

# Channel Equalization Using Dynamic Fuzzy Neural Networks

M. J. Er, F. Liu, and M. B. Li

## Abstract

Channel equalization is a major method for reducing distortion and interference effects on a communication channel. In this paper, channel equalization using soft computing methods is attempted. To be more specific, Dynamic Fuzzy Neural Networks (DFNN) which combines fuzzy rules and neural networks is adopted. The DFNN is functionally equivalent to a Takagi-Sugeno-Kang (TSK) fuzzy system possessing the learning ability of a Radial Basis Function (RBF) neural network. The hidden neurons (rules) of the DFNN can be added and pruned dynamically during the training process based on the significance of each neuron to achieve a compact topology structure. Simulation studies demonstrate that the performance of the DFNN-based equalizer is superior to some other existing equalizers in terms of Bit Error Rate (BER).

**Keywords:** Fuzzy logic, Neural networks, DFNN, Channel equalization, Minimal Resource Allocation Network (MRAN).

## 1. Introduction

It is well known that in high speed digital communication systems, the communication channel will invariably distort the transmitted signals in both amplitude and phase thereby causing distortion in the received signals. Figure 1 shows a standard base-band discrete model of a communication system, where the channel input signal,

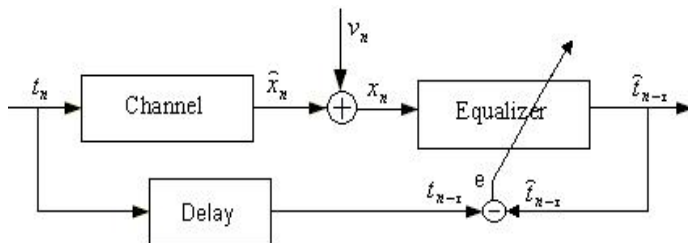


Figure 1. Discrete model of a communication system.

Corresponding Author: M. J. Er is with the School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue Rd., Singapore, 639798.

E-mail: emjer@ntu.edu.sg.

Manuscript received 22 Dec. 2008; revised 16 Mar. 2009; accepted 25 Mar. 2009.

is passed through a channel which is of linear or nonlinear dynamics.

The channel output  $x_n$  is given by

$$x_n = \sum_{i=1}^{n_h-1} h_i t_{n-i} + v_n \quad (1)$$

where  $n_h$  is the channel order. As depicted in Figure 1, the channel noise-free output  $\hat{x}_n$  is corrupted by additive zero-mean Gaussian noise  $v_n$  to produce the equalizer input signal  $x_n$ . After equalization, the equalizer's output signal  $\hat{t}_{n-\tau}$  is compared with the delayed channel input signal  $t_{n-\tau}$  and the error between these two signals,  $e$  is used to adjust the parameters associated with the equalizer. The purpose of the equalizer is to reconstruct the transmitted signal  $t_{n-\tau}$  based on the received signal sequence  $X_n = [x_n, x_{n-1}, \dots, x_{n-m+1}]^T$  where  $\tau$  is the equalizer's decision delay and  $m$  is the input dimension of the equalizer.

Adaptive channel equalization is a major issue in digital communications. Many adaptive filters can be used to improve performance of digital communication systems [1] [2]. Use of adaptive equalizers in digital communication system can significantly improve the performance of the overall system. A novel scheme with the adaptive filtering and the pre-conditioned conjugate gradient algorithm for channel equalization was proposed in [3]. The authors of [4] proposed a channel equalizer based on Adaptive Filtering with Averaging (AFA). The main advantages of AFA are that it has high convergence rate comparable to that of the recursive least squares equalizer and has low computational complexity. Blind equalization is a particularly useful type of equalization when training sequences are undesirable. Blind equalization schemes such as the Tricestrum Equalization Algorithm (TEA) [5] and the maximum likelihood joint data and channel estimation algorithm [6] have been developed for linear channel. Blind equalizers used for nonlinear channels were developed in [7] and [8].

By virtue of the excellent mapping and classification ability of neural networks [9], many types of neural networks have been applied for equalization problems, such as Multi-Layer Perceptions (MLP), Radial Basis Function Neural Networks (RBFNN) and Recurrent Neural Networks (RNN) [10]-[16]. The authors of [10] developed an adaptive equalizer using MLP to overcome

channel nonlinearities and additive noise correlation. They also investigated a RBFNN equalizer to reconstruct binary signals in a dispersive channel and showed that the RBFNN nonlinear equalizer can realize optimal equalization and were beneficial in practical implementations [11] [12]. RNN equalizer has gained great attention because of its feedback property and was developed by Kechriotis et al. [13] and Parisi et al. [16], respectively. Kechriotis et al. proposed an adaptive RNN equalizer for linear and nonlinear channels [13]. Another fast adaptive RNN digital equalizer was presented by Parisi et al. [16], which was superior to Kechriotis' RNN equalizer in terms of Bit Error Rate (BER) and the speed of convergence. Kumar et al. [14] presented a RBF equalizer using Minimal Resource Allocation Network (MRAN) and evaluated performance of the proposed equalizer in different channels for 2-PAM and 4-QAM signals. The equalizer using the Function Link Artificial Neural Networks (FLANN) was developed by Weng et al. [17] and Yen et al. [18]. The performance of the FLANN equalizer was compared with that of an MLP equalizer and linear least-mean-square-based equalizer in several channels for QAM signals. The nonlinear channel equalizers based on self-constructing fuzzy neural network (SCFNN) was designed and developed by Weng et al. [19]. It has been demonstrated that the SCFNN-based digital channel equalizer possesses the ability to recover the channel distortion effectively. Another self-constructing recurrent fuzzy neural network (SCRFNN)-based digital channel equalizer was proposed in [20]. Growing and pruning RBF (GAP-RBF) network which is similar to the MRAN algorithm was applied to solve channel equalization problem in [15]. The GAP-RBF is able to determine an appropriate number of hidden neurons automatically. Using the growing and pruning criteria, the number of hidden-layer neurons is dynamically adjusted to achieve a compact network structure.

In the past few years, fuzzy logic has gained great attention in various industrial applications because of its ability to approximate any continuous function on a compact set to any accuracy. It is well known that fuzzy logic incorporates simple IF-Then rules to model and control an engineering system and it typically requires expert experiences in the design. However, any designer will encounter great difficulty in conventional fuzzy system design when the system is too complicated to extract an appropriate number of fuzzy rules. Recently, Dynamic Fuzzy Neural Networks (DFNN) which combines fuzzy logic and neural networks was proposed by Wu and Er [21]. By leveraging on the reasoning ability of fuzzy logic and the learning ability of neural networks, the DFNN has been applied to solve a lot of interesting engineering problems. The key idea of the DFNN is that

the system starts with no hidden units and it dynamically adds and deletes neurons according to their significance to system performance. A parsimonious structure is achieved by virtue of the self-adaptive learning algorithm. Many interesting applications of the DFNN have been accomplished since it was proposed [22]-[24]. However, it is not clear whether the DFNN is suitable for handling channel equalization problems and if so, how its performance will be compared with some state-of-the-art methods. In this paper, the DFNN is applied to the channel equalization problem and its performance is evaluated in several channel models. Simulation results show that the DFNN equalizer is a useful and efficient method for linear and nonlinear channel equalization

This paper is organized as follows. Section 2 reviews the DFNN architecture. Section 3 presents the DFNN learning algorithm. Section 4 compares the performance of the DFNN equalizer with the Bayesian, MRAN, RNN and other equalizers. Section 5 discusses the simulation results. Section 6 concludes the paper.

## 2. Architecture of Dynamic Fuzzy Neural Networks

Figure 2 depicts the architecture of DFNN. The architecture of the DFNN is made up of five layers:

Layer 1: Input layer. Each node represents an input linguistic variable.

Layer 2: Each node represents a membership function (MF) which is in the form of a Gaussian function:

$$\mu_{ij}(x_i) = \exp\left[-\frac{(x_i - c_{ij})^2}{\sigma_j^2}\right] \quad i = 1, 2, \dots, m, \quad j = 1, 2, \dots, u \quad (2)$$

where  $\mu_{ij}$  is the  $j$ th membership function of  $x_i$ ,  $c_{ij}$  is the center of the  $j$ th Gaussian membership function of  $x_i$ ,  $\sigma_j$  is the width of the  $j$ th Gaussian membership function,  $m$  is the number of input variables and  $u$  is the number of membership functions.

Layer 3: Each node represents a possible IF-part of a fuzzy rule. If the T-norm operator is chosen as multiplication to calculate each rule's firing strength, the output of the  $j$ th rule  $R_j$  is given by:

$$R_j = \exp\left[-\frac{\sum_{i=1}^m (x_i - c_{ij})^2}{\sigma_j^2}\right] = \exp\left[-\frac{\|(X - C_j)\|^2}{\sigma_j^2}\right] \quad j = 1, 2, \dots, u \quad (3)$$

Layer 4: In this layer, the outputs from the previous layer are normalized to the interval  $[0, 1]$ , i.e.

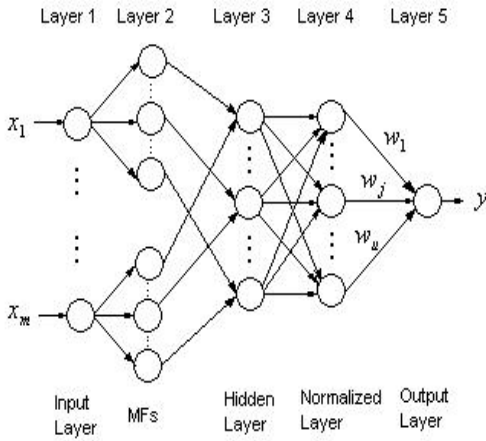


Figure 2. Architecture of DFNN.

$$a_j = \frac{R_j}{\sum_{k=1}^u R_k} = \frac{\exp\left[-\frac{\|X - C_j\|^2}{\sigma_j^2}\right]}{\sum_{k=1}^u \exp\left[-\frac{\|X - C_k\|^2}{\sigma_k^2}\right]} \quad j = 1, 2, \dots, u \quad (4)$$

Layer 5: Each node in this layer represents an output variable which is the weighted sum of the incoming signals, i.e.

$$y(X) = \sum_{j=1}^u w_j a_j \quad (5)$$

where  $y$  is the value of an output variable and  $w_j$  is the connection weight of each rule.

For the TSK model:

$$w_j = k_{j0} + k_{j1}x_1 + \dots + k_{jm}x_m \quad j = 1, 2, \dots, u \quad (6)$$

where  $m$  is the number of input variables.

### 3. Learning Algorithm of DFNN

#### A. Criteria of Growing Neurons

The learning algorithm flowchart of DFNN is depicted in Figure 3. Two criteria of growing neurons are defined.

- System Error: For each observation  $(P_i, t_i)$  where  $P_i$  is the input vector and  $t_i$  is the desired output, calculate the overall DFNN output  $y_i$  on the existing structure according to (5)

Define

$$\|e_i\| = \|t_i - y_i\| \quad (7)$$

as system error.

- Accommodation Boundary For each observation  $(P_i, t_i)$ , calculate the distance  $d_i(j)$  between the observation  $P_i$  and the center  $C_j$  of existing RBF units.

Denote

$$d_i(j) = \|P_i - C_j\| \quad j = 1, 2, \dots, u \quad (8)$$

where  $u$  is the number of existing RBF units.

Find

$$d_{\min} = \arg \min(d_i(j)) \quad j = 1, 2, \dots, u \quad (9)$$

If

$$\|e_i\| > k_e \quad (10)$$

$$d_{\min} > k_d \quad (11)$$

where  $k_e$  is chosen a priori according to the desired accuracy of the DFNN and  $k_d$  is the effective radius of the accommodation boundary, an RBF unit should be considered. We choose

$$k_e = \max[e_{\max} \times \beta^i, e_{\min}] \quad (12)$$

$$k_d = \max[d_{\max} \times \gamma^i, d_{\min}] \quad (13)$$

where  $e_{\max}$  is the predefined maximum error,  $e_{\min}$  is the desired accuracy of the DFNN output,  $\beta$  is a convergence constant (typically,  $\beta = 0.9 \sim 0.95$ ),  $d_{\max}$  is the largest length of the input space,  $d_{\min}$  is the smallest length of interest and  $\gamma$  is a decay constant (in general, we choose  $\gamma = 0.94 \sim 0.998$ ).

After a new neuron has grown, we allocate the new RBF unit with centre,  $C_i$  and width,  $\sigma_i$  as follows:

$$C_i = P_i \quad (14)$$

$$\sigma_i = k \times d_{\min} \quad (15)$$

where  $k$  is an overlap factor that determines the overlap of responses of RBF units. In general, we choose  $k = 1.05 \sim 1.2$ .

#### B. Weight Adjustment

At any time, the  $n$ th observation enters the DFNN and all these  $n$  data are “memorized” by the DFNN in the input data matrix  $IN$  and the output data matrix  $OUT$  respectively, based upon which the weights are determined. The input data matrix is given by

$$IN_{m \times n} = \begin{bmatrix} p_{11} & \dots & p_{1n} \\ \vdots & \vdots & \vdots \\ p_{m1} & \dots & p_{mn} \end{bmatrix} \quad (16)$$

and the output data matrix is given by

$$OUT_{1 \times n} = (t_1, \dots, t_n) \quad (17)$$

where  $m$  is the number of input variables.

If  $u$  RBF units are generated according to the aforementioned criteria of growing neurons for these  $n$  observations, the output of normalized nodes can be obtained according to (4)

Denote

$$\phi_{u \times n} = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \vdots & \vdots \\ a_{u1} & \dots & a_{un} \end{bmatrix} \quad (18)$$

For any input  $P_j(p_{1j}, p_{2j}, \dots, p_{mj})$ , the system output  $y_j$

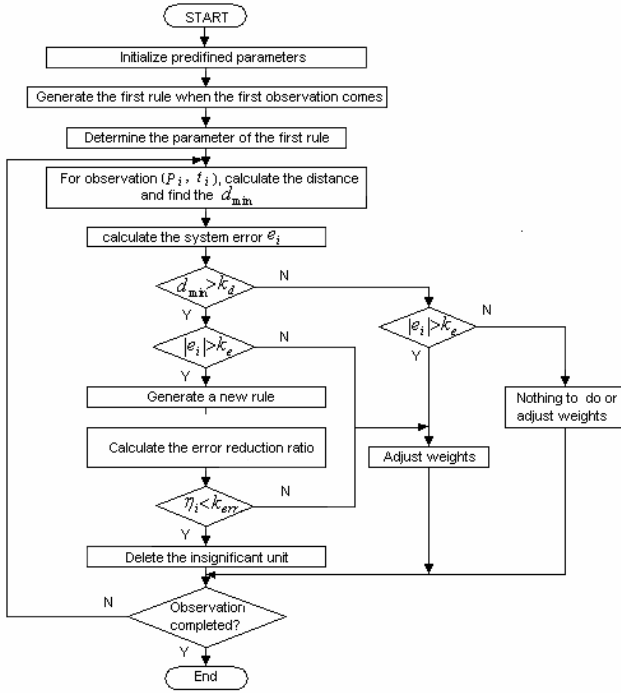


Figure 3. Learning algorithm of DFNN.

can be derived as follows:

$$y_j = \sum_{i=1}^u w_i \times \phi_i \quad (19)$$

Rewriting (19) in a more compact form, we have

$$Y = W \Psi \quad (20)$$

where

$$W = [k_{10}, \dots, k_{u0}, k_{11}, \dots, k_{u1}, k_{1m}, \dots, k_{um}] \quad (21)$$

$$\Psi = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \vdots & \vdots \\ a_{u1} & \dots & a_{un} \\ a_{11}p_{11} & \dots & a_{1n}p_{1n} \\ \vdots & \vdots & \vdots \\ a_{u1}p_{11} & \dots & a_{un}p_{1n} \\ \vdots & \vdots & \vdots \\ a_{11}p_{m1} & \dots & a_{1n}p_{mn} \\ \vdots & \vdots & \vdots \\ a_{u1}p_{m1} & \dots & a_{un}p_{mn} \end{bmatrix} \quad (22)$$

Our objective is the following: Given  $\Psi_{(m+1)u \times n}$  and  $OUT_{1 \times n}$  related by

$$Y = W \times \Psi \quad (23)$$

$$\tilde{E} = \|OUT - Y\| \quad (24)$$

find an optimal coefficient vector  $W^*$  such that the error energy  $\tilde{E}^T \tilde{E}$  is minimized. This problem can be solved by the linear least squares (LLS) method by approximating

$$OUT = W^* \times \Psi \quad (25)$$

The optimal  $W^*$  is in the form of

$$W^* = OUT \times \Psi^+ \quad (26)$$

where  $\Psi^+$  is the pseudo-inverse of  $\Psi$

$$\Psi^+ = (\Psi^T \Psi)^{-1} \Psi^T \quad (27)$$

### C. Criteria of Pruning Neurons

In order to realize a compact network structure, inactive hidden neurons should be detected and removed during the learning progress.

The error reduction ratio method presented in [21] is used to calculate the error reduction ratio matrix  $ERR = (\delta_1, \delta_2, \dots, \delta_u) \in R^{(m+1) \times u}$ . If

$$\eta_i = \sqrt{\frac{\delta_i^T \delta_i}{m+1}} < k_{err} \quad (28)$$

where  $k_{err}$  is a predefined threshold,  $m+1$  is the row number of matrix  $ERR$ , the  $i$ th RBF unit should be deleted.

## 4. Simulation Results

The performance of the DFNN equalizer for 2-PAM signals is evaluated for three different channel models used in [13] and [16]. In Example 1 and Example 2, the simulation environment is set the same as that of [16] in order to have a fair comparison of the results. A total of  $10^6$  testing data are used to calculate the BER after the equalizer is trained at different signal-to-noise ratio (SNR).

Example 1: In this example, the third-order non-minimum-phase channel model is used to evaluate the performance of the DFNN equalizer. The channel transfer function used in [13] and [16] is given by

$$H(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2} \quad (29)$$

The input dimension of the DFNN equalizer is set to  $m=1$  and the equalizer decision delay is  $\tau=1$ . The DFNN network parameters are set as:  $d_{\max}=4$ ,  $d_{\min}=0.3$ ,  $e_{\max}=1$ ,  $e_{\min}=0.02$ ,  $k_{err}=0.002$ ,  $k=1.1$ ,  $\beta=0.9$ ,  $\gamma=0.97$ .

The activation functions are chosen to be Gaussian functions for the MRAN, GAP-RBF and RBFNN equalizers and hyperbolic tangent function is used for the RNN equalizer. The RNN equalizer of [13] uses three units in the simulation study. The DFNN equalizer is trained with 300 samples at different SNR and Figure 4 is the DFNN equalizer output at 10dB SNR. Seven fuzzy rules have been generated by the DFNN equalizer at the end of the training process shown in Figure 5. The distribution of neurons is shown in Table 1 and the membership functions of the input variable are shown in Figure 6. Figure 7 shows plots of BER versus SNR for the

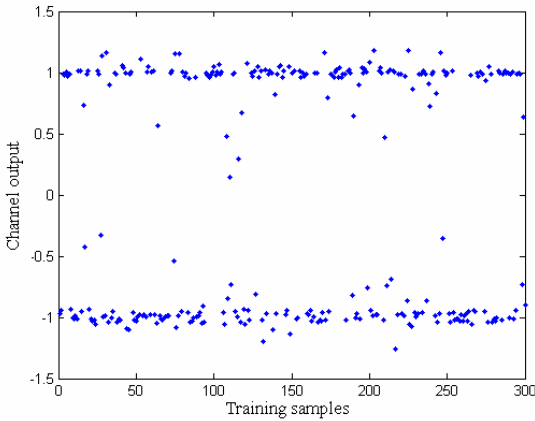


Figure 4. Equalizer output (Example 1).

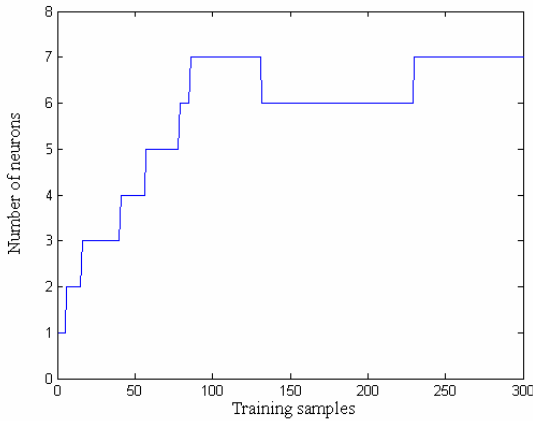


Figure 5. Generation of fuzzy rules (Example 1).

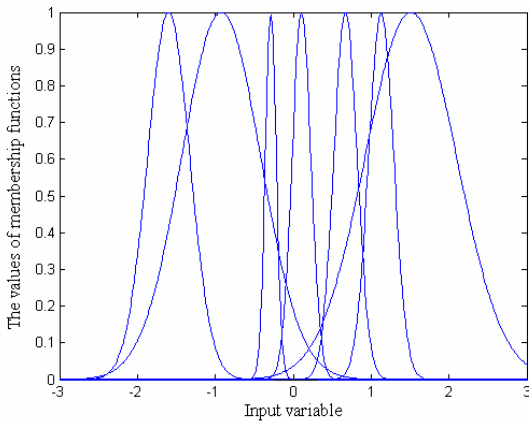


Figure 6. Membership functions of input variable (Example 1).

Bayesian, the MRAN, the RNN, the DFNN, the GAP-RBF and the RBFNN equalizers. As shown in Figure 7, the Bayesian equalizer (star line) attains the best BER performance. It can be seen clearly that the DFNN equalizer (circle line) always produces superior performance than the MRAN equalizer (dot line), the

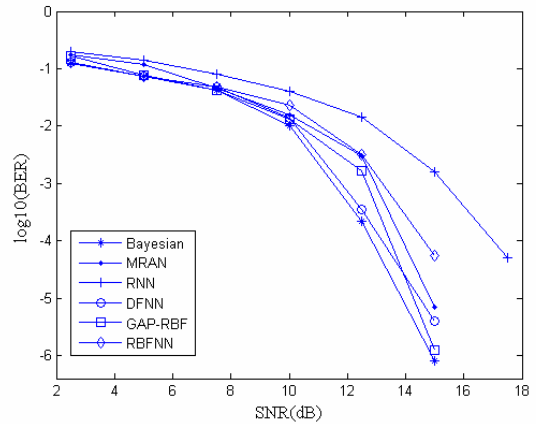


Figure 7. Error probabilities \*: Bayesian. •: MRAN. +: RNN

o: DFNN. □GAP-RBF. ◆: RBFNN equalizers (Example1).

RNN equalizer (plus line) and the RBFNN equalizer (diamond line). It should be highlighted that for the large SNR, the GAP-RBF performs better than the DFNN, while for small SNR, the DFNN performs better than the GAP-RBF.

The fuzzy rules generated by the DFNN are:

Rule 1: If  $x$  is  $A(0.9,0.7)$ , then  $t = -1.7 - 0.8x$ .

Rule 2: If  $x$  is  $A(1.5,0.8)$ , then  $t = 0.1 + 0.6x$ .

Rule 3: If  $x$  is  $A(0.1,0.2)$ , then  $t = 0.5 + 5x$ .

Rule 4: If  $x$  is  $A(1.1,0.2)$ , then  $t = 1.3$ .

Rule 5: If  $x$  is  $A(0.7,0.2)$ , then  $t = 0.9 + 0.5x$ .

Rule 6: If  $x$  is  $A(-0.3,0.1)$ , then  $t = -0.5 + 0.7x$ .

Rule 7: If  $x$  is  $A(-1.6,0.4)$ , then  $t = -2.8 - 0.9x$ .

Example 2: In this case, a more complicated channel model is used to evaluate the performance of the DFNN equalizer. The nonlinear channel transfer function used in [13] and [16] is given by

$$H(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2} \quad (30)$$

$$x_n = \hat{x}_n + 0.2\hat{x}_n^2 + v_n \quad (31)$$

where  $\hat{x}_n$  is the linear noise-free channel output and  $v_n$  is the zero mean Gaussian white noise. The input dimension of the DFNN equalizer is set to  $m=1$  and the equalizer decision delay is  $\tau=1$ . The DFNN network parameters are set as:  $d_{\max}=4$ ,  $d_{\min}=0.3$ ,  $e_{\max}=1$ ,  $e_{\min}=0.02$ ,  $k_{err}=0.002$ ,  $k=1.1$ ,  $\beta=0.9$ ,  $\gamma=0.97$ .

The DFNN equalizer is trained at different SNR with 500 training samples. Figure 8 and Figure 9 show the generation of fuzzy rules and membership functions of the input variable respectively.

The fuzzy rules generated by the DFNN are:

Rule 1: If  $x$  is  $A(-0.7,0.7)$ , then  $t = -0.9 + 0.2x$ .

Rule 2: If  $x$  is  $A(2.1,1.0)$ , then  $t = 4.9 - 1.7x$

Rule 3: If  $x$  is  $A(0.2,0.1)$ , then  $t = 2.6 - 7.5x$ .

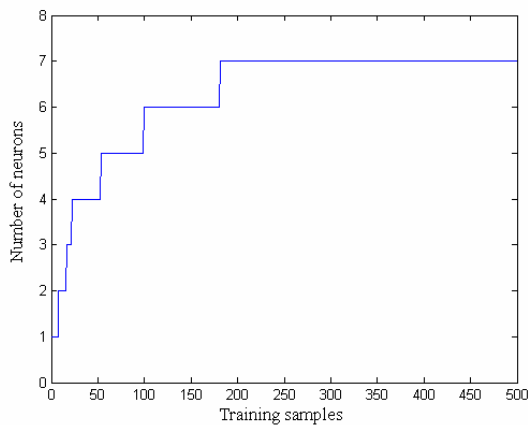


Figure 8. Generation of fuzzy rules (Example 2).

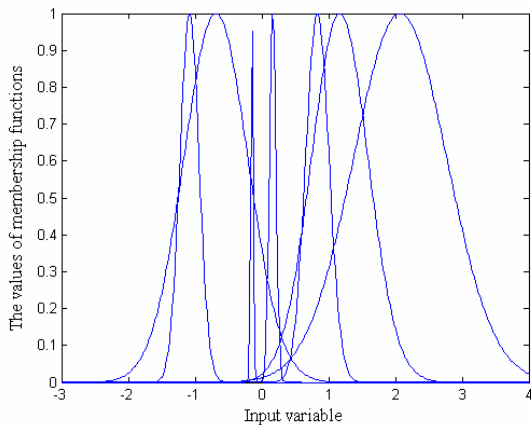


Figure 9. Membership functions of input variable (Example 2).

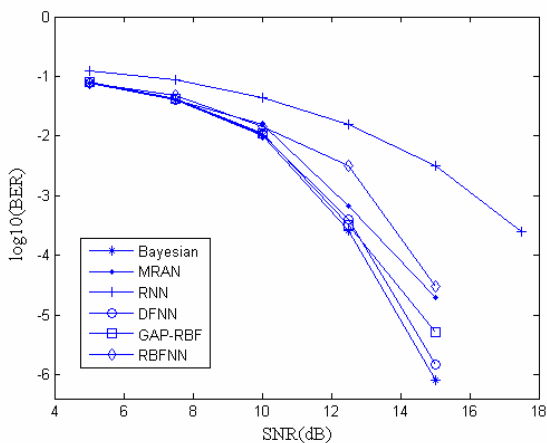


Figure 10. Error probabilities. \*: Bayesian. •: MRAN. +: RNN  
o: DFNN. □ GAP-RBF. ◆: RBFNN equalizers. (Example2).

- Rule 4: If  $x$  is  $A(1.2,0.6)$ , then  $t = 2.7 - 2.3x$ .
- Rule 5: If  $x$  is  $A(-0.2,0.02)$ , then  $t = -1.0 + 1.3x$ .
- Rule 6: If  $x$  is  $A(0.8,0.3)$ , then  $t = -1.3 + 2.1x$ .
- Rule 7: If  $x$  is  $A(-1.1,0.2)$ , then  $t = -0.5 + 0.4x$ .

After the training process, a total of one million test data at various SNRs were used for the BER calculation. A comparison of BER results with five other equalizers is shown in Figure 10. It can be seen from the figure that the DFNN equalizer attains better BER performance than other equalizers, except the Bayesian equalizer.

Example 3: The nonlinear channel model used in [13] and [14] is chosen for simulation study. The channel transfer function is given by

$$H(z) = 1 + 0.7z^{-1} \tag{32}$$

$$x_n = \hat{x}_n + \hat{x}_n^2 + 0.7\hat{x}_n^3 + 0.5\hat{x}_n^4 + v_n \tag{33}$$

The DFNN equalizer is trained with 50 training samples at 10dB SNR. The growth of fuzzy rules during training is depicted in Figure 11. Figure 12 shows membership functions of the input variable.

A total of one million test data at various SNRs are used for the BER calculation. A comparison between the Bayesian, the DFNN, the MRAN, the GAP-RBF and the RBFNN equalizers in terms of BER is shown in Figure 13. It can be seen from the figure that the DFNN equalizer is superior to the MRAN, the GAP-RBF and the RBFNN equalizers.

The fuzzy rules generated by the DFNN are:

Rule 1: If  $x$  is  $A(-0.2,1.5)$ , then  $t = -0.8 - 2.7x$ .

Rule 2: If  $x$  is  $A(12.2,11.7)$ , then  $t = 0.9 + 0.01x$

Rule 3: If  $x$  is  $A(0.2,0.3)$ , then  $t = 1.4 - 27.7x$ .

Rule 4: If  $x$  is  $A(-0.6,0.3)$ , then  $t = 15 - 20.0x$ .

## 5. Conclusions

From the simulation results, for example 1, for the large SNR, the GAP-RBF performs better than the DFNN, while for small SNR, the DFNN performs better than the GAP-RBF, and for example 2 and 3 we can see that the DFNN equalizer can obtain better equalization performance than other equalizers, except the Bayesian equalizer. This is because the Bayesian equalizer uses

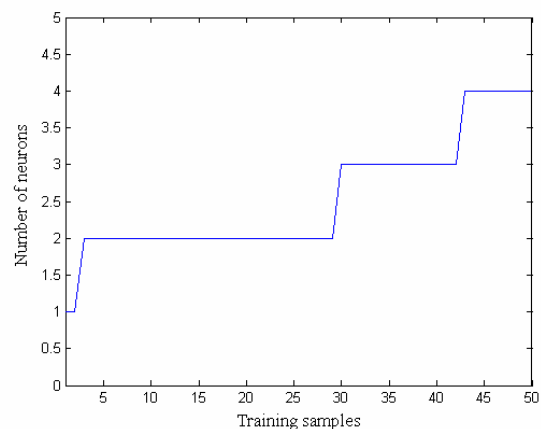


Figure 11. Generation of fuzzy rules (Example 3).

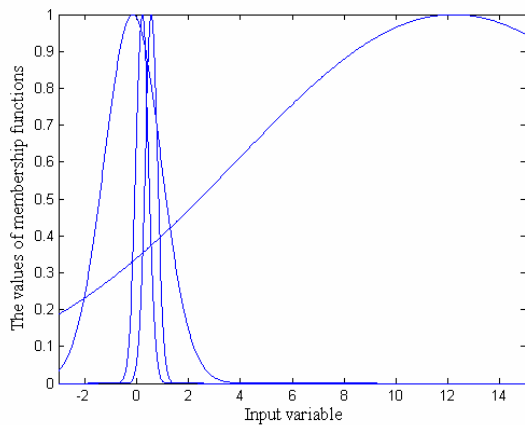


Figure 12. Membership functions of input variable (Example 3).

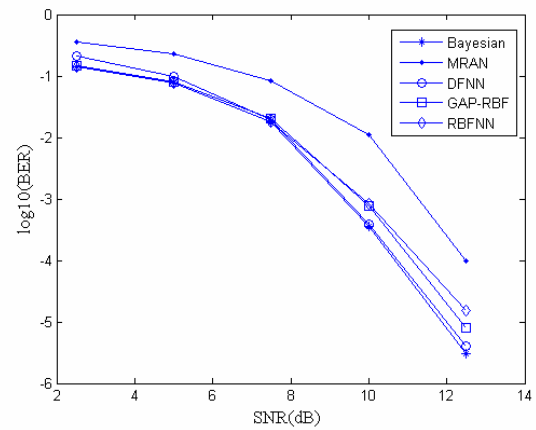


Figure 13. Error probabilities. \*: Bayesian. •: MRAN. o:DFNN. □: GAP-RBF. ◆: RBFNN equalizers (Example3).

Table 1. Distribution of neurons (Example 1).

neuron	1	2	3	4	5	6	7
Center $C$	-0.9240	1.5101	0.1060	1.1294	0.6714	-0.2877	-1.5994
Width $\sigma$	0.7170	0.8290	0.1711	0.2279	0.2122	0.0986	0.3841

Table 2. Complexities of equalizers.

	Algorithms	Neurons	Number of Data		equalization time	
			training	testing	training(s)	testing(ms/sample)
Example 1	DFNN	7	300	$10^6$	0.206	0.119
	GAP-RBF	6	300	$10^6$	0.389	0.083
	MRAN	8	300	$10^6$	0.732	0.107
	RBFNN	8	300	$10^6$	0.089	0.132
	Bayesian	8	—	$10^6$	—	0.094
Example 2	DFNN	7	500	$10^6$	0.234	0.104
	GAP-RBF	5	500	$10^6$	0.577	0.072
	MRAN	6	500	$10^6$	1.307	0.083
	RBFNN	8	500	$10^6$	0.112	0.136
	Bayesian	8	—	$10^6$	—	0.094
Example 3	DFNN	4	50	$10^6$	0.031	0.108
	GAP-RBF	4	50	$10^6$	0.108	0.061
	MRAN	4	50	$10^6$	0.203	0.061
	RBFNN	4	50	$10^6$	0.018	0.068
	Bayesian	4	—	$10^6$	—	0.049

noise-free channel states as its centers and a priori probability as its desired output, which results in an optimal solution. The limitation of the Bayesian method is that the channel model needs to be known exactly, which is difficult to achieve in real-world applications. The RBFNN proposed in [12] uses a supervised  $\kappa$ -means clustering procedure to eliminate noise effects so that the RBF centers can converge to the desired ones. The properties of the clustered centers will directly determine the equalization performance.

For Example 1 and Example 2, the channel models are complicated which is the reason why the RBFNN equalizer cannot perform better than the DFNN, GAP-RBF and MRAN equalizers in terms of BER. For Example 3, the channel model is simple and the RBFNN equalizer can obtain almost the same equalization accuracy as the DFNN and the GAP-RBF equalizers. During training, the RBFNN weights linking the hidden layer to the output layer are adjusted using the least mean square (LMS) algorithm. From Table 2, it can be seen that the RBFNN can achieve the fastest learning speed in all methods by virtue of simple training algorithms. The DFNN, GAP-RBF and MRAN equalizers are all self-constructing neural networks and they use growing and pruning criteria to search the compact network structure. At the end of the training process, they generated almost the same number of hidden neurons as the Bayesian method which is the optimal solution (Table 2). For BER performance, the DFNN equalizer is the best in these three equalizers because it combines the reasoning ability of fuzzy system with the learning ability of neural networks. The LLS method used to determine the output weights enables the DFNN achieve global generalization property rapidly. Though the GAP-RBF and the MRAN equalizers can achieve compact network structure, the output weights are modified using the Extended Kalman Filter (EKF) method which leads to more computational time and cost than the DFNN equalizer during training (Table 2). It should be highlighted from Table 2 that there is not much difference between the equalizer time of all equalizers. This is because the equalizer time (testing time) is mainly affected by the number of hidden neurons.

## 6. Conclusions

In this paper, channel equalization with 2-PAM signals has been successfully attempted by using the DFNN. Performance evaluation of the DFNN equalizer has been carried out using several channel models with increasing complexity. Simulation results show that the DFNN equalizer is a useful and efficient method for linear and nonlinear channel equalization. The DFNN equalizer has been applied to three different Finite Impulse Response

(FIR) channel models, how to use the DFNN equalizer to handle Infinite Impulse Response (IIR) channel case is under investigation. Further enhancement of our equalizer is still under investigation. Nonlinearities is a very serious problem for channel communication. Nonlinear channel is coped with using a pass-band filter in front of the receiver in order to somewhat suppress harmonics. How to compare our equalizer with the case of supervised equalization using an adaptive FIR filter is a promising research for our future research.

## References

- [1] S. U. H. Oureshi, "Adaptive equalization," In *Proc. of IEEE*, vol. 73, no. 9, pp. 1349-1387, 1986.
- [2] E. A. Lee and D. G. Messerschmitt, *Digital Communication*, Boston, MA: Kluwer, 1994.
- [3] Soni, Robert A., Jenkins, W. Kenneth, "Channel equalization with adaptive filtering and the pre-conditioned conjugate gradient algorithm," In *Proc. of IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 2284-2287, 1997.
- [4] Iliev, G., Kasabov, N., "Channel Equalization Using Adaptive Filtering with Averaging," In *Proc. of the Joint Conference on Information Sciences*, vol. 5, no. 2, pp. 870-873, 2000.
- [5] D. Hatzinakos and C. L. Nikias, "Blind equalization using a tricepstrum based algorithm," *IEEE Trans. Commun.*, 1991.
- [6] N. Seshadri, "Joint data and channel estimation using fast blind trellis search techniques," In *Proc. of Globecom*, pp. 1659-1663, 1990.
- [7] Rao, K. D., Plotkin, E.I., Swamy, M. N. S., "Adaptive blind equalization of nonlinear channels and chaotic systems using coupled EKF and RLS estimator," *IETE Journal of Research*, vol. 53, no. 3, pp. 181-192, 2005.
- [8] Han, S, Pedrycz, W, Han, C., "Nonlinear channel blind equalization using hybrid genetic algorithm with simulated annealing," *Mathematical and Computer Modelling*, vol. 41, pp. 697-709, 2005.
- [9] G. B. Huang, Y. Q. Chen and H. A. Babri, "Classification ability of single hidden layer feed-forward neural networks," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 799-801, 2000.
- [10] 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, no. 2, pp.107-119, 1990.
- [11] S. Chen, G. J. Gibson, C. F. N. Cowan and P. M. Grant, "Reconstruction of binary signals using an adaptive radial-basis function equalizer," *Signal Processing*, vol. 22, no. 1, pp. 77-93, 1991.
- [12] S .Chen, B. Mulgrew, and P. M. Grant, "A cluster-

- ing technique for digital communications channel equalization using radial basis function networks,” *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 570-579, 1993.
- [13] G. Kechriotis, E. Zervas, and E. S. Manolakos, “Using recurrent neural networks for adaptive communication channel equalization,” *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 267-278, 1994.
- [14] P. C. Kumar, P. Saratchandran, and N. Sundararajan, “Minimal radial basis function neural networks for nonlinear channel equalization,” *IEEE Proceedings, Vision, Image and Signal Processing*, vol. 147, no. 5, pp. 428-435, 2000.
- [15] M. B. Li, G. B. Huang, P. Saratchandran, and N. Sundararajan, “Performance evaluation of GAP-RBF network in channel equalization,” *Neural Processing Letters*, vol. 22, no. 2, pp. 223-233, 2005.
- [16] R. Parisi, E. D. D. Claudio, G. Orlandi, and B. D. Rao, “Fast adaptive digital equalization by recurrent neural networks,” *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2731-2739, 1997.
- [17] W. D. Weng and C. T. Yen, “Reduced-decision feedback FLANN nonlinear channel equaliser for digital communication systems,” *IEEE Proceedings on communications*, vol. 155, no. 4, pp. 305-311, 2004.
- [18] C. T. Yen, W. D. Weng and Y. T. Lin, “FPGA realization of a neural-network-based nonlinear channel equalizer,” *IEEE Transactions on Industrial Electronics*, vol. 51, no. 2, pp. 472-479, 2004.
- [19] Wan-de Weng, Rui-chang Lin, Chung-ta Hsueh, “The Design of an SCFNN based Nonlinear Channel Equalizer,” *Journal of Information Science and Engineering*, vol. 21, no. 4, pp. 695-709, 2005.
- [20] Rui-chang Lin, Wan-de Weng and Chung-ta Hsueh, “Design of an SCRFNN-based nonlinear channel equaliser,” *IEE Proceedings on Communications*, vol. 152, Issue 6, pp. 771-779, 2005.
- [21] S. Wu and M. J. Er, “Dynamic Fuzzy Neural Networks - A Novel Approach to Function Approximation,” *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics*, vol. 30, no. 2, pp. 358-364, 2000.
- [22] M. J. Er, Z. R. Li, H. N. Cai and Q. Chen, “Adaptive Noise Cancellation Using Enhanced Dynamic Fuzzy Neural Networks,” *IEEE Transactions on Fuzzy Systems*, vol. 13, no. 3, pp. 331-342, 2005.
- [23] M. J. Er and D. Chen, “Obstacle Avoidance of a Mobile Robot Using Hybrid Learning Approach,” *IEEE Transactions on Industrial Electronics*, vol. 52, no. 3, pp. 898-905, 2005.
- [24] S. Wu and M. J. Er, Y. Gao, “A Fast Approach for Automatic Generation of Fuzzy Rules by Generalized Dynamic Neural Networks,” *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 4, pp. 578-594, 2001.



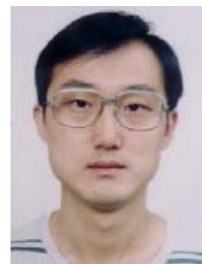
**Meng Joo Er** received his B. Eng and M. Eng degrees in Electrical Engineering from the National University of Singapore in 1985 and 1988 respectively and a Ph. D degree in System Engineering from the Australian National University in 1992. From 1987 to 1989, he worked as a Research and Development Engineer in Chartered Electronics Industries Pte

Ltd and a Software Engineer in Telerate Research and Development Pte Ltd respectively. He served as Director of the Intelligent Systems Center, a University Centre co-funded by Nanyang Technological University (NTU) and Singapore Engineering Technologies from 2003 to 2006. Currently, he is a Professor with the Division of Control and Instrumentation, School of Electrical and Electronic Engineering (EEE), NTU. His research interests include control theory and applications, fuzzy logic and neural networks, computational intelligence, cognitive systems, robotics and automation, sensor networks and biomedical engineering. He has authored three books entitled “Dynamic Fuzzy Neural Networks: Architectures, Algorithms and Applications”, “Engineering Mathematics with Real-World Applications” (published by McGraw Hill in 2003 and 2005 respectively) and “Machine Learning” (to appear in 2009), twelve book chapters and more than 300 refereed journal and conference papers in his research areas of interest. Dr. Er was the winner the Institution of Engineers, Singapore (IES) Prestigious Publication (Application) Award in 1996 and IES Prestigious Publication (Theory) Award in 2001. In recognition of his outstanding research work as a promising young academic staff, he was awarded a Commonwealth Fellowship tenable at University of Strathclyde from 14 February to 11 October, 2000. Due to his excellent performance in teaching, he received the Teacher of the Year Award for the School of EEE in 1999 and the Year 2 Teaching Excellence Award in 2008. He also won the Best Session Presentation Award at the World Congress on Computational Intelligence held from 17 to 21 July, 2006 in Vancouver, Canada. He served as the Editor of IES Journal on Electronics and Computer Engineering from 1995 to 2004. Currently, he serves as the Editor-in-Chief of the IES Journal B—Intelligent Devices and Systems, an Area Editor of International Journal of Intelligent Systems Science and a Guest Editor of International Journal of Neural Systems. Moreover, he serves as an Associate Editor of seven refereed international journals, namely IEEE Transactions on Fuzzy Systems, International Journal of Fuzzy Systems, Neurocomputing, Applied Computational Intelligence and Soft Computing, International Journal of Humanoid Robots, Journal of Robotics and International Journal of Mathematical Control Science and Applications and an Editorial Board Member of three other journals, namely the International Journal of Automation and Smart Technology,

Open Electrical and Electronic Engineering Journal and Birkhauser Autonomics Editorial Board.



**Fan Liu** received her B. Eng. degree and M. Eng. degree from Northwestern Polytechnical University, China, in 2003 and 2006 respectively. She is currently working towards the Ph. D degree at Nanyang Technological University. Her research interests include fuzzy neural networks, genetic algorithms, nonlinear system identification, intelligent control and robotic application.



**Ming Bin Li** received his B. Eng. Degree from the Shenyang Institute of Technology, China in 1998 and his M. Eng. Degree from Northeastern University, China, in 2001. He obtained his Ph. D. degree in Nanyang Technological University (NTU) Singapore in 2006. He is currently a research associate at Intelligent Systems Centre, Nanyang Technological University. His main research interests include neural network, fuzzy logic, system modeling, channel equalization and dynamic control.