East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.2023-025.080823

Vol. **xx**, No. x, pp. 1-19
xxx 2024

Boundedness and Convergence Analysis of a Pi-Sigma Neural Network Based on Online Gradient Method and Sparse Optimization

Qinwei Fan[1,2,*], Le Liu[1], Shuai Zhao[1], Zhiwen Zhang[1], Xiaofei Yang[1], Zhiwei Xing[1] and Xingshi He[1]

[1]*School of Science, Xi'an Polytechnic University, Xi'an 710048, China.*
[2]*School of Mathematics and Information Science, Guangzhou University, Guangzhou, 510006, China.*

Received 18 January 2023; Accepted (in revised version) 8 August 2023.

Abstract. High order neural networks have strong nonlinear mapping ability, but the network structure is more complex, which restricts the efficiency of the network, and the relevant theoretical analysis is still not perfect up to now. To solve these problems, an online gradient learning algorithm model of Pi-Sigma neural network with a smooth set lasso regular term is proposed. Since the original lasso regular term contains absolute values and is not differentiable at the origin, it causes experiment oscillations and poses a great challenge to the convergence analysis of the algorithm. We use grinding technology to overcome this deficiency. The main contribution of this paper lies in the adoption of online learning algorithm, which effectively improves the efficiency of the algorithm. At the same time, strict theoretical proofs are presented, including strong convergence and weak convergence. Finally, the effectiveness of the algorithm and the correctness of the theoretical results are verified by numerical experiments.

AMS subject classifications: 65M10, 78A48

Key words: Online gradient method, Pi-Sigma neural network, regularizer, convergence.

1. Introduction

As an important type of high order neural network, Pi-Sigma neural network has high learning efficiency, strong robustness, and powerful nonlinear processing ability. Therefore, such networks have attracted wide attention and are widely used in various fields [2, 7–9, 12, 27, 37, 38].

Backpropagation (BP) algorithm is the most popular method in supervised training of feedforward neural networks. It takes the form of minimizing the mean square error between the expected response and the actual response [11, 25, 29, 31]. In theory, a network

*Corresponding author. *Email addresses:* qinweifan@xpu.edu.cn (Q. Fan), lliu737784717@163.com (L. Liu), z402633008@126.com (S. Zhao), zhiwenzhang0316@126.com (Z. Zhang), yangxiaofei2002@163.com (X. Yang), zwxing@xpu.edu.cn (Z. Xing), xsh1002@126.com (X. He)

with enough neurons can approximate any function with any accuracy, but determining a reasonable structure of the neural network is still a challenging problem.

The gradient method is a commonly used neural network training method to minimize the error function. This can be achieved by a batch gradient method or an online gradient method [19,33,42]. Specifically, batch learning algorithms update weights only once after all samples are presented to the network [6, 20, 32]. Online learning algorithms, on the other hand, update the weights every time a sample is presented to the network [13].

According to the difference of input form of the sample points, it can be divided into online gradient algorithm based on sequential input samples, online gradient algorithm based on specific random input samples and online gradient algorithm based on completely random input samples, including random more strong more conducive to jump out of local minimum, however, due to the introduction of randomness, it becomes challenging to analyze the theoretical performance of the algorithm [16, 34]. In the recent years, there are some theories and applications of gradient-based neural networks have been reported. In [41], a novel finite-time convergent gradient-based neural network model is proposed for solving the dynamic Moore-Penrose inverses problem, and its finite-time convergence is preserved even in the presence of additive bounded dynamic noises. Also, a unified gradient-based neural networks model is proposed for both static matrix inversion and time-varying matrix inversion with finite-time convergence regardless of the existence of bounded additive noises [40].

Generally speaking, the generalization effect of neural networks with smaller weights is better [1]. Two main aspects of neural network learning consist in preventing the over fitting caused by excessive weights and eliminating unnecessary weight connections to achieve a sparse network structure. To solve these problems and optimize the network, one often uses a simplified model, early stop, data enhancement and regularization.

As we all know, adding regularization term to traditional error function can effectively sparsely optimize network structure and obtain better generalization performance [3,4,15]. The error function with the regularization term is as follows:

$$\text{Error} = \frac{1}{2}\sum_i \|\text{Output}^i - \text{Target}^i\| + \lambda \ell(W),$$

where parameter $\lambda > 0$ is the regularization coefficient.

Here we introduce several common regularization terms, L_0 regularization produces the most sparse solution, but it is not easy to calculate [21, 35]. On the other hand, L_1 regularization can produce sparse weight matrices — i.e. sparse models that can be used for feature selection [24]. At the same time, the L_1 regularizer is the optimal convex approximation of L_0 regularizer. Unfortunately, they cannot sparsely select weights at the group level, and both of them are NP-hard problems and are not easy to solve. L_2 regularization can effectively inhibit the excessive growth of weights and prevent the model from overfitting [18, 26, 39, 42]. But L_2 regularization does not any have sparsity.

As a compromise between L_0 and L_1 regular term, a regularization method of $L_{1/2}$ is proposed. However, the regularization term is not differentiable at the origin of coordinates, which makes theoretical analysis difficult. To overcome this defect, a smoothing technique is proposed [5, 14, 19, 23].

Moreover, as far as the regularization techniques are concerned, we note that the least absolute shrinkage and selection operator (lasso) is a compressive estimation technique. As an extension of the lasso method, the group lasso method became a viable approach to variable selection [17, 22]. The regular terms in the group lasso are not only able to remove unessential heavy weights at the group level, but are also characterised by sparsity and good generalisation properties [30]. It also can be seen that this regular term has good performance in optimizing network structure. However, the regular term poses challenges for analysis at its origin, which can in turn cause the error function to oscillate in numerical experiments. In reference [10, 28], the group lasso regularization method is studied and the convergence analysis of the algorithm is given.

Inspired by the above literature, here we study the lasso regular term Pi-Sigma neural network online gradient learning algorithm. The major innovations of this work are described as follows:

(i) Adding a smoothing group lasso regularization term, we prune the redundant weights both within and between groups and achieve a sparse optimization of the network structure. In addition, smoothing technique is used to overcome the non-differentiable defect of traditional group lasso regularization at the origin. This reduces numerical oscillations in the learning process, maintains good sparsity, and at the same time overcomes the obstacles of theoretical analysis.

(ii) Weak and strong convergence theorems of the algorithm are shown under reasonable assumptions. Finally, the superiority of the algorithm and the theoretical results are verified by numerical experiments.

(iii) Compared to batch learning, online learning uses less information storage, so learning efficiency is improved.

The subsequent sections of this article are structured in the following manner. Section 2 describes Pi-Sigma neural network and online gradient algorithms with regularizer. Section 3 outlines the convergence of the submitted algorithm. Besides, the simulation results are presented in Section 4. Finally, a succinct summary is provided in Section 5.

2. Algorithm Description

2.1. The error function with group lasso regularizer

The variables P, d and 1 represent the number of dimensions of the input layer, summation layer, and quadrature layer in Pi-Sigma neural network, respectively. More specifically, P refers to the number of input features, d to the dimensionality of the summation layer, and 1 represents a one-dimensional output from the quadrature layer. Besides, $w_k = (w_{k1}, \ldots, w_{kP}) \epsilon R^P$, $1 \leq k \leq d$ is the weight vector linking the input layer and the k-th element of the summation layer, and $w = (w_1^T, \ldots, w_d^T)^T \epsilon R^{dP}$. It should be noted that the weights between the summing layer and quadrature layer are held constant at a value of 1. We have included a extraordinary input module $x_P = 1$, matching deviations w_{dP}. To

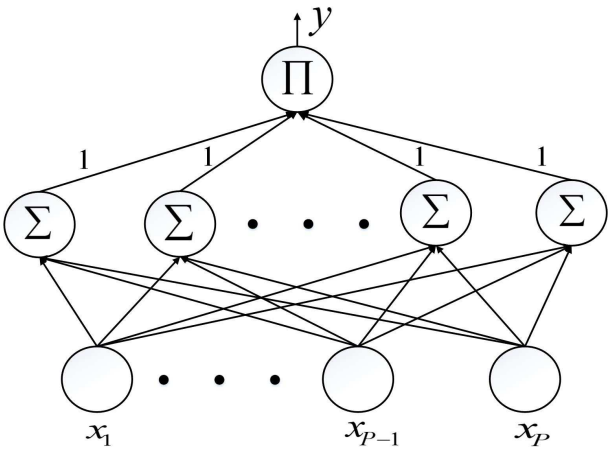

Figure 1: Topology structure of a Pi-Sigma neural network.

define $g : R \to R$ be the activation function. The structure chart of Pi-Sigma neural network algorithm is shown in Fig. 1.

Through the provision of a vector $x \epsilon R^P$ as input, the realized output of the neural network is

$$y = g\left(\prod_{k=1}^{d}(w_k \cdot x)\right),$$

where $w_k \cdot x$ is the in-house product of w_k and x. Suppose that a cluster of training samples $\{x^b, O^b\}_{b=1}^{B} \subset R^P \times R$ is provided to the neural network, where O^b is the intended output corresponding to the input vector x^b. By incorporating the group lasso regularization term into the standard error function, the resulting overall loss function can be expressed as follows:

$$\begin{aligned} E(w) &= \frac{1}{2}\sum_{b=1}^{B}\left(O^b - g\left(\prod_{k=1}^{d}(w_k \cdot x^b)\right)\right)^2 + \lambda\sum_{k=1}^{d}\|w_k\| \\ &= \sum_{b=1}^{B} g_b\left(\prod_{k=1}^{d}(w_k \cdot x^b)\right) + \lambda\sum_{k=1}^{d}\|w_k\|. \end{aligned} \tag{2.1}$$

In the formula above, $\lambda > 0$ is the regularization factor, $g_b(s) = (O^b - g(s))^2/2$. It is easy to calculate the gradient of Eq. (2.1) with respect to w_m, viz.

$$E_{w_m}(w) = \sum_{b=1}^{B} g_b^{'}\left(\prod_{k=1}^{d}(w_k \cdot x^b)\right)\prod_{k=1 k\neq m}^{d}\left(w_k \cdot x^b\right)x^b + \lambda\frac{w_m}{\|w_m\|},$$

where $m = 1, 2, \ldots, d$, $E_w(w) = (E_{w_1}(w), E_{w_2}(w), \cdots, E_{w_d}(w))^T$. We start with an random preliminary value w^0, and the online gradient method adjusts the weight parameters $\{w^l\}$ iteratively by

$$w^{lB+b} = w^{lB+b-1} - \eta_l \Delta_b^l w^{lB+b-1},$$

where $l = 0, 1, 2, \ldots$, $b = 1, 2, \ldots, B$ and

$$\Delta_b^l w_m^{lB+b-1} = g_b^{'}\left(\prod_{k=1}^{d}\left(w_k^{lB+b-1}\cdot x^b\right)\right)\prod_{k=1k\neq m}^{d}\left(w_k^{lB+b-1}\cdot x^b\right)x^b + \frac{\lambda w_m^{lB+b-1}}{B\|w_m^{lB+b-1}\|}.$$

2.2. The error function with smoothing group lasso regularizer

In this section, we introduce a smoothing group lasso (SGL) algorithm, which aims to overcome the problem of non-differentiability of the regularisation term at the origin due to the absolute value function involved in the group lasso regularisation term used in Eq. (2.1), and then uses the smoothing function $h(A,\beta)$ instead of the absolute value function, thus circumventing the oscillations that occur in numerical simulations. For the purposes of this paper, the smoothing function is defined as follows:

$$f(A) = \begin{cases} |A|, & \text{if} \quad |A| > \beta, \\ \dfrac{A^2}{2\beta} + \dfrac{\beta}{2} & \text{if} \quad |A| \leq \beta, \end{cases}$$

where β is a minor normal number, so we can comfortably access

$$f(A) \in [\beta/2, +\infty), \quad f'(A) \in [-1, 1], \quad f''(A) \in [0, 1/\beta].$$

Currently, the common form of the loss function incorporating the smoothed group lasso regularization term is

$$\begin{aligned} E(w) &= \frac{1}{2}\sum_{b=1}^{B}\left(O^b - g\left(\prod_{k=1}^{d}\left(w_k\cdot x^b\right)\right)\right)^2 + \lambda\sum_{k=1}^{d} f(w_k) \\ &= \sum_{b=1}^{B} g_b\left(\prod_{k=1}^{d}\left(w_k\cdot x^b\right)\right) + \lambda\sum_{k=1}^{d} f(w_k). \end{aligned} \tag{2.2}$$

It is easily shown that the partial derivative of Eq. (2.2) has the form w_m is

$$E_{w_m}(w) = \sum_{b=1}^{B} g_b^{'}\left(\prod_{k=1}^{d}\left(w_k\cdot x^b\right)\right)\prod_{k=1k\neq m}^{d}\left(w_k\cdot x^b\right)x^b + \lambda f'(w_m),$$

where $m = 1, 2, \ldots, d$.

Starting with an arbitrary initial value w^0, the online gradient method updates the weight $\{w^l\}$ iteratively by

$$w^{lB+b} = w^{lB+b-1} - \eta_l \Delta_b^l w^{lB+b-1}, \tag{2.3}$$

where $l = 0, 1, 2, \ldots$, $b = 1, 2, \ldots, B$ and

$$\Delta_b^l w_m^{lB+b-1} = g_b^{'}\left(\prod_{k=1}^{d}\left(w_k^{lB+b-1}\cdot x^b\right)\right)\prod_{k=1k\neq m}^{d}\left(w_k^{lB+b-1}\cdot x^b\right)x^b + \frac{\lambda}{B}f'\left(w_m^{lB+b-1}\right). \tag{2.4}$$

We again note that the initial weights w^0 and w^l should be randomly selected.

3. Main Results

The following criteria will be used to certify the convergence theorem.

Assumption 3.1. The functions $|g(s)|, |g^{'}(s)|$ are Lipschitz continuous for $s \in R$.

Assumption 3.2. The variable learning rate η_l is subject to the following conditions:

$$0 < \eta_l < 1, \quad \sum_{l=0}^{\infty} \eta_l < \infty.$$

For simplicity, we denote

$$\begin{aligned} C_2 &= \max\Big\{ \sup_{s\in\mathbb{R}} |g(s)|, \sup_{s\in\mathbb{R}} |g'(s)|, \sup_{s\in\mathbb{R}} |g''(s)|, \sup_{s\in\mathbb{R}, 1\le b\le B} |g_b'(s)|, \sup_{s\in\mathbb{R}, 1\le b\le B} |g_b''(s)| \Big\}, \\ C_3 &= \max_{1\le b\le B} \left\{ \|x^b\|, |w_m^l x^b| \right\}. \end{aligned} \tag{3.1}$$

Theorem 3.1. *Let $E(w)$ be the error function defined in Eq.* (2.2)*, w^0 be an arbitrary initial value, and the weight sequence $\{w^l\}$ be generated by the iteration algorithm Eq.* (2.3)*. Then under Assumptions* 3.1-3.2*, there exists a unique $w^* \in \Omega_0$ such that*

$$\begin{aligned} &\lim_{l\to\infty} w^l = w^*, \\ &\lim_{l\to\infty} \|E_w(w^l)\| = \|E_w(w^*)\| = 0. \end{aligned}$$

Before proving this theorem, let us recall two auxiliary results which will be needed in what follows.

Lemma 3.1 (cf. Wu *et al.* [33, Lemma 4.2])**.** *Suppose that the learning rate η_l satisfies Assumption* 3.2 *and that the sequence $\{a_l\}$, $l \in \mathbb{N}$ satisfies $a_l \ge 0$, $\sum_{l=0}^{\infty} \eta_l a_l^{\beta} < \infty$ and $|a_{l+1} - a_l| \le \mu\eta_l$ for positive constants β and μ. Then*

$$\lim_{l\to\infty} a_l = 0.$$

Lemma 3.2 (cf. Xu *et al.* [36])**.** *Let Y_s, W_s and Z_s be three sequences such that W_s is nonnegative and Y_s is bounded for all s. If*

$$Y_{s+1} \le Y_s - W_s + Z_s, \quad s = 0, 1, \ldots$$

and the series $\sum_{s=0}^{\infty} Z_s$ converges, then Y_s also converges and $\sum_{s=0}^{\infty} W_s < \infty$.

Proof of Theorem 3.1. According to Assumption 3.2, we have $\sum_{l=0}^{\infty} \eta_l < \infty$. Therefore, it can be readily deduced that the sequence $S_l = \eta_0 + \eta_1 + \cdots + \eta_{l-1}$ converges. Applying the Cauchy convergence test, we can conclude that for any given positive value of for ε, there exists a positive integer $N_1 \in \mathbb{N}$ such that for all $l > N_1$ and all $P \in \mathbb{N}$ we have

$$|S_{l+P} - S_l| = \eta_l + \eta_{l+1} + \cdots + \eta_{l+P-1} < \varepsilon,$$

$$|S_{l+P+1}-S_{l+1}|=\eta_{l+1}+\eta_{l+2}+\cdots+\eta_{l+P}<\varepsilon.$$

Updating Eqs. (2.3), (2.4), (3.1) and $f'(A)\in[-1,1]$ yields

$$\begin{aligned}\left|w_k^{lB+b}-w_k^{lB+b-1}\right|&=\eta_l\left|\Delta_b^l w_k^{lB+b-1}\right|\\&\le\eta_l\left(\left|g_b'\left(\prod_{k=1}^{d}\left(w_k^{lB+b-1}\cdot x^b\right)\right)\prod_{k=1,k\neq m}^{d}\left(w_k^{lB+b-1}\cdot x^b\right)\cdot x^b\right|\right.\\&\qquad\left.+\left|\frac{\lambda f'(w_m^{lB+b-1})}{B}\right|\right)\\&\le\eta_l\left(C_2C_3^d+\frac{\lambda}{B}\right)\le\eta_l A_1,\end{aligned}$$

where $A_1=C_2C_3^d+\lambda/B$. Since

$$\begin{aligned}\left|w_k^{(l+P)B+b}-w_k^{lB+b}\right|&\le\left|w_k^{(l+P)B+b}-w_k^{(l+P-1)B+b}\right|\\&\quad+\left|w_k^{(l+P-1)B+b}-w_k^{(l+P-2)B+b}\right|\\&\quad+\cdots+\left|w_k^{(l+1)B+b}-w_k^{lB+b}\right|\\&\le A_1b(\eta_{l+P}+\eta_{l+P-1}+\cdots+\eta_{l+1})\\&\quad+A_1(B-b)(\eta_{l+P-1}+\eta_{l+P-2}+\cdots+\eta_l)\\&\le BA_1\varepsilon,\end{aligned}$$

it follows that

$$\begin{aligned}\left|w_k^{(l+1)B+b}-w_k^{lB+b}\right|&\le\left|w_k^{(l+1)B+b}-w_k^{(l+1)B+b-1}\right|\\&\quad+\left|w_k^{(l+1)B+b-1}-w_k^{(l+1)B+b-2}\right|\\&\quad+\cdots+\left|w_k^{(l+1)B+1}-w_k^{(l+1)B}\right|\\&\quad+\left|w_k^{lB+B}-w_k^{lB+B-1}\right|\\&\quad+\left|w_k^{lB+B-1}-w_k^{lB+B-2}\right|\\&\quad+\cdots+\left|w_k^{lB+b+1}-w_k^{lB+b}\right|\\&\le\left(b\eta_{l+1}+(B-b)\eta_l\right)A_1.\end{aligned}$$

Therefore, the weight sequence $\{w_k^{lB+b}\}$ converges. It follows that there exists a constant $\bar{M}\ge0$ such that the following inequality holds:

$$\left\|\Delta_b^l w_k^{lB+b-1}\right\|\le\bar{M}\tag{3.2}$$

for $b=1,2,\ldots,B$.

For the sake of convenience and simplicity, we introduce the following notations:

$$r_k^{l,b} = \Delta_b^l w_k^{lB+b-1} - \Delta_b^l w_k^{lB}, \qquad v_k^{l,b} = w_k^{lB+b} - w_k^{lB}, \tag{3.3}$$

$$\tau_b^{lB+b} = \Pi_{k=1}^d \left(w_k^{lB+b} \cdot x^b\right), \qquad \psi_b^{l,b} = \tau_b^{lB+b} - \tau_b^{lB}, \tag{3.4}$$

$$\bar{\tau}_{b,m}^{lB+b} = \Pi_{k=1,k\neq m}^d \left(w_k^{lB+b} \cdot x^b\right), \quad \bar{\psi}_{b,m}^{l,b} = \bar{\tau}_{b,m}^{lB+b} - \bar{\tau}_{b,m}^{lB}, \tag{3.5}$$

$$\varphi_{m,b}^{l,b} = v_m^{l,b} \cdot x^b. \tag{3.6}$$

Note that for all $k = 0, 1, 2, \ldots, d$ and $l = 0, 1, 2, \ldots$, we have

$$r_k^{l,1} = 0, \quad v_k^{l,b} = -\eta_l \sum_{m=1}^{b} \left(\Delta_m^l w_k^{lB} + r_k^{l,m}\right), \quad b = 1, 2, \ldots, B, \tag{3.7}$$

$$\left\|v_k^{l,b}\right\| = \eta_l b \bar{M}. \tag{3.8}$$

Indeed

$$\begin{aligned} v_k^{l,b} &= w_k^{lB+b} - w_k^{lB} \\ &= -\eta_l \Delta_b^l w_k^{lB+b-1} - \eta_l \Delta_{b-1}^l w_k^{lB+b-2} \\ &\quad - \eta_l \Delta_{b-2}^l w_k^{lB+b-3} - \cdots - \eta_l \Delta_2^l w_k^{lB+1} - \eta_l \Delta_1^l w_k^{lB} \\ &= -\eta_l \sum_{m=1}^{b} \Delta_m^l w_k^{lB+m-1} \\ &= -\eta_l \sum_{m=1}^{b} \left(\Delta_m^l w_k^{lB} + r_k^{l,m}\right). \end{aligned} \tag{3.9}$$

Subsequently, utilizing the error function Eq. (2.2), we can derive the following expression:

$$\begin{aligned} E\left(w^{(l+1)B}\right) &= \sum_{b=1}^{B} g_b \left(\prod_{k=1}^{d} \left(w_k^{(l+1)B} \cdot x^b\right)\right) + \lambda \sum_{k=1}^{d} f\left(w_k^{(l+1)B}\right) \\ &= \sum_{b=1}^{B} g_b \left(\tau_b^{(l+1)B}\right) + \lambda \sum_{k=1}^{d} f\left(w_k^{(l+1)B}\right), \\ E(w^{lB}) &= \sum_{b=1}^{B} g_b \left(\tau_b^{lB}\right) + \lambda \sum_{k=1}^{d} f\left(w_k^{lB}\right). \end{aligned}$$

Applying the Taylor theorem with the Lagrange remainder, and Eqs. (2.4), (3.6), (3.9), we conclude that

$$\begin{aligned} &E\left(w^{(l+1)B}\right) - E\left(w^{lB}\right) \\ &= \sum_{b=1}^{B} \left(g_b\left(\tau_b^{(l+1)B}\right) - g_b\left(\tau_b^{lB}\right)\right) + \lambda \sum_{k=1}^{d} \left(f\left(w_k^{(l+1)B}\right) - f\left(w_k^{lB}\right)\right) \end{aligned}$$

$$
\begin{aligned}
&= \sum_{b=1}^{B}\Bigg[g_b'\left(\tau_b^{lB}\right)\left(\sum_{m=1}^{d}\left(\bar{\tau}_{b,m}^{lB}\varphi_{m,b}^{l,B}\right)+\frac{1}{2}\sum_{m_1,m_2=1,m_1\neq m_2}^{d}\left(\left(\prod_{k=1,k\neq m_1,m_2}^{d} s_{k,l,b}\right)\varphi_{m_1,b}^{l,B}\varphi_{m_2,b}^{l,B}\right)\right)\\
&\qquad\qquad + \frac{1}{2}g_b''(s_{b,l})\left(\psi_b^{l,B}\right)^2\Bigg] + \lambda\sum_{k=1}^{d}\left(f\left(w_k^{(l+1)B}\right)-f\left(w_k^{lB}\right)\right)\\
&= \sum_{b=1}^{B}\left(g_b'\left(\tau_b^{lB}\right)\left(\sum_{m=1}^{d}\left(\bar{\tau}_{b,m}^{lB}v_m^{l,B}\cdot x^b\right)\right)\right)+\lambda\sum_{k=1}^{d}\left(f\left(w_k^{(l+1)B}\right)-f\left(w_k^{lB}\right)\right)+\delta_1+\delta_2\\
&= \sum_{b=1}^{B}\sum_{k=1}^{d}\left(\left(\Delta_b^l w_k^{lB}-\frac{\lambda}{B}f'\left(w_k^{lB}\right)\right)v_k^{l,B}\right)+\lambda\sum_{k=1}^{d}\left(f\left(w_k^{(l+1)B}\right)-f\left(w_k^{lB}\right)\right)+\delta_1+\delta_2\\
&= \sum_{b=1}^{B}\sum_{k=1}^{d}\left(\Delta_b^l w_k^{lB}\cdot v_k^{l,B}\right)-\frac{\lambda}{B}\sum_{b=1}^{B}\sum_{k=1}^{d}f'\left(w_k^{lB}\right)v_k^{l,B}+\lambda\sum_{k=1}^{d}\left(f\left(w_k^{(l+1)B}\right)-f\left(w_k^{lB}\right)\right)+\delta_1+\delta_2\\
&= \sum_{b=1}^{B}\sum_{k=1}^{d}\left(\Delta_b^l w_k^{lB}\left(-\eta_l\sum_{b=1}^{B}\left(\Delta_b^l w_k^{lB}+r_k^{l,b}\right)\right)\right)-\frac{\lambda}{B}\sum_{b=1}^{B}\sum_{k=1}^{d}f'\left(w_k^{lB}\right)v_k^{l,B}\\
&\quad+\lambda\sum_{k=1}^{d}\left(f\left(w_k^{(l+1)B}\right)-f\left(w_k^{lB}\right)\right)+\delta_1+\delta_2\\
&= -\eta_l\sum_{k=1}^{d}\left(\sum_{b=1}^{B}\Delta_b^l w_k^{lB}\right)^2-\eta_l\sum_{k=1}^{d}\left(\sum_{b=1}^{B}\Delta_b^l w_k^{lB}\cdot\sum_{b=1}^{B}r_k^{l,b}\right)-\frac{\lambda}{B}\sum_{b=1}^{B}\sum_{k=1}^{d}f'\left(w_k^{lB}\right)v_k^{l,B}\\
&\quad+\lambda\sum_{k=1}^{d}\left(f\left(w_k^{(l+1)B}\right)-f\left(w_k^{lB}\right)\right)+\delta_1+\delta_2\\
&= -\eta_l\left\|E_w(w^{lB})\right\|^2-\eta_l\sum_{k=1}^{d}\left(\sum_{b=1}^{B}\Delta_b^l w_k^{lB}\cdot\sum_{b=1}^{B}r_k^{l,b}\right)+M\lambda\sum_{k=1}^{d}\left(v_k^{l,B}\right)^2+\delta_1+\delta_2,
\end{aligned}
$$

where

$$
\begin{aligned}
\delta_1 &= \frac{1}{2}\sum_{b=1}^{B}\left(g_b'\left(\tau_b^{lB}\right)\sum_{m_1,m_2=1,m_1\neq m_2}^{d}\left(\left(\prod_{k=1,k\neq m_1,m_2}^{d}s_{k,l,b}\right)\varphi_{m_1,b}^{l,B}\varphi_{m_2,b}^{l,B}\right)\right),\\
\delta_2 &= \frac{1}{2}\sum_{b=1}^{B}g_b''(s_{b,l})\left(\psi_b^{l,B}\right)^2,
\end{aligned}
$$

$s_{k,l,b}\in\mathbb{R}$ is a constant between $w_k^{(l+1)B}\cdot x^b$ and $w_k^{lB}\cdot x^b$, $s_{b,l}\in\mathbb{R}$ is a constant between $\tau_b^{(l+1)B}$ and τ_b^{lB}, and $M=1/(2\beta)$. It follows from (3.2), (3.9) that

$$
\sum_{k=1}^{d}\left(v_k^{l,B}\right)^2=\sum_{k=1}^{d}\left(-\eta_l\sum_{m=1}^{B}\Delta_m^l w_k^{lB+m-1}\right)^2\leq dB^2\bar{M}^2\eta_l^2=C_1\eta_l^2
$$

with $C_1=dB^2\bar{M}^2$.The relations (2.4), (3.3), (3.4) and the mean value theorem gives

$$
\begin{aligned}
\left|r_k^{l,b}\right| &= \left|\Delta_b^l w_k^{lB+b-1} - \Delta_b^l w_k^{lB}\right| \\
&= \left| g_b'\left(\tau_b^{lB+b-1}\right)\cdot \bar{\tau}_{b,m}^{lB+b-1}\cdot x^b - g_b'\left(\tau_b^{lB}\right)\cdot \bar{\tau}_{b,m}^{lB}\cdot x^b + \frac{\lambda}{B} f'\left(w_k^{lB+b-1}\right) - \frac{\lambda}{B} f'\left(w_k^{lB}\right)\right| \\
&= \left| g_b'\left(\tau_b^{lB+b-1}\right)\cdot \bar{\tau}_{b,m}^{lB+b-1}\cdot x^b + g_b'\left(\tau_b^{lB}\right)\cdot \bar{\tau}_{b,m}^{lB+b-1}\cdot x^b - g_b'\left(\tau_b^{lB}\right)\cdot \bar{\tau}_{b,m}^{lB+b-1}\cdot x^b \right. \\
&\qquad \left. - g_b'\left(\tau_b^{lB}\right)\cdot \bar{\tau}_{b,m}^{lB}\cdot x^b + \frac{\lambda}{B} f'\left(w_k^{lB+b-1}\right) - \frac{\lambda}{B} f'\left(w_k^{lB}\right)\right| \\
&\le \left| g_b''(t_{b,l})\psi_b^{l,b-1}\bar{\tau}_{b,m}^{lB+b-1}\cdot x^b\right| + \left| g_b'(\tau_b^{lB})\bar{\psi}_{b,m}^{l,b-1}\cdot x^b\right| + \frac{\lambda}{B} f''(s_{k,B})\left|v_k^{l,b-1}\right|,
\end{aligned}
$$

where $t_{b,l} \in \mathbb{R}$ is a constant between τ_b^{lB+b-1} and τ_b^{lB}, and $s_{k,B} \in \mathbb{R}$ is a constant between w_k^{lB+b-1} and w_k^{lB}. Using Assumption 3.1 and Eqs. (3.4)-(3.8) yield

$$
\begin{aligned}
\left|\psi_b^{l,b}\right| &= \left|\tau_b^{lB+b} - \tau_b^{lB}\right| \\
&= \left| \prod_{k=1}^{d}\left(w_k^{lB+b}\cdot x^b\right) - \prod_{k=1}^{d}\left(w_k^{lB}\cdot x^b\right)\right| \\
&= \left| \prod_{k=1}^{d}\left(w_k^{lB+b}\cdot x^b\right) - \prod_{k=1}^{d-1}\left(w_k^{lB+b}\cdot x^b\right) w_d^{lB}\cdot x^b \right. \\
&\qquad + \prod_{k=1}^{d-1}\left(w_k^{lB+b}\cdot x^b\right) w_d^{lB}\cdot x^b - \prod_{k=1}^{d-2}\left(w_k^{lB+b}\cdot x^b\right) w_d^{lB}\cdot x^b\cdot w_{d-1}^{lB}\cdot x^b \\
&\qquad \left. + \cdots + \left(w_1^{lB+b}\cdot x^b\right)\prod_{k=2}^{d}\left(w_k^{lB}\cdot x^b\right) - \prod_{k=1}^{d}\left(w_k^{lB}\cdot x^b\right)\right| \\
&\le \left| \prod_{k=1}^{d-1}\left(w_k^{lB+b}\cdot x^b\right)\left(w_d^{lB+b} - w_d^{lB}\right)x^b\right| \\
&\quad + \left| \prod_{k=1}^{d-2}\left(w_k^{lB+b}\cdot x^b\right)\cdot w_d^{lB}\cdot x^b\cdot\left(w_{d-1}^{lB+b} - w_{d-1}^{lB}\right)x^b\right| \\
&\quad + \cdots + \left| \prod_{k=2}^{d}\left(w_k^{lB}\cdot x^b\right)\left(w_1^{lB+b} - w_1^{lB}\right)x^b\right| \\
&\le \left| \prod_{k=1}^{d-1}\left(w_k^{lB+b}\cdot x^b\right)\right|\left|\varphi_{d,b}^{l,b}\right| + \left| \prod_{k=1}^{d-2}\left(w_k^{lB+b}\cdot x^b\right)\right|\left|w_d^{lB}\cdot x^b\right|\left|\varphi_{d-1,b}^{l,b}\right| \\
&\quad + \cdots + \left| \prod_{k=2}^{d}\left(w_k^{lB}\cdot x^b\right)\right|\left|\varphi_{1,b}^{l,b}\right| \\
&\le C_3^d\left(\left|v_d^{l,b}\right| + \left|v_{d-1}^{l,b}\right| + \cdots + \left|v_1^{l,b}\right|\right) \\
&\le C_3^d\, d\, b\overline{M}\eta_l = A_1\eta_l,
\end{aligned}
$$

where $A_1 = C_3^d d\, b\overline{M}$. Similarly,

$$\left|\overline{\psi}_{b,m}^{l,b}\right| \le A_2\eta_l$$

with $A_2 = C_3^{d-1} d\, b\overline{M}$. Thus

$$\begin{aligned}
&\left|r_k^{l,b}\right| \le C_2A_1\eta_l C_3^{d-1}C_3 + C_2A_2\eta_l C_3 + \frac{\lambda}{B}\frac{1}{\beta}\eta_l(b-1)\overline{M}\\
&\quad = \left(C_2A_1C_3^d + C_2A_2C_3 + \frac{\lambda}{B}\frac{1}{\beta}(b-1)\overline{M}\right)\eta_l = C_4\eta_l,\\
&\left|-\eta_l\sum_{k=1}^{d}\left(\sum_{b=1}^{B}\Delta_b^l w_k^{lB}\cdot\sum_{b=1}^{B}r_k^{l,b}\right)\right|\\
&\le \sum_{k=1}^{d}\left(B^2\bar{M}C_4\right)\eta_l^2 = dB^2\bar{M}C_4\eta_l^2 = C_5\eta_l^2,\\
&|\delta_1| = \left|\frac{1}{2}\sum_{b=1}^{B}\left(g_b'\left(\tau_b^{lB}\right)\sum_{m_1,m_2=1,m_1\neq m_2}^{d}\left(\left(\prod_{k=1,k\neq m_1,m_2}^{d}s_{k,l,b}\right)\varphi_{m_1,b}^{l,B}\varphi_{m_2,b}^{l,B}\right)\right)\right|\\
&\quad \le \frac{1}{2}\left|\sum_{b=1}^{B}C_2\sum_{m_1,m_2=1,m_1\neq m_2}^{d}C_3^{d-2}\cdot v_{m_1}^{l,B}x^b\cdot v_{m_2}^{l,B}x^b\right|\\
&\quad \le \frac{1}{2}C_2C_3^dB\sum_{m_1,m_2=1,m_1\neq m_2}^{d}\left(\left\|v_{m_1}^{l,B}\right\|\cdot\left\|v_{m_2}^{l,B}\right\|\right)\\
&\quad \le \left(\frac{1}{2}C_2C_3^dB(d-1)dB^2\overline{M}^2\right)\eta_l^2 \le C_6\eta_l^2,\\
&|\delta_2| = \left|\frac{1}{2}\sum_{b=1}^{B}g_b''(s_{b,l})\left(\psi_b^{l,B}\right)^2\right| \le \frac{1}{2}C_2\left|\sum_{b=1}^{B}\left(C_3^d dB\overline{M}\right)^2\eta_l^2\right|\\
&\quad \le \frac{C_2}{2}B\left(C_3^d dB\overline{M}\right)^2\eta_l^2 = C_7\eta_l^2,\\
&E\left(w^{(l+1)B}\right) - E\left(w^{lB}\right)\\
&\le -\eta_l\left\|E_w(w^{lB})\right\|^2 + C_5\eta_l^2 + M\lambda C_1\eta_l^2 + C_6\eta_l^2 + C_7\eta_l^2\\
&= -\eta_l\left\|E_w(w^{lB})\right\|^2 + M_0\eta_l^2,
\end{aligned}$$

where

$$\begin{aligned}
C_4 &= C_2A_1C_3^d + C_2A_2C_3 + \frac{\lambda}{B}\frac{1}{\beta}(b-1)\overline{M},\\
C_5 &= dB^2\bar{M}C_4,\\
C_6 &= \frac{1}{2}C_2C_3^dB(d-1)dB^2\overline{M}^2,\\
C_7 &= \frac{C_2}{2}B\left(C_3^d dB\overline{M}\right)^2,
\end{aligned}$$

$$M_0 = C_5 + M\lambda C_1 + C_6 + C_7.$$

Assumption 3.2, Lemma 3.2 and the above equation show that

$$\sum_{l=0}^{\infty} \eta_l \left\| E_w(w^{lB}) \right\|^2 < \infty$$

and

$$\begin{aligned} &\left| E_{w_k}(w^{(l+1)B}) - E_{w_k}(w^{lB}) \right| \\ &= \sum_{b=1}^{B} \left| r_k^{l,B+1} \right| \le B \left(C_2 A_1 C_3^d + C_2 A_2 C_3 + \frac{\lambda}{B} \frac{1}{\beta} (B-1) \overline{M} \right) \eta_l = C_8 \eta_l. \end{aligned} \tag{3.10}$$

Combining the inequalities above and using Lemma 3.1 gives $\lim_{l\to\infty} E_w(w^{lB}) = 0$, and it follows from Assumption 3.2 that $\lim_{l\to\infty} \eta_l = 0$. Using Eq. (3.10), we obtain

$$\begin{aligned} \left| E_w\left(w^{(l+1)B}\right) \right| &\le \sum_{b=1}^{B} \left| r_k^{l,B+1} \right| + \left| E_w\left(w^{lB}\right) \right| \\ &\le C_8 \eta_l + \left| E_w\left(w^{lB}\right) \right|, \end{aligned}$$

so that

$$\lim_{l\to\infty} \left\| E_w\left(w^{(l+1)B}\right) \right\| = 0.$$

Assumption 3.1 ensures the continuity of $E_w(w)$. Based on the aforementioned proof, one can conclude that the sequence $\{w^{lB+b}\}$ converges. Let $\lim_{l\to\infty} w^{lB+b} = w^*$, then $\lim_{l\to\infty} E_w(w^{lB+b}) = E_w(w^*) = 0$. Hence,

$$\lim_{l\to\infty} w^{lB+b} = w^*, w^* \in \Omega_0.$$

The proof is complete. □

4. Numerical Experiments

To verify the validity of SGL (Pi-Sigma neural network with smoothing group lasso regularizer), we compare it with L2 (Pi-Sigma neural network with L2 regularizer) and OGL (Pi-Sigma neural network with original group lasso regularizer) through parity problem and function regression problem and benchmarking with classification problem.

Example 4.1 (Parity Problem)**.** The parity problem is a classical binary classification problem. In this section, the input set consists of 2^n patterns in n-dimensional space and each pattern is an n-bit binary vector. The target output O^j is equal to 1 if the number of 1 in the pattern is odd, otherwise it is equal to zero. Without loss of generality, this section solves the three-dimensional parity problem ($n = 3$), for which the inputs and the desired outputs are given in Table 1. The input sample dimension is 3, there are 5 hidden nodes

Table 1: Inputs and ideal outputs for 3-dimensional parity problems.

Input	Output	Input	Output
1 1 1	1	1 1 -1	0
1 -1 1	0	-1 -1 -1	0
1 -1 -1	1	-1 1 1	0
-1 -1 1	1	-1 1 -1	1

Table 2: Performance comparison of parity problem.

Algorithm	RWT	RNT	Train time(s)
L2	11.8	3.4	1.6995
OGL	10.8	3.0	1.4289
SGL	9.4	2.0	1.3884

and 1 output, where the penalty term $\lambda = 0.00001$ and the learning rate $\eta = 0.04$. The training process stops when 3000 iterations or the error is less than $1e-6$. Each algorithm was performed 10 times. The comparison diagrams of error function, norm of gradient, norm of weight and absolute error (ideal output vs. actual output) are given in the parity problem. Table 2 shows their performance comparison. In the function regression problem experiment, we give the comparison graph of function approximation effect and the absolute error graph. At its conclusion, the average training time, remaining weight (RWT) and number of neurons (RNT) after pruning of the hidden layers were compared. Figs. 2 and 3 show that the error function and the gradient norm decrease monotonically and tend to 0, so that the SGL overcomes the oscillation phenomenon. It can be seen from Fig. 4 that SGL converges faster and more effectively. Fig. 5 depicts the absolute error, or discrepancy, between the desired output and the actual output. This clearly indicates that the error produced by SGL algorithm is relatively smaller.

Example 4.2 (Function Regression Problem)**.** Let $f(x) = \cos(x/2\pi)$. We employ three algorithms to approximate the function $f(x) = \cos(x/2\pi)$, $x \in [-1, 1]$, the input consisted of 101 samples, with 4 hidden nodes and a single output, the penalty term is 0.001, and the learning rate is 0.00024. The training process stops when 20000 iterations or the error is less than $1e-6$.

Fig. 6 shows that L2, OGL and SGL approximate $f(x)$, with SGL providing the best approximation. It is apparent from Fig. 7 that the absolute error of SGL is minimal.

Let $h(x) = \sin(\pi x)$. We again use three algorithms for approximation of this function in the interval $[-1.5, 0.5]$. The input consists of 101 samples, with 4 hidden nodes and a single output. The training process stops when 40000 iterations or the error is less than $1e-6$. The penalty term of is 0.001, and the learning rate of is $1e-5$. Fig. 8 demonstrates that L2, OGL and SGL approximate $f(x)$, and SGL delivers the best approximation. It is apparent from Fig. 9 that the absolute error of SGL is minimal.

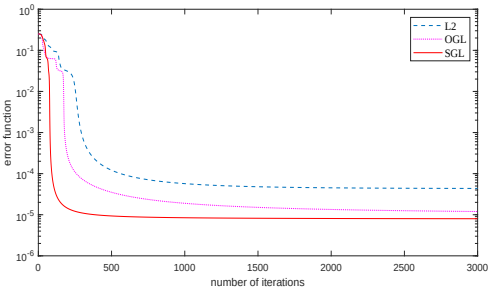

Figure 2: Error function for parity problem.

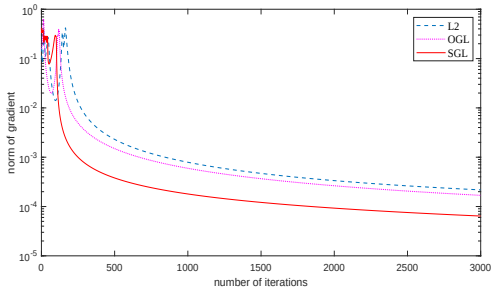

Figure 3: Norm of gradient for parity problem.

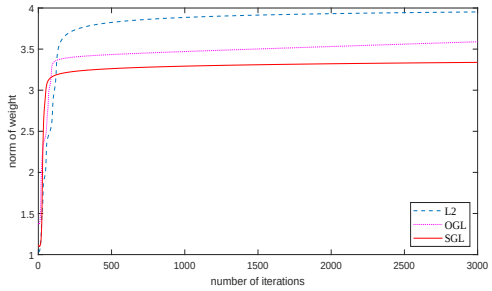

Figure 4: Norm of weight for parity problem.

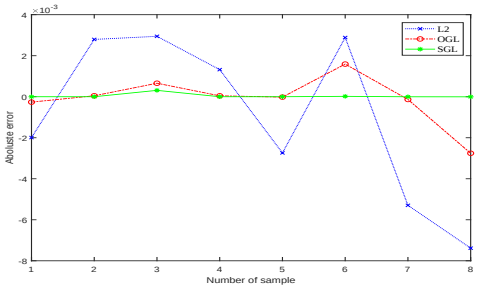

Figure 5: Error comparison of a parity problem.

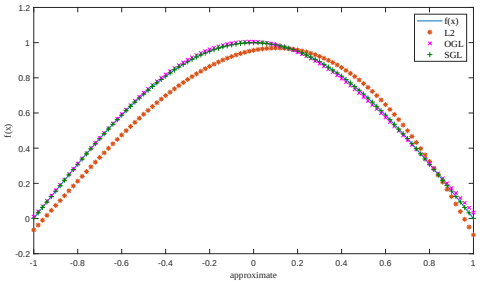

Figure 6: Function approximation graph.

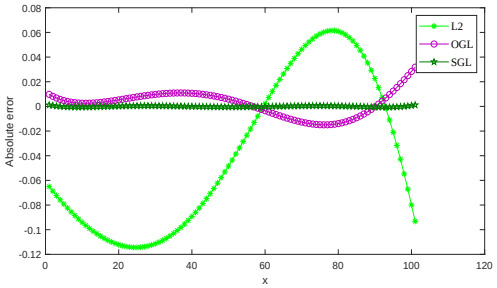

Figure 7: Approximation error diagram.

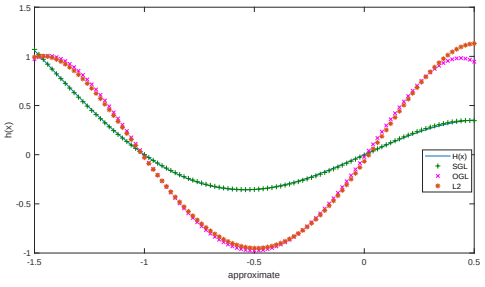

Figure 8: Function approximation graph.

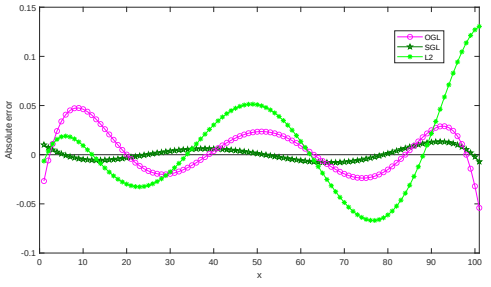

Figure 9: Approximation error diagram.

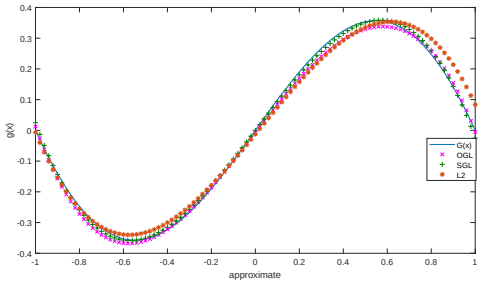

Figure 10: Function approximation graph.

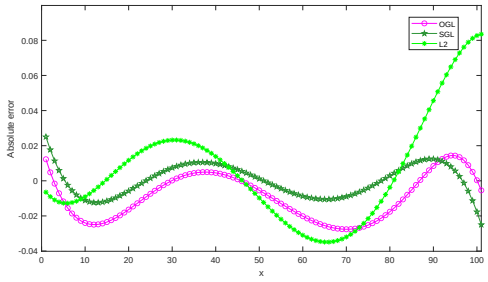

Figure 11: Approximation error diagram.

Now let $g(x) = \cos(x/2\pi) \cdot x$. Employing the same algorithms as before, we approximate the function $g(x) = \cos(x/2\pi) \cdot x$ on the interval $x \in [-1, 1]$. The input consists of 101 samples, with 3 hidden nodes and a single output. The training process stops when 5000 iterations or the error is less than $1e-4$. The penalty term of SGL is 0.001, and the learning rate of SGL is 0.00007. The penalty term of OGL is 0.001, and the learning rate of OGL is 0.0005. The penalty term of L2 is 0.0009, and the learning rate of L2 is 0.0001.

Fig. 10 shows that L2, OGL and SGL approximate $f(x)$, and SGL provides the best approximation. It is apparent from Fig. 11, the absolute error of SGL is minimal.

4.1. Benchmarking with classification problem

To verify the validity of the proposed algorithm, we employ some baseline data sets from the UCI machine learning repository. These include one binary classification problems and three multi-class classification problems, and the details of these data sets are shown in Table 3. Each algorithm runs 15 tests on each set of data and averages them.

In all experiments, the Pi-Sigma neural network model is described similarly in Examples 4.1 and 4.2, which assumed to have with 15 hidden nodes and the output layer nodes are related to several classifications of the data set. The initial weights are randomly chosen in the interval $[-0.5, 0.5]$. The regular term coefficient set to be $\lambda = 0.002$, In addition, each dataset is randomly split into training and testing subset with a percentage 70% to 30%, and we adopt normalization technique to preprocess the training and testing samples. To compare the stability of the algorithms, the k-fold cross validation method is used to train the networks. A training process stops when one of the following two conditions is met: The validation error is below 0.002; or the number of training steps exceeds 5000.

Table 3: Detailed description of the classification data sets.

Data sets	Data size	Training set	Testing set	Input features	Classes
Iris	150	105	45	4	3
Banknote Authentication	1372	960	412	5	2
User Knowledge Modeling	403	282	121	5	4
Vertebral Column	310	217	93	6	3

Table 4: Performance comparison for classification problems.

Data sets	Algorithm	Training accuracy	Testing accuracy	Redundant nodes
Iris	L2	0.9382	0.9113	0
	OGL	0.9233	0.9038	4.5
	SGL	**0.9526**	**0.9221**	**5.6**
Banknote Authentication	L2	0.8743	0.8512	0
	OGL	0.8860	0.8625	5.2
	SGL	**0.9060**	**0.8726**	**6.7**
User Knowledge Modeling	L2	0.8757	0.8336	0
	OGL	0.8612	0.8203	5.9
	SGL	**0.8730**	**0.8403**	**6.8**
Vertebral Column	L2	0.8132	0.8010	0
	OGL	0.8261	0.7909	5.3
	SGL	**0.8513**	**0.8200**	**7.1**

Table 4 shows the results of the training accuracy, the testing accuracy, the number of hidden nodes that have been pruned (Redundant nodes). The improved algorithm has better performance in terms of training accuracy and test accuracy. More importantly, the improved algorithm can effectively perform sparse optimization on the network structure.

5. Conclusion

In this paper, a new Pi-Sigma neural network online gradient learning algorithm based on group lasso regularization terms is proposed, and polishing technology is used to overcome the defect that the objective function is not differentiable at the origin of coordinates. The oscillation phenomenon in the numerical experiment is eliminated, and more importantly, the strong and weak convergence of the algorithm is strictly proved, and the theoretical results of high order neural networks are effectively enriched and developed. At the same time, the effectiveness of the algorithm and the correctness of the theoretical results are verified by the numerical experiments of the classical parity problem and the function approximation problem.

The theoretical results of this paper can provide reference for the theoretical analysis of other types of high order neural networks, such as Sigma-Pi neural network and Ridge polynomial neural network. Unfortunately, the selection of article parameters still adopts the method of experience plus multiple tests to take the average value, which is not efficient. Therefore, in the follow-up work, we will consider using evolutionary algorithms to optimize the parameters. In addition, the structural sparse optimization and theoretical results of convolutional neural networks, a typical representative of deep neural networks, will be considered, and the performance of the algorithm on high-dimensional data sets will be verified.

Acknowledgments

This work was supported by the Natural Science Basic Research Plan in Shaanxi Province of China (Grant No. 2021JM-446) and by the Shaanxi Computer Society and Xi'an Xiangteng Microelectronics Technology Co., Ltd (Grant No. 2023KJ-473).

References

[1] P.L. Bartlett, *For valid generalization the size of the weights is more important than the size of the network*, Adv. Neural Inf. Process Syst. **9**, 134–140 (1997).

[2] E. Bas, E. Egrioglu and E. Kolemen, *A novel intuitionistic fuzzy time series method based on bootstrapped combined Pi-Sigma artificial neural network*, Eng. Appl. Artif. Intell. **114**, 105030 (2022).

[3] Q.W. Fan, Q. Kang and J.M. Zurada, *Convergence analysis for Sigma-Pi-Sigma neural network based on some relaxed conditions*, Inf. Sci. **585**, 70–88 (2022).

[4] Q.W. Fan, Q. Kang, J.M. Zurada, T.W. Huang and D.P. Xu, *Convergence analysis of online gradient method for high-order neural networks and their sparse optimization*, IEEE. Trans. Neural Netw. Learn. Syst. 1–15 (2023). doi: 10.1109/TNNLS.2023.3319989

[5] Q.W. Fan and T. Liu, *Smoothing L_0 regularization for extreme learning machine*, Math. Probl. Eng. **2020**, 1–10 (2020).

[6] T. Heskes and W. Wiegerinck, *A theoretical comparison of batch-mode, on-line, cyclic, and almost-cyclic learning*, IEEE Trans. Neural Netw. **7(4)**, 919–925 (1996).

[7] A.N. Husaini, R. Ghazali, M.N. Nawi, L.H. Ismail, M.M. Deris and T. Herawan, *Pi-Sigma neural network for a one-step-ahead temperature forecasting*, Int. J. Comput. Intell Appl. **13(4)**, 1450023 (2014).

[8] J.A. Hussain and P. Liatsis, *Recurrent Pi-Sigma networks for DPCM image coding*, Neurocomputing. **55**, 363–382 (2003).

[9] J.L. Jiang, *Application of Pi-Sigma neural network to real-time classification of seafloor sediments*, Appl. Acoust. **24**, 346–250 (2005).

[10] Q. Kang, Q.W. Fan and J.M. Zurada, *Deterministic convergence analysis via smoothing group lasso regularization and adaptive momentum for Sigma-Pi-Sigma neural network*, Inf. Sci. **553**, 66–82 (2021).

[11] Q. Kang, Q.W. Fan, J.M. Zurada and T.W. Huang, *A pruning algorithm with relaxed conditions for high-order neural networks based on smoothing group $L_{1/2}$ regularization and adaptive momentum*, Knowl. Based. Syst. **257**, 109858 (2022).

[12] R. Kumar, *A Lyapunov-stability-based context-layered recurrent Pi-Sigma neural network for the identification of nonlinear systems*, Appl. Soft Comput. **122**, 108836 (2022).

[13] T. Liu, S. Chen, S. Liang, S.J. Gan and C.J. Harris, *Fast adaptive gradient RBF networks for online learning of nonstationary time series*, IEEE Trans. Signal Process. **68**, 2015–2030 (2020).

[14] Y. Liu, Z.X. Li, D.K. Yang, K.S. Mohamed, J. Wang and W. Wu, *Convergence of batch gradient learning algorithm with smoothing $L_{1/2}$ regularization for Sigma-Pi-Sigma neural networks*, Neurocomputing. **151**, 333–341 (2015).

[15] Y. Liu and D.K. Yang, *Convergence analysis of the batch gradient-based neuro-fuzzy learning algorithm with smoothing $L_{1/2}$ regularization for first-order Takagi-Sugeno system*, Fuzzy Sets Syst. **319(15)**, 28–49 (2017).

[16] H. Lu, W. Wu and Z. Li, *Convergence of online gradient method with a penalty term for BP neural network with stochastic inputs*, J. Math. Res. Expo. **27(3)**, 643–653 (2007).

[17] L. Meier, S. Van and P. Buhlmann, *The group lasso for logistic regression*, J. R. Stat. Soc. Ser. B. Stat. Methodol. **70**, 53–71 (2008).
[18] K.S. Mohamed, Y. Liu, W. Wu and Z.A. Habtamu, *Batch gradient method for training of Pi-Sigma neural network with penalty*, Int. J. Artif. Intell. Appl. **7(1)**, 11–20 (2016).
[19] K.S. Mohamed, W. Wu and Y. Liu, *A modified higher-order feed forward neural network with smoothing regularization*, Neural Netw. World. **27(6)**, 577–592 (2017).
[20] T. Nakama, *Theoretical analysis of batch and on-line training for gradient descent learning in neural networks*, Neurocomputing **73**, 151–159 (2009).
[21] B.K. Natarajan, *Sparse approximate solutions to linear systems*, SIAM J. Comput. **24(2)**, 227–234 (1995).
[22] S. Noah, J. Friedman, T. Hastie and R. Tibshirani, *A sparse-group lasso*, J. Comput. Graph. Statist. **22(2)**, 231–245 (2013).
[23] R. Reed, *Pruning algorithms: A survey*, IEEE. Trans. Neural Netw. Learn. Syst. **4**, 740–747 (1993).
[24] T. Robert, *Regression shrinkage and selection via the Lasso*, J. R. Stat. Soc. Ser. B. Stat. Methodol. **73(1)**, 273–282 (1996).
[25] D.E. Rumelhart, G.E. Hinton and R.J. Williams, *Learning representations byback-propagating errors*, Nature **323(6088)**, 533–536 (1986).
[26] K. Saito and R. Nakano, *Second-order learning algorithm with squared penalty term*, Neural Comput. **12(3)**, 709–729 (2000).
[27] U.D. Shuanping and M. Song, *An application of Pi-Sigma network in underwater acoustic objects classification*, Acta Acust. **22**, 345–351 (1997).
[28] J. Wang, Q.L. Cai, Q.Q. Chang and J.M. Zurada, *Convergence analyses on sparse feedforward neural networks via group lasso regularization*, Inf. Sci. **381**, 250–269 (2017).
[29] J. Wang, Y.Q. Wen, Z.Y. Ye, L. Jian and H. Chen, *Convergence analysis of BP neural networks via sparse response regularization*, Appl. Soft Comput. **61**, 354–363 (2017).
[30] J. Wang, C. Xu, X.F. Yang and J.M. Zurada, *A novel pruning algorithm for smoothing feedforward neural networks based on group lasso method*, IEEE Trans. Neural Netw. Learn Syst. **29(5)**, 2012–2024 (2018).
[31] P.J. Werbos, *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, Ph.D. Thesis, Harvard University (1974).
[32] D.R. Wilson and T.R. Martinez, *The general inefficiency of batch training for gradient descent learning*, Neural Netw. **16**, 1429–1451 (2003).
[33] W. Wu, J. Wang, M.S. Cheng and Z.X. Li, *Convergence analysis of online gradient method for BP neural networks*, Neural Netw. **24(1)**, 91–98 (2011).
[34] Y. Xiong, W. Wu, X. Kang and C. Zhang, *Training pi-sigma network by online gradient algorithm with penalty for small weight update*, Neural Comput. **19(12)**, 3356–3368 (2007).
[35] Z. Xu, X. Chang, F. Xu and H. Zhang, $L_{1/2}$ *regularization: A thresholding representation theory and a fast solver*, IEEE. Trans. Neural Netw. Learn. Syst. **23(7)**, 1013–1027 (2012).
[36] Z.B. Xu, R. Zhang and W.F. Jing, *When does online BP training convergence*, IEEE. Trans. Neural Netw. **20(10)**, 1529–1539 (2009).
[37] G.D. Xue, J. Wang, K. Zhang and N.R. Pal, *High-dimensional fuzzy inference systems*, IEEE. Trans. Syst. Man. Cybern. Syst. **54(1)**, 507–519 (2023).
[38] B.J. Zhang, X.L. Gong, J. Wang, F.Z. Tang, K. Zhang and W. Wu, *Nonstationary fuzzy neural network based on FCMnet clustering and a modified CG method with Armijo-type rule*, Inf. Sci. **608**, 313–338 (2022).
[39] H. Zhang, W. Wu and M. Yao, *Boundedness and convergence of batch backpropagation algorithm with penalty for feedforward neural networks*, Neurocomputing **89**, 141–146 (2012).

[40] Y.Y. Zhang, S. Li, J. Weng and B.L. Liao, *GNN model for time-varying matrix inversion with robust finite-time convergence*, IEEE. Trans. Neural Netw. Learn. Syst. **35(1)**, 559–569 (2024).
[41] Y.Y. Zhang, J.L. Zhang and J. Weng, *Dynamic Moore-Penrose inversion with unknown derivatives: Gradient neural network approach*, IEEE. Trans. Neural Netw. Learn. Syst. **34(12)**, 10919–10929 (2023).
[42] L. Zhou, Q.W. Fan, X.D. Huang and Y. Liu, *Weak and strong convergence analysis of Elman neural networks via weight decay regularization*, Optimization **72(9)**, 2287–2309 (2022).