Chapter 5

Compensatory Neuro-Fuzzy Filter (CNFF)

In this Chapter, we analyze the functioning of the channel equalizer based on the Compensatory Neuro-Fuzzy Filter (CNFF). It is stated in Chapter 4 that the channel equalization is a non-linear problem, so that a non-linear solution is more appropriate. Moreover, for the mobile channel, Blind Equalization is the most preferred technique, due to the very special nature of the channel.

The rest of the Chapter is organized as follows. In Section 5.1, we introduce the working principle of Compensatory Neuro-Fuzzy Filter (CNFF). The channel equalizer based on CNFF is introduced in Section 5.2. In Section 5.3, we consider the detailed structure of the Compensatory Neuro-Fuzzy Filter (CNFF) [20, 21] for non-linear channel equalization. Conclusions are made in Section 5.4.

5.1 Introduction

A fuzzy logic system (FLS) is unique in that it is able to simultaneously handle numerical data and linguistic knowledge. It is a nonlinear mapping of an input data (feature) vector into a scalar output, i.e., it maps numbers into numbers.

For many problems, two distinct forms of problem knowledge exist: the first one is objective knowledge, which is used all the time in engineering problem formulations (e.g., mathematical models), and the second one is subjective knowledge, which represents linguistic information that is usually impossible to quantify using traditional mathematics (e.g., rules, expert information, design requirements).

No standard methods exist for optimally transforming human knowledge or experience into the rule base and database of a fuzzy inference system. There is a need for effective methods for tuning the membership functions (MFs) so as to minimize the output error measure or maximize performance index. The novel architecture of Adap-
CHAPTER 5. COMPENSATORY NEURO-FUZZY FILTER (CNFF)

tive Network based Fuzzy Inference System (ANFIS), was discussed in greater detail in Chapter 4.

In this Chapter, we discuss the Compensatory Neuro-Fuzzy Filter (CNFF), as an alternate technique for channel equalization. The Compensatory Neuro-Fuzzy Filter (CNFF) 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.

5.2 Compensatory Neuro-Fuzzy Filter (CNFF)

The solution to the problem of channel equalization is targeted towards 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 section, we consider a Compensatory 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 online 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 White Gaussian Noise (AWGN). Computer simulation results show that the bit error rate of the CNFF is close to the optimal equalizer [20].

5.2.1 The Outline of CNFF

The CNFF is a four-layer structure (see Figure 5.1). 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 provides that the neuro-fuzzy filter owns a higher flexibility and can approach the optimized result more accurately [20]. Besides, the compensatory fuzzy reasoning method is used in adaptive fuzzy operations that can make the fuzzy logic system more adaptive and effective. An on-line learning algorithm is used to construct the CNFF automatically. 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 similar to the existing one [20]. The back-propagation learning is then used for tuning input/output membership functions. The learning method has the advantage that it does not require the human expert’s assistance and it can converge quickly.

5.2.2 The Details of Compensatory Operations.

It is Zimmermann [78] who first define the essence of compensatory operations. Zhang and Kandel [79] have 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, \( p(x_1, x_2, \ldots, x_n) = \text{MIN}(x_1, x_2, \ldots, x_n) \) or \( \prod_{i=1}^{n} x_i \). Actually, the \( l - \text{norm} \) fuzzy operation is a 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 in the best case.
CHAPTER 5. COMPENSATORY NEURO-FUZZY FILTER (CNFF)

For example, $o(x_1, x_2, \ldots, x_n) = \text{MAX}(x_1, x_2, \ldots, x_n)$. Actually, the $t$-conorm fuzzy operation is an optimistic operation [20]. 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 have used the compensatory operation to fuzzy systems successfully.

The general fuzzy if-then rule is shown as follows

$$R_j : \text{IF } x_1 \text{ is } A_{1j} \text{ and } \ldots \text{ and } x_n \text{ is } A_{nj} \text{ THEN } y_b = b_j \quad (5.1)$$

where $x_i$, $y_i$ 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; $n$ is input dimension, $i$ is the dimension index, $i = 1, \ldots, n$; $n$ is the number of existing dimension; $j$ is the number of the rule, $j = 1, \ldots, R$; $R$ is the number of existing rules. For an input fuzzy set $A'$ in $U$, the $j^{th}$ fuzzy rule 5.1 can generate an output fuzzy set $b_j'$ in $\nu$ by using the $sup - dot$ composition

$$\mu_{b_j'} = \sup_{x \in U} \left[ \mu_{A_{1j} \times \ldots \times A_{nj}} - b_j \circ \mu_{A'}(x) \right] \quad (5.2)$$

where $x = (x_1, x_2, \ldots, x_n)$. The $\mu_{A_{1j} \times \ldots \times A_{nj}}(x)$ is defined in a compensatory operation form 5.3 using the pessimistic operation 5.4 and the optimistic operation 5.5.

$$\mu_{A_{1j} \times \ldots \times A_{nj}}(x) = (u_j)^{1-\gamma_j}(v_j)^{\gamma_j} \quad (5.3)$$

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

$$u_j = \prod_{i=1}^{n} \mu_{A_{ij}}(x_i) \quad (5.4)$$

$$v_j = \prod_{i=1}^{n} \mu_{A_{ij}}(x_i) \quad (5.5)$$

After simplification, we can write

$$\mu_{A_{1j} \times \ldots \times A_{nj}}(x) = \left[ \prod_{i=1}^{n} \mu_{A_{ij}}(x_i) \right]^{1-\gamma_j} \times \frac{\gamma_j}{n} \quad (5.6)$$

Since $\mu_{A}(x) = 1$ for the singleton fuzzifier and $\mu_{b_j'}(x) = 1$, according to 5.2 we have

$$\mu_{b_j'}(y) = \left[ \prod_{i=1}^{n} \mu_{A_{ij}}(x_i) \right]^{1-\gamma_j} \times \frac{\gamma_j}{n} \quad (5.7)$$
CHAPTER 5. COMPENSATORY NEURO-FUZZY FILTER (CNFF)

5.3 The Structure of Compensatory Neuro-Fuzzy Filters

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.,

$$\text{net - input} = f(z_1^{(k)}, z_2^{(k)}, \ldots, z_p^{(k)}; w_1^{(k)}, w_2^{(k)}, \ldots, w_p^{(k)})$$ (5.8)

where \(z_i^{(k)}\) is the \(i\)th 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 [20]. This notation will also be used in the following equations. Each node also outputs an activation value as a function of its net-input

$$\text{output} = a[f(.)]$$ (5.9)

where \(a(.)\) denotes the activation function. The CNFF is a network of four-layer structure in the Figure 5.1, 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, i.e.,

$$f(x_i^{(1)}) = x_i^{(1)}$$ (5.10)

and \(a[f(.)] = f(.)\), where \(i\) is the input dimension index.

Layer 2: The nodes in this layer are term nodes that act as the Pseudo-Gaussian (PG) membership function [20]. They can react on the terms of the respective input-linguistic variables. For the \(j\)th rule node

$$f(z_i^{(2)}) = \exp \left( -\frac{(z_i^{(2)} - m_{ji})^2}{\sigma_{ji}^2} \right) U(z_i^{(2)}; -\infty, m_{ij}) + \exp \left( -\frac{(z_i^{(2)} - m_{ji})^2}{\sigma_{ji}^2} \right) U(z_i^{(2)}; m_{ij}, \infty)$$ (5.11)

and \(a[f(.)] = f(.)\), where \(U(z_i^{(2)}; a, b) = \begin{cases} 1 & \text{if } a \leq z_i^{(2)} < b \\ 0 & \text{otherwise.} \end{cases}\)

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

$$f(z_i^{(3)}) = \prod_{i=1}^{n} z_i^{(3)}$$ (5.12)

and \(a[f(.)] = f(.)\), where \(n\) is dimension number.

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

\[ f(z_i^{(4)}) = \sum_{j=1}^{M} w_j^{(3)} z_i^{(4)} \quad (5.13) \]

and \( a[f(.)] = f(.) \), where \( M \) is rule number; \( w_j^{(3)} \) is link weight.

### 5.3.1 The Online Learning Algorithm

The on-line learning algorithm 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 similar 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 back propagation 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 [20].

#### The Structure Learning Algorithm

The proposition of the structure learning algorithm is to decide proper fuzzy partitions by the input patterns. The procedure 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 Equation (5.11), that is used as the degree measure \( \mu_{A_i} \). In layer 3, the firing strength of fuzzy logic rule is obtained from Equation (5.12), which is used as the precondition part’s degree measure

\[ P = \prod_{j=1}^{M(t)} \mu_{A_i}^j \quad (5.14) \]

where \( i \) is the input dimension, \( i = 1, \ldots, n; \ j \) is the rule number, \( j = 1, \ldots, M(t) \), \( M(t) \) is the number of existing rules at time \( t \).

To avoid the newly generated membership function too similar 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, the new fuzzy
CHAPTER 5. COMPENSATORY NEURO-FUZZY FILTER (CNFF)

rule will be added in the CNFF. It can make neural fuzzy inference system to gain more performance. Therefore, we use similarity measure of asymmetric Gaussian membership functions to estimate the rule’s similarity degree [20]. Recall that for fuzzy sets A and B, their equivalence measure is calculated as

\[ E(A, B) \triangleq \frac{A \cap B}{A \cup B} \]  \hspace{1cm} (5.15)

The Parameter Learning Algorithm

After the structure network has been accordingly adjusted to the current training pattern, the network enters the parameter learning algorithm. The procedure of the parameter learning algorithm is to adjust the parameters of CNFF optimally with the same training pattern. The back propagation 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 [20]. The goal is to minimize the error function

\[ E = \frac{1}{2} \left( y^d(t) - y(t) \right)^2 \]  \hspace{1cm} (5.16)

where \( y^d(t) \) is the 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 generally used learning rule is

\[ w(t + 1) = w(t) - \eta \left( \frac{\partial E}{\partial w} \right) \]  \hspace{1cm} (5.17)

\[ \frac{\partial E}{\partial w} = \frac{\partial E}{\partial f} \cdot \frac{\partial f}{\partial w} \]

\[ = \frac{\partial E}{\partial u} \cdot \frac{\partial f}{\partial w} \]  \hspace{1cm} (5.18)

where \( \eta \) is the learning rate.

A Digital Communication System with AWGN and CCI

The discrete time model of digital communication system with AWGN and CCI is shown in Figure 4.4. \( H_0(z) \) is the desired channel and \( H_i(z), 1 \leq i \leq n, \) are the interfering co-channels. The impulse response of the channels and co-channels can be represented as

\[ H_i(z) = \sum_{j=0}^{p_i} a_{ij} z^{-j} \]  \hspace{1cm} (5.19)

Here \( p_i \) and \( a_{ij} \) are the length and tap weights of the \( i^{th} \) channel impulse response. The transmitted sequences, \( s_i(k); 1 \leq i \leq n, \) are mutually independent and are taken
CHAPTER 5. COMPENSATORY NEURO-FUZZY FILTER (CNFF)

from independent, identically distributed data set with values \{-1, 1\}. The input to the equalizer forms the observation vector from channel output [20]. Each of the components of this vector can be presented as

\[
x(k) = \hat{x}(k) + \hat{x}_{co}(k) + e(k)
\]  

(5.20)

where \(\hat{x}(k)\) is the desired received signal, and \(\hat{x}_{co}(k)\) is the interfering signal. The noise \(e(k)\) is assumed to be Gaussian with variance \(\sigma^2\) and is uncorrelated with the data. The task of the equalizer is to estimate the delayed transmitted sequence \(\hat{x}_{io}(k-d)\) based on the channel observation vector \(x(k) = [x(k), x(k-1), \ldots, x(k-N+1)]^T\). For the communication system with CCI and AWGN, the decision function of the Bayesian Equalizer is

\[
f(x(k)) = \sum_{i=1}^{n} \sum_{m=1}^{m_{co}} \prod_{l=1}^{p-1} w_i \exp \left( -\frac{1}{2} \frac{(x(k-l) - \hat{x}_i(k-l) - \hat{x}_{io}^m(k-l))^2}{\sigma_i^2 + \sigma_{io}^2} \right)
\]

(5.21)

where \(\hat{x}_{io}^m(k-l)\) is the \(l^{th}\) element of \(m^{th}\) co-channel state [20].

Channel Models and Simulation

The channel and the co-channels are characterized by their respective impulse responses.

\[
H_{ch}(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}
\]

(5.22)

\[
H_{co1}(z) = \lambda(0.5 + 0.81z^{-1} + 0.31z^{-2})
\]

(5.23)

The decision delay \(d\) is 1, and the input dimension \(n\) is 2. The initial parameters are chosen as: \(\eta = 0.01\) (learning rate), \(E = 0.6\) (similarity threshold), and the standard deviation of interference noise due to the \(i^{th}\) co-channel, \(\sigma\), is 0.5.

5.3.2 Simulation Results

The results of simulations (the decision boundaries) after the on-line training stopped at instances \(k = 50\) and \(k = 100\) are shown in Figures 5.2 and 5.3, respectively, with the number of rules generated being 5. To observe the actual BER, a realization of \(10^6\) points of the sequence \(s(k)\) and \(c(k)\) are used to test the BER of the trained network. The CNFF scheme is compared with Bayesian (optimal), Neural Network based and Linear Vector Quantization (LVQ) methods [20]. The BER curves for different SNR are shown in Figure 5.4.
CHAPTER 5. COMPENSATORY NEURO-FUZZY FILTER (CNFF)

Figure 5.2: Decision Boundaries of the CNFF for k=50; the boundary is marked by the mesh. The plot is a scattergram of the symbols received $x(k-1)$ and $x(k)$ at consecutive instances [20].

Figure 5.3: Decision Boundaries of the CNFF for k=100; the boundary is marked by the mesh. The plot is a scattergram of the symbols received $x(k-1)$ and $x(k)$ at consecutive instances [20].

5.4 Conclusion

In this chapter, we considered an alternative solution to the non-linear channel equalization problem. It was shown in Chapter 4 that the ANFIS based equalizer per-
forms nearly as good as the optimal Bayesian equalizer, as long as the SNR is greater than about 10 dB and the standard deviation of noise is low.

We considered yet another equalizer structure for the equalization of non-linear mobile cellular channels, which is the Compensatory Neuro-Fuzzy Filter (CNFF). It was found that the bit error rate (BER) versus SNR performance of the CNFF is also close to other Neural Network based equalizers. Both the ANFIS and CNFF methods are non-linear system approximators.