INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 8, Number 3, Pages 484–495

CONVERGENCE OF GRADIENT METHOD FOR DOUBLE PARALLEL FEEDFORWARD NEURAL NETWORK

JIAN WANG, WEI WU, ZHENGXUE LI, AND LONG LI

Abstract. The deterministic convergence for a Double Parallel Feedforward Neural Network (DPFNN) is studied. DPFNN is a parallel connection of a multi-layer feedforward neural network and a single layer feedforward neural network. Gradient method is used for training DPFNN with finite training sample set. The monotonicity of the error function in the training iteration is proved. Then, some weak and strong convergence results are obtained, indicating that the gradient of the error function tends to zero and the weight sequence goes to a fixed point, respectively. Numerical examples are provided, which support our theoretical findings and demonstrate that DPFNN has faster convergence speed and better generalization capability than the common feedforward neural network.

Key Words. Double parallel feedforward neural network, gradient method, monotonicity, convergence.

1. Introduction

A Double Parallel Feedforward Neural Network (DPFNN) is a parallel connection of a multi-layer feedforward neural network and a single layer feedforward neural network. In a DPFNN, the output nodes not only receive the recodification of the external information through the hidden nodes, but also receive the external information itself directly through the input nodes. DPFNN involves a paratactic relationship between linear and nonlinear mappings [4, 1]. As in the case for the common feedforward neural networks [18, 13, 19, 20], the most widely used learning method for DPFNN remains to be the gradient method [17, 10, 15, 2]. It is shown (cf. [5]) that the training speed and accuracy are greatly improved for DPFNN compared with corresponding multi-layer feedforward neural networks [8, 12, 11, 3, 9, 7]. A double parallel feedforward process neural network with similar structure and updating rule as DPFNN is proposed in [22]. In [16], an alternate learning iterative algorithm for DPFNN is presented. The truncation error caused by word length on the accuracy of DPFNN is analyzed in [6].

We are concerned in this paper with the convergence of the gradient method for training DPFNN. In particular, we first prove the monotonicity of the error function in the gradient learning iteration for DPFNN. Then, some weak and strong convergence results are obtained, indicating that the gradient of the error function tends to zero and the weight sequence goes to a fixed point, respectively. Some supporting numerical examples are also provided, which support our theoretical

Received by the editors May 4, 2009 and, in revised form, March 22, 2011.
2000 *Mathematics Subject Classification.* 68W40, 92B20, 62M45.
This research was supported by the National Natural Science Foundation of China (No.10871220).

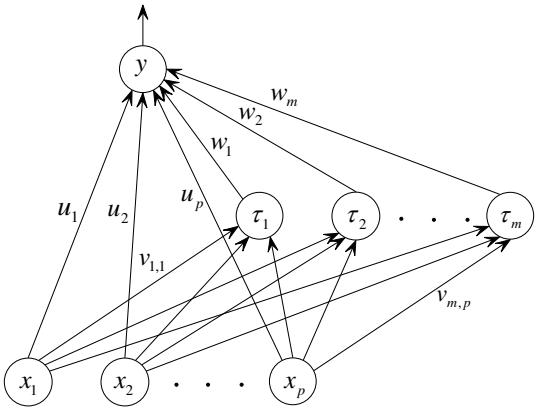

FIGURE 1. Topological Structure of DPFNN.

findings and demonstrate that DPFNN has faster convergence speed and better generalization capability than the common feedforward neural network.

The rest part of this paper is organized as follows. The structure of and the gradient method for DPFNN are introduced in Section 1. In Section 2 the convergence results are presented. Section 3 provides a few numerical examples to support our theoretical findings. Some brief conclusions are drawn in Section 4. Finally, an appendix is given, in which the details of the proof are gathered.

2. Double Parallel Feedforward Neural Networks

Figure 1 shows the DPFNN structure considered in this paper. It is a three-layer network with p input nodes, m hidden nodes and 1 output node. We denote the weight vector connecting the hidden layer and the output layer by $\mathbf{w} = (w_1, \cdots, w_m)^T \in \mathbb{R}^m$, and the weight matrix connecting the input layer and the hidden layer by $\mathbf{V} = (v_{i,j})_{m\times p}$, where $\mathbf{v}_i = (v_{i,1}, \cdots, v_{i,p})^T \in \mathbb{R}^p$ is the weight vector connecting the input layer and the i-th node of the hidden layer. Similarly, we denote the weight vector connecting the input layer and the output layer by $\mathbf{u} = (u_1, \cdots, u_p)^T \in \mathbb{R}^p$.

For simplicity, all the weight vectors are incorporated into a total weight vector $\mathbf{W} = (\mathbf{u}^T, \mathbf{v}_1^T, \cdots, \mathbf{v}_m^T, \mathbf{w}^T)^T \in \mathbb{R}^{p+mp+m}$. Let $g : R \to R$ be an activation function for the hidden and the output layers. For any $\mathbf{z} = (z_1, \cdots, z_m)^T \in \mathbb{R}^m$, we define

$$G(\mathbf{z}) = (g(z_1), g(z_2), \cdots, g(z_m))^T \in \mathbb{R}^m. \tag{1}$$

For any given input vector $\mathbf{x} \in \mathbb{R}^p$, the actual output $y \in \mathbb{R}$ of the neural system is computed by

$$y = g(\mathbf{w} \cdot G(\mathbf{V}\mathbf{x}) + \mathbf{u} \cdot \mathbf{x}). \tag{2}$$

We remark that the bias terms should be involved in the neural system. However, following a common strategy, we set the last component of, say, the input vector $\mathbf{x}$ to be -1, and so the last component of $\mathbf{v}_i$ corresponds to the bias term. This strategy allows us not to write explicitly the bias terms in the description of our problem.

For a given set of training samples $\left\{\mathbf{x}^j, O^j\right\}_{j=1}^J \subset \mathbb{R}^p \times \mathbb{R}$ supplied to the neural network, the error function is defined as

$$E(\mathbf{W}) = \frac{1}{2}\sum_{j=1}^{J}\left(y^j - O^j\right)^2 = \sum_{j=1}^{J} g_j\left(\mathbf{w}\cdot G\left(\mathbf{V}\mathbf{x}^j\right) + \mathbf{u}\cdot\mathbf{x}^j\right), \tag{3}$$

where

$$g_j(t) = \frac{1}{2}\left(g(t) - O^j\right)^2. \tag{4}$$

The purpose of network learning is to find $\mathbf{W}^*$ such that

$$E(\mathbf{W}^*) = \min E(\mathbf{W}). \tag{5}$$

The gradient descent algorithm is often used to solve this optimization problem. There are two practical ways for the implementation of the gradient method: batch learning and online learning. This paper follows the batch learning approach. The partial derivatives of the error function $E(\mathbf{W})$ with respect to $\mathbf{u}$, $\mathbf{v}_i$ and $\mathbf{w}$ are given respectively by

$$E_{\mathbf{u}}(\mathbf{W}) = \sum_{j=1}^{J} g_j'\left(\mathbf{w}\cdot G\left(\mathbf{V}\mathbf{x}^j\right) + \mathbf{u}\cdot\mathbf{x}^j\right)\mathbf{x}^j, \tag{6}$$

$$E_{\mathbf{v}_i}(\mathbf{W}) = \sum_{j=1}^{J} g_j'\left(\mathbf{w}\cdot G\left(\mathbf{V}\mathbf{x}^j\right) + \mathbf{u}\cdot\mathbf{x}^j\right) w_i g'\left(\mathbf{v}_i\cdot\mathbf{x}^j\right)\mathbf{x}^j, \tag{7}$$

$$E_{\mathbf{w}}(\mathbf{W}) = \sum_{j=1}^{J} g_j'\left(\mathbf{w}\cdot G\left(\mathbf{V}\mathbf{x}^j\right) + \mathbf{u}\cdot\mathbf{x}^j\right) G\left(\mathbf{V}\mathbf{x}^j\right). \tag{8}$$

Let the initial value $\mathbf{W}^0$ be arbitrarily chosen. Then, the weights are refined by the following iteration process:

$$\mathbf{W}^{n+1} = \mathbf{W}^n + \Delta\mathbf{W}^n, n = 0, 1, 2, \cdots, \tag{9}$$

where $\Delta\mathbf{W}^n = \left((\Delta\mathbf{u}^n)^T, (\Delta\mathbf{v}_1^n)^T, \cdots, (\Delta\mathbf{v}_m^n)^T, (\Delta\mathbf{w}^n)^T\right)^T$, and

$$\Delta\mathbf{u}^n = -\eta\sum_{j=1}^{J} g_j'\left(\mathbf{w}^n\cdot G\left(\mathbf{V}^n\mathbf{x}^j\right) + \mathbf{u}^n\cdot\mathbf{x}^j\right)\mathbf{x}^j, \tag{10}$$

$$\Delta\mathbf{v}_i^n = -\eta\sum_{j=1}^{J} g_j'\left(\mathbf{w}^n\cdot G\left(\mathbf{V}^n\mathbf{x}^j\right) + \mathbf{u}^n\cdot\mathbf{x}^j\right) w_i^n g'\left(\mathbf{v}_i^n\cdot\mathbf{x}^j\right)\mathbf{x}^j, \tag{11}$$

$$\Delta\mathbf{w}^n = -\eta\sum_{j=1}^{J} g_j'\left(\mathbf{w}^n\cdot G\left(\mathbf{V}^n\mathbf{x}^j\right) + \mathbf{u}^n\cdot\mathbf{x}^j\right) G\left(\mathbf{V}^n\mathbf{x}^j\right), \tag{12}$$

where $\eta > 0$ is the learning rate.

3. Main Results

To analyze the convergence of the algorithm, we need the following assumptions.

$(A1)$ $|g(t)|$, $|g'(t)|$ and $|g''(t)|$ are uniformly bounded for any $t \in \mathbb{R}$.

$(A2)$ The weights $\{\mathbf{w}^n\}\,(n = 0, 1, \cdots)$ keep uniformly bounded in the training process.

$(A3)$ The set $\Omega_0 = \{\mathbf{W} \in \Omega : E_{\mathbf{W}}(\mathbf{W}) = 0\}$ contains finitely many points, where Ω is a bounded closed region. Now we are in a position to present the main theorem, and its detail proof is relegated to the Appendix.

Theorem 3.1. *Assume that Conditions (A1) and (A2) are valid and the learning rate η satisfies the formula (27) below. Then for arbitrary initial values $\mathbf{W}^0$, the sequence $\{E(\mathbf{W}^n)\}$ decreases monotonically:*

$$E\left(\mathbf{W}^{n+1}\right) \leq E\left(\mathbf{W}^n\right); \tag{13}$$

there exists $E^* \geq 0$ such that

$$\lim_{n\to\infty} E\left(\mathbf{W}^n\right) = E^*; \tag{14}$$

and there holds the following weak convergence:

$$\lim_{n\to\infty} \|E_{\mathbf{W}}\left(\mathbf{W}^n\right)\| = 0. \tag{15}$$

If, in addition, the assumption (A3) is also valid, then there holds the following strong convergence: there exists $\mathbf{W}^* \in \Omega_0$ such that

$$\lim_{n\to\infty} \mathbf{W}^n = \mathbf{W}^*. \tag{16}$$

4. Numerical Simulations

In the following subsections, we investigate the performance of DPFNN with batch gradient method by three simulation examples.

Table 1. 4-bit parity problem.

input				output	input				output
1	1	1	1	0	-1	1	-1	-1	1
-1	1	1	1	1	1	1	-1	-1	0
1	1	1	-1	1	1	-1	1	1	1
-1	1	1	-1	0	-1	-1	1	1	0
-1	-1	-1	1	1	1	-1	-1	1	0
1	-1	1	-1	0	-1	-1	1	-1	1
-1	-1	-1	-1	0	1	1	-1	1	1
1	-1	-1	-1	1	-1	1	-1	1	0

4.1. Example 1: Parity problem. Parity problem is a difficult classification problem. The famous XOR problem is just the 2-bit parity problem. In this example, we use the 4-bit parity problem to test the performance of DPFNN. Table 1 shows the inputs and desired outputs of the training samples. The network is of three layers with the structure 5-4-1, and the logistic activation function $g(t) = 1/(1 + e^{-t})$ is used for the hidden and output nodes. The initial weights are chosen stochastically in $[-0.2, 0.2]$ and the learning rate η is 0.2. The performance of the batch gradient method is shown in Figure 2. We see that $E(\mathbf{W})$ decreases monotonically and the norm of $E_{\mathbf{W}}(\mathbf{W})$ trends to zero, as depicted by the convergence theorem.

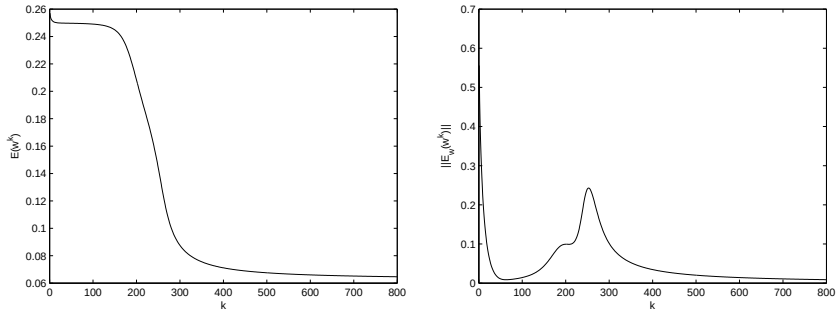

FIGURE 2. The error and the norm of gradient for Example 1.

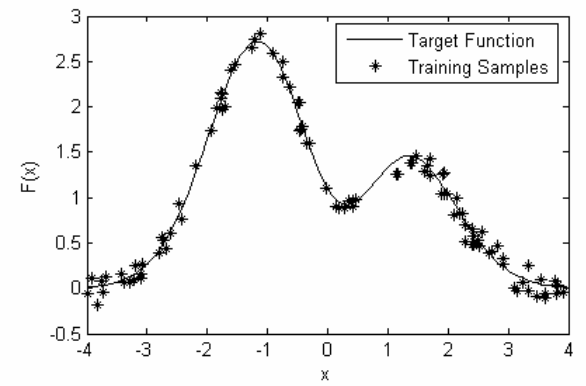

FIGURE 3. Target function and training samples for Example 2.

4.2. Example 2: An approximation problem. We consider the following function defined by Mackay (cf. [14]) to show the function approximation capability of DPFNN.

$$F(x) = 1.1\left(1 - x + 2x^2\right)\exp\left(-\frac{x^2}{2}\right). \tag{17}$$

The training samples are generated in the following manner: 100 input points x^i $(i = 1, \cdots, 100)$ are stochastically chosen from the interval $[-4, 4]$ with the corresponding outputs $F(x^i) + e_i$, where $e_i \in N(0, 0.1)$ is the noise and $N(0, 0.1)$ denotes the normal distribution with expectation and variance being 0 and 0.1, respectively. The target function and the training samples (marked by "$*$") are shown in Figure 3. For this example, we use the network with one input node (plus a bias node with fixed input -1), ten hidden nodes and one output node, respectively. The logistic activation function $g(t) = 1/(1 + e^{-t})$ is used for the hidden nodes, while the linear identity function $f(t) = t$ is used for the output node. The parameters in this example take the following values: $\eta = 0.003$, target error $\varepsilon = 0.5$, the maximum number of training epochs is 20,000, and the initial weights are chosen stochastically in $[-0.1, 0.1]$. Figure 4(a)-(b) show the good approximation to the target function.

4.3. Example 3: A real world prediction problem. A standard feedforward neural network model using back propagation learning (BPNN) for water diversion

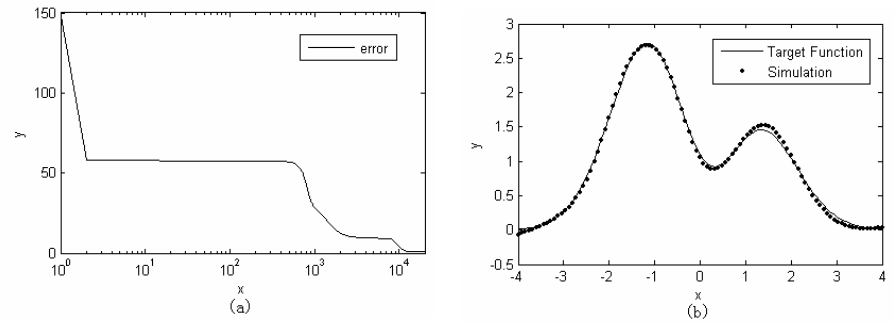

FIGURE 4. Error and simulations for Example 2.

TABLE 2. Water diversion demand from Yellow River and corresponding impact parameters.

year	average precipitation/ mm	irrigation area/ $\times 10^4 hm^2$	demand of He Nan/ $(m^3 \cdot hm^{-2})$	demand of Shan Dong/ $(m^3 \cdot hm^{-2})$	diversion demand/ $10^8 m^3$
1983	596	101.3	8 355	6 150	67.7
1984	704	138.1	7 785	4 215	66.8
1985	630	134.1	7 410	4 065	58.8
1986	381	145.4	8 490	5 670	89.1
1987	544	150.4	9 780	5 055	81.6
1988	432	156.4	9 165	5 280	89.8
1989	460	174.2	7 050	6 900	120.7
1990	850	166.2	8 355	4 845	85.2
1991	569	195.3	7 830	4 405	85.6
1992	514	202.7	7 080	5 100	100.6
1993	632	221.5	7 185	4 155	93.2
1994	694	199.6	6 045	4 380	79.3
1995	615	192.3	6 150	4 455	79.9

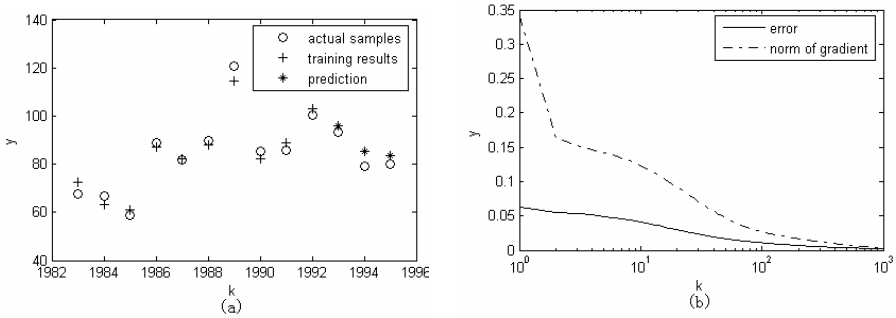

FIGURE 5. Effect of DPFNN for Example 3.

demand estimate is developed (cf. [2]). To investigate the effectiveness of DPFNN, we choose the same data (Table 2). The first 10 years data are regarded as the training set, while the latter 3 years data as the testing set. Firstly, we normalize each data vector $\mathbf{x} = (x_1, \cdots, x_{13})$ (i.e., each column of Table 2) by the following

TABLE 3. Comparison of predictions of DPFNN and BPNN.

sample types	year	diversion demand/ $10^8 m^3$	demand by BPNN/ $10^8 m^3$	relative error of BPNN/ %	demand by DPFNN/ $10^8 m^3$	relative error of DPFNN/ %
training	1983	67.7	67.7	0.00	70.13	3.59
	1984	66.8	66.8	0.00	61.77	-7.53
	1985	58.8	58.8	0.00	59.63	1.41
	1986	89.1	89.1	0.00	88.07	-1.15
	1987	81.6	81.6	0.00	82.63	1.26
	1988	89.8	89.8	0.00	88.83	-1.08
	1989	120.7	120.7	0.00	118.04	-2.20
	1990	85.2	85.2	0.00	84.37	-0.98
	1991	85.6	85.6	0.00	86.27	0.78
	1992	100.6	100.6	0.00	102.64	2.02
testing	1993	93.2	86.2	-7.51	90.8	-2.58
	1994	79.3	81.3	2.52	81.1	2.27
	1995	79.9	79.4	-0.63	80.3	0.55

formula ($x_{max} = \max\{x_p\}$ etc.):

$$\alpha \frac{x_p - x_{min}}{x_{max} - x_{min}} + \beta \Longrightarrow x_p,\ \alpha \in (0,1),\ \beta = \frac{1-\alpha}{2},\ p = 1, \cdots, 13. \tag{18}$$

In this example, we choose the parameter α as 0.9, then the values of the training data are transformed into the interval $[0.05, 0.95]$. The network is of three layers with the architecture 5-4-1, and the logistic activation function $g(t) = 1/(1+e^{-t})$ is used for both the hidden and output nodes. The learning rate η is 0.3 and the initial weights are chosen stochastically in $[-0.2, 0.2]$. The performance of the batch gradient method is shown in Figure 5(a), where the symbol "o" stands for the actual sample value, "+" indicates the training result, and "*" indicates the predicting result. From Figure 5(b), we observe that the error function decreases monotonically, and that the norm of $E_{\mathbf{W}}(\mathbf{W})$ trends to zero. It is clear from Table 3 that DPFNN has much stronger prediction capability than the common BPNN. In BPNN model, the average relative error and the maximum relative error are, respectively, 3.55% and 7.51% from 1993 to 1995, while the corresponding relative errors are 1.80% and 2.58% for DPFNN.

5. Conclusions

The batch gradient learning method for DPFNN with a hidden layer is considered. The learning rate η is a positive constant, and the initial guess of the weights are arbitrarily chosen. The monotonicity of the error function in the learning process is proved. Weak and strong convergence results are presented. Here the weak convergence means $\|E_{\mathbf{W}}(\mathbf{W}^n)\| \to 0$ as $n \to \infty$. The strong convergence $\mathbf{W}^n \to \mathbf{W}^*$ as $n \to \infty$ is proved in an additional condition that $E_{\mathbf{W}}(\mathbf{W})$ contains finitely many zero points, where $\mathbf{W}^*$ is a local minimum point of $E(\mathbf{W})$. Three numerical examples for the learning algorithm are provided to support our theoretical findings, and demonstrate that DPFNN has faster convergence rate and better generalization capability than common BPNN.

Appendix

We first present two lemmas, then use them to prove the main results. For sake of consistency, we write

$$\Delta \mathbf{w}^n = \mathbf{w}^{n+1} - \mathbf{w}^n,\ \Delta \mathbf{v}_i^n = \mathbf{v}_i^{n+1} - \mathbf{v}_i^n,\ \Delta \mathbf{u}^n = \mathbf{u}^{n+1} - \mathbf{u}^n, \tag{19}$$

$$G^{n,j} = G\left(\mathbf{V}^n \mathbf{x}^j\right),\ \psi^{n,j} = G^{n+1,j} - G^{n,j}, \tag{20}$$

$$\sigma_1^n = \|\Delta \mathbf{w}^n\|^2,\ \sigma_2^n = \sum_{i=1}^m \|\Delta \mathbf{v}_i^n\|^2,\ \sigma_3^n = \|\Delta \mathbf{u}^n\|^2. \tag{21}$$

Lemma 5.1. *Assume that Conditions (A1) and (A2) are valid, then there are* $C_i > 0$ *such that*

$$\|G(\mathbf{z})\| \le C_1,\ \mathbf{z} \in \mathbb{R}^m, \tag{22}$$

$$\|\psi^{n,j}\|^2 \le C_1 \sum_{i=1}^m \|\Delta \mathbf{v}_i^n\|^2,\ j = 1, \cdots, J;\ n = 1, 2, \cdots, \tag{23}$$

$$|g_j'(t)| \le C_2,\ |g_j''(t)| \le C_2,\ t \in \mathbb{R}. \tag{24}$$

Proof. By the definition of norm, we have that

$$\|G(\mathbf{z})\| \le \sqrt{m\left(\sup_{1\le i\le m}(|g(z_i)|)\right)^2} \le \sqrt{m}\sup_{t\in\mathbb{R}}|g(t)| \le C_1,$$

Using the mean value theorem and Assumption (A1), we conclude that

$$\begin{aligned}
\|\psi^{n,j}\|^2 &= \left\| \begin{pmatrix} g\left(\mathbf{v}_1^{n+1}\cdot\mathbf{x}^j\right) - g\left(\mathbf{v}_1^n\cdot\mathbf{x}^j\right) \\ \vdots \\ g\left(\mathbf{v}_m^{n+1}\cdot\mathbf{x}^j\right) - g\left(\mathbf{v}_m^n\cdot\mathbf{x}^j\right) \end{pmatrix} \right\|^2 \\
&\le \left(\sup_{t\in\mathbb{R}}|g'(t)|\max_{1\le j\le J}\|\mathbf{x}^j\|\right)^2 \sum_{i=1}^m \|\Delta \mathbf{v}_i^n\|^2 \\
&\le C_1 \sum_{i=1}^m \|\Delta \mathbf{v}_i^n\|^2,
\end{aligned}$$

where $C_1 = \max\left\{\sqrt{m}\sup_{t\in\mathbb{R}}|g(t)|,\ \left(\sup_{t\in\mathbb{R}}|g'(t)|\max_{1\le j\le J}\|\mathbf{x}^j\|\right)^2\right\}$, and $t_{i,j,n}$ $(1 \le i \le m)$ lies between $\mathbf{v}_i^n \cdot \mathbf{x}^j$ and $\mathbf{v}_i^{n+1} \cdot \mathbf{x}^j$. By $(A1)$, we can easily obtain

$$|g_j'(t)| \le C_2,\ |g_j''(t)| \le C_2, t \in \mathbb{R};\ j = 1, 2, \cdots, J, \tag{25}$$

where $C_2 = \max\left\{\sup_{t\in\mathbb{R}}|(g(t) - O^j)g'(t)|,\ \sup_{t\in\mathbb{R}}[(g'(t))^2 + |(g(t) - O^j)g''(t)|]\right\}$. □

The following Lemma is an essential tool for proving the strong convergence, which is basically the same as Theorem 14.1.5 (cf. [21]). Its proof is thus omitted.

Lemma 5.2. *Let* $F : \Omega \subset \mathbb{R}^n \to \mathbb{R}^m$ $(n, m \ge 1)$ *be continuous for a bounded closed region* $(\Omega \subset \mathbb{R}^n)$*, and* $\Omega_0 = \{\mathbf{z} \in \Omega : F(\mathbf{z}) = 0\}$ *be finite. Let* $\{\mathbf{z}^k\} \subset \Omega$ *be a sequence satisfying*

(1) $\lim_{k\to\infty} F(\mathbf{z}^k) = 0$;
(2) $\lim_{k\to\infty} \|\mathbf{z}^{k+1} - \mathbf{z}^k\| = 0$.

Then, there exists a $\mathbf{z}^* \in \Omega_0$ *such that* $\lim_{k\to\infty} \mathbf{z}^k = \mathbf{z}^*$.

Next, we prove successively the conclusions (13)-(16) of the convergence theorem.

Proof to (13). Using Taylor formula, we have

$$\begin{aligned}
&g_j'\left(\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j\right)\mathbf{w}^n\cdot\psi^{n,j}\\
&=g_j'\left(\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j\right)\left(\sum_{i=1}^m w_i^n\left(g\left(\mathbf{v}_i^{n+1}\cdot\mathbf{x}^j\right)-g\left(\mathbf{v}_i^n\cdot\mathbf{x}^j\right)\right)\right)\\
&=g_j'\left(\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j\right)\cdot\\
&\qquad\left(\sum_{i=1}^m w_i^n g'\left(\mathbf{v}_i^n\cdot\mathbf{x}^j\right)\Delta\mathbf{v}_i^n\cdot\mathbf{x}^j+\frac{1}{2}\sum_{i=1}^m w_i^n g''\left(\widetilde{s}_{i,j,n}\right)\left(\Delta\mathbf{v}_i^n\cdot\mathbf{x}^j\right)^2\right).
\end{aligned}$$

where $\widetilde{s}_{i,j,n}$ lies between $\mathbf{v}_i^n\cdot\mathbf{x}^j$ and $\mathbf{v}_i^{n+1}\cdot\mathbf{x}^j$. Employing (11), we conclude that

$$\begin{aligned}
&\sum_{j=1}^J g_j'\left(\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j\right)\mathbf{w}^n\cdot\psi^{n,j}\\
&=\sum_{i=1}^m\sum_{j=1}^J g_j'\left(\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j\right)w_i^n g'\left(\mathbf{v}_i^n\cdot\mathbf{x}^j\right)\Delta\mathbf{v}_i^n\cdot\mathbf{x}^j+\delta_1\\
&=-\frac{1}{\eta}\sum_{i=1}^m\|\Delta\mathbf{v}_i^n\|^2+\delta_1,
\end{aligned}$$

where $\delta_1=\frac{1}{2}\sum_{i=1}^m\sum_{j=1}^J w_i^n g_j'\left(\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j\right)g''\left(\widetilde{s}_{i,j,n}\right)\left(\Delta\mathbf{v}_i^n\cdot\mathbf{x}^j\right)^2$.
By virtue of (10)-(12) and the mean value theorem, we obtain that

$$\begin{aligned}
&E\left(\mathbf{W}^{n+1}\right)-E\left(\mathbf{W}^n\right)\\
&=\sum_{j=1}^J\left(g_j\left(\mathbf{w}^{n+1}\cdot G^{n+1,j}+\mathbf{u}^{n+1}\cdot\mathbf{x}^j\right)-g_j\left(\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j\right)\right)\\
&=\sum_{j=1}^J g_j'\left(\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j\right)\left(\mathbf{w}^{n+1}\cdot G^{n+1,j}-\mathbf{w}^n\cdot G^{n,j}+\left(\mathbf{u}^{n+1}-\mathbf{u}^n\right)\cdot\mathbf{x}^j\right)\\
&\quad+\frac{1}{2}\sum_{j=1}^J g_j''\left(s_{n,j}\right)\left(\mathbf{w}^{n+1}\cdot G^{n+1,j}-\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^{n+1}\cdot\mathbf{x}^j-\mathbf{u}^n\cdot\mathbf{x}^j\right)^2
\end{aligned}$$

It is easy to obtain that

$$\begin{aligned}
&\mathbf{w}^{n+1}\cdot G^{n+1,j}-\mathbf{w}^n\cdot G^{n,j}+\left(\mathbf{u}^{n+1}-\mathbf{u}^n\right)\cdot\mathbf{x}^j\\
&=\Delta\mathbf{w}^n\cdot G^{n,j}+\mathbf{w}^n\cdot\psi^{n,j}+\Delta\mathbf{w}^n\cdot\psi^{n,j}+\Delta\mathbf{u}^n\cdot\mathbf{x}^j
\end{aligned}$$

Then, we get

$$\begin{aligned}
&E\left(\mathbf{W}^{n+1}\right)-E\left(\mathbf{W}^n\right)\\
&=-\frac{1}{\eta}\|\Delta\mathbf{w}^n\|^2-\frac{1}{\eta}\sum_{i=1}^m\|\Delta\mathbf{v}_i^n\|^2+\delta_1+\delta_2-\frac{1}{\eta}\|\Delta\mathbf{u}^n\|^2+\delta_3\\
&=-\frac{1}{\eta}\left(\|\Delta\mathbf{w}^n\|^2+\sum_{i=1}^m\|\Delta\mathbf{v}_i^n\|^2+\|\Delta\mathbf{u}^n\|^2\right)+\delta_1+\delta_2+\delta_3,
\end{aligned}$$

where $s_{n,j}$ lies between $\mathbf{w}^{n+1}\cdot G^{n+1,j}+\mathbf{u}^{n+1}\cdot\mathbf{x}^j$ and $\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j$,

$$\delta_2=\sum_{j=1}^{J}g_j'\left(\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^n\cdot\mathbf{x}^j\right)\Delta\mathbf{w}^n\cdot\psi^{n,j},$$

$$\delta_3=\frac{1}{2}\sum_{j=1}^{J}g_j''\left(s_{n,j}\right)\left(\mathbf{w}^{n+1}\cdot G^{n+1,j}-\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^{n+1}\cdot\mathbf{x}^j-\mathbf{u}^n\cdot\mathbf{x}^j\right)^2.$$

By $(A1)$, $(A2)$ and (25), we see that

$$\delta_1\le C_3\sum_{i=1}^{m}\|\Delta\mathbf{v}_i^n\|^2,$$

where $C_3=\frac{1}{2}JC_2\sup_{n\in\mathbb{N}}\|\mathbf{w}^n\|\sup_{t\in\mathbb{R}}|g''(t)|\max_{1\le j\le J}\|\mathbf{x}^j\|^2$. Similarly, using Lemma 5.1 and Cauchy-Schwartz Inequality, we conclude that

$$\begin{aligned}\delta_2&\le C_2\sum_{j=1}^{J}\|\Delta\mathbf{w}^n\|\|\psi^{n,j}\|\le\frac{C_2}{2}\sum_{j=1}^{J}\left(\|\Delta\mathbf{w}^n\|^2+\|\psi^{n,j}\|^2\right)\\&\le C_4\left(\|\Delta\mathbf{w}^n\|^2+\sum_{i=1}^{m}\|\Delta\mathbf{v}_i^n\|^2\right),\end{aligned}$$

where $C_4=\frac{1}{2}JC_2\left(1+C_1\right)$,

$$\begin{aligned}\delta_3&\le\frac{C_2}{2}\sum_{j=1}^{J}\|\mathbf{w}^{n+1}\cdot G^{n+1,j}-\mathbf{w}^n\cdot G^{n,j}+\mathbf{u}^{n+1}\cdot\mathbf{x}^j-\mathbf{u}^n\cdot\mathbf{x}^j\|^2\\&=\frac{C_2}{2}\sum_{j=1}^{J}\|\left(\mathbf{w}^{n+1}-\mathbf{w}^n\right)\cdot G^{n+1,j}+\mathbf{w}^n\cdot\left(G^{n+1,j}-G^{n,j}\right)+\left(\mathbf{u}^{n+1}-\mathbf{u}^n\right)\cdot\mathbf{x}^j\|^2\\&\le\frac{C_2}{2}\left(\max\{C_1,\sup_{n\in\mathbb{N}}\|\mathbf{w}^n\|,\sup_{1\le j\le J}\|\mathbf{x}^j\|\}\right)^2\sum_{j=1}^{J}\left(\|\Delta\mathbf{w}^n\|+\|\psi^{n,j}\|+\|\Delta\mathbf{u}^n\|\right)^2\\&\le C_5\left(\|\Delta\mathbf{w}^n\|^2+\sum_{i=1}^{m}\|\Delta v_i^n\|^2+\|\Delta\mathbf{u}^n\|^2\right),\end{aligned}$$

where $C_5=\frac{JC_2}{2}\left(1+C_1\right)^2\left(\max\{C_1,\sup_{n\in\mathbb{N}}\|\mathbf{w}^n\|\sup_{1\le j\le J}\|\mathbf{x}^j\|\}\right)^2$.
Let $C_6=C_3+C_4+C_5$, $\beta=\frac{1}{\eta}-C_6$, we have

$$\begin{aligned}E\left(\mathbf{W}^{n+1}\right)-E\left(\mathbf{W}^n\right)&\le-\left(\frac{1}{\eta}-C_6\right)\left(\|\Delta\mathbf{w}^n\|^2+\sum_{i=1}^{m}\|\Delta v_i^n\|^2+\|\Delta\mathbf{u}^n\|^2\right)\\&=-\left(\frac{1}{\eta}-C_6\right)\left(\sigma_1^n+\sigma_2^n+\sigma_3^n\right)=-\beta\left(\sigma_1^n+\sigma_2^n+\sigma_3^n\right).\end{aligned}\tag{26}$$

We require the learning rate η to satisfy

$$0<\eta<\frac{1}{C_6}.\tag{27}$$

Then we have

$$E\left(\mathbf{W}^{n+1}\right)\le E\left(\mathbf{W}^n\right).$$

The monotonicity of the error function is proved. □

Proof to (14). Note that $E\left(\mathbf{W}^n\right)\ge 0$ for any $n=0,1,2,\cdots$. By (13), we see that

the sequence $\{E(\mathbf{W}^n)\}$ monotonically decreases. Thus, there exists $E^* \geq 0$ such that

$$\lim_{n\to\infty} E(\mathbf{W}^n) = E^*.$$

□

Proof to (15). By (26), we have

$$\begin{aligned} E\left(\mathbf{W}^{n+1}\right) &\leq E(\mathbf{W}^n) - \beta\left(\sigma_1^n + \sigma_2^n + \sigma_3^n\right) \\ &\leq E\left(\mathbf{W}^{n-1}\right) - \beta\left(\sigma_1^{n-1} + \sigma_2^{n-1} + \sigma_3^{n-1}\right) - \beta\left(\sigma_1^n + \sigma_2^n + \sigma_3^n\right) \\ &\leq \cdots \leq E\left(\mathbf{W}^0\right) - \beta\sum_{i=0}^{n}\left(\sigma_1^i + \sigma_2^i + \sigma_3^i\right). \end{aligned}$$

Since $E(\mathbf{W}^n) \geq 0$ holds for any $n \geq 1$, then

$$\sum_{i=0}^{n}\beta\left(\sigma_1^i + \sigma_2^i + \sigma_3^i\right) \leq E\left(\mathbf{W}^0\right).$$

Using (6)-(12) and (21), taking $n \to \infty$, and exchanging indexes, we get

$$\beta\sum_{n=0}^{\infty}\left(\sigma_1^n + \sigma_2^n + \sigma_3^n\right) = \beta\eta^2\sum_{n=0}^{\infty}\|E_{\mathbf{W}}(\mathbf{W}^n)\|^2 \leq E\left(\mathbf{W}^0\right) < \infty. \tag{28}$$

where β and η are constants. Hence, we derive that

$$\lim_{n\to\infty}\|E_{\mathbf{W}}(\mathbf{W}^n)\| = 0.$$

The weak convergence is then proved. □

Proof to (16). By virtue of (15), we know that $\lim_{n\to\infty} E_{\mathbf{W}}(\mathbf{W}^n) = 0$. Using (9)-(12), we conclude that $\lim_{n\to\infty}\|\mathbf{W}^{n+1} - \mathbf{W}^n\| = 0$. Furthermore, by $(A3)$, the conditions of Lemma 5.2 are all satisfied. Then there exists $\mathbf{W}^* \in \Omega_0$ such that $\lim_{n\to\infty}\mathbf{W}^n = \mathbf{W}^*$. □

References

[1] O. K. Ersoy and D. Hong, Parallel, Self-Organizing, Hierarchical Neural Networks, IEEE T. Neural. Networ., 1(1990), no. 2, 167-178.

[2] G. Huang and R. He, Analyzing Water Diversion Demand for Irrigation Areas at Lower Reach of Yellow River with BP Neural Network Techniques, J. Irriga. Drain., 19(2000), no. 3, 20-23.

[3] K. Hornik, Approximation Capabilities of Multilayer Feedforward Networks, Neural. Networ., 4(1991), no. 2, 251-257.

[4] M. He, Theory, Application and Related Problems of Double Parallel Feedforward Neural Networks, Xi'an: Xidian University, 1993.

[5] M. He, Double Parallel Feedforward Neural Networks with Application to Simulation Study of Flight Fault Inspection, Acta. Aerona. ET. Astrona. Sinica., 15(1994), no. 7, 877-881.

[6] M. He, Error Analysis of Double Parallel Feedforward Neural Networks, J. Northwest. Poly. Univer., 15(1997), no. 1, 125-130.

[7] S. Haykin, Neural Networks, A Comprehensive Foundation. 2nd edition, Prentice Hall, Englewood Cliffs, N.J, 1999.

[8] C. G. Looney, Pattern Recognition Using Neural Networks, Oxford University Press, New York, 1997.

[9] M. Leshno, V. Y. Lin and A. Pinkus et al, Multilayer Feedforward Networks with a Nonpolynomial Activation Function Can Approximate Any Function, Neural. Networ., 6(1993), no. 6, 861-867.

[10] S. Luan, G. Ding and S. Zhong, Aeroengine Lubricating Oil Metal Elements Concentration Prediction Based on Double Parallel Process Neural Networks, Lubrica. Engineer., 37(2006), no. 5, 32-34.

[11] Y. C. Liang, D. P. Feng and H. P. Lee et al, Successive Approximation Training Algorithm for Feedforward Neural Networks, Neurocomputing, 42(2002)311-322.

[12] Y. C. Liang, H. P. Lee and S. P. Lim et al, Proper Orthogonal Decomposition and Its Application - Part II: Model Reduction for MEMS Dynamical Analysis, J. Sound. Vibration., 252(2002), no. 3, 527-544.
[13] Z. Li, W. Wu and Y. Tian, Convergence of An Online Gradient Method for Feedforward Neural Networks with Stochastic Inputs, J. Comput. Appl. Math., 163(2004), no. 1, 165-176.
[14] D. J. C. Mackay, Bayesian Interpolation, Neural Comput., 4(1992), no. 3, 415-447.
[15] X. Meng, G. Ding and L. Tang, Calculation for the Exhaust Enthalpy of a Steam Turbine Based on Parallel Connection Feed-forward Network, Turbine Technology, 68(2006), no. 1, 14-15.
[16] D. Wei, H. Xi and L. Zhao, Alternate Iterative Algorithm of Double Parallel Artificial Neural Network and Its Application, Mini-Micro Systems, 17(1996), no. 11, 65-68.
[17] D. Wei, L. Zhao and L. Tang, Study on Application of Parallel Connection Artificial Neural Networks to Hydraulic Turbine Test, Water. Res. Hydro. Eng., 6(2005)79-84.
[18] W. Wu, G. Feng and X. Li, Training Multilayer Perceptrons via Minimization of Sum of Ridge Functions, Adv. Comput. Math., 17(2002)331-347.
[19] W. Wu, G. Feng and Z. Li et al, Deterministic Convergence of An Online Gradient Method for BP Neural Networks, IEEE T. Neural. Networ., 16(2005), no. 3, 533-540.
[20] W. Wu, N. Zhang and Z. Li et al, Convergence of gradient method with momentum for back-propagation neural networks, J. Comput. Math., 26(2008), no. 4, 613-623.
[21] Y. Yuan and W. Sun, Optimization Theory and Methods. Science Press, Beijing, 2001.
[22] S. Zhong and G. Ding, Research on Double Parallel Feedforward Process Neural Networks and Its Application, Contr. Decis., 20(2005), no. 7, 764-768.

School of Mathematical Sciences, Dalian University of Technology, Dalian, 116024, P. R. China
E-mail: wuweiw@dlut.edu.cn

School of Mathematics and Computational Sciences, Petroleum University of China, Dongying, 257061, P. R. China
E-mail: wangjiannl@mail.dlut.edu.cn