Non-Linear Filtering and Equalization in Non-Gaussian Noise Using Radial Basis Function and Related Networks

Inhyok Cha
Bell Laboratories
Lucent Technologies, Inc.
Whippany, NJ 07960
E-mail) icha@lucent.com

Saleem A. Kassam
Department of Electrical Engineering
University of Pennsylvania
Philadelphia, PA 19104
E-mail) kassam@ee.upenn.edu

Abstract

We review the RBFN and several related networks in the context of using them in non-linear filtering and channel equalization applications under non-Gaussian noise environments. We present simulation results that suggest that these networks are very useful for such applications.

1 Introduction

Signal processing in the presence of non-Gaussian noise has been a topic of interest for a long time, it being recognized that many real-world signal processing problems involve non-Gaussian noise environments. Many problems become analytically intractable under non-Gaussian noise, and optimum solutions often require complex and generally non-linear mappings. Thus, there is considerable interest in processing tools that enable approximations to the optimum solutions through supervised training.

The radial basis function network (RBFN) has been studied as a useful tool for signal processing in recent years. It has been applied with promising results to many signal processing problems [1]-[3]. In addition, several extensions of the RBFN structure were proposed to improve its functional capability in the literature [4]. Theoretically, it has been shown that the Gaussian RBFN and its variants provide optimum solutions in certain problems of non-linear filtering and channel equalization in Gaussian noise [4],[5]. In this paper, we investigate the RBFN structures for signal processing in non-Gaussian noise environments.

2 RBFN and related networks

In this section, we will review the Gaussian RBFN and its two previously proposed extensions, namely, the linear/RBF hybrid network, and the Gaussian Mixture Basis Function Network (GMBFN).

2.1 The Gaussian RBFN

The Gaussian radial basis function network (RBFN) gives its output $\mathbf{f}_{RB} : R^{N_i} \to R^{N_o}$ according to

$$\mathbf{f}_{RB}(\mathbf{x}) = \sum_{j=1}^{M} \mathbf{w}_j \ \exp(-\|\mathbf{x} - \mathbf{c}_j\|^2/\sigma_j^2) \qquad (1)$$

where $\mathbf{x} \in R^{N_i}$ is the input, M is the network size, $\mathbf{w}_j$, $(j = 1, \cdots, M)$ are the weights, $\mathbf{c}_j$, $(j = 1, \cdots, M)$, are parameters known as the centers, and σ_j, $(j = 1, \cdots, M)$ are the spread parameters. In training of the network, these network parameters must be determined via some learning algorithm. Examples of such algorithms are given in [3].

2.2 The Linear/RBF Hybrid Network

In the linear/RBF hybrid network, the output functional $\mathbf{f}_{HB} : (R^{N_i}, R^{N_L}) \to R^{N_o}$ is given by

$$\mathbf{f}_{HB}(\mathbf{x}_L, \mathbf{x}_R) = \mathbf{h}_{N_L}^T \cdot \mathbf{x}_L + \sum_{j=1}^{M} \mathbf{w}_j \exp(\frac{-\|\mathbf{x}_R - \mathbf{c}_j\|^2}{\sigma_j^2}) \qquad (2)$$

where $\mathbf{x}_R \in R^{N_i}$ is the input vector for an RBFN and $\mathbf{x}_L \in R^{N_L}$ is the input vector for a linear FIR filter, whose impulse response is represented by the vector $\mathbf{h} \in R^{N_L}$. All the other parameters follow the descriptions given for the RBFN in equation (1). The hybrid network is thus essentially a GRBFN augmented by an FIR linear filter. Simple LMS type algorithms can be used to train such a network.

2.3 The Gaussian Mixture Basis Function Network

The Gaussian mixture basis function network (GMBFN) [4] is a normalized extension of the RBFN based on a mixture-of-Gaussian data model. In the model, an input $\mathbf{x} \in R^{N_i}$ and a (desired) output $\mathbf{y} \in R^{N_o}$ in a data of interest are assumed to be realizations of statistically dependent random vectors $\mathbf{X}$ and $\mathbf{Y}$, respectively. The vectors $\mathbf{X}$ and $\mathbf{Y}$ are assumed to follow a joint mixture distribution with M populations, each having an *a priori* probability of occurrence $\pi_j, (j = 1, 2, \cdots, M)$ and each following a Gaussian distribution. Let $\mathbf{Z}_j = [\mathbf{X}_j^T \ \mathbf{Y}_j^T]^T$, where $\mathbf{X}_j$ and $\mathbf{Y}_j$ denote the input and the output random vectors, respectively, when the data is picked from the j-th population. Also, let $\mathbf{Z}_j$ have mean $\mathbf{m}_j$ and covariance $\mathbf{\Sigma}_j$. The mean vector $\mathbf{m}_j$ is given by

$$\mathbf{m}_j = [\mathbf{c}_j^T \ \ \mathbf{w}_j^T]^T \tag{3}$$

where $\mathbf{c}_j = E(\mathbf{X}_j)$ and $\mathbf{w}_j = E(\mathbf{Y}_j)$. The covariance matrix $\mathbf{\Sigma}_j$ is given by

$$\mathbf{\Sigma}_j = \begin{bmatrix} \mathbf{\Sigma}_{\mathbf{x}_j} \doteq Cov(\mathbf{X}_j, \mathbf{X}_j^T) & \mathbf{\Sigma}_{\mathbf{x}_j,\mathbf{y}_j} \doteq Cov(\mathbf{X}_j, \mathbf{Y}_j^T) \\ \mathbf{\Sigma}_{\mathbf{y}_j,\mathbf{y}_j} \doteq \mathbf{\Sigma}_{\mathbf{x}_j,\mathbf{y}_j}^T & \mathbf{\Sigma}_{\mathbf{y}_j} \doteq Cov(\mathbf{Y}_j, \mathbf{Y}_j^T) \end{bmatrix} \tag{4}$$

From the definition of the GM model parameters, the operation of a GMBFN can now be defined. A GMBFN of size M produces an output $\mathbf{f}_{GM}(\mathbf{x})$ for an input $\mathbf{x} \in R^{N_i}$ according to

$$\mathbf{f}_{GM}(\mathbf{x}) = \sum_{j=1}^{M} \left(\mathbf{w}_j + \mathbf{\Sigma}_{\mathbf{y}_j,\mathbf{x}_j}\mathbf{\Sigma}_{\mathbf{x}_j}^{-1}(\mathbf{x} - \mathbf{c}_j)\right) \cdot \Phi_j(\mathbf{x}) \tag{5}$$

where the GM basis function $\Phi_j(\mathbf{x})$ is defined by

$$\Phi_j(\mathbf{x}) = \frac{\pi_j|\mathbf{\Sigma}_{\mathbf{x}_j}|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{c}_j)^T\mathbf{\Sigma}_{\mathbf{x}_j}^{-1}(\mathbf{x} - \mathbf{c}_j)\right)}{\sum_{k=1}^{M} \pi_k|\mathbf{\Sigma}_{\mathbf{x}_k}|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{c}_k)^T\mathbf{\Sigma}_{\mathbf{x}_k}^{-1}(\mathbf{x} - \mathbf{c}_k)\right)}. \tag{6}$$

We see from the above expressions that the GMBFN can be viewed as an extended GRBFN where (i) the Gaussian basis functions are normalized by their sum, and (ii) there are additional weight terms that are linear in the input space.

For this paper, we used a simpler GMBFN model. Here, we assume in the underlying GM model that (i) $\Sigma_{\mathbf{x}_j,\mathbf{y}_j} = 0, \ \forall j$, and (ii) $\Sigma_{\mathbf{x}_j} = \sigma_j^2 \ I, \ \forall j$. In this case, the GMBFN function becomes

$$\mathbf{f}_{GM}(\mathbf{x}) = \sum_{j=1}^{M} \mathbf{w}_j \frac{\pi_j \exp\left(-\frac{||\mathbf{x} - \mathbf{c}_j||^2}{2\sigma_j^2}\right)}{\sum_{k=1}^{M} \pi_k \exp\left(-\frac{||\mathbf{x} - \mathbf{c}_k||^2}{2\sigma_k^2}\right)}. \tag{7}$$

and the GMBFN reduces to a Gaussian RBFN with normalized basis functions.

The GMBFN has a number of interesting and useful properties [4]. First, it implements the minimum mean-squared-error (MMSE) estimate $\mathbf{y}$ of an unknown desired output $\mathbf{Y}$ given an input $\mathbf{x}$ when the input and output variables follow the GM model described above. Secondly, the GMBFN can, unlike the RBFN or the MLP, exactly implement not only non-linear but also linear mappings. This feature may be useful in realizing a flexible signal processor which adapts to operate either linearly or non-linearly, depending on the input's statistical characteristics.

In our experiments, the GMBFNs were trained using a slightly modified version of the popular Expectation Maximization (EM) algorithm [6]. The original EM algorithm estimates the network parameters via a recursive procedure of i) parameter estimation (E step), and ii) maximization of an empirical likelihood function (M step). Our modified algorithm seeks to recursively minimize the MSE between the desired output and the network output instead of minimizing the log-likelihood. In this way, the GMBFN is tilted to be trained as a non-linear MMSE estimator regardless of the true joint distribution of the input and output within the constraint of the network structure.

3 Applications and Results

In this section, we present some of the results of applying the RBFN and the other two related networks discussed in the previous section to two problems; a non-Gaussian noise filtering problem and a PAM non-Gaussian channel equalization problem. For comparison purposes, linear FIR filter structures as well as conventional Gaussian RBFN structures were also tested in each application.

Performance in the simulations was measured by the normalized mean-squared-error (NMSE) between the desired output and the network output in the case of non-linear filtering, and both the NMSE and the symbol-error-rate (SER) in the case of PAM equalization.

3.1 Non-linear Filtering

The problem in this experiment was to filter out a non-Gaussian noise component $i(t)$ from a corrupted signal $y(t)$ to reconstruct the clean, desired signal $x(t)$. A 2000-sample-long training set was available, and the RBF nets were trained on this training set to operate as a non-linear filter.

The desired signal $x(t)$ in this example was a sinusoid given by

$$x(t) = \sin(2\pi \cdot 0.21t). \tag{8}$$

The corrupted signal $y(t)$ was given by $y(t) = x(t) + i(t)$ and the corrupting signal $i(t)$ was generated according to

$$i(t) = n(t) + 0.5n(t-1) - 0.5n(t-2) \tag{9}$$

where the $n(t)$ was a two-state Markov process whose state transition mechanism is depicted in Figure 1. In the figure, $\mathcal{N}(0,0.1)$ denotes the Gaussian distribution with zero mean and variance 0.1, and $\mathcal{E}(0,1.0)$ denotes the double-exponential distribution with zero mean and variance 1.0.

The desired signal $x(t)$ and the corrupted signal $y(t)$ are depicted in Figures 2(a) and (b), respectively. Training for the non-linear filters was done on the 2000-sample training set. Once trained, the filters were tested on separate test-sets where the non-linear filters were applied to the corrupted signal to filter out the unknown noise.

Figure 3 depicts the NMSE learning curves, tracked up to 2000 iterations, of four filter structures: they are FIR(10), RBF(5,10), HYB(5,5,10), and GMBFN(5,10) filters.[1] The linear/RBF hybrid structure gave the best results, with NMSE a little over -11.6 dB after convergence around 1000 iterations. The GMBFN was the second best, with NMSE=-11.3 dB, but after a somewhat slower convergence around 1200 iterations. The RBF(5,10), with NMSE around -9.5 dB after convergence around 1000 samples, was still about 2 to 3 dB better than the FIR filter. Figures 4(a), (b) and (c) depict the restored signals produced by FIR(10), RBF(5,10), HYB(5,5,10) filters, all after 2000 iterations.

[1]FIR(10) denotes a linear FIR filter with 10 taps. RBF(5,10) and GMBFN(5,10) denote, respectively, a GRBFN and a GMBFN with 5 inputs and 10 centers. HYB(5,5,10) denotes a hybrid filter consisting of an FIR(5) filter and an RBF(5,10) filter.

3.2 PAM Channel Equalization

Digital channel equalization is one area where the RBFN and its variants have been shown to be very useful [1],[3]. The results reported in [3], for example, suggest that RBFN-based equalizers provide significantly better equalization performance compared to conventional linear filter-based equalizers, especially when the channel's distortion mechanism has relatively short memory and/or the channel is significantly non-linear. In this work, we investigate the performance of RBFN-based equalizers in the presence of non-Gaussian noise.

Here, we present the results of one of our simulations. In this example, the desired-symbol signal $x(t)$ was an i.i.d. 4-PAM sequence, where the symbols were equi-probable and their values are -3, -1, 1, and 3. The channel was linear with non-Gaussian additive noise $n(t)$. The channel output $y(t)$ was given by

$$y(t) = 0.9x(t) + 0.5x(t-1) + n(t) \tag{10}$$

where the noise process $n(t)$ was i.i.d. and given by

$$n(t) \sim \begin{cases} \mathcal{N}(0, 0.01) & \text{with } p = 0.99 \\ \mathcal{E}(0, 0.10) & \text{with } p = 0.01 \end{cases} \tag{11}$$

Figure 5(a) and (b) depict the learning curves of NMSE and SER, respectively, for FIR equalizers with 5 and 10 taps, Figures 5(c) and (d) depict the NMSE and SER learning curves of the RBF(5,20), hybrid(5,5,20), and the GMBFN(5,20) equalizers. After training with upto 8000 samples, the RBF-based equalizers were from 2 to 6 dBs better in NMSE and up to an order better in SER compared to the FIR linear equalizers. We can see that the GMBFN showed the best results, followed by the hybrid network, the GRBFN, and the FIR filter equalizer.

4 Conclusions

The Gaussian RBFN and two related networks were investigated in non-linear filtering and equalization in non-Gaussian noise. Simulations results suggest that the Gaussian RBFN and its variants often perform significantly better than conventional linear structures in problems where the memory of the distortion mechanism was relatively short. However, getting similar performance improvements was difficult when the distortion mechanisms had a longer memory, requiring the RBF nets to have larger input dimensions. Finding ways to mitigate such dimensionality effects should be a topic of future investigations.

References

[1] S. A. Kassam and I. Cha, "Radial basis function networks in non-linear signal processing," *Proc. 27th Annual Conference on Information Sciences and Systems,* pp. 0122-1025, Asilomar, CA, November 1993.

[2] S. Elanayar V.T. and Y. C. Shin, "Radial basis function neural network for approximation and estimation of non-linear stochastic dynamic systems," *IEEE Trans. on Neural Networks,* vol.5, no.4, July 1994, pp. 594-603

[3] I. Cha and S. A. Kassam, "Channel equalization using adaptive complex radial basis function networks," *IEEE Journal of Selected Areas in Communications*, vol.13, no.1, January 1995, pp. 122-131.

[4] I. Cha and S. A. Kassam, "Gaussian-mixture basis function networks for non-linear signal processing," *Proc. IEEE Workshop on Non-Linear Signal Processing,* vol. 1, pp. 74-77, Halkidiki, Greece, June 1995.

[5] I. Cha, "Minimum MSE Channel Equalization using Extended Radial Basis Function Networks," to be presented at the IASTED Conference on Signal and Image Processing, December 1997.

[6] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review,* vol. 26, April 1984, pp. 195-239.

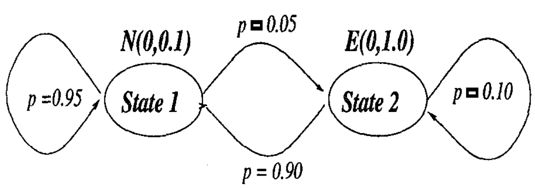
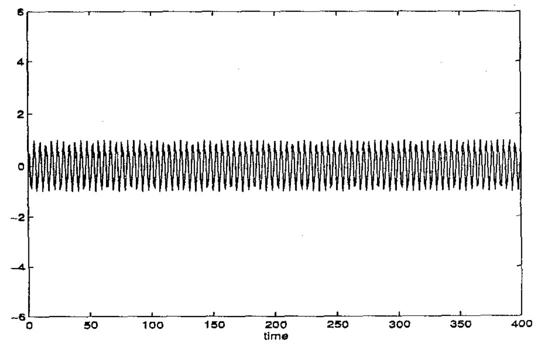
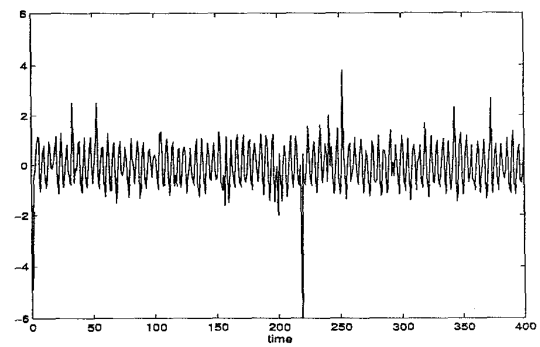


Figure 1: **Case 1** State transition diagram of the noise $n(t)$



(a)



(b)

Figure 2: **Case 1** (a) Desired signal $x(t)$, (b) Corrupted signal $y(t)$.

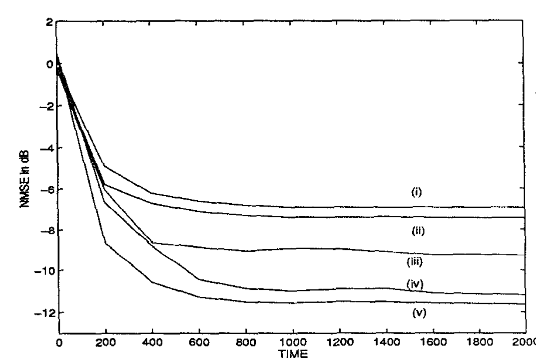
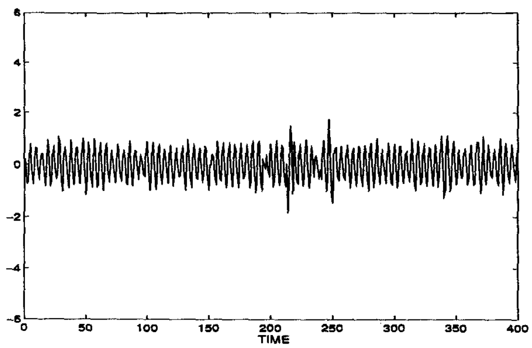
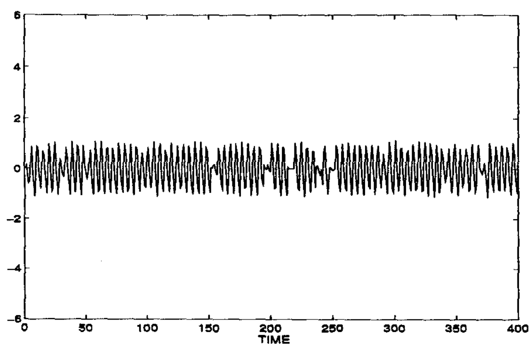


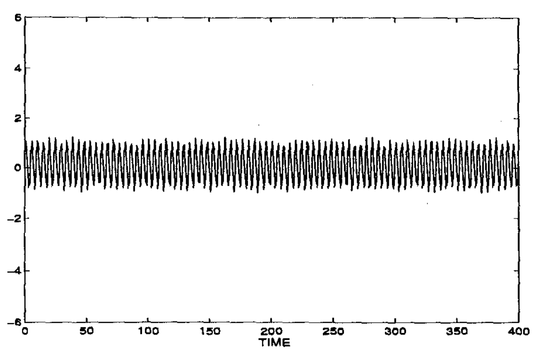
Figure 3: **Case 1** NMSE learning curves of various filters: (i) FIR(5), (ii) FIR(10), (iii) RBF(5,10), (iv) GMBFN(5,10), (v) HYB(5,5,10)



(a)

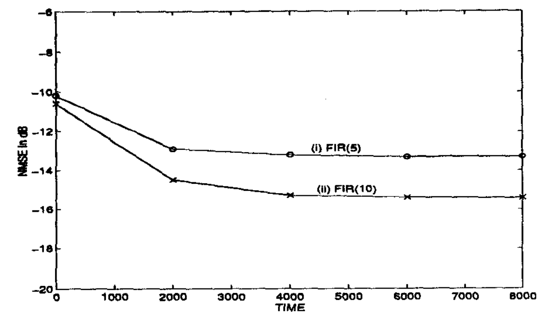


(b)

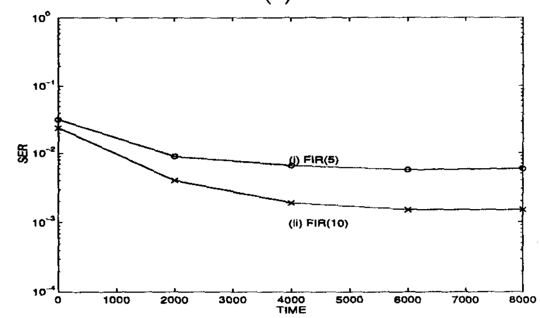


(c)

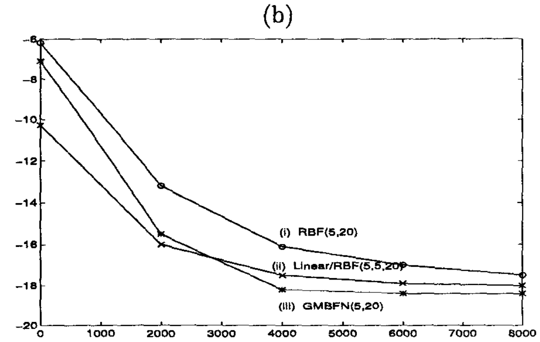
Figure 4: **Case 1** Restored signals: (a) by FIR(10), (b) by RBF(5,10), and (c) by HYB(5,5,10), all after 2000 iterations.



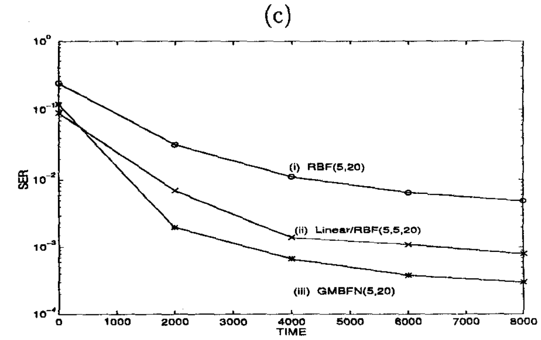
(a)



(b)



(c)



(d)

Figure 5: **Case 2** Learning curves for (a): NMSE of FIR equalizers, (b) SER of FIR equalizers, (c) NMSE of RBF equalizers, (d) SER of RBF equalizers