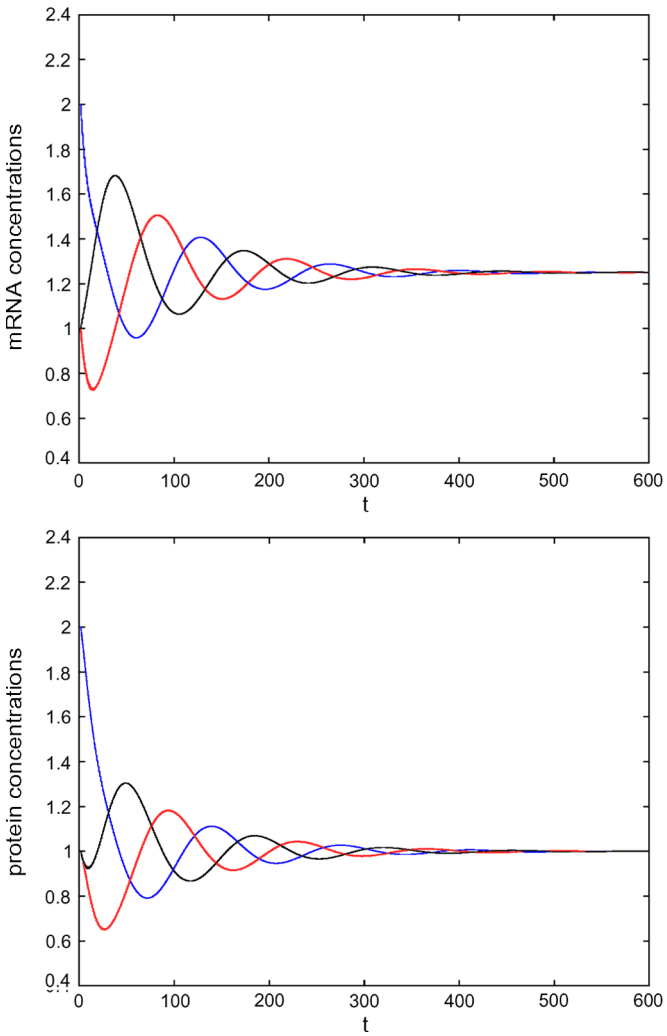

Fig. 1. Numeric simulation of (19).

Moreover, the work in [23] presents a powerful tool to study the GRNs and it will be considered in the future.

REFERENCES

[1] T. S. Gardner, C. R. Cantor, and J. J. Collins, "Construction of a genetic toggle switch in Escherichia coli," *Nature*, vol. 403, pp. 339–342, 2000.
[2] D. W. Austin, M. S. Allen, J. M. McCollum, R. D. Dar, J. R. Wilgus, G. S. Sayler, N. F. Samatova, C. D. Cox, and M. L. Simpson, "Gene network shaping of inherent noise spectra," *Nature*, vol. 439, pp. 608–611, 2006.
[3] H. McAdams and L. Shapiro, "Circuit simulation of genetic networks," *Science*, vol. 269, pp. 650–656, 1995.
[4] N. A. M. Monk, "Oscillatory expression of Hes1, p53, and NF−κB driven by transcriptional time delays," *Current Biol.*, vol. 13, pp. 1409–1413, 2003.
[5] B. Grammaticos, A. S. Carstea, and A. Ramani, "On the dynamics of a gene regulatory network," *J. Phys. A, Math. Gen.*, vol. 39, pp. 2965–2971, 2006.
[6] K. Gu, "An integral inequality in the stability problem of time-delay systems," in *Proc. IEEE Conf. Decision Control*, Australia, Dec. 2000, pp. 2805–2810.
[7] P. Smolen, D. A. Baxter, and J. H. Byrne, "Modelling circadian oscillations with interlocking positive and negative feedback loops," *J. Neurosci.*, vol. 21, pp. 6644–6656, 2001.
[8] P. Smolen, D. A. Baxter, and J. H. Byrne, "Mathematical modeling of gene networks," *Neuron*, vol. 26, pp. 567–580, 2000.
[9] H. De Jong, "Modelling and simulation of genetic regulatory systems: A literature review," *J. Comput. Biol.*, vol. 9, pp. 67–103, 2002.
[10] H. Bolouri and E. H. Davidson, "Modelling transcriptional regulatory networks," *BioEssay*, vol. 24, pp. 1118–1129, 2002.
[11] A. Becskei and L. Serrano, "Engineering stability in gene networks by autoregulation," *Nature*, vol. 405, pp. 590–593, 2000.
[12] L. Chen and K. Aihara, "Stability of genetic regulatory networks with time delay," *IEEE Trans. Circuits Syst. I, Fund. Theory Appl.*, vol. 49, no. 5, pp. 602–608, May 2002.
[13] C. Li, L. Chen, and K. Aihara, "Stability of genetic networks with SUM regulatory logic: Lur'e system and LMI approach," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 11, pp. 2451–2458, Nov. 2006.
[14] C. Li, L. Chen, and K. Aihara, "Synchronization of coupled nonidentical genetic oscillators," *Phys. Biol.*, vol. 3, pp. 37–44, 2006.
[15] J. Liang and J. Cao, "Exponential stability of continuous-time and discrete-time bidirectional associative memory networks with delays," *Chaos Solitons Fractals*, vol. 22, pp. 773–785, 2004.
[16] Y. Liu, Z. Wang, A. Serrano, and X. Liu, "Discrete-time recurrent neural networks with time-varying delays: Exponential stability analysis," *Phys. Lett. A*, vol. 362, no. 5–6, pp. 480–488, 2007.
[17] S. Mohamad and A. Naim, "Discrete-time analogues of integrodifferential equations modelling bidirectional neural networks," *J. Comput. Appl. Math.*, vol. 138, pp. 1–20, 2002.
[18] M. B. Elowitz and S. Leibler, "A synthetic oscillatory network of transcriptional regulators," *Nature*, vol. 403, pp. 335–338, 2000.
[19] T. Kobayashi, L. Chen, and K. Aihara, "Modeling genetic switches with positive feedback loops," *J. Theor. Biol.*, vol. 221, pp. 379–399, 2003.
[20] R. Wang, T. Zhou, Z. Jing, and L. Chen, "Modelling periodic oscillation of biological systems with multiple time scale networks," *Syst. Biol.*, vol. 1, pp. 71–84, 2004.
[21] C.-H. Yuh, H. Bolouri, and E. H. Davidson, "Genomic cis-regulatory logic: Experimental and computational analysis of a sea urchin gene," *Science*, vol. 279, pp. 1896–1902, 1998.
[22] S. Kalir, S. Mangan, and U. Alon, "A coherent feed-forward loop with a SUM input function prolongs flagella expression in *Escherichia coli*," *Molecular Syst. Biol.*, 2005, 10.1038/msb4100010.
[23] Z. Wang, F. Yang, D. W. C. Ho, and X. Liu, "Robust H—Infinity filtering for stochastic time-delay systems with missing measurements," *IEEE Trans. Signal Process.*, vol. 54, no. 7, pp. 2579–2587, Jul. 2006.

Wavelet Basis Function Neural Networks for Sequential Learning

Ning Jin and Derong Liu

***Abstract*—In this letter, we develop the wavelet basis function neural networks (WBFNNs). It is analogous to radial basis function neural networks (RBFNNs) and to wavelet neural networks (WNNs). In WBFNNs, both the scaling function and the wavelet function of a multiresolution approximation (MRA) are adopted as the basis for approximating functions. A sequential learning algorithm for WBFNNs is presented and compared to the sequential learning algorithm of RBFNNs. Experimental results show that WBFNNs have better generalization property and require shorter training time than RBFNNs.**

***Index Terms*—Radial basis function neural network (RBFNN), sequential learning, wavelet basis function neural network (WBFNN).**

I. INTRODUCTION

Radial basis function neural networks (RBFNNs) are used to approximate complex functions directly from the input–output data with a simple topological structure [2], [3], [13]–[15], [18]. RBFNNs have

Manuscript received April 20, 2007; revised August 18, 2007; accepted September 7, 2007. This work was supported in part by the National Science Foundation under Grants ECCS-0621694, ECS-0529292, and ECS-0355364.

The authors are with the Department of Electrical and Computer Engineering, University of Illinois, Chicago, IL 60607-7053 USA (e-mails: njin@ece.uic.edu; dliu@ece.uic.edu).

Digital Object Identifier 10.1109/TNN.2007.911749

good generalization ability as compared to the multilayer feedforward neural networks. In an RBFNN, a function $f(x)$ is approximated as

$$\hat{f}(x) = \sum_i w_i \phi\left(\frac{\|x - a_i\|}{b_i}\right) \tag{1.1}$$

where $\phi(r)$ is the basis function. The most commonly used basis function is the Gaussian function $\exp(-r^2/2)$. In sequential learning, a neural network is trained to approximate a function while a series of training sample pairs are randomly drawn and presented to the network. The sample pairs are learned by the network one by one. There are several different sequential learning algorithms for RBFNNs [3], [5], [6], [9], [10], [12], [16], [17].

Wavelet neural networks (WNNs) are also used to approximate functions by a single basis function [1], [7], [8], [19]–[21]. In a WNN, a function $f(x)$ is approximated as

$$\hat{f}(x) = \sum_i w_i \phi\left(\frac{x - a_i}{b_i}\right) \tag{1.2}$$

where $\phi(x)$ is the basis function coming from wavelet theory [4], [11]—the scaling function, the wavelet function, or the basis function of continuous wavelet transform. WNNs can approximate functions more accurately and they have better generalization property than RBFNNs. However, all existing training algorithms are not specially designed for sequential learning and the orthogonal properties of wavelets have not been used in these algorithms.

In this letter, we study wavelet basis function neural networks (WBFNNs) for sequential learning. Both the scaling function ϕ and the wavelet function ψ are used as basis functions in WBFNNs. Functions ϕ and ψ are orthogonal to each other. In a wavelet decomposition of a function, they will give approximations in different level of details, i.e., coarse and fine approximations, respectively. In a WBFNN, function $f(x)$ is approximated as

$$\hat{f}(x) = \sum_i \alpha_i \phi\left(\frac{x - a_i}{b_i}\right) + \sum_i \beta_i \psi\left(\frac{x - c_i}{d_i}\right). \tag{1.3}$$

A sequential learning algorithm will be developed from the orthogonal properties of multiresolution approximation (MRA) and Mallat's formula of wavelet decomposition [4], [11].

This letter is organized as follows. Section II gives a brief review of the wavelet theory. Section III presents the definition of WBFNNs and provides a sequential learning algorithm for WBFNNs. Section IV shows quantitative performance comparisons between the sequential learning algorithm of WBFNNs and RBFNNs. Section V summarizes the conclusion.

II. MRA and Wavelets

In this section, a brief review of the wavelet theory is given (cf. [4] and [11] for details). A *wavelet basis* is constructed with an MRA of function space. An MRA presents a way to approximate functions in multiple resolutions. Recall that the inner product of functions f and $g \in L^2$ is defined as $\langle f, g\rangle = \int f(x)g(x)dx$. An MRA of the function space L^2 is a doubly infinite nested sequence of subspaces of L^2

$$\cdots \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \cdots$$

with the following properties [4], [11].

S2.1) $\cup_j V_j$ is dense in L^2, i.e., $\overline{\lim_j V_j} = L^2$.

S2.2) $\cap_j V_j = \{0\}$.

S2.3) $f(x) \in V_j \iff f(2x) \in V_{j+1}$ for all $j \in \mathbb{Z}$.

S2.4) $f(x) \in V_j \iff f(x - 2^{-j}k) \in V_j$ for all $j, k \in \mathbb{Z}$.

S2.5) There exists a function $\phi \in L^2$ so that $\{\phi(x - k) : k \in \mathbb{Z}\}$ forms an orthonormal basis of V_0, where $\mathbb{Z}$ is the set of all integers. The function ϕ is called the *scaling function* of the MRA $\{V_j\}$.

S2.6) Let $\phi_j^k(x) = 2^{j/2}\phi(2^j x - k)$ for $j, k \in \mathbb{Z}$. Then, for any fixed $j \in \mathbb{Z}$, the set of functions $\{\phi_j^k : k \in \mathbb{Z}\}$ is an orthonormal basis of V_j.

S2.7) Let W_0 be the orthogonal complement of V_0 in V_1, i.e., $W_0 \perp V_0$ and $V_1 = V_0 \oplus W_0$, where the symbol $\oplus$ denotes the orthogonal direct sum of function spaces. Then, there exists a function $\psi \in W_0$ such that $\langle \phi, \psi\rangle = 0$, $\|\psi\| = 1$, and $\{\psi(x - k) : k \in \mathbb{Z}\}$ is an orthonormal basis of W_0. ψ is called the *wavelet function* of the MRA $\{V_j\}$.

S2.8) Let $\psi_j^k(x) = 2^{j/2}\psi(2^j x - k)$ for $j, k \in \mathbb{Z}$. Let $W_j = \overline{\mathrm{Span}\{\psi_j^k : k \in \mathbb{Z}\}}$. Then, the set of functions $\{\psi_j^k : k \in \mathbb{Z}\}$ is an orthonormal basis of W_j, $W_j \perp V_j$, and $V_{j+1} = V_j \oplus W_j$.

S2.9) For any $J, H \in \mathbb{Z}$, $J < H$, we have

$$V_H = V_J \oplus W_J \oplus W_{J+1} \oplus \cdots \oplus W_{H-1}.$$

Furthermore

$$L^2 = \overline{\lim_{H \to \infty} V_H} = V_J \oplus (\oplus_{k=J}^{\infty} W_k).$$

S2.10) For any $f(x) \in L^2$ and any integer $j \in \mathbb{Z}$, there exists a unique function $P_j f \in V_j$ such that $f - P_j f \perp V_j$. $P_j f$ is called the approximation of f at resolution level j. We have $\lim_{j \to \infty} P_j f = f$.

S2.11) For any $f(x) \in L^2$ and any integer $j \in \mathbb{Z}$, there exists a unique function $Q_j f \in W_j$ such that $f - Q_j f \perp W_j$. $Q_j f$ is called the deviation of f at resolution level j. We have $P_j f + Q_j f = P_{j+1} f$. Consequently, for any $J, H \in \mathbb{Z}$, $J < H$, we have

$$P_H f = P_J f + \sum_{J \le k < H} Q_k f.$$

S2.12) Let $f \in L^2$. Since $P_j f \in V_j$ and $\{\phi_j^k : k \in \mathbb{Z}\}$ is an orthonormal basis of V_j, we have

$$P_j f = \sum_k C_{j,k} \phi_j^k$$

where

$$C_{j,k} = \left\langle f, \phi_j^k \right\rangle. \tag{2.1}$$

S2.13) Let $f \in L^2$. Since $Q_j f \in W_j$ and $\{\psi_j^k : k \in \mathbb{Z}\}$ is an orthonormal basis of W_j, we have

$$Q_j f = \sum_k D_{j,k} \psi_j^k$$

where

$$D_{j,k} = \left\langle f, \psi_j^k \right\rangle. \tag{2.2}$$

S2.14) Let $f \in L^2$. For any given $J, H \in \mathbb{Z}$, $J < H$, we have the approximation

$$f(x) \approx \sum_{k \in \mathbb{Z}} C_{J,k} \phi_J^k(x) + \sum_{j=J}^{H} \sum_{k \in \mathbb{Z}} D_{j,k} \psi_j^k(x). \tag{2.3}$$

Equation (2.3) is called the wavelet approximation of f with resolutions from J to H. It is also called the wavelet decomposition of f with resolutions from J to H. The coefficients

$C_{j,k}$ and $D_{j,k}$ in (2.3) can be calculated by the inner product given in S2.12) and S2.13). However, in numerical computation, it will take a long time to obtain all these coefficients by inner products. Fortunately, there is a fast algorithm, i.e., the "fast wavelet decomposition" of Mallat and Daubechies.

For the scaling function ϕ and the associated wavelet function ψ of an MRA $\{V_j\}$, we know that $V_0 \subset V_1$ and $W_0 \subset V_1$. Hence, ϕ and ψ can be written in terms of the basis $\{\phi_1^k\}$ of V_1. For $k \in \mathbb{Z}$, define

$$\begin{cases} h_k = \left\langle \phi(x), \phi_1^k(x) \right\rangle \\ g_k = \left\langle \psi(x), \phi_1^k(x) \right\rangle \end{cases}. \tag{2.4}$$

Then

$$\begin{cases} g_k = (-1)^{k-1} h_{1-k} \\ \sum_s h_s h_{s-2k} = \delta_{0k} \end{cases}$$

and

$$\begin{cases} \phi(x) = \sum_{k\in\mathbb{Z}} h_k \phi_1^k(x) \\ \psi(x) = \sum_{k\in\mathbb{Z}} g_k \phi_1^k(x) \end{cases} \tag{2.5}$$

where $\delta_{0k} = 1$ if $k = 0$ and $\delta_{0k} = 0$ for all $k \neq 0$. It was proved that in the wavelet decomposition (2.3), the weights $C_{j,k}$ and $D_{j,k}$ of different resolution levels have the following relationship:

$$\begin{cases} C_{j-1,m} = \sum_{k\in\mathbb{Z}} h_{k-2m} C_{j,k} \\ D_{j-1,m} = \sum_{k\in\mathbb{Z}} g_{k-2m} C_{j,k} \end{cases}. \tag{2.6}$$

Equation (2.6) is the so-called *fast wavelet decomposition algorithm* of Mallat and Daubechies. It describes the way to obtain the coefficients $C_{j,k}$ and $D_{j,k}$ for lower resolution level from the coefficients with higher resolution level. Now, we have two ways to obtain the coefficients $C_{j,k}$ and $D_{j,k}$ in the wavelet decomposition (2.3). We can calculate all the $C_{j,k}$ and $D_{j,k}$ by inner product according to (2.1) and (2.2). We can also obtain $C_{j,k}$ and $D_{j,k}$ by fast wavelet decomposition (2.6), in case we have already had the values of $C_{j,k}$ and $D_{j,k}$ for some higher resolution level.

III. WBFNNs and Sequential Learning

WBFNNs are analogous to both RBFNNs and WNNs. Same as WNNs, the structure of WBFNNs is based on the theory of wavelets. Instead of one basis function, both the scaling function and the wavelet function of an MRA are used as basis functions in a WBFNN. Besides, we use not only the ability of wavelets to approximate functions, but also the orthogonal properties and the relationships between the approximations of different resolution levels. Similar to RBFNNs, WBFNNs can be employed for sequential learning, but WBFNNs can be trained faster, perform more accurately, and have better generalization properties.

The structure of WBFNNs are based on the approximation in (2.3). J and H are given to indicate the minimum and the maximum of the resolution levels. There are two types of neurons in a WBFNN, equipped with activation function ϕ_j^k and ψ_j^k, respectively.

For purpose of real-world applications, we assume that the domain X of the function $f(x)$ is finite. We will choose Daubechies' wavelets with compact support as the basis. The Daubechies' wavelets have the following properties.

S3.1) Suppose that the scaling function ϕ and wavelet function ψ form a Daubechies' wavelet. Let h_k be the coefficients defined in (2.4). Then, there is a positive integer K_s such that $h_k \neq 0$ only when $k = 0, \ldots, K_s$. Furthermore, the support sets of ϕ and ψ are $\operatorname{supp}\phi = [0, K_s)$ and $\operatorname{supp}\psi = [1 - K_s, 1)$, respectively.

S3.2) For $j, k \in \mathbb{Z}$

$$\begin{aligned} \operatorname{supp}\phi_j^k &= \left[2^{-j}k, 2^{-j}(k + K_s)\right) \\ \operatorname{supp}\psi_j^k &= \left[2^{-j}(k + 1 - K_s), 2^{-j}(k + 1)\right) \\ \operatorname{supp}\phi_j^k \cap \operatorname{supp}\psi_j^k &= \left[2^{-j}k, 2^{-j}(k + 1)\right). \end{aligned}$$

When J is small enough, we have $X \subseteq [2^{-J}k_0, 2^{-J}(k_0 + 1))$ for certain choice of k_0. We only need to consider those neurons with $X \cap \operatorname{supp}\phi_j^k \neq \emptyset$ or $X \cap \operatorname{supp}\psi_j^k \neq \emptyset$, so the neurons of resolution level J in the network are $\phi_J^{k_0-k}$ and $\psi_J^{k_0+k}$ for $0 \leq k \leq K_s - 1$. We call these neurons the root neurons. All the other neurons are $\psi_j^k(x)$ for $J < j \leq H$.

The neurons ϕ_j^k and ψ_j^k will fire on an input x only if x is inside the interval $\operatorname{supp}\phi_j^k = [2^{-j}k, 2^{-j}(k + K_s))$ or $\operatorname{supp}\psi_j^k = [2^{-j}(k + 1 - K_s), 2^{-j}(k + 1))$. The length of these intervals is $2^{-j}K_s$. During the training of the network, for a training sample pair (x_i, y_i), we need to adjust the root neurons and those neurons such that

$$x_i \in \left[2^{-j}(k + 1 - K_s), 2^{-j}(k + 1)\right). \tag{3.1}$$

It is easy to see that (3.1) is equivalent to

$$2^j x_i - 1 < k \leq 2^j x_i + K_s - 1. \tag{3.2}$$

Hence, the output of the whole network is

$$\begin{aligned} \hat{f}(x) &= \sum_{k=0}^{K_s-1} \left(\varpi_k \phi_J^{k_0-k}(x) + w_{J,k_0+k}\psi_J^{k_0+k}(x)\right) \\ &\quad + \sum_{j=J+1}^{H} \sum_{k=\lfloor 2^j x \rfloor}^{\lfloor 2^j x + K_s - 1 \rfloor} w_{j,k} k \psi_j^k(x) \\ &= \sum_{k=0}^{K_s-1} \varpi_k \phi_J^{k_0-k}(x) \\ &\quad + \sum_{j=J}^{H} \sum_{k=\lfloor 2^j x \rfloor}^{\lfloor 2^j x + K_s - 1 \rfloor} w_{j,k} k \psi_j^k(x) \end{aligned} \tag{3.3}$$

where $\lfloor K \rfloor$ means the biggest integer not bigger than K. For each j, the number of k satisfying (3.2) is always K_s. Hence, since we have the limit that $J \leq j \leq H$, for each input x (or training sample (x_i, y_i)), at most $(H - J + 2)K_s$ neurons will fire (or be adjusted during the training).

A WBFNN will grow during the training. It starts with root neurons $\phi_J^{k_0-k}$, $0 \leq k < K_s$, and sequentially increases (or decreases, when some existing neurons have very small significance) the number of neurons until the approximation error is sufficiently small. The growth of neurons will depend on the error of the approximation, the position of existing neurons, and the resolution level of existing neurons. To adjust the weights of neurons, we use (2.1), (2.2), and (2.6). Now, we give our growing and pruning algorithm for sequential learning of WBFNNs. Given $e_{\min}$ to be the minimal error of the approximation and given $\delta_{\min}$ to be the minimal significance of the neurons, the network is initialized as $\hat{f}(x) = \sum_{k=0}^{K_s-1} \varpi_k \phi_J^{k_0-k}(x)$ with $\varpi_k = 0$. The sequence of sample pairs $\{(x_i, y_i)\}$ are chosen randomly with respect to certain probability distribution.

SLWBF Algorithm (sequential learning of wavelet basis function neural networks)

A01. For each pair (x_i, y_i), do steps A02 and A03.

A02. Compute the overall network output $\hat{f}(x_i) = \sum_k \varpi_k \phi_J^{k_0-k}(x_i) + \sum_{j,k} w_{j,k}\psi_j^k(x_i)$. Find the error $e = y_i - \hat{f}(x_i)$.

A03. If $|e| > e_{\min}$, then do steps A04–A08.

A04. Find the maximal resolution level H_1 of existing neurons which will fire at x_i. If $H_1 < H$, then $H_1 = H_1 + 1$.

A05. Define the error function

$$E(x) = \max\left\{0, 1 - 2^{H_1}|x - x_i|\right\} e.$$

A06. From $j = H_1$ to $j = J$, compute

$$\triangle C_{j,k} = \begin{cases} \left\langle E(x), \phi_{H_1}^k \right\rangle, & \text{if } j = H_1 \\ \sum_{s\in\mathbb{Z}} h_{s-2k} \triangle C_{j+1,s}, & \text{if } J \le j < H_1 \end{cases}$$

$$\triangle D_{j,k} = \begin{cases} \left\langle E(x), \psi_{H_1}^k \right\rangle, & \text{if } j = H_1 \\ \sum_{s\in\mathbb{Z}} g_{s-2k} \triangle C_{j+1,s}, & \text{if } J \le j < H_1 \end{cases}$$

where $2^j x_i - 1 < k \le 2^j x_i + K_s - 1$.

A07. Adjust the weights by $\varpi_k = \varpi_k + \triangle C_{J,k_0-k}$ and $w_{j,k} = w_{j,k} + \triangle D_{j,k}$.

A08. If $|w_{j,k}| \ge \delta_{\min}$ and there is no neuron with activation function ψ_j^k, add a neuron $w_{j,k}\psi_j^k$.
If $|w_{j,k}| < \delta_{\min}$ and the neuron $w_{j,k}\psi_j^k$ exists, prune this neuron.

The parameters $e_{\min}$, $\delta_{\min}$, and H determine the accuracy of the algorithm. Smaller $e_{\min}$ and $\delta_{\min}$ give better approximation while larger H gives smaller errors in approximation.

IV. Numerical Experiments

In this section, numerical experiments are performed for the SLWBF algorithm. The experiments run on a Dell Optiplex 745 computer using Matlab. As a comparison of our WBFNNs and SLWBF algorithm, we choose sequential learning algorithm "minimal resource allocation network" (MRAN) [18] of RBFNNs. In [18], MRAN algorithm was compared with some other sequential learning algorithms of RBFNNs, such as resource allocation network (RAN) [12] and resource allocation of network via extended Kalman filter (RANEKF) [5], and so on. The results in [18] show that MRAN has better performance in terms of generalization, network size, and training speed, whenever the input data are uniformly or not uniformly distributed. Let

$$f(x) = \begin{cases} \max\left\{1 - x/2, \sin(x^4/50)\right\}, & \text{if } 0 \le x \le 2 \\ \sin(x^4/50), & \text{if } 2 < x \le 10 \end{cases}. \quad (4.1)$$

Fig. 1(a) is the graph of $f(x)$. We will approximate $f(x)$ by RBFNNs and WBFNNs, respectively, while the training data are generated under uniform distribution or Gaussian distribution $N(4, 1.25^2)$.

In each case, 10 000 training pairs (x_i, y_i) $(i = 1, \ldots, 10\,000)$ are taken from the range $X = [0, 10]$ under according probability distribution. For testing the resulting neural networks, we use 1000 data which is uniformly distributed in the same range $x \in [0, 10]$. The wavelet adopted is Haar wavelet, which is the wavelet with the smallest support among Daubechies' wavelets. For Haar wavelet, the parameter

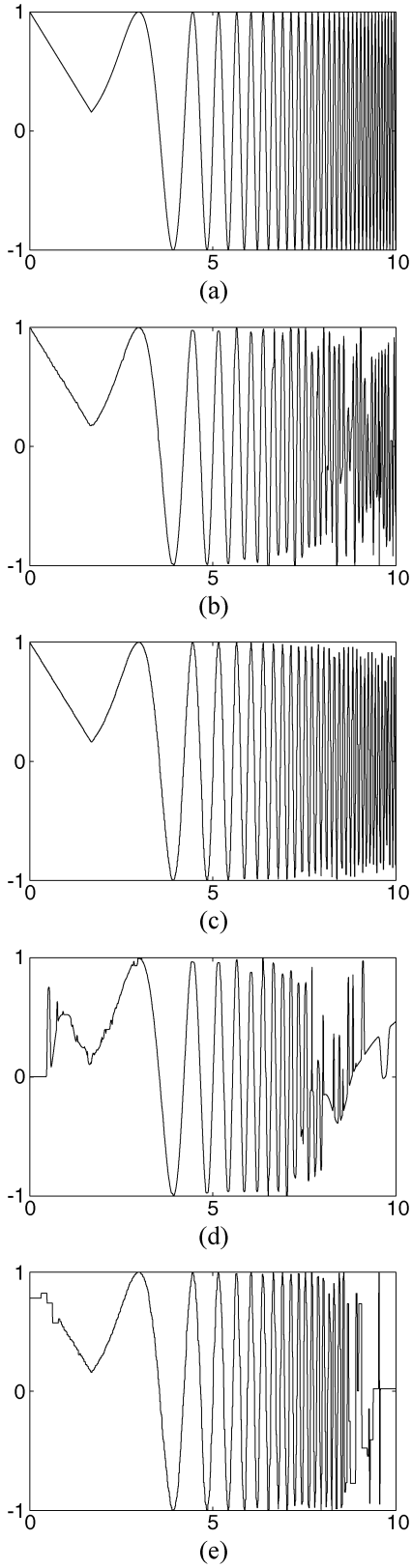

Fig. 1. $f(x)$ and $\hat{f}(x)$. (a) $f(x)$. (b) RBF and uniform distribution. (c) WBF and uniform distribution. (d) RBF and Gaussian distribution. (e) WBF and Gaussian distribution.

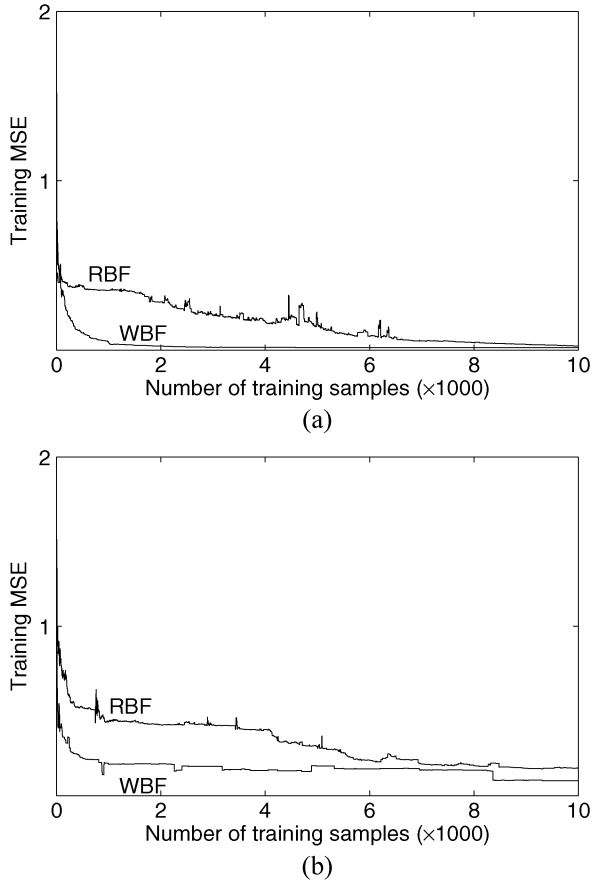

Fig. 2. Comparison of mean square errors. (a) Training data with uniform distribution. (b) Training data with Gaussian distribution.

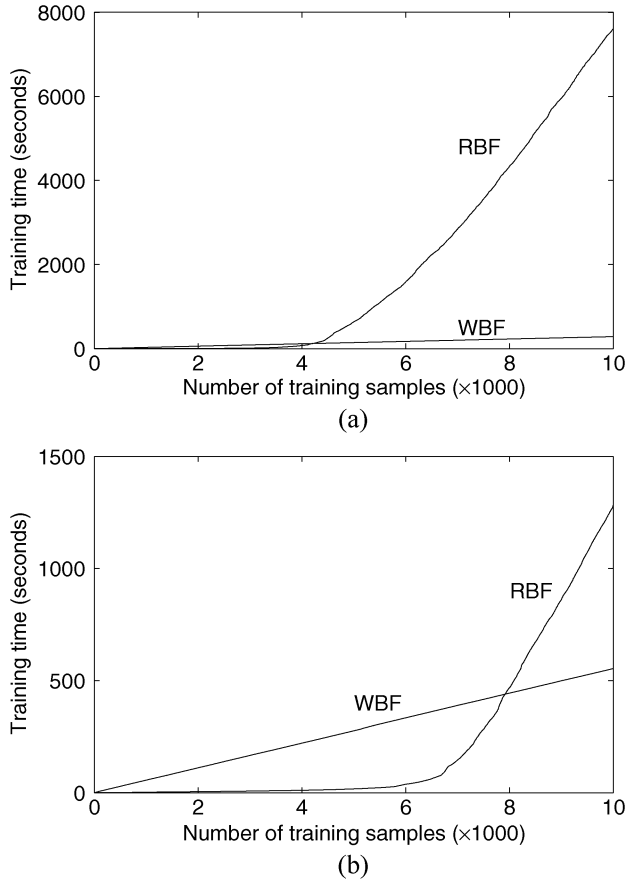

Fig. 3. Comparison of training time. (a) Training data have uniform distribution. (b) Training data have Gaussian distribution.

$K_s = 2$. Hence, both the structure of WBFNNs and the calculation of algorithm SLWBF are very simple in this case.

Fig. 1(b)–(e) shows the resulting approximation $\hat{f}(x)$ after presenting 10 000 training pairs. Fig. 1(b) is the approximation by RBFNN and the training sequence obeys uniform distribution. Fig. 1(c) is the approximation by WBFNN and the training sequence also obeys uniform distribution. In Fig. 1(d) and (e), the training data sequences obey Gaussian distribution $N(4, 1.25^2)$ and the network is an RBFNN in Fig. 1(d) and a WBFNN in Fig. 1(e). These results show that WBFNNs give better approximation than RBFNNs.

Fig. 2 gives the comparison of approximation mean square error $\int_0^{10}(f(x)-\hat{f}(x))^2 dx/10$. In Fig. 2(a), the training data obey uniform distribution, and in Fig. 2(b), the training data obey Gaussian distribution. In both cases of uniform distributed and Gaussian distributed samples, WBFNNs perform better approximation and converge faster.

Fig. 3 is the comparison of training time for RBFNNs and WBFNNs. We can see that the training time of SLWBF algorithm is linear and the time of MRAN algorithm looks like a square function. This is because, in the MRAN algorithm, extended Kalman filter is applied to justify the parameters of neurons, whose training time is proportional to the square of the number of neurons, while in the SLWBF algorithm, the time to learn from a single training data is fixed.

Fig. 4 is the comparison of the number of neurons. At the beginning, RBFNNs have less neurons than WBFNNs. However, in the end, RBFNNs have more neurons. We can see that at the beginning of the training SLWBF algorithm requires more neurons while it gives higher accuracy in the approximation and faster converge speed. In the end, RBFNNs require more neurons but still provide less accuracy in the approximation.

According to the performance comparison results, WBFNNs have better approximation accuracy. When the training data obey uniform distribution, although WBFNNs and RBFNNs give similar approximation errors after all 10 000 training samples have been applied, WBFNNs converge faster. When the training data obey normal distribution, WBFNNs give smaller errors. Hence, we know that WBFNNs have better generalization than RBFNNs. In both cases, WBFNNs can give better approximation of functions from fewer training data pairs. The time to train a WBFNN is proportional to the number of training data pairs. However, the time for training an RBFNN is nonlinear and will grow faster when the number of training data grows. For the size of the network, RBFNNs have fewer neurons at the beginning and, in the end, RBFNNs have more neurons than WBFNNs.

V. Conclusion

In this letter, a new kind of neural networks, WBFNNs, is defined. A sequential learning algorithm called SLWBF algorithm is presented for WBFNNs. WBFNNs are a development of both RBFNNs and WNNs. Both the scaling function and wavelet function of an MRA are applied to construct a WBFNN. Since wavelets have the ability to approximate functions in multiresolution, WBFNNs can give better approximation of functions with better generalization.

Performance of the algorithm SLWBF has been compared with the algorithm MRAN. The training sequences coming from both uniform and Gaussian distributions are used for numerical experiments. The results indicate that WBFNNs give more accurate approximation, faster converge, and less training time, while they require smaller size of networks. Furthermore, SLWBF gives better generalization performance, especially when the input data are not uniformly distributed.

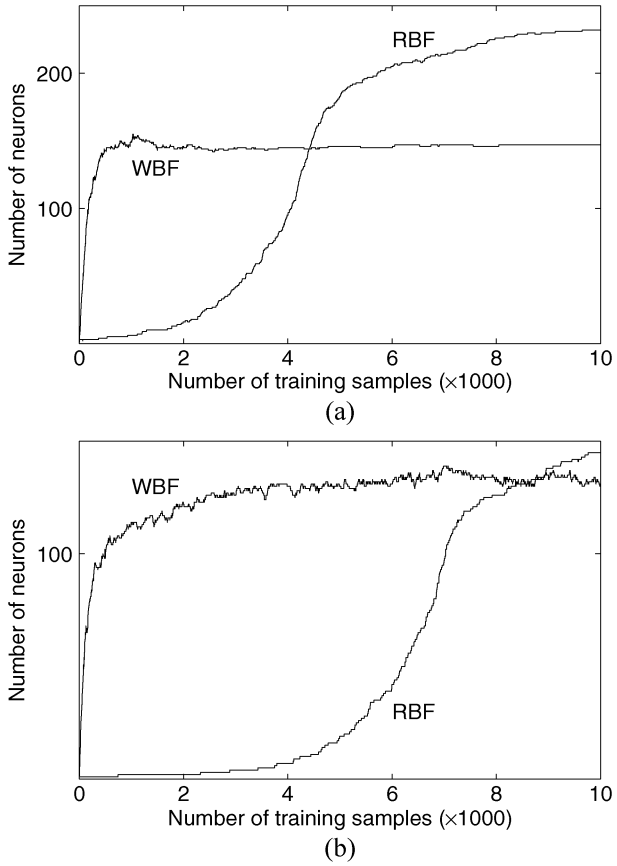

Fig. 4. Comparison of numbers of neurons. (a) Training data have uniform distribution. (b) Training data have Gaussian distribution.

REFERENCES

[1] S. A. Billings and H. Wei, "A new class of wavelet networks for nonlinear system identification," *IEEE Trans. Neural Netw.*, vol. 16, no. 4, pp. 862–874, Jul. 2005.

[2] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Syst.*, vol. 2, pp. 321–355, 1988.

[3] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 302–309, Mar. 1991.

[4] I. Daubechies, "Ten Lectures on Wavelets," in *CBMS-NSF Regional Conference Series in Applied Mathematics*. Philadelphia, PA: SIAM, 1992, vol. 61.

[5] V. Kadirkamanathan and M. Niranjan, "A function estimation approach to sequential learning with neural networks," *Neural Comput.*, vol. 5, no. 6, pp. 954–975, Nov. 1993.

[6] N. B. Karayiannis and G. W. Mi, "Growing radial basis neural networks: Merging supervised and unsupervised learning with network growth techniques," *IEEE Trans. Neural Netw.*, vol. 8, no. 6, pp. 1492–1506, Nov. 1997.

[7] H. R. Karimi, B. Moshiri, B. Lohmann, and P. J. Maralani, "Haar wavelet-based approach for optimal control of second-order linear systems in time domain," *J. Dyn. Control Syst.*, vol. 11, no. 2, pp. 237–252, Apr. 2005.

[8] Y. Lin and F. Wang, "Modular structure of fuzzy system modeling using wavelet networks," in *Proc. IEEE Netw. Sens. Control Conf.*, Tuscon, AZ, Mar. 2005, pp. 671–676.

[9] Y. Lu, N. Sundararajan, and P. Saratchandran, "A sequential learning scheme for function approximation using minimal radial basis function (RBF) neural networks," *Neural Comput.*, vol. 9, no. 2, pp. 461–478, 1997.

[10] Y. Lu, N. Sundararajan, and P. Saratchandran, "Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm," *IEEE Trans. Neural Netw.*, vol. 9, no. 2, pp. 308–318, Mar. 1998.

[11] S. Mallat, "Multiresolution approximation and wavelet orthonomal bases of L^2," *Trans. Amer. Math. Soc.*, vol. 315, pp. 69–87, 1989.

[12] J. Platt, "A resource-allocating network for function interpolation," *Neural Comput.*, vol. 3, no. 2, pp. 213–225, 1991.

[13] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, vol. 78, no. 9, pp. 1481–1497, Sep. 1990.

[14] M. J. D. Powell, "Radial basis functions for multivariable interpolation: A review," in *Algorithms Approximation*, J. C. Mason and M. G. Cox, Eds. Oxford, U.K.: Clarendon, 1987, pp. 143–167.

[15] M. J. D. Powell, "Radial basis functions approximations to polynomials," in *Proc. 12th Biennial Numer. Anal. Conf.*, Dundee, Scotland, Jun. 1987, pp. 223–241.

[16] A. Roy, S. Govil, and R. Miranda, "A neural-network learning theory and a polynomial time RBF algorithm," *IEEE Trans. Neural Netw.*, vol. 8, no. 6, pp. 1301–1313, Nov. 1997.

[17] M. Salmeróon, J. Ortega, C. G. Puntonet, A. Prieto, and I. Rojas, "SSA, SVD, QR-cp, and RBF model reduction," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2002, vol. 2415, pp. 589–594.

[18] N. Sundararajan, P. Saratchandran, and Y. Lu, *Radial Basis Function Neural Networks With Sequential Learning: MRAN and Its Applications*. Singapore: World Scientific, 1999.

[19] F. Wang and H. Kim, "Implementing adaptive fuzzy logic controllers with neural networks: A design paradigm," *J. Intell. Fuzzy Syst.*, vol. 3, no. 2, pp. 165–180, 1995.

[20] J. Zhang, G. G. Walter, Y. Miao, and W. N. W. Lee, "Wavelet neural networks for function learning," *IEEE Trans. Signal Process.*, vol. 43, no. 6, pp. 1485–1497, Jun. 1995.

[21] Q. Zhang and A. Benveniste, "Wavelet networks," *IEEE Trans. Neural Netw.*, vol. 3, no. 6, pp. 889–898, Nov. 1992.

An Improved Algebraic Criterion for Global Exponential Stability of Recurrent Neural Networks With Time-Varying Delays

Yi Shen and Jun Wang

***Abstract*—This brief paper presents an M-matrix-based algebraic criterion for the global exponential stability of a class of recurrent neural networks with decreasing time-varying delays. The criterion improves some previous criteria based on M-matrix and is easy to be verified with the connection weights of the recurrent neural networks with decreasing time-varying delays. In addition, the rate of exponential convergence can be estimated via a simple computation based on the criterion herein.**

***Index Terms*—Global exponential stability, M-matrix, recurrent neural networks, time-varying delays.**

I. INTRODUCTION

The stability of recurrent neural networks is necessary for most successful neural network applications. Since the resurgence of neural network research in 1980s, numerous studies on the stability analysis of various neural networks have been reported [1]–[4].

Manuscript received May 31, 2007; revised August 20, 2007; accepted September 24, 2007. This work was supported by the Research Grants Council of the Hong Kong Special Administrative Region, China under Project CUHK4165/03E and the Natural Science Foundation of China under Grant 60574025.

Y. Shen is with the Department of Control Science and Engineering and the Key Laboratory of Ministry of Education for Image Processing and Intelligent Control, Huazhong University of Science and Technology, Wuhan, Hubei 430074, China (e-mail: yishen64@163.com).

J. Wang is with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Shatin, New Territories, Hong Kong (e-mail: jwang@mae.cuhk.edu.hk).

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2007.911751