for the Euclidean norm and “ \odot ” means Hadamard product which is a binary operation taking two vectors of the same dimensions, and producing another vector where each element is the product of the corresponding elements of the original two vectors.

We take b_{li} s, rather than σ_{li} s, as independent variable. Then we can easily calculate that

$$\begin{aligned} \nabla E_{j,\mathbf{u}}(\mathbf{u} \cdot \mathbf{h}^j) &= E'_j(\mathbf{u} \cdot \mathbf{h}^j) \mathbf{h}^j \\ \nabla E_{j,\mathbf{c}_i}(\mathbf{u} \cdot \mathbf{h}^j) &= 2E'_j(\mathbf{u} \cdot \mathbf{h}^j) u_i h_i^j \left((\mathbf{x}^j - \mathbf{c}_i) \odot \mathbf{b}_i \odot \mathbf{b}_i \right) \\ \nabla E_{j,\mathbf{b}_i}(\mathbf{u} \cdot \mathbf{h}^j) &= -2E'_j(\mathbf{u} \cdot \mathbf{h}^j) u_i h_i^j \left((\mathbf{x}^j - \mathbf{c}_i) \odot (\mathbf{x}^j - \mathbf{c}_i) \odot \mathbf{b}_i \right) \end{aligned}$$

Started from an arbitrary initial guess \mathbf{w}^0 , Our proposed online gradient learning algorithm (OGLA) can be formulated as the following iteration procedure:

$$\mathbf{u}^{mJ+j+1} = \mathbf{u}^{mJ+j} + \Delta_j \mathbf{u}^{mJ+j} \quad (9)$$

$$\mathbf{c}_i^{mJ+j+1} = \mathbf{c}_i^{mJ+j} + \Delta_j \mathbf{c}_i^{mJ+j} \quad (10)$$

$$\mathbf{b}_i^{mJ+j+1} = \mathbf{b}_i^{mJ+j} + \Delta_j \mathbf{b}_i^{mJ+j} \quad (11)$$

where for $m \in \mathbb{N}$; $i = 1, 2, \dots, n$; $j = 0, 1, \dots, J - 1$

$$\Delta_j \mathbf{u}^{mJ+j} = -\eta_m E'_j(\mathbf{u}^{mJ+j} \cdot \mathbf{h}^{mJ+j,j}) \mathbf{h}^{mJ+j,j} \quad (12)$$

$$\begin{aligned} \Delta_j \mathbf{c}_i^{mJ+j} &= -2\eta_m E'_j(\mathbf{u}^{mJ+j} \cdot \mathbf{h}^{mJ+j,j}) u_i^{mJ+j} h_i^{mJ+j,j} \times \\ &\quad \left((\mathbf{x}^j - \mathbf{c}_i^{mJ+j}) \odot \mathbf{b}_i^{mJ+j} \odot \mathbf{b}_i^{mJ+j} \right) \end{aligned} \quad (13)$$

$$\begin{aligned} \Delta_j \mathbf{b}_i^{mJ+j} &= 2\eta_m E'_j(\mathbf{u}^{mJ+j} \cdot \mathbf{h}^{mJ+j,j}) u_i^{mJ+j} h_i^{mJ+j,j} \times \\ &\quad \left((\mathbf{x}^j - \mathbf{c}_i^{mJ+j}) \odot (\mathbf{x}^j - \mathbf{c}_i^{mJ+j}) \odot \mathbf{b}_i^{mJ+j} \right) \end{aligned} \quad (14)$$

Here the parameter η_m is the adaptive learning rate, whose value may be changed after each cycle of the training procedure.

Obviously, the standard gradient of the error function, $E(\mathbf{w})$, with respect to \mathbf{w} is as follows:

$$\nabla E_{\mathbf{w}}(\mathbf{w}) = (\nabla E_{\mathbf{u}}(\mathbf{w}), \nabla E_{\mathbf{c}_1}(\mathbf{w}), \dots, \nabla E_{\mathbf{c}_n}(\mathbf{w}), \nabla E_{\mathbf{b}_1}(\mathbf{w}), \dots, \nabla E_{\mathbf{b}_n}(\mathbf{w}))$$

where $\nabla E_{\mathbf{u}}(\mathbf{w}) = \sum_{j=0}^{J-1} \nabla E_{j,\mathbf{u}}(\mathbf{u} \cdot \mathbf{h}^j)$, $\nabla E_{\mathbf{c}_i}(\mathbf{w}) = \sum_{j=0}^{J-1} \nabla E_{j,\mathbf{c}_i}(\mathbf{u} \cdot \mathbf{h}^j)$ and $\nabla E_{\mathbf{b}_i}(\mathbf{w}) = \sum_{j=0}^{J-1} \nabla E_{j,\mathbf{b}_i}(\mathbf{u} \cdot \mathbf{h}^j)$, for $i = 1, 2, \dots, n$.

In OGLA, we take the reciprocals of the variances of Gaussian membership functions, b_{li} ($l = 1, 2, \dots, p$, $i = 1, 2, \dots, n$), rather than the variances themselves, as independent variables. This strategy was firstly introduced in our previous work (cf. [30]). Due to this seemingly simple modification, the differentiation with respect to the denominator is avoided, which may result in oscillation in the learning process, and the convergence results can be obtained. Although we borrow the idea of our previous work to set the parameters, it should be noted that the proposed algorithm differs from our previous work which focused on batch learning scheme. As mentioned before, the present work focuses on the online learning scheme formulated by (9)-(11).

3.2. Convergence analysis

In this subsection, we will make a comprehensive study on the convergence for OGLA. That is, we will show that the error function $E(\mathbf{w}^m)$ approaches to a constant while the gradient of the error function $\nabla E_{\mathbf{w}}(\mathbf{w}^m)$ goes to zero, and the parameter sequence $\{\mathbf{w}^m\}$ converges to a limit point as $m \rightarrow \infty$.

Let $\Omega_0 = \{\mathbf{w} \in \Omega : \nabla E_{\mathbf{w}}(\mathbf{w}) = 0\}$ be the stationary point set of the error function $E(\mathbf{w})$, where $\Omega \in \mathbb{R}^{n(2p+1)}$ is a bounded region. Let $\Omega_{0,s} \in \mathbb{R}$ be the projection of Ω_0 onto the s -th coordinate axis, that is,

$$\Omega_{0,s} = \{w_s \in \mathbb{R} : \mathbf{w} = (w_1, \dots, w_s, \dots, w_{n(2p+1)}) \in \Omega_0\}$$

for $s = 1, 2, \dots, n(2p + 1)$. To analyze the convergence of the algorithm, we need the following assumptions.

(A1) There exists a bounded open set $\Omega \in \mathbb{R}^{n(2p+1)}$ such that $\{\mathbf{w}^m\}_{m=0}^{\infty} \subset \Omega$, that is, there exists a constant C_0 such that $\sup_{m \in \mathbb{N}} \|\mathbf{w}^m\| = C_0$;

$$(A2) \eta_m > 0, \sum_{m=0}^{\infty} \eta_m = \infty, \sum_{m=0}^{\infty} \eta_m^2 < \infty;$$

(A3) $\Omega_{0,s}$ does not contain any interior for every $s = 1, 2, \dots, n(2p + 1)$.

Assumption (A1) requires that the parameter sequence $\{\mathbf{w}^m\}$ is bounded during the iteration process as a precondition. Assumption (A2) provides a more generalized choice of the adaptive learning rate η_m to guarantee the convergence. Assumption (A3) means the stationary points set of the error function is required not to contain any interior point to prove $\{\mathbf{w}^m\}$ converges to a limit point.

We first need to establish a series of lemmas as preparation for the analysis of our convergence results. Let the parameter sequence $\{\mathbf{w}^{mJ+j}\}$ ($m \in \mathbb{N}$, $j = 0, 1, \dots, J - 1$) be generated by (9)-(11). We define the following notations:

$$R^{m,j} = \Delta_j \mathbf{u}^{mJ+j} - \Delta_j \mathbf{u}^{mJ} \tag{15}$$

$$r_i^{m,j} = \Delta_j \mathbf{c}_i^{mJ+j} - \Delta_j \mathbf{c}_i^{mJ} \tag{16}$$

$$\tilde{r}_i^{m,j} = \Delta_j \mathbf{b}_i^{mJ+j} - \Delta_j \mathbf{b}_i^{mJ} \tag{17}$$

$$d^{m,j} = \mathbf{u}^{mJ+j} - \mathbf{u}^{mJ} = \sum_{k=0}^{j-1} \Delta_k \mathbf{u}^{mJ+k} = \sum_{k=0}^{j-1} \Delta_k \mathbf{u}^{mJ} + \sum_{k=0}^{j-1} R^{m,k} \tag{18}$$

$$v_i^{m,j} = \mathbf{c}_i^{mJ+j} - \mathbf{c}_i^{mJ} = \sum_{k=0}^{j-1} \Delta_k \mathbf{c}_i^{mJ+k} = \sum_{k=0}^{j-1} \Delta_k \mathbf{c}_i^{mJ} + \sum_{k=0}^{j-1} r_i^{m,k} \tag{19}$$

$$\tilde{v}_i^{m,j} = \mathbf{b}_i^{mJ+j} - \mathbf{b}_i^{mJ} = \sum_{k=0}^{j-1} \Delta_k \mathbf{b}_i^{mJ+k} = \sum_{k=0}^{j-1} \Delta_k \mathbf{b}_i^{mJ} + \sum_{k=0}^{j-1} \tilde{r}_i^{m,k} \tag{20}$$

$$\Psi^{m,l,j} = \mathbf{h}^{mJ+l,j} - \mathbf{h}^{mJ,j}, \quad \Phi_i^{mJ+l,j} = (\mathbf{x}^j - \mathbf{c}_i^{mJ+l}) \odot \mathbf{b}_i^{mJ+l} \quad (21)$$

$m \in \mathbb{N}; j, l = 0, 1, \dots, J-1; i = 1, 2, \dots, n.$

According to (18)-(20), we can rewrite (9)-(11) as

$$\mathbf{u}^{mJ+j} = \mathbf{u}^{mJ} + \sum_{k=0}^{j-1} (\Delta_k \mathbf{u}^{mJ} + R^{m,k}) \quad (22)$$

$$\mathbf{c}_i^{mJ+j} = \mathbf{c}_i^{mJ} + \sum_{k=0}^{j-1} (\Delta_k \mathbf{c}_i^{mJ} + r_i^{m,k}) \quad (23)$$

$$\mathbf{b}_i^{mJ+j} = \mathbf{b}_i^{mJ} + \sum_{k=0}^{j-1} (\Delta_k \mathbf{b}_i^{mJ} + \tilde{r}_i^{m,k}) \quad (24)$$

For a fixed and finite set of training samples, there exists a constant C_1 such that

$$C_1 = \max_{0 \leq j \leq J-1} \{\|\mathbf{x}^j\|, |o^j|\} \quad (25)$$

The next two lemmas give a very useful estimation on the norm of $d^{m,j}$, $\Psi^{m,l,j}$, $R^{m,j}$, $r_i^{m,j}$, $\tilde{r}_i^{m,j}$, and the deviation of error functions $E(\mathbf{w}^{(m+1)J})$ and $E(\mathbf{w}^{mJ})$. To make the paper more readable, we move the proofs of the two lemmas to the Appendix.

Lemma 1. *Suppose Assumption (A1) holds and the sequence $\{\mathbf{w}^{mJ+j}\}$ is generated by (9)-(11), then there exist constants C_2 - C_6 such that*

$$\begin{aligned} \|d^{m,j}\| &\leq C_2 \eta_m, & \|\Psi^{m,l,j}\| &\leq C_3 \eta_m \\ \|R^{m,j}\| &\leq C_4 \eta_m^2, & \|r_i^{m,j}\| &\leq C_5 \eta_m^2, & \|\tilde{r}_i^{m,j}\| &\leq C_6 \eta_m^2 \end{aligned}$$

where $m \in \mathbb{N}; j, l = 0, 1, \dots, J-1; i = 1, 2, \dots, n.$

Lemma 2. *Suppose assumption (A1) holds and the sequence $\{\mathbf{w}^{mJ+j}\}$ is generated by (9)-(11), then we have*

$$E(\mathbf{w}^{(m+1)J}) \leq E(\mathbf{w}^{mJ}) - \eta_m \|\nabla E_{\mathbf{w}}(\mathbf{w}^{mJ})\|^2 + C_7 \eta_m^2 \quad (26)$$

where $m \in \mathbb{N}$ and $C_7 > 0$ is a constant independent of m and η_m .

The following three lemmas are important tools in the analysis on the convergence of iterative algorithm. They have been proved in [31], [2] and [27], respectively.

Lemma 3. *Let Y_t , W_t and Z_t be three sequences such that W_t is nonnegative and Y_t is bounded for all t . If*

$$Y_{t+1} \leq Y_t - W_t + Z_t, \quad t = 0, 1, \dots$$

and the series $\sum_{t=0}^{\infty} Z_t$ is convergent, then Y_t converges to a finite value and $\sum_{t=0}^{\infty} W_t < \infty$.

Lemma 4. *Suppose that the learning rate η_m satisfies (A2) and that the sequence $\{a_m\}$ ($m \in \mathbb{N}$) satisfies $a_m \geq 0$, $\sum_{m=0}^{\infty} \eta_m a_m^\beta < \infty$ and $|a_{m+1} - a_m| \leq \mu \eta_m$ for some positive constants β and μ . Then we have*

$$\lim_{m \rightarrow \infty} a_m = 0 \quad (27)$$

Lemma 5. *Let $F : \mathbf{D} \subset \mathbb{R}^p \rightarrow \mathbb{R}$, ($p \geq 1$) be continuous for a bounded closed region \mathbf{D} , and $\mathbf{D}_0 = \{\mathbf{z} \in \mathbf{D} : F(\mathbf{z}) = 0\}$. The projection of \mathbf{D}_0 on each coordinate axis does not contain any interior point. Let the sequence $\{\mathbf{z}^n\}$ satisfy:*

$$(i) \lim_{n \rightarrow \infty} F(\mathbf{z}^n) = 0;$$

(ii) $\lim_{n \rightarrow \infty} \|\mathbf{z}^{n+1} - \mathbf{z}^n\| = 0$;
 Then, there exists a unique $\mathbf{z}^* \in \mathbf{D}_0$ such that $\lim_{n \rightarrow \infty} \mathbf{z}^n = \mathbf{z}^*$.

By virtue of Lemmas 1-5, we can prove the following main results on the convergence of OGLA.

Theorem 1. Let the error function $E(\mathbf{w})$ be defined in (4), and the parameter sequence $\{\mathbf{w}^{mJ+j}\}$ be generated by (9)-(11) with \mathbf{w}^0 being an arbitrary initial value. If Assumptions (A1) and (A2) are valid, then we have the following convergence results for $j = 0, 1, \dots, J - 1$:

(i) There exists a constant $E^* \geq 0$ such that $\lim_{m \rightarrow \infty} E(\mathbf{w}^{mJ+j}) = E^*$;

(ii) $\lim_{m \rightarrow \infty} \|\nabla E_{\mathbf{w}}(\mathbf{w}^{mJ+j})\| = 0$.

Moreover, if Assumption (A3) is also valid, it holds the strong convergence: there exists a unique fixed point $\mathbf{w}^* \in \Omega_0$ such that

(iii) $\lim_{m \rightarrow \infty} \mathbf{w}^{mJ+j} = \mathbf{w}^*$.

Proof. The proof is divided into three parts, dealing with the statements (i), (ii) and (iii) respectively.

Proof of Statement (i). Considering that the sequence $\{E(\mathbf{w}^{mJ})\}$ satisfies Lemma 2, one can apply Lemma 3 to it. Thus, there exists a constant $E^* \geq 0$ as $E(\mathbf{w}^{mJ}) \geq 0$ such that

$$\lim_{m \rightarrow \infty} E(\mathbf{w}^{mJ}) = E^*$$

According to Assumption (A2), it holds that $\lim_{m \rightarrow \infty} \eta_m = 0$. Similar to the proof of Lemma 2, one can also conclude that

$$\lim_{m \rightarrow \infty} |E(\mathbf{w}^{mJ+j}) - E(\mathbf{w}^{mJ})| = 0, \quad j = 0, 1, \dots, J - 1$$

Hence, one can obtain that for $j = 0, 1, \dots, J - 1$

$$\lim_{m \rightarrow \infty} E(\mathbf{w}^{mJ+j}) = \lim_{m \rightarrow \infty} E(\mathbf{w}^{mJ}) = E^*$$

This completes the proof of Statement (i).

Proof of Statement (ii). By Assumption (A2), Lemma 2 and Lemma 3, one can obtain that

$$\begin{aligned} & \sum_{m=0}^{\infty} \eta_m \|\nabla E_{\mathbf{w}}(\mathbf{w}^{mJ})\|^2 \\ &= \sum_{m=0}^{\infty} \eta_m \|\nabla E_{\mathbf{u}}(\mathbf{w}^{mJ})\|^2 + \sum_{m=0}^{\infty} \sum_{i=1}^n \eta_m \left(\|\nabla E_{\mathbf{c}_i}(\mathbf{w}^{mJ})\|^2 + \|\nabla E_{\mathbf{b}_i}(\mathbf{w}^{mJ})\|^2 \right) \\ &< \infty \end{aligned}$$

which results in

$$\sum_{m=0}^{\infty} \eta_m \|\nabla E_{\mathbf{u}}(\mathbf{w}^{mJ})\|^2 < \infty \tag{28}$$

According to Lemma 1, one arrives at that

$$\begin{aligned} & \|\nabla E_{\mathbf{u}}(\mathbf{w}^{(m+1)J}) - \nabla E_{\mathbf{u}}(\mathbf{w}^{mJ})\| \\ &= \left\| \sum_{j=0}^{J-1} \left[E'_j(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j}) \mathbf{h}^{(m+1)J,j} - E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j}) \mathbf{h}^{mJ,j} \right] \right\| \\ &\leq \sum_{j=0}^{J-1} \left\| E'_j(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j}) \Psi^{m,J,j} \right\| \end{aligned}$$

$$\begin{aligned}
 & + \left(E'_j \left(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j} \right) - E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{(m+1)J,j} \right) \right) \mathbf{h}^{mJ,j} \\
 & + \left(E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{(m+1)J,j} \right) - E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \right) \mathbf{h}^{mJ,j} \Big\| \leq C_8 \eta_m
 \end{aligned} \tag{29}$$

where $C_8 = JC_4$. This immediately leads to

$$\left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{(m+1)J} \right) \right\| - \left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ} \right) \right\| \leq \left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{(m+1)J} \right) - \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ} \right) \right\| \leq C_8 \eta_m \tag{30}$$

The combination of (28), (30) and Lemma 4 results in

$$\lim_{m \rightarrow \infty} \left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ} \right) \right\| = 0$$

As in the proof to (29), there exists a positive constant C_9 such that for $j = 0, 1, \dots, J - 1$

$$\left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ+j} \right) - \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ} \right) \right\| \leq C_9 \eta_m$$

Since

$$\begin{aligned}
 \left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ+j} \right) \right\| & \leq \left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ+j} \right) - \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ} \right) \right\| + \left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ} \right) \right\| \\
 & \leq C_9 \eta_m + \left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ} \right) \right\|
 \end{aligned}$$

one attains

$$\lim_{m \rightarrow \infty} \left\| \nabla E_{\mathbf{u}} \left(\mathbf{w}^{mJ+j} \right) \right\| = 0, \quad j = 0, 1, \dots, J - 1$$

Similarly, one deduces that $\lim_{m \rightarrow \infty} \left\| \nabla E_{\mathbf{c}_i} \left(\mathbf{w}^{mJ+j} \right) \right\| = 0$ and $\lim_{m \rightarrow \infty} \left\| \nabla E_{\mathbf{b}_i} \left(\mathbf{w}^{mJ+j} \right) \right\| = 0$ for $j = 0, 1, \dots, J - 1$. This immediately gives

$$\lim_{m \rightarrow \infty} \left\| \nabla E_{\mathbf{w}} \left(\mathbf{w}^{mJ+j} \right) \right\| = 0, \quad j = 0, 1, \dots, J - 1 \tag{31}$$

and proves Statement (ii).

Proof of Statement (iii). Since

$$\begin{aligned}
 & \left\| \mathbf{w}^{mJ+j} - \mathbf{w}^{mJ} \right\|^2 \\
 & = \left\| \mathbf{u}^{mJ+j} - \mathbf{u}^{mJ} \right\|^2 + \sum_{i=1}^n \left(\left\| \mathbf{c}_i^{mJ+j} - \mathbf{c}_i^{mJ} \right\|^2 + \left\| \mathbf{b}_i^{mJ+j} - \mathbf{b}_i^{mJ} \right\|^2 \right),
 \end{aligned}$$

this together with (9)-(14) and Lemma 1 leads to

$$\lim_{m \rightarrow \infty} \left\| \mathbf{w}^{mJ+j} - \mathbf{w}^{mJ} \right\| = 0, \quad j = 0, 1, \dots, J - 1$$

It immediately results in

$$\lim_{m \rightarrow \infty} \left\| \mathbf{w}^{mJ+j} - \mathbf{w}^{mJ+j-1} \right\| = 0 \tag{32}$$

Noting that $\nabla E_{\mathbf{w}}(\mathbf{w})$ is continuous and according to Assumption (A3), (31) and (32), one can obtain that Statement (iii) holds by employing Lemma 5. This completes the whole proof of Theorem 1. \square

The statements (i) and (ii) of Theorem 1 present a weak convergence result for OGLA, which implies the gradient of error function goes to zero and the error function decreases to a stable value. The strong convergence result for OGLA is also confirmed in Statement (iii), which implies that the parameter sequence will stabilize to a fixed value at which the error function attains its minimum (may be a local minimum).

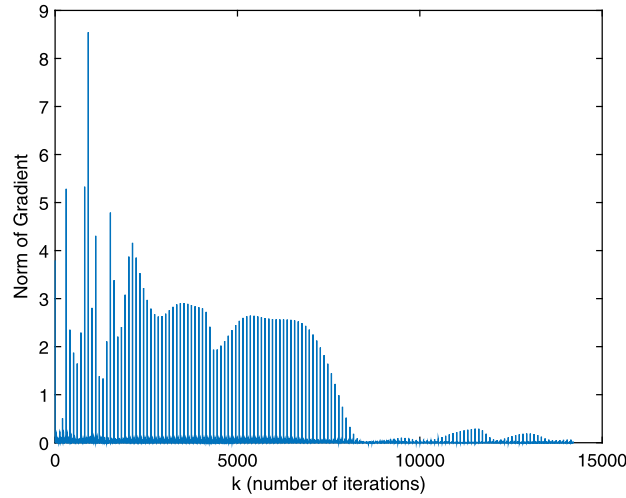


Fig. 2. Norm of gradient with OGLA0 for Example 1.

4. Numerical examples

In this section, three numerical examples are provided to illustrate the performance of OGLA for training the neuro-fuzzy systems and to exemplify the theoretical convergence analysis described in Theorem 1, where three assumptions are needed. Assumption (A1) indicates that our convergence result requires as a precondition the boundedness of the weights during the learning iteration process. This assumption is also used for the convergence analysis of the gradient method for ordinary BP neural networks. The boundedness of the weight sequence for online gradient algorithm can be guaranteed by introducing l_2 regularization into the error function (cf. [34]). Actually, the weights are bounded during the learning iteration process in most cases, which is shown in the following examples. Assumption (A2) is an usual condition for the convergence of online gradient method, and a typical choice for η_m to satisfy Assumption (A2) is am^{-b} , where a is a positive constant and $b \in (0.5, 1)$.

The performance of OGLA is illustrated by the learning and generalizing abilities for identifying nonlinear functions and predicting a chaotic time series. Moreover, we also use Example 1 to show OGLA, compared with OGLA0, can effectively avoid the oscillation during the iteration process.

Example 1. (Identification of a quadratic Hermite function). This example uses the neuro-fuzzy systems with OGLA to identify a quadratic Hermite function (33), which was presented by Mackay in [19]:

$$y(t) = 1.1(1 - t + 2t^2)e^{-t^2/2} \quad (33)$$

In this example, the training patterns are generated with t_i , $i = 1, 2, \dots, 100$ evenly extracted in the interval $[-4, 4]$. They are taken as the system's inputs and $y_i = y(t_i)$, $i = 1, 2, \dots, 100$ are the corresponding desired outputs. To demonstrate the performance of OGLA, four fuzzy rules are used. The initial fuzzy parameters are selected randomly in the interval $[-2, 2]$, the learning rate $\eta_m = 0.4m^{-0.6}$, where m is the number of cycle. The training procedure will be terminated once the maximum number of iterations 20,000 is reached or the mean squared error criterion of 0.003 is satisfied. The generalization of OGLA is also tested by using testing patterns which are taken uniformly between -4 and 4 with a size of 101. Furthermore, to demonstrate OGLA, compared with OGLA0, can avoid the oscillation during the iteration process, we apply OGLA0 with the same initial parameters for this example. The change of the norm of gradient with OGLA0 in the learning process is shown in Fig. 2, which shows that OGLA0 may result in oscillation.

The mean square error (MSE), the norm of gradient and weight vector with OGLA in the learning process are shown in Fig. 3, which shows the convergence performance of OGLA, indicating that the error function decreases monotonically and at last it tends to a constant, and the norm of the gradient of the error function tends to zero, and the norm of weight vector is bounded.

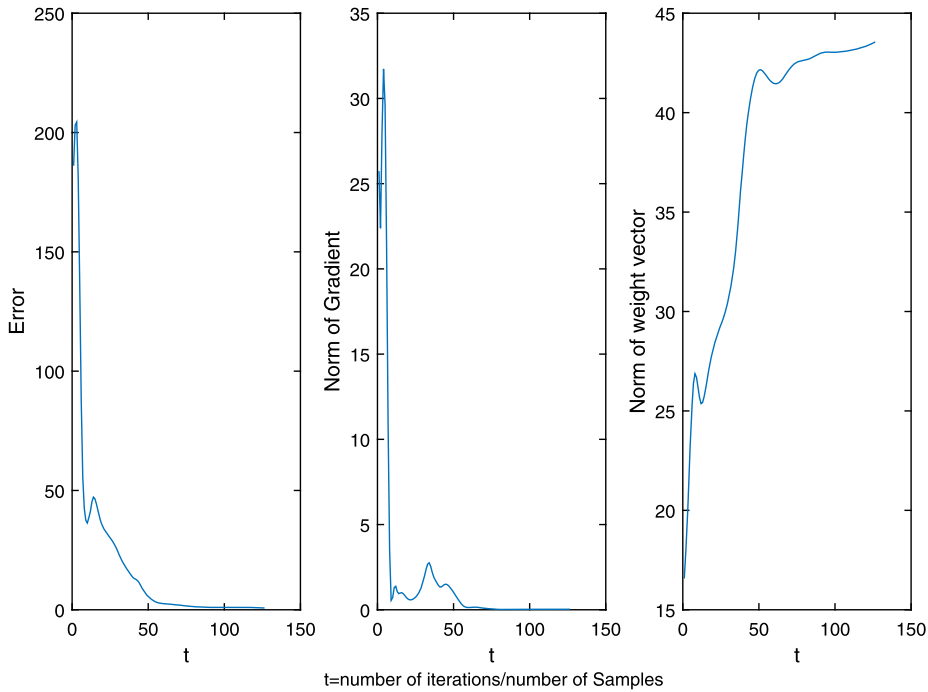


Fig. 3. Learning curves of OGLA for Example 1.

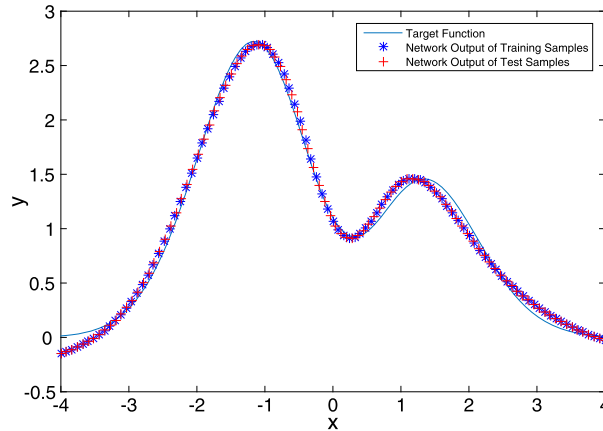


Fig. 4. The identification results for Example 1.

The approximation performances of OGLA for training and testing patterns are show in Fig. 4.

Example 2. (Identification of the Gabor function). In this example, we also illustrate the performance of OGLA by applying it to a two-dimensional function approximation problem. This system is trained and tested by approximating the following two-dimensional Gabor function:

$$h(x, y) = \frac{1}{2\pi(0.5)^2} e^{-\frac{x^2+y^2}{2(0.5)^2}} \cos(2\pi(x + y)) \tag{34}$$

Twenty-five training points are selected from an evenly spaced 5×5 grid on the square $-0.5 \leq x \leq 0.5$ and $-0.5 \leq y \leq 0.5$. Similarly, two hundred and fifty-six test patterns are generated from an evenly spaced 16×16 grid on the square $-0.5 \leq x \leq 0.5$ and $-0.5 \leq y \leq 0.5$. Eight fuzzy rules are used. The initial fuzzy parameters are selected randomly in the interval $[-1, 1]$, the learning rate $\eta_m = 0.5m^{0.51}$, where m is the number of cycle. The training

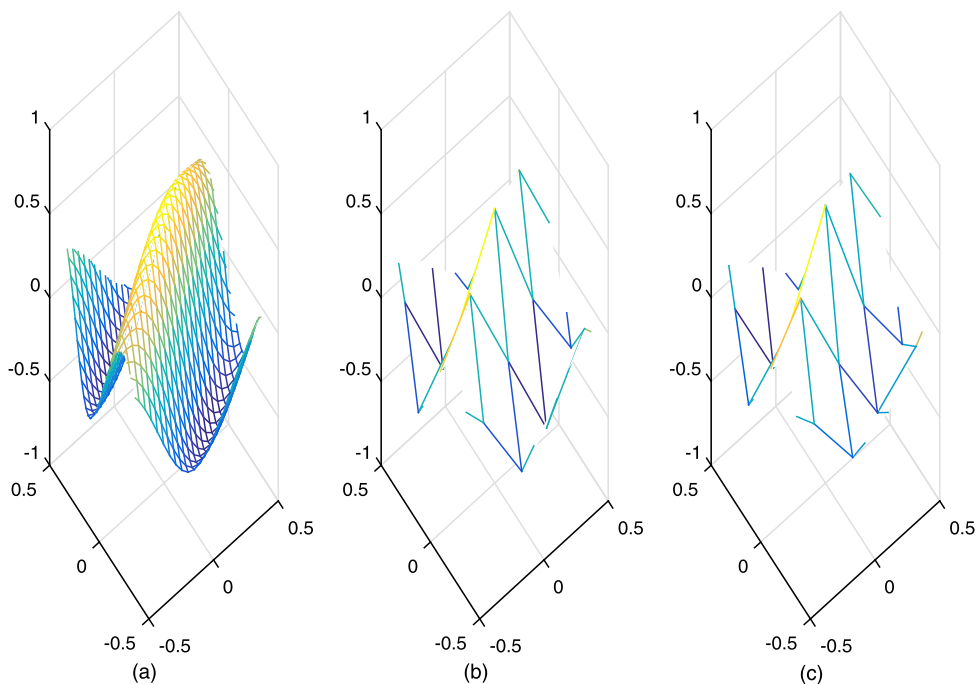


Fig. 5. Performance of training patterns: (a) Gabor function; (b) 25 training patterns; (c) approximation results of training patterns.

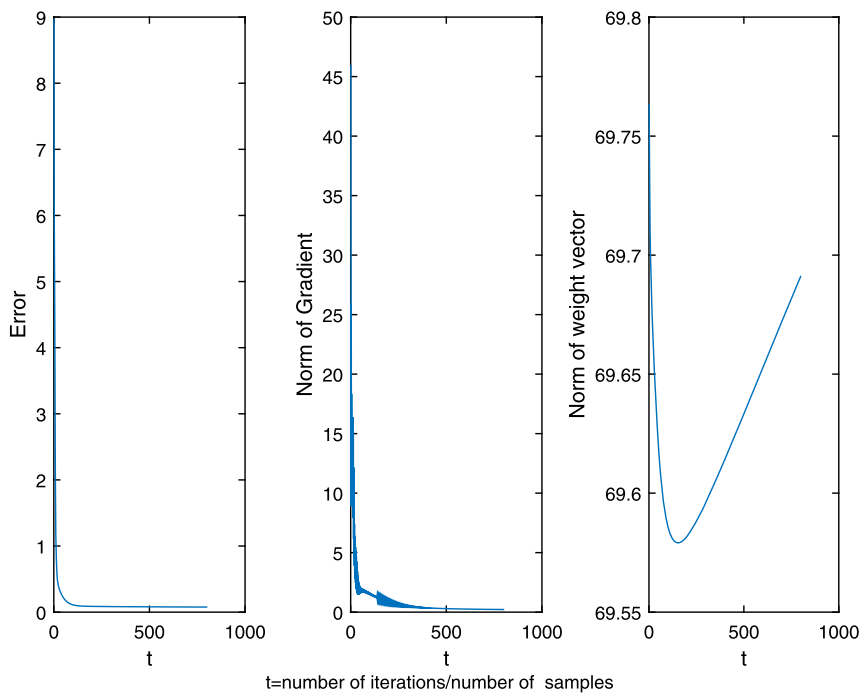


Fig. 6. Learning curves of OGLA for Example 2.

procedure will be terminated once the maximum number of iterations 20,000 is reached or the mean squared error criterion of 0.001 is satisfied. The training performance after 20,000 iterations is shown in Fig. 5 and Fig. 6, and the MSE value of training patterns is 0.0031. Fig. 5 shows the Gabor function, training patterns and the approximation results of training patterns. The learning curves in Fig. 6 also substantiate the validity of our theoretical convergence

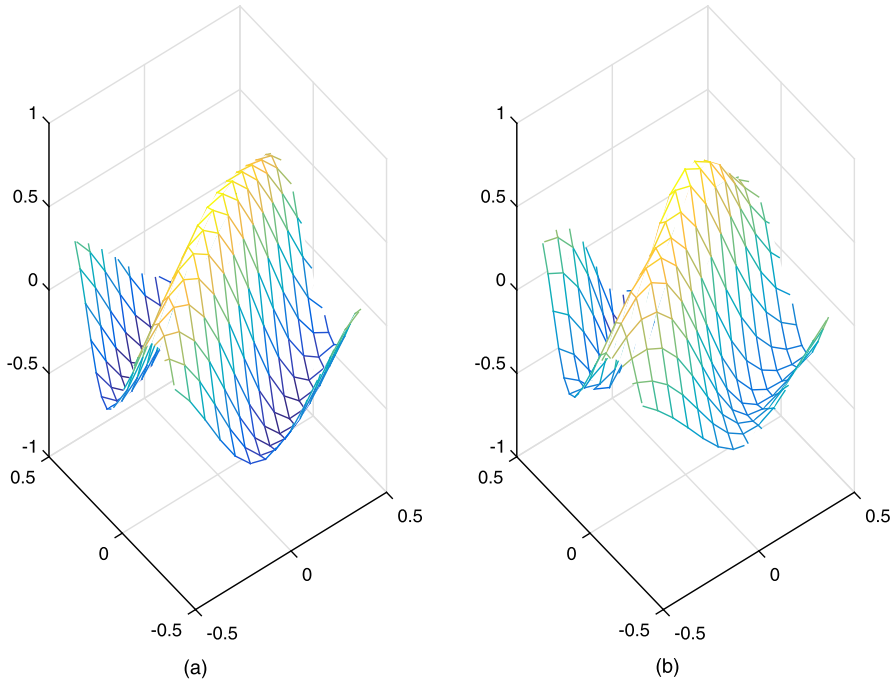


Fig. 7. Performance of test patterns: (a) 256 test patterns; (b) approximation results of test patterns.

results. Furthermore, The generalization capability of the learned system is illustrated by using the test patterns. The MSE value of test patterns is 0.0063 and the approximation performance of test patterns is shown in Fig. 7.

Example 3. (Prediction of a chaotic time series). The time series prediction problem used in this example is the Mackey-Glass chaotic time series that is generated from the following delay differential equation defined as

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \quad (35)$$

where $\tau = 17$ and $x(0) = 1.2$. Four past values are used to predict $x(t)$, and the input-output data format is

$$[x(t-24), x(t-18), x(t-12), x(t-6); x(t)].$$

In this example, twelve fuzzy rules are used. The initial fuzzy parameters are selected randomly in the interval $[-1, 1]$, the learning rate $\eta_m = 0.3m^{-0.51}$, where m is the number of cycle. The training procedure will be terminated once the maximum number of iterations 5×10^5 is reached or the mean squared error criterion of 0.001 is satisfied. One thousand samples are generated from $t = 124$ to $t = 1123$, with the first 500 samples being used for training and the last 500 samples for testing. Those samples are shown in Fig. 8. The training and testing results are shown in Fig. 9. and Fig. 10. It is easily seen that OGLA can efficiently learn and predict the Mackey-Glass time series and the absolute errors of prediction lie between $[-0.04, 0.08]$ for both training and testing samples.

5. Conclusion

In neuro-fuzzy systems, two major types of learning are required: structure learning algorithms to find appropriate fuzzy logical rules; and parameter learning algorithms to fine-tune the membership functions and other parameters. We are concerned with the parameter learning algorithm in this paper. An online gradient-based learning algorithm with adaptive learning rate is proposed to train the neuro-fuzzy systems representing the Mamdani fuzzy model. At each step, online learning uses a per-instance gradient which is not the true gradient over the entire training set, to update weights. We modify the original setting of the parameter to calculate the per-instance gradient by taking the reciprocals of the variances of Gaussian membership functions, rather than the variances themselves, as independent variables.

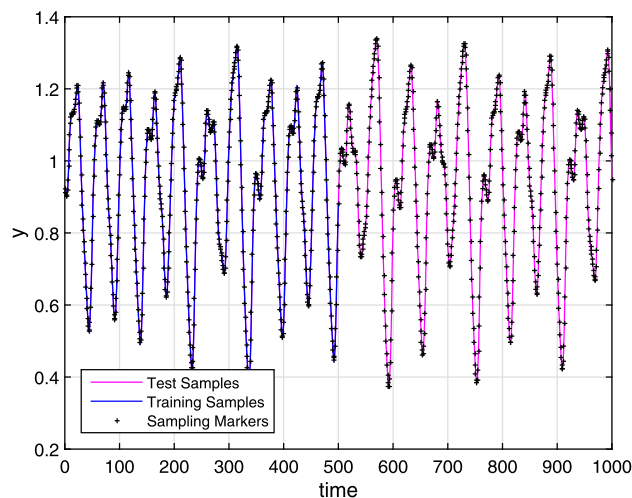


Fig. 8. Training and test samples.

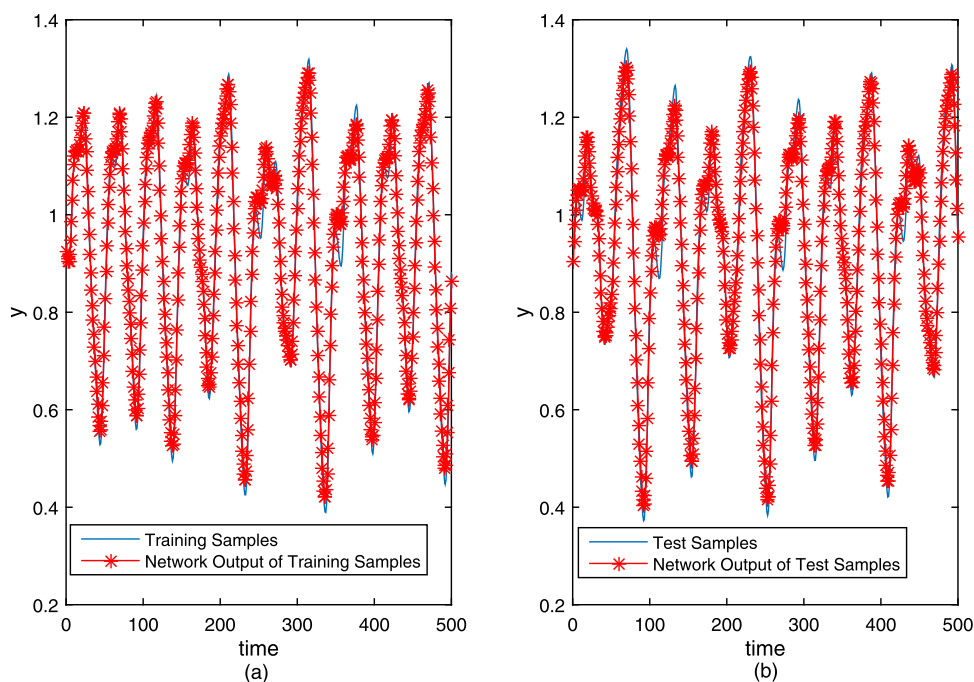


Fig. 9. Prediction results: (a) training results; (b) test results.

Due to this modification, the differentiation with respect to the denominator is avoided, which may result in oscillation in the learning process. Furthermore, a rigorous convergence analysis for the proposed algorithm is made. The weak convergence of this algorithm is proved under some assumptions, showing the gradient function tends to zero and the error function tends to a constant. The strong convergence is also theoretically obtained, indicating that the weight sequence converges to a fixed point. These theoretical results can provide a theoretic guarantee for the application of our proposed online learning algorithm. Three numerical examples are also given to show the effectiveness of our proposed algorithm and to support the theoretical findings.

Hybrid learning algorithms, which combine the online gradient learning algorithm with other algorithms, are often employed to improve the learning performance of neuro-fuzzy systems. In this respect, we expect that our proposed online gradient method can be used in place of the original online gradient method to combine with other algorithms.

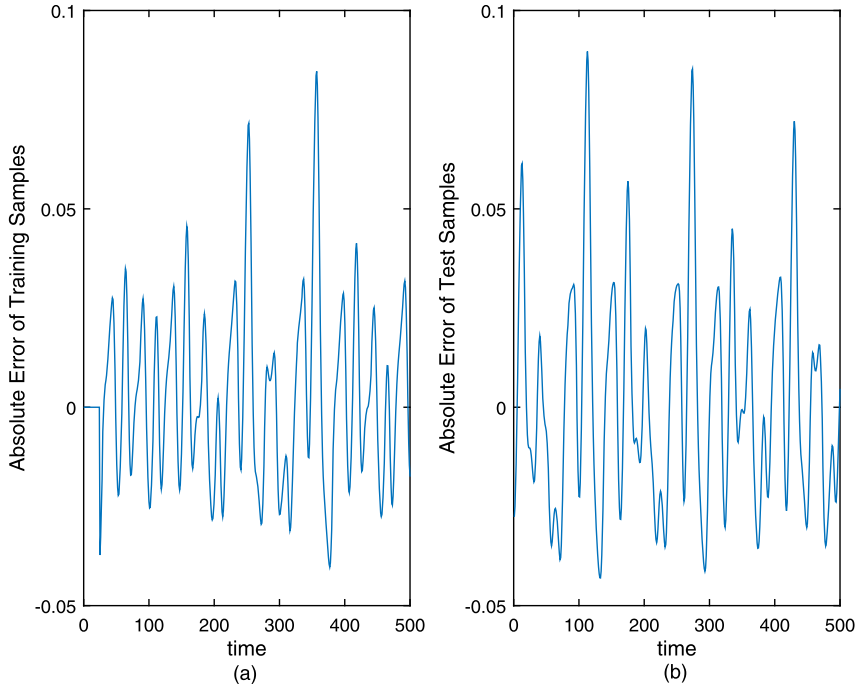


Fig. 10. Prediction errors: (a) errors of training samples; (b) errors of test samples.

Moreover, we only discuss the neuro-fuzzy systems representing the Mamdani fuzzy model with Gaussian fuzzy sets. Our further investigation will be focused on formulating learning algorithm for other kinds of neuro-fuzzy systems.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is partially supported by the Natural Science Foundation of China (11401185), the Scientific Research Fund of Hunan Provincial Education Department (19A063, 17A031), the Science and Technology Plan Project of Hunan Province (Hunan Provincial Key Laboratory of Intelligent Information Processing and Application, 2016TP1020), the Science and Technology Plan Project of Hengyang City (2017KJ183), the Application-oriented Characterized Disciplines, Double First-Class University Project of Hunan Province (Xiangjiaotong [2018]469), China Scholarship Council (No. 201608430257) and the 2019-2020 Hunan overseas distinguished professorship project (No. 2019014).

Appendix

The proofs of Lemma 1 and Lemma 2 presented in Section 4 are given in this section.

Proof of Lemma 1. By the definition of $E_j(t)$, it is easy to know that $E_j'(t) = t - o^j$. It follows from (8) and Assumption (A1) that for $m \in \mathbb{N}$ and $k = 0, 1, \dots, J-1$

$$\left\| E_k'(\mathbf{u}^{mJ+k} \cdot \mathbf{h}^{mJ+k,k}) \right\| = \left\| \mathbf{u}^{mJ+k} \cdot \mathbf{h}^{mJ+k,k} - o^k \right\| \leq \sqrt{n}C_0 + C_1 \quad (36)$$

and

$$\left\| \Delta_k \mathbf{u}^{mJ+k} \right\| = \left\| -\eta_m E_k'(\mathbf{u}^{mJ+k} \cdot \mathbf{h}^{mJ+k,k}) \mathbf{h}^{mJ+k,k} \right\| \leq (nC_0 + \sqrt{n}C_1)\eta_m \quad (37)$$

Then, we have

$$\|d^{m,j}\| = \|\mathbf{u}^{mJ+j} - \mathbf{u}^{mJ}\| = \left\| \sum_{k=0}^{j-1} \Delta_k \mathbf{u}^{mJ+k} \right\| \leq C_2 \eta_m \tag{38}$$

where $C_2 = (nC_0 + \sqrt{n}C_1)J$

Using the Mean Value Theorem, for $m \in \mathbb{N}$, $j, l = 0, 1, \dots, J - 1$ and $i = 1, 2, \dots, n$, we have

$$\begin{aligned} \Psi_i^{m,l,j} &= \mathbf{h}_i^{mJ+l,j} - \mathbf{h}_i^{mJ,j} = \exp\left(-\Phi_i^{mJ+l,j} \cdot \Phi_i^{mJ+l,j}\right) - \exp\left(-\Phi_i^{mJ,j} \cdot \Phi_i^{mJ,j}\right) \\ &= -\exp\left(t_i^{s,j}\right) \left(\Phi_i^{mJ+l,j} + \Phi_i^{mJ,j}\right) \left(\Phi_i^{mJ+l,j} - \Phi_i^{mJ,j}\right) \end{aligned} \tag{39}$$

where $t_i^{s,j}$ lies between $-\Phi_i^{mJ+l,j} \cdot \Phi_i^{mJ+l,j}$ and $-\Phi_i^{mJ,j} \cdot \Phi_i^{mJ,j}$. According to triangle inequality and Hadamard product “ \odot ”, we have

$$\begin{aligned} |\Psi_i^{m,l,j}| &\leq 2C_0(C_0 + C_1) \left\| \Phi_i^{mJ+l,j} - \Phi_i^{mJ,j} \right\| \\ &= 2C_0(C_0 + C_1) \left\| (\mathbf{x}^j - \mathbf{c}_i^{mJ+l}) \odot \mathbf{b}_i^{mJ+l} - (\mathbf{x}^j - \mathbf{c}_i^{mJ}) \odot \mathbf{b}_i^{mJ+l} \right. \\ &\quad \left. + (\mathbf{x}^j - \mathbf{c}_i^{mJ}) \odot \mathbf{b}_i^{mJ+l} - (\mathbf{x}^j - \mathbf{c}_i^{mJ}) \odot \mathbf{b}_i^{mJ} \right\| \\ &= 2C_0(C_0 + C_1) \left\| (-v_i^{m,l}) \odot \mathbf{b}_i^{mJ+l} + (\mathbf{x}^j - \mathbf{c}_i^{mJ}) \odot \tilde{v}_i^{m,l} \right\| \\ &\leq 2C_0(C_0 + C_1) \left(C_0 \left\| v_i^{m,l} \right\| + (C_0 + C_1) \left\| \tilde{v}_i^{m,l} \right\| \right) \\ &\leq 2C_0^2(C_0 + C_1) \left(\sum_{k=0}^{l-1} \left\| \Delta_k \mathbf{c}_i^{mJ+k} \right\| \right) + 2C_0(C_0 + C_1)^2 \left(\sum_{k=0}^{l-1} \left\| \Delta_k \mathbf{b}_i^{mJ+k} \right\| \right) \end{aligned} \tag{40}$$

By virtue of (13) and (14), we can get

$$\left\| \Delta_k \mathbf{c}_i^{mJ+k} \right\| \leq C_0^3(C_0 + C_1)(\sqrt{n}C_0 + C_1)\eta_m \tag{41}$$

and

$$\left\| \Delta_k \mathbf{b}_i^{mJ+k} \right\| \leq C_0^2(C_0 + C_1)^2(\sqrt{n}C_0 + C_1)\eta_m \tag{42}$$

The combination of (40)-(42) leads to

$$\|\Psi^{m,l,j}\| = \sqrt{\sum_{i=1}^n (\Psi_i^{m,l,j})^2} \leq C_3 \eta_m \tag{43}$$

where $C_3 = 2JC_0^3(C_0 + C_1)^2(nC_0 + \sqrt{n}C_1)(2C_0^2 + 2C_0C_1 + C_1^2)$.

By the definition of $E_j(t)$, it is easy to know that

$$\left| E'_j(\mathbf{u}^{mJ+j} \cdot \mathbf{h}^{mJ+j,j}) - E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ+j,j}) \right| \leq \sqrt{n} \|d^{m,j}\| \tag{44}$$

and

$$\left| E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ+j,j}) - E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j}) \right| \leq C_0 \|\Psi^{m,j,j}\| \tag{45}$$

By the definition of $R^{m,j}$, we have that

$$\begin{aligned} R^{m,j} &= \Delta_j \mathbf{u}^{mJ+j} - \Delta_j \mathbf{u}^{mJ} \\ &= -\eta_m \left(E'_j(\mathbf{u}^{mJ+j} \cdot \mathbf{h}^{mJ+j,j}) \mathbf{h}^{mJ+j,j} - E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j}) \mathbf{h}^{mJ,j} \right) \\ &= -\eta_m \left(E'_j(\mathbf{u}^{mJ+j} \cdot \mathbf{h}^{mJ+j,j}) \Psi^{m,j,j} \right) \end{aligned}$$

$$\begin{aligned}
 &+ \left(E'_j(\mathbf{u}^{mJ+j} \cdot \mathbf{h}^{mJ+j,j}) - E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ+j,j}) \right) \mathbf{h}^{mJ,j} \\
 &+ \left(E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ+j,j}) - E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j}) \right) \mathbf{h}^{mJ,j}
 \end{aligned} \tag{46}$$

The combination of (38) and (43)-(46) leads to

$$\|R^{m,j}\| \leq \eta_m \left((2C_0\sqrt{n} + C_1)\|\Psi^{m,j}\| + n\|d^{m,j}\| \right) \leq C_4\eta_m^2 \tag{47}$$

where $C_4 = nC_2 + (2C_0\sqrt{n} + C_1)C_3$

By the definition of $r_i^{m,j}$, we have that

$$\begin{aligned}
 r_i^{m,j} = &-2\eta_m \left[E'_j(\mathbf{u}^{mJ+j} \cdot \mathbf{h}^{mJ+j,j})u_i^{mJ+j}h_i^{mJ+j,j} \left(\Phi_i^{mJ+j,j} \odot \mathbf{b}_i^{mJ+j} \right) \right. \\
 &\left. - E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j})u_i^{mJ}h_i^{mJ,j} \left(\Phi_i^{mJ,j} \odot \mathbf{b}_i^{mJ} \right) \right]
 \end{aligned} \tag{48}$$

Let

$$F^{mJ+j,j} = u_i^{mJ+j}h_i^{mJ+j,j} \left(\Phi_i^{mJ+j,j} \odot \mathbf{b}_i^{mJ+j} \right)$$

and

$$F^{mJ,j} = u_i^{mJ}h_i^{mJ,j} \left(\Phi_i^{mJ,j} \odot \mathbf{b}_i^{mJ} \right)$$

Similarly, we can prove that there exist constants C_5 and C_6 such that

$$\|r_i^{m,j}\| \leq C_5\eta_m^2, \quad \|\tilde{r}_i^{m,j}\| \leq C_6\eta_m^2 \tag{49}$$

This completes the whole proof of Lemma 1. \square

Proof of Lemma 2. Expanding $E_j(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j})$ with Taylor formula, we know that

$$\begin{aligned}
 &E_j(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j}) \\
 &= E_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j}) + E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j}) \left(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j} - \mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \\
 &\quad + \frac{1}{2} \left(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j} - \mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right)^2 \\
 &= E_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j}) + E'_j(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j}) \left(d^{m,J} \cdot \mathbf{h}^{mJ,j} + \mathbf{u}^{mJ} \cdot \Psi^{m,J,j} + d^{m,J} \cdot \Psi^{m,J,j} \right) \\
 &\quad + \frac{1}{2} \left(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j} - \mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right)^2
 \end{aligned} \tag{50}$$

Expanding $\mathbf{h}_i^{(m+1)J,j}$ with Taylor formula, we have that

$$\begin{aligned}
 &h_i^{(m+1)J,j} - h_i^{mJ,j} \\
 &= \exp\left(-\|\Phi_i^{(m+1)J,j}\|^2\right) - \exp\left(-\|\Phi_i^{mJ,j}\|^2\right) \\
 &= -\exp\left(-\|\Phi_i^{mJ,j}\|^2\right) \left(\|\Phi_i^{(m+1)J,j}\|^2 - \|\Phi_i^{mJ,j}\|^2 \right) + \frac{1}{2} \exp(\tilde{t}_i^{s,j}) \left(\|\Phi_i^{(m+1)J,j}\|^2 - \|\Phi_i^{mJ,j}\|^2 \right)^2 \\
 &= -\exp\left(-\|\Phi_i^{mJ,j}\|^2\right) \left[2\Phi_i^{mJ,j} \cdot \left(\Phi_i^{(m+1)J,j} - \Phi_i^{mJ,j} \right) + \|\Phi_i^{(m+1)J,j} - \Phi_i^{mJ,j}\|^2 \right] + \delta_{i0} \\
 &= -2\exp\left(-\|\Phi_i^{mJ,j}\|^2\right) \Phi_i^{mJ,j} \cdot \left[\left(\mathbf{b}_i^{mJ} + \tilde{v}_i^{m,J} \right) \odot \left(-v_i^{m,J} \right) + \left(\mathbf{x}^j - \mathbf{c}_i^{mJ} \right) \odot \tilde{v}_i^{m,J} \right] + \delta_{i1} + \delta_{i0}
 \end{aligned} \tag{51}$$

where $\tilde{t}_i^{s,j}$ lies between $-\|\Phi_i^{(m+1)J,j}\|^2$ and $-\|\Phi_i^{mJ,j}\|^2$, $\delta_{i0} = \frac{1}{2} \exp(\tilde{t}_i^{s,j}) \left(\|\Phi_i^{(m+1)J,j}\|^2 - \|\Phi_i^{mJ,j}\|^2 \right)^2$ and $\delta_{i1} = -\exp\left(-\|\Phi_i^{mJ,j}\|^2\right) \left(\|\Phi_i^{(m+1)J,j} - \Phi_i^{mJ,j}\|^2 \right)$. By (51) and Hadamard product “ \odot ”, we can get that

$$\begin{aligned}
 & E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \left(\mathbf{u}^{mJ} \cdot \Psi^{m,J,j} \right) \\
 &= E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \sum_{i=1}^n u_i^{mJ} \left(h_i^{(m+1)J,j} - h_i^{mJ,j} \right) \\
 &= 2E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \sum_{i=1}^n u_i^{mJ} h_i^{mJ,j} \left(\Phi_i^{mJ,j} \odot \mathbf{b}_i^{mJ} \right) \cdot v_i^{m,J} \\
 &\quad - 2E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \sum_{i=1}^n u_i^{mJ} h_i^{mJ,j} \left(\Phi_i^{mJ,j} \odot \left(\mathbf{x}^j - \mathbf{c}_i^{mJ} \right) \right) \cdot \tilde{v}_i^{m,J} + \delta_1 + \delta_2 + \delta_3 \\
 &= -\frac{1}{\eta_m} \sum_{i=1}^n \Delta_j \mathbf{c}_i^{mJ} \cdot v_i^{m,J} - \frac{1}{\eta_m} \sum_{i=1}^n \Delta_j \mathbf{b}_i^{mJ} \cdot \tilde{v}_i^{m,J} + \delta_1 + \delta_2 + \delta_3 \tag{52}
 \end{aligned}$$

where $\delta_1 = 2E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \sum_{i=1}^n u_i^{mJ} h_i^{mJ,j} \left(\Phi_i^{mJ,j} \odot \tilde{v}_i^{m,J} \right) \cdot v_i^{m,J}$, $\delta_2 = E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \sum_{i=1}^n u_i^{mJ} \delta_{i1}$ and $\delta_3 = E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \sum_{i=1}^n u_i^{mJ} \delta_{i0}$.

Combining (4), (18)-(20), (50) and (52), we can deduce that

$$\begin{aligned}
 & E(\mathbf{w}^{(m+1)J}) \\
 &= E(\mathbf{w}^{mJ}) + \sum_{j=0}^{J-1} E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \left(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j} - \mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \\
 &\quad + \frac{1}{2} \sum_{j=0}^{J-1} \left(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j} - \mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right)^2 \\
 &\leq E(\mathbf{w}^{mJ}) - \frac{1}{\eta_m} \left[\left\| \sum_{j=0}^{J-1} \Delta_j \mathbf{u}^{mJ} \right\|^2 + \sum_{i=1}^n \left(\left\| \sum_{j=0}^{J-1} \Delta_j \mathbf{c}_i^{mJ} \right\|^2 + \left\| \sum_{j=0}^{J-1} \Delta_j \mathbf{b}_i^{mJ} \right\|^2 \right) \right] + \delta_m \\
 &= E(\mathbf{w}^{mJ}) - \eta_m \|E_{\mathbf{w}}(\mathbf{w}^{mJ})\|^2 + \delta_m \tag{53}
 \end{aligned}$$

where

$$\begin{aligned}
 \delta_m &= -\frac{1}{\eta_m} \sum_{j=0}^{J-1} \Delta_j \mathbf{u}^{mJ} \cdot \sum_{j=0}^{J-1} R^{m,j} - \frac{1}{\eta_m} \sum_{i=1}^n \left(\sum_{j=0}^{J-1} \Delta_j \mathbf{c}_i^{mJ} \cdot \sum_{j=0}^{J-1} r_i^{m,j} \right) \\
 &\quad - \frac{1}{\eta_m} \sum_{i=1}^n \left(\sum_{j=0}^{J-1} \Delta_j \mathbf{b}_i^{mJ} \cdot \sum_{j=0}^{J-1} \tilde{r}_i^{m,j} \right) \\
 &\quad + 2 \sum_{j=0}^{J-1} E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \sum_{i=1}^n u_i^{mJ} h_i^{mJ,j} \left(\Phi_i^{mJ,j} \odot \tilde{v}_i^{m,J} \right) \cdot v_i^{m,J} \\
 &\quad - \sum_{j=0}^{J-1} E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \sum_{i=1}^n u_i^{mJ} h_i^{mJ,j} \|\Phi_i^{(m+1)J,j} - \Phi_i^{mJ,j}\|^2 \\
 &\quad + \frac{1}{2} \sum_{j=0}^{J-1} E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) \sum_{i=1}^n u_i^{mJ} \exp(\tilde{r}_i^{s,j}) \left(\|\Phi_i^{(m+1)J,j}\|^2 - \|\Phi_i^{mJ,j}\|^2 \right)^2 \\
 &\quad + \sum_{j=0}^{J-1} E'_j \left(\mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right) d^{m,J} \cdot \Psi^{m,J,j} + \frac{1}{2} \sum_{j=0}^{J-1} \left(\mathbf{u}^{(m+1)J} \cdot \mathbf{h}^{(m+1)J,j} - \mathbf{u}^{mJ} \cdot \mathbf{h}^{mJ,j} \right)^2
 \end{aligned}$$

According to Lemma 4, the first term of δ_m can be estimated as follows.

$$\left\| -\frac{1}{\eta_m} \sum_{j=0}^{J-1} \Delta_j \mathbf{u}^{mJ} \cdot \sum_{j=0}^{J-1} R^{m,j} \right\| \leq \frac{1}{\eta_m} \sum_{j=0}^{J-1} \|\Delta_j \mathbf{u}^{mJ}\| \sum_{j=0}^{J-1} \|R^{m,j}\| \leq C_{71} \eta_m^2$$

where $C_{71} = J^2 C_2 (nC_0 + \sqrt{n} C_1)$.

Similar estimates for the other terms of δ_m can be obtained with corresponding constants $C_{7t} > 0$ for $t = 2, 3, \dots, 8$. Hence, there exists a constant $C_7 = \sum_{t=1}^8 C_{7t}$ such that $\delta_m \leq C_7 \eta_m^2$. This completes the proof of Lemma 2. \square

References

- [1] P.P. Angelov, D. Filev, An approach to online identification of Takagi-Sugeno fuzzy models, *IEEE Trans. Syst. Man Cybern., Part B, Cybern.* 34 (1) (2004) 484–498.
- [2] D.P. Bertsekas, J.N. Tsitsiklis, Gradient convergence in gradient methods with errors, *SIAM J. Optim.* 3 (2000) 627–642.
- [3] Y. Bodyanskiy, O. Vynokurova, G. Setlak, D. Peleshko, P. Mulesa, Adaptive multivariate hybrid neuro-fuzzy system and its on-board fast learning, *Neurocomputing* 230 (2017) 409–416.
- [4] I. Campo, J. Echanobe, G. Bosque, J. Tarela, Efficient hardware/software implementation of an adaptive neuro-fuzzy system, *IEEE Trans. Fuzzy Syst.* 16 (2008) 761–778.
- [5] K. Chaturved, M. Pandit, L. Srivastava, Modified neo-fuzzy neuron-based approach for economic and environmental optimal power dispatch, *Appl. Soft Comput.* 8 (2008) 1428–1438.
- [6] C.L.P. Chen, Y.J. Liu, G.X. Wen, Fuzzy neural network-based adaptive control for a class of uncertain nonlinear stochastic systems, *IEEE Trans. Cybern.* 44 (5) (2014) 583–592.
- [7] A.K. Das, S. Sundaram, N. Sundararajan, A self-regulated interval type-2 neuro-fuzzy inference system for handling nonstationarities in EEG signals for BCI, *IEEE Trans. Fuzzy Syst.* 24 (6) (2016) 1565–1577.
- [8] H.G. Han, Z.L. Lin, J.F. Qiao, Modeling of nonlinear systems using the self-organizing fuzzy neural network with adaptive gradient algorithm, *Neurocomputing* 266 (2017) 566–578.
- [9] H. Ichihashi, I.B. Tüksten, A neuro-fuzzy approach to data analysis of pairwise comparisons, *Int. J. Approx. Reason.* 9 (1993) 227–248.
- [10] J.S.R. Jang, ANFIS: adaptive-network-based fuzzy inference system, *IEEE Trans. Syst. Man Cybern.* 23 (2) (1993) 665–685.
- [11] C.F. Juang, C.T. Lin, An on-line self-constructing neural fuzzy inference network and its applications, *IEEE Trans. Fuzzy Syst.* 6 (1) (1998) 12–32.
- [12] S. Kar, S. Das, P.K. Ghosh, Applications of neuro fuzzy systems: a brief review and future outline, *Appl. Soft Comput.* 15 (2014) 243–259.
- [13] J. Kim, N. Kasabov, HyFIS: adaptive neuro-fuzzy inference systems and their application to nonlinear dynamical systems, *Neural Netw.* 12 (1999) 1301–1319.
- [14] J.N. Li, J.Q. Yi, D.B. Zhao, G.C. Xi, A new fuzzy identification approach for complex systems based on neural-fuzzy inference network, *Acta Autom. Sin.* 32 (2006) 695–730.
- [15] C.T. Lin, C.S.G. Lee, Real-time supervised structure parameter learning for fuzzy neural network, in: *Proc. IEEE Int. Conf. Fuzzy Systems*, San Diego, USA, 1992, pp. 1283–1291.
- [16] Y. Liu, W. Wu, Q.W. Fan, D.K. Yang, J. Wang, A modified gradient learning algorithm with smoothing $L_{1/2}$ regularization for Takagi-Sugeno fuzzy models, *Neurocomputing* 138 (2014) 229–237.
- [17] Y. Liu, D.K. Yang, Convergence analysis of the batch gradient-based neuro-fuzzy learning algorithm with smoothing $L_{1/2}$ regularization for the first-order Takagi-Sugeno system, *Fuzzy Sets Syst.* 319 (2017) 28–49.
- [18] X.G. Luo, D. Liu, B.W. Wan, An adaptive fuzzy neural inferring network, *Fuzzy Syst. Math.* 12 (1998) 26–33.
- [19] David J.C. Mackay, Bayesian interpolation, *Neural Comput.* 3 (4) (1992) 415–447.
- [20] S.D. Nguyen, Q.H. Nguyen, Tae-Il Seo, ANFIS deriving from jointed input-output data space and applying insmart-damper identification, *Appl. Soft Comput.* 53 (2017) 45–60.
- [21] H. Nomura, I. Hayashi, N. Wakami, A learning method of fuzzy inference rules by descent method, in: *Proc. IEEE Int. Conf. Fuzzy Systems*, San Diego, USA, 1992, pp. 203–210.
- [22] H. Qin, S.X. Yang, Adaptive neuro-fuzzy inference systems based approach to nonlinear noise cancellation for images, *Fuzzy Sets Syst.* 158 (2007) 1036–1063.
- [23] Y. Shi, M. Mizumoto, Some considerations on conventional neuro-fuzzy learning algorithm by gradient descent method, *Fuzzy Sets Syst.* 112 (2000) 51–63.
- [24] L.X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1994.
- [25] L.X. Wang, J.M. Mendel, Generating fuzzy rules by learning from examples, in: *Proc. 6th IEEE Int. Symposium on Intelligent Control*, Washington D.C., 1991, pp. 263–268.
- [26] L.X. Wang, J.M. Mendel, Back-propagation fuzzy systems as nonlinear dynamic system identifiers, in: *Proc. of the IEEE Int. Conf. Fuzzy Systems*, San Diego, USA, 1992, pp. 1409–1416.
- [27] J. Wang, W. Wu, J.M. Zurada, Deterministic convergence of conjugate gradient method for feedforward neural networks, *Neurocomputing* 74 (2011) 2368–2376.

- [28] D.R. Wilson, T.R. Martinez, The general inefficiency of batch training for gradient descent learning, *Neural Netw.* 16 (2003) 1429–1451.
- [29] W. Wu, G.R. Feng, Z.X. Li, Y.S. Xu, Deterministic convergence of an online gradient method for BP neural networks, *IEEE Trans. Neural Netw.* 16 (3) (2005) 533–540.
- [30] W. Wu, L. Li, J. Yang, Y. Liu, A modified gradient-based neuro-fuzzy learning algorithm and its convergence, *Inf. Sci.* 180 (2010) 1630–1642.
- [31] W. Wu, J. Wang, M.S. Chen, Z.X. Li, Convergence analysis of online gradient method for BP neural networks, *Neural Netw.* 24 (2011) 91–98.
- [32] Z.B. Xu, R. Zhang, W.F. Jing, When does online BP training converge?, *IEEE Trans. Neural Netw.* 20 (10) (2009) 1529–1539.
- [33] H.S. Zhang, Y.L. Tang, Online gradient method with smoothing l_0 regularization for feedforward neural networks, *Neurocomputing* 224 (2017) 1–8.
- [34] H.S. Zhang, W. Wu, F. Liu, M.C. Yao, Boundedness and convergence of online gradient method with penalty for feedforward neural networks, *IEEE Trans. Neural Netw.* 20 (6) (2009) 1050–1054.