International Journal of Applied Science and Engineering
2004. 2, 1: 72-89

Blind Equalization Using Pseudo-Gaussian-Based Compensatory Neuro-Fuzzy Filters

Cheng-Jian Lin* and Wen-Hao Ho

Department of Computer Science and Information Engineering,
Chaoyang University of Technology,
Wufeng, Taichung county 413, Taiwan, R.O.C.

Abstract: An important problem in data communications is that of channel equalization, i.e., the removal of interference introduced by linear or nonlinear message corrupting mechanisms, so that the originally transmitted symbols can be recovered correctly at the receiver. In this paper we introduce a Compensation-Based Neuro-Fuzzy Filter (CNFF) based equalizer whose high performance makes it suitable for high-speed channel equalization. The compensatory fuzzy reasoning method is used in adaptive fuzzy operations that can make the fuzzy logic system more adaptive and effective. Besides, the pseudo-gaussian membership function can provide the compensatory neuro-fuzzy filter which owns a higher flexibility and approaches the optimized result more accurately**.** An on-line learning algorithm, which consists of the structure learning and the parameter learning, is proposed. The structure learning is based on the similarity measure of asymmetry Gaussian membership functions and the parameter learning is based on the supervised gradient decent method. We apply the proposed CNFF to co-channel interference suppression (CCI) and additive with Gaussian noise (AWGN). Computer simulation results show that the bit error rate of the CNFF is close to the optimal equalizer.

Keywords: equalization; neuro-fuzzy filters; co-channel interference; additive with Gaussian noise; pseudo Gaussian; asymmetry similarity measure.

1. Introduction

Adaptive filtering has achieved widespread application and success such as control, image processing, and communication [1]. Among the various adaptive filters, the adaptive linear filter is the most widely used one mainly due to its hardware implementation cost and its properties, like the convergence, global minimum, misadjustment error and training algorithms, and can be analyzed and derived. The adaptive linear filtering has achieved a large amount of success in many situations. The maximum likelihood sequence estimators (MLSE) [2] were implemented using the Viterbi algorithm. The large computational complexity associated with the Viterbi algorithm and the poor performance of the linear equalizers has led to the development of symbol-by-symbol equalizers using the maximum a posteriori probability (MAP) principle -- Bayesian equalizers [3]. These Bayesian equalizers have been approximated using nonlinear signal processing techniques like artificial neural networks (ANN) [4, 5], radial basis functions (RBF)[6, 7], recurrent neural networks [8], fuzzy filters [9-13]. The study of these new techniques can provide adaptive

* Corresponding author: e-mail: cjlin@mail.cyut.edu.tw *Accepted for Publication: Jan. 20, 2004*

adaptive equalizers which have the advantages of both good performance and low computational cost. Fuzzy filters are nonlinear filters that incorporate linguistic information in the form of IF-THEN fuzzy rules. Fuzzy filters have been used for equalization due to their success in the related area of pattern classification [9-12]. Wang and Mendel [9] presented fuzzy basis functions (FBF) for channel equalization. Lin and Juang [10] developed the ANFFs and used it for equalization and noise reduction. This ANFF constructs its rule base in a dynamic way with the training samples. Patra and Mulgrew [11] derived the close relationship between the fuzzy equalizers and the equalizer based on maximum a posteriori probability principle (MAP). Liang and Mendel [12] developed type-2 fuzzy adaptive filters and demonstrated that it can implement the Bayesian equalizer. The structures and learning algorithms of these models [10-12] are both complicated and not suitable for practical implementation.

In this paper, we proposed a Compensation-Based Neuro-Fuzzy Filter (CNFF) which can be constructed by learning from training examples. It can be contrasted with the traditional fuzzy logic control systems in their network structure and learning ability. The CNFF is a four-layer structure (see Fig. 1). Nodes at layer one are input nodes (linguistic nodes) which represent input linguistic variables. Layer four is the output layer. Nodes at layer two are term nodes which act as membership functions to represent the terms of the respective linguistic variable. Each node at layer three is a compensatory rule node which represents one fuzzy logic rule. Thus all layer-three nodes form a fuzzy rule base. The pseudo Gaussian (PG) membership functions proved that the neuro-fuzzy filter owns a higher flexibility and can approach the optimize result more accurately [14]-[15]. Besides, the compensatory fuzzy reasoning method is used in adaptive fuzzy operations that can make the fuzzy logic system more adaptive and effective [16]-[20]. An on-line learning algorithm is proposed to automatically construct the CNFF. It consists of structure learning and parameter learning. The structure learning algorithm decides to add a new node which is satisfying the fuzzy partition of the input data. The similarity measure of asymmetry Gaussian membership functions is used to avoid the newly generated membership function being too similarity to the existing one. The back-propagation learning is then used for tuning input/output membership functions. The proposed learning method has the advantages that it does not require the human expert's assistance and it can converge quickly.

This paper is organized as follows: In Section 2, the compensatory operations are described. Section 3 describes the structure of the CNFF model. The on-line learning algorithm that consists of structure learning and parameter learning is presented in Section 4.Section 5 describes the Bayesian equalizer of a digital communication system with AWGN and CCI. In Section 6, the CNFF model is used to equalize nonlinear channels to demonstrate its learning capability. Comparisons with that of other existing equalizers are also made. Conclusions are summarized in the last section.

2. The compensatory operations

Zimmermann [16] first defined the essence of compensatory operations. Zhang and Kandel [17] proposed more extensive compensatory operations based on the pessimistic operation and the optimistic operation. The pessimistic operation can map the inputs x_i to the pessimistic output by making a conservative decision for the pessimistic situation or even the worst case. For example, $p(x_1, x_2,..., x_n)=MIN(x_1, x_2,..., x_n)$ or $\Pi\ x_i$. Actually, the t-norm fuzzy operation is pessimistic operation. The optimistic operation can map the inputs x_i to the optimistic output by making an optimistic decision for the optimistic situation or even the best case. For example, $o(x_1, x_2,...,$

$x_n)=MAX(x_1, x_2,..., x_n)$. Actually, the *t*-conorm fuzzy operation is optimistic operation. The compensatory operation can map the pessimistic input x_1 and the optimistic input x_2 to make the relatively compromised decision for the situation between the worst case and the best case. For example, $c(x_1, x_2) = x_1^{1-\gamma} x_2^{\gamma}$, where $\gamma \in [0,1]$ is called the compensatory degree. Many researchers [18]-[20] have used the compensatory operation to fuzzy systems successfully.

The general fuzzy if-then rule shown as follows

$$R_j : \text{IF } x_1 \text{ is } A_{1j} \text{ and.....and} x_n \text{ is } A_{nj} \text{ THEN y} = \text{b}_\text{j} \tag{1}$$

where x_i, y are input dimensions and output variables; A_{ij} is linguistic term of the precondition part with membership function $\mu_{A_{ij}}$; b_j is constant consequent; i is input dimension, i =1,..,n; n is the number of existing dimension; j is the number of rule, j=1,..,R; R is the number of existing rule.

For an input fuzzy set A' in U, the jth fuzzy rule (1) can generate an output fuzzy set b_j' in v by using the sup-dot composition

$$\mu_{b_j'} = \sup_{\underline{x} \in U} [\mu_{A_{1j} \times \cdots \times A_{nj} \to b_j}(\underline{x}, y) \bullet \mu_{A'}(\underline{x})] \tag{2}$$

where $\underline{x} = (x_1, x_2, ..., x_n)$. The $\mu_{A_{1j} \times \cdots \times A_{nj}}(\underline{x})$ is defined in a compensatory operation form (3) using the pessimistic operation (4) and theoptimistic operation (5).

$$\mu_{A_{1j} \times \cdots \times A_{nj}}(\underline{x}) = (u_j)^{1-\gamma_j} (v_j)^{\gamma_j} \tag{3}$$

where $\gamma_j \in [0,1]$ is a compensatory degree,

$$u_j = \prod_{i=1}^{n} \mu_{A_{ij}}(x_i) \tag{4}$$

$$v_j = [\prod_{i=1}^{n} \mu_{A_{ij}}(x_i)]^{1/n} \tag{5}$$

For simplicity, we can rewrite

$$\mu_{A_{1j} \times \cdots \times A_{nj}}(\underline{x}) = [\prod_{i=1}^{n} \mu_{A_{ij}}(x_i)]^{1-\gamma_j + \gamma_j/n} \tag{6}$$

Since $\mu_{A'}(x_i) = 1$ for the singleton fuzzifier and $\mu_{b_j'}(y) = 1$, according to (2) we have

$$\mu_{b_j'}(y) = [\prod_{i=1}^{n} \mu_{A_{ij}}(x_i)]^{1-\gamma_j + \gamma_j/n} \tag{7}$$

3. The structure of Compensation-Based Neuro-Fuzzy Filters (CNFF)

A typical network consists of nodes with some finite number of fan-in connections from other nodes represented by weight values, and fan-out connections to other nodes. Associated with the fan-in of a node is an integration function which combines information, activation, or evidence from other nodes, and provides the net input, i.e.,

$$\text{net - input} = f(z_1^{(k)}, z_2^{(k)}, ..., z_p^{(k)}; w_1^{(k)}, w_2^{(k)}, ..., w_p^{(k)}) \tag{8}$$

where $z_i^{(k)}$ is the ith input to a node in layer k and $w_i^{(k)}$ is the weight of the associated link. The superscript in the above equation indicates the layer number. This notation will be also used in the following equations. Each node also outputs an activation value as a function of its net-input

$$\text{output} = a[f(\cdot)] \tag{9}$$

where $a(\cdot)$ denotes the activation function.

The CNFF is a network of four-layer structure in the Figure 1, where the functions of the nodes in each layer are described as follows:

Layer 1: The nodes in this layer are input nodes (i.e., input-linguistic nodes), which represent input- linguistic variables and pass input signals to the next layer directly.

$$f(x_i^{(1)}) = x_i^{(1)} \tag{10}$$

and

$$a[f(\cdot)] = f(\cdot) \tag{11}$$

where i is input dimension.

Layer 2: The nodes in this layer are term nodes that act as the Pseudo-Gaussian (PG) membership function. They can react the terms of the respective input-linguistic variables. For the jth rule node

$$f(z_i^{(2)}) = \exp(-\frac{(z_i^{(2)} - m_{ji})^2}{\sigma_{ji,-}^2})U(z_i^{(2)}; -\infty, m_{ji}) + \exp(-\frac{(z_i^{(2)} - m_{ji})^2}{\sigma_{ji,+}^2})U(z_i^{(2)}; m_{ji}, \infty) \tag{12}$$

and

$$a[f(\cdot)] = f(\cdot) \tag{13}$$

where $U(z_i^{(2)}; a, b) = \begin{cases} 1 & if\ a \le z_i^{(2)} < b \\ 0 & otherwise \end{cases}$.

Layer 3: The nodes in this layer are compensatory fuzzy nodes. They represent the precondition part of fuzzy logic rule, which can input the multiple incoming signals and output the product result. For the rule node

$$f(z_i^{(3)}) = [\prod_i^n z_i^{(3)}]^{1-\gamma+\frac{\gamma}{n}} \tag{14}$$

and

$$a[f(\cdot)] = f(\cdot) \tag{15}$$

where n is dimension number.

Layer 4: The nodes in this layer are denoted by Σ. That is, it receives the multiple incoming signals and outputs the result of summation. For the output

$$f(z_i^{(4)}) = \sum_{j=1}^{M} w_j^{(3)} z_i^{(4)} \tag{16}$$

and

$$a[f(\cdot)] = f(\cdot) \tag{17}$$

where M is rule number; $w_j^{(3)}$ is link weight.

4. The on-line learning algorithm for CNFF

In this section, we propose an on-line learning algorithm, which consists of the structure learning algorithm and the parameter learning algorithm. The structure learning algorithm is used to find proper fuzzy partitions in the input space and create fuzzy logic rules. An asymmetry similarity measure is proposed to avoid the newly generated membership function being too similarity to the existing one. The parameter learning algorithm is the most general supervised learning scheme, which is used to adjust PG membership functions and compensatory operations in the precondition part, and modify the link weight in the consequent part. As a result, the parameter learning algorithm is based on the backpropagation algorithm, which minimizes the cost function to approximate desired results. The procedure of the structure/parameter learning algorithm is through inputting the training pattern to learn successively.

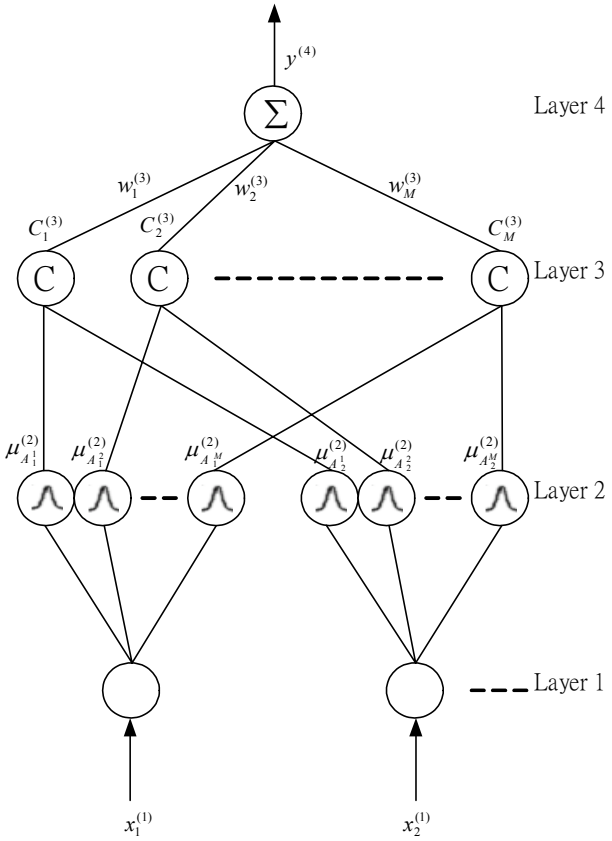


Figure 1. The structure of Compensation-Based Neuro-Fuzzy Filters (CNFF)

4.1. The structure learning algorithm

The proposition of the structure learning algorithm is to deicide proper fuzzy partitions by the input patterns. The procedure of our structure learning algorithm is to find the proper fuzzy logic rules. However, the structure learning algorithm determines whether or not to add a new node in layer 2 via the input pattern data, and decides whether or not to add the associated fuzzy logic rule in layer 3.

After the input pattern is entered in layer 2, the firing strength of the PG membership function will be obtained from Eq.(12), that is used as the degree measure $\mu_{A_i^j}$. In layer 3, the firing strength of fuzzy logic rule obtained from Eq.(14), that is used as the precondition part's degree measure

$$P = \prod_{j=1}^{M(t)} \mu_{A_i^j} \tag{18}$$

where i is input dimension, i =1,…,n;j is rule number, j=1,...,$M(t)$,$M(t)$ is the number of existing rules at time t. $P_{\min} \in (0,1)$ presets positive constants, which should be decreased when the structure learning algorithm limits the rule of CNFF.; P_{max} represents existing maximum P in the firing strength of fuzzy logic rule.

$$P_{\max} = \max_{1 \le j \le M(t)} P_j \tag{19}$$

If P_{min}>P_{max}, the structure learning needs to add a new node in the CNFF model. The new mean, positive deviation and negative devia-

tion of the PG membership function presets values according to the input pattern or heuristic. The process of how the node increases is showed as follow:

$$m_{ji}^{(new)} = x_i \tag{20}$$

$$\sigma_{ji,-}^{(new)} = \sigma_{ji,+}^{(new)} = \sigma_i \tag{21}$$

where x_i is the new input pattern; σ_i is preset constant; i is input dimension; j is rule number.

To avoid the newly generated membership function being too similarity to the existing one, the similarities between the new membership function and existing ones must be checked. If the new fuzzy rule is different from the existing fuzzy rule, we confirmed the new fuzzy rule will be added in the CNFF. It can make neural fuzzy inference system to gain more performance. Therefore, we use similarity measure of asymmetry Gaussian membership functions to estimate the rule's similarity degree.

Since the area of the asymmetry Gaussian membership function, Eq. (12), is $\sigma_+\sqrt{\pi}$, $\sigma_-\sqrt{\pi}$, height is always 1 and center of bottom-line at m_i on x-axis. We can approximate it by an asymmetry triangular $\Delta(m_i,\sigma_{i,-},\sigma_{i,+})$ with unity height and the length of bottom edge $\sigma_+\sqrt{\pi}+\sigma_-\sqrt{\pi}$ (see Figure 2). Assume the two end-point of the bottom-line of $\Delta(m_1,\sigma_{1,-},\sigma_{1,+})$ are a and b on x-axis, and another end-point of $\Delta(m_2,\sigma_{2,-},\sigma_{2,+})$ are c and d on x-axis. That is, $a=m_1-\sigma_{1,-}\sqrt{\pi}$, $b=m_1+\sigma_{1,+}\sqrt{\pi}$, $c=m_2-\sigma_{2,-}\sqrt{\pi}$, $d=m_2+\sigma_{2,+}\sqrt{\pi}$

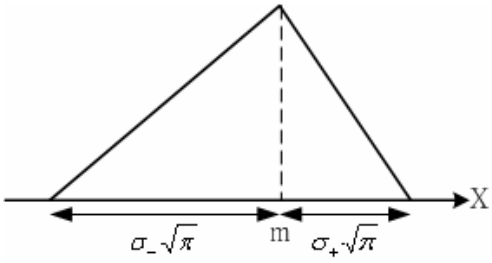


Figure 2. An asymmetry triangle with the unitheight and the length of bottom edge

First, if $m_1=m_2$, when $a\ge c$ and $b\le d$, $|A\cap B|=\frac{1}{2}(\sigma_{2,+}+\sigma_{2,-})\sqrt{\pi}$, $c\ge a$ and $d\le b$,

$|A\cap B|=\frac{1}{2}(\sigma_{1,+}+\sigma_{1,-})\sqrt{\pi}$, $a\ge c$ and $b\ge d$,

$|A\cap B|=\frac{1}{2}(\sigma_{2,+}+\sigma_{1,-})\sqrt{\pi}$, $a\le c$ and $b\le d$,

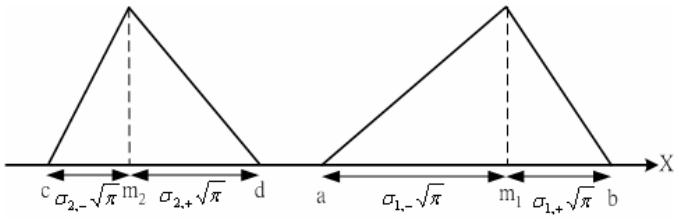
$|A\cap B|=\frac{1}{2}(\sigma_{1,+}+\sigma_{2,-})\sqrt{\pi}$.

In the following discussion, we assume $m_1>m_2$. Let us consider the following five possible situations (see Figure 3):

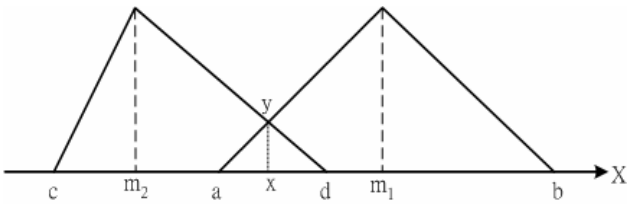
Case 1: If $a\ge d$, then $|A\cap B|=0$ since the two membership functions are not overlapped.

Case 2: If $b\ge d>a\ge c$, then

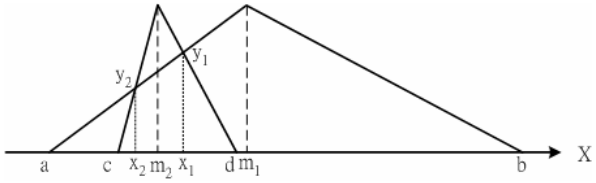
$$|A\cap B|=\frac{1}{2}(d-a)y = \frac{1}{2}\cdot\frac{(m_2+\sigma_{2,+}\sqrt{\pi}-m_1+\sigma_{1,-}\sqrt{\pi})^2}{\sigma_{1,-}\sqrt{\pi}+\sigma_{2,+}\sqrt{\pi}} \tag{22}$$



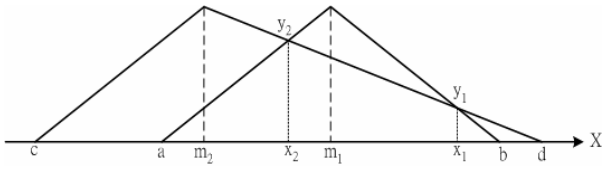
(a) Case 1: $a \geq d$



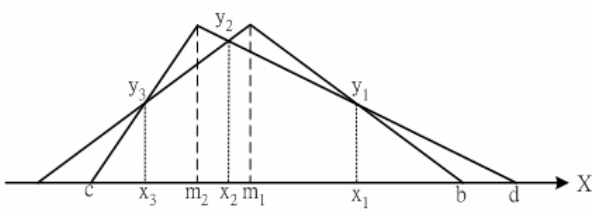
(b) Case 2: $b \geq d > a \geq c$



(c) Case 3: $b > d$ and $c > a$



(d) Case 4: $d > b$ and $a > c$



(e) Case 5: $d > b$ and $c > a$

Figure 3. The five possible situations between two asymmetry triangles

Case 3: If $b > d$ and $c > a$, then

$$\begin{aligned}|A \cap B| &= \frac{1}{2}(x_2 - c)y_2 + \frac{1}{2}(y_1 + y_2)\cdot(x_1 - x_2) + \frac{1}{2}(d - x_1)y_1 \\ &= \frac{1}{2}\cdot\frac{(m_2 - \sigma_{2,-}\sqrt{\pi} - m_1 + \sigma_{1,-}\sqrt{\pi})^2}{\sigma_{1,-}\sqrt{\pi} - \sigma_{2,-}\sqrt{\pi}} + \frac{1}{2}\cdot\frac{(m_2 + \sigma_{2,+}\sqrt{\pi} - m_1 + \sigma_{1,-}\sqrt{\pi})^2}{\sigma_{1,-}\sqrt{\pi} + \sigma_{2,+}\sqrt{\pi}}\end{aligned} \tag{23}$$

Case 4: If $d > b$ and $a > c$, then

$$\begin{aligned}|A \cap B| &= \frac{1}{2}(x_2 - a)y_2 + \frac{1}{2}(y_1 + y_2)\cdot(x_1 - x_2) + \frac{1}{2}(b - x_1)y_1 \\ &= \frac{1}{2}\cdot\frac{(m_2 + \sigma_{2,+}\sqrt{\pi} - m_1 + \sigma_{1,-}\sqrt{\pi})^2}{\sigma_{1,-}\sqrt{\pi} + \sigma_{2,+}\sqrt{\pi}} + \frac{1}{2}\cdot\frac{(-m_2 - \sigma_{2,+}\sqrt{\pi} + m_1 + \sigma_{1,-}\sqrt{\pi})^2}{\sigma_{1,-}\sqrt{\pi} - \sigma_{2,+}\sqrt{\pi}}\end{aligned} \tag{24}$$

Case 5: If $d > b$ and $c > a$, then

$$\begin{aligned}|A \cap B| &= \frac{1}{2}(x_3 - c)y_3 + \frac{1}{2}(y_2 + y_3)\cdot(x_2 - x_3) + \frac{1}{2}(y_1 + y_2)(x_1 - x_2) + \frac{1}{2}(b - x_1)y_1 \\ &= \frac{1}{2}\cdot\frac{(m_2 + \sigma_{2,-}\sqrt{\pi} - m_1 + \sigma_{1,-}\sqrt{\pi})^2}{\sigma_{1,-}\sqrt{\pi} - \sigma_{2,-}\sqrt{\pi}} + \frac{1}{2}\cdot\frac{(m_2 + \sigma_{2,+}\sqrt{\pi} - m_1 + \sigma_{1,-}\sqrt{\pi})^2}{\sigma_{1,-}\sqrt{\pi} + \sigma_{2,+}\sqrt{\pi}} \\ &\quad + \frac{1}{2}\cdot\frac{(-m_2 - \sigma_{2,+}\sqrt{\pi} + m_1 + \sigma_{1,+}\sqrt{\pi})^2}{\sigma_{1,+}\sqrt{\pi} - \sigma_{2,+}\sqrt{\pi}}\end{aligned} \tag{25}$$

We can conclude a general formula for $|A \cap B|$

$$\begin{aligned}|A \cap B| &= \frac{1}{2}\cdot\frac{h^2(m_2 + \sigma_{2,+}\sqrt{\pi} - m_1 + \sigma_{1,-}\sqrt{\pi})}{\sigma_{1,-}\sqrt{\pi} + \sigma_{2,+}\sqrt{\pi}} + \frac{1}{2}\cdot\frac{h^2(m_2 - \sigma_{2,-}\sqrt{\pi} - m_1 + \sigma_{1,-}\sqrt{\pi})}{-\sigma_{1,-}\sqrt{\pi} + \sigma_{2,-}\sqrt{\pi}} \\ &\quad + \frac{1}{2}\cdot\frac{h^2(m_2 + \sigma_{2,+}\sqrt{\pi} - m_1 - \sigma_{1,+}\sqrt{\pi})}{\sigma_{1,+}\sqrt{\pi} - \sigma_{2,+}\sqrt{\pi}}\end{aligned} \tag{26}$$

where $h(x) = \max\{0, x\}$. So the approximate similarity measure of fuzzy sets is

$$\begin{aligned}E(A,B) &= \frac{|A \cap B|}{|A \cup B|} \\ &= \frac{|A \cap B|}{\frac{1}{2}\sigma_{1,+}\sqrt{\pi} + \frac{1}{2}\sigma_{1,-}\sqrt{\pi} + \frac{1}{2}\sigma_{2,+}\sqrt{\pi} + \frac{1}{2}\sigma_{2,-}\sqrt{\pi} - |A \cap B|}\end{aligned} \tag{27}$$

The similarity measure E between the new membership function and all existing ones are calculated and maximum one E_{max}, is found as follows:

$$E_{\max} = \max_{1\le j\le M(t)} E(\mu(m_1^{(new)}, \sigma_{1,+}^{(new)}, \sigma_{1,-}^{(new)}), \mu(m_1^{j}, \sigma_{1,+}^{j}, \sigma_{i,-}^{j})) \tag{28}$$

If $E_{\max} \le E_{\min}$, where $E_{\min} \in (0,1)$ is a pre-specified value, then the new fuzzy logic rule is adopted and the rule number is incremented.

$$M = M + 1 \tag{29}$$

Therefore, the new mean, deviation and link weight are generated randomly.

4.2. The Parameter learning algorithm

After the structure network has been according adjusted to the current training pattern, the network enters the parameter learning algorithm. The procedure of the parameter learning algorithm is to adjust the parameter of CNFF optimally with same training pattern. The backpropagation is used for this supervised learning to find the output errors of the node in each layer and analyze the error to perform parameter adjustment. The goal is to minimize the error function

$$E = \frac{1}{2}\left(y^d(t) - y(t)\right)^2 \tag{30}$$

where $y^d(t)$ is desired output and $y(t)$ is the model output. Then the parameter learning algorithm based on back-propagation is as follows:

Assuming that w is the adjustable parameter in a node, the general used learning rule is

$$w(t+1) = w(t) - \eta(\frac{\partial E}{\partial w}) \tag{31}$$

$$\begin{aligned}\frac{\partial E}{\partial w} &= \frac{\partial E}{\partial f}\cdot\frac{\partial f}{\partial w} \\ &= \frac{\partial E}{\partial a}\cdot\frac{\partial a}{\partial f}\cdot\frac{\partial f}{\partial w}\end{aligned} \tag{32}$$

where η is the learning rate.

To show the learning rules, we derive the parameter learning layer by layer.

Layer 4: There is no parameter to be adjusted in this layer.

Layer 3: Using Eqs. (31) and (32), the link weight is adjusted by the amount

$$w_j^{(3)}(t+1) = w_j^{(3)}(t) - \eta_w(\frac{\partial E}{\partial w_j^{(3)}}) \tag{33}$$

where

$$\frac{\partial E}{\partial w_j^{(3)}} = (y^d - y^{(4)})\cdot[\prod_{i=1}^{n}\mu_{A_i^j}(x_i)]^{1-\gamma+\gamma/n} \tag{34}$$

To eliminate the constraint $\gamma \in [0,1]$, we redefine γ as follow:

$$r = \frac{c^2}{c^2 + d^2} \tag{35}$$

The parameter of γ adjust as follows:

$$c(t+1)=c(t)-\eta_c\left(\frac{2c(t)d^2(t)}{\left[c^2(t)+d^2(t)\right]^2}\right)\frac{\partial E}{\partial r} \quad (36)$$

$$d(t+1)=d(t)-\eta_d\left(\frac{2c^2(t)d(t)}{\left[c^2(t)+d^2(t)\right]^2}\right)\frac{\partial E}{\partial r} \quad (37)$$

where

$$\frac{\partial E}{\partial r}=(y^d-y^{(4)})\cdot w_j^{(3)}\cdot\left(\frac{1-n}{n}\right)\cdot\prod_{i=1}^{n}\mu_{A_i^j}^{(2)}\cdot\ln\left[\prod_{i=1}^{n}\mu_{A_i^j}^{(2)}\right] \quad (38)$$

$$r(t+1)=\frac{c^2(t+1)}{c^2(t+1)+d^2(t+1)} \quad (39)$$

Layer 2: The m_{ji} of the PG membership function is adjusted by the amount

$$m_{ji}(t+1)=m_{ji}(t)-\eta_m\cdot\left(\frac{\partial E}{\partial m_{ji}}\right) \quad (40)$$

where

$$\frac{\partial E}{\partial m_{ji}}=(y^d-y^{(4)})\cdot w_j^{(3)}\cdot(1-r+\frac{r}{n})\cdot(\prod_{i=1}^{n}\mu_{A_i^j}^{(2)})^{-r+\frac{r}{n}}\cdot\prod_{\substack{l=1\\l\neq i}}^{n}\mu_{A_l^j}^{(2)}\cdot$$
$$\left[\frac{2(x_i^{(1)}-m_{ji})}{\sigma_{ji,-}^2}\cdot\exp(-\frac{(x_i^{(1)}-m_{ji})^2}{\sigma_{ji,-}^2})U(x_i^{(1)};-\infty,m_{ji})+\right.$$
$$\left.\frac{2(x_i^{(1)}-m_{ji})}{\sigma_{ji,+}^2}\cdot\exp(-\frac{(x_i^{(1)}-m_{ji})^2}{\sigma_{ji,+}^2})U(x_i^{(1)};m_{ji},\infty)\right] \quad (41)$$

The $\sigma_{ji,-}$ of the PG membership function is adjusted by the amount

$$\sigma_{ji,-}(t+1)=\sigma_{ji,-}(t)-\eta_{\sigma_-}\cdot\left(\frac{\partial E}{\partial\sigma_{ji,-}}\right) \quad (42)$$

where

$$\frac{\partial E}{\partial\sigma_{ji,-}}=(y^{(4)}-y^d)\cdot w_j^{(3)}\cdot(1-r+\frac{r}{n})\cdot(\prod_{i=1}^{n}\mu_{A_i^j}^{(2)})^{-r+\frac{r}{n}}\cdot\prod_{\substack{l=1\\l\neq i}}^{n}\mu_{A_l^j}^{(2)}\cdot$$
$$\left[\frac{2(x_i^{(1)}-m_{ji})^2}{\sigma_{ji,-}^3}\cdot\exp(-\frac{(x_i^{(1)}-m_{ji})^2}{\sigma_{ji,-}^2})U(x_i^{(1)};-\infty,m_{ji})\right] \quad (43)$$

The $\sigma_{ji,+}$ of PG membership function is adjusted by the amount

$$\sigma_{ji,+}(t+1)=\sigma_{ji,+}(t)-\eta_{\sigma_+}\cdot\left(\frac{\partial E}{\partial\sigma_{ji,+}}\right) \quad (44)$$

where

$$\frac{\partial E}{\partial\sigma_{ji,+}}=(y^{(4)}-y^d)\cdot w_j^{(3)}\cdot(1-r+\frac{r}{n})\cdot(\prod_{i=1}^{n}\mu_{A_i^j}^{(2)})^{-r+\frac{r}{n}}\cdot\prod_{\substack{l=1\\l\neq i}}^{n}\mu_{A_l^j}^{(2)}\cdot$$
$$\left[\frac{2(x_i^{(1)}-m_{ji})^2}{\sigma_{ji,+}^3}\cdot\exp(-\frac{(x_i^{(1)}-m_{ji})^2}{\sigma_{ji,+}^2})U(x_i^{(1)};m_{ji,}\infty)\right] \quad (45)$$

where η_m, η_{σ_+} and η_{σ_-} represent the learning rate parameters of the PG membership function respectively.

5. The bayesian equalizer of a digital communication system

A. A digital communication system with AWGN

Nonlinear channel equalization is a technique used to combat some imperfect phenomenon in high-speed channel. The digital communication system with AWGN is pre-

sented in Figure 4. The transmission input signal $s(k)$ is a sequence of statistically independent random binary symbols taking values $s(k) \in \{-1,1\}$. The equalizer uses an input receiver signal vector $x(k) \in \Re^m$, the m dimensional space, then the channel function can be described as

$$\hat{x}(k) = f[s(k), s(k-1), ..., s(k-N)] \quad (46)$$

In general, f is a nonlinear function of the past transmitted signal, and the channels change slowly but significantly over time, so a nonlinear channel equalizer with adaptation ability is needed. At the receiving end, the observed signal *x(k)* is the channel output $\hat{x}(k)$ corrupted by additive noise *e(k)*, that is

$$x(k) = \hat{x}(k) + e(k) \quad (47)$$

The noise source $e(k)$ is assumed to be zero mean white Gaussian with a variance of σ_e^2. The task of the equalizer is to reconstruct the transmitted signal $s(k-d)$ from the observed information sequence $x(k), x(k-1), ..., x(k-N+1)$ (where d and N denoted the lag and order, respectively) such that greater speed and higher reliability can be achieved. With this the signal-to-noise ratio (SNR) [11] can be represented as

$$SNR = 10 \log \frac{\sigma_s^2}{\sigma_e^2} \quad (48)$$

where σ_s^2 represents the signal power. In a communication channel with AWGN, but no CCI, the decision function of Bayesian

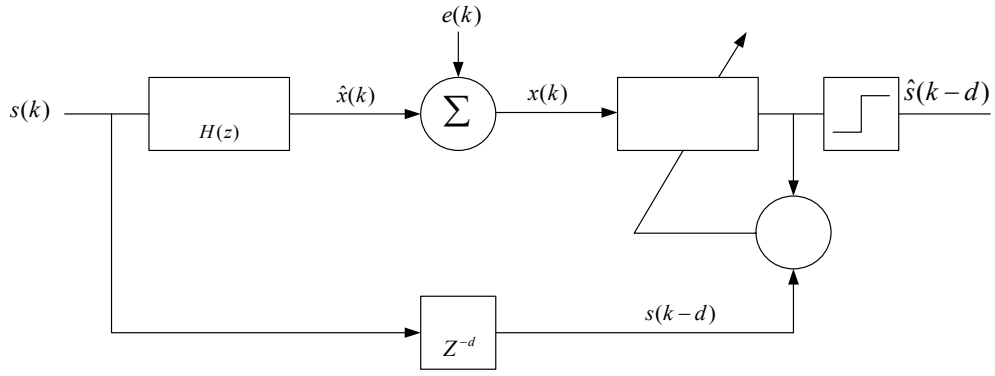


Figure 4. Discrete-time model of a digital communication system with AWGN

equalizer is

$$f(x(k)) = \sum_{i=1}^{n_s} \prod_{l=1}^{p-1} w_i \exp\left(-\frac{1}{2} \frac{[x(k-l) - \hat{x}_i(k-l)]^2}{\sigma_e^2}\right) \quad (49)$$

where w_i equals either +1 or -1 as determined by the channel state category.

B. A digital communication system with AWGN and CCI

In Figure 5, the discrete-time model is gen-

erally used to represent a communication system corrupted with CCI and AWGN. $H_0(z)$ is the desired channel and $H_i(z)$, $1 \le i \le n$, are the interference co-channels. The impulse response of the channels and co-channels can be represented as

$$H_i(z) = \sum_{j=0}^{p_i} a_{i.j} z^{-j} \tag{50}$$

Here p_i and $a_{i,j}$ are the length and tap weights of *i*th channel impulse response. Only the transmitted signal $s_0(k)$ of the desired channel is available at the receiver during training period. Without loss of generality, it can be assumed that the communication system is binary. The transmitted sequences $s_i(k)$; $1 \le i \le n$, are mutually independent and are taken from independent identically distributed. Data set with values {1,-1}. The input to the equalizer forms the observation vector from channel output. Each of the components of this vector can be presented as

$$x(k) = \hat{x}(k) + \hat{x}_{co}(k) + e(k) \tag{51}$$

where, $\hat{x}(k)$ is the desired received signal, $\hat{x}_{co}(k)$ is the interfering signal. The noise $e(k)$ is assumed to be Gaussian with variance σ_e^2 and is uncorrelated with the data. With this the signal-to-interference ratio (SIR) and the signal-to-interference noise ratio (SINR) [11] can be represented as

$$SIR = 10\log\frac{\sigma_s^2}{\sigma_{co}^2} \tag{52}$$

$$SINR = 10\log\frac{\sigma_s^2}{(\sigma_{co}^2 + \sigma_e^2)} \tag{53}$$

where σ_s^2 and σ_{co}^2 represent the signal power and the co-channel signal powers, respectively. The task of the equalizer is to estimate the delayed transmitted sequence $s_0(k-d)$ based on the channel observation vector $x(k) = [x(k), x(k-1), ..., x(k-N+1)]^T$. The communication system with CCI and AWGN, the decision function of the Bayesian equalizer is

$$f(x(k)) = \sum_{i=1}^{n_s} \sum_{m=1}^{n_{co}} \prod_{l=1}^{p-1} w_i \exp\left(-\frac{1}{2}\frac{[x(k-l) - \hat{x}_i(k-l) - \hat{x}_{co}^m(k-l)]^2}{\sigma_e^2 + \sigma_{co}^2}\right) \tag{54}$$

where $\hat{x}_{co}^m(k-l)$ is the *l*th element of *m*th co-channel state.

6. Illustrative examples

To certify the performance of the CNFF for communication channel problems, several examples are presented in this section. The first example is a communication channel with AWGN; the second example is a communication channel with AWGN and CCI.

Example 1: *A Communication Channel with AWGN*

For the purpose of validation of the CNFF model developed in the preceding sections extensive simulation were carried out. We considered the following nonlinear channel model for simulation:

$$\hat{x}(k) = \hat{O}(k) + 0.1\hat{O}^3(k) \qquad (55)$$

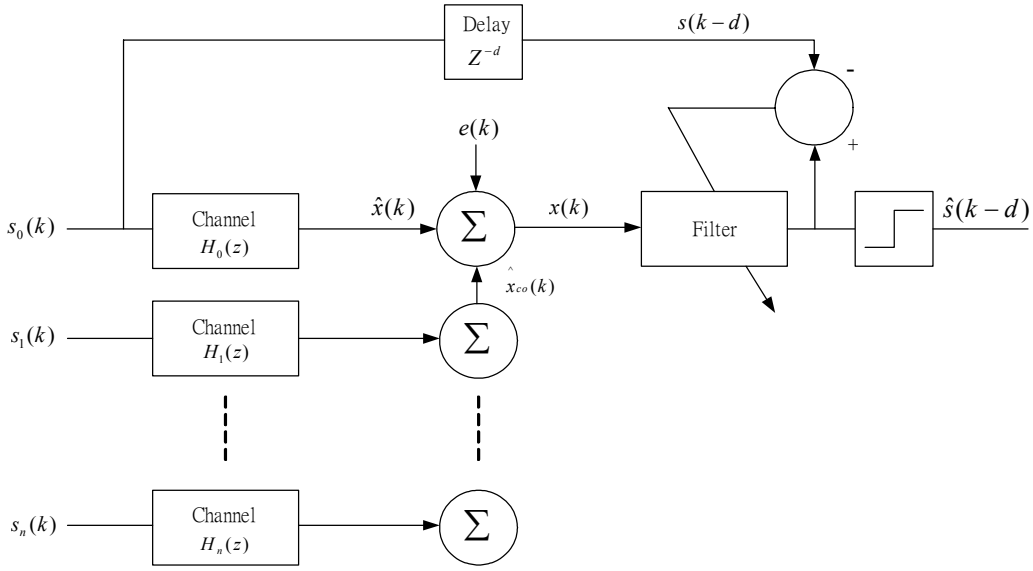


Figure 5. Discrete-time model of a digital communication system with CCI and AWG

where $\hat{O}(k) = 0.5s(k) + s(k-1)$ and noise e(k) is zero-mean colored Gaussian distributed with $E[e(k)^2] = 0.2$ and $E[e(k)e(k-1)] = 0.1$. Assume the desired output y is $\hat{s}(k-d)$ which $d = 1$. The input dimension is $N = 2$. Then by Eq. (49) the optimal Bayesian equalizer boundary can be derived and shown in Figure 6. In Figure 6, the shaded region is the region where the transmitted signal is classified as 1, otherwise it is classified as -1. Also shown in the figure are the symbol " * " and " o " which denote the signal during transmission channel without noise, respectively.

We now used the CNFF as an equalizer to solve the above problems. The learning rate is set as $\eta = 0.1$. The $E = 0.5$ is similarity threshold which to decide two PG membership function whether similar or not. There are no rules initially and they are generated during the training process. When new rule come into being, the new deviation of PG membership function is set as $\sigma_i^{(new)} = 0.4$ and the new mean of PG membership function is set as $m_i^{(new)} = input\_value$. The parameters *c* and *d* are generated randomly. The consequent weight $\omega \in [-1, 1]$ defines by training data desired, respectively. We trained the CNFF for each on-line incoming training pattern. Since the desired output is either 1 or -1, there are only two clusters centered at $c_0 = 1$ and $c_1 = -1$ in the output space. There are no rules initially and they are generated during the training process. The simulation results (the decision boundaries) after the on-line training stopped at k=10 and k=50

are shown in Figures 7 and 8 with the generated rule numbers being 6 and 7, where k denotes the number of time steps (sampling points).

To explain the meaning of the above results, let us consider Figure 7. In Figure 7, there are six fuzzy rules (marked as *R1*, *R2*, *R3*, *R4*, *R5*, *R6*) are generated during the learning process. The six fuzzy rules are described as follows:

Rule 1: IF x is *R1*, THEN y is O_1
Rule 2: IF x is *R2*, THEN y is O_1
Rule 3: IF x is *R3*, THEN y is O_2
Rule 4: IF x is *R4*, THEN y is O_2
Rule 5: IF x is *R5*, THEN y is O_1
Rule 6: IF x is *R6*, THEN y is O_2

where $x = (x(k),x(k\text{-}1))^T$, and O_1 and O_2 are the two output clusters meaning the decided result being 1 or -1, respectively. The functions of these rules are in fact to classify whether the received samples contains 1 or -1. From this point of view, the equalizer can be viewed as a classifier, and the problem can be considered as a classification problem [6]. In Figure 6 and Figure 7, the results show that the decision boundary has the tendency of converging to the optimal one. To see the actual bit-error-rate (BER), a realization of 10^6 points of sequence $s(k)$ and $e(k)$ are used to test the BER of trained CNFF equalizer. The resulting BER curve of the CNFF equalizer under the different SNR is shown in Figure 9.

We now compare the performance of our model with that of other existing methods. The neural network 1 differs greatly from the neural network 50 in training times. The neural network 1 and the neural network 50 represent the training patterns are trained one time and fifty times. The LVQ (Learning Vector Quantization) is a clustering method. The basic idea is to represent the input vectors with a smaller set of prototypes that provide a good approximation to the input space. The Bayesian equalizer is near optimal method for communication channel equalizer. Figure 9 shows the bit-error-rate curves for the optimal Bayesian equalizer, CNFF, neural networks, and LVQ. Simulation results show that the proposed CNFF model is very close to the optimal Bayesian equalizer and obtain better performance than neural networks and LVQ.

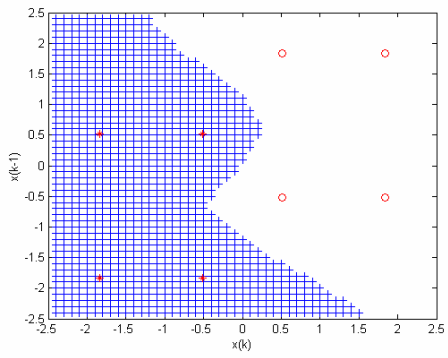


Figure 6. Channel output point and Bayesian equalizer (optimal) decision region

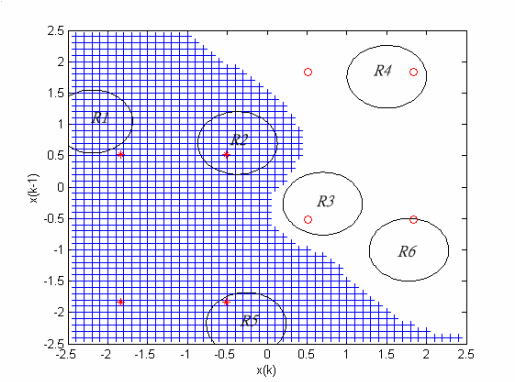


Figure 7. Decision region of the CNFF in channel output d=1 when the adaptation is stop at k=10

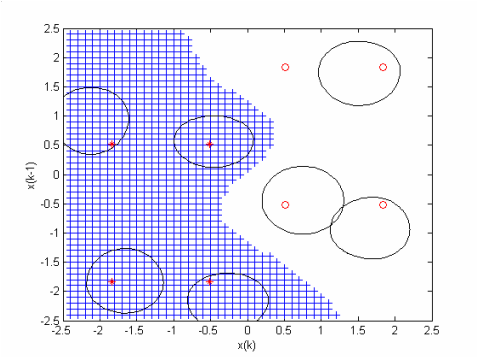


Figure 8. Decision region of the CNFF in channel output d=1 when the adaptation is stop at k=50

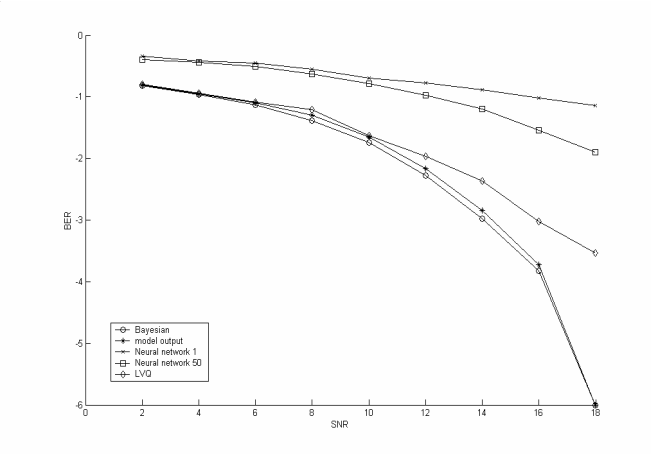


Figure 9. Comparison of bit-error-rate curves for the optimal equalizer, CNFF, neural networks, LVQ

Example 2: *A Communication Channel with AWGN and CCI*

The performance of fuzzy equalizers in CCI environment is considered next. The channel and the co-channel are characterized by their impulse responses

$$H_{ch}(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2} \quad (56)$$

$$H_{co1}(z) = \lambda(0.5 + 0.81z^{-1} + 0.31z^{-2}) \quad (57)$$

Assume $d = 1$ and $N = 2$. We choose the initial parameters $\eta = 0.01$, E=0.6, and $\sigma_i = 0.5$. The simulation results (the decision boundaries) after the on-line training stopped at k=50 and k=100 are shown in Figure 10 and Figure 11 with the generated rule numbers being 5.

To see the actual bit-error-rate (BER), a realization of 10^6 points of sequence $s(k)$ and $e(k)$ are used to test the BER of trained network. We also compare the performance of our model with other methods. The BER curves under the different SNR are shown in Figure 12. Simulation results also show that the proposed CNFF model can obtain better performance than other methods.

7. Conclusion

We have implemented a new elegant CNFF model which performs close to the optimal Bayesian with substantial reduction in computational complexity. We address the automatic determination of the structure of the CNFF and the simultaneous optimization of both membership functions and fuzzy rule conclusions. The PG membership function is used to construct the general neural fuzzy system and to make the variability and the flexibility of the CNFF higher. The asymmetry similarity measure is proposed to estimate the rule's similarity degree. The proposed CNFF can elevate the converging speed and generate less fuzzy logic rules. Computer simulation results show that the bit error rate of the CNFF is close to the optimal equalizer.

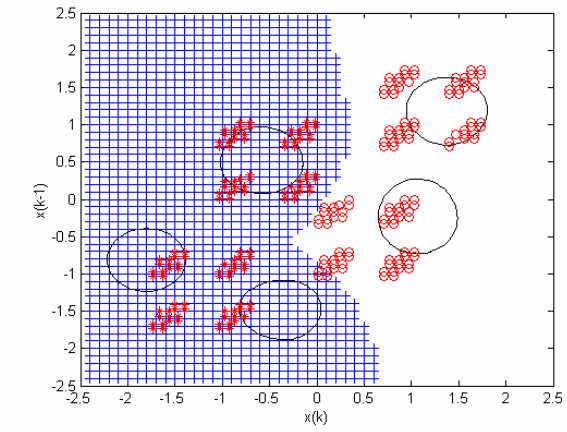


Figure 10. Decision region of the CNFF in co-channel output d=1 when the adaptation is stop at k=50

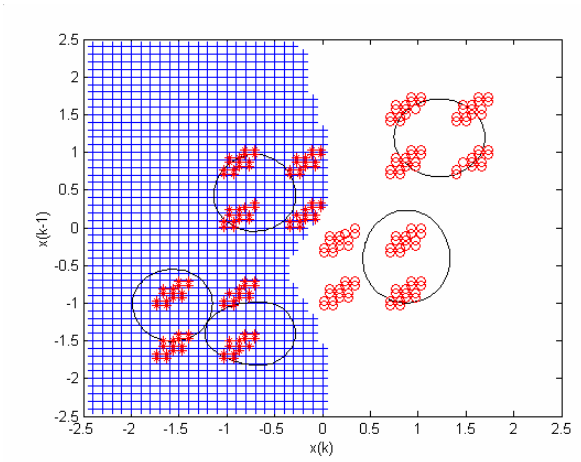


Figure 11. Decision region of the CNFF in co-channel output d=1 when the adaptation is stop at k=100

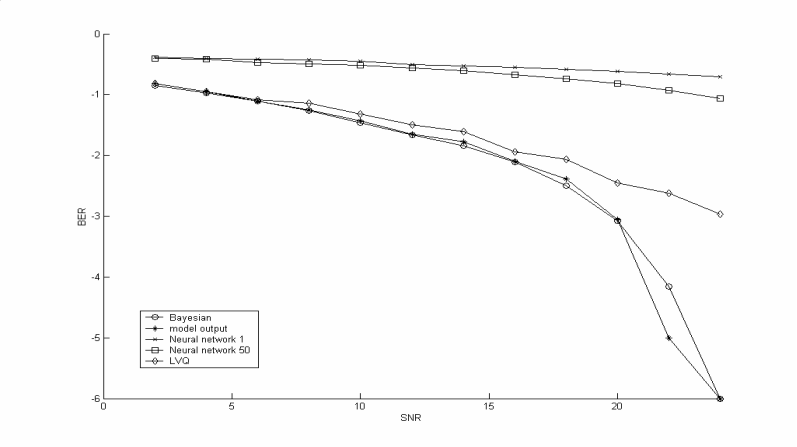


Figure 12. Comparison of bit-error-rate curves for the optimal equalizer, CNFF, neural networks, LVQ in co-channel

Acknowledgement

This research is supported by the National Science Council of the R.O.C. under grant NSC 92-2213-E-324-002.

References

[1] Widrow, B. and Stearns, S. D. 1985. *"Adaptive Signal Processing"*. Englewood Cliffs, NJ: Prentice-Hall.

[2] Forney, G. D. 1972. Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference, *IEEE Transactions on Information Theory,* 18: 363-378.

[3] Duda, R. O. 1973. *"Pattern Classification and Scene Analysis"*. Wiley, New York.

[4] Gibson, G. J., Siu, S. and C. F. N. Cowan, 1991. The application of

nonlinear structures to reconstruction of binary signals. *IEEE Transactions on Signal Processing,* 39: 1877-1884.

[5] AlMashouq, K. A. and Reed, I. S. 1994. The use of neural nets to combine equalization with decoding for severe intersynbol interference channels. *IEEE Transactions on Neural Networks,* 5: 982-988.

[6] Chen, S., Mulgrew, B., and S. McLaughlin, 1993. A clustering technique for digital communications channel equalization using radial basis function network. *IEEE Transactions on Neural Networks,* 4: 570-579.

[7] Chen, S., Gibson, G. J., Cowan, C. F. N., and Grant, P. M. 1991. Reconstruction of binary signals using radial-basis-function equalizer. *Signal Processing,* 22: 77-93.

[8] Kechriotis, G. and Manolakos, E. S. 1994. Using recurrent neural networks for adaptive communication channel equalization. *IEEE Transactions on Neural Networks,* 5, 2: 267-278.

[9] Wang, L.X. and Mendel, J. M. 1993. Fuzzy adaptive filters with application to nonlinear channel equalization. *IEEE Transactions Fuzzy Systems,* 1: 161-170.

[10] Lin, C. T. and Juang, C. F. 1997. Adaptive neural fuzzy filter and its applications. *IEEE Transactions on Systems, Man, and Cybernetics (B),* 27, 4: 640-656.

[11] Patra, S. K. and Mulgrew, B. 2000. Fuzzy techniques for adaptive nonlinear equalization. *Signal Processing,* 80: 985-1000.

[12] Liang, Q. and Mendel, J. M. 2000. Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters. *IEEE Transactions on Fuzzy Systems,* 8, 5: 551-563.

[13] Cowan, C. F. N. and Semnani, S. 1998. Time-variant equalization using a novel nonlinear adaptive structure. *International Journal of Adaptive Control and Signal Processing,* 12, 2: 195-206.

[14] Rojas, I. Pomares, H. Fernandez, F. J. Bernier, J. L. Pelayo, F. J., and Prieto, A. 1999. A new methodology to obtain fuzzy systems autonomously from training data. *IEEE Conference on Fuzzy Systems,* 1: 527-532.

[15] Rojas, I. Pomares, H. Fernandez, F. J. Bernier, J. L. Pelayo, F. J., and A. Prieto. 2000. A new radial basis function networks structure: application to time series prediction. *IEEE International Joint Conference on Neural Networks*: 449-454.

[16] Zimmermann, H. J. and Zysno, P. 1980. Latent connective in human decision. *Fuzzy Sets and Systems,* 4: 31-51.

[17] Zhang, Y. Q. and Kandel, A. 1998. Compensatory neurofuzzy systems with fast learning algorithms. *IEEE Transactions on Neural Networks,* 9, 1: 83-105.

[18] Ouyang, C. S. and Lee, S. J. 1999. An improved learning algorithm for rule refinement in neuro-fuzzy modeling. *Third International Conference on Knowledge-Based Intelligent Information Engineering Systems*: 238-241.

[19] Lin, C. J. and Chen, C. H. 2003. Nonlinear system control using compensatory neuro-fuzzy networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences,* 86, 9: 2309-2316.

[20] Ho, W. H. and Lin, C. J. 2002. A pseudo-gaussian-based neural fuzzy system and its applications. *The Seventh Conference on Artificial Intelligence and Applications*: 12-16.