IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 28, 2018, accepted November 20, 2018, date of current version April 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2883381

Strong Convergence of Neuro-Fuzzy Learning With Adaptive Momentum for Complex System

YAN LIU[1], FANG LIU[2], AND LONG LI[3]

[1]School of Information Science and Engineering, Dalian Polytechnic University, Dalian 116034, China
[2]School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, China
[3]Department of Mathematics and Computational Science, Hengyang Normal University, Hengyang 421002, China

Corresponding author: Long Li (long_li1982@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61403056 and Grant 11401185, in part by the Natural Science Foundation Guidance Project of Liaoning Province under Grant 201602050, and in part by the Dalian Youth Science and Technology Star Project under Grant 2017RQ129.

ABSTRACT This paper studies a split-complex-valued neuro-fuzzy algorithm for fuzzy inference system, which realizes a frequently used zero-order Takagi–Sugeno–Kang system. Here, adaptive momentum is utilized to speed up the learning convergence. Some strong convergence results are demonstrated based on the weak convergence results, which expresses that the weight sequence of fuzzy parameters converges to a fixed point. Simulation results support the theoretical findings.

INDEX TERMS Strong convergence, neuro-fuzzy algorithm, complex, adaptive momentum, fuzzy inference system.

I. INTRODUCTION

Applications of telecommunications and signal processing technologies have encountered a rapid growth, resulting in an explosion in the research on complex data. Subsequently, there has been a growing interest in the theoretical research and practical implementation of complex-valued systems [1], [2]. In addition, studies on the competence of complex-valued neurons have proclaimed that they possess more superior computational power than their real-valued counterpart [3].

Currently, in order to explore and extend real-valued neural networks for the unique ability of getting the optimal solution formulation in complex domain, various complex-valued neural models are raised. For instance, [4] established a class of fractional-order complex-valued neural networks with time delay, and intensively studied the problem of dissipativity and global asymptotic stability based on the fractional Halanay inequality and suitable Lyapunov functions. Reference [5] proposed a improved complex-valued RBF neural network with reduced search space moving technique in its second stage for multiple crack damage identification. Complex models can be sorted into two classes in accordance with the types of the activation function: the split complex-valued network [6], [7] and the fully complex-valued network [8]. As stated by Liouville's theorem, a bounded function must be a constant in $\mathbb{C}$, where an entire function is defined as analytic, i.e., differentiable at every point in $\mathbb{C}$ [9]. Therefore, there is no analytic complex nonlinear function that is bounded everywhere on the entire complex plane. In split-complex networks, a pair of real-valued activation functions is splitted and then utilized to dispose the real and imaginary parts of a weighted input signals individually. This splitting trick aims at efficiently avoiding the occurrence of singular points in the adaptive training procedure [10]. Therefore, we concentrate the study on the analysis of the split complex-valued network.

Neuro-fuzzy inference system (NFIS) are verified to be efficient when applied to various fields. As for the split-complex valued neuro-fuzzy inference systems (SCNFIS), [11] demonstrated a CIT2FIS (complex-valued interval type-2 fuzzy inference system) and deduced its metacognitive projection-based learning (PBL) algorithm, which implemented a TSK type fuzzy system in a complex-valued neural network framework. Reference [12] presented a complex neuro-fuzzy system to achieve high accuracy for the problem of function approximation with the inheritance of the property of universal approximation of neuro-fuzzy system.

Unfortunately, slow convergence is obstacle that always limits its algorithm competence. To break through this obstacle, the momentum strategy has been raised, being capable of speeding up the convergence performance and decreasing the steepest descent error efficiently [13], since it lowers the energy value along the gradient direction in a close-to-optimal way. Reference [14] pointed out the necessity to use a momentum technique that both adapt to reinforcement learning and accelerate the learning process and compared the performance of different momentum techniques.

This paper aims to multiply adaptive momentum to SCNFA, which is self-adaptive updating iteratively by compositing the current weights coefficients with the previous-step updated coefficients. In the case of nonzero momentum coefficients, they are redefined as positive value to speed up training by renewing momentum. Otherwise the momentum is expected as zero to maintain the error downhill, being reduced to the gradient-based algorithm.

Another contribution of this paper is to present some strict convergence results of the SCNFA with adaptive momentum. We borrow some idea from [15] and [16], but we utilize some different proof techniques obtaining a new relaxed learning rate restriction which is much easier to inspect than the counterpart in [17].

The rest of this paper is arranged as follows. Section II briefly introduces the SCNFA with a adaptive momentum. Section III presents the strong convergence results. Simulations are carried out in section IV to support the theory, showing its the superior performance in regard to convergence rate and steady-state behavior. Conclusions are given in Section V. Finally, the rigorous proof of the strong convergence results is demonstrated in the appendix.

mds
Received: date / Accepted: date

II. COMPLEX-VALUED NEURO FUZZY INFERENCE SYSTEM WITH MOMENTUM

A. ARCHITECTURE OF ZERO-ORDER TSK

A general TSK fuzzy system constitutes by a set of IF-THEN rules taking the following shape [18]:

$$\begin{aligned} &Rule\ q: \ IF\ x_1\ is\ A_{1q}\ and\ x_2\ is\ A_{2q}\ and \\ &\qquad \ldots\ and\ x_L\ is\ A_{Lq}\ THEN\ y\ is\ y_q, \end{aligned} \tag{1}$$

where Q is the number of the fuzzy rules, and $x_l, y_q\,(l = 1, \ldots, L, q = 1, \ldots, Q)$ are complex numbers.

A four-layered network realizes zero-order TSK based on fuzzy inference system, whose topological structure is shown in Figure 1.

Layer 1 is the input layer formed by L input units, each unit rely on one complex input variable of $\mathbf{x} = \mathbf{x}^R + i\mathbf{x}^I = (x_1, x_2, \ldots, x_L)^T \in \mathbb{C}^L$, $\mathbf{x}^R, \mathbf{x}^I \in \mathbb{R}^L$, where $x_l = x_l^R + ix_l^I$, $x_l^R, x_l^I \in \mathbb{R}^1$ and $i = \sqrt{-1}$. Layer 2 is a Gaussian layer, $A_{lq}(x_l)$ represents the type of Gaussian membership for the fuzzy judgment "x_l *is* A_{lq}":

$$\begin{aligned} A_{lq}(x_l) &= \exp\big(-(x_l - a_{lq})(x_l - a_{lq})^*/\sigma_{lq}^2\big) \\ &= \exp\big(-(x_l - a_{lq})(x_l - a_{lq})^*\, b_{lq}^2\big), \end{aligned} \tag{2}$$

where "$*$" signifies complex conjugate, $a_{lq} \in \mathbb{C}^1$ and $\sigma_{lq} \in \mathbb{R}^1$ are the center and the width of Gaussian rule antecedent,

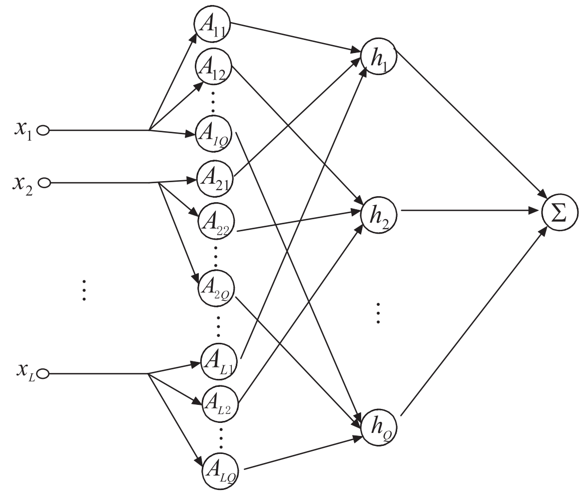


FIGURE 1. Topological Structure of the zero-order TSK inference system.

respectively. b_{lq} is the reciprocal of $\sigma_{lq}, l = 1, 2, \ldots, L$, $q = 1, 2, \ldots, Q$. We denote (cf. [15])

$$\begin{aligned} \mathbf{a}_q &= \left(a_{1q}, a_{2q}, \ldots, a_{Lq}\right)^T, \\ \mathbf{b}_q &= (b_{1q}, b_{2q}, \ldots, b_{Lq})^T = \left(\frac{1}{\sigma_{1q}}, \frac{1}{\sigma_{2q}}, \ldots, \frac{1}{\sigma_{Lq}}\right)^T, \end{aligned} \tag{3}$$

where $\mathbf{a}_q = \mathbf{a}_q^R + i\mathbf{a}_q^I \in \mathbb{C}^L$, $\mathbf{a}_q^R, \mathbf{a}_q^I \in \mathbb{R}^L$, $a_{l,q} = a_{l,q}^R + ia_{l,q}^I$, $a_{l,q}^R, a_{l,q}^I \in \mathbb{R}^1$, $1 \le l \le L$, $1 \le q \le Q$.

Layer 3 is the rule layer with Q nodes, $\mathbf{h} = (h_1, \ldots, h_Q)^T \in \mathbb{R}^Q$ and the agreement of the q-th antecedent part is computed by

$$\begin{aligned} h_q &= h_q(\mathbf{x}) = \prod_{l=1}^{L} A_{lq}(x_l) \\ &= \exp[\sum_{l=1}^{L} -(x_l - a_{l,q})(x_l - a_{l,q})^* \,(b_{l,q})^2] \\ &= \exp[\sum_{l=1}^{L} -\big((x_l^R - a_{l,q}^R)^2 + (x_l^I - a_{l,q}^I)^2\big)(b_{l,q})^2]. \end{aligned} \tag{4}$$

The weights linking Gaussian Layer and rule Layer are fixed as constant 1.

Layer 4 exports the sole output unit:

$$d = \sum_{q=1}^{Q} h_q y_q = d^R + id^I = \sum_{q=1}^{Q} h_q y_q^R + i \sum_{q=1}^{Q} h_q y_q^I. \tag{5}$$

Let the conclusion parameters be $\mathbf{a}_0 = (y_1, y_2, \ldots, y_Q)^T \in \mathbb{C}^Q$, where $\mathbf{a}_0 = \mathbf{a}_0^R + i\mathbf{a}_0^I$, $\mathbf{a}_0^R, \mathbf{a}_0^I \in \mathbb{R}^Q$, $y_q = y_q^R + iy_q^I$, $y_q^R, y_q^I \in \mathbb{R}^1$and take $\mathbf{a}_0$ as the weight vector connecting Layer 3 and Layer 4. Then, (5) can be taken as another form

$$d = \mathbf{a}_0^R \cdot \mathbf{h} + i\mathbf{a}_0^I \cdot \mathbf{h}. \tag{6}$$

B. LEARNING ALGORITHM OF SCNFA

Let $\{\mathbf{x}^j, O^j\}_{j=1}^J \subset \mathbb{C}^L \times \mathbb{C}^1$ be a training set with J training samples transmitted to the system. The square error function of the system trained by SCNFA is defined as following:

$$\begin{aligned}
E(\mathbf{W}) &= \frac{1}{2}\sum_{j=1}^{J}(O^j - d^j)(O^j - d^j)^* \\
&= \frac{1}{2}(\sum_{j=1}^{J}(O^{j,R} - d^{j,R})^2 + (O^{j,I} - d^{j,I})^2) \\
&= \sum_{j=1}^{J}[\epsilon_{j,R}(\mathbf{a}_0^R \cdot \mathbf{h}^j) + \epsilon_{j,I}(\mathbf{a}_0^I \cdot \mathbf{h}^j)] \\
&= \sum_{j=1}^{J}[\epsilon_{j,R}(\sum_{q=1}^{Q} y_q^R h_q) + \epsilon_{j,I}(\sum_{q=1}^{Q} y_q^I h_q)] \\
&= \sum_{j=1}^{J}[\epsilon_{j,R}(\sum_{q=1}^{Q} y_q^R \exp[\sum_{l=1}^{L} -((x_l^{j,R} - a_{l,q}^R)^2 \\
&\quad + (x_l^{j,I} - a_{l,q}^I)^2)(b_{l,q})^2]) \\
&\quad + \epsilon_{j,I}(\sum_{q=Q}^{Q} y_q^I \exp[\sum_{l=1}^{L} -((x_l^{j,R} - a_{l,q}^R)^2 \\
&\quad + (x_l^{j,I} - a_{l,q}^I)^2)(b_{l,q})^2])],
\end{aligned} \tag{7}$$

where O^j is the ideal output for the j-th training pattern $\mathbf{x}^j$, d^j is the corresponding fuzzy reasoning result, and for $j = 1, \ldots, J$

$$\mathbf{h}^j = (h_1^j, h_2^j, \ldots, h_Q^j) = \mathbf{h}(\mathbf{x}^j),$$

$$\epsilon_{j,R}(t) = \frac{1}{2}(t - d^{j,R})^2, \quad \epsilon_{j,I}(t) = \frac{1}{2}(t - d^{j,I})^2, \ t \in \mathbb{R}. \tag{8}$$

For simplicity, all weighted parameters are combined into a weight vector $\mathbf{W} \in \mathbb{C}^{L(2Q+1)}$:

$$\mathbf{W} = \left((\mathbf{a}_0)^T, (\mathbf{a}_1)^T, \ldots, (\mathbf{a}_Q)^T, (\mathbf{b}_1)^T, \ldots, (\mathbf{b}_Q)^T\right)^T.$$

The objective of the network learning is to obtain $\mathbf{W}^\star$ such that

$$\mathbf{W}^\star = \arg min_{\mathbf{W}} E(\mathbf{W}). \tag{9}$$

We differentiate $E(\mathbf{W})$ with respect to the real parts and the imaginary parts of the weight vectors,

$$\begin{aligned}
\frac{\partial E(\mathbf{W})}{\partial \mathbf{a}_0^R} &= \sum_{j=1}^{J} \epsilon'_{j,R}(\mathbf{a}_0^R \cdot \mathbf{h}^j)\mathbf{h}^j, \\
\frac{\partial E(\mathbf{W})}{\partial \mathbf{a}_0^I} &= \sum_{j=1}^{J} \epsilon'_{j,I}(\mathbf{a}_0^I \cdot \mathbf{h}^j)\mathbf{h}^j.
\end{aligned} \tag{10}$$

Hadamard product operator "$\odot$" is introduced for easy reading. Noting

$$\begin{aligned}
h_q^j &= h_q(\mathbf{x}^j) \\
&= \exp[\sum_{l=1}^{L} -\left((x_l^{j,R} - a_{l,q}^R)^2 + (x_l^{j,I} - a_{l,q}^I)^2\right)(b_{l,q})^2],
\end{aligned} \tag{11}$$

we have

$$\frac{\partial h_k^j}{\partial \mathbf{a}_q^R} = \frac{\partial h_k^j}{\partial \mathbf{a}_q^I} = 0, \quad \forall k \neq q, \tag{12}$$

and the partial gradient of h_q^j with respect to the real parts and the imaginary parts of $\mathbf{a}_q$ are

$$\begin{aligned}
\frac{\partial h_q^j}{\partial \mathbf{a}_q^R} &= (\frac{\partial h_q^j}{\partial a_{1,q}^R}, \ldots, \frac{\partial h_q^j}{\partial a_{L,q}^R}) \\
&= (2h_q^j b_{1,q}^2(x_l^{j,R} - a_{1,q}^R), \ldots, 2h_q^j b_{L,q}^2(x_L^{j,R} - a_{L,q}^R)) \\
&= 2h_q^j\left((\mathbf{x}^{j,R} - \mathbf{a}_q^R) \odot \mathbf{b}_q \odot \mathbf{b}_q\right),
\end{aligned} \tag{13}$$

$$\begin{aligned}
\frac{\partial h_q^j}{\partial \mathbf{a}_q^I} &= (\frac{\partial h_q^j}{\partial a_{1,q}^I}, \ldots, \frac{\partial h_q^j}{\partial a_{L,q}^I}) \\
&= (2h_q^j b_{1,q}^2(x_l^{j,I} - a_{1,q}^I), \ldots, 2h_q^j b_{L,q}^2(x_L^{j,I} - a_{L,q}^I)) \\
&= 2h_q^j\left((\mathbf{x}^{j,I} - \mathbf{a}_q^I) \odot \mathbf{b}_q \odot \mathbf{b}_q\right).
\end{aligned} \tag{14}$$

In the light of (12), (13) and (13), for $q = 1, \ldots, Q$, the partial gradient of the cost function $E(\mathbf{W})$ with respect to the real parts and the imaginary parts of $\mathbf{a}_q$ are

$$\begin{aligned}
\frac{\partial E(\mathbf{W})}{\partial \mathbf{a}_q^R} &= \sum_{j=1}^{J}[\epsilon'_{j,R}(\mathbf{a}_0^R \cdot \mathbf{h}^j)(\sum_{k=1}^{Q} y_k^R \frac{\partial h_k^j}{\partial \mathbf{a}_q^R}) \\
&\quad + \epsilon'_{j,I}(\mathbf{a}_0^I \cdot \mathbf{h}^j)(\sum_{k=1}^{Q} y_k^I \frac{\partial h_k^j}{\partial \mathbf{a}_q^R})] \\
&= 2\sum_{j=1}^{J} \epsilon'_{j,R}(\mathbf{a}_0^R \cdot \mathbf{h}^j) y_q^R h_q^j\left((\mathbf{x}^{j,R} - \mathbf{a}_q^R) \odot \mathbf{b}_q \odot \mathbf{b}_q\right) \\
&\quad + 2\sum_{j=1}^{J} \epsilon'_{j,I}(\mathbf{a}_0^I \cdot \mathbf{h}^j) y_q^I h_q^j\left((\mathbf{x}^{j,R} - \mathbf{a}_q^R) \odot \mathbf{b}_q \odot \mathbf{b}_q\right),
\end{aligned} \tag{15}$$

$$\begin{aligned}
\frac{\partial E(\mathbf{W})}{\partial \mathbf{a}_q^I} &= \sum_{j=1}^{J} \epsilon'_{j,R}(\mathbf{a}_0^R \cdot \mathbf{h}^j)(\sum_{k=1}^{Q} y_k^R \frac{\partial h_k^j}{\partial \mathbf{a}_q^I}) \\
&\quad + \sum_{j=1}^{J} \epsilon'_{j,I}(\mathbf{a}_0^I \cdot \mathbf{h}^j)(\sum_{k=1}^{Q} y_k^I \frac{\partial h_k^j}{\partial \mathbf{a}_q^I}) \\
&= 2\sum_{j=1}^{J} \epsilon'_{j,R}(\mathbf{a}_0^R \cdot \mathbf{h}^j) y_q^R h_q^j\left((\mathbf{x}^{j,I} - \mathbf{a}_q^I) \odot \mathbf{b}_q \odot \mathbf{b}_q\right) \\
&\quad + 2\sum_{j=1}^{J} \epsilon'_{j,I}(\mathbf{a}_0^I \cdot \mathbf{h}^j) y_q^I h_q^j\left((\mathbf{x}^{j,I} - \mathbf{a}_q^I) \odot \mathbf{b}_q \odot \mathbf{b}_q\right).
\end{aligned} \tag{16}$$

Similarly, the partial gradient of the error function $E(\mathbf{W})$ with respect to $\mathbf{b}_q$ are

$$\begin{aligned}
\frac{\partial E(\mathbf{W})}{\partial \mathbf{b}_q} &= \sum_{j=1}^{J} \epsilon'_{j,R}(\mathbf{a}_0^R \cdot \mathbf{h}^j)(\sum_{k=1}^{Q} y_k^R \frac{\partial h_k^j}{\partial \mathbf{b}_q}) \\
&\quad + \sum_{j=1}^{J} \epsilon'_{j,I}(\mathbf{a}_0^I \cdot \mathbf{h}^j)(\sum_{k=1}^{Q} y_k^I \frac{\partial h_k^j}{\partial \mathbf{b}_q})
\end{aligned}$$

$$
\begin{aligned}
&= -2\sum_{j=1}^{J}\epsilon'_{j,R}(\mathbf{a}_0^R\cdot\mathbf{h}^j)y_q^R h_q^j\big[(\mathbf{x}^{j,R}-\mathbf{a}_q^R)\\
&\quad\odot(\mathbf{x}^{j,R}-\mathbf{a}_q^R)\odot\mathbf{b}_q\\
&\quad+(\mathbf{x}^{j,I}-\mathbf{a}_q^I)\odot(\mathbf{x}^{j,I}-\mathbf{a}_q^I)\odot\mathbf{b}_q\big]\\
&\quad-2\sum_{j=1}^{J}\epsilon'_{j,I}(\mathbf{a}_0^I\cdot\mathbf{h}^j)y_q^I h_q^j\big[(\mathbf{x}^{j,R}-\mathbf{a}_q^R)\\
&\quad\odot(\mathbf{x}^{j,R}-\mathbf{a}_q^R)\odot\mathbf{b}_q\\
&\quad+(\mathbf{x}^{j,I}-\mathbf{a}_q^I)\odot(\mathbf{x}^{j,I}-\mathbf{a}_q^I)\odot\mathbf{b}_q\big].
\end{aligned}\tag{17}
$$

Write $\mathbf{W}^t = ((\mathbf{a}_0^t)^T,\cdots,(\mathbf{a}_Q^t)^T,(\mathbf{b}_1^t)^T,\cdots,(\mathbf{b}_Q^t)^T)^T$, let $\mathbf{W}^0$ be arbitrarily chosen initial weights, and let $\Delta\mathbf{a}_n^{0,R} = \Delta\mathbf{a}_n^{0,I} = 0\,(n=0,1,\cdots,Q)$, $\Delta\mathbf{b}_q^0 = 0\,(q=1,\cdots,Q)$, the SCNFA updates the real parts and the imaginary parts of the weights separately:

$$
\begin{cases}
\Delta\mathbf{a}_n^{t,R} = \mathbf{a}_n^{t+1,R}-\mathbf{a}_n^{t,R} = -\eta\dfrac{\partial E(\mathbf{W}^t)}{\partial\mathbf{a}_n^R}+\tau_n^{t,R}\Delta\mathbf{a}_n^{t,R};\\
\Delta\mathbf{a}_n^{t,I} = \mathbf{a}_n^{t+1,I}-\mathbf{a}_n^{t,I} = -\eta\dfrac{\partial E(\mathbf{W}^t)}{\partial\mathbf{a}_n^I}+\tau_n^{t,I}\Delta\mathbf{a}_n^{t,I},\\
\qquad n=0,1,\cdots,Q;\\
\Delta\mathbf{b}_q^t = \mathbf{b}_q^{t+1}-\mathbf{b}_q^t = -\eta\dfrac{\partial E(\mathbf{W}^t)}{\partial\mathbf{b}_q}+\tau_q^t\Delta\mathbf{b}_q^t,\\
\qquad q=1,\cdots,Q,
\end{cases}\tag{18}
$$

where $\eta\in(0,1)$ is the learning rate, $\tau_n^{t,R}, \tau_n^{t,I}(n=0,1,\cdots,Q)$ and $\tau_q^t(q=1,\cdots,Q)$ are the momentum parameters. For simplicity, let us denote

$$
\mathbf{p}_n^{t,R} = \frac{\partial E(\mathbf{W}^t)}{\partial\mathbf{a}_n^R},\quad \mathbf{p}_n^{t,I} = \frac{\partial E(\mathbf{W}^t)}{\partial\mathbf{a}_n^I},\ \boldsymbol{\kappa}_q^t = \frac{\partial E(\mathbf{W}^t)}{\partial\mathbf{b}_q}.\tag{19}
$$

Then (18) can be rewritten as

$$
\begin{cases}
\Delta\mathbf{a}_n^{t+1,R} = \tau_n^{t,R}\Delta\mathbf{a}_n^{t,R}-\eta\mathbf{p}_n^{t,R};\\
\Delta\mathbf{a}_n^{t+1,I} = \tau_n^{t,I}\Delta\mathbf{a}_n^{t,I}-\eta\mathbf{p}_n^{t,I};\\
\Delta\mathbf{b}_q^{t+1} = \tau_q^t\Delta\mathbf{b}_q^t-\eta\boldsymbol{\kappa}_q^t.
\end{cases}\tag{20}
$$

Similar to BPM (BP with Momentum) ([19]), we choose the adaptive momentum parameters $\tau_n^{t,R}$, $\tau_n^{t,I}$ and τ_q^t as follows:

$$
\tau_n^{t,R} = \begin{cases}\dfrac{\tau\|\mathbf{p}_n^{t,R}\|}{\|\Delta\mathbf{a}_n^{t,R}\|}, & \text{if } \|\Delta\mathbf{a}_n^{t,R}\|\neq 0;\\ 0, & \text{else},\end{cases}\tag{21}
$$

$$
\tau_n^{t,I} = \begin{cases}\dfrac{\tau\|\mathbf{p}_n^{t,I}\|}{\|\Delta\mathbf{a}_n^{t,I}\|}, & \text{if } \|\Delta\mathbf{a}_n^{t,I}\|\neq 0;\\ 0, & \text{else},\end{cases}\tag{22}
$$

$$
\tau_q^{t} = \begin{cases}\dfrac{\tau\|\boldsymbol{\kappa}_q^{t}\|}{\|\Delta\mathbf{b}_q^{t}\|}, & \text{if } \|\Delta\mathbf{b}_q^{t}\|\neq 0;\\ 0, & \text{else},\end{cases}\tag{23}
$$

where $\tau\in(0,1)$ is a constant parameter. The above three formulas demonstrates self-adaptive updating iteratively by compositing the current weights coefficients with the previous-step updated coefficients. In the case of nonzero momentum coefficients, they are redefined as positive value to speed up training by renewing momentum. Otherwise the momentum is expected as zero to maintain the error downhill, being reduced to the gradient-based algorithm.

C. ALGORITHM FLOW

Algorithm 1 Next, the SCNFA With Adaptive Momentum is Illustrated in Brief

Input: The data set $\{\mathrm{x}^j, \mathrm{O}^j\}_{j=1}^J$ will be learnt
Output: Parameters of the network: center ($\mathbf{a}_q$), reciprocal of width ($\mathbf{b}_q$) of Gaussian membership ($q=1,2,\ldots,Q$) and conclusion parameters ($\mathbf{a}_0$)
Begin
  Initialize $\mathbf{a}_q$, $\mathbf{b}_q$ and $\mathbf{a}_0$
  Select the learning rate η and momentum parameter τ
  Select the number of fuzzy rules Q
  Select the maximum training steps N
 for $t=1,2,\ldots,N$ **do**
  Compute the Gaussian function A_{lq} using Eq. (2)
  Compute the rule agreement h_q using Eq. (4)
  Compute the network output d using Eq. (5)
  Compute the error using Eq. (7)
  Update parameters $\mathbf{a}_q$, $\mathbf{b}_q$ and $\mathbf{a}_0$ as given in Eqs. (15)-(23)
 end
end

III. MAIN CONVERGENCE RESULTS

In this section we present a convergence theorem of the learning iteration process (18)–(23). Its proof has been relegated to the Appendix. Some sufficient assumptions for the convergence are given as follows:

$(A1)$ There exists a constant $C_0>0$ such that $\max\{\|\mathbf{a}_0^{t,R}\|,\|\mathbf{a}_0^{t,I}\|\}\le C_0$, $\max\{\|\mathbf{a}_q^{t,R}\|,\|\mathbf{a}_q^{t,I}\|\}\le C_0$ and $\|\mathbf{b}_q^t\|\le C_0$ for all $q=1,2,\ldots,Q,\ t=1,2,\ldots;$

$(A2)$ The set $\mathbf{\Omega} = \{\mathbf{W}|\frac{\partial E(\mathbf{W})}{\partial\mathbf{a}_0^R}=0,\frac{\partial E(\mathbf{W})}{\partial\mathbf{a}_0^I}=0,\frac{\partial E(\mathbf{W})}{\partial\mathbf{a}_q^R}=0,\frac{\partial E(\mathbf{W})}{\partial\mathbf{a}_q^I}=0,\frac{\partial E(\mathbf{W})}{\partial\mathbf{b}_q}=0, q=1,2,\ldots,Q\}$ contains finite points, where Ω is a compact set.

Theorem 1: Suppose Assumption $(A1)$ is valid, and that $\{\mathbf{W}^t\}$ is the weight vector sequence generated by (18)–(23), with arbitrary initial value $\mathbf{W}^0$. Then, there exists a constant $C>0$ such that for $0<g<1$, $\tau=g\eta$ and $\eta\le\frac{1-g}{C(1+g)^2}$, then we have the following weak convergence

(i) $E(\mathbf{W}^{t+1})\le E(\mathbf{W}^t),\quad t=0,1,2,\ldots;$

(ii) There is $E^*\ge 0$, such that $\lim_{t\to\infty}E(\mathbf{W}^t)=E^*;$

Furthermore, if Assumption $(A2)$ is also satisfied, then we get the strong convergence, that is, there exists a point $\mathbf{W}^\star\in\Omega$ such that

(iii) $\lim_{t\to\infty}\mathbf{W}^t=\mathbf{W}^*.$

IV. SIMULATIONS

The performance of the proposed zero-order TSK system with the well defined adaptive momentum-weighted SCNFA is studied in this section. The simulation results support the validity of the theoretical conclusions and exhibit the high performance of the proposed algorithm, compared with SCNFA with no momentum (NM) and with constant momentum (CM). The effectiveness and convergence property of SCNFA with adaptive momentum (AM) are shown. The effectiveness is judged by how small the cost function is at the end of the training process, and by how fast the algorithm convergent, while the convergence property is detected by whether the error and the norm of gradient of the error function go to zero when the training course terminates. The convergence performance of the algorithm is demonstrated through the complex XOR benchmark problem.

Example (Complex XOR Benchmark): The complex XOR is commonly utilized in literature to evaluate the convergence capacity of the algorithms. In complex domain, the input and ideal output of each sample is given as [20]. In this simulation, 5 complex fuzzy rules were involved. The real and imaginary components of weights for fuzzy reasoning were randomly initialized out of an interval in the range $[-1, 1]$, the learning rate μ was set to 0.01, and the momentum coefficient τ was set as the legend of Fig. 2 shown. The maximum iteration steps was 500.

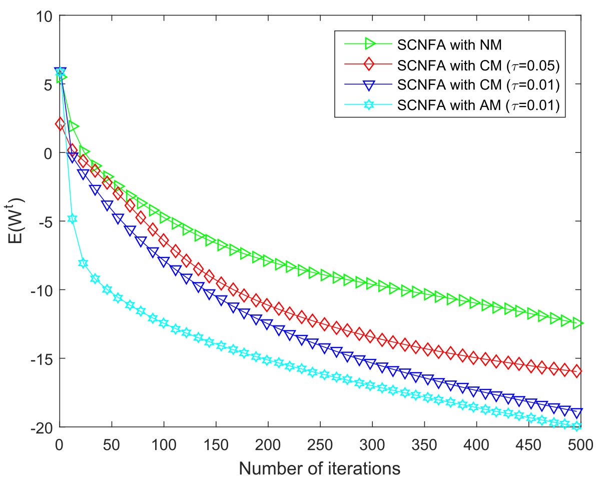


FIGURE 2. Error curves comparison of SCNFA with NM, CM and AM .

Fig. 2 also demonstrated that the error curve of FCNFA with AM was monotonously decreasing, which was consistent with the theoretical findings. Meanwhile, the comparisons of experimental results of the three algorithms were exhibited, which showed that FCNFA with AM performed best among all algorithms as for convergence speed and the terminated error. Specifically, the error curve of FCNFA with AM dropped fastest and deepest, bearing 0.024 in the middle of the training process i.e. 250 iterations compared with 0.039, 0.057 and 0.130. FCNFA with AM also behaved the least terminated error, bearing 0.009 at the end of the training process i.e. 500 iterations compared with 0.013, 0.025 and 0.056.

Basically the gradient-based neuro-fuzzy algorithm was nothing just a gradient descent method, which tried to minish the gap between the ideal and real outputs in an iterative manner. As for AM, the weights involved in the system were renewed so as to not only render the error decreased along a descent direction, but rather exploited the message benefited by the previous and the next preceding steps at each iteration. As can be observed, the AM has remarkable advantages in both accelerated convergence and least error value. It can be noticed that our approach very significantly accelerate the learning process in comparison with the conventional CM.

V. CONCLUSION

In this paper, a split-complex valued neuro-fuzzy algorithm (SCNFA) for fuzzy inference system has been developed to promote the potential capacities of TSK system. The momentum strategy successfully speeds up the convergence performance and decreasing the steepest descent error efficiently. The strong convergence of the SCNFA is strictly investigated. The convergence of the weight sequence of parameters is also given by adding a moderate condition. This improved version of adaptive momentum has been simulated to support the superiority in contrast with other methods.

APPENDIX

We first present a lemma, then use it to prove the strong convergence results.

Lemma 1: Suppose that $F : \mathbb{R}^{Q(2L+1)} \to \mathbb{R}^1$ is continuous and differentiable on a compact set $\boldsymbol{D} \subset \mathbb{R}^{Q(2L+1)}$, and that $\overline{\boldsymbol{\Omega}} = \{\varpi \in \boldsymbol{D} \mid \frac{\partial F(\varpi)}{\partial \varpi} = 0\}$ contains only finite points. If a sequence $\{\varpi^t\} \subset \boldsymbol{D}$ satisfies

$$\lim_{t\to\infty} \|\varpi^{t+1} - \varpi^t\| = 0, \qquad \lim_{t\to\infty} \left\| \frac{\partial F(\varpi^t)}{\partial \varpi} \right\| = 0,$$

then there exists a point $\varpi^\star \in \overline{\boldsymbol{\Omega}}$ such that $\lim_{t\to\infty} \varpi^t = \varpi^\star$.

Proof: This result is almost the same as in [15, Lemma 1], and the detail of the proof is omitted.

Proof of Theorem 1: Conclusions (i) and (ii) in Theorem 1 can be similarly proved as in [15].

Next, we prove Conclusion (iii) i.e. strong convergence. In the following, we suppose conditions of Theorem 1 are valid, let $\alpha = -(\tau - \eta + (C_2 + C_4)(\tau + \eta)^2)$, $\beta = -(\tau - \eta + (C_2 + C_3 + C_4)(\tau + \eta)^2)$, then there hold $\alpha \geq 0$, $\beta \geq 0$ and

$$\begin{aligned} E(\mathbf{W}^{t+1}) - E(\mathbf{W}^t) &\leq \big((\tau - \eta) + (C_2 + C_3 + C_4)(\tau + \eta)^2\big) \\ &\quad \times \Big(\sum_{q=1}^{Q}(\|\mathbf{p}_q^{t,R}\|^2 + \|\mathbf{p}_q^{t,I}\|^2) + \sum_{q=1}^{Q}\|\boldsymbol{\kappa}_q^t\|^2\Big) \\ &\quad + ((\tau - \eta) + (C_2 + C_4)(\tau + \eta)^2) \\ &\quad \times (\|\mathbf{p}_0^{t,R}\|^2 + \|\mathbf{p}_0^{t,I}\|^2). \end{aligned} \tag{24}$$

Then we have

$$
\begin{aligned}
E(\mathbf{W}^{t+1}) \leq E(\mathbf{W}^{t}) &- \alpha(\|\mathbf{p}_0^{t,R}\|^2 + \|\mathbf{p}_0^{t,I}\|^2) \\
&- \beta \sum_{q=1}^{Q} (\|\mathbf{p}_q^{t,R}\|^2 + \|\mathbf{p}_q^{t,I}\|^2 + \|\boldsymbol{\kappa}_q^{t}\|^2) \\
&\leq \ldots \leq E(\mathbf{W}^0) - \sum_{k=0}^{t} [\alpha(\|\mathbf{p}_0^{k,R}\|^2 + \|\mathbf{p}_0^{k,I}\|^2) \\
&+ \beta \sum_{q=1}^{Q} (\|\mathbf{p}_q^{k,R}\|^2 + \|\mathbf{p}_q^{k,I}\|^2 + \|\boldsymbol{\kappa}_q^{k}\|^2)].
\end{aligned}
$$

Since $E(\mathbf{W}^{t+1}) \geq 0$, there holds

$$
\begin{aligned}
&\sum_{k=0}^{t} [\alpha(\|\mathbf{p}_0^{k,R}\|^2 + \|\mathbf{p}_0^{k,I}\|^2) \\
&\qquad + \beta \sum_{q=1}^{Q} (\|\mathbf{p}_q^{k,R}\|^2 + \|\mathbf{p}_q^{k,I}\|^2 + \|\boldsymbol{\kappa}_q^{k}\|^2)] \leq E(\mathbf{W}^0).
\end{aligned}
$$

Letting $t \to \infty$ then

$$
\begin{aligned}
&\sum_{k=0}^{\infty} [\alpha(\|\mathbf{p}_0^{k,R}\|^2 + \|\mathbf{p}_0^{k,I}\|^2) \\
&\qquad + \beta \sum_{q=1}^{Q} (\|\mathbf{p}_q^{k,R}\|^2 + \|\mathbf{p}_q^{k,I}\|^2 + \|\boldsymbol{\kappa}_q^{k}\|^2)] \leq E(\mathbf{W}^0) < \infty.
\end{aligned}
$$

Hence, there holds

$$
\begin{aligned}
&\lim_{t\to\infty} (\|\mathbf{p}_n^{t,R}\|^2 + \|\mathbf{p}_n^{t,I}\|^2) = \lim_{t\to\infty} \|\boldsymbol{\kappa}_q^{t}\|^2 = 0, \\
&\qquad n = 0, 1 \cdots, Q, \; q = 1, \cdots, Q, \qquad (25)
\end{aligned}
$$

which implies

$$
\begin{aligned}
\lim_{t\to\infty} \|\frac{\partial E(\mathbf{W}^t)}{\partial \mathbf{a}_n^R}\| &= \lim_{t\to\infty} \|\frac{\partial E(\mathbf{W}^t)}{\partial \mathbf{a}_n^I}\| \\
&= \lim_{t\to\infty} \|\frac{\partial E(\mathbf{W}^t)}{\partial \mathbf{b}_q}\| = 0. \qquad (26)
\end{aligned}
$$

Finally, we use (18)-(20) and (26) to obtain

$$
\begin{aligned}
\lim_{t\to\infty} \|\mathbf{a}_n^{t+1,R} - \mathbf{a}_n^{t,R}\| &= \lim_{t\to\infty} \|\mathbf{a}_n^{t+1,I} - \mathbf{a}_n^{t,I}\| \\
&= \lim_{t\to\infty} \|\mathbf{b}_q^{t+1} - \mathbf{b}_q^{t}\| = 0. \qquad (27)
\end{aligned}
$$

From Assumption (A2), (26), (27) and Lemma IV, we obtain that there is a W^* such that $\lim_{t\to\infty} \mathbf{W}^t = \mathbf{W}^*$. The statement (*iii*) is proved. We thus complete the proof.

REFERENCES

[1] T. Adali and P. J. Schreier, "Optimization and estimation of complex-valued signals: Theory and applications in filtering and blind source separation," *IEEE Trans Signal Process. Mag.*, vol. 31, no. 5, pp. 112–128, Sep. 2014.

[2] D. P. Mandic and V. S. Lee Goh, *Complex Valued Nonlinear Adaptive Filters: Noncircularity, Widely Linear and Neural Models*. Hoboken, NJ, USA: Wiley, 2009.

[3] T. Nitta, "The computational power of complex-valued neuron," in *Artificial Neural Networks and Neural Information Processing—ICANN/ICONIP*. Berlin, Germany: Springer, 2003, pp. 993–1000.

[4] G. Velmurugan, R. Rakkiyappan, V. Vembarasan, J. Cao, and A. Alsaedi, "Dissipativity and stability analysis of fractional-order complex-valued neural networks with time delay," *Neural Netw.*, vol. 86, pp. 42–53, Feb. 2017.

[5] M. Rajendra and K. Shankar, "Improved complex-valued radial basis function (ICRBF) neural networks on multiple crack identification," *Appl. Soft Comput.*, vol. 28, pp. 285–300, Mar. 2015.

[6] M. Kobayashi, "Dual–numbered Hopfield neural networks," *IEEJ Trans. Elect. Electron.*, vol. 13, no. 2, pp. 280–284, 2018.

[7] Y. Liu, L. Li, and D. Yang, "Boundedness and convergence of split complex gradient descent algorithm with momentum and regularizer for TSK fuzzy models," *Neurocomputing*, vol. 311, no. 15, pp. 270–278, 2018.

[8] D. Xu, J. Dong, and C. Zhang, "Convergence of quasi-newton method for fully complex-valued neural networks," *Neural Process. Lett.*, vol. 46, no. 3, pp. 961–968, 2017.

[9] W. Rudin, *Real and Complex Analysis*. New York, NY, USA: McGraw-Hill, 1987.

[10] S. S. Yang, S. Siu, and C. L. Ho, "Analysis of the initial values in split-complex backpropagation algorithm," *IEEE Trans. Neural Netw.*, vol. 19, no. 9, pp. 1564–1573, Sep. 2008.

[11] K. Subramanian, R. Savitha, and S. Suresh, "A metacognitive complex-valued interval type-2 fuzzy inference system," *IEEE Trans Neur. Netw. Learn. Syst.*, vol. 25, no. 9, pp. 1659–1672, Sep. 2014.

[12] C. Li and T.-W. Chiang, "Function approximation with complex neuro-fuzzy system using complex fuzzy sets—A new approach," *New Gener. Comput.*, vol. 29, pp. 261–276, Jul. 2011.

[13] A. A. Hameed, B. Karlik, and M. S. Salman, "Back-propagation algorithm with variable adaptive momentum," *Knowl. Based Syst.*, vol. 114, pp. 79–87, Dec. 2016.

[14] M. L. Sarigül and M. Avci, "Performance comparison of different momentum techniques on deep reinforcement learning," *J. Inf. Telecommun.*, vol. 2, no. 2, pp. 205–216, 2018.

[15] Y. Liu and D. 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.*, vol. 319, pp. 28–49, Jul. 2017.

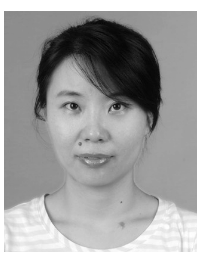
[16] Y. Liu, D. Yang, N. Nan, L. Guo, and J. Zhang, "Strong convergence analysis of batch gradient-based learning algorithm for training Pi-Sigma network based on TSK fuzzy models," *Neural Process. Lett.*, vol. 43, no. 3, pp. 745–758 2016.

[17] W. Wu, N. Zhang, and Z. Li, "Convergence of gradient method with momentum for back-propagation neural networks," *J. Comput. Math.*, vol. 26, no. 4, pp. 613–623, 2008.

[18] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, no. 1, pp. 116–132, Jan./Feb. 1985.

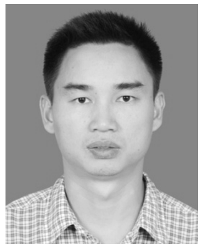
[19] N. Zhang, W. Wu, and G. Zheng, "Convergence of gradient method with momentum for two-layer feedforward neural networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 2, pp. 522–525, Mar. 2006.

[20] H. Zhang, D. Xu, and Y. Zhang, "Boundedness and convergence of split-complex back-propagation algorithm with momentum and penalty," *Neural Process. Lett.*, vol. 39, no. 3, pp. 297–307, Jun. 2014.



YAN LIU received the Ph.D. degree in computational mathematics from the Dalian University of Technology in 2012. She is currently with the School of Information Sciences and Engineering, Dalian Polytechnic University, Dalian, China. Her research interests include machine learning, fuzzy neural networks, and regularization theory.

FANG LIU is currently a Graduate Student with the School of Mathematical Sciences, Dalian University of Technology. Her major interests include computational mathematics. And her research interests include machine learning and neural networks theory.



LONG LI received the M.S. and Ph.D. degrees in computational mathematics from the Dalian University of Technology, Dalian, China, in 2007 and 2010, respectively. He is currently with the College of Mathematics and Statistics, Hengyang Normal University, Hengyang, China. His research interests include numerical analysis, neural network computation, and fuzzy system.