Special Issue Paper

Mathematical Methods in the Applied Sciences

Received 1 April 2011

Published online in Wiley Online Library

(wileyonlinelibrary.com) DOI: 10.1002/mma.2641
MOS subject classification: 68T05; 46C05

Functional data learning by Hilbert feedforward neural networks

Jianwei Zhao[*†]

Communicated by T. Qian

This paper focuses on learning algorithms for approximating functional data that are chosen from some Hilbert spaces. An effective algorithm, called Hilbert parallel overrelaxation backpropagation (HPORBP) algorithm, is proposed for training the Hilbert feedforward neural networks that are extensions of feedforward neural networks from Euclidean space $\mathbb{R}^n$ to some Hilbert spaces. Furthermore, the convergence of the iterative HPORBP algorithm is analyzed, and a deterministic convergence theorem is proposed for the HPORBP algorithm on the basis of the perturbation results of Mangasarian and Solodov. Some experimental results of learning functional data on some Hilbert spaces illustrate the convergence theorem and show that the proposed HPORBP algorithm has a better accuracy than the Hilbert backpropagation algorithm.

Keywords: functional data; feedforward neural network; learning algorithm; convergence

1. Introduction

As we all know, feedforward neural networks (FNNs) have widespread application areas, such as pattern recognition, signal processing, and so on, as FNNs can approximate some classes of functions from some data ([1–12]). In those applications of FNNs, the most important problem is how to choose an FNN to realize the interpolation or approximation of an unknown function from some sample data, that is, give a learning algorithm for determining all the parameters (weights) of the FNN. Up to now, there have been many learning algorithms for training FNNs, such as backpropagation (BP) algorithm and its various improved algorithms ([13–16]), radial basis function algorithm [17], extreme learning machine [10], and so on. All these learning algorithms investigate the approximation of some classes of functions that act on the Euclidean space $\mathbb{R}^n$.

However, with the rapid development of information science and technology, more and more high-dimensional data rise in practical applications, such as spatio-temporal images, pattern recognition, and so on, which results in that the higher the dimension of input vector, the more the number of weights to be computed. Rossi *et al*. [18], Preda [19], Muńoz, and Conźlez [20] disposed those high-dimensional data by regarding them as discretized functions. Furthermore, functional analysis is the theory foundation of many modern subjects, such as differential equation, dynamics, control theory, and so on. Therefore, it is a key point to discuss the approximation of some classes of functionals with functional data to improve the development of those modern subjects. Recently, Cao *et al*. [21–24] and Corrieu [25] have investigated the approximation of some special functionals that act on a non-Euclidean space with FNNs from the viewpoint of approximation theory. Zhao *et al*. [26], He, and Wu [8] have studied the learning algorithms for the classification problem from some functional data.

Because the BP algorithm is the most classical and broadly applied learning algorithm for training the FNNs acting on a Euclidean space $\mathbb{R}^n$, we will investigate the corresponding BP algorithms for training the general FNNs that act on some Hilbert spaces. In this paper, we will first construct an FNN acting on a Hilbert space and give a corresponding learning algorithm, called Hilbert parallel overrelaxation backpropagation (HPORBP) algorithm, to learn the functional data. Then, we will analyze the convergence of the iterative HPORBP algorithm and give a deterministic convergence theorem for the HPORBP algorithm on the basis of the perturbation results of Mangasarian and Solodov [14]. At last, we will give some simulated experiments of the proposed HPORBP algorithm for functional data learning to show the good performance of the proposed HPORBP algorithm.

We organize the rest of this paper as follows. Section 2 constructs a Hilbert feedforward neural network (HFNN) and proposes the HPORBP algorithm for training the HFNNs. Section 3 discusses the convergence of the iterative HPORBP algorithm and gives a deterministic convergence theorem for the HPORBP algorithm by means of the perturbation results of Mangasarian and Solodov [14].

Department of Mathematics, China Jiliang University, Hangzhou 310018, Zhejiang Province, China
**Correspondence to: Jianwei Zhao, Department of Mathematics, China Jiliang University, Hangzhou 310018, Zhejiang Province, China.*
† E-mail: zhaojw@amss.ac.cn

Experiments on the performance of the HPORBP algorithm compared with Hilbert backpropagation (HBP) algorithm on some Hilbert space are given in Section 4. Section 5 gives the conclusions of this paper.

2. Hilbert parallel overrelaxation backpropagation algorithm

In this section, we will construct an HFNN and propose the HPORBP algorithm for functional data learning on the basis of the theory of functional analysis [27] and the perturbation results of Mangasarian and Solodov [14].

2.1. Hilbert feedforward neural networks

At first, we give a definition of HFNN with M layers based on the theory of functional analysis to provide an approximation tool for functional data.

In this paper, let $\mathcal{H}$ be a real Hilbert space with an inner product $\langle , \rangle$, and $\mathcal{H}^n$ be the direct sum of n Hilbert spaces $\mathcal{H}$ with the inner product $\langle , \rangle_n$ defined by

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{H}^n} = \sum_{i=1}^{n} \langle x_i, y_i \rangle, \tag{2.1}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_n]^\top, \mathbf{y} = [y_1, y_2, \ldots, y_n]^\top \in \mathcal{H}^n$, and $\top$ is the transpose of a matrix or a vector (see [27]). Then, the definition of an HFNN with M layers is given as follows.

Definition 2.1
For a real Hilbert space $\mathcal{H}$, an HFNN with M layers is constructed as in Figure 1, and the ith output $y_i^{\bar{m}}$ of the mth layer can be expressed as

$$y_i^{\bar{m}} = f_m \left(n_i^{\bar{m}} \right), \tag{2.2}$$

where

$$n_i^{\bar{m}} = \begin{cases} \sum_{j=1}^{n} \langle w_{i,j}^{\bar{1}}, x_j \rangle, & m = 1; \\ \sum_{j=1}^{S^{m-1}} w_{i,j}^{\bar{m}} y_j^{\overline{m-1}}, & m = 2, \ldots, M; \end{cases} \qquad i = 1, 2, \ldots, S^m, \tag{2.3}$$

$\mathbf{x} = [x_1, \ldots, x_n]^\top \in \mathcal{H}^n$, $w_{i,j}^{\bar{1}} \in \mathcal{H}$ is the weight connecting the ith neuron in the first layer with the jth input, $i = 1, 2, \ldots, S^1, j = 1, 2, \ldots, n$, $w_{i,j}^{\bar{m}} \in \mathbb{R}$ is the weight connecting the ith neuron in the mth layer with the jth output of the $(m-1)$th layer, $i = 1, 2, \ldots, S^m, j = 1, 2, \ldots, S^{m-1}$, and f_m is the activation function for the mth layer neuron, $m = 1, 2, 3, \ldots, M$.

Remark 2.2
When the Hilbert space $\mathcal{H}^n = \mathbb{R}^n$, then the HFNN with M layers in Definition 2.1 is the traditional M layers FNN.

2.2. Proposed Hilbert parallel overrelaxation backpropagation algorithm

Let $\left\{ \left(\xi^{(i)}, \eta^{(i)} \right) \right\}_{i=1}^{p}$ be a set of p functional data, where $\xi^{(i)} = \left[\xi_1^{(i)}, \xi_2^{(i)}, \ldots, \xi_n^{(i)} \right]^\top \in \mathcal{H}^n$ and $\eta^{(i)} = \left[\eta_1^{(i)}, \eta_2^{(i)}, \ldots, \eta_{S^2}^{(i)} \right]^\top \in \mathbb{R}^{S^M}$. Now, we begin to give a learning algorithm, named HPORBP algorithm, for learning the weights of the HFNNs with the theory of functional

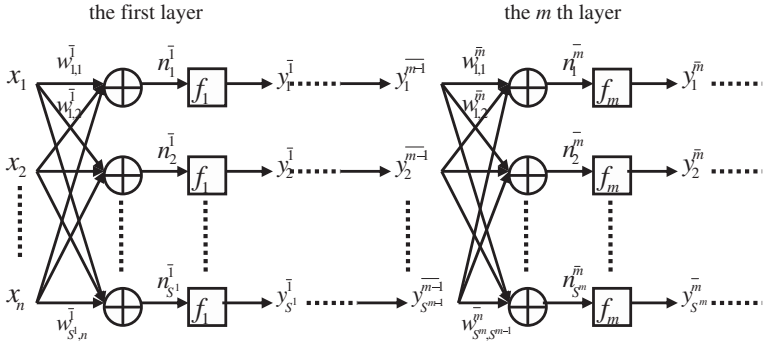

Figure 1. A Hilbert feedforward neural networks with M layers acting on a Hilbert space.

analysis and the idea of parallel BP algorithm. Because the set of FNNs is dense in the set of continuous functionals, we will train the HFNNs with two layers, that is, $M = 2$. It is sure that this algorithm can be generalized to the case for $M \in \mathbb{N}$.

Suppose that the derivative of each activation function f_1, f_2 is continuous, that is, $f_1, f_2 \in C^1$. Let $W^{\bar{m}} = \left[w_{i,j}^{\bar{m}}\right]_{S^m \times S^{m-1}}$, $m = 1, 2$, then the individual error functional for the sample datum $\left(\xi^{(k)}, \eta^{(k)}\right)$ is defined by

$$\phi_k\left(W^{\bar{1}}, W^{\bar{2}}\right) = \frac{1}{2}\sum_{i=1}^{S^2}\left(\eta_i^{(k)} - O_i^{\bar{2}}\left(\xi^{(k)}\right)\right)^2, \quad k = 1, 2, \ldots, p, \tag{2.4}$$

and the objective functional we want to minimize is

$$\phi\left(W^{\bar{1}}, W^{\bar{2}}\right) = \sum_{k=1}^{p}\phi_k\left(W^{\bar{1}}, W^{\bar{2}}\right), \tag{2.5}$$

where $O_i^{\bar{2}}(\xi^{(k)})$ is the ith output of the second layer in (2.2) when the input vector $\mathbf{x} = \xi^{(k)}$, $i = 1, 2, \ldots, S^2$.

As we all know, BP algorithm is the steepest descent algorithm for each individual error functional ϕ_k, that is,

$$w_{i,j}^{\bar{m}}(k+1) = w_{i,j}^{\bar{m}}(k) - \alpha_k \frac{\partial \phi_k}{\partial w_{i,j}^{\bar{m}}}, \tag{2.6}$$

where α_k is the learning speed, $m = 1, 2, i = 1, 2, \ldots, S^m, j = 1, 2, \ldots, S^{m-1}$. However, for the HFNNs with two layers, we can compute the partial derivative $\partial\phi_k / \partial w_{i,j}^{\bar{2}}$ of the functional ϕ_k w.r.t. the variable $w_{i,j}^{\bar{2}} \in \mathbb{R}$, $i = 1, 2, \ldots, S^2, j = 1, 2, \ldots, S^1$, but we cannot compute $\partial\phi_k / \partial w_{i,j}^{\bar{1}}$ similarly for $w_{i,j}^{\bar{1}} \in \mathcal{H}$. So, it is necessary to look for more general theory of derivation to solve the problem. Here, we will use the Fréchet derivative of a functional at one point [28].

Definition 2.3 ([28])
Let X and Y be Banach spaces and $U \subseteq X$ be an open subset of X. A map $f : U \longrightarrow Y$ is called Fréchet differentiable at $\mathbf{x} \in U$ if there exists a bounded linear operator $D_{f,\mathbf{x}} : X \longrightarrow Y$ such that

$$\lim_{\mathbf{h}\to 0}\frac{\|f(\mathbf{x}+\mathbf{h}) - f(\mathbf{x}) - D_{f,\mathbf{x}}(\mathbf{h})\|_Y}{\|\mathbf{h}\|_X} = 0. \tag{2.7}$$

And the bounded linear operator $D_{f,\mathbf{x}} : X \to Y$ is called the Fréchet derivative of f at $\mathbf{x}$.

Lemma 2.4
Suppose that $f_{\mathbf{y}}$ is the bounded linear functional defined by $f_{\mathbf{y}}(\mathbf{x}) = \langle \mathbf{x}, \mathbf{y}\rangle$ on a Hilbert space $\mathcal{H}$, then the Fréchet derivative of $f_{\mathbf{y}}$ at $\mathbf{x} \in \mathcal{H}$ is $D_{f_{\mathbf{y}},\mathbf{x}} = y$.

Proof
Because

$$\begin{aligned}\lim_{\mathbf{h}\to 0}&\frac{|f_{\mathbf{y}}(\mathbf{x}+\mathbf{h}) - f(\mathbf{x}) - D_{f_{\mathbf{y}},\mathbf{x}}(\mathbf{h})|}{\|\mathbf{h}\|}\\ &= \lim_{\mathbf{h}\to 0}\frac{|\langle \mathbf{x}+\mathbf{h}, \mathbf{y}\rangle - \langle \mathbf{x}, \mathbf{y}\rangle - \langle \mathbf{h}, \mathbf{y}\rangle|}{\|h\|}\\ &= 0,\end{aligned}$$

then $D_{f_{\mathbf{y}},\mathbf{x}}(\mathbf{h}) = \langle \mathbf{h}, \mathbf{y}\rangle$ for any $\mathbf{h}$ in $\mathcal{H}$. Therefore, $D_{f_{\mathbf{y}},\mathbf{x}} = \mathbf{y}$ by Reisz representation theorem [27]. □

It is well known that the chain rule holds for Fréchet derivative [28], that is, for a pair of C^1 maps $F : U \to V$ and $G : V \to Z$, there holds

$$D_{G\circ F,\mathbf{u}}(\mathbf{h}) = D_{G,F(\mathbf{u})}(D_{F,\mathbf{u}}(\mathbf{h})), \tag{2.8}$$

where X, Y, Z are Fréchet spaces and U, V are open subsets of X and Y, respectively.

As the activation functions $f_1, f_2 \in C^1$ and Hilbert space is a Fréchet space, then we can obtain

$$\begin{aligned}D_{\phi_k, w_{i,j}^{\bar{1}}}(\mathbf{h}) &= D_{\phi_k, n_i^{\bar{1}}}\left(D_{n_i^{\bar{1}}, w_{i,j}^{\bar{1}}}(\mathbf{h})\right)\\ &= \frac{\partial \phi_k}{\partial n_i^{\bar{1}}}\left\langle \mathbf{h}, \xi_j^{(k)}\right\rangle\\ &= \left\langle \mathbf{h}, \frac{\partial \phi_k}{\partial n_i^{\bar{1}}}\xi_j^{(k)}\right\rangle\end{aligned}$$

by applying the rule chain for $\phi_k\left(w_{i,j}^{\bar{1}}\right) = \phi_k\left(n_i^{\bar{1}}\left(w_{i,j}^{\bar{1}}\right)\right)$.

Hence, by Reisz representation theorem,

$$D_{\phi_k, w_{i,j}^{\bar{1}}} = \frac{\partial \phi_k}{\partial n_i^{\bar{1}}} \xi_j^{(k)}. \tag{2.9}$$

Let $r_i^{\bar{m}} \equiv \frac{\partial \phi_k}{\partial n_i^{\bar{m}}}$ denote the sensitivity of ϕ_k to changes in the ith element of the network input in the mth layer, then

$$\frac{\partial \phi_k}{\partial w_{i,j}^{\bar{m}}} = r_i^{\bar{m}} O_j^{\overline{m-1}}. \tag{2.10}$$

Now, we can propose the HPORBP algorithm for training HFNN with two layers on the basis of the improved parallel BP algorithm as follows.

Proposed HPORBP Algorithm 2.5

For $T(T \geq 1)$ parallel processors, divide the set of functional data $\{(\xi^{(i)}, \eta^{(i)})\}_{i=1}^{p}$ into T classes $B_l = \{(\xi^{(\mu)}, \eta^{(\mu)}) | \mu \in J_l\}, l = 1, 2, \ldots, T$, where each $J_l = \{l_1, \ldots, l_{n_l}\}$ and $\{J_l | l = 1, 2, \ldots, T\}$ is a partition of $\{1, 2, \ldots, p\}$. Choose a set of initial weights $\{w_{i,j}^{\bar{m}}(0) | i = 1, 2, \ldots, S^m, j = 1, 2, \ldots, S^{m-1}, m = 1, 2\}$ and set $k = 0$. Having $w_{i,j}^{\bar{m}}(k)$, stop if $\frac{\partial \phi}{\partial w_{i,j}^{\bar{m}}} = 0$ for all weights, else compute $w_{i,j}^{\bar{m}}(k+1)$ as follows.

- For $l = 1$ to T, put $w_{i,j}^{\bar{m},l} = w_{i,j}^{\bar{m}}, i = 1, 2, \ldots, S^m, j = 1, 2, \ldots, S^{m-1}, m = 1, 2$.
  - For $\mu = l_1$ to l_{n_l}, calculate the output $O^{\bar{m}}(\mu)$ of the mth layer when the input vector $\mathbf{x} = \xi^{(\mu)}, m = 1, 2$;
    - For $i = 1$ to S^2,
      for $j = 1$ to S^1, calculate

$$\begin{cases} r_i^{\bar{2}} = -\dfrac{d(f_2)}{d(n_i^{\bar{2}})}(\eta_i^{(\mu)} - f_2(n_i^{\bar{2}})); \\ w_{i,j}^{\bar{2},l}(new) = w_{i,j}^{\bar{2},l}(old) - \alpha_\mu r_i^{\bar{2}} O_j^{\bar{1}}(\mu); \\ n_i^{\bar{2}} \leftarrow n_i^{\bar{2}} + (w_{i,j}^{\bar{2}}(new) - w_{i,j}^{\bar{2}}(old)) O_j^{\bar{1}}(\mu). \end{cases} \tag{2.11}$$

      end j;
    end i.
    - For $i = 1$ to S^1,
      for $j = 1$ to n, calculate

$$\begin{cases} r_i^{\bar{1}} = \dfrac{d(f_1)}{d(n_i^{\bar{1}})} \displaystyle\sum_{k=1}^{S^2} w_{k,i}^{\bar{2}} r_k^{\bar{2}}; \\ w_{i,j}^{\bar{1},l}(new) = w_{i,j}^{\bar{1},l}(old) - \alpha_\mu r_i^{\bar{1}} \xi_j^{(\mu)}; \\ n_i^{\bar{1}} \leftarrow n_i^{\bar{1}} + \langle w_{i,j}^{\bar{1},l}(new) - w_{i,j}^{\bar{1},l}(old), \xi_j^{(\mu)} \rangle_{\mathcal{H}}. \end{cases} \tag{2.12}$$

      end j;
    end i;
  end μ;
end l;
- Calculate

$$w_{i,j}^{\bar{m}}(k+1) = \sum_{l=1}^{T} \frac{n_l}{p} w_{i,j}^{\bar{m},l}(k), \quad i = 1, 2, \ldots, S^m, j = 1, 2, \ldots, S^{m-1}, m = 1, 2 \tag{2.13}$$

Remark 2.6

When the Hilbert space $\mathcal{H}^n$ is taken to be the Euclidean space $\mathbb{R}^n$ and $T = 1$, the HPORBP algorithm is the serial overrelaxation backpropagation algorithm acting on the Euclidean space $\mathbb{R}^n$ proposed in the paper [14].

Remark 2.7

The serial overrelaxation backpropagation algorithm is improved on the basis of the traditional BP algorithm. So, by the similar method of HPORBP algorithm, the traditional BP algorithm can also be generalized to the corresponding BP algorithm acting on a Hilbert space, named HBP algorithm.

3. Convergence of the Hilbert parallel overrelaxation backpropagation algorithm

White [29] has proven that the sequence of weights produced by the traditional BP algorithm diverges or converges almost surely under stochastic assumptions. So, it is necessary to discuss the convergence of the HPORBP algorithm for training the functional data. In this section, we will give a deterministic convergence theorem for HPORBP algorithm by generalizing the results of the paper [14] from the space $\mathbb{R}^n$ to $\mathcal{H}^n$. For convenience, we will first give a general minimization problem and prove the convergence theorem for a minimization algorithm that induces the HPORBP algorithm.

Suppose that we have $T(T \geq 1)$ parallel processors and $\{J_l | l = 1, 2, \ldots, T\}$ be a partition of $\{1, 2, \ldots, p\}$, where $J_l = \{l_1, \ldots, l_{n_l}\}$. So, our goal is the unconstrained minimization problem

$$\min_{\mathbf{w} \in \mathcal{H}^m} \sum_{l=1}^{T} \sum_{j=1}^{n_l} \frac{n_l}{p} \phi_{l_j}(\mathbf{w}). \tag{3.1}$$

The following algorithm is the generalization of the parallel algorithm studied in [14] from the space $\mathbb{R}^n$ to $\mathcal{H}^n$.

Proposed Parallel Algorithm 3.1

Start with any $\mathbf{w}^0 \in \mathcal{H}^m$, set $i = 0$. Having $\mathbf{w}^i$, stop if $\nabla\phi(\mathbf{w}^i) = 0$, else calculate $\mathbf{w}^{i+1}$ as follows.

- For $l = 1$ to T, put $\mathbf{w}^{i,l,1} = \mathbf{w}^i$;
  - For $j = 1$ to n_l, use $\mathbf{w}^{i,l,j}$ to compute $\mathbf{w}^{i,l,j+1}$, where the kth component of $\mathbf{w}^{i,l,j+1}$ is calculated by the following formula:

$$w_k^{i,l,j+1} = w_k^{i,l,j} - \alpha_i \frac{\partial \phi_{l_j}}{\partial w_k}(\mathbf{w}^{i,l,j,k}), \tag{3.2}$$

here

$$\mathbf{w}^{i,l,j,k} = \left[w_1^{i,l,j+1}, \ldots, w_{k-1}^{i,l,j+1}, w_k^{i,l,j}, \ldots, w_m^{i,l,j}\right]^\top \tag{3.3}$$

  end j;

end l;

- Calculate

$$\mathbf{w}^{i+1} = \sum_{l=1}^{T} \frac{n_l}{p} \mathbf{w}^{i,l,n_l+1}. \tag{3.4}$$

Now, we are ready to prove the convergence of the proposed parallel Algorithm 3.1. If f has Lipschitz continuous partial derivatives on $\mathcal{H}^m$ with some constant $L > 0$, that is, $||\nabla f(\mathbf{w}) - \nabla f(\mathbf{v})|| \leq L||\mathbf{w} - \mathbf{v}||$, we write $f \in LC_L^1(\mathcal{H}^m)$.

Lemma 3.2
If $\phi \in LC_L^1(\mathcal{H}^m)$, then for any $\mathbf{w}, \mathbf{v} \in \mathcal{H}^m$,

$$|\phi(\mathbf{w}) - \phi(\mathbf{v}) - \langle \nabla\phi(\mathbf{v}), \mathbf{w} - \mathbf{v}\rangle| \leq \frac{L}{2}\|\mathbf{w} - \mathbf{v}\|^2. \tag{3.5}$$

Proof
For any $\mathbf{w}, \mathbf{v} \in \mathcal{H}^m$, define a function ψ on [0,1] by $\psi(t) = \phi(\mathbf{v} + t(\mathbf{w} - \mathbf{v}))$. Because $\phi \in LC_L^1(\mathcal{H}^m)$, then

$$\begin{aligned}
\phi(\mathbf{w}) - \phi(\mathbf{v}) &= \psi(1) - \psi(0) \\
&= \int_0^1 \psi'(t)dt \\
&= \int_0^1 \langle \nabla\phi(\mathbf{v} + t(\mathbf{w} - \mathbf{v})), \mathbf{w} - \mathbf{v}\rangle dt \\
&= \left\langle \int_0^1 \nabla\phi(\mathbf{v} + t(\mathbf{w} - \mathbf{v}))dt, \mathbf{w} - \mathbf{v} \right\rangle.
\end{aligned}$$

Therefore,

$$\begin{aligned}|\phi(\mathbf{w})-\phi(\mathbf{v})-\langle\nabla\phi(\mathbf{v}),\mathbf{w}-\mathbf{v}\rangle| &= \left|\left\langle\int_0^1[\nabla\phi(\mathbf{v}+t(\mathbf{w}-\mathbf{v}))-\nabla\phi(\mathbf{v})]\mathrm{d}t,\mathbf{w}-\mathbf{v}\right\rangle\right| \\ &\le \left\|\int_0^1[\nabla\phi(\mathbf{v}+t(\mathbf{w}-\mathbf{v}))-\nabla\phi(\mathbf{v})]\mathrm{d}t\right\|\cdot\|\mathbf{w}-\mathbf{v}\| \\ &\le \int_0^1\|\nabla\phi(\mathbf{v}+t(\mathbf{w}-\mathbf{v}))-\nabla\phi(\mathbf{v})\|\mathrm{d}t\cdot\|\mathbf{w}-\mathbf{v}\| \\ &\le L\int_0^1 t\|\mathbf{w}-\mathbf{v}\|^2\mathrm{d}t \\ &= \frac{L}{2}\|\mathbf{w}-\mathbf{v}\|^2\end{aligned}$$

by the Cauchy–Schwartz inequality on the Hilbert space $\mathcal{H}^m$ [27]. □

Lemma 3.3
Suppose that

$$\phi\in LC_L^1(\mathcal{H}^m),\ \inf_{\mathbf{w}\in\mathcal{H}^m}\phi(\mathbf{w})>-\infty,\ \|\nabla\phi(\mathbf{w})\|\le M,\forall\mathbf{w}\in\mathcal{H}^m \tag{3.6}$$

for some $M>0$. Start with any $\mathbf{w}^0\in\mathcal{H}^m$. Having $\mathbf{w}^i$, stop if $\nabla\phi(\mathbf{w}^i)=0$, else compute

$$\mathbf{w}^{i+1}=\mathbf{w}^i+\alpha_i\mathbf{d}^i, \tag{3.7}$$

where $\mathbf{d}^i=-\nabla\phi(\mathbf{w}^i)+\mathbf{e}^i$ for some $\mathbf{e}^i\in\mathcal{H}^m$. Choose $\alpha_i\in\mathbb{R}^+$ such that

$$\sum_{i=1}^{\infty}\alpha_i=\infty,\ \sum_{i=1}^{\infty}\alpha_i^2<\infty,\ \sum_{i=1}^{\infty}\alpha_i\|\mathbf{e}^i\|<\infty,\ \|\mathbf{e}^i\|\le\gamma\ \ (\forall i\in\mathbb{N},\ \gamma>0). \tag{3.8}$$

Then, it follows that

(1) $\{\phi(\mathbf{w}^i)\}$ converges;
(2) $\{\nabla\phi(\mathbf{w}^i)\}\to 0$;
(3) For each accumulation point $\bar{\mathbf{w}}$ of the sequence $\{\mathbf{w}^i\}$, $\nabla\phi(\bar{\mathbf{w}})=0$.

Proof
By the Cauchcy–Schwartz inequality, we have

$$\begin{aligned}-\langle\nabla\phi(\mathbf{w}^i),\mathbf{d}^i\rangle &= \langle\nabla\phi(\mathbf{w}^i),\nabla\phi(\mathbf{w}^i)-\mathbf{e}^i\rangle \\ &= \|\nabla\phi(\mathbf{w}^i)\|^2-\langle\nabla\phi(\mathbf{w}^i),\mathbf{e}^i\rangle \\ &\ge \|\nabla\phi(\mathbf{w}^i)\|^2-\|\nabla\phi(\mathbf{w}^i)\|\cdot\|\mathbf{e}^i\| \\ &\ge \|\nabla\phi(\mathbf{w}^i)\|^2-M\|\mathbf{e}^i\|.\end{aligned}$$

Because $\|\mathbf{d}^i\|=\|-\nabla\phi(\mathbf{w}^i)+\mathbf{e}^i\|\le\|\nabla\phi(\mathbf{w}^i)\|+\|\mathbf{e}^i\|\le M+\gamma$,

$$\begin{aligned}\phi(\mathbf{w}^i)-\phi(\mathbf{w}^{i+1}) &\ge \langle\nabla\phi(\mathbf{w}^i),\mathbf{w}^i-\mathbf{w}^{i+1}\rangle-\frac{L}{2}\|\mathbf{w}^i-\mathbf{w}^{i+1}\|^2 \\ &= -\alpha_i\langle\nabla\phi(\mathbf{w}^i),\mathbf{d}^i\rangle-\frac{L}{2}\alpha_i^2\|\mathbf{d}^i\|^2 \\ &\ge -\alpha_i\langle\nabla\phi(\mathbf{w}^i),\mathbf{d}^i\rangle-\frac{L}{2}\alpha_i^2(M+\gamma)^2\end{aligned}$$

by Lemma 3.2. Let $\lambda_i=M\|\mathbf{e}^i\|, v_i=\frac{L}{2}\alpha_i^2(M+\gamma)^2$, then the aforementioned conclusions hold by the similar method in the proof of Theorem 2.1 in [14]. □

Theorem 3.4
Suppose that

$$\phi\in LC_L^1(\mathcal{H}^m),\ \inf_{\mathbf{w}\in\mathcal{H}^m}\phi(\mathbf{w})>-\infty,\ \|\nabla\phi(\mathbf{w})\|\le M,\forall\mathbf{w}\in\mathcal{H}^m$$

for some $M > 0$. If the sequence $\{\alpha_i\}_{i\in\mathbb{N}}$ of learning rate satisfies

$$\sum_{i=1}^{\infty}\alpha_i = \infty,\ \sum_{i=1}^{\infty}\alpha_i^2 < \infty, \tag{3.9}$$

then for any sequence $\{\mathbf{w}^i\}_{i\in\mathbb{N}}$ obtained by the parallel algorithm, it satisfies

(1) $\{\phi(\mathbf{w}^i)\}$ converges;
(2) $\{\nabla\phi(\mathbf{w}^i)\}\to 0$;
(3) For each accumulation point $\bar{\mathbf{w}}$ of the sequence $\{\mathbf{w}^i\}$, $\nabla\phi(\bar{\mathbf{w}}) = 0$.

Proof
It is sufficient to show that the assumptions of Lemma 3.3 hold for this case. From the updating formula (32) for $\mathbf{w}_k^{i,l,j+1}$, $k = 1, 2, \ldots, m$, we have

$$\begin{aligned} w_k^{i,l,j+1} - w_k^{i,l,1} &= \sum_{t=1}^{j}\left(w_k^{i,l,t+1} - w_k^{i,l,t}\right) \\ &= -\alpha_i\sum_{t=1}^{j}\frac{\partial\phi_{l_t}}{\partial w_k}\left(\mathbf{w}^{i,l,t,k}\right). \end{aligned}$$

Therefore,

$$\begin{aligned} \mathbf{w}_k^{i+1} - \mathbf{w}_k^i &= \sum_{l=1}^{T}\frac{n_l}{p}\mathbf{w}_k^{i,l,n_l+1} - \mathbf{w}_k^{i,l,1} \\ &= \sum_{l=1}^{T}\frac{n_l}{p}\left(\mathbf{w}_k^{i,l,n_l+1} - \mathbf{w}_k^{i,l,1}\right) \\ &= -\alpha_i\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l}{p}\frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^{i,l,t,k}) \\ &= -\alpha_i\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l}{p}\frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^i) + \alpha_i\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l}{p}\left[\frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^i) - \frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^{i,l,t,k})\right] \\ &= \alpha_i\left[\mathbf{e}_k^i - \sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l}{p}\frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^i)\right], \end{aligned}$$

where $\mathbf{e}_k^i = \sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l}{p}\left[\frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^i) - \frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^{i,l,t,k})\right]$. Now, we only have to verify that there exists some $\gamma > 0$ such that

$$\sum_{i=1}^{\infty}\alpha_i\|\mathbf{e}^i\| < \infty,\ \|\mathbf{e}^i\| \le \gamma. \tag{3.10}$$

Because

$$\begin{aligned} \|\mathbf{e}^i\| &= \sqrt{\sum_{k=1}^{m}||\mathbf{e}_k^i||^2} \\ &= \sqrt{\sum_{k=1}^{m}\left\|\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l}{p}\left[\frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^i) - \frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^{i,l,t,k})\right]\right\|^2} \\ &\le \sqrt{\sum_{k=1}^{m}\left[\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l}{p}\left\|\frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^i) - \frac{\partial\phi_{l_t}}{\partial w_k}(\mathbf{w}^{i,l,t,k})\right\|\right]^2} \\ &\le \sqrt{\sum_{k=1}^{m}\left[\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l L}{p}||\mathbf{w}^i - \mathbf{w}^{i,l,t,k}||\right]^2} \end{aligned}$$

$$\leq \sqrt{\sum_{k=1}^{m}\left[\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l L}{p}\sqrt{\sum_{r=1}^{m}||\mathbf{w}_r^i-\mathbf{w}_r^{i,l,t,k}||^2}\right]^2}$$

$$\leq \sqrt{\sum_{k=1}^{m}\left[\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l L}{p}\sqrt{\sum_{r=1}^{m}\left(\max\left\{\left\|\mathbf{w}_r^i-\mathbf{w}_r^{i,l,t}\right\|,\left\|\mathbf{w}_r^i-\mathbf{w}_r^{i,l,t+1}\right\|\right\}\right)^2}\right]^2}$$

$$\leq \alpha_i L\sqrt{\sum_{k=1}^{m}\left[\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l}{p}\sqrt{\sum_{r=1}^{m}\left\|\sum_{h=1}^{v(t)}\frac{\partial\phi_{l_h}}{\partial w_r}(\mathbf{w}^{i,l,h,r})\right\|^2}\right]^2},$$

where $v(t)=t-1$ or $v(t)=t$, and the gradient of the functional ϕ is bounded, there exists $H>0$ such that

$$\sqrt{\sum_{k=1}^{m}\left[\sum_{l=1}^{T}\sum_{t=1}^{n_l}\frac{n_l}{p}\sqrt{\sum_{r=1}^{m}||\sum_{h=1}^{v(t)}\frac{\partial\phi_{l_h}}{\partial w_r}(\mathbf{w}^{i,l,h,r})||^2}\right]^2}\leq H. \tag{3.11}$$

By the way of choosing α_i, we can easily obtain that there exists $\gamma>0$ such that $\|\mathbf{e}^i\|\leq\gamma$ and

$$\sum_{i=1}^{\infty}\alpha_i\|e^i\|\leq\sum_{i=1}^{\infty}\alpha_i^2 LH=LH\sum_{i=1}^{\infty}\alpha_i^2<\infty. \tag{3.12}$$

□

Remark 3.1

In the proposed HPORBP Algorithm 2.5, the functional $\phi(W^{\bar{1}},W^{\bar{2}})$ to be minimized is the mean of individual functionals $\phi_{l_\mu}(W^{\bar{1}},W^{\bar{2}})$ with respect to the sample data. While each functional is the squared composition of linear functionals and the activation functions f_1 and f_2. There are many proposed methods for choosing those active functions, which make each individual error functional and the total error functional both satisfy the assumptions of Theorem 3.4.

4. Experiments

In this section, we will illustrate the performances of the proposed HPORBP algorithm, such as accuracy, complexity, convergence, and stability by comparing with HBP algorithm. We will learn the functional data in some Hilbert space by an HFNN with two layers. All the experiments are conducted in MATLAB 6.5 (MathWorks Company, Natick, Massachusetts, USA) environment running on a desktop with CPU 2.69 GHZ and 1.96 GB RAM (Lianxiang Company, Peking, China).

The usual root-mean-square error (RMSE) is taken to be the measure of accuracy of a learning algorithm, that is, for N sampling data $\{(\xi^{(i)},t^{(i)})\}_{i=1}^{p}\subseteq\mathcal{H}\times\mathbb{R}$ and a learned functional f,

$$\mathrm{RMSE}(f)=\left[\frac{1}{p}\sum_{i=1}^{p}|f(\xi^{(i)})-t^{(i)}|^2\right]^{\frac{1}{2}}. \tag{4.1}$$

Let $\mathcal{H}=L^2[0,1]$ be the Hilbert space of real Lebesgue measurable functions acting on $[0,1]$ with

$$\int_{[0,1]}|x(t)|^2\mathrm{d}t<+\infty, \tag{4.2}$$

where the inner product on $\mathcal{H}$ is

$$\langle x_1,x_2\rangle=\int_{[0,1]}x_1(t)x_2(t)\mathrm{d}t. \tag{4.3}$$

For convenience, we will take the functional data (ξ,η) with the form that $\xi\in\mathcal{H}$ and $\eta\in\mathbb{R}$ in the following experiments. Furthermore, because Riemann integrable functions are dense in $\mathcal{H}$, we can do the experiments on the set of Riemann integrable functions, so that the inner product can be computed by computers.

Experiment 4.1

Choose a set of functional data $\{(\xi^{(k)},\eta^{(k)})\}_{k=1}^{5}$ as follows:

$$D_1=\{(2t+1,3.8556),(\sin 3t,3.1117)\},\ D_2=\left\{(2t^2,3.2887),(\cos(2t+1),1.4691)\right\},\ D_3=\left\{(t^3,2.7105)\right\}. \tag{4.4}$$

We will choose an HFNN with two layers as in (2.2) acting on the Hilbert space $L^2[0,1]$ to learn functional data as in (4.4), where $S^1 = 3$ and $S^2 = 1$, the numbers of hidden nodes in the first and second layers , and the sequence of learning rate $\{\alpha_i\}$ is taken to be

$$f_1(t) = \frac{1}{1+e^{-t}}, \quad f_2(t) = t, \quad \text{and} \quad \alpha_i = \frac{2}{(i/500)+9}, \tag{4.5}$$

respectively.

Now, we begin to train the HFNN in (2.2) by HBP algorithm and HPORBP Algorithm 3.1 with the following three groups of different initial weights with 40 iterations.

$$(A)\ w_{1,1}^{\bar{1}}(t) = 3t, w_{2,1}^{\bar{1}}(t) = 3, w_{3,1}^{\bar{1}}(t) = -t^2, w_{1,1}^{\bar{2}} = 3, w_{1,2}^{\bar{2}} = 1, w_{1,3}^{\bar{2}} = 0.5; \tag{4.6}$$

$$(B)\ w_{1,1}^{\bar{1}}(t) = 2, w_{2,1}^{\bar{1}}(t) = t, w_{3,1}^{\bar{1}}(t) = 3t^2, w_{1,1}^{\bar{2}} = 1, w_{1,2}^{\bar{2}} = 3, w_{1,3}^{\bar{2}} = 1; \tag{4.7}$$

$$(C)\ w_{1,1}^{\bar{1}}(t) = 1, w_{2,1}^{\bar{1}}(t) = 3t, w_{3,1}^{\bar{1}}(t) = t^3, w_{1,1}^{\bar{2}} = 4, w_{1,2}^{\bar{2}} = 2, w_{1,3}^{\bar{2}} = 5 \tag{4.8}$$

Table I gives the comparison of training RMSE and training time(s) between HBP and HPORBP algorithms for training the same HFNN and functional data with different initial weights. The experimental results imply that HPORBP algorithm takes a little more training time than HBP algorithm, but it has an obviously better accuracy than that of HBP algorithm.

Figure 2 shows the relation of total error function with the number of iteration of the HPORBP and HBP algorithms when the initial weights are taken to be the group (B) of (4.7). As observed from Figure 2, the total error of HBP algorithm vibrates with the number of iteration. However, HPORBP algorithm has a stable convergence.

Because the initial weights for HBP and HPORBP algorithms are randomly assigned, it is necessary to consider the stability of these two learning algorithms. Here, the standard deviation is taken to be a tool for measuring the stability of a learning algorithm. That is,

Table I. Comparison of training RMSE and time(s) between HBP and HPORBP algorithms on $L^2[0,1]$.

	HPORBP algorithm		HBP algorithm	
Initial weights	RMSE	Time(s)	RMSE	Time(s)
Group (A)	0.0153	4.199	0.0244	4.012
Group (B)	0.0688	4.614	0.0828	4.571
Group (C)	0.1004	5.126	0.2890	5.032

RMSE, root-mean-square error; HBP, Hilbert backpropagation; HPORBP, Hilbert parallel overrelaxation backpropagation.

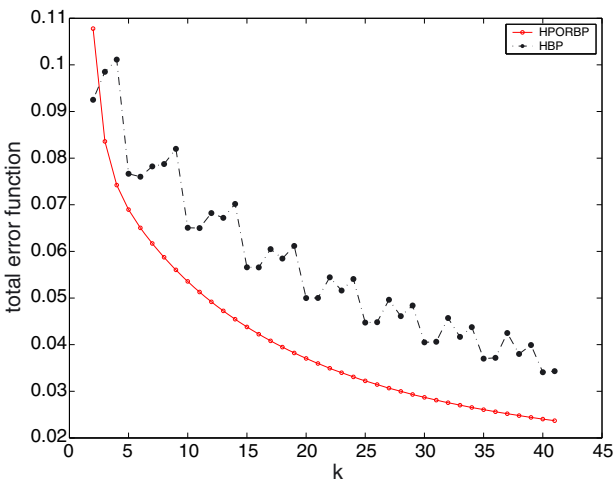

Figure 2. Total error function of the Hilbert parallel overrelaxation backpropagation (HPORBP) and Hilbert backpropagation (HBP) algorithms with the numbers of iteration.

for a set of functional data $\{(\xi^{(i)}, t^{(i)})\}_{i=1}^{p} \subseteq \mathcal{H} \times \mathbb{R}$ and q runs with randomly assigned initial weights, the standard deviation σ of a learning algorithm is given by

$$\sigma = \left[\frac{1}{q}\sum_{k=1}^{q}\sum_{i=1}^{p}|f_k(\xi^{(i)}) - \bar{\mathbf{u}}^{(i)}|^2\right]^{\frac{1}{2}}, \tag{4.9}$$

where f_k is the k-th learned functional by the k-th experiment, and

$$\bar{\mathbf{u}}^{i} = \frac{1}{q}\sum_{k=1}^{q} f_k\left(\xi^{(i)}\right), \quad i = 1, 2, \ldots, p. \tag{4.10}$$

Experiment 4.2

In this experiment, we will compare the stability of HPORBP and HBP algorithms by standard deviation of training HFNNs acting on the Hilbert space $L^2[0, 1]$ with 20 runs. The set of functional data and the real Hilbert space $\mathcal{H}$ are taken to be the same as in Experiment 4.1. We train the HFNN same as in Experiment 4.1 by HBP algorithm and HPORBP Algorithm 2.5 for 20 runs with 20 groups of different initial weights, respectively.

According to the formula (4.9), the standard deviations of HBP and HPORBP algorithms are 0.4457 and 0.1596, respectively, which implies that HPORBP algorithm has a better stability than HBP algorithm.

5. Conclusions

In this paper, we firstly constructed an HFNN with M layers for functional data learning. Secondly, we proposed an HPORBP algorithm for training the HFNNs. In the proposed HPORBP algorithm, each processor is computed simultaneously, and the weights are updated by the mean of each weight in different processors. Adopting this algorithm, all the recent information is used to obtain the new weights. Thirdly, we gave a deterministic convergence theorem of the HPORBP algorithm by generalizing the results of paper [14] from the Euclidean space $\mathbb{R}^n$ to $\mathcal{H}^n$. Some experimental results on functional data learning on some Hilbert spaces indicate that the proposed HPORBP algorithm is convergent, and it has a better accuracy and a strong stability than the HBP algorithm.

Acknowledgements

The research was supported by the National Natural Science Foundation of China(No. 61101240), the Zhejiang Provincial Natural Science Foundation of China (No. Y6110117), and the Science Foundation of Zhejiang Education Office (No. Y201122002).

References

1. Anastassiou GA. Univariate fuzzy-random neural network approximation operators. *Computer and Mathematics with Applications* 2004; **48**:1263–1283.
2. Anastassiou GA. Multivariate sigmoidal neural network approximation. *Neural Networks* 2011; **24**:378–386.
3. Anastassiou GA. Multivariate hyperbolic tangent neural network approximation. *Computer and Mathematics with Applications* 2011; **61**:809–821.
4. Cao FL, Xie TF, Xu ZB. The estimate for approximation error of neural networks: a constructive approach. *Neurocomputing* 2008; **71**:626–630.
5. Chen TP, Chen H, Liu RW. Approximation capability in $C(\mathbb{R}^n)$ by multiplayer feedforward networks and related problems. *IEEE Transactions on Neural Networks* 1995; **6**:25–30.
6. Cybenko G. Approximation by superposition of sigmoidal functions. *Mathematics of Control, Signals and System* 1989; **2**:303–314.
7. Funahashi KI. On the approximate realization of continuous mappings by neural networks. *Neural Networks* 1989; **2**:183–192.
8. He Q, Wu CX. Separating theorem of samples in Banach space for support vector machine learning. *International Journal of Machine and Cybernetics* 2011; **2**:49–54.
9. Hornik K. Some new results on neural network approximation. *Neural Networks* 1993; **6**:1069–1072.
10. Huang GB, Zhu QY, Siew C. KExtreme learning machine: theory and applications. *Neurocomputing* 2006; **70**:489–501.
11. Xu ZB, Cao FL. The essential order of approximation for neural networks. *Sciences in China, Series F* 2004; **47**:97–112.
12. Xu ZB, Cao FL. Simultaneous L^p approximation order for neural networks. *Neural Networks* 2005; **18**:914–923.
13. Fukuoka Y, Matsuki H, Minamitani H. A modified backpropagation method to avoid false local minima. *Neural Networks* 1998; **11**:1059–1072.
14. Mangasarian OL, Solodov MV. Serial and parallel backpropagation convergence via nonmonmonotone perturbed minimization. *Optimization Methods and Software* 1994; **4**:103–116.
15. Rumelhart DE, Hinton GE, Williams RJ. Learning representations by backpropagation errors. *Nature* 1986; **323**:533–536.
16. Wu W, Wang J, Cheng MS, Li ZX. Convergence analysis of online gradient method for BP neural networks. *Neural Networks* 2011; **24**:91–98.
17. Park J, Sandberg IW. Universal approximation using radial-basis-function networks. *Neural Computation* 1991; **3**:246–257.
18. Rossi F, Delannay N, Guez BC, Verleysen M. Representation of functional data in neural networks. *Neurocomputing* 2005; **64**:83–210.
19. Preda C. Regression models for functional data by reproducing kernel Hilbert spaces methods. *Journal of Statistical Planning and Inference* 2007; **137**:829–840.
20. Muńoz A, Gonźlez J. Representing functional data using support vector machines. *Pattern Recognition Letters* 2010; **31**:511–516.
21. Cao FL, Zhang R. The errors of approximation for feedforward neural networks in the L^p metric. *Mathematical and Computer Modelling* 2009; **49**:1563–1572.
22. Cao FL, Zhang YQ, He ZR. Interpolation and rate of convergence for a class of neural networks. *Applied Mathematical Modelling* 2009; **33**:1441–1456.

23. Cao FL, Lin SB, Xu ZB. Constructive approximate interpolation by neural networks in the metric space. *Mathematical and Computer Modelling* 2010; **52**:1674–1681.
24. Cao FL, Lin SB, Xu ZB. Approximation capability of interpolation neural networks. *Neurocomputing* 2010; **74**:457–460.
25. Corrieu P. Function approximation on non-Euclidean spaces. *Neural Networks* 2005; **18**:91–102.
26. Zhao JW, Wang ZH, Cao FL. Hilbert perceptron and its applications in separation theorem. *Journal of Computational Information Systems* 2010; **6**:3247–3256.
27. Conway JB. *A Course in Functional Analysis*. GTM, Springer-Verlag: New York, 1985.
28. Yamamuro S. Differential calauls in toplogical linear spaces. In *Lecture Notes in Mathematics*. Springer-Verlag: New York, 1970; 374.
29. White H. Some asymptotic results for learning in single hidden-layer feedforward network models. *Journal of the American Statistical Association* 1989; **84**:1003–1013.