Channel Equalization Using Adaptive Complex Radial Basis Function Networks

Inhyok Cha, *Student Member, IEEE,* and Saleem A. Kassam, *Fellow, IEEE*

***Abstract*—It is generally recognized that digital channel equalization can be interpreted as a problem of nonlinear classification. Networks capable of approximating nonlinear mappings can be quite useful in such applications. The radial basis function network (RBFN) is one such network. In this paper we consider an extension of the RBFN for complex-valued signals (the complex RBFN or CRBFN). We also propose a stochastic-gradient (SG) training algorithm that adapts all free parameters of the network. We then consider the problem of equalization of complex nonlinear channels using the CRBFN as part of an equalizer. Results of simulations we have carried out show that the CRBFN with the SG algorithm can be quite effective in channel equalization.**

I. Introduction

THIS paper investigates the problem of digital channel equalization using a complex extension of the radial basis function network (RBFN). Consider the standard baseband-equivalent model of a communication system where we assume that an i.i.d. M-ary signal sequence $\{s_n\}$ is transmitted through the channel (which may be nonlinear and slowly time-varying) and corrupted by additive zero-mean i.i.d. complex Gaussian noise $\{v_n\}$. The channel may be modeled as a complex-valued FIR filter with impulse response $\{h_i, 0 \le i \le L-1\}$. The channel output y_n is then given by

$$y_n = \sum_{i=0}^{L-1} h_i s_{n-i} + v_n, \qquad v_n \sim \mathcal{N}(0, \sigma_v^2). \tag{1}$$

The integer L is called the order of the channel. A more general channel model gives

$$y_n = g(s_n, \cdots, s_{n+L+1}; \underline{\theta}) + v_n \tag{2}$$

where $g(\cdot; \underline{\theta})$ is some nonlinear function and $\underline{\theta}$ is a channel parameter vector. The channels are assumed to be bounded-input-bounded-output (BIBO) stable. Note that in the absence of noise, an output y_n of such a finite-memory channel can take on only finitely many values. A symbol-decision equalizer of order N reconstructs each input symbol s_n using N channel outputs $\{y_n \cdots y_{n-N+1}\}$. Often a delay τ is introduced in the equalizer so that at time n the equalizer estimates the input symbol $s_{n-\tau}$ rather than s_n. Typically, an equalizer consists of an FIR linear filter followed by a symbol-decision device. For a noiseless linear FIR channel, the equalizer filter coefficients may be chosen so that the overall channel-plus-equalizer filter has an impulse response that best approximates a unit impulse at time $n-\tau$. Even when noise is considered, the equalization problem may be treated essentially in the perspective of inverse filtering [1].

However, it is also clear that digital channel equalization can be viewed alternatively as an optimum classification (decision) problem [2], [3]. Consider an M-ary input sequence with alphabet $\mathcal{A} = \{a_1, \cdots, a_M\}$ going through a noiseless channel of order L. Suppose an equalizer uses as its input an $N \times 1$ channel observation vector[1] $\mathbf{y}_n = [y_n \cdots y_{n-N+1}]^T$. The equalizer input vector $\mathbf{y}_n$ is generated from the $L+N-1$ input symbols $\{s_n, \cdots, s_{n-L}, s_{n-L-1}, \cdots, s_{n-L-N-2}\}$. Since the input symbols are M-ary, the number of different equalizer input states is at most M^{L+N-1}. The goal of an equalizer is then to partition the set of the M^{L+N-1} equalizer input states of dimension N into M subsets corresponding to each symbol of the alphabet $\mathcal{A}$.

A similar interpretation holds for noisy channels. It is readily shown that the minimum probability-of-error equalizer estimates the input symbol $s_{n-\tau}$ according to

$$\begin{aligned} \hat{s}_{n-\tau} &= a_j \\ j &= \operatorname{argmax}\{\pi_k \cdot p(\mathbf{y}_n)|s_{n-\tau} = a_k, 1 \le k \le M\} \end{aligned} \tag{3}$$

where π_k is the *a priori* occurrence probability of the input symbol a_k and $p(\mathbf{y}_n|s_{n-\tau} = a_k)$ is the conditional probability density of the equalizer input state vector $\mathbf{y}_n = [y_n, \cdots, y_{n-N+1}]^T$ conditioned on $s_{n-\tau} = a_k$. Equation (3) also defines the optimum decision boundaries for the M-ary partition on the set of the equalizer inputs. It is known that these optimum decision boundaries are nonlinear hypersurfaces in general. Interestingly, even for the classical models of linear FIR channels and additive white Gaussian noise, the optimum decision boundaries are generally nonlinear except in the trivial cases of memoryless channels. Traditional linear filter (LF) equalizers can realize only linear decision boundaries and are inherently suboptimal. Therefore, networks capable of realizing nonlinear mappings are receiving attention as alternative structures for channel equalizers. The multilayer feedforward sigmoidal neural network is one structure that has been used to implement nonlinear mappings in many applications including channel equalization [4]–[7].

The radial basis function network (RBFN) [8]–[11] is a promising alternative to the multilayer feedforward neural network. Being a single-hidden-layer network, an RBFN imple-

[1] T here denotes vector transpose.

Received September 7, 1993; revised June 5, 1994. This work was supported in part by the Office of Naval Research under Grant N0014-89-J-1538.

The authors are with the Department of Electrical Engineering, University of Pennsylvania, Philadelphia, PA 19104 USA.

IEEE Log Number 9406084.

ments a nonlinear mapping on an input by linearly combining outputs of the hidden nodes, each of which is computed by evaluating a radially symmetric *basis function* of the distance between the input and a parameter vector called the *center* associated with each hidden node. RBFN's are known to have several advantages over feedforward neural nets, including easy trainability and simpler structure.

In recent investigations [3], [12]–[15], RBFN's have been considered for use in channel equalization. Most of the existing work has considered real channels and signals; in [14] complex RBFN equalizers for complex channels have been considered, and very recently they have been discussed in [15]. In this paper we define the extension of the RBFN for operation on complex signals (the complex RBFN or CRBFN). We also define an adaptive stochastic-gradient (SG) learning algorithm for all free parameters of the RBFN. The SG algorithm provides an effective means to overcome poor network initialization and resultant performance degradation, which can be especially problematic for networks with localized basis functions. We consider the application of the CRBFN in equalization of complex channels with QAM signals. We have carried out extensive simulations that confirm the usefulness of CRBFN's in this application. The results indicate that the SG training algorithm can significantly enhance the performance of CRBFN's. We give discussions on simulation results as well as some practical considerations on the use of the CRBFN in channel equalization.

II. The Complex Radial Basis Function Network

An RBFN with N-dimensional real vector input and a real output implements a mapping $f\colon R^N \rightarrow R$ according to

$$f(\mathbf{x}) = \sum_{j=1}^{M} w_j \phi(\|\mathbf{x} - \mathbf{c}_j\|) \tag{4}$$

where $\mathbf{x} \in R^N$ is the input vector, $\phi(\cdot)$ is a given basis function from R^+ to R, $\|\cdot\|$ denotes the L_2 norm, $w_j, 1 \le j \le M$ are connection weights, and $\mathbf{c}_j, 1 \le j \le M$ are known as centers of the RBFN. An RBFN is thus completely specified given the M weights and centers, and the nonlinearity $\phi(\cdot)$.

Two popular choices for the nonlinearity $\phi(\cdot)$ are the thin-plate-spline (TPS) function [8]

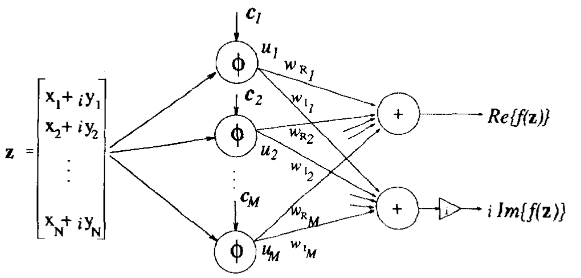
$$\phi(r) = \frac{r^2}{\sigma^2} \log(r/\sigma) \tag{5}$$

and the Gaussian [6]

$$\phi(r) = \exp(-r^2/\sigma^2) \tag{6}$$

where σ is a real parameter that determines the spread of the function. We note that behavior of these two functions is quite different for $r \rightarrow \infty$. For the function (5) $\phi(r) \rightarrow \infty$ as $r \rightarrow \infty$ (unbounded, nonlocalized), while for (6) $\phi(r) \rightarrow 0$ as $r \rightarrow \infty$ (bounded, localized).

We have generalized the RBFN to operate on complex-valued signals using an extension of the real RBFN (Kassam and Cha [14]). Fig. 1 depicts such a network, which we call a complex RBFN or CRBFN. In a CRBFN, both the input and the centers are complex-valued. The L_2 norm of the difference between the input and each center is now defined in the usual way for complex-valued vectors, and the basis functions remain real-valued. In order to implement complex-valued outputs, we assign two different sets of weights, one for the real part of the network output and the other for the imaginary part. Therefore, our CRBFN operates on a complex-valued input vector $\mathbf{z} \in C^N$ to produce a complex output according to



$$u_j = \phi(\|\mathbf{z} - \mathbf{c}_j\|),\ j = 1, 2, \cdots, M$$

Fig. 1. Schematic of the complex radial basis function network (CRBFN).

$$f(\mathbf{z}) = \sum_{j=1}^{M} (w_{R_j} + iw_{I_j})\phi(\|\mathbf{z} - \mathbf{c}_j\|) \tag{7}$$

where

$$\|\mathbf{z} - \mathbf{c}_j\|^2 \triangleq (\mathbf{z} - \mathbf{c}_j)^H (\mathbf{z} - \mathbf{c}_j). \tag{8}$$

Here the superscript H denotes complex conjugate transpose, $\phi(\cdot)\colon R^+ \rightarrow R$ is a real-valued radial basis function, $w_j = w_{R_j} + iw_{I_j}$, and $\mathbf{c}_j \in C^N, 1 \le j \le M$ denote the M complex connection weights and N-dimensional complex-valued centers, respectively. An examination of (7) and (8) reveals that the CRBFN treats the real and imaginary parts of an input as if they were two separate real inputs. Therefore, a CRBFN with N complex inputs and a complex output can be viewed alternatively as a real RBFN with $2N$ real inputs and two real outputs. In this regard, the CRBFN is a straightforward extension of the real RBFN.[2]

Generalization of real-valued structures for operation on complex signals has received much attention in the field of neural networks [4]–[7]. The current trend in this area is to develop new definitions for complex-valued pseudosigmoidal neuron activation functions and backpropagation-type learning algorithms based on the defined activation functions. A problem with such an approach is that it is difficult to find natural complex-valued extension of a real sigmoid. Simple analytic extension, for example, results in unbounded activation functions. Modification to bounded activation functions often results in unnatural extensions of real sigmoids.

The CRBFN model, since it preserves the radial symmetry of its real counterpart, is a natural extension of the RBFN. Since the operation of the CRBFN is based on superposition of real-valued basis functions (which are nonanalytic), the

[2] Very recently, an essentially identical complex RBFN extension structure has been independently proposed in [11].

functions a CRBFN implements are generally nonanalytic. However, the CRBFN can approximate any analytic function arbitrarily well in a compact domain. In fact, it is a universal approximator of any continuous mapping in a compact domain.

We now indicate why for any continuous mapping $g(\mathbf{z}): U \subset \mathcal{C}^N \rightarrow \mathcal{C}$ where U is compact, there exists a CRBFN that can approximate $g(\mathbf{z})$ arbitrarily well. For any such U we can construct an equivalent compact domain $V \subset R^{2N}$ such that $V = \{\mathbf{v} | \mathbf{v} = [v_1, v_2, \cdots, v_{2N-1}, v_{2N}]^T$ with $v_{2k-1} = x_k$, $v_{2k} = y_k$ and $x_k + iy_k \in U\}$. We note that U and V are one-to-one. Let $g_R(\mathbf{v})$ and $g_I(\mathbf{v})$ denote, respectively, the real and imaginary parts of $g(\mathbf{z})$ expressed as functions of $\mathbf{v}$. Then, it is obvious that both g_R and g_I are continuous on $V \subset R^{2N}$, for $g(\mathbf{z})$ is continuous on U. Now, it has been shown that for any continuous function $h: R^{2N} \rightarrow R$, there exists an RBFN that approximates it arbitrarily closely in a compact domain.[3]

Therefore, there exist two $2N$-dimensional RBFN's, one approximating $g_R(\mathbf{v})$ and the other $g_I(\mathbf{v})$ on the compact set V.

A CRBFN that approximates $g(\mathbf{z})$ can then be constructed in the following way. First, let the CRBFN centers be specified by converting all the $2N$-dimensional center vectors of the two real RBFN's into a set of N-dimensional complex-valued vectors with the σ_j parameters associated with each of the centers retained for the CRBFN. The CRBFN needs two sets of weights, one for the real part of the network output and the other for the imaginary part. For the real weights of the CRBFN, the weights of the real RBFN that approximates $g_R(\mathbf{v})$ are assigned to the corresponding nodes that came from this real RBFN. For those nodes that came from the other real RBFN approximating $g_I(\mathbf{v})$, we simply set their real weights to zero. Likewise, the imaginary weights of the CRBFN are specified by retaining those weights that correspond to the real RBFN approximating $g_I(\mathbf{v})$ and setting the remaining weights to zero. It is readily seen that the CRBFN so constructed approximates $g(\mathbf{z})$ in the compact domain U.

III. The Stochastic-Gradient (SG) Training Algorithm

Training or learning of an RBFN of a given size M and a basis function $\phi(\cdot)$ is equivalent to finding a set of network parameters including the M weights $\{w_j\}$, centers $\{\mathbf{c}_j\}$, and the spread parameters $\{\sigma_j\}$ for individual basis function nodes, such that the resulting network closely approximates the underlying mapping from the input to the output in the training set. In this section we will consider a stochastic-gradient-based algorithm for RBFN training.

Since the CRBFN is a natural extension of the real RBFN, training algorithms that work for real RBFN's should also work for CRBFN's. Some algorithms use unsupervised clustering techniques to choose the centers and then obtain the linear least-squares (LS) weights once the centers are fixed. The Moody-Darken algorithm [17] is one such method that uses the K-means clustering algorithm. A hybrid algorithm [18] combins both unsupervised and supervised training by clustering the centers (via K-means) and simultaneously adjusting the weights (via RLS or LMS). Proper choice of the centers is often critical for good performance. The orthogonal-least-squares (OLS) method [12] chooses the centers one by one from training input samples so that addition of each new center may maximize the incremental gain in the total squared error of approximation.

Stochastic-Gradient (SG) Algorithm

In an earlier paper [19], we have proposed a simple stochastic-gradient (SG) training algorithm for *real* RBFN's. The method adapts all the free parameters of the network simultaneously by using stochastic gradient descent for the error criterion.[4] Simulation results suggest that the algorithm is very useful, with performance achieved often being superior to that of some existing algorithms. We now present the SG algorithm for complex RBFN's.

Let $f(\mathbf{z}_n)$ denote the CRBFN output corresponding to input $\mathbf{z}_n$ at time n as given by

$$f(\mathbf{z}_n) = \sum_{j=1}^{M} w_{j,n}\phi(\|\mathbf{z}_n - \mathbf{c}_{j,n}\|; \sigma_{j,n}). \tag{9}$$

Here the subscript n denotes the time index, $w_{j,n}$ is the jth complex-valued CRBFN weight at time n, $\phi(\cdot; \sigma)$ is the basis function parametrized by σ as in (5) and (6), and the rest of the notation follows (7). Also, let d_n and $e_n = f(\mathbf{z}_n) - d_n$ denote the complex-valued desired response and error at time n, respectively. The training set consists of a number of pairs of input and desired response $(\mathbf{z}_n, d_n)$. The SG algorithm takes the instantaneous gradient of the squared error $\|e_n\|^2$ and moves the network parameters in the opposite direction of their respective gradients. Thus, a network parameter ρ of the network (which can be a weight, a center, or a spread parameter) is adapted at time n according to

$$\rho_{n+1} = \rho_n - \mu_\rho \frac{\partial \|e_n\|^2}{\partial \rho_n} \tag{10}$$

where μ_ρ controls the speed of adaptation. Momentum versions of the SG algorithm can be readily defined to utilize some degree of memory in adaptation.

The SG algorithm does not guarantee convergence to globally optimum network parameters. However, it does appear to converge to reasonable solutions in practice. The method can be used as a single-stage learning algorithm if training data are only sequentially available or as the second-stage method of a two-stage algorithm where centers and spread parameters are predetermined by a method such as the OLS or a clustering technique. Often, pretraining on the centers improves approximation performance. For the single-pass case, centers can be initialized by, for example, forming them with the first few training samples.

The SG algorithm has certain advantages over existing methods. First, all free network parameters are adapted simultaneously, usually yielding improved overall solutions. The method can also provide greater robustness to poor initial choices of parameters, especially the centers. Second, the

[3]The proof can be found in [16].

[4]A similar algorithm that uses gradients of total, rather than instantaneous, squared error has been reported in [20].

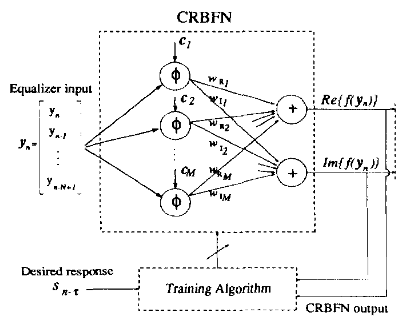


Fig. 2. Schematic of a CRBFN equalizer.

algorithm is well-suited for on-line adaptive signal processing unlike block-processing algorithms such as the Moody-Darken or the OLS algorithms. It is also computationally quite feasible.

It has been observed [9], [10] that RBFN's with certain nonlocalized basis functions such as the thin-plate-spline (5) sometimes give better approximation performance compared to networks with localized functions. However, nonlocalized basis functions are not well-suited for training by the SG algorithm, since gradients of these basis functions with respect to centers and σ's can be nonlocalized themselves or may even become unbounded. Unbounded gradients may cause numerical instability in the SG method, for example, when the RBFN input is a noisy outlier located quite far from all the centers. We believe that the SG algorithm is more suited for RBFN's with localized basis functions.

Among localized functions, the Gaussian is the most popular choice. We therefore present specifically the SG algorithm for the Gaussian CRBFN. Such a network with M basis functions gives the output $f(\mathbf{z}_n)$ at time n as

$$f(\mathbf{z}_n) = \sum_{j=1}^{M} w_{j,n} \exp\left(-\|\mathbf{z}_n - \mathbf{c}_{j,n}\|^2/\sigma_{j,n}^2\right). \tag{11}$$

With $\phi_j(\mathbf{z}_n) \triangleq \exp(-\|\mathbf{z}_n - \mathbf{c}_{j,n}\|^2/\sigma_{j,n}^2)$, the SG algorithm (without momentum) adapts the network parameters according to (12)–(14), at the bottom of this page.

Since Gaussians are fast-decaying functions, it can be assumed that not all the basis function units contribute significantly to the network output. Hence, instead of training all the hidden nodes, one could train only a selected number of basis function nodes with the largest output values. When the input dimension is high, it is the adaptation of the centers that requires the most computation. Computation can be significantly reduced if, for example, at each training iteration only one center is adaptively moved while the weights and the σ parameters are adapted for all nodes.

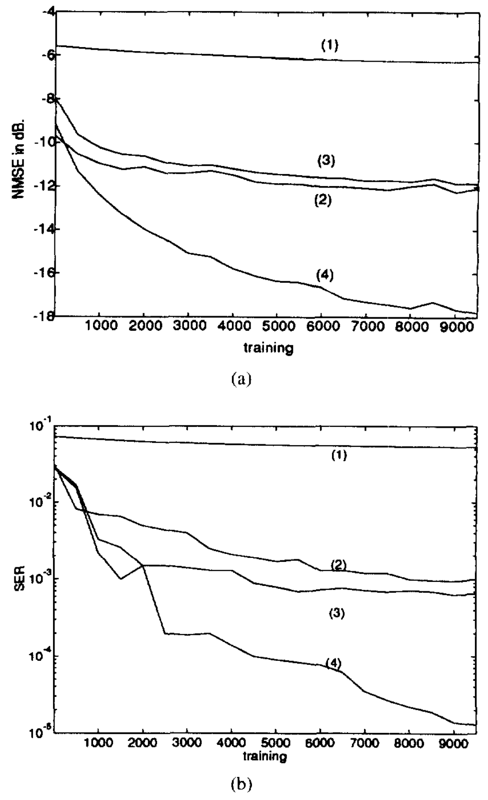


Fig. 3. Example 1: Performance of equalizers for an order-2 channel with nonlinearity, 4-QAM. (a) NMSE values in dB. (b) SER values—(1) linear FIR(20), (2) Gaussian/hybrid with 30 centers, (3) TPS/MD with 30 centers, and (4) Guassian/SG with 20 centers.

IV. CRBFN Equalizer Simulation Results

In this section we present and discuss simulation results for complex channel equalization using CRBFN's. The CRBFN equalizer we considered is depicted in Fig. 2. Here y_n denotes the complex-valued output of a complex channel at time n. For an equalizer of order N, the equalizer input vector at time n is given by the N-dimensional complex-valued channel observation vector $\mathbf{y}_n = [y_n y_{n-1} \cdots y_{n-N+1}]^T$. The CRBFN output $f(\mathbf{y}_n)$ is fed into a nearest-neighbor decision device to give an estimate $\hat{s}_{n-\tau}$ of the transmitted symbol $s_{n-\tau}$, where τ is the equalizer decision delay. A specific number of input-response pairs $\{(\mathbf{y}_k, s_{k-\tau})\}$ forms the training set.

$$w_{j,n+1} = w_{j,n} + \mu_w e_n \phi_j(\mathbf{z}_n) \tag{12}$$

$$\sigma_{j,n+1} = \sigma_{j,n} + \mu_\sigma \phi_j(\mathbf{z}_n)[w_{R_{j,n}} \mathrm{Re}\{e_n\} + w_{I_{j,n}} \mathrm{Im}\{e_n\}] \cdot \frac{\|\mathbf{z}_n - \mathbf{c}_{j,n}\|^2}{\sigma_{j,n}^3} \tag{13}$$

$$\mathbf{c}_{j,n+1} = \mathbf{c}_{j,n} + \mu_c \phi_j(\mathbf{z}_n) \left[\frac{w_{R_{j,n}} \mathrm{Re}\{e_n\} \mathrm{Re}\{\mathbf{z}_n - \mathbf{c}_{j,n}\} + i w_{I_{j,n}} \mathrm{Im}\{e_n\} \mathrm{Im}\{\mathbf{z}_n - \mathbf{c}_{j,n}\}}{\sigma_{j,n}^2} \right]. \tag{14}$$

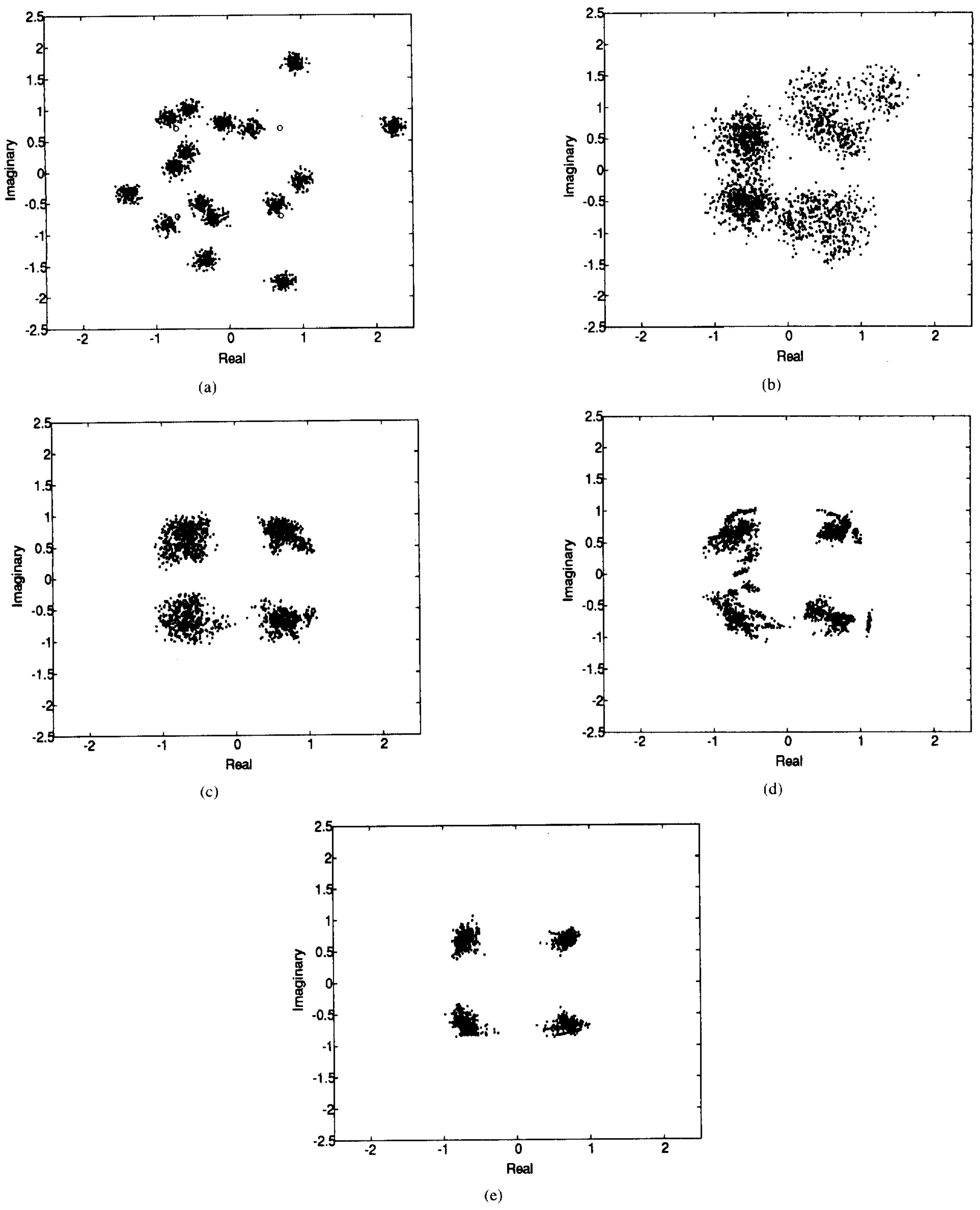


Fig. 4. Example 1: 4-QAM signals, order-2 channel with nonlinearity (all after 5000 iterations). (a) Test set channel output distribution. (b) FIR linear equalizer output. (c) TPS/MD equalizer output. (d) Gaussian/hybrid equalizer output. (e) Gaussian/SG equalizer output sets.

We tested CRBFN's with both localized (Gaussian) and nonlocalized (TPS) basis functions. For the CRBFN's, the training methods compared were the SG algorithm, Moody-Darken (MD) algorithm [17], and the hybrid LMS algorithm [18]. FIR transversal LF equalizers with LMS training were also tested for comparison. Also, a complex multilayer perceptron (MLP) as defined in [7] with complex backpropagation (BP) training was tested. In the simulations, performance was measured by the symbol-error-rate (SER) as well as the normalized mean-squared-error (NMSE) between the equalizer output and the correct symbols. Samples of the training set and the test set were generated independently from each other according to the specific channel and input signaling scheme of each example. All test sets had 100 000 samples that were used for the performance evaluation of the trained equalizers.

We tested the following equalizer/training-algorithm combinations.

1) FIR linear equalizer/LMS algorithm,
2) TPS CRBFN/Moody-Darken (MD) algorithm,
3) Gaussian CRBFN/hybrid algorithm,
4) Gaussian CRBFN/SG algorithm, and
5) Complex MLP/BP algorithm.

The sizes of the CRBFN's in the simulations reflect those for which the resulting networks gave reasonably good performance. The initial centers were formed from the first few successive channel output samples of the training set. For the networks trained with Moody's algorithm, K-means clustering was done on the entire training set. The CRBFN weights were initialized to small random values. The spread parameters σ_j were initially set to one common value for all the basis function units of the CRBFN's. We have found that the performance is rather robust to variation over a significant range of values for the initial spread parameter. In the examples, we set this initial value based on a consideration of the average or the minimum distance between signal constellation values.[5] The values of adaptation coefficients for training algorithms cited in the examples were chosen to result in good balance of convergence speed and final equalization performance.

Example 1: In this example, we considered the performance of CRBFN equalizers for 4-QAM signals in the presence of an instantaneous channel nonlinearity and noise. The overall channel output was given by

$$y_n = o_n + 0.2o_n^2 + 0.1o_n^3 + v_n, \qquad v_n \sim \mathcal{N}(0, 0.01) \quad (15)$$
$$o_n = (1 - i0.3434)s_n + (0.5 + i0.2912)s_{n-1}. \quad (16)$$

The equalizer input dimension was set to $N = 2$ and the initial common value of the spread parameter was set to $\sigma = \sqrt{2}$. Also, the equalizer decision delay was $\tau = 0$. The Gaussian/SG equalizer had 20 hidden units. Both the TPS/MD equalizer and the Gaussian/hybrid equalizer had 30 hidden units. The adaptation coefficients for the SG algorithm were $\mu_w = 0.05, \mu_\sigma = 0.03$, and $\mu_c = 0.05$ with no momentum. The linear filter equalizer had 20 taps. The LMS adaptation coefficient $\mu_w = 0.03$ for the linear equalizer.

Fig. 3(a) and (b) show the NMSE and SER convergence of the various equalizers. We can see that the FIR linear equalizer performed very poorly due to the nonlinearity. All the CRBFN equalizers gave better performance than the linear equalizer. Note the superiority of the Gaussian/SG equalizer to the other equalizers based on the CRBFN. The major performance improvement of the Gaussian/SG equalizer over the Gaussian/hybrid equalizer is particularly noteworthy. The TPS/MD network with 30 hidden units gave better performance than the Gaussian/hybrid network with 30 hidden units. However, the Gaussian/SG equalizer significantly outperformed the thin-plate-spline function network. Fig. 4(a) shows the 4-QAM test set channel output distribution. Fig. 4(b)–(e) show the test set equalizer output sets for the various equalizers after training on 5000 samples.

[5]This initial value can also be set to be the average pairwise distance between the initial centers.

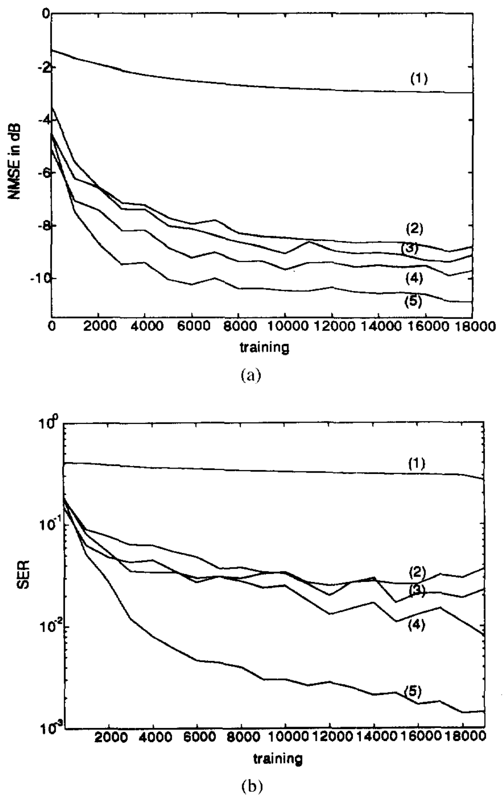


Fig. 5. Example 2: Performance of equalizers for an order-3 non-minimum-phase channel with nonlinearity, 4-QAM. (a) NMSE values in dB. (b) SER values—(1) linear FIR(30), (2) complex MLP(3, 20, 10), (3) Gaussian/hybrid with 30 centers, (4) TPS/MD with 30 centers, and (5) Gaussian/SG with 20 centers.

Example 2: This example shows equalization of a nonminimum-phase channel of order 3 with nonlinear distortion for 4-QAM signaling. The channel output was given by

$$y_n = o_n + 0.1o_n^2 + 0.05o_n^3 + v_n, \qquad v_n \sim \mathcal{N}(0, 0.01) \quad (17)$$
$$o_n = (0.34 - i0.27)s_n + (0.87 + i0.43)s_{n-1} + (0.34 - i0.21)s_{n-2}. \quad (18)$$

Here, the equalizer input dimension was chosen to be $N = 3$, and the initial common value of the spread parameter was $\sigma = \sqrt{2.5}$. Also, the equalizer decision delay was $\tau = 0$. The Gaussian/SG equalizer had 30 hidden units. Both the TPS/MD equalizer and the Gaussian/hybrid equalizer had 60 hidden units. In this example, we also tested a complex MLP with two hidden layers, one with 20 hidden nodes and the other with 10 nodes (we call this structure MLP(3, 20, 10)). Weight adaptation coefficient for the MLP backpropagation was 0.05. The parameter adaptation coefficients for the CRBFN equalizers were the same as in Example 1. The linear filter equalizer had 30 taps with LMS adaptation coefficient 0.03.

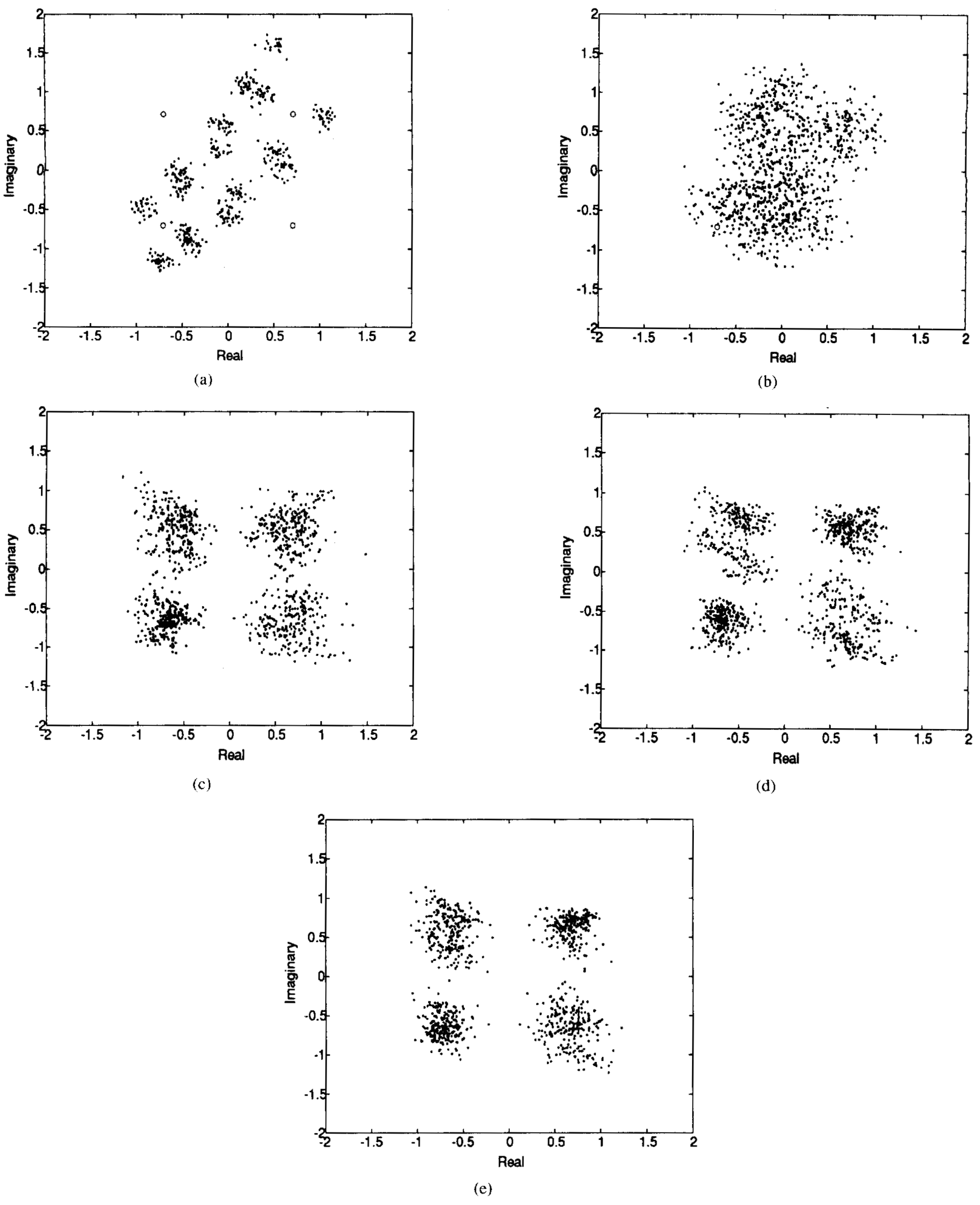
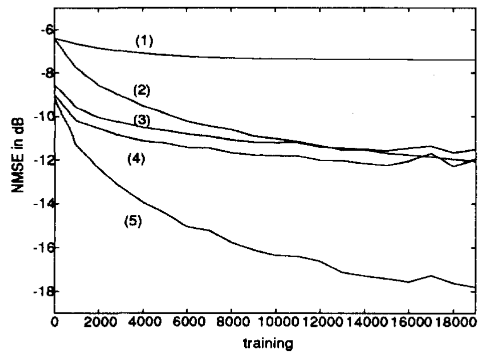
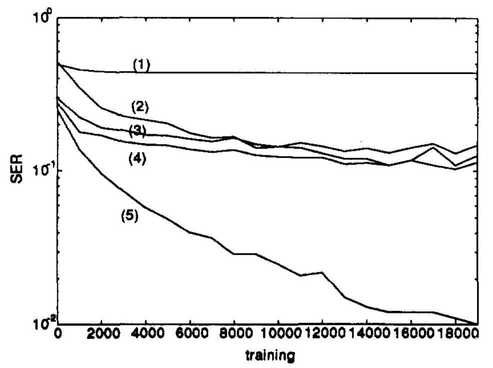


Fig. 6. Example 2: 4-QAM signals, order-3 nonminimum-phase channel with nonlinearity (all after 20 000 iterations). (a) Test set channel output distribution. (b) FIR(30) LF equalizer output. (c) complex MLP(3, 20, 10) equalizer output. (d) TPS/MD (60 centers) equalizer output. (e) Gaussian/SG (30 centers) equalizer output.

Fig. 5(a) and (b) show, respectively, the NMSE and the SER convergence of the various equalizers. Fig. 6(a) shows the channel output distribution and Fig. 6(b)–(e) show the equalizer output sets of various equalizers after training on 20 000 training samples. We can see that with no decision delay the FIR linear filter equalizer totally failed to equalize the nonminimum-phase channel. CRBFN equalizers again gave much better performance than the linear equalizer. However, due to the increased input dimension, the required number of hidden nodes for the CRBFN equalizers were higher than in Example 1. We note the clear superiority of the Gaussian/SG equalizer over the other CRBFN equalizers. The complex MLP



(a)



(b)

Fig. 7. Example 3: Performance of equalizers for an infinite-memory channel with nonlinearity, 16-QAM. (a) NMSE values in dB. (b) SER values—(1) linear FIR(30), (2) TPS/MD with 60 centers, (3) Gaussian/hybrid with 60 centers, (4) complex MLP(2, 20, 10), and (5) Gaussian/SG with 30 centers.

equalizer and the CRBFN equalizers without SG training had roughly equal performance. However, the number of adapted parameters for the MLP was much higher than for the CRBFN equalizers.

Example 3: In this example we considered a 16-QAM signal and a nonlinear channel with infinite memory. The channel output y_n was given by

$$y_n = -0.3y_{n-1} + o_n + 0.02o_n^2 + 0.01o_n^3 + v_n, \quad v_n \sim \mathcal{N}(0, 0.01) \tag{19}$$

$$o_n = (1 - i0.3434)s_n + (0.5 + i0.2912)s_{n-1}. \tag{20}$$

The channel had infinite memory due to the feedback in (19). For the CRBFN-based equalizers, the input dimension was chosen to be $N = 2$ and the initial spread parameter was $\sigma = \sqrt{5}$. The Gaussian/SG equalizer had 30 hidden units. The Gaussian/hybrid equalizer and the TPS/MD equalizer both had 60 hidden units. The linear FIR equalizer had 30 taps. We also tested a complex MLP(2, 20, 10) equalizer.

The NMSE and the SER convergence of the various equalizers are shown in Fig. 7(a) and (b), respectively. Fig. 8(a) depicts the channel output distribution. Fig. 8(b)–(e) depict the equalizer output distributions for the various equalizers when training was done once on the 20 000-sample training set. The Gaussian/SG equalizer with 30 units performed markedly better than all the other equalizers. The complex MLP equalizer's performance was slightly better than that of the TPS/MD equalizer and the Gaussian/hybrid equalizer, both with 60 hidden units. The linear FIR equalizer performed very poorly. This example again shows the advantage of the SG algorithm.

V. Discussions

From the above and other simulation results, we conclude that the proposed CRBFN can be quite useful in digital channel equalization, especially for low-order channels with nonlinear distortion. Also, the proposed stochastic-gradient (SG) algorithm was shown to be very effective and superior to several existing training algorithms. However, we have found that for increasing channel order, both the input dimension N and the number of hidden nodes M have to increase quite fast to maintain good performance. So far, CRBFN equalizers with few tens of hidden units have been effective mainly on relatively low-order channels ($L \leq 4$). This suggests the need to investigate more efficient structures.

In some existing work [3], [12], [15], use of Gaussian basis functions have been advocated on the grounds that, in the presence of Gaussian noise, Gaussian basis functions are naturally fitting to realize the conditional probability densities $p(\mathbf{y}_n | s_{n-\tau} = a_k), 1 \leq k \leq M$, since these densities turn out to be superpositions of Gaussians. The purpose of using the Gaussian RBFN in this perspective is to match the conditional densities centered on the noise-free equalizer input states with the RBFN.

We believe, however, that the advantage of Gaussian basis functions has another explanation. First, we note that the optimum decision regions in (3) are realized not by *superposing* Gaussian densities, but by *comparing* different pairs of superposed-Gaussian conditional densities. A Gaussian RBFN with one output and a thresholding decision device cannot realize the multiple optimum boundaries required of an M-ary signaling case just because its basis functions are centered exactly on the noise-free equalizer input states and the σ value is matched to the channel noise variance. In other words, such an approach does not result in the realization of the optimum classifiers, except in binary signaling cases where the comparison is only between two conditional densities. Second, there are practical hurdles in the method of density matching by Gaussian RBFN's. The exact number and locations of the noise-free equalizer input states (which should match the RBFN centers) are not known, since the channel is unknown. Also, the variance of the Gaussian noise can at best be estimated. In addition, even if this information can be provided, it often requires impractically many hidden nodes to match the number of the noise-free equalizer input states. The total number of noise-free equalizer input states (thus the number of hidden units) in this approach is M^{L+N-1}, where M is the signal alphabet size and L and N are the orders of the channel and the equalizer, respectively. Consider 16-QAM signals. Even for a channel of order $L = 2$ and equalizer order $N = 2$, the total number of the noise-free equalizer input states is $16^{2+2-1} = 4096$! In our simulations,

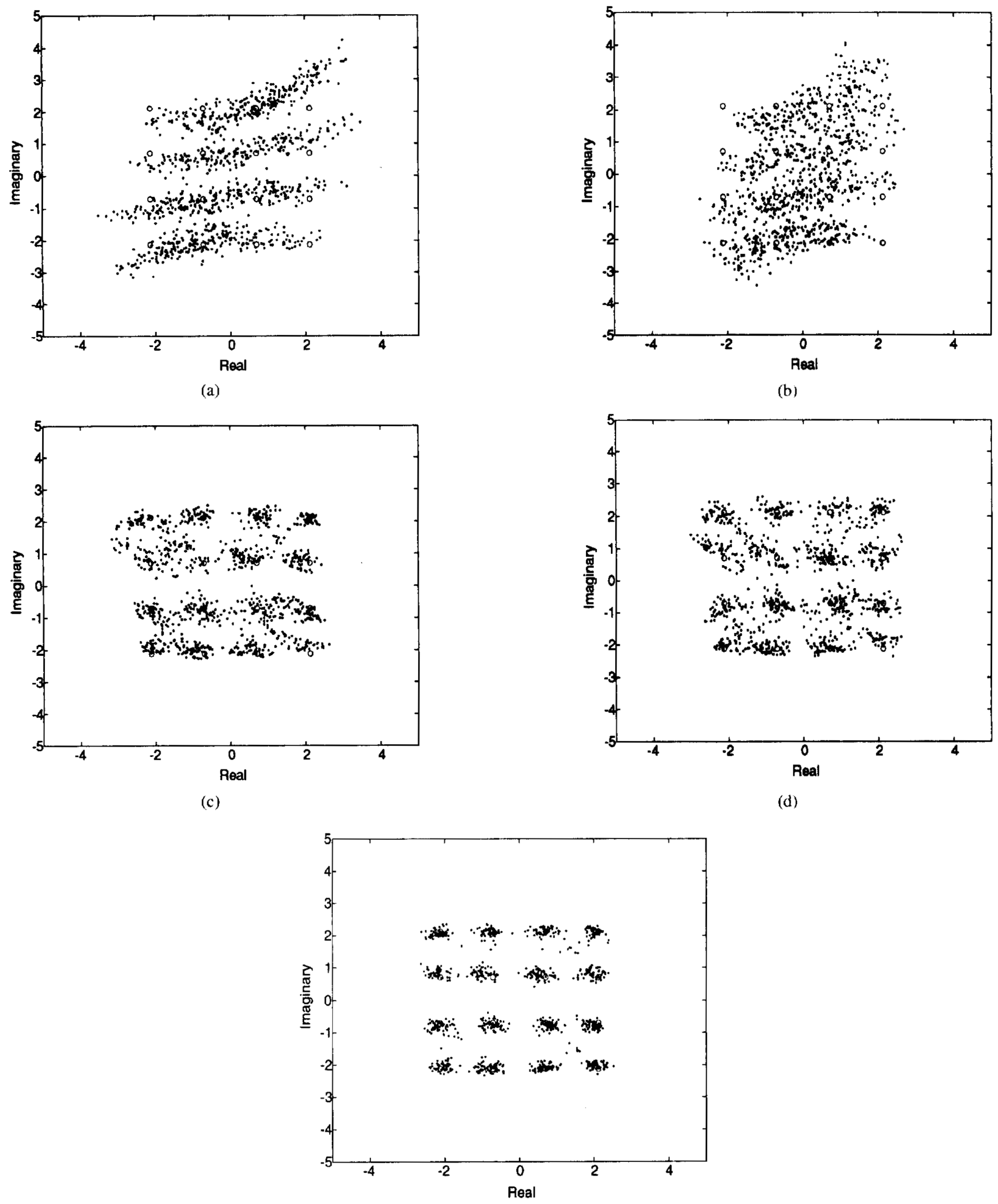


Fig. 8. Example 3: 16-QAM signals, infinite-memory channel with nonlinearity (all after 20 000 iterations). (a) Test set channel output distribution. (b) FIR linear equalizer output. (c) TPS/MD equalizer (60 centers). (d) complex MLP(2, 20, 10) equalizer. (e) Gaussian/SG equalizer (30 centers) output set.

the TPS/MD CRBFN equalizers performed as well as or even better than the Gaussian/hybrid equalizers of comparable size, although the TPS/MD equalizers did not reach the performance of the Gaussian/SG equalizers. We believe, therefore, that any advantage that Gaussians have over other choices of basis functions may be more related to the availability of effective gradient-based training algorithms such as the SG algorithm.

A way of achieving better performance with a network with modest size is to use decision-feedback (DFB) equalizers [15]. If prior symbols are estimated correctly, DFB equalizers can achieve faster convergence and better performance. It is shown in [15] that the decision boundaries of the Bayes DFB equalizers are also determined by comparing superpositions of Gaussian conditional densities, only in this case the number of equalizer input states required to implement the optimum boundaries is significantly reduced from the case of transversal equalizers. Application of the SG algorithm in DFB equalizers would be a worthwhile investigation.

Obviously, RBFN's are not the only structure for implementing nonlinear mappings. In our simulations, the complex MLP equalizer with its structure and training algorithm as defined in [7] did not perform as well as the Gaussian/SG CRBFN equalizers. However, this does not mean that neural network structures (or their complex extensions) are generally inferior to the CRBFN equalizers. More rigorous analysis is required for any meaningful comparison between these two types of nonlinear structures. Also, further investigation on RBFN's with different basis functions might be needed. The elliptic basis function [10] and the normalized basis function [21] are just two of the interesting alternatives.

An interesting application of RBFN's is in blind equalization. In earlier work [13], we have reported an RBFN blind equalizer scheme for real signals, in which a linear blind equalizer and an RBFN work in parallel so that the RBFN can be trained using the output of the linear equalizer. We are currently exploring feasibility of such schemes applied to CRBFN blind equalizers.

References

[1] S. U. H. Qureshi, "Adaptive equalization," *Proc. IEEE,* vol. 73, pp. 1349–1387, Sept. 1985.

[2] S. Chen, G. J. Gibson, C. F. N. Cowan, and P. M. Grant, "Adaptive equalization of finite non-linear channels using multilayer perceptrons," *Signal Processing,* vol. 20, pp. 107–109, 1990.

[3] S. Chen, G. J. Gibson, C. F. N. Cowan, and P. Grant, "Reconstruction of binary signals using an adaptive radial-basis-function equalizer," *Signal Processing,* vol. 22, no. 1, pp. 77–93, 1991.

[4] Z. Xiang and G. Bi, "Complex neuron model with its applications to M-QAM data communications in the presence of co-channel interferences," in *Proc. ICASSP 92,* vol. II, Mar. 1992, pp. II-305-308.

[5] N. Benvenuto, M. Marchsi, F. Piazza, and A. Uncini, "A comparison between real and complex-valued neural networks in communication applications," in *Proc. Int. Conf. Artificial Neural Networks* (Espoo, Finland), June 1991, pp. 1177–1180.

[6] H. Leung and S. Haykin, "The complex backpropagation algorithm," *IEEE Trans. Signal Processing,* vol. 39, no. 9, pp. 2101–2104, Sept. 1991.

[7] N. Benvenuto and F. Piazza, "On the complex backpropagation algorithm," *IEEE Trans. Signal Processing,* vol. 40, no. 4, pp. 967–969, Apr. 1992.

[8] C. A. Micchelli, "Interpolation of scattered data: Distance matrices and conditionally positive definite functions," *Constructive Approximation,* vol. 2, pp. 11–22, 1986.

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

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

[11] S. Chen, S. McLaughlin, and B. Mulgrew, "Complex-valued radial basis function networks, Part I: Network architecture and learning algorithms," *Signal Processing,* vol. 35, pp. 19–31, Jan. 1994.

[12] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Networks,* vol. 2, no. 2, pp. 302–308, Mar. 1991.

[13] I. Cha and S. A. Kassam, "Blind equalization using radial basis function networks," *Proc. Canadian Conf. Elect. Eng. Comput. Sci.,* vol. 1, Sept. 1992, pp. TM.3.13.1-4.

[14] S. A. Kassam and I. Cha, "Radial basis function networks in nonlinear signal processing," presented at *The 27th Annu. Asilomar Conf. Signals, Syst., Comput.,* Nov. 1993.

[15] S. Chen, S. McLaughlin, and B. Mulgrew, "Complex-valued radial basis function networks, Part II: Application to digital communications channel equalisation," *Signal Processing,* vol. 36, pp. 175–188, Feb. 1994.

[16] J. Park and I. W. Sandberg, "Universal approximation using radial-basis function networks," *Neural Computations,* vol. 3, pp. 246–257, 1991.

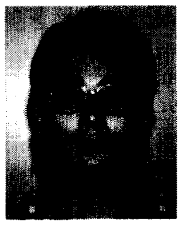
[17] J. E. Moody and C. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Networks,* vol. 1, no. 2, pp. 281–294, 1989.

[18] S. Chen, S. A. Billings, and P. M. Grant, "Recursive hybrid algorithm for non-linear system identification using radial basis function networks," *Int. J. Cont.,* vol. 55, no. 5, pp. 1051–1070, 1992.

[19] I. Cha and S. A. Kassam, "Time-series prediction by adaptive radial basis function networks," in *Proc. IEEE Twenty-Sixth Annu. Conf. Inform. Sci., Syst.,* Mar. 1993, pp. 818–823.

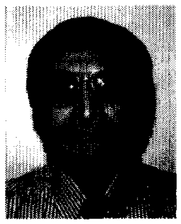
[20] A. G. Bors and M. Gabbouji, "Minimal topology for a radial basis function neural network for pattern classification," in *Digital Signal Processing.* New York: Academic, Apr. 1994.

[21] R. D. Jones *et al.,*, "Function approximation and time series prediction with neural networks," in *Proc. Int. Joint Conf. Neural Networks,* June 1990, pp. I649-666.



Inhyok Cha (S'92) was born in Seoul, Korea, on January 11, 1966. He received the B.S. and M.S. degrees in electronic engineering from the Seoul National University, Seoul, Korea, in 1988 and 1990, respectively. He is working toward the Ph.D. degree in electrical engineering at University of Pennsylvania, Philadelphia.

His research interests include nonlinear signal processing, image processing, and neural networks.



Saleem A. Kassam (F'93) was born in 1949 in Dar-es-Salaam, Tanzania. He received the B.S. degree in engineering (with Distinction) from Swarthmore College, Swarthmore, PA, in 1972, the M.S.E. and M.A. degrees in electrical engineering from Princeton University, Princeton, NJ, in 1974, and the Ph.D. degree in electrical engineering from Princeton in 1975.

Since 1975 he has been on the faculty of the Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia. He is currently the Solomon and Sylvia Charp Professor of Electrical Engineering there and has been Chairman of the Department of Electrical Engineering since 1992. He has made contributions to signal detection theory, quantization, robust signal processing, nonlinear and adaptive signal processing, image processing, spectrum estimation, and microwave and acoustic imaging. He has coedited a book on nonparametric detection and is author of *Signal Detection in Non-Gaussian Noise* (Springer-Verlag, 1988). He consults for industry and government laboratories in the areas of signal detection, spectrum estimation, acoustic imaging, image analysis, and signal processing for medical instrumentation, and holds two patents. His research interests are in the areas of statistical signal processing and communication theory.

Dr. Kassam has served as an Associate Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY and is currently a member of the board of governors of the IEEE Information Theory Society. He is a member of Phi Beta Kappa and Sigma Xi.