A MODIFIED HIGHER-ORDER FEED FORWARD NEURAL NETWORK WITH SMOOTHING REGULARIZATION

Kh.Sh. Mohamed[*], *W. Wu*[*], *Y. Liu*[†]

Abstract: This paper proposes an offline gradient method with smoothing $L_{1/2}$ regularization for learning and pruning of the pi-sigma neural networks (PSNNs). The original $L_{1/2}$ regularization term is not smooth at the origin, since it involves the absolute value function. This causes oscillation in the computation and difficulty in the convergence analysis. In this paper, we propose to use a smooth function to replace and approximate the absolute value function, ending up with a smoothing $L_{1/2}$ regularization method for PSNN. Numerical simulations show that the smoothing $L_{1/2}$ regularization method eliminates the oscillation in computation and achieves better learning accuracy. We are also able to prove a convergence theorem for the proposed learning method.

Key words: *convergence, pi-sigma neural network, offline (batch) gradient method, smoothing $L_{1/2}$ regularization*

Received: July 30, 2016 **DOI:** 10.14311/NNW.2017.27.032
Revised and accepted: November 13, 2017

1. Introduction

Pi-sigma neural networks (PSNNs) [4, 10–12] as a kind of higher-order neural networks can provide more powerful mapping capability than conventional feedforward neural networks, and have been successfully applied to many applications such as the equalization of nonlinear satellite channels, the real classification of seafloor sediments, and the image coding. The PSNN was introduced by Ghosh and Shin [11]. It computes the product of sums of the input layer, instead of the sums of the products. And the weights connecting the product layers and the summation layers are fixed to 1.

There are two practical ways to accomplish the gradient weights updating in the network learning process: the online gradient approach, in which weights are updated promptly after a training sample is supplied into the network; and the offline gradient approach, where the network weights are updated after all the

[*] Khidir Shaib Mohamed – Corresponding author; Wei Wu; Dalian University of Technology, School of Mathematical Sciences, Dalian 116024, China, China E-mail: khshm7@yahoo.com wuweiw@dlut.edu.cn

[†] Yan Liu; Dalian Polytechnic University, School of Information Science and Engineering, Dalian 116034, China, China E-mail: liuyan@dlpu.edu.cn

training samples have been treated by the network (cf. [15, 16]). We shall use the offline gradient approach in this paper.

Various kinds of regularization terms have been used in many applications, such as the method in [2] in terms of weight decay [5, 6], the methods based on Akaike's information criterion (AIC) [3, 13], and the method employing model prior in the Bayesian structure [8]. These regularization terms are inserted into the standard cost function for network learning to improve the learning ability and to get a sparse network.

Many regularization terms take the form of the L_p norm of the weights, leading to the following new error function

$$E(\mathbf{W}) = \bar{E}(\mathbf{W}) + \lambda \|\mathbf{W}\|_p^p, \tag{1}$$

where $\bar{E}(\mathbf{W})$ is a usual error function depending on the weights $\mathbf{W}$ of the network, $\|\mathbf{W}\|_p = (\sum_{k=1}^{n} |w_k|^p)^{\frac{1}{p}}$ is the p-norm of the weights of the network, and λ is the regularization parameter. In particular, the L_0 regularizer is the earliest regularization method applied for variable selection, and its solution is the most sparse. But it is a combinatory optimization problem and is shown to be NP-hard [14]. The popular L_1 regularizer is discussed in [9]. The $L_{1/2}$ regularizer is suggested in [17] and, among many favourable properties, it is shown to result in better sparsity than the L_1 regularizer. A smoothing $L_{1/2}$ regularizer is proposed in [1, 7]. And in the numerical experiments, it behaves equally good as or even better than the L_1 regularizer and the standard $L_{1/2}$ regularizer for neural networks. Therefore, in this paper, we shall use the smoothing $L_{1/2}$ regularizer for the network regularization.

The organization of this paper is as follows. In section 2, we describe PSNN and the offline gradient method with smoothing $L_{1/2}$ regularizer. In Section 3, a convergence theorem is given. Section 4 contains some supporting simulation results. The proof of the convergence theorem is given in Section 5. Finally, a brief conclusion is provided in Section 6.

2. Offline gradient method with smoothing $L_{1/2}$ regularization

In this section, we describe the network structure of PSNN and the off-line gradient method with smoothing $L_{1/2}$ regularization.

2.1 Error function with $L_{1/2}$ regularization

Let p, n and 1 respectively be the dimensions of the input layer, the summation layer and the product layer of PSNN. Denote by $\omega_j = (\omega_{j1}, \ldots, \omega_{jp})^{\mathrm{T}} \in \mathbb{R}^p$ $(1 \leq j \leq n)$ the weight vector connecting the input layer and the k-th summing unit, and write $\omega = (\omega_1^{\mathrm{T}}, \ldots, \omega_n^{\mathrm{T}}) \in \mathbb{R}^{np}$. Note that the weights from summing units to product unit are fixed to 1. Let $g : \mathbb{R} \to \mathbb{R}$ be a given activation function. The topological structure of SPNN algorithm is shown in Fig. 1.

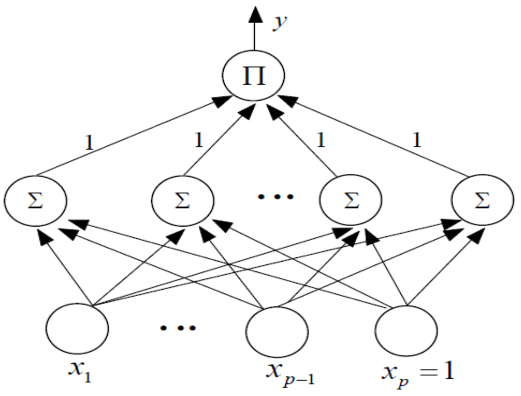

Fig. 1 *Pi-sigma neural network structure.*

For an input data vector $x \in \mathbb{R}^p$, the network output is

$$y = g\left(\prod_{j=1}^{n}\left(\sum_{i=1}^{p}(\omega_{ji}x_i)\right)\right) = g\left(\prod_{j=1}^{n}(\omega_j \cdot x)\right), \tag{2}$$

where $\omega_j \cdot x$ is the usual inner products of ω_j and x.

Suppose that we are supplied with a set of training samples $\{x^l, O^l\}_{l=1}^L \subset \mathbb{R}^p \times \mathbb{R}$, where O^l is the desired ideal output for the input x^l. By adding an $L_{1/2}$ regularization term into the the usual error function, the final error function takes the form

$$E(\omega) = \frac{1}{2}\sum_{l=1}^{L}\left(O^l - g\left(\prod_{j=1}^{n}(\omega_j \cdot x^l)\right)\right)^2 + \lambda\sum_{j=1}^{n}\sum_{i=1}^{p}|\omega_{ji}|^{1/2}. \tag{3}$$

The partial derivative of the above error function with respect to ω_{ji} is

$$E_{\omega_{ji}}(\omega) = \sum_{l=1}^{L}\delta_l\left(\prod_{j=1}^{n}(\omega_j \cdot x^l)\right)\prod_{\substack{k=1\\k\neq j}}^{n}(\omega_k \cdot x^l)x_i^l + \frac{\lambda sgn(\omega_{ji})}{2|\omega_{ji}|^{1/2}}, \tag{4}$$

$$i = 1, 2, \cdots, p;\ j = 1, 2\cdots, n;$$

where $\lambda > 0$ is the regularization parameter, $E_{\omega_{ji}}(\omega) = \frac{\partial E(\omega)}{\partial \omega_{ji}}$ and $\delta_l(t) = -(O^l - g(t))g'(t)$.

Starting from an arbitrary initial value ω^0 , the offline gradient method with $L_{1/2}$ regularization term updates the weights ω^m iteratively by

$$\omega_{ji}^{m+1} = \omega_{ji}^m + \triangle\omega_{ji}^m, \tag{5}$$

with

$$\triangle\omega_{ji}^m = -\eta\left(\sum_{l=1}^{L}\delta_l\left(\prod_{j=1}^{n}(\omega_j^m \cdot x^l)\right)\prod_{\substack{k=1\\k\neq j}}^{n}(\omega_k^m \cdot x^l)x_i^l + \frac{\lambda sgn(\omega_{ji}^m)}{2|\omega_{ji}^m|^{1/2}}\right), \tag{6}$$

$$i = 1, 2, \cdots, p;\ j = 1, 2, \cdots, n;\ m = 0, 1, \cdots$$

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

2.2 Error function with smoothing $L_{1/2}$ regularization

The usual $L_{1/2}$ regularization is non-differentiable at the origin, since it involves the absolute value function. We replace the absolute value function by a smooth function $f(t)$ defined below:

$$f(t) = \begin{cases} -t, & t \leq -a; \\ -\frac{1}{8a^3}t^4 + \frac{3}{4a}t^2 + \frac{3}{8}a, & -a < t < a; \\ t, & t \geq a; \end{cases} \tag{7}$$

where a is a small positive constant. Then we have

$$f'(t) = \begin{cases} -1, & t \leq -a; \\ -\frac{1}{2a^3}t^3 + \frac{3}{2a}t, & -a < t < a; \\ 1, & t \geq a; \end{cases}$$

$$f''(t) = \begin{cases} 0, & |t| \leq -a; \\ -\frac{3}{2a^3}t^2 + \frac{3}{2a}, & |t| \geq a; \end{cases}$$

and

$$f(t) \in [\frac{3}{8}a, \infty), f'(t) \in [-1, 1], f''(t) \in [0, \frac{3}{2a}].$$

Now, the new error function with smoothing $L_{1/2}$ regularization term is

$$E(\omega) = \frac{1}{2}\sum_{l=1}^{L}\left(O^l - g\left(\prod_{j=1}^{n}(\omega_j \cdot x^l)\right)\right)^2 + \lambda\sum_{j=1}^{n}\sum_{i=1}^{p} f^{1/2}(\omega_{ji}). \tag{8}$$

The partial derivative of the error function $E(\omega)$ in Eq. (8) with respect to ω_{ji} is

$$E_{\omega_{ji}}(\omega) = \sum_{l=1}^{L}\delta_l(\prod_{j=1}^{n}(\omega_j \cdot x))\prod_{\substack{k=1\\k\neq j}}^{n}(\omega_k \cdot x^l)x_i^l + \frac{\lambda f'(\omega_{ji})}{2f^{1/2}(\omega_{ji})}, \tag{9}$$

$$i = 1, 2, \cdots, p;\ j = 1, 2, \cdots, n.$$

Starting from an arbitrary initial value ω^0, the offline gradient method with smoothing $L_{1/2}$ regularization term updates the weights ω^m iteratively by

$$\omega_{ji}^{m+1} = \omega_{ji}^m + \triangle\omega_{ji}^m, \tag{10}$$

with

$$\triangle\omega_{ji}^m = -\eta\left[\sum_{l=1}^{L}\delta_l\left(\prod_{j=1}^{n}(\omega_j^m \cdot x)\right)\prod_{\substack{k=1\\k\neq j}}^{n}(\omega_k^m \cdot x^l)x_i^l + \frac{\lambda f'(\omega_{ji}^m)}{2f^{1/2}(\omega_{ji}^m)}\right], \tag{11}$$

$$i = 1, 2, \cdots, p;\ j = 1, 2, \cdots, n;\ m = 0, 1, \cdots$$

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

3. Main results

The following conditions will be used later on to prove the convergence theorem.

Proposition 1. $|\delta_l(t)|, |\delta_l'(t)| \leq C_0, |g(t)|, |g'(t)|, |g''(t)| \leq C_1, \forall t \in \mathbb{R}, 1 \leq l \leq L.$

Proposition 2. $\max\{\|x^l\|, |\omega_j^m \cdot x^l| \} \leq C_0, \forall 1 \leq j \leq n, 1 \leq l \leq L, m = 0, 1, \cdots$

Proposition 3. *The parameters η and λ are chosen to satisfy: $0 < \eta < \frac{2}{M\lambda + C}$ where $M = \frac{\sqrt{6}}{\sqrt[4]{a^3}}$ and $C = C_2/2 + C_3$.*

Proposition 4. *There exists a compact set Φ such that $\omega^m \in \Phi$ and the set $\Phi_0 \in \{\omega \in \Phi : E_\omega(\omega) = 0\}$ contains finite points.*

Theorem 5. *Let the error function $E(\omega)$ be defined by Eq. (8), and the weight $\{\omega^m\}$ be generated by the iteration algorithm Eq. (10) for an arbitrary initial value ω^0. If propositions 1-3 are valid, we have the following error estimates:*

(i) $E(\omega^{m+1}) \leq E(\omega^m)$

(ii) There exists $E^* \geq 0$ such that $\lim_{m\to\infty} E(\omega^m) = E^*$

(iii) $\lim_{m\to\infty} |E_{\omega_{ji}}(\omega^m)| = 0,\ i = 1, 2, \ldots, p;\ j = 1, 2, \ldots, n.$

Furthermore, if proposition 4 is also valid, we have the following strong convergence:

(iv) There exists a point $\omega^* \in \Phi_0$ such that $\lim_{m\to\infty} \omega^m = \omega^*$.

4. Numerical simulations

To illustrate the efficiency of our proposed learning method, we consider the numerical simulations of two classification problems (XOR and Parity problems) and two function approximation problems (Gabor and Mayas functions). We compared our batch gradient method with smoothing $L_{1/2}$ regularization (BGSL1/2) with the batch gradient method with $L_{1/2}$ regularization (BGL1/2) and batch gradient method with L_2 regularization (BGL2).

4.1 Classification problems

The activation function is chosen as $g(x) = 1/(1+e^{-x})$. Each of the three algorithms takes 10 trials for XOR and parity problems, respectively. Typical performances are shown in Figs. 2–5. For the XOR problem the network has 2 input nodes, 2 summation nodes, and 1 output node, with the learning rate $\eta = 0.06$ and the regularization parameter $\lambda = 0.001$. For the parity problem, the network has 4 input nodes, 5 summation nodes, and 1 output node, with the learning rate $\eta = 0.04$ and the regularization parameter $\lambda = 0.001$. For both the two cases, the initial weights are selected randomly within the interval $[-0.5, 0.5]$, and the maximum iteration number is 3000.

From Figs. 2–5, we see that the proposed learning method BGSL1/2 enjoys the best learning accuracy, while BGL2 is the worst. In particular, as predicted by

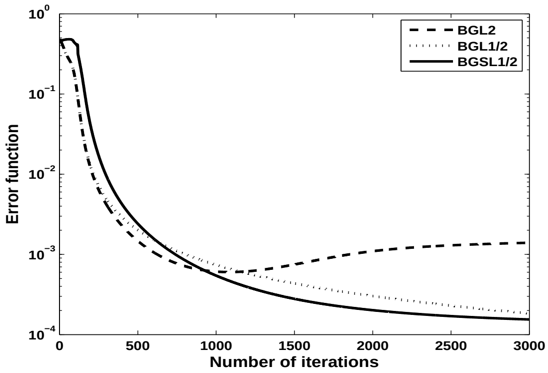

Fig. 2 *Learning errors of different algorithms for XOR problem.*

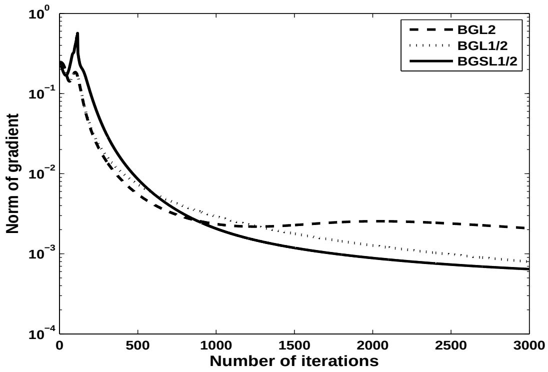

Fig. 3 *Norms of gradient of different algorithms for XOR problem.*

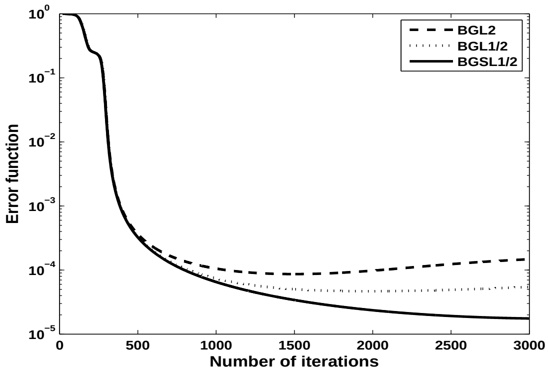

Fig. 4 *Learning errors of different algorithms for parity problem.*

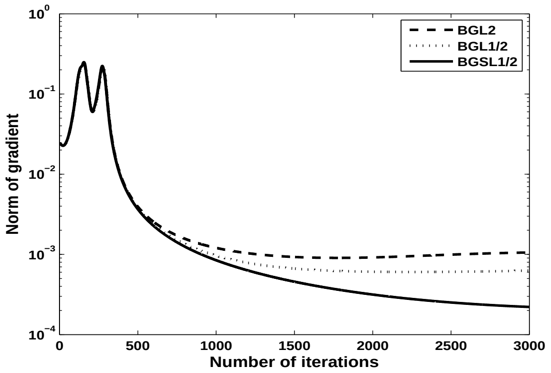

Fig. 5 *Norms of gradient of different algorithms for parity problem.*

Theorem 5, the error of BGSL1/2 decreases monotonically and the gradient of the error function goes to zero in the learning process.

Tabs. I and II show the results of the average errors over the 10 trials and the average norms of gradients for each learning algorithms. The Average Numbers of neurons Eliminated (ANE in brief) by the pruning over the 10 trials are also shown in Tabs. I and II. The comparison convincingly shows that BGSL1/2 is more efficient and has better sparsity-promoting property than BGL1/2 and BGL2.

Algorithm	Average training error	Norm of gradient	ANE
BGSL1/2	1.2336e-004	0.0233	5.4
BGL1/2	1.8418e-004	0.0301	5.1
BGL2	1.7076e-004	0.0387	4.6

Tab. I *Numerical results for solving XOR problem.*

Algorithm	Average training error	Norm of gradients	ANE
BGSL1/2	5.9008e-006	0.0108	7.1
BGL1/2	1.1690e-005	0.0164	6.7
BGL2	2.4495e-005	0.0211	4.9

Tab. II *Numerical results for solving Parity problem.*

4.2 Approximation of Gabor function

Now, we try to approximate the following Gabor function (cf. Fig. 6)

$$F(x,y)=\frac{1}{2\pi(0.5)^2}\exp\left(\frac{x^2+y^2}{2(0.5)^2}\right)\cos(2\pi(x+y)).$$

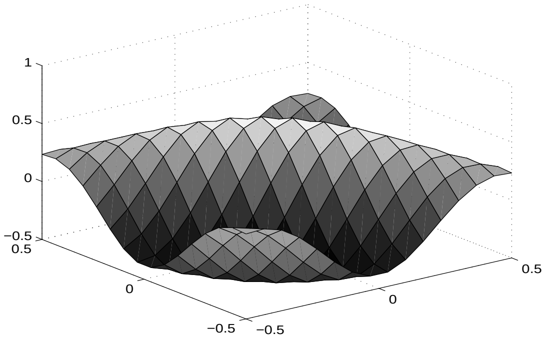

Fig. 6 *Gabor function.*

36 input training samples are selected from an evenly 6×6 grid on $-0.5 \leq x \leq 0.5$ and $-0.5 \leq y \leq 0.5$. Similarly, the input test samples are 256 points selected from the 16×16 grid on $-0.5 \leq x \leq 0.5$ and $-0.5 \leq y \leq 0.5$. The neural network has 3 input nodes, 6 summation nodes, and 1 output node. The initial weights are randomly chosen from $[-0.5, 0.5]$. The learning rate $\eta = 0.6$, and the regularization parameter $\lambda = 0.00001$. The maximum iteration number is 30000.

Typical network approximations by using the three algorithms are plotted in Figs. 7–9. We observe that our method BGSL1/2 gives the best approximation of the Gabor function. Tab. III presents the average training errors, the average test errors, and the ANE, over the 10 trials for the three learning algorithms. Again, we see that our method BGSL1/2 has the best accuracy and the best sparsity-promoting property.

Algorithm	Average training error	Average test error	ANE
BGSL1/2	0.0104	0.1285	2.7
BGL1/2	0.0293	0.1875	2.6
BGL2	0.0269	0.2444	2.8

Tab. III *Numerical results for approximating Gabor function.*

4.3 Approximation of Mayas function

This example is to approximate the following Mayas function

$$F(x, y) = 0.26(x^2 + y^2) - 0.48x \times y.$$

64 input training samples are selected from an evenly 8×8 grid on $-0.5 \leq x \leq 0.5$ and $-0.5 \leq y \leq 0.5$. Similarly, the input test samples are 900 points selected from the 30×30 grid on $-0.5 \leq x \leq 0.5$ and $-0.5 \leq y \leq 0.5$. The neural network has 3 input nodes, 5 summation nodes, and 1 output node. The initial weights are

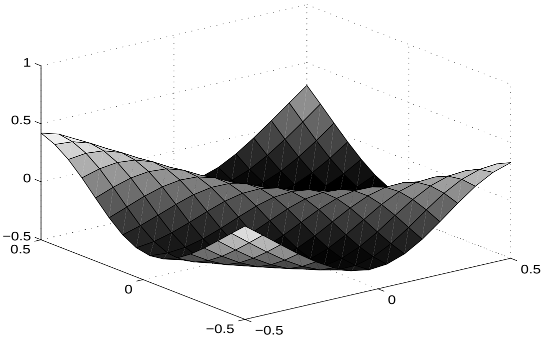

Fig. 7 *Gabor function Approximation performane by BGSL1/2.*

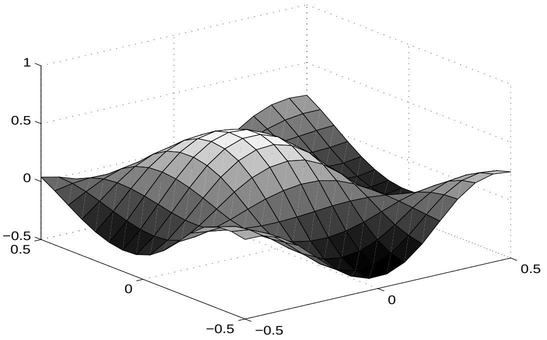

Fig. 8 *Gabor function Approximation performane by BGL1/2.*

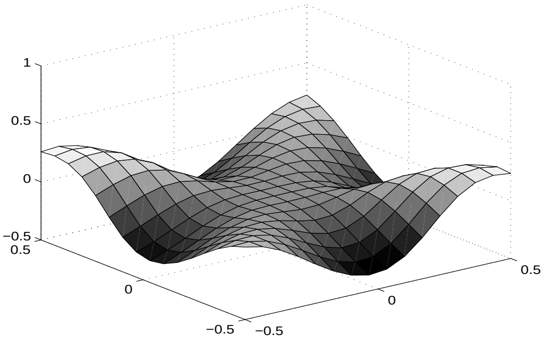

Fig. 9 *Gabor function Approximation performane by BGL2.*

randomly chosen from $[-0.5, 0.5]$. The learning rate $\eta = 0.9$, and the regularization parameter $\lambda = 0.0001$. The maximum iteration number is 10000.

Tab. IV shows that, among the three algorithms, BGSL1/2 exhibits the best approximation accuracy and the best sparsity for the Mayas function.

Algorithms	Average training error	Average test error	ANE
BGSL1/2	0.0013	0.0141	10.0
BGL1/2	0.0035	0.0344	7.3
BGL2	0.0465	0.2043	3.3

Tab. IV *Numerical results for approximating Mayas function.*

5. Proofs

In this section, the convergence theorem of the smoothing $L_{1/2}$ regularization algorithm is proved. First,we give two lemmas.

Lemma 6. *Suppose that the propositions 1 and 2 are satisfied and $\{\omega^m\}$ is generated by Eq. (10), then we have*

$$\left(\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l)-\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)^2 \le C_2\sum_{j=1}^{n}\sum_{i=1}^{p}\|\triangle\omega_{ji}^m\|^2, \tag{12}$$

and

$$\sum_{l=1}^{L}\delta_l\left(\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)\left(\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l)-\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)$$
$$\le(-\frac{1}{\eta}+C_3)\sum_{j=1}^{n}\sum_{i=1}^{p}\|\triangle\omega_{ji}^m\|^2. \tag{13}$$

Proof. Using the Taylor expansion to first and second orders, we have

$$\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l)-\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)=\sum_{j=1}^{n}\left(\prod_{\substack{k=1\\k\neq j}}^{n}t_k\right)(\Delta\omega_j^m\cdot x^l)$$
$$\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l)-\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)$$
$$\tag{14}$$

$$
\begin{aligned}
&= \sum_{j=1}^{n} \prod_{\substack{k=1\\k\neq j}}^{n} (\omega_k^m \cdot x^l)(\triangle\omega_j^m \cdot x^l) + \frac{1}{2}\sum_{\substack{j=1\\s=1\\j\neq s}}^{n} \left(\prod_{\substack{k=1\\k\neq j\\k\neq s}}^{n} t_k^{j,s}\right)(\triangle\omega_j^m \cdot x^l)(\triangle\omega_s^m \cdot x^l)\\
&= \sum_{j=1}^{n} \prod_{\substack{k=1\\k\neq j}}^{n} (\omega_k^m \cdot x^l)(\triangle\omega_j^m \cdot x^l) + \sigma, \qquad (15)
\end{aligned}
$$

where t_k and $t_k^{i,s}$ are in between $\omega_k^{m+1} \cdot x^l$ and $\omega_k^m \cdot x^l$. By proposition 2, we can see that

$$|t_k| \leq 2C_0, |t_k^{i,s}| \leq 2C_0, k = 1, 2, \ldots, n. \qquad (16)$$

Thus, Eq. (12) can be easily obtained from Eq. (14) and Eq. (16). By Eq. (16), we have

$$\sigma \leq \frac{1}{4}(2C_0)^{n-2} \max_{1\leq l\leq L} \|x^l\|^2 \sum_{\substack{j=1\\s=1\\j\neq s}}^{n} (\|\triangle\omega_j^m\|^2 + \|\triangle\omega_s^m\|^2) \leq C_3' \sum_{j=1}^{n}\sum_{i=1}^{p} \|\triangle\omega_{ji}^m\|^2. \qquad (17)$$

A combination of proposition 1, Eq. (10), Eq. (11), Eq. (14) and Eq. (17) leads to

$$
\begin{aligned}
&\sum_{l=1}^{L} \delta_l \left(\prod_{j=1}^{n} (\omega_j^m \cdot x^l)\right)\left(\prod_{j=1}^{n} (\omega_j^{m+1} \cdot x^l) - \prod_{j=1}^{n} (\omega_j^m \cdot x^l)\right)\\
&= \sum_{l=1}^{L} \delta_l \left(\prod_{j=1}^{n} (\omega_j^m \cdot x^l)\right)[\sum_{j=1}^{n} (\triangle\omega_j^m \cdot x^l) \prod_{\substack{k=1\\k\neq j}}^{n} (\omega_k^m \cdot x^l) + \sigma]\\
&\leq \sum_{j=1}^{n}[\sum_{l=1}^{L} \delta_l \left(\prod_{j=1}^{n} (\omega_j^m \cdot x^l)\right) \prod_{\substack{k=1\\k\neq j}}^{n} (\omega_k^m \cdot x^l)x^l] \cdot (\triangle\omega_j^m) + LC_0C_3' \sum_{j=1}^{n}\sum_{i=1}^{p} \|\triangle\omega_{ji}^m\|^2\\
&= \sum_{j=1}^{n}\sum_{i=1}^{p}[\sum_{l=1}^{L} \delta_l \left(\prod_{j=1}^{n} (\omega_j^m \cdot x^l)\right) \prod_{\substack{k=1\\k\neq j}}^{n} (\omega_k^m \cdot x^l)x_i^l] \triangle\omega_{ji}^m + LC_0C_3' \sum_{j=1}^{n}\sum_{i=1}^{p} \|\triangle\omega_{ji}^m\|^2\\
&= -\frac{1}{\eta}\sum_{j=1}^{n}\sum_{i=1}^{p} \|\triangle\omega_{ji}^m\|^2 + C_3 \sum_{j=1}^{n}\sum_{i=1}^{p} \|\triangle\omega_{ji}^m\|^2\\
&= (-\frac{1}{\eta} + C_3) \sum_{j=1}^{n}\sum_{i=1}^{p} \|\triangle\omega_{ji}^m\|^2. \qquad (18)
\end{aligned}
$$

Here $C_3 = LC_0C_3'$. This completes the proof. □

Lemma 7. *Suppose that* $H : \mathbb{R}^u \to \mathbb{R}$ *is continuous and differentiable on a compact set* $\check{D} \subset \mathbb{R}$ *and that* $\Omega = \left\{Z \in \check{D} | \nabla H(Z) = 0\right\}$ *has only finite number of points. If a sequence* $\{Z^m\}_{m=1}^{\infty} \in \check{D}$ *satisfies* $\lim_{m\to\infty} \|Z^{m+1} - Z^m\| = 0$ *and* $\lim_{m\to\infty} \|\nabla H(Z^m)\| = 0$*, then there exists a point* $Z^* \in \Omega$ *such that* $\lim_{m\to\infty} Z^m = Z^*$.

Proof. Lemma 7 is almost the same as Theorem 14.1.5 in [18] and the detail of the proof is omitted □

The Proof of Theorem 5 is alienated into four parts dealing with statements (i), (ii), (iii) and (iv), respectively.

Proof. For convenience, we use the following notations

$$\rho_m = \sum_{j=1}^{n}\sum_{i=1}^{p}(\triangle\omega_{ji}^{m})^2. \tag{19}$$

Then, it follows from the error function $E(\omega)$ in Eq. (8) that

$$E(\omega^{m+1}) = \frac{1}{2}\sum_{l=1}^{L}\left(O^l - g\left(\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l)\right)\right)^2 + \lambda\sum_{j=1}^{n}\sum_{i=1}^{p} f^{1/2}(\omega_{ji}^{m+1}), \tag{20}$$

and

$$E(\omega^{m}) = \frac{1}{2}\sum_{l=1}^{L}\left(O^l - g\left(\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)\right)^2 + \lambda\sum_{j=1}^{n}\sum_{i=1}^{p} f^{1/2}(\omega_{ji}^{m}). \tag{21}$$

Proof to (i) of Theorem 5. By Eq. (12), Eq. (13), Eq. (20), Eq. (21) and the Taylor expansion, we have

$$\begin{aligned}
& E(\omega^{m+1}) - E(\omega^m) \\
&= \frac{1}{2}\sum_{l=1}^{L}\left[\left(O^l - g\left(\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l)\right)\right)^2 - \left(O^l - g\left(\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)\right)^2\right] \\
&\quad + \lambda\sum_{j=1}^{n}\sum_{i=1}^{p}[f^{1/2}(\omega_{ji}^{m+1}) - f^{1/2}(\omega_{ji}^{m})] \\
&= \sum_{l=1}^{L}\delta_l\left(\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)\left(\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l) - \prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right) \\
&\quad + \frac{1}{2}\sum_{l=1}^{L}\delta_l'(t_l')[\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l) - \prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)]^2 + \lambda\sum_{j=1}^{n}\sum_{i=1}^{p}[f^{1/2}(\omega_{ji}^{m+1}) - f^{1/2}(\omega_{ji}^{m})] \\
&= \sum_{j=1}^{n}\sum_{i=1}^{p}\sum_{l=1}^{L}\delta_l\Bigg(\Bigg(\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\left(\prod_{\substack{t=1\\t\neq j}}^{n}(\omega_t^{m}\cdot x^l)x_i^l\triangle\omega_{ji}^{m}\right) + \sum_{l=1}^{L}\delta_l\left(\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)\sigma \\
&\quad + \frac{1}{2}\sum_{l=1}^{L}\delta_l'(t_l')\left(\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l) - \prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)^2 \\
&\quad + \lambda\sum_{j=1}^{n}\sum_{i=1}^{p}[f^{1/2}(\omega_{ji}^{m+1}) - f^{1/2}(\omega_{ji}^{m})]
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{j=1}^{n}\sum_{i=1}^{p}\left[-\frac{1}{\eta}\triangle\omega_{ji}^{m} - \lambda\frac{f'(\omega_{ji}^{m})}{2f^{1/2}(\omega_{ji}^{m})}\right](\triangle\omega_{ji}^{m}) + \sum_{j=1}^{n}\delta_j\left(\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)\sigma \\
&\quad + \frac{1}{2}\sum_{l=1}^{L}\delta_l'(t_l')\left(\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l) - \prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)^2 \\
&\quad + \lambda\sum_{j=1}^{n}\sum_{i=1}^{p}[f^{1/2}(\omega_{ji}^{m+1}) - f^{1/2}(\omega_{ji}^{m})] \\
&= -\frac{1}{\eta}\sum_{j=1}^{n}\sum_{i=1}^{p}(\triangle\omega_{ji}^{m})^2 + \lambda\sum_{j=1}^{n}\sum_{i=1}^{p}\left[f^{1/2}(\omega_{ji}^{m+1}) - f^{1/2}(\omega_{ji}^{m}) - \frac{f'(\omega_{ji}^{m})}{2f(\omega_{ji}^{m})^{1/2}}\right] \\
&\quad + \sum_{l=1}^{L}\delta_l\left(\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)\sigma + \frac{1}{2}\sum_{l=1}^{L}\delta_l'(t_l')\left(\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l) - \prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)^2 \\
&= -\frac{1}{\eta}\sum_{j=1}^{n}\sum_{i=1}^{p}(\triangle\omega_{ji}^{m})^2 + \sum_{l=1}^{L}\delta_l\left(\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)\sigma \\
&\quad + \frac{1}{2}\sum_{l=1}^{L}\delta_l'(t_l')\left(\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l) - \prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)\right)^2 + \frac{\lambda}{2}F''(t_{j,m})\sum_{j=1}^{n}\sum_{i=1}^{p}(\triangle\omega_{ji}^{m})^2 \\
&\leq \left[-\left(\frac{1}{\eta} - \frac{\lambda}{2}F''(t_{j,m})\right) + \left(\frac{C_2}{2} - \frac{1}{\eta} + C_3\right)\right]\sum_{j=1}^{n}\sum_{i=1}^{p}(\triangle\omega_{ji}^{m})^2, \qquad (22)
\end{aligned}
$$

where t_l' is in between $\prod_{j=1}^{n}(\omega_j^{m+1}\cdot x^l)$ and $\prod_{j=1}^{n}(\omega_j^{m}\cdot x^l)$, $t_{j,m}\in\mathbb{R}$ is in between ω_j^{m+1} and ω_j^{m}, $M = \frac{\sqrt{6}}{2\sqrt{a^3}}$ and $F(x)\equiv(f(x))^{1/2}$. Note that

$$
F'(x) = \frac{f'(x)}{2\sqrt{f(x)}}, \tag{23}
$$

and that

$$
F''(x) = \frac{f''(x)}{2\sqrt{f(x)}} \leq \frac{\sqrt{6}}{2\sqrt{a^3}}. \tag{24}
$$

Thus, by using the Lagrangian mean value theorem for $f(x)$ and proposition 3, we have

$$
\begin{aligned}
E(\omega^{m+1}) - E(\omega^{m}) &= [-(\frac{1}{\eta} - \frac{\lambda}{2}F''(t_{j,m})) + (\frac{C_2}{2} - \frac{1}{\eta} + C_3))]\sum_{j=1}^{n}\sum_{i=1}^{p}(\triangle\omega_{ji}^{m})^2 \\
&\leq -(\frac{2}{\eta} - \lambda M - \frac{C_2}{2} - C_3)\sum_{j=1}^{n}\sum_{i=1}^{p}(\triangle\omega_{ji}^{m})^2 \\
&\leq 0, \qquad (25)
\end{aligned}
$$

This completes the proof to statement (i) of Theorem 5.

Proof to (ii) of Theorem 5. From the conclusion (i), we know that the non-negative sequence $\{E(\omega^m)\}$ decreases monotonously. But it is also bounded below. Hence, there must exist $E^* \geq 0$ such that $\lim_{m\to\infty} E(\omega^m) = E^*$. The proof to (ii) is thus completed.

Proof to (iii) of Theorem 5. Write $\beta = \frac{1}{\eta} - \lambda M - \frac{C_2}{2} - C_3$ and $\rho_q = \sum_{j=1}^{n} \sum_{i=1}^{p} (\triangle\omega_{ji}^q)^2$. It follows from proposition 2 that $\beta > 0$. By using Eq. (25) we get

$$E(\omega^{m+1}) \leq E(\omega^m) - \beta\rho_m \leq \cdots \leq E(\omega^0) - \beta \sum_{q=0}^{m} \rho_q.$$

Since $E(\omega^{m+1}) > 0$, we have

$$\beta \sum_{q=0}^{m} \rho_q \leq E(\omega^0) < \infty.$$

Setting $m \to \infty$, we obtain

$$\sum_{q=0}^{\infty} \rho_q \leq \frac{1}{\beta} E(\omega^0) < \infty.$$

Thus,

$$\lim_{m\to\infty} \rho_m = 0.$$

This leads to

$$\lim_{m\to\infty} |\triangle\omega_{ji}^m| = 0, \tag{26}$$

and completes the proof.

Proof to (iv) of Theorem 5. Note that the error function $E(\omega)$ defined in Eq. (8) is continuous and differentiable. According to Eq. (26), proposition 4 and lemma 7, we can easily get the desired result, i.e ., there exists a point $\omega^* \in \Phi_0$ such that

$$\lim_{m\to\infty} \omega^m = \omega^*.$$

This completes the proof to (iv). □

6. Conclusion

Some weak and strong convergence results are established for an offline gradient method with smoothing $L_{1/2}$ regularization for PSNN training. Criterions for choosing the appropriate learning rate and the penalty parameter are given to guarantee the convergence of the learning process. Numerical experiments show that the sparsity and the accuracy of BGSL1/2 are better than those of BGL1/2 and BGL2.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Nos. 61473059 and 61403056), Natural Science Foundation Guidance Project of Liaoning Province (No. 201602050) and Project of Dalian Young Technology Stars.

References

[1] FAN Q., ZURADA J.M., WU W. Convergence of online gradient method for feedforward neural networks with smoothing $L_{1/2}$ regularization penalty. *Neurocomputing*. 2014, 5(131), pp. 208–16, doi.org/10.1016/j.neucom.2013.10.023

[2] FRIEDMAN J.H. An overview of predictive learning and function approximation. In: From Statistics to Neural Networks. Springer, Berlin, Heidelberg. 1994, pp. 1–61, https://doi.org/10.1007/978-3-642-79119-2_1

[3] HINTON G. Connectionist learning procedures. *Artif Intell.* 1989, 40, pp. 185–235, doi.org/10.1016/0004-3702(89)90049-0

[4] HUSSAINA A.J., LIATSISB P. Recurrent pi-sigma networks for DPCM image coding. *Neurocomputing.* 2003, 55(1–2), pp. 363–382, doi.org/10.1016/S0925-2312(02)00629-X

[5] KROGH A., HERTZ J., PALMER R.G. Introduction to the theory of neural computation. Reedwood City CA: *Addison-Wesley.* 1991, doi.org/10.1063/1.2810360

[6] LE CUN Y., DENKER J., Solla S., Howard R.E, JACKEL L.D., Optimal Brain Damage-Advances in Neural Information Processing Systems II. Morgan Kauffman, San Mafeo, CA. 1990.

[7] LIU Y., LI Z.X., YAN D.K., MOHAMED K.H. S.H., WANG J., Wu W. Convergence of batch gradient learning algorithm with smoothing $L_{1/2}$ regularization for Sigma–Pi–Sigma neural networks, *Neurocomputing* 2015, 151, pp. 333–341, doi.org/10.1016/j.neucom.2014.09.031

[8] MACKAY D.J.C., Probable networks and plausible predictions- a review of practical Bayesian methods for supervised neural networks, Network: Computation in Neural Systems, 995, 6(3), pp. 469–505. Available from: https://pdfs.semanticscholar.org/dc5f/59b6c93553d45aa877ddb6db39122690797b.pdf

[9] MC LOONE S., IRWIN G. Improving neural network training solutions using regularisation. *Neurocomputing.* 2001 Apr 30; 37(1), pp. 71–90, doi.org/10.1016/s0925-2312(00)00314-3

[10] PIAO J.L.J.X.F., WEN-PING S.C.D. Application of pi-sigma neural network to real-time classification of seafloor sediments. *Applied Acoustics.* 2005, 6, pp. 004. Available from: http://en.cnki.com.cn/Journal_en/A-A005-YYSN-2005-06.htm

[11] SHIN Y., GHOSH J. The pi-sigma network: an efficient higher-order neural network for pattern classification and function approximation. *International joint conference on neural networks.* 1991, 1, pp. 13–18, doi:10.1109/IJCNN.1991.155142

[12] SHIN Y., GHOSH J., SAMANI D. Computationally efficient invariant pattern recognition with higher order Pi-Sigma Networks. The University of Texas at Austin. 1992.

[13] SRIVASTAVA N., HINTON G., KRIZHEVSKY A., SUTSKEVER I., SALAKHUTDINOW R. Dropout: A simple way to prevent neural networks from overfitting. *J Mach Learn Res.* 2014, 15(1), pp. 1929–1958. Available from: http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

[14] WANG C., VENKATESH S.S., JUDD J.S. Optimal stopping and effective machine complexity in learning. In: *Advances in neural information processing systems* 1994, pp. 303–310. Available from: http://papers.nips.cc/paper/816-optimal-stopping-and-effective-machine-complexity-in-learning.pdf

[15] WU W., FENG G., LI X. Training multilayer perceptrons via minimization of sum of ridge functions. *Advances in Computational Mathematics.* 2002, 17(4), pp. 331–47, doi.org/10.1023/A:1016249727555

[16] WU W., XU Y. Deterministic convergence of an online gradient method for neural networks. *Journal of Computational and Applied Mathematics*. 2002, 144(1), pp. 335–47, doi.org/10.1016/S0377-0427(01)00571-4

[17] XU Z.B., ZHANG H., WANG Y., CHANG X.Y., LING Y. $L_{1/2}$ regularization. *Science China Information Sciences*. 2010, 53(6), pp. 1159–1169, doi.org/10.1007/s11432-010-0090-0

[18] YUAN Y.X., SUN W.Y. Optimization Theory and methods, Science Press, Beijing. 2001.