Robust Radial Basis Function Neural Networks

Chien-Cheng Lee, Pau-Choo Chung, Jea-Rong Tsai, and Chein-I Chang, *Senior Member, IEEE*

***Abstract*— Function approximation has been found in many applications. The radial basis function (RBF) network is one approach which has shown a great promise in this sort of problems because of its faster learning capacity. A traditional RBF network takes Gaussian functions as its basis functions and adopts the least-squares criterion as the objective function. However, it still suffers from two major problems. First, it is difficult to use Gaussian functions to approximate constant values. If a function has nearly constant values in some intervals, the RBF network will be found inefficient in approximating these values. Second, when the training patterns incur a large error, the network will interpolate these training patterns incorrectly. In order to cope with these problems, an RBF network is proposed in this paper which is based on sequences of sigmoidal functions and a robust objective function. The former replaces the Gaussian functions as the basis function of the network so that constant-valued functions can be approximated accurately by an RBF network, while the latter is used to restrain the influence of large errors. Compared with traditional RBF networks, the proposed network demonstrates the following advantages:**

1) **better capability of approximation to underlying functions;**
2) **faster learning speed;**
3) **better size of network;**
4) **high robustness to outliers.**

***Index Terms*— Function approximation, Hample's estimator, radial basis function, robust objective function.**

I. Introduction

IT is important in many scientific and engineering applications to seek a function which can describe adequately a set of input-output pairs such as system identification. One widely used method is function approximation. In general, function approximation can be accomplished by either parametric or nonparametric methods. A parametric method assumes that the relationship between input and output patterns can be represented by a given functional model with specific parameters so that the approximation problem can be simplified to finding these parameters. In contrast, a nonparametric method does not assume *a priori* knowledge for the set of input-output pairs, even though in real world, these input-output pairs generally exhibit a highly nonlinear relationship.

Recently feedforward neural networks have been proposed as new tools for the above problems. The main reason is that a neural network can be regarded as a universal approximator [1] and possesses self-learning capability. The complex mapping in input-output pair may be constructed and learned by examples. This reduces the complexity of model selection.

There exist many types of feedforward neural networks in the literature, for example, multilayer perceptron (MLP), radial basis function (RBF) network [2] etc. Among them is the RBF network which is considered as a good candidate for approximation problems because of its faster learning capability compared with other feedforward networks. In traditional RBF networks, the Gaussian function and the least squares (LS) criterion are selected as the activation function of network and the objective function, respectively. A network adjusts iteratively parameters of each node by minimizing the LS criterion according to gradient descent algorithm. Since a neural network can accomplish a highly nonlinear mapping from input space to output space, the approximate curve generated by the network may be able to interpolate all training patterns. Nevertheless, there still exist some problems in this approach. First, if an underlying curve representing training patterns is nearly constant in a particular interval, it is difficult to utilize a Gaussian function to approximate this constant valued function unless its bandwidth (i.e., variance) is very large approaching to infinite. In this case, an RBF network will be found inefficient in approximating constant valued functions. Second, when some of the training patterns encounter large errors resulting from the presence of outliers, the network will yield inadequate responses in the neighborhood of outliers due to the LS criterion. In order to take care of these two problems, a new activation function and objective function are presented in this paper.

The new proposed activation function is a composite of a set of sigmoidal functions [3] and aims at approximating a given function with nearly constant values. Because of that, the new activation function will work better in approximating constant valued functions. The objective function is derived from robust statistics [4] in order to reduce the influence of outliers in training patterns. It has a type of Hample's M-estimator with robust property against large errors of training patterns. Nonetheless, they are different in certain aspects. First, our objective function is a single smooth function, but Hample's M-estimator is not. Second, the shape of Hample's M-estimator cannot be changed, but ours is adjustable during training phases. Advantages of adjustable shape and some important properties in robust objective function were discussed in [4]. By means of this new activation function along with the robust objective function, the RBF networks can improve approximation to constant valued functions and outliers.

In some case, the proposed RBF networks may diverge because the number of nodes used in the network is not properly chosen. Of course, the optimal size of network for a

Manuscript received July 5, 1997; revised September 18, 1999. This work was supported by the National Science Council, Taiwan, R.O.C., under Grants NSC 84-2213-E-006-025 and NSC 84-2213-E-006-086. This paper was recommended by Associate Editor P. Borne.

J.-R. Tsai, P.-C. Chung, and C.-C. Lee are with the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, 70101 R.O.C.

C.-I. Chang is with the Department of Electrical Engineering, University of Maryland—Baltimore County, Baltimore, MD 21228-5398 USA.

Publisher Item Identifier S 1083-4419(99)09703-4.

1083–4419/99$10.00 © 1999 IEEE

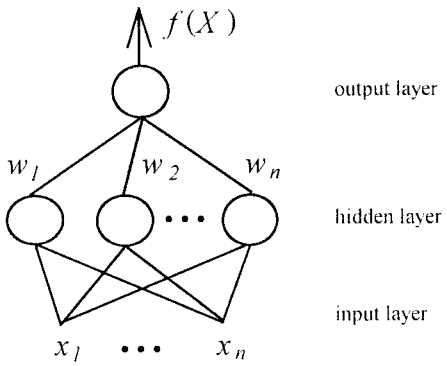


Fig. 1. Basic architecture of RBF neural network.

given problem is usually unknown. In [5], [6] adaptive growing techniques were developed in RBF network to determine an appropriate number of nodes. However, these techniques generally assume that training patterns do not have outliers. When training patterns contain outliers, the number of nodes determined by traditional growing techniques can only grow to a certain number beyond which a desired number cannot be reached. So traditional techniques will fail if outliers occur. In order to avoid influence of outliers, a memory queue is incorporated into traditional growing techniques. By so doing, the network can produce a proper size for a given problem.

This paper is organized as follows. Section II reviews RBF networks and discusses some related problems. Section III constructs the new activation function of RBF networks based on a set of sigmoidal functions. Section IV presents the idea to obtain the robust objective function. Section V discusses the learning algorithm of the network. The experimental results are conducted in Section VI. Finally, conclusions are included in Section VII.

II. RBF Networks and Function Approximation

A. Basic Architecture of Radial Basis Function Networks

The basic architecture of an RBF network with n inputs and a single output is shown in Fig. 1. The RBF network is a two-layer network. The nodes in adjacent layers are fully connected. Such a network can be represented by the following parametric model

$$f(X) = \sum_{i=1}^{r} w_i \phi_i(||X - \mu_i||, \theta_i) \tag{1}$$

where $X \in R^n$ is an input vector, ϕ_i is the basis function of the network from R^n to R, w_i's are weights of network, $\mu_i = (\mu_{i1}, \mu_{i2}, \cdots, \mu_{in})^T$ is called the center vector of the ith node, $\theta_i = (\theta_{i1}, \theta_{i2}, \cdots, \theta_{in})^T$ is called the bandwidth vector of the ith node, and $||\ ||$ denotes the Euclidean norm. If the basis function of the network is a Gaussian function, then $\phi_i(||X - \mu_i||, \theta_i) = \exp\{[-||X - \mu_i||/(\theta_{i1}, \theta_{i2}, \cdots, \theta_{in})]^2\}$.

B. Some Problems in RBF Networks

In most of RBF networks, Gaussian functions and the LS criterion are widely used as basis function and criterion for optimality, respectively. However, this type of RBF networks suffers from some problems. One problem which has not been addressed yet by other researchers is to approximate a function with nearly constant values or constant values in some intervals under the considered domain. This problem is analogous to Fourier expansion of a periodic constant function where the neighborhood of endpoints in each period still produces ripples even if a great number of terms are used. To eliminate this phenomenon, more terms must be added to the expansion. However, this will increase computational complexity. Fig. 2(a) shows the result of approximating a piecewise constant function. The network uses sixty training patterns and twenty nodes. The solid curve indicates the underlying curve and the dashed curve is the approximate curve generated by an RBF network with a Gaussian activation function. In this example, the training procedure is terminated when learning cycles reach 3000. From Fig. 2(a), it can be seen that the underlying curve still cannot be well represented by the approximate curve constructed by the neural network. As expected, the network produces ripples in the endpoints of the each line segment.

Another problem occurs when some of the training patterns incur large errors due to the presence of outliers. These errors will cause some training patterns moving far away from the underlying position. As a result, approximation will not be good since all training patterns must be interpolated. Fig. 2(b) shows a poor approximation result when training patterns contain outliers. These large errors pull approximate curve toward outliers because of inaccurate responses for networks in the neighborhood of outliers.

III. Radial Basis Function Based on Composite of Sigmoidal Functions

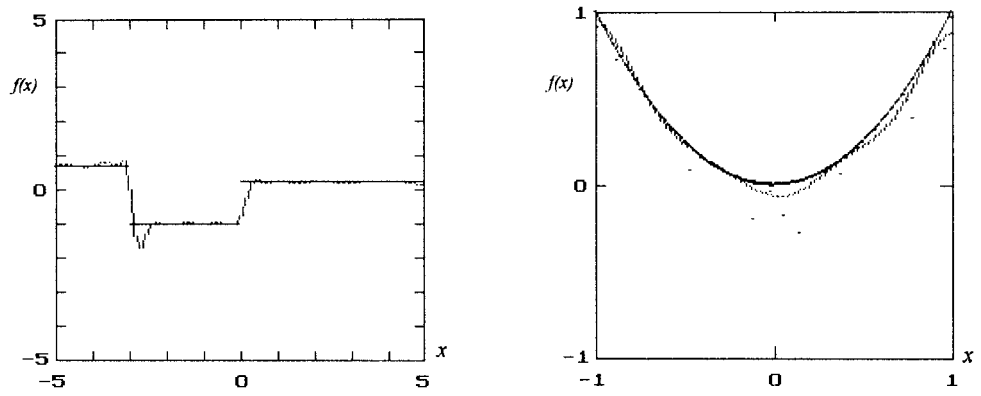
As stated previously, the RBF networks using Gaussian activation function cannot effectively approximate a constant valued function. To deal with this problem a composite of a set of sigmoidal functions is proposed to replace Gaussian functions.

Let us first consider a one-dimensional case. Let $f(x) = (1 + e^{-bx})^{-1}$ which is a sigmoidal function with $b > 0$. To obtain an activation function for RBF networks, we combine two sigmoidal functions as follows:

$$g(x) = \frac{1}{1 + e^{-\beta[(x-\mu)+\theta]}} - \frac{1}{1 + e^{-\beta[(x-\mu)-\theta]}} \tag{2}$$

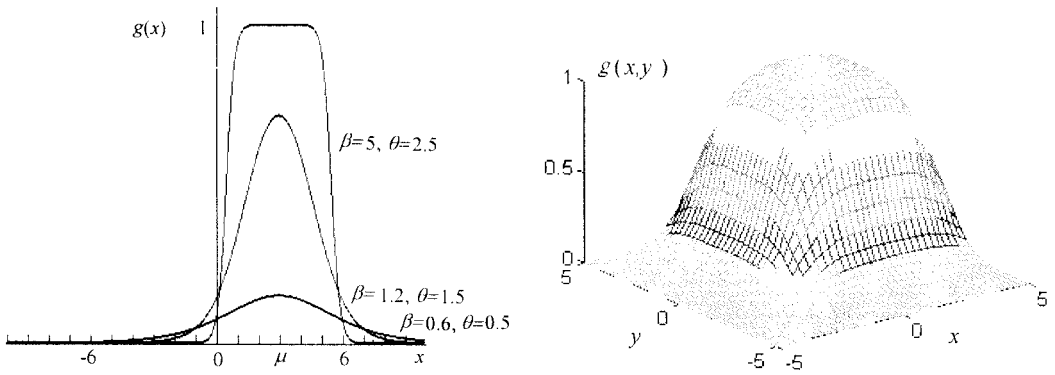
where $q > 0$. Three cases all with the same m are plotted in Fig. 3(a). From Fig. 3(a), an observation can be made: the shape of $g(x)$ is approximately rectangular if b or q is very large. This implies that the $g(x)$ should be a good candidate used to approximate one-dimensional constant functions. Moreover, as shown in Fig. 3(a), $g(x)$ has a unique maximum at μ, radial symmetry, and local support property which meet fundamental properties of radial basis functions used in neural networks. So we may use $g(x)$ as activation function of RBF network.

To extend to higher dimensions, the proposed RBF function can be defined based on a composite of sigmoidal functions



(a) (b)

Fig. 2. The Gaussian function is used as activation function of network to approximate a function with (a) content value in some intervals under the considered domain. The result shows that neighborhood of endpoints in each line segment produce ripple. On the other hand, (b) the LS criterion attempts to interpolate all training pattern generated by x^2, and cause a inaccurate response in some position of approximated curve.



(a) (b)

Fig. 3. (a) Three one-dimensional sigmoidal RBF's with the same center = 3. (b) A two-dimensional sigmoidal RBF with center vector = [1,1], shift vector = [1,1], and shape parameter = 5.

as follows:

$$g(X)=\prod_{j=1}^{n}\left(\frac{1}{1+e^{-\beta_j[(x_j-\mu_j)+\theta_j]}}-\frac{1}{1+e^{-\beta_j[(x_j-\mu_j)-\theta_j]}}\right) \tag{3}$$

where $X=[x_1,x_2,\cdots,x_n]^T$ is an input vector, $\mu=(\mu_1,\mu_2,\cdots,\mu_n)^T$ is the center vector, $\theta=(\theta_1,\theta_2,\cdots,\theta_n)^T$ is the bandwidth vector, $\theta_i>0$ and $\beta=(\beta_1,\beta_2,\cdots,\beta_n)^T$ is the shape vector of $f(X)$. A two-dimensional case of (3) is plotted in Fig. 3(b). Since (3) is generated from a composite of a set of sigmoidal functions, (3) can be viewed as a sigmoidal radial basis function (SRBF). Using the SRBF as the activation function of an RBF network, the network, to be called SRBF network yields the following parametric form:

$$f(X)=\sum_{i=1}^{r} w_i g_i(X) \tag{4}$$

where

$$g_i(X)=\prod_{j=1}^{n}\left(\frac{1}{1+e^{-\beta_j[(x_j-\mu_j)+\theta_j]}}-\frac{1}{1+e^{-\beta_j[(x_j-\mu_j)-\theta_j]}}\right)$$

is the ith composite of sigmoidal functions given by (3), $X\in R^n$ is an input vector, $\beta_i=(\beta_{i1},\beta_{i2},\cdots,\beta_{in})^T$ is referred to as the shape vector of the ith node, μ_{ij} is the jth component of the center vector in the ith node and $\theta_i=(\theta_{i1},\theta_{i2},\cdots,\theta_{in})^T$ is the bandwidth vector with in the ith node.

IV. Robust Objective Function for Function Approximation

A. Influence Function

As mentioned previously, when some of the training patterns are outliers, the RBF network using the LS criterion will not approximate the underlying curve quite well. The reasons are as follows. Consider a network with a single output node $f(X)$. (Note that the same idea may be applied to a network with multiple-output node.) Assume that θ is the parameter

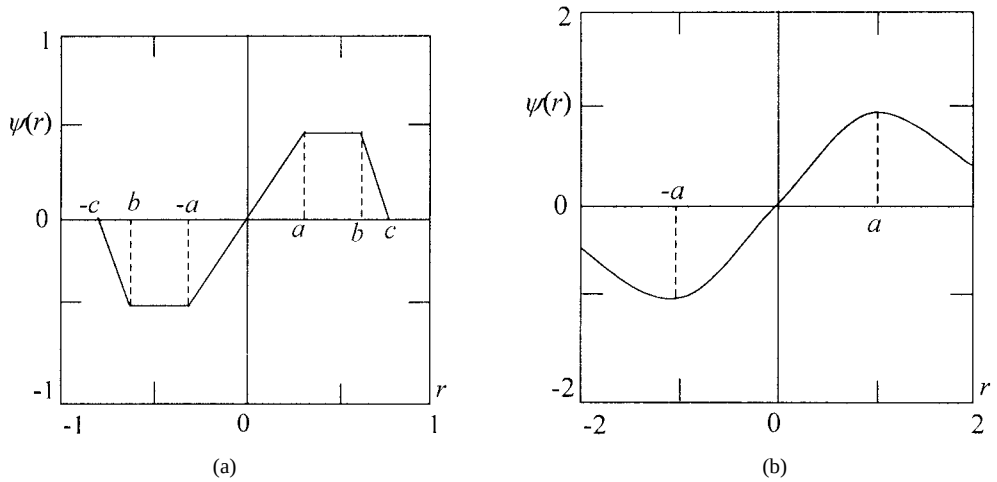


Fig. 4. (a) Shape of Hample M-estimator and (b) its similar shape.

set of the network whose parameters are adjusted at each time step by minimizing a given function E, i.e.,

$$\theta_{k+1} = \theta_k - \eta \sum_{p=1}^{P} \frac{\partial E(r_p)}{\partial \theta_k} \tag{5}$$

where $r_p = t_p - f(x_p)$ is the residual for the pth training pattern with desired value t_p, η is a step size parameter, and $E(r_p)$ often is referred to as the objective function of the network. The objective function generally requires even-symmetric property, $E(0) = 0$ and continuity. The gradient $\partial E(r_p)/\partial \theta_k$ in (5) can be obtained as

$$\begin{aligned} \sum_{p=1}^{P} \frac{\partial E(r_p)}{\partial \theta_k} &= \sum_{p=1}^{P} \frac{\partial E(r_p)}{\partial r_p} \frac{\partial r_p}{\partial \theta_k} \\ &= \sum_{p=1}^{P} \psi(r_p) \frac{\partial f(X_p)}{\partial \theta_k} \end{aligned} \tag{6}$$

where $\psi(r_p) = \partial E(r_p)/\partial r_p$ is called influence function [7].

In order that the performance of a network be accepted, the difference between output of a network and desired output should approach zero for all training patterns, i.e., all $r_p \cong 0$. Referring to (5), the criterion of terminating training is $\Sigma_p \, (\partial E(r_p)/\partial \theta_k) \cong 0$, i.e., the value of parameters of network tend to nearly the same. In the LS criterion, $\psi(r_p)$ is equal to r_p. If outliers disappear, all training patterns will be positioned accurately with their residuals close to zero, i.e., $r_p \cong 0$. However, when outliers are present, their position will be far away from their underlying position so that each residual of outliers becomes very large and $\Sigma_p[\partial E(r_p)/\partial \theta_k]$ is far above zero. According to (5), this network will keep adjusting parameters. As a consequence, the underlying curve cannot be approximated by minimizing the LS criterion.

In order to alleviate the outlier problem, M-estimators are used as the objective function of the networks. An M-estimator is of the following form:

$$\min_{\theta} \sum_{p=1}^{P} \rho(r_p) \tag{7}$$

where ρ is a function and $r_1, r_2, \cdots, r_p$ are samples. In neural network applications, r_p and r can be treated as the residual and objective function of network, respectively. Suppose that $\psi(r)$ is the derivative of $\rho(r)$. Then obtaining the solution to (7) is equivalent to solving the following equation:

$$\sum_{p=1}^{P} \psi(r_p) = 0, \tag{8}$$

where $\psi(r) = d\rho(r)/dr$ is called the influence function. There are many M-estimators which may be used as robust objective functions. However, Hample's M-estimator is found to best fit applications of neural networks and has the following parametric form [Fig. 4(a)]:

$$\psi(r) = \begin{cases} r, & \text{if } a < |r| \\ a \operatorname{sgn}(r), & \text{if } a \le |r| \le b \\ \{(c - |r|)/(c - |b|)\} \, a \operatorname{sgn}(r), & \text{if } b \le |r| \le c \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where a, b and c are constants with $a < b < c$.

B. Construction of Robust Objective Functions

In theory, Hample's M-estimator is a good candidate to be used as a base to construct a robust objective function. As a matter of fact, if a function has similar shape [Fig. 4(b)] to that of Hample's M-estimator, it can be also chosen as a robust objective function for a network. As such, we can construct a class of functions which generate the similar shapes to Fig. 4(b) so that the selection of a robust objective function is not necessary to limit to Hample's M-estimators.

The next step is how to obtain such a class of functions. Observing Fig. 4(b), the functions in the class should possess the following properties:

1) they pass through the origin;
2) they have a unique maximizing point a for $r > 0$;
3) they have a unique minimizing point $-a$ for $r < 0$.

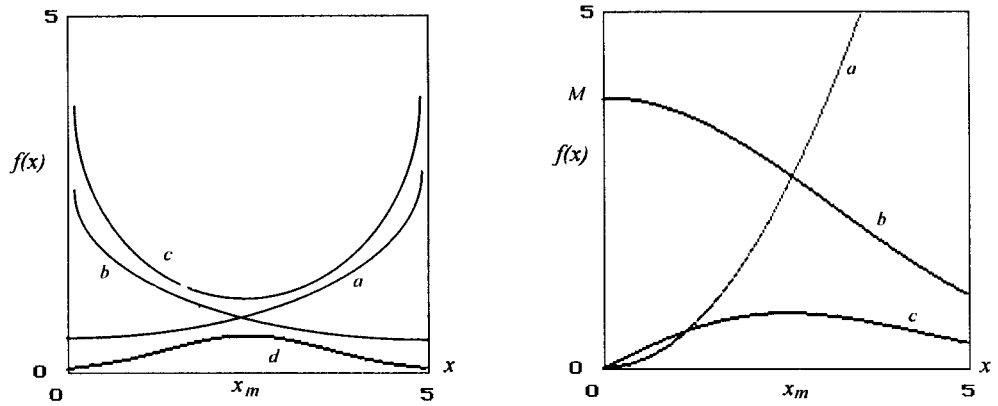


Fig. 5. (a) The two tradeoff functions and (b) their product. Utilizing the product of $f_a(x)$ and $f_b(x)$, we may obtain $f_c(x)$ which has the similar shape of Hample M-estimator in $r > 0$.

Notice that the interval with these two extreme points as endpoints can be considered to be the confidence interval of the residual. From Fig. 5(a), it includes some special functions which can be found in many applications such as data analysis, inventory control etc. The $f_a(x)$ and $f_b(x)$ are complements each other and $f_a(x) > 0$ and $f_b(x) > 0$ $\forall x$. The former is strictly increasing and the latter is strictly decreasing. Their product generates a new function $f_c(x)$. If the rate of the increase of $f_a(x)$ is larger than the rate of decrease of $f_b(x)$ as $x \to \infty$, i.e.

$$\lim_{x \to \pm\infty} f_a(x)/[1/f_b(x)] = \infty \tag{10}$$

$f_c(x)$ can be obtained as an asymmetric U shape with a distinct minimum. Without loss of generality, let the minimum be the intersection of $f_a(x)$ and $f_b(x)$.

On the contrary, if (10) approaches zero instead of infinity, i.e.,

$$\lim_{x \to \pm\infty} f_a(x)/[1/f_b(x)] = 0 \tag{11}$$

a function $f_d(x)$ with a distinct maximum will be also obtained. Since $f_d(x)$ has only a maximizing point, it is of interest. Let x_m denote this point. Examining $f_d(x)$ carefully, we find that its curve looks similar to the part of $r \geq 0$ in Fig. 4(b). The interval $[0, x_m]$ can be used as the confidence interval of the residual corresponding to the interval $[0, a]$ in Hample's M-estimator. However, we need $f_d(0) = 0$. To satisfy this condition, we must modify $f_a(x)$ and $f_b(x)$ in Fig. 5(a). Assume that only the case of $x \geq 0$ is considered. The $f_a(x)$ still maintains a strictly increasing function in $[0, \infty)$ with $f_a(0) = 0$. The $f_b(x)$ also remains a strictly decreasing function in $[0, \infty)$ but has $f(0) = M$ which is a maximum. Their product $f_c(x)$ is plotted in Fig. 5(b). Similarly, a composite function can be also obtained for $r \leq 0$ with its shape similar to that of Hample's M-estimator.

Having the above discussions, we are ready to define a class of robust objective functions for the SRBF network with the form given as follows:

$$E_R(r_p) = \sum_{p=1}^{P} [\phi(r_p) - \phi(0)] \tag{12}$$

where $\phi(r_p)$ is a continuous function, $\phi(0)$ is a constant, and P is the total number of inputs. Note that $E_R(r_p)$ becomes the LS criterion when $\phi(r_p) = r^2$ and $\phi(0) = 0$. The derivative function $\psi(r) = d\phi(r)/dr$ will be written as

$$\psi(r) = s(r)t(r) \tag{13}$$

which has the following properties:

(F1) $s(r)$ is an odd-symmetric, monotone and continuous function with $s(0) = 0$;

(F2) $t(r)$ is an even-symmetric and continuous function which satisfies;

(F2-1) $t(r)$ has a unique maximum at $r = 0$;

(F2-2) $0 < t(r) \leq M$, M is a real number;

(F2-3) $t(r)$ increases strictly for $-\infty < r \leq 0$ and decrease strictly for $0 \leq r < \infty$;

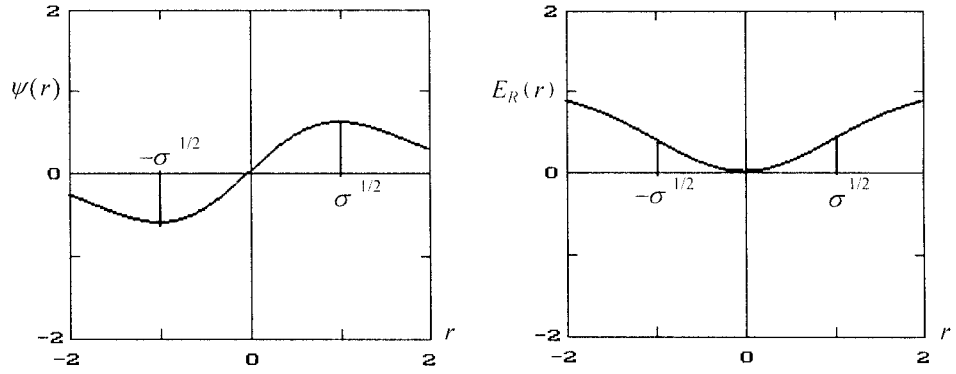
(F3) $\lim_{r \to \pm\infty} s(r)/g(r) = 0$, where $g(x) = 1/t(x)$.

From the above properties, $\psi(r)$ is a function with $r = 0$ as its symmetry center and two extreme points, one in $r > 0$ and another in $r < 0$. Assume that a and $-a$ are the points rendering the two extrema of $\psi(r)$, then $[-a, a]$ is the confidence interval of the residual for $\psi(r)$. Moreover, in order to improve the efficiency of network, an additional property is imposed on $y(r)$ and is stated as follows.

Adjustable Property for Objective Function: $y(r)$ should have an adjustable parameter σ which can be used to adjust its shape when there is a need; in other words, positions of a and $-a$ in $\psi(r)$ should be flexible subject to change during a training procedure.

Basically, *a priori* knowledge gives a reasonable initial guess for confidence interval of residual in a given problem from which the endpoints of the confidence interval, or positions of cutoff points will be adjusted based on training. The update rules for the cutoff points are solutions to the equation of setting the differential of $\psi(r)$ to zero. Because $\psi(r)$ is made up of $s(r)$ and $t(r)$, it is sufficient to require one of these two functions to be adjustable parameters, say, $t(r)$.

The adjustment of the endpoints of the confidence interval can be done with either of the following two methods. Since it is expected that the confidence interval should shrink as a training or learning procedure carries on, the first method is



(a) (b)

Fig. 6. (a) $\psi(r)$ in Example 4.1 and (b) the corresponding robust objective function $ER(r)$.

to use a decreasing function with a proper decreasing rate κ for confidence interval of residuals where how to choose κ is the key issue.

The second method of reducing the confidence interval of residuals is based on the rate of change in the objective function. When the objective function of a network changes drastically between two time steps in approximation, this implies that the approximate curve is pulled more closer to the underlying curve in the current time step than in the previous time step. Hence, the confidence interval of the residuals is reduced. In general, the number of outliers is very small compared to the number of training patterns. The average of the residuals of all training patterns should be capable of representing the residual distribution. Based on this assumption, the average of all residuals represent the error incurred by approximation. This is very similar to noise considered in communications and signal processing applications. What we do to reduction of the confidence interval of the residuals is the same as what we design a low-pass filter to reduction of noise effect.

The average of all residuals can be estimated by

$$r_{\text{ave}} = \Sigma_p \, [t_p - f(X_p)]/P. \tag{14}$$

Here, P is the total number of training patterns. Let c be a constant. $\pm cr_{\text{ave}}$ can be considered as two cutoff points of the influence function and $[-cr_{\text{ave}}, cr_{\text{ave}}]$ is confidence interval of the residual. Generally speaking, cr_{ave} is smaller than the largest residual of network. A simple method is to set $c = 1$. As soon as cutoff points are determined, the adjustable parameter σ in the influence function can be calculated immediately.

Example 4.1: (construction of a robust objective function)

Following the three properties mentioned in Section IV-B, we define $s(r) = r$ and $t(r) = e^{r^2/2\sigma}$. Then $\psi(r)$ can be expressed as

$$\psi(r) = s(r)t(r) = re^{-r^2/2\sigma}. \tag{15}$$

The derivative of $\psi(r)$ is calculated as

$$\begin{aligned} d\psi(r)/dr &= e^{-r^2/2\sigma} - (r^2/\sigma)e^{-r^2/2\sigma} \\ &= (1 - r^2/\sigma)e^{-r^2/2\sigma}. \end{aligned} \tag{16}$$

Setting $d\psi(r)/dr = 0$ yields two extreme points of $\psi(r), \pm\sigma^{1/2}$, one in $r > 0$, and the other in $r < 0$ which can be chosen to be two cutoff points of $\psi(r)$. The confidence interval of the residual is $[-\sigma^{1/2}, \sigma^{1/2}]$. Also

$$\phi(r) = \int re^{-r^2/2\sigma} \, dr = -\sigma e^{-r^2/2\sigma}. \tag{17}$$

The corresponding objective function is therefore obtained as

$$E_R(r) = \phi(r) - \phi(0) = \sigma(1 - e^{-r^2/2\sigma}). \tag{18}$$

The σ here can be adjusted adaptively during a training procedure to improve the accuracy of approximation. Fig. 6 shows the shape of $\psi(r) = re^{-r^2/2}$ with $\sigma = 1$ and its corresponding robust objective function $1 - e^{-r^2/2}$.

V. Learning of Network Parameters

The algorithm described in this section is based on the gradient decent method and an adaptive growing technique for networks. While the former provides update rules for parameters, the latter suggests a method to grow the network until its size reaches the optimum.

A. Parameters Update Rules

To determine all parameters of the network, the robust objective function described in (12) must be minimized. For convenience, the following notations are used.

$$\phi'(x) = d\phi(x)/dx, \quad X_p = (x_1^p, x_2^p, \cdots, x_n^p), \tag{19}$$

$$R_i(X_p) = \prod_{j=1}^{n} (h_i(lx_{ij}^p) - h_i(rx_{ij}^p)), \tag{20}$$

$$\begin{aligned} h_i(lx_{ij}^p) &= 1/[1 + \exp(-\beta_{ij}(lx_{ij}^p))], \\ h_i(rx_{ij}^p) &= 1/[1 + \exp(-\beta_{ij}(rx_{ij}^p))], \end{aligned} \tag{21}$$

$$\begin{aligned} lx_{ij}^p &= (x_j^p - \mu_{ij}^p) - \theta_{ij}^p, \quad \text{and} \\ rx_{ij}^p &= (x_j^p - \mu_{ij}^p) + \theta_{ij}^p. \end{aligned} \tag{22}$$

Using the gradient decent method, the equations for updating network parameters can be obtained as follows. Their derivations are referred to the Appendix. Let w_i be the weight between the output and the ith SRBF node, β_i be the steepness parameter in the ith SRBF node, μ_{ij} the jth component of the

mean vector in the ith SRBF node and θ_{ij} the jth component of the shift vector in the ith SRBF node.

$$\frac{\partial E_R(r_p)}{\partial w_i} = \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p} \frac{\partial r_p}{\partial w_i} = -\sum_{p=1}^{p} \phi'(r_p) R_i(X_p). \tag{23}$$

See (24)–(26) at the bottom of the page.

Including all the parameters in a vector W, the learning rule can be rewritten as

$$W(t+1) = W(t) - \eta \frac{\partial E_R(r_p)}{\partial W} \tag{27}$$

where t is the time step and η is the learning rate of network.

B. Adaptive Growing Technique of Network

Ideally, an RBF network with a robust objective function should be capable of approximating accurately a given function. However, this can be only accomplished when a proper size of the network is used. But, in most cases, we do not know what size is adequate for a given problem. In order to overcome this problem, an adaptive growing technique is suggested, which dynamically adjusts the number of nodes based on the following rules.

1) When the objective function value is larger than the termination criterion or/and the network is not converging, a node will be added. The center of the newly added node is placed on the position of the training pattern with the largest residual and other parameters may be initialized randomly.
2) When the outgoing weight of a node is smaller than a prescribed threshold after a certain number of learning iterations, this node will be deleted because it would not provide significant contribution to the network.

Although this method had been proved to be successful in seeking a proper size of network for a given problem, its success depends on the quality of training patterns. When there are no outliers, the adaptive growing technique using the above two rules work very well. But, if training patterns contain outliers, such rules may suffer from some problems. We first consider this case of initial nodes. If an RBF network has too many initial nodes, the outliers will be interpolated by some of these initial nodes in early stage of training. As a result, using the LS criterion cannot improve the accuracy of approximation .

As mentioned above, a network should not use too many initial nodes. An alternative approach is to start out with a small size of a network. As the number of nodes increases gradually, the approximate curve will be more accurate to represent the underlying function so as to distinguish outliers and normal training patterns. If a training pattern has a large residual, it will be regarded as an outlier. However, according to rule 1) of adaptive growing technique, the network will select a training pattern with the largest residual as the center of newly added node and place it in the position of the outlier. As a result, the residual in this position will be beyond confidence interval of residual. However, as indicated in Section IV, the confidence interval of the residual in a robust objective function must be smaller than the residual of outlier. Consequently, the parameter of this newly added node will not be updated. Thus, the node will be deleted according to rule 2) because it has not made contribution to the network. However, this node will be eventually selected again even it was deleted before. As a result, the algorithm will repeat a whole cycle again and never ends.

In order to avoid the above problem, a memory queue is introduced to record positions of nodes added to the network during the training. Before a new node is added to network according to rule 1), the memory queue is checked. If the position of the node to be added has not been recorded in

$$\frac{\partial E_R(r_p)}{\partial \beta_{ij}} = \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p} \frac{\partial r_p}{\partial \beta_{ij}} = \sum_{p=1}^{P} \phi'(r_p) w_i R_i(X_p) \frac{(lx_{ij}^p) h_i(lx_{ij}^p)[1 - h_i(lx_{ij}^p)] - (rx_{ij}^p) h_i(rx_{ij}^p)[1 - h_i(rx_{ij}^p)]}{h_i(lx_{ij}^p) - h_i(rx_{ij}^p)} \tag{24}$$

$$\frac{\partial E_R(r_p)}{\partial \mu_{ij}} = \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p} \frac{\partial r_p}{\partial \mu_{ij}} = \sum_{p=1}^{P} \phi'(r_p) w_i R_i(X_p) \frac{(-\beta_i)\{h_i(lx_{ij}^p)[1 - h_i(lx_{ij}^p)] - h_i(rx_{ij}^p)[1 - h_i(rx_{ij}^p)]\}}{h_i(lx_{ij}^p) - h_i(rx_{ij}^p)} \tag{25}$$

$$\frac{\partial E_R(r_p)}{\partial \theta_{ij}} = \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p} \frac{\partial r_p}{\partial \theta_{ij}} = \sum_{p=1}^{P} \phi'(r_p) w_i R_i(X_p) \frac{(-\beta_i)\{h_i(lx_{ij}^p)[1 - h_i(lx_{ij}^p)] + h_i(rx_{ij}^p)[1 - h_i(rx_{ij}^p)]\}}{h_i(lx_{ij}^p) - h_i(rx_{ij}^p)} \tag{26}$$

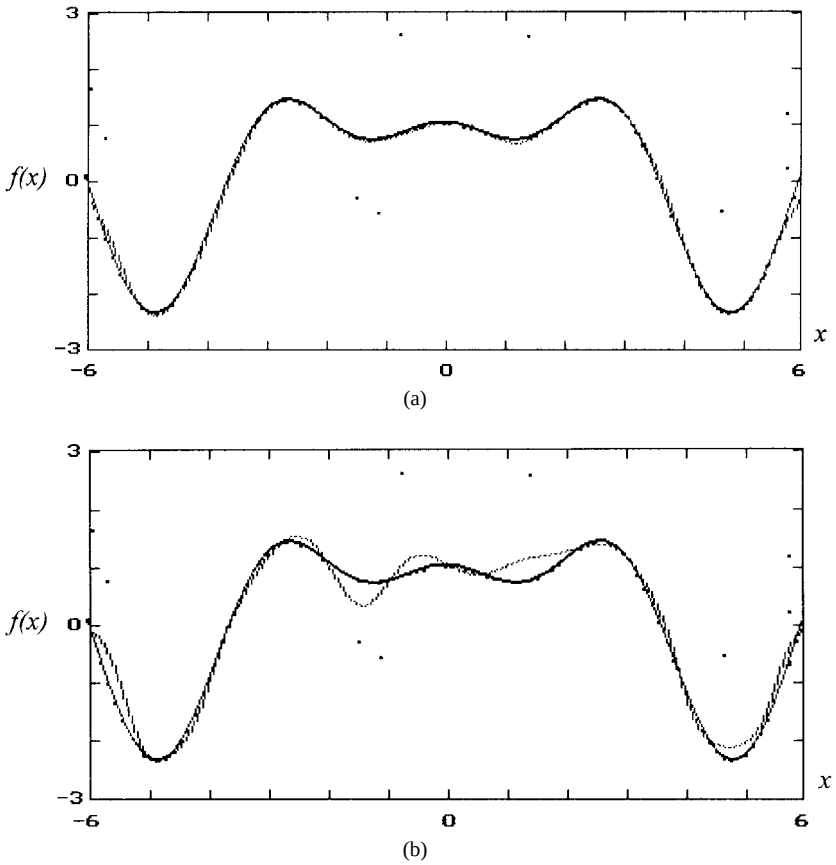


Fig. 7. Approximation of a given function $f(x) = 0.5x\ \sin(x) + \cos^2(x)$ by two SRBF network with ten nodes after 1000 learning iterations. (a) When robust objective function is used and (b) when LS criterion is used.

the memory queue, this node is then added to the network. Otherwise, a pattern with the second larger residual is sought for a new node.

C. Algorithm Implementation

In this section, the algorithm of implementing the SRBF network with a robust objective function and proposed adaptive growing technique is presented. First of all, we introduce the following useful notations which will be used in the description of the algorithm.

1) Check period T: if the number of iterations between consecutive updates of the objective function is a multiple of the period, we should check the state of the network.
2) Objective function $E_R(i)$: the value of the objective function in the ith iteration.
3) Threshold $\delta_{\text{en\_upper}}$: if the difference between the current and previous values of the objective function is larger than the threshold, it implies that the SRBF network moves more closer to underlying function; hence the confidence interval of the residual is reduced.
4) Threshold $\delta_{\text{en\_lower}}$: if the difference between the current and previous values of objective function is smaller than the threshold, it means that the number of nodes of network is insufficient in approximating the underlying function; hence a new node needs to be added to the network
5) Threshold δ_{weight}: Nodes with weights smaller than this threshold will be deleted.
6) Threshold δ_{stop}: criterion of terminating the learning process.

With all necessary notations defined, we are ready to describe the algorithm as follows.

Step 1: Set up the network initial conditions.

Step 1-1. Select an initial number of nodes for the network.

Step 1-2. Set initial parameter values of each node.

Step 1-3. Set $E_S = \epsilon$ to a small value, where E_S is used for recording the value of the objective function.

Step 1-4. Set $i = 1$ and initialize the check period T.

Step 2: Construct the robust objective function.

Step 2-1. Select proper $s(r)$ and $t(r)$.

Step 2-2. Compute extrema of $y(r) = s(r)t(r)$ by solving the equation of $y'(r) = 0$.

The extrema positions represent the two cutoff points. Let this two cutoff points be $\pm s$.

Step 2-3. Let $[-s, s]$ be the initial confidence interval of the residual.

Step 2-4. Compute the corresponding robust objective function $E_R(r_p)$ by integrating $y(r)$

Step 3: Compute output of the network for all training patterns.

Step 4: Compute the value of robust objective function of the network.

Step 5: If the iteration number of i is a multiple of T,

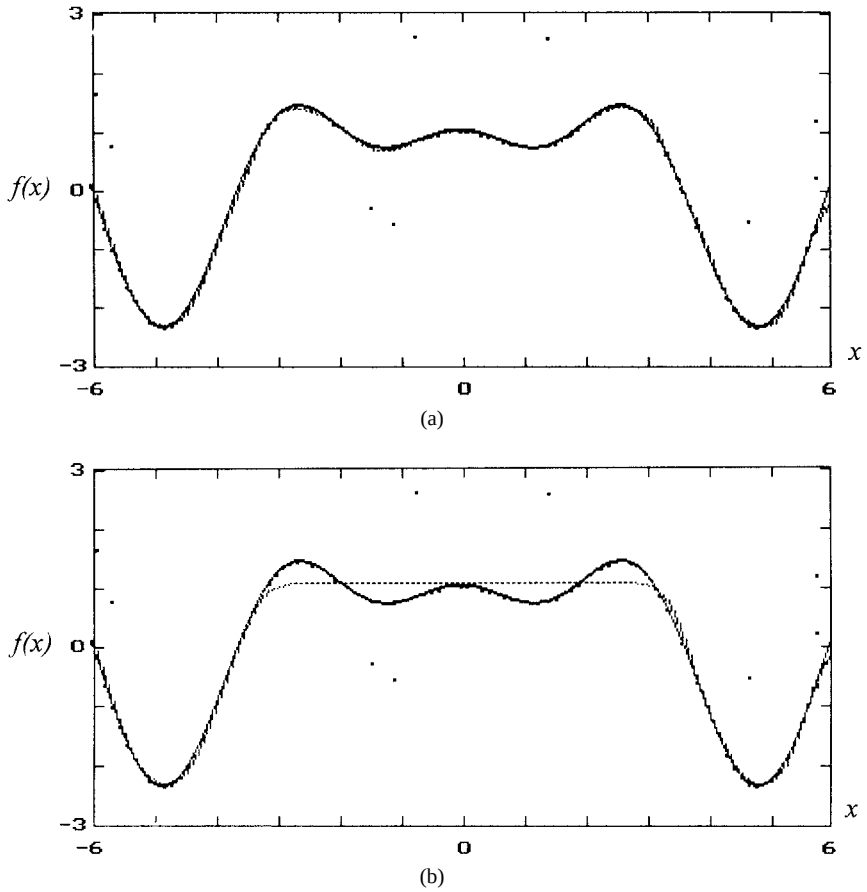


Fig. 8. Approximating a given function $f(x) = 0.5x\ \sin(x) + \cos^2(x)$ by adaptive growing technique. (a) When a memory queue is used and (b) without memory queue. (a) SRBF network, adaptive growing technique with memory queue robust objective function, four nodes, 3300 learning cycle. (b) Gaussian network, adaptive growing technique with memory queue, robust objective function, five nodes, 3300 learning cycle.

adjust the size of the network and the confidence interval of the residuals based on the following procedure,

Step 5-1. If $|E_R(i) - E_S| > \delta_{\text{en\_upper}}$ (objective function changes rapidly), then reduce the confidence interval of the residual by finding new cutoff points

$$\sigma_{new} = \begin{cases} \sigma_{old} * r_d & \text{if } \sigma > \sigma_L \\ \sigma_{old} & \text{otherwise,} \end{cases} \tag{28}$$

where r_d is the decreasing rate and σ_{L} is the lower bound of the confidence interval of residual.

Step 5-2. If $|E_R(i) - E_S| < \delta_{\text{en\_lower}}$ (objective function does not change rapidly), then add a new node to the network by the adaptive growing technique with a memory queue.

Step 5-3. Check the outgoing weight of each node and remove those nodes with outgoing weights smaller than prescribed threshold (σ_{weight}).

Step 5-4. Store the current objective function $E_R(i)$ to E_S.

Step 6: If $E_R(i) < \delta$stop, then terminate the learning procedure; otherwise, goto step 7.

Step 7: Update parameters of the network and set $i = i+1$, goto step 3.

VI. Experimental Results

In this section, several experiments are conducted to evaluate the performance of the proposed new RBF with the SRBF activation function and the robust learning algorithm. Let $f(x_p, y_p)$ be input-output pairs of a selected function. The input x_p is generated by a uniform distribution under the considered domain D. The corresponding output y_p is taken as $y_p = f(x_p)$. For some y_p's, y_p is incurred by a large error, i.e., $y_p = f(x_p) + e_p$, where e_p is an error. In this case, y_p's will be considered as outliers.

It is often the situation that networks terminate their training procedures when the values of the objective functions approach zero. However, if training patterns happen to be outliers or contain outliers which are far away from the underlying function, the value of objective function of outliers will not approach zero. In Section IV, we had addressed this issue and suggested that the average of all residuals could well represent the desired residual distribution r_{ave}. Thus, (12) can be modified by the following condition.

$$E_R(r_p) = \sum_{p=1}^{P} [\phi(r_p) - \phi(0)], \quad \text{subject to } r_p < r_{\text{ave}}. \tag{29}$$

Here, r_{ave} is the average of all residuals.

Example 6.1:

In the example, the underlying function $f(x) = 0.5x\ \sin(x) + \cos^2(x)$ defined on $[-6, 6]$ is used. One hundred data samples are randomly selected from $[-6, 6]$

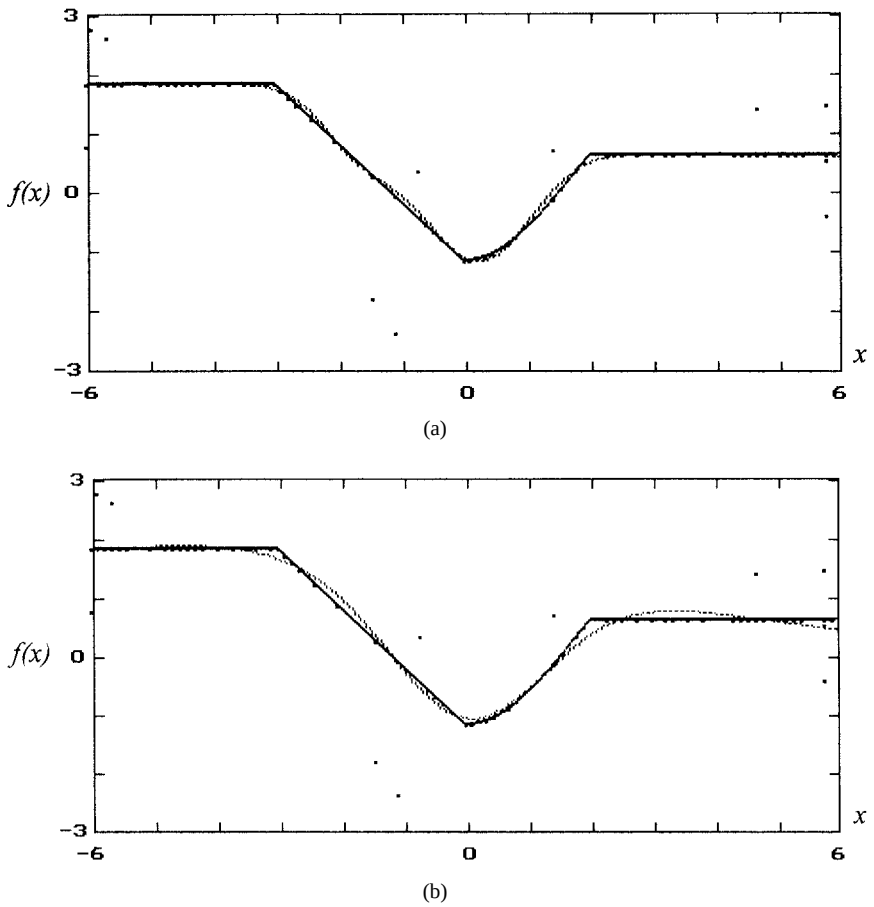


Fig. 9. Approximating the function in Example 6.3 by two distinct activation function, one is (a) SRBF and the another is (b) Gaussian function.

and used as training patterns of which nine are outliers. The initial value of σ in the objective function is five which will be reduced gradually according to a proper decrement rate. Furthermore, we also assume that the number of nodes in the example is fixed during the training phase. Fig. 7(a) shows the result when a robust objective function $E_R(r) = ([1 - \exp(-r^2/2\sigma)]$ is used. For the purpose of comparison we also show the result using the LS criterion in Fig. 7(b). The solid lines represent the function $f(x)$ and dashed lines show the LS approximation. From Fig. (7), we can see that with the robust objective function, the network generates a better approximation for underlying function in the neighborhood of outliers.

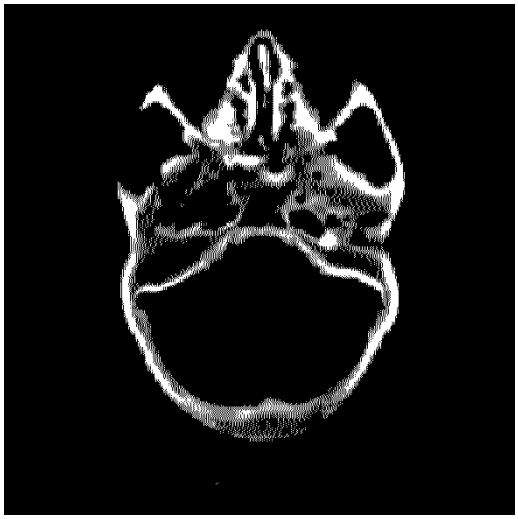
Example 6.2: In the example, we will verify the effect of the suggested adaptive growing technique with a memory queue. The function used in this example is $f(x) = 0.5x\ \sin(x) + \cos^2(x)$. The number of initial nodes is two. Check period T is set to 300. For comparison we also approximate the same function by the same growing technique without a memory queue. Fig. 8 shows the results of these approximations. As shown in the figure, with using the traditional adaptive growing technique without a memory queue the number of nodes can only grow to the maximum number six which is obviously insufficient for this example. However, if the technique is used in conjunction with a memory queue, the number of nodes may grow to nine and all training patterns are successfully interpolated.

Example 6.3: In the example, we verify that the SRBF is indeed a good candidate to be used to approximate constant valued functions. The network is similar to that used in Example 6.2. The objective function is $E_R(r) = \sigma(1 - e^{-r^2/2\sigma})$, with $\sigma = 5$. The learning algorithm uses the proposed adaptive growing technique with a memory queue. Check period is 300. Initial number of nodes is 2. The underlying function is

$$f(x) = \begin{cases} 1.8 & x < -3 \\ -x - 1.2 & -3 \le x < 0 \\ 3e^{-0.1x}\ \sin(0.2x^2) - 1.2 & 0 \le x < 2 \\ 0.6 & 2 \le x. \end{cases} \tag{30}$$

Fig. 9(a) shows the results. The result obtained by using an RBF network with Gaussian activation functions are also plotted in Fig. 9(b). Apparently, the SRBF generates better approximation in the neighborhood of $x = -3$ and 2 than does a Gaussian function-based RBF network.

Example 6.4: To demonstrate its applicability to real applications, the SRBF network is also tested for the segmentation of head–skull magnetic resonance (MR) images. Samples obtained from the skull and the soft tissues of a head skull MR image are used to train the SRBF network. The trained network is then applied to segmenting each test image into skull, soft tissues, and background. The initial number of nodes is set to two, and the proposed adaptive growing technique with a memory queue is used during the training. Check period



(a)

(b)

Fig. 10. Image segmentation using the SRBF network with two initial nodes. The values of B are 0.8 and 5, θ are 0.8 and 2.5, μ are 30 and 100, and w are 2 and 2.5. (a) The input image of brain phantom (256 × 256 pixels), and (b) segmentation result.

T is set to 100. After 3000 learning cycles, the number of network nodes is grown to 21, and the image is segmented into three classes. The segmentation result is illustrated in Fig. 10 where Fig. 10(a) is the original image while Fig. 10(b) is the segmentation results classifying the image into background, skull and soft tissues. The result shows that the proposed method can be applied in the medical image classification problems.

VII. Conclusions

Traditionally, the sigmoidal function is considered to be not suitable for the activation function of RBF networks. However, it is shown in this paper that a composite of a proper set of sigmoidal functions may still be good for RBF networks. In addition, a network with this type of activation function presents powerful capability for function approximation, especially for constant valued functions.

In order to reduce the influence of outliers, a robust objective function is proposed for RBF networks so that the outlier problem can be taken care of and the underlying functions can be approximated more accurately. Since outliers occur in the real world applications, it is our belief that using a robust objective function is better than the LS criterion in most of practical applications.

An adaptive growing learning algorithm is also proposed to find an appropriate size of an RBF network. Furthermore, in order to avoid the effects of outliers, a memory queue is used to store the positions of nodes previously added to the network so that the network can eventually achieve a proper size for a given problem when outliers occur.

The advantages of the proposed robust RBF networks are

1) better capability of approximation to underlying functions, particularly, constant valued functions;
2) faster learning speed;
3) better size of the network;
4) higher robustness to outliers.

Appendix

The derivation of parameter update rule

$$\begin{aligned}\frac{\partial E_R(r_p)}{\partial w_i} &= \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p}\,\frac{\partial r_p}{\partial w_i} = \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p}\,\frac{\partial (t_p - f(x_p))}{\partial w_i} \\ &= \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p}\left(\partial\left[t_p - \sum_{j=1}^{n} w_j R_j(x_p)\right]\Big/\partial w_i\right) \\ &= -\sum_{p=1}^{P} \phi'(r_p) R_i(X_p), \end{aligned} \tag{31}$$

$$\begin{aligned}\frac{\partial E_R(r_p)}{\partial \beta_{ij}} &= \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p}\,\frac{\partial r_p}{\partial \beta_{ij}} = \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p} \\ &\quad \cdot \left(\partial\left[t_p - \sum_{s=1}^{r} w_s R_s(x_p)\right]\Big/\partial \beta_{ij}\right) \\ &= \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p}\left(\partial\left[t_p - \sum_{x=1}^{r} w_x \prod_{k=1}^{n}\right.\right. \\ &\quad \left.\left. \cdot \left(\frac{1}{1+e^{-\beta_{sk}(lx_{sk}^p)}} - \frac{1}{1+e^{-\beta_{sk}(rx_{sk}^p)}}\right)\right]\Big/\partial \beta_{ij}\right) \\ &= \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p}\, w_i \left\{\left[lx_{ij}^p\left(\frac{1}{1+\exp[-\beta_{ij}(lx_{ij}^p)]}\right)\right.\right. \\ &\quad \cdot \left(1 - \frac{1}{1+\exp[-\beta_{ij}(rx_{ij}^p)]}\right) \\ &\quad - rx_{ij}^p\left(\frac{1}{1+\exp[-\beta_{ij}(lx_{ij}^p)]}\right) \\ &\quad \left.\left. \cdot \left(1 - \frac{1}{1+\exp[-\beta_{ij}(rx_{ij}^p)]}\right)\right] \prod_{\substack{k=1 \\ k\neq j}}^{n}\right. \\ &\quad \cdot \left(\frac{1}{1+\exp[-\beta_{ij}(lx_{ij}^p)]} - \frac{1}{1+\exp[-\beta_{ij}(rx_{ij}^p)]}\right) \end{aligned}$$

$$
\begin{aligned}
&= \sum_{p=1}^{P} \frac{\partial \phi(r_p)}{\partial r_p} w_i \Bigg(\{ lx_{ij}^p h_i(lx_{ij}^p)[1 - h_i(lx_{ij}^p)] \\
&\quad - rx_{ij}^p h_i(rx_{ij}^p)[1 - h_i(rx_{ij}^p)] \} \\
&\quad \cdot \prod_{\substack{k=1 \\ k \neq j}}^{n} [h_i(lx_{ij}^p) - h_i(rx_{ij}^p)] \Bigg) \\
&= \sum_{p=1}^{P} \frac{\partial \phi(\eta(r_p)}{\partial r_p} w_i \\
&\quad \cdot \Bigg\{ \frac{\begin{array}{l}(lx_{ij}^p) h_i(lx_{ij}^p)[1 - h_i(lx_{ij}^p)] \\ \quad - (rx_{ij}^p) h_i(rx_{ij}^p)[1 - h_i(rx_{ij}^p)]\end{array}}{h_i(lx_{ij}^p) - h_i(rx_{ij}^p)} \\
&\quad \cdot \prod_{k=1}^{n} [h_i(lx_{ij}^p) - h_i(rx_{ij}^p)] \Bigg\} \\
&= \sum_{p=1}^{P} \phi'(r_p) w_i R_i(X_p) \\
&\quad \cdot \frac{\begin{array}{l}(lx_{ij}^p) h_i(lx_{ij}^p)[1 - h_i(lx_{ij}^p)] \\ \quad - (rx_{ij}^p) h_i(rx_{ij}^p)[1 - h_i(rx_{ij}^p)]\end{array}}{h_i(lx_{ij}^p) - h_i(rx_{ij}^p)}. \qquad (32)
\end{aligned}
$$

The derivation of m and q is same as β.

REFERENCES

[1] T. Poggio and F. Giorsi, "Network for approximation and learning," *Proc. IEEE*, vol. 78, pp. 1481–1496, 1990.

[2] A. Saha, C. L. Wu, and D. S. Tang, "Approximation, dimension reduction and nonconvex optimization using linear superposition of Gaussians," *IEEE Trans. Comput.*, vol. 42, pp. 1222–1233, 1993.

[3] S. Geva and J. Sitte, "A constructive method for multivariate function approximation by multilayer perceptrons," *IEEE Trans. Neural Networks*, vol. 3, pp. 621–624, 1991.

[4] D. S. Chen and R. C. Jain, "A robust back propagation learning algorithm for function approximation," *IEEE Trans. Neural Networks*, vol. 5, pp. 467–479, 1994.

[5] S. Lee and R. M. Kil, "A Gaussian potential function network with hierarchically self-organizing learning," *Neural Network*, vol. 4, pp. 207–224, 1991.

[6] Y. H. Cheng and C. S. Lin, "A learning algorithm for radial basis function network: With the capacity of adding and pruning neurons," in *Proc. ICNN'94*, vol. 2, pp. 797–801, 1994.

[7] K. Liano, "A robust approach to supervised learning in neural network," in *Proc. ICNN'94*, vol. 1, pp. 513–516, 1994.

Chien-Cheng Lee was born in Taipei, Taiwan, R.O.C., in 1971. He received the B.S. degree in computer and information science from National Chiao Tung University, Hsinchu, Taiwan, in 1994, and the M.S. degree in electrical engineering from National Cheng Kung University, Tainan, Taiwan, in 1996. He is currently pursuing the Ph.D. degree at National Cheng Kung University.

His research interests include pattern recognition, image processing, and neural networks.

Pau-Choo Chung received the B.S. and the M.S. degrees in electrical engineering from National Cheng Kung University, Tainan, Taiwan, R.O.C., in 1981 and 1983, respectively, and the Ph.D. degree in electrical engineering from Texas Tech University, Lubbock, in 1991.

From 1983 to 1986, she was with the Chung Shan Institute of Science and Technology, Taiwan. Since 1991, she has been with Department of Electrical Engineering, National Cheng Kung University, where she is currently a Full Professor. Her current research includes neural network, and their applications to medical image processing, medical image analysis, and video image analysis.

Jea-Rong Tsai, photograph and biography not available at the time of publication.

Chein-I Chang (S'81–M'87–SM'92) received the B.S. degree from the Institute of Mathematics, National Tsing Hua University, Hsinchu, Taiwan, R.O.C., in 1973; the M.S. degree from Soochow University, Taipei, Taiwan, in 1975; and the M.A. degree from the State University of New York, Stony Brook, in 1977. He received the M.S. and M.S.E.E. degrees from the University of Illinois, Urbana-Champaign, in 1982, and the Ph.D. degree from the University of Maryland, College Park, in 1987.

He was a Visiting Assistant Professor from January 1987 to August 1987, an Assistant Professor from 1987 to 1993, and is currently an Associate Professor in the Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County. He was a National Science Council of Taiwan Sponsored Visiting Specialist at the Institute of Information Engineering, National Cheng Kung University, Tainan, from 1994 to 1995. His research interests include information theory and coding, signal detection and estimation, multispectral/hyperspectral image processing, neural networks, and pattern recognition.

Dr. Chang is a member of SPIE, INNS, Phi Kappa Phi, and Eta Kappa Nu.