USC-SIPI REPORT #204

Fuzzy Adaptive Filter, with Application to Nonlinear Channel Equalization

by

Li-Xin Wang and Jerry M. Mendel

May 1992

Signal and Image Processing Institute
UNIVERSITY OF SOUTHERN CALIFORNIA
Department of Electrical Engineering-Systems
3740 McClintock Avenue, Room 404
Los Angeles, CA 90089-2564 U.S.A.

FUZZY ADAPTIVE FILTERS, WITH APPLICATION TO NONLINEAR CHANNEL EQUALIZATION

Li-Xin Wang and Jerry M. Mendel

May 21, 1992

Abstract

A fuzzy adaptive filter is constructed from a set of fuzzy IF-THEN rules which change adaptively to minimize some criterion function as new information becomes available. In this paper, two fuzzy adaptive filters are developed: one uses a recursive least squares (RLS) adaptation algorithm, and the other uses a least mean squares (LMS) adaptation algorithm. The RLS fuzzy adaptive filter is constructed through the following four steps: 1) define fuzzy sets in the filter input space $U \subset R^n$ whose membership functions cover U; 2) construct a set of fuzzy IF-THEN rules which either come from human experts or are determined during the adaptation procedure by matching input-output data pairs; 3) construct a filter based on the set of rules; and, 4) update the free parameters of the filter using the RLS algorithm. The design procedure of the LMS fuzzy adaptive filter is similar. The most important advantage of the fuzzy adaptive filters is that linguistic information (in the form of fuzzy IF-THEN rules) and numerical information (in the form of input-output pairs) can be combined into the filters in a uniform fashion. Finally, these two fuzzy adaptive filters are applied to nonlinear communication channel equalization problems; the simulation results show that: 1) without using any linguistic information, the RLS and LMS fuzzy adaptive filters are well-performing nonlinear adaptive filters (similar to polynomial and neural-net adaptive filters); 2) by incorporating some linguistic description (in fuzzy terms) about the channel into the fuzzy adaptive filters, the adaptation speed is greatly

improved; and, 3) the bit error rates of the fuzzy equalizers are close to that of the optimal equalizer.

1 INTRODUCTION

Filters are information processors. In practice, information usually comes from two sources: sensors which provide numerical data associated with a problem, and human experts who provide linguistic descriptions (often in the form of fuzzy IF-THEN rules) about the problem. Existing filters can only process numerical data, whereas existing expert systems can only make use of linguistic information; therefore, their successful applications are limited to problems (or portions of problems) where either linguistic rules or numerical data do not play a critical role. There are, however, a large number of practical problems in economics, seismology, management, etc., where both linguistic and numerical information are critical. At present, when we are faced with such problems, we use linguistic information, consciously or unconsciously, in the: choice among different filters, evaluation of filter performance, choice of filter orders, interpretation of filtering results, etc.. There are serious limitations to using linguistic information in this way, because for most practical problems the linguistic information (in its natural form) is not about which kind of filter should be chosen or what the order of the filter should be, etc., but is in the form of IF-THEN rules concerning fuzzy concepts like "small", "hot", "not very fast", "very large but not very very large", etc.. The purpose of this paper is to develop new kind of nonlinear adaptive filters, which we refer to as *fuzzy adaptive filters*, that make use of both linguistic and numerical information in their natural form, i.e., as fuzzy IF-THEN rules and input-output data pairs.

A fuzzy adaptive filter is constructed from a set of changeable fuzzy IF-THEN rules. These fuzzy rules come either from human experts or by matching input-output pairs through an adaptation procedure. The adaptive algorithms adjust the parameters of the membership functions which

characterize the fuzzy concepts in the IF-THEN rules, by minimizing some criterion functions. Two fuzzy adaptive filters are developed in this paper which use recursive least squares (RLS) and least mean squares (LMS) algorithms, respectively.

In Sections 2 and 3, the RLS and LMS fuzzy adaptive filters are designed, respectively. In Section 4, the two fuzzy adaptive filters are applied to nonlinear channel equalization problems. Section 5 concludes the paper.

2 RLS FUZZY ADAPTIVE FILTER

Our RLS fuzzy adaptive filter solves the following problem.

Problem 1: Consider a real-valued vector sequence $[\underline{x}(k)]$ and a real-valued scalar sequence $[d(k)]$, where $k = 0, 1, 2, ...$ is the time index, and $\underline{x}(k) \in U \equiv [C_1^-, C_1^+] \times [C_2^-, C_2^+] \times \cdots \times [C_n^-, C_n^+] \subset R^n$ (we call U and R the input and output spaces of the filter, respectively). At each time point k, we are given the values of $\underline{x}(k)$ and $d(k)$. The problem is: at each time point $k = 0, 1, 2, ...$, determine an adaptive filter $f_k : U \subset R^n \rightarrow R$ such that

$$J(k) = \sum_{i=0}^{k} \lambda^{k-i} [d(i) - f_k(\underline{x}(i))]^2 \tag{1}$$

is minimized, where $\lambda \in (0, 1]$ is a forgetting factor.

The above problem is quite general. As a particular example, consider the following time-series prediction problem: we measure the values of a bounded time-series $[y_k]$ at each time point $k = 0, 1, 2, ...$; at time point $k-1$, we want to determine a filter $f_{k-1} : U \rightarrow R$ such that $f_{k-1}[y(k-1), ..., y(k-n)]$ is an optimal prediction of $y(k)$ in some sense. For this problem, we have $\underline{x}(k) = [y(k-1), ..., y(k-n)]^T$, $d(k) = y(k)$, and the "in some sense" means to minimize the $J(k)$ of (1). If we constrain the f_k's to be linear functions, the problem becomes an FIR adaptive filter design problem [4,23]. If the f_k's are Volterra series expansions, we have an adaptive polynomial filter

design problem [11,17]. If the f_k's are multi-layer perceptrons or radial basis function expansions, the problem becomes the neural nets adaptive filter design problem [2, 3].

Design Procedure of the RLS Fuzzy Adaptive Filter:

Step 1: Define m_i fuzzy sets [24] in each interval $[C_i^-, C_i^+]$ of the input space U, which are labeled as F_i^{ji} $(i = 1, 2, ..., n;\ ji = 1, 2, ..., m_i)$, in the following way: the m_i membership functions $\mu_{F_i^{ji}}$ cover the interval $[C_i^-, C_i^+]$ in the sense that for each $x_i \in [C_i^-, C_i^+]$ there exists at least one $\mu_{F_i^{ji}}(x_i) \neq 0$. These membership functions are fixed and will not change during the adaptation procedure of Step 4.

Step 2: Construct a set of $\prod_{i=1}^n m_i$ fuzzy IF-THEN rules in the following form:

$$R^{(j1,...,jn)}: \ IF\ x_1\ is\ F_1^{j1}\ and\ \cdots\ and\ x_n\ is\ F_n^{jn},\ THEN\ d\ is\ G^{(j1,...,jn)}, \qquad (2)$$

where $\underline{x} = (x_1, ..., x_n)^T \in U$ (the filter input), $d \in R$ (the filter output), $ji = 1, 2, ..., m_i$ with $i = 1, 2, ..., n$, F_i^{ji}'s are the same labels of the fuzzy sets defined in Step 1, and the $G^{(j1,...,jn)}$'s are labels of fuzzy sets defined in the output space which are determined in the following way: if there are linguistic rules from human experts in the form of (2), set $G^{(j1,...,jn)}$ to be the corresponding linguistic terms of these rules; otherwise, set $\mu_{G^{(j1,...,jn)}}$ to be an arbitrary membership function over the output space R. *It is in this way that we incorporate linguistic rules into the fuzzy adaptive filter, i.e., we use linguistic rules to construct the initial filter.*

Step 3: Construct the filter f_k based on the $\prod_{i=1}^n m_i$ rules in Step 2 as follow:

$$f_k(\underline{x}) = \frac{\sum_{j1=1}^{m_1} \cdots \sum_{jn=1}^{m_n} \theta^{(j1,...,jn)} (\mu_{F_1^{j1}}(x_1) \cdots \mu_{F_n^{jn}}(x_n))}{\sum_{j1=1}^{m_1} \cdots \sum_{jn=1}^{m_n} (\mu_{F_1^{j1}}(x_1) \cdots \mu_{F_n^{jn}}(x_n))}, \qquad (3)$$

where $\underline{x} = (x_1, ..., x_n)^T \in U$, $\mu_{F_i^{ji}}$'s are membership functions defined in Step 1, and $\theta^{(j1,...,jn)} \in R$ is the point at which $\mu_{G^{(j1,...,jn)}}$ achieves its maximum value. Due to the way in which we defined

the $\mu_{F_i^{j_i}}$'s in Step 1, the denominator of (3) is nonzero for all the points of U, therefore the filter f_k of (3) is well-defined. Equation (3) is obtained by combining the $\prod_{i=1}^n m_i$ rules of Step 2 using product inference and centroid defuzzification [9,10]. Another way of interpreting (3) is as follows. For a given input $\underline{x} \in U$, we determine the filter output $f_k(\underline{x})$ as a weighted sum of the $\prod_{i=1}^n m_i$ points $\theta^{(j1,\ldots,jn)}$ in the output space at which the fuzzy sets $G^{(j1,\ldots,jn)}$ of the THEN parts of the $\prod_{i=1}^n m_i$ rules have maximum membership values; and, the weight $\mu_{F_1^{j1}}(x_1)\cdots\mu_{F_n^{jn}}(x_n)$ for $\theta^{(j1,\ldots,jn)}$ is proportional to the membership values of which $\underline{x}$ satisfies the IF part of $R^{(j1,\ldots,jn)}$. This is a reasonable filter because $\theta^{(j1,\ldots,jn)}$'s are the "most likely" points in the output space based on the $\prod_{i=1}^n m_i$ rules, and the point $\theta^{(j1,\ldots,jn)}$ should be given more weight if the given input point $\underline{x}$ satisfies the corresponding IF part "more likely" (in the sense of larger membership value).

In (3), the weights $\mu_{F_1^{j1}}(x_1)\cdots\mu_{F_n^{jn}}(x_n)$ are fixed functions of $\underline{x}$; therefore, the free design parameters of the fuzzy adaptive filter are the $\theta^{(j1,\ldots,jn)}$'s which are now collected as a $\prod_{i=1}^n m_i$-dimensional vector

$$\begin{aligned}\underline{\theta} \;=\; & (\theta^{(1,1,\ldots,1)},\ldots,\theta^{(m_1,1,\ldots,1)};\theta^{(1,2,1,\ldots,1)},\ldots,\theta^{(m_1,2,1,\ldots,1)};\ldots;\\ & \theta^{(1,m_2,1,\ldots,1)},\ldots,\theta^{(m_1,m_2,1,\ldots,1)};\ldots;\theta^{(1,m_2,\ldots,m_n)},\ldots,\theta^{(m_1,m_2,\ldots,m_n)})^T. \end{aligned} \tag{4}$$

Define the *fuzzy basis functions* [22]

$$p^{(j1,\ldots,jn)}(\underline{x}) = \frac{\mu_{F_1^{j1}}(x_1)\cdots\mu_{F_n^{jn}}(x_n)}{\sum_{j1=1}^{m_1}\cdots\sum_{jn=1}^{m_n}(\mu_{F_1^{j1}}(x_1)\cdots\mu_{F_n^{jn}}(x_n))}, \tag{5}$$

and collect them as a $\prod_{i=1}^n m_i$-dimensional vector $\underline{p}(\underline{x})$ in the same ordering as the $\underline{\theta}$ of (4), i.e.,

$$\begin{aligned}\underline{p}(\underline{x}) \;=\; & (p^{(1,1,\ldots,1)}(\underline{x}),\ldots,p^{(m_1,1,\ldots,1)}(\underline{x});p^{(1,2,1,\ldots,1)}(\underline{x}),\ldots,p^{(m_1,2,1,\ldots,1)}(\underline{x});\ldots;\\ & p^{(1,m_2,1,\ldots,1)}(\underline{x}),\ldots,p^{(m_1,m_2,1,\ldots,1)}(\underline{x});\ldots;p^{(1,m_2,\ldots,m_n)}(\underline{x}),\ldots,p^{(m_1,m_2,\ldots,m_n)}(\underline{x}))^T. \end{aligned} \tag{6}$$

Based on (4) and (6) we can now rewrite (3) as

$$f_k(\underline{x}) = \underline{p}^T(\underline{x})\underline{\theta}. \tag{7}$$

We see from (7) that f_k is linear in the parameter vector $\underline{\theta}$; therefore, we can use the fast-convergent RLS algorithm to update $\underline{\theta}$.

Step 4: Use the following RLS algorithm [4] to update $\underline{\theta}$: let the initial estimate of $\underline{\theta}$, $\underline{\theta}(0)$, be determined as in Step 2, and $P(0) = \sigma I$, where σ is a small positive constant, and I is the $\prod_{i=1}^{n} m_i - by - \prod_{i=1}^{n} m_i$ identity matrix; at each time point $k = 1, 2, ...,$ do the following:

$$\underline{\phi}(k) = \underline{p}(\underline{x}(k)), \tag{8}$$

$$P(k) = \frac{1}{\lambda}[P(k-1) - P(k-1)\underline{\phi}(k)(\lambda + \underline{\phi}^T(k)P(k-1)\underline{\phi}(k))^{-1}\underline{\phi}^T(k)P(k-1)], \tag{9}$$

$$K(k) = P(k-1)\underline{\phi}(k)[\lambda + \underline{\phi}^T(k)P(k-1)\underline{\phi}(k)]^{-1}, \tag{10}$$

$$\underline{\theta}(k) = \underline{\theta}(k-1) + K(k)(d(k) - \underline{\phi}^T(k)\underline{\theta}(k-1)), \tag{11}$$

where $[\underline{x}(k)]$ and $[d(k)]$ are the sequences defined above in Problem 1, $\underline{p}(*)$ is defined in (6), and λ is the forgetting factor in (1).

Some comments on this RLS fuzzy adaptive filter are now in order.

Remark 2.1: The RLS algorithm (9)-(11) is obtained by minimizing $J(k)$ of (1) with f_k constrained to be the form of (7). Because f_k of (7) is linear in the parameter, the derivation of (9)-(11) is the same as that for the FIR linear adaptive filter [4]; therefore, we omit the details.

Remark 2.2: The RLS algorithm (9)-(11) can be viewed as updating the $\prod_{i=1}^{n} m_i$ rules in the form of (2) by changing the "centers" $\theta^{(j1,...,jn)}$ of the THEN parts of these rules in the direction of minimizing the criterion function (1). We are allowed only to change these "centers." The

membership functions $\mu_{F_i^{j_i}}$ of the IF parts of the rules are fixed at the very beginning and are not allowed to change; therefore, a good choice of $\mu_{F_i^{j_i}}$'s is important to the success of the entire filter. In the next section, we will lighten this constraint by allowing the $\mu_{F_i^{j_i}}$'s also to change during the adaptation procedure.

Remark 2.3: It was proven in [18,22] that functions in the form of (3) are universal approximators, i.e., for any real continuous function g on the compact set U, there exists a function in the form of (3) such that it can uniformly approximate g over U to arbitrary accuracy. Consequently, our fuzzy adaptive filter is a powerful nonlinear adaptive filter in the sense that it has the capability of performing very difficult nonlinear filtering operation.

Remark 2.4: The fuzzy adaptive filter (7) performs a two stage operation on the input vector $\underline{x}$: first, it performs a nonlinear transformation $\underline{p}(*)$ on $\underline{x}$; then, the filter output is obtained as a linear combination of these transformed signals. In this sense, our fuzzy adaptive filter is similar to the radial basis function [3,16] and potential function [12] approaches. The unique feature of our fuzzy adaptive filter, which is not shared by other nonlinear adaptive filters, is that linguistic rules can be incorporated into the filter, as discussed next.

Remark 2.5: Linguistic information (in the form of the fuzzy IF-THEN rules of (2)) and numerical information (in the form of desired input-output pairs $(\underline{x}(k), d(k))$) are combined into the filter in the following way: due to Steps 2-4, linguistic IF-THEN rules are directly incorporated into the filter (3) by constructing the initial filter based on the linguistic rules; and, due to the adaptation Step 4, numerical pairs $(\underline{x}(k), d(k))$ are incorporated into the filter by updating the filter parameters such that the filter output "matches" the pairs in the sense of minimizing (1). It is natural and reasonable to assume that linguistic information from human experts is provided in the form of (2) because the rules of (2) state what the filter outputs should be in some input situations, where "what should be" and "some situations" are represented by linguistic terms which are characterized by fuzzy membership functions. On the other hand, it is obvious that the most

natural form of numerical information is provided in the form of input-output pairs $(\underline{x}(k), d(k))$.

Remark 2.6: By fixing the fuzzy membership functions on the input space U at the very begining, we obtained a nonlinear filter which is linear in the parameter; therefore, we could use the fast-convergent RLS algorithm in the adaptation procedure. The price paid is that we had to include all the $\prod_{i=1}^{n} m_i$ possible rules in the filter, because if a region of U is not covered by any rules and an input $\underline{x}$ to the filter happens to be in this region, then the filter response will be very poor. As a result, for problems of high dimension n and large m_i the computations involved in this fuzzy adaptive filter are intense, because at each time point k we need to perform the $\prod_{i=1}^{n} m_i$-dimensional matrix-to-vector multiplications of (9)-(11) and to evaluate the values of the $\prod_{i=1}^{n} m_i$ fuzzy basis functions of (8) (see also (6) and (5)). Although these computations are highly parallelizable, we may not be able to use the filter in some practical situations where computing power is limited; therefore, we will develop another fuzzy adaptive filter which involves much less computations, next.

3 LMS FUZZY ADAPTIVE FILTER

Our LMS fuzzy adaptive filter solves the following problem.

Problem 2: Consider the same input sequence $[\underline{x}(k)]$ and output sequence $[d(k)]$ as in Problem 1. The problem is: at each time point $k = 1, 2, ...$, determine an adaptive filter $f_k : U \rightarrow R$ such that

$$L = E[(d(k) - f_k(\underline{x}(k)))^2] \tag{12}$$

is minimized.

Design Procedure of the LMS Fuzzy Adaptive Filter:

Step 1: Define M fuzzy sets F_i^l in each interval $[C_i^-, C_i^+]$ of U with the following *Gaussian*

membership functions

$$\mu_{F_i^l}(x_i) = exp[-\frac{1}{2}(\frac{x_i - \bar{x}_i^l}{\sigma_i^l})^2], \tag{13}$$

where $l = 1, 2, ..., M$, $i = 1, 2, ..., n$, $x_i \in [C_i^-, C_i^+]$, and $\bar{x}_i^l$ and σ_i^l are free parameters which will be updated in the LMS adaptation procedure of Step 4.

Step 2: Construct a set of M fuzzy IF-THEN rules in the following form:

$$R^l: \ IF \ x_1 \ is \ F_1^l \ and \ \cdots \ and \ x_n \ is \ F_n^l, \ THEN \ d \ is \ G^l, \tag{14}$$

where $\underline{x} = (x_1, ..., x_n)^T \in U$, $d \in R$, F_i^l's are defined in Step 1, $M \leq \prod_{i=1}^n m_i$ (in general, $M << \prod_{i=1}^n m_i$), and G^l's are fuzzy sets defined in R which are determined as follows: *if there are linguistic rules in the form of (14), set F_i^l's and G^l to be the labels of these linguistic rules; otherwise, choose μ_{G^l} and the parameters $\bar{x}_i^l$ and σ_i^l arbitrarily.* The (parameters of) membership functions $\mu_{F_i^l}$ and μ_{G^l} in these rules will change during the LMS adaptation procedure of Step 4; therefore, the rules constructed in this step are initial rules of the fuzzy adaptive filter. As in the RLS fuzzy adaptive filter, we incorporate linguistic rules into the LMS fuzzy adaptive filter by constructing the initial filter based on these rules.

Step 3: Construct the filter $f_k : U \rightarrow R$ based on the M rules of Step 2 as follows:

$$f_k(\underline{x}) = \frac{\sum_{l=1}^M \theta^l (\prod_{i=1}^n \mu_{F_i^l}(x_i))}{\sum_{l=1}^M (\prod_{i=1}^n \mu_{F_i^l}(x_i))}, \tag{15}$$

where $\underline{x} = (x_1, ..., x_n)^T \in U$, $\mu_{F_i^l}$'s are the Gaussian membership functions of (13), and $\theta^l \in R$ is any point at which μ_{G^l} achieves its maximum value. The filter (15) is constructed in the same way as (3), and shares the same interpretation. Because we chose the membership functions $\mu_{F_i^l}(x_i)$ to be Gaussian functions which are nonzero for any $x_i \in [C_i^-, C_i^+]$, the denominator of (15) is nonzero for any $\underline{x} \in U$; therefore, the filter f_k of (15) is well-defined. Because the θ^l as well as $\bar{x}_i^l$ and σ_i^l are

free parameters, the filter (15) is nonlinear in the parameters.

Step 4: Use the following LMS algorithm [23] to update the filter parameters $\theta^l, \bar{x}_i^l$ and σ_i^l: let the initial $\theta^l(0), \bar{x}_i^l(0)$ and $\sigma_i^l(0)$ as determined in Step 2; at each time point $k = 1, 2, ...$, do the following:

$$\theta^l(k) = \theta^l(k-1) + \alpha[d(k) - f_k]\frac{a^l(k-1)}{b(k-1)}, \tag{16}$$

$$\bar{x}_i^l(k) = \bar{x}_i^l(k-1) + \alpha[d(k) - f_k]\frac{\theta^l(k-1) - f_k}{b(k-1)}a^l(k-1)\frac{x_i(k) - \bar{x}_i^l(k-1)}{(\sigma_i^l(k-1))^2}, \tag{17}$$

$$\sigma_i^l(k) = \sigma_i^l(k-1) + \alpha[d(k) - f_k]\frac{\theta^l(k-1) - f_k}{b(k-1)}a^l(k-1)\frac{(x_i(k) - \bar{x}_i^l(k-1))^2}{(\sigma_i^l(k-1))^3}, \tag{18}$$

where $a^l(k-1) = \prod_{i=1}^n exp[-\frac{1}{2}(\frac{x_i(k)-\bar{x}_i^l(k-1)}{\sigma_i^l(k-1)})^2]$, $b(k-1) = \sum_{l=1}^M a^l(k-1)$, $f_k = \frac{\sum_{l=1}^M \theta^l a^l(k-1)}{b(k-1)}$, α is a small positive stepsize, $l = 1, 2, ..., M$, and $i = 1, 2, ..., n$. Equations (16)-(18) are obtained by taking the gradient of L (12) (ignore the expection E) with respect to the parameters and using the specific formula of (15) and (13).

Some comments on this LMS fuzzy adaptive filter are now in order.

Remark 3.1: From Steps 2-4 we see that the initial LMS fuzzy adaptive filter is constructed based on linguistic rules from human experts and some arbitrary rules (in the sense that the parameters of membership functions $\mu_{F_i^l}$ and μ_{G^l} which characterize these rules are chosen arbitrarily). Both sets of rules are updated during the LMS adaptation procedure of Step 4 by changing the parameters in the direction of minimizing the L of (12). Because minimizing (12) can be viewed as matching the input-output pairs $[\underline{x}(k); d(k)]$, our LMS fuzzy adaptive filter combines both linguistic and numerical information in its design.

Remark 3.2: Because the LMS algorithm is a gradient algorithm, a good choice of initial parameters is very important to its convergence. Because we use linguistic information to choose the initial parameters, the adaptation procedure should converge quickly if the linguistic rules provide good instructions for how the filter should perform, i.e, good descriptions of the input-ouput

pairs $[\underline{x}(k); d(k)]$. Therefore, although LMS algorithms in general are slow to converge, our LMS algorithm in particular may converge fast, provided that there are sufficient linguistic rules.

Remark 3.3: The filter f_k of (15) can match any input-output pair $[\underline{x}(k); d(k)]$ to arbitrary accuracy by properly choosing the parameters $\theta^l, \bar{x}_i^l$ and σ_i^l, as we show next. For given $[\underline{x}(k); d(k)]$, let $\bar{x}_i^1 = x_i(k)$, $\bar{x}_i^l \neq x_i(k)$ for $l \neq 1$, and $\theta^1 = d(k)$; therefore, for $\underline{x} = \underline{x}(k)$ in (15), the weight $(\prod_{i=1}^n \mu_{F_i^1}(x_i(k)))$ for $\theta^1 = d(k)$ equals one for any choice of σ_i^1, and the other $M-1$ weights $(\prod_{i=1}^n \mu_{F_i^l}(x_i(k)))$ for θ^l with $l \neq 1$ can be arbitrarily close to zero if we choose all σ_i^l to be sufficiently small (see (13) and notice that $x_i(k) \neq \bar{x}_i^l$ for $l \neq 1$). As a result, $|d(k) - f_k(\underline{x}(k))|$ can be arbitrarily small. Because of this property and the freedom to update the parameters during the adaptation procedure, we can hope that we have a well-performing filter using only a small number of rules (i.e., we may choose $M << \prod_{i=1}^n m_i$).

4 APPLICATION TO NONLINEAR CHANNEL EQUALIZATION

Nonlinear distortion over a communication channel is now a significant factor hindering further increase in the attainable data rate in high-speed data transimission [1,6]. Because the received signal over a nonlinear channel is a nonlinear function of the past values of the transmitted symbols, and the nonlinear distortion varies with time and from place to place, effective equalizers for nonlinear channels should be nonlinear and adaptive.

In [1,6], ploynomial adaptive filters were developed for nonlinear channel equalization. In [2,3], multi-layer perceptrons and radial basis function expansions were used as adaptive equalizers for nonlinear channels. Because nonlinear channels include a very broad spectrum of nonlinear distortion, it is very difficult to say which nonlinear adaptive filter is dominantly better than the others. Therefore, it is worth trying other new nonlinear structures as pototypes of nonlinear adaptive

filters in addition to the existing Volterra series, multi-layer perceptron, radial basis function expansions, etc.. The RLS and LMS adaptive filters are such new nonlinear adaptive filters. In this section, we use them as equalizers for nonlinear channels.

The digital communication system considered in this paper is shown in Fig. 1, where the "channel" includes the effects of the transmitter filter, the transmission medium, the receiver matched filter, and other components. The transmitted data sequence $s(k)$ is assumed to be an independent sequence taking values from $\{-1, 1\}$ with equal probability. The inputs to the equalizer, $x(k), x(k-1), \cdots, x(k-n+1)$, are the channel outputs corrupted by an additive noise $e(k)$. The task of the equalizer at the sampling instant k is to produce an estimate of the transmitted symbol $s(k-d)$ using the information contained in $x(k), x(k-1), \cdots, x(k-n+1)$, where the integers n and d are known as the order and the lag of the equalizer, respectively.

We use the geometric formulation of the equalization problem due to [2,3]. Using similar notation to that in [2,3], define

$$P_{n,d}(1) = \{\underline{\hat{x}}(k) \in R^n | s(k-d) = 1\}, \tag{19}$$

$$P_{n,d}(-1) = \{\underline{\hat{x}}(k) \in R^n | s(k-d) = -1\}, \tag{20}$$

where

$$\underline{\hat{x}}(k) = [\hat{x}(k), \hat{x}(k-1), \cdots, \hat{x}(k-n+1)]^T, \tag{21}$$

$\hat{x}(k)$ is the noise-free output of the channel (see Fig. 1), and $P_{n,d}(1)$ and $P_{n,d}(-1)$ represent the two sets of possible channel noise-free output vectors $\underline{\hat{x}}(k)$ that can be produced from sequences of the channel inputs containing $s(k-d) = 1$ and $s(k-d) = -1$, respectively. The equalizer can be characterized by the function

$$g_k : R^n \rightarrow \{-1, 1\} \tag{22}$$

with

$$\hat{s}(k-d) = g_k(\underline{x}(k)), \tag{23}$$

where

$$\underline{x}(k) = [x(k), x(k-1), \cdots, x(k-n+1)]^T \tag{24}$$

is the observed channel output vector. Let $p_1[\underline{x}(k)|\underline{\hat{x}}(k) \in P_{n,d}(1)]$ and $p_{-1}[\underline{x}(k)|\underline{\hat{x}}(k) \in P_{n,d}(-1)]$ be the conditional probability density functions of $\underline{x}(k)$ given $\underline{\hat{x}}(k) \in P_{n,d}(1)$ and $\underline{\hat{x}}(k) \in P_{n,d}(-1)$, respectively. It was shown in [2,3] that the equalizer which is defined by

$$f_{opt}(\underline{x}(k)) = sgn[p_1(\underline{x}(k)|\underline{\hat{x}}(k) \in P_{n,d}(1)) - p_{-1}(\underline{x}(k)|\underline{\hat{x}}(k) \in P_{n,d}(-1))] \tag{25}$$

achieves the minimum bit error rate for the given order n and lag d, where $sgn(y) = 1(-1)$ if $y \geq 0$ ($y < 0$). If the noise $e(k)$ is zero-mean and Gaussian with covariance matrix

$$Q = E[(e(k), ..., e(k-n+1))(e(k), ..., e(k-n+1))^T]. \tag{26}$$

then from $x(k) = \hat{x}(k) + e(k)$ we have that

$$\begin{aligned} & p_1[\underline{x}(k)|\underline{\hat{x}}(k) \in P_{n,d}(1)] - p_{-1}[\underline{x}(k)|\underline{\hat{x}}(k) \in P_{n,d}(-1)] \\ = & \sum exp[-\frac{1}{2}(\underline{x}(k) - \underline{\hat{x}}_+)^T Q^{-1}(\underline{x}(k) - \underline{\hat{x}}_+)] - \sum exp[-\frac{1}{2}(\underline{x}(k) - \underline{\hat{x}}_-)^T Q^{-1}(\underline{x}(k) - \underline{\hat{x}}_-)] \end{aligned} \tag{27}$$

where the first (second) sum is over all the points $\underline{\hat{x}}_+ \in P_{n,d}(1)$ ($\underline{\hat{x}}_- \in P_{n,d}(-1)$).

Now consider the nonlinear channel

$$\hat{x}(k) = s(k) + 0.5s(k-1) - 0.9[s(k) + 0.5s(k-1)]^3, \tag{28}$$

and white Gaussian noise $e(k)$ with $E[e^2(k)] = 0.2$. For this case, the optimal decision region for $n = 2$ and $d = 0$,

$$[\underline{x}(k) \in R^2 | p_1[\underline{x}(k)|\hat{\underline{x}}(k) \in P_{2,0}(1)] - p_{-1}[\underline{x}(k)|\hat{\underline{x}}(k) \in P_{2,0}(-1)] \geq 0], \tag{29}$$

is shown in Fig. 2 as the shaded area. The elements of the sets $P_{2,0}(1)$ and $P_{2,0}(-1)$ are illustrated in Fig. 2 by the "o" and "*", respectively. From Fig. 2 we see that the optimal decision boundary for this case is severely nonlinear. We now use the RLS and LMS fuzzy adaptive filters to solve this specific equalization problem (channel (28), $e(k)$ white Gaussian with variance 0.2, equalizer order $n = 2$ and lag $d = 0$) under various conditions (Examples 1-4).

Example 1: Here we used the RLS fuzzy adaptive filter without any linguistic information. We chose $\lambda = 0.999, \sigma = 0.1$, $m_1 = m_2 = 9$, and $\mu_{F_i^j}(x_i) = exp[-\frac{1}{2}(\frac{x_i - \bar{x}_i^j}{0.3})^2]$ with $\bar{x}_i^j = -2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2$ for $j = 1, 2, ..., 9$, respectively, where $i = 1, 2$, $x_1 = x(k)$ and $x_2 = x(k-1)$. For the same realization of the sequence $s(k)$ and the same randomly chosen initial parameters $\underline{\theta}(0)$ (within [-0.5, 0.5]), we simulated the cases when the adaptation algorithm (9)-(11) stopped at: (i) $k = 30$, (ii) $k = 50$, and (iii) $k = 100$. The final decision regions, $[\underline{x}(k) \in R^2 | f_k(\underline{x}(k)) \geq 0]$, for the three cases are shown in Figs. 3-5, respectively. From Figs. 3-5 we see that the decision regions obtained from the RLS fuzzy adaptive filter tended to converge towards the optimal decision region.

Example 2: Next, we used the RLS fuzzy adaptive filter and incorporated the following linguistic information about the decision region. From the geometric formulation we see that the equalization problem is equivalent to determining a decision boundary in the input space of the equalizer. Suppose that there are human experts who are very familiar with the specific situation, such that although they cannot draw the specific decision boundary in the input space of the equalizer, they can assign degrees to different regions in the input space which reflect their belief that the regions should belong to 1-catalog or -1-catalog. Take Fig. 2 as an example. We see from Fig. 2 that

the difficulty is to determine which catalog the middle portion should belong to; in other words, as we move away from the middle portion, we have less and less uncertainty about which catalog the region should belong to. For example, for the left-most region in Fig. 2, we have more confidence that it should belong to the 1-catalog rather than the -1-catalog. Similarly, for the right-most region in Fig. 2, we have more confidence that it should belong to the -1-catalog rather than the 1-catalog. Also, we assume that the human experts know that a portion of the boundary is somewhere around $x(k) = -1.2$ for $x(k-1)$ less than 1 and around $x(k) = 1.2$ for $x(k-1)$ greater than 1. To make these observations more specific, we have the fuzzy rules shown in Fig. 6 where the membership functions N3, N2, etc. are the $\mu_{F_i^j}$'s defined in Example 1. We have 48 rules in Fig. 6, corresponding to the boxes with numbers; for example, the bottom-left box corresponds to the rule: "IF $x(k)$ is N4 and $x(k-1)$ is N4, THEN f_k is G," where f_k is the filter output, and the center of μ_G is 0.6. Because the filter output f_k is a weighted sum of these centers (see (3)), the numbers 0.6, 0.4, -0.4, -0.6 in Fig. 6 reflect our belief that the regions should correspond to the 1-catalog or the -1-catalog. For example, if the input point $[x(k), x(k-1)]$ falls in the left-most region of Fig. 6, then we have more confidence that the transmitted $s(k)$ should be 1 rather than -1, and, we represent this confidence by assigning the center of the fuzzy term in the corresponding THEN part to be 0.6.

It should be emphasized that the rules in Fig. 6 provide very fuzzy information about the decision region, because: 1) the regions are fuzzy, i.e., there are no clear boundaries between the regions, and 2) the numbers 0.6, 0.4, -0.4, -0.6 are conservative, i.e., they are away from the real transmitted values 1 or -1. We now show that although these rules are fuzzy, the speed of adaptation is greatly improved by incorporating them into the RLS fuzzy adaptive equalizer (filter). Figure 7 shows the final decision region determined by the RLS fuzzy adaptive filter, $[\underline{x}(k) \in R^2 | f_k(\underline{x}(k)) \geq 0]$ (shaded area), when the adaptation stopped at $k = 30$ after the rules in Fig. 6 were incorporated,

where the $\mu_{F_i^j}$'s and the sequence $s(k)$ were the same as those in Example 1. Comparing Figs. 7 and 3 we see that the adaptation speed was greatly improved by incorporating these fuzzy rules.

Example 3: Here we used the LMS fuzzy adaptive filter without any linguistic information. We chose: $M = 20, \alpha = 0.05$, the initial $\theta^l(0)$ randomly in $[-0.5, 0.5]$, $\bar{x}_i^j(0)$'s randomly in $[-2, 2]$, and $\bar{\sigma}_i^j(0)$'s randomly in $[0.1, 0.3]$. For the same sequence $s(k)$ (in Example 1) and the same initial parameters, we simulated the cases when the adaptation algorithm (16)-(18) stopped at: (i) $k = 100$, (ii) $k = 200$, and (iii) $k = 500$. The decision regions for the three cases are shown in Figs. 8-10, respectively.

Example 4: Next, we used the LMS fuzzy adaptive filter and incorporated some of the fuzzy rules from Fig. 6. We still chose $M = 20$ and $\alpha = 0.05$ and used the same $s(k)$ sequence. Since the filter is constructed from 20 rules, whereas Fig. 6 contains 48 rules, we can only choose a portion of the rules in Fig. 6 to construct the initial LMS fuzzy adaptive filter. We chose 20 rules arbitrarily from the boxes labeled 0.4 and -0.4. The final decision region for this case when the adaptation stopped at $k = 100$ is shown in Fig. 11. Comparing Figs. 11 and 8 we see that the adaptation speed was greatly improved by incorporating these fuzzy rules.

Example 5: Here, we considered the same situation as in Example 1, except that we chose $d = 1$ rather than $d = 0$. The optimal decision region for this case is shown in Fig. 12. Figures 13 and 14 show the final decision regions determined by the RLS fuzzy adaptive filter when the adaptation stoped at $k = 20$ and $k = 50$, respectively. We also simulated the LMS fuzzy adaptive filter for this case, and the final decision region was similar to Fig. 14 when the adaptation stoped at $k = 300$. Since we showed this kind of comparisons in Examples 1-4, we omit the details.

Example 6: In this final example, we compared the bit error rates achieved by the optimal equalizer (25) and the fuzzy adaptive equalizers for different signal-to-noise ratios, for the channel (28) with equalizer order $n = 2$ and lag $d = 1$. The optimal bit error rate was computed by applying the optimal equalizer (25) to a realization of 10^6 points of the sequences $s(k)$ and $e(k)$. For the

RLS fuzzy adaptive filter, we chose the filter parameters to be the same as in Example 1. For the LMS fuzzy adaptive filter, the parameters were chosen as in Example 3. We ran the RLS and LMS fuzzy adaptive filters for the first 1000 points in the same 10^6 point realization of $s(k)$ and $e(k)$ as for the optimal equalizer, and then used the trained fuzzy equalizers to compute the bit error rate for the same 10^6 point realization. Figure 15 shows the bit error rates of the optimal equalizer and the two fuzzy equalizers for different signal-to-noise ratios, where the bit error rate curves for the RLS and LMS fuzzy equalizers are indistinguishable. We see from Fig. 15 that the bit error rates of the fuzzy equalizers are very close to the optimal one.

5 CONCLUSIONS

In this paper, we developed two new nonlinear adaptive filters, namely: RLS and LMS fuzzy adaptive filters. The key elements of the fuzzy adaptive filters are a fuzzy system, which is constructed from a set of fuzzy IF-THEN rules, and an adaptive algorithm for updating the parameters in the fuzzy system, which, for the RLS fuzzy adaptive filter, is an RLS type of algorithm, and for the LMS fuzzy adaptive filter, is an LMS type of algorithm. The most important advantage of the fuzzy adaptive filters is that linguistic information from human experts (in the form of fuzzy IF-THEN rules) can be directly incorporated into the filters. If no linguistic information is available, the fuzzy adaptive filters become well-defined nonlinear adaptive filters, similar to the polynomial, neural nets, or radial basis function adaptive filters. We applied the two fuzzy adaptive filters to nonlinear channel equalization problems. Simulation results showed that: 1) the fuzzy adaptive filters worked quite well without using any linguistic information; 2) by incorporating some linguistic rules into the fuzzy adaptive filters, the adaptation speed was greatly improved; and, 3) the bit error rates of the fuzzy equalizers were close to that of the optimal equalizer.

Continuing the work in this paper, we may do the following: 1) in the RLS and LMS fuzzy

adaptive filters, we need to determine the order of the filters before processing; a good way to determine the order as well as the parameters is to use the lattice filter idea, i.e., we can develop a fuzzy lattice filter; and, 2) to extend the work in this paper to blind equalization, we may use unsupervised learning algorithms to determine the cluster centers of data, then view the centers, in some way, as learning samples, and use the fuzzy adaptive filters to process them.

6 REFERENCES

[1] Biglieri, E., A. Gersho, R. D. Gitlin and T. L. Lim, "Adaptive cancellation of nonlinear intersymbol interference for voiceband data transmission," *IEEE J. on Selected Areas in Communications*, Vol. SAC-2, No. 5, pp. 765-777, 1984.

[2] Chen, S., G. J. Gibson, C. F. N. Cowan and P. M. Grant, "Adaptive equalization of finite non-linear channels using multilayer perceptrons," *Signal Processing*, Vol. 20, pp. 107-119, 1990.

[3] Chen, S., 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, pp. 77-93, 1991.

[4] Cowan, C. F. N. and P. M. Grant (ed.), "Adaptive Filters," Prentice-Hall, Inc., New Jersey, 1985.

[5] Dubois, D. and H. Prade, "Fuzzy Sets and Systems: Theory and Applications," Academic Press, New York, 1980.

[6] Falconer, D. D., "Adaptive equalization of channel nonlinearities in QAM data Transmission systems," *The Bell System Technical Journal*, Vol. 57, No. 7, pp. 2589-2611, 1978.

[7] Klir, G. J. and T. A. Folger, "Fuzzy Sets, Uncertainty, and Information," Prentice Hall, New Jersey, 1988.

[8] Lapedes, A. and R. Farber, "Nonlinear signal processing using neural networks: prediction and system modeling," *Los Alamos National Laboratory Report, LA-UR-87-2662*, 1987.

[9] Lee, C. C., "Fuzzy logic in control systems: fuzzy logic controller, part I," *IEEE Trans. on Syst., Man, and Cybern.*, Vol.SMC-20, No.2, pp.404-418, 1990.

[10] Lee, C. C., "Fuzzy logic in control systems: fuzzy logic controller, part II," *IEEE Trans. on Syst., Man, and Cybern.*, Vol.SMC-20, No.2, pp.419-435, 1990.

[11] Mathews, V. J., "Adaptive ploynomial filters," *IEEE Signal Processing Magazine*, pp. 10-26, July, 1991.

[12] Meisel, W. S., "Potential functions in mathematical pattern recognition," *IEEE Trans. on Computers*, Vol. C-18, No. 10, pp. 911-918, 1969.

[13] Mulgrew, B. and C. F. N. Cowan, "Adaptive filters and Equalisers," Kluwer Academic Publishers, 1988.

[14] Narendra, K. S. and A. M. Annaswamy, *Stable Adaptive Systems*, Englewood Cliffs, NJ, Prentice-Hall, 1989.

[15] Pitas, I. and A. N. Venetsanopoulos, "Nonlinear Digital Filters," Kluwer Academic Publishers, 1990.

[16] Powell, M. J. D., "Radial basis functions for multivariable interpolation: A review," in *Algorithms for Approximation*, J. C. Mason and M. G. Cox, Eds., Oxford, pp. 143-167, 1987.

[17] Schetzen, M., "The Volterra and Wiener Theories of Nonlinear Filters," J. Wiley, 1980.

[18] Wang, L. X., "Fuzzy systems are universal approximators," *Proc. IEEE Conf. on Fuzzy Systems*, pp. 1163-1170, San Diego, 1992.

[19] Wang, L. X. and J. M. Mendel, "Generating fuzzy rules by learning from examples," *Proc. 6th IEEE International Symposium on Intelligent Control*, pp. 263-268, Washington D. C., 1991; also, *IEEE Trans. on Systems, Man, and Cybern.*, to be published, 1992.

[20] Wang, L. X. and J. M. Mendel, "Analysis and design of fuzzy logic controller," USC SIPI Report, No. 184, 1991; also, accepted for publication in *IEEE Trans. on Automatic Control*, 1992.

[21] Wang, L. X. and J. M. Mendel, "Back-propagation fuzzy systems as nonlinear dynamic

system identifiers," *Proc. IEEE International Conf. on Fuzzy Systems*, pp. 1409-1418, San Diego, 1992.

[22] Wang, L. X. and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least squares learning," *IEEE Trans. on Neural Networks*, to be published, Sept., 1992.

[23] Widrow, B. and S. D. Stearns, "*Adaptive Signal Processing*," Prentice-Hall, Englewood Cliffs, NJ, 1984.

[24] Zadeh, L. A., "Fuzzy sets," *Informat. Control*, Vol. 8, pp. 338-353, 1965.

[25] Zadeh, L. A., "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Trans. on Systems, Man, and Cybern.*, Vol. SMC-3, No. 1, pp. 28-44, 1973.

List of Figures

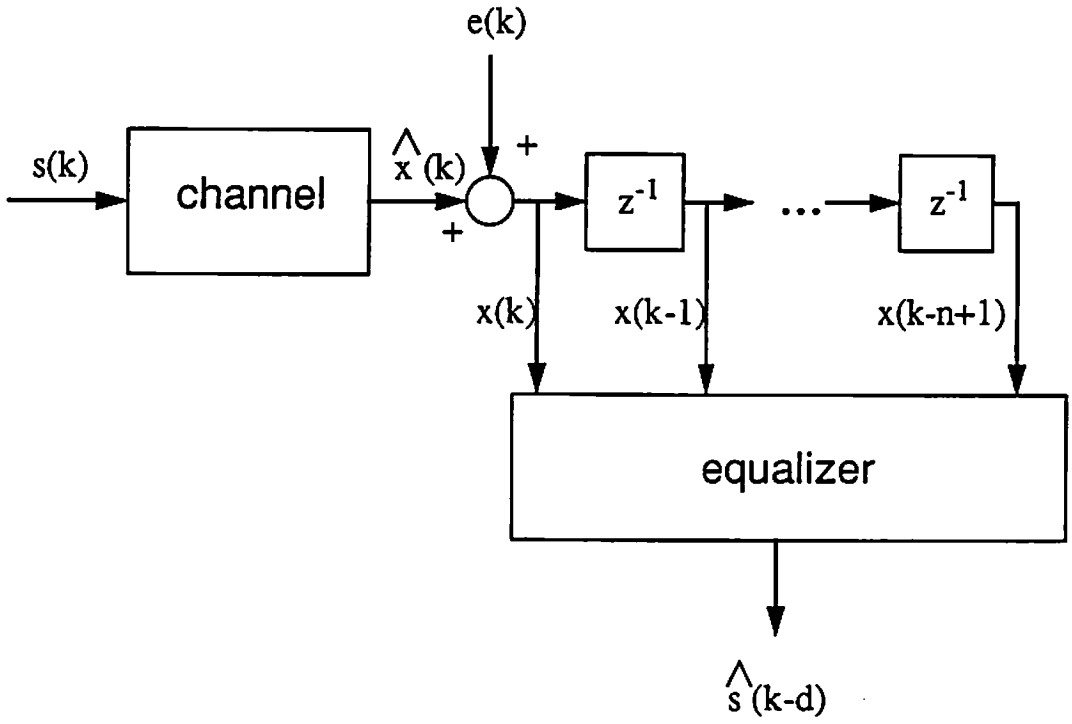

Figure 1: Schematic of data transmission system.

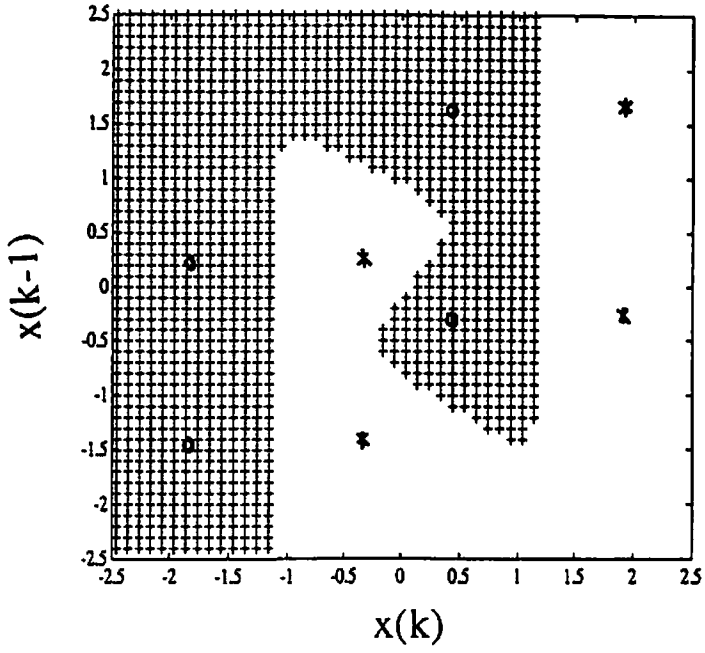

Figure 2: Optimal decision region for the channel (28), Gaussian white noise with variance $\sigma_e^2 = 0.2$, and equalizer order $n = 2$ and lag $d = 0$.

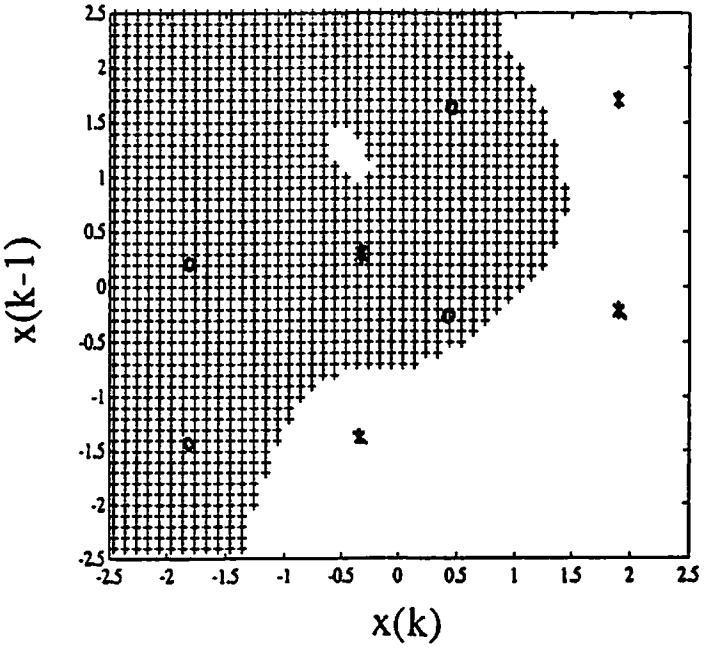

Figure 3: Decision region of the RLS fuzzy adaptive filter without using any linguistic information and when the adaptation stopped at $k = 30$.

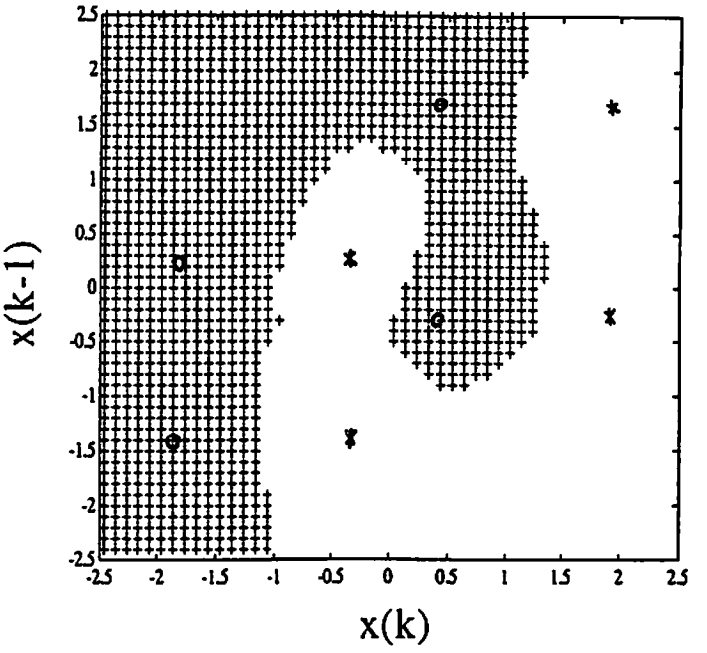

Figure 4: Decision region of the RLS fuzzy adaptive filter without using any linguistic information and when the adaptation stopped at $k = 50$.

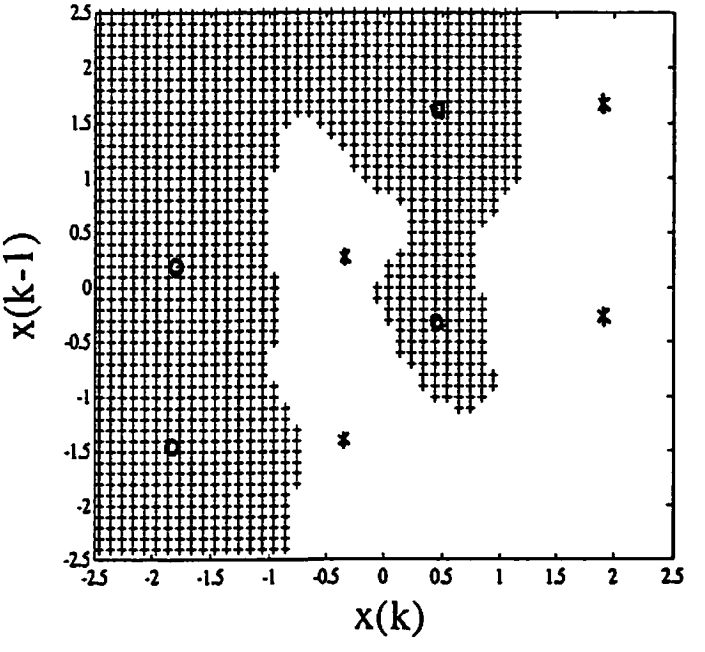

Figure 5: Decision region of the RLS fuzzy adaptive filter without using any linguistic information and when the adaptation stopped at $k = 100$.

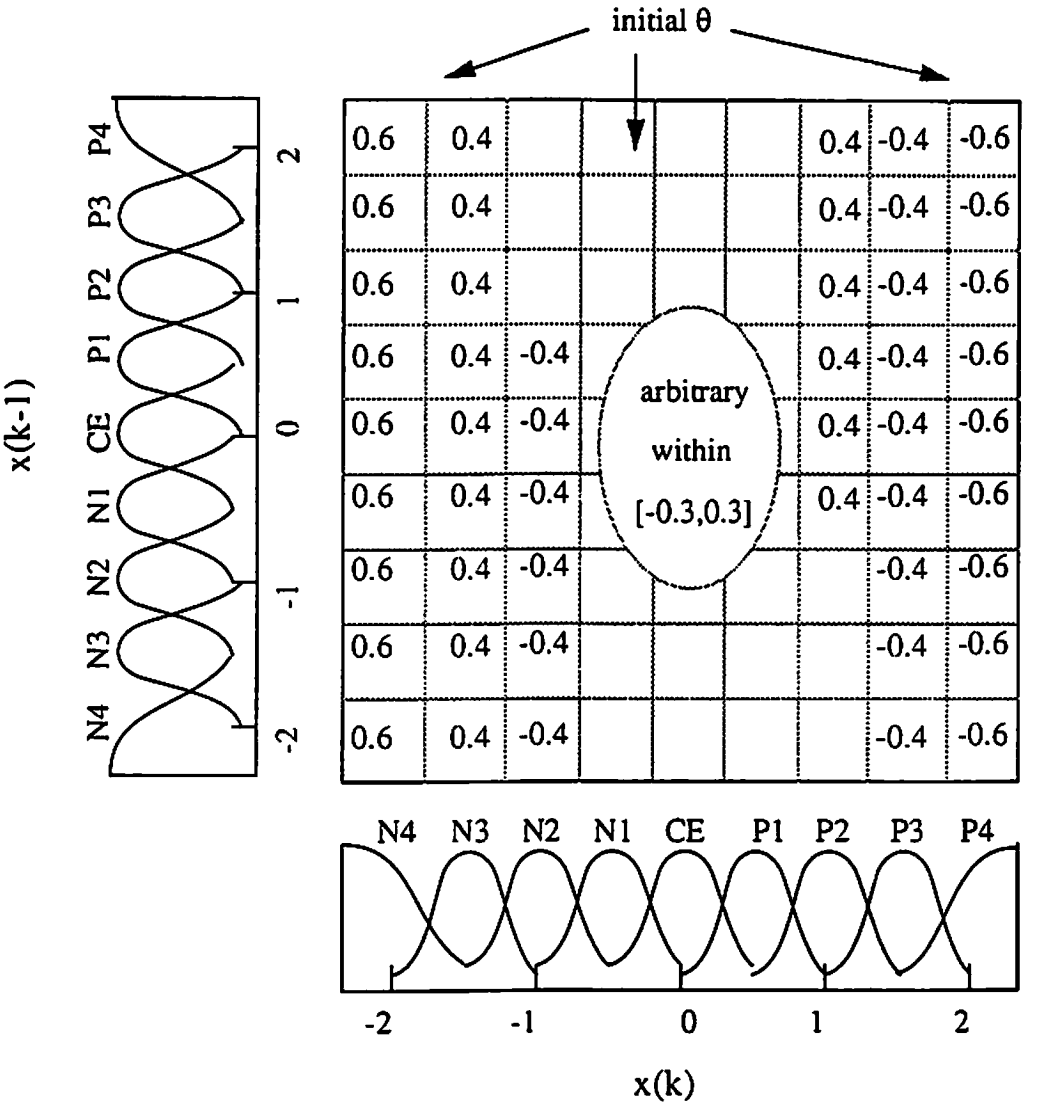

Figure 6: Illustration of some fuzzy rules about the decision region.

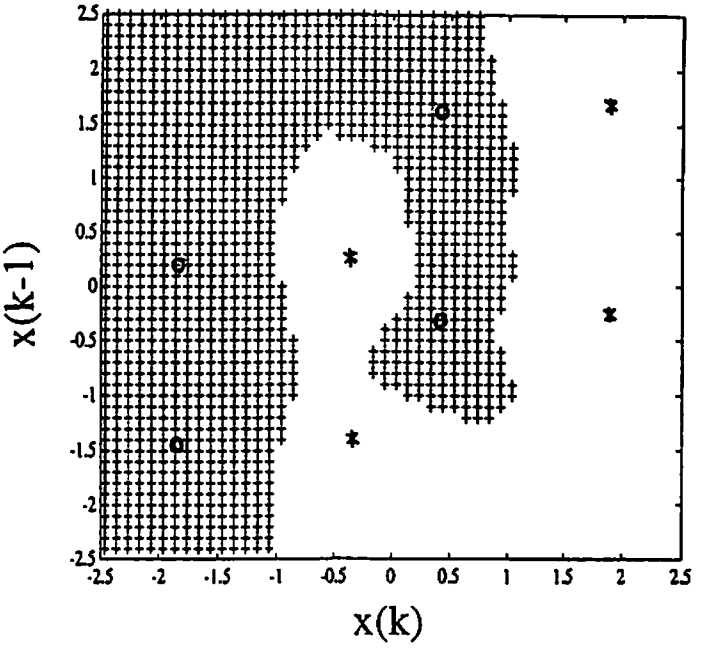

Figure 7: Decision region of the RLS fuzzy adaptive filter after incorporating the fuzzy rules illustrated in Fig. 6 and when the adaptation stopped at $k = 30$.

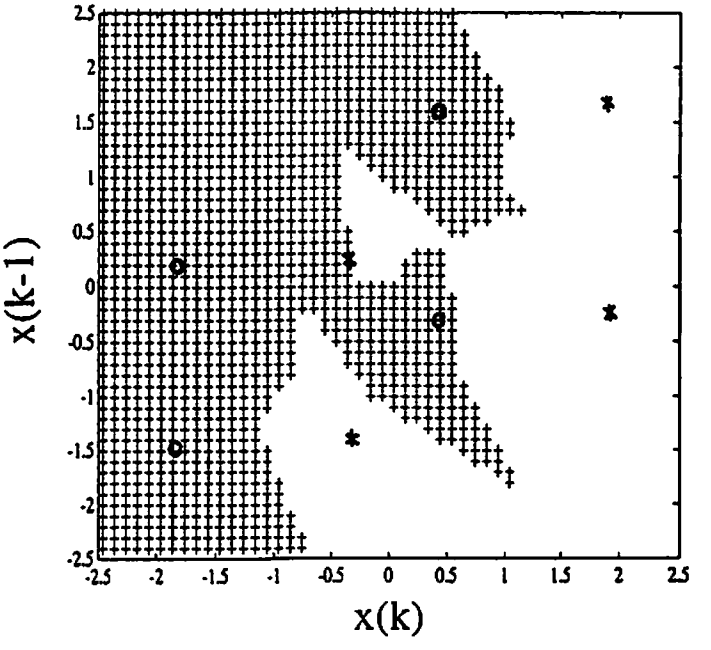

Figure 8: Decision region of the LMS fuzzy adaptive filter without using any linguistic information and when the adaptation stopped at $k = 100$.

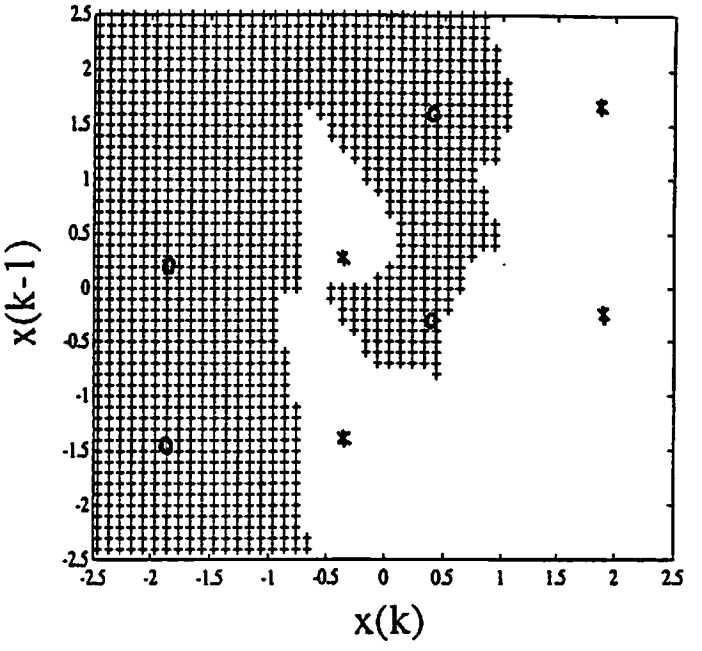

Figure 9: Decision region of the LMS fuzzy adaptive filter without using any linguistic information and when the adaptation stopped at $k = 200$.

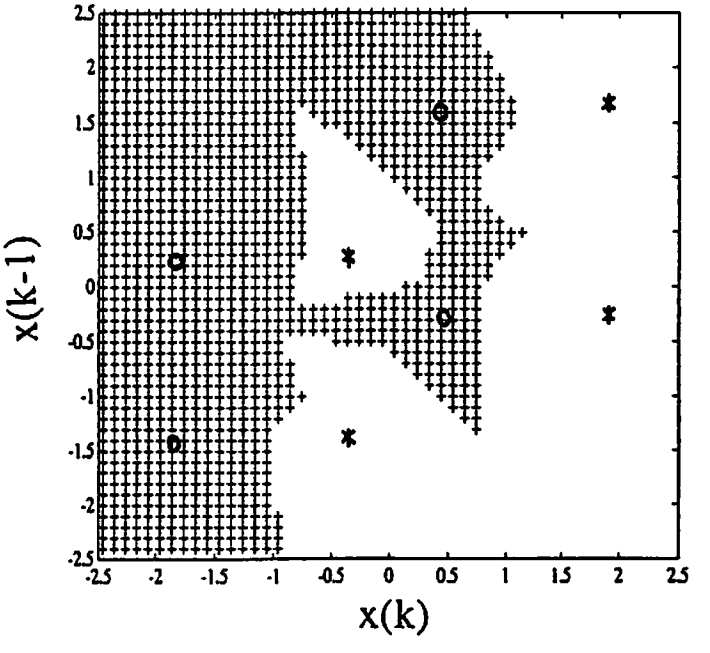

Figure 10: Decision region of the LMS fuzzy adaptive filter without using any linguistic information and when the adaptation stopped at $k = 500$.

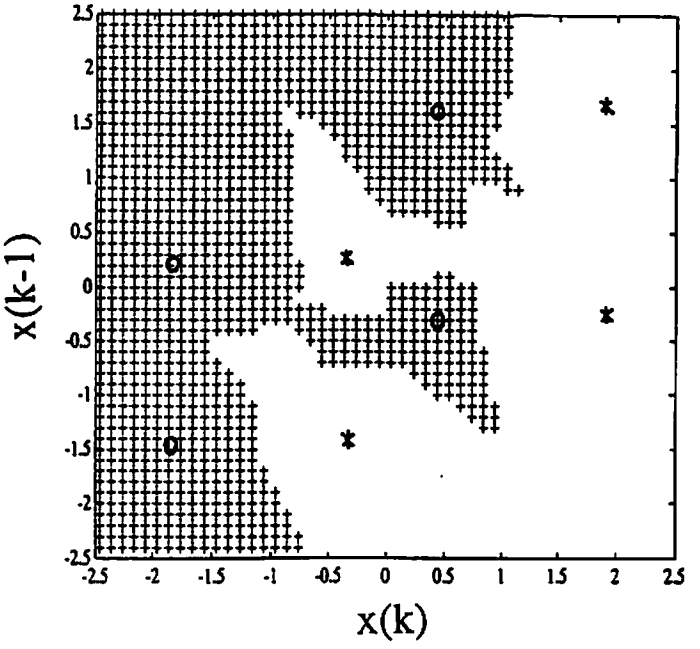

Figure 11: Decision region of the LMS fuzzy adaptive filter after incorporating some of the fuzzy rules illustrated in Fig. 6 and when the adaptation stopped at $k = 100$.

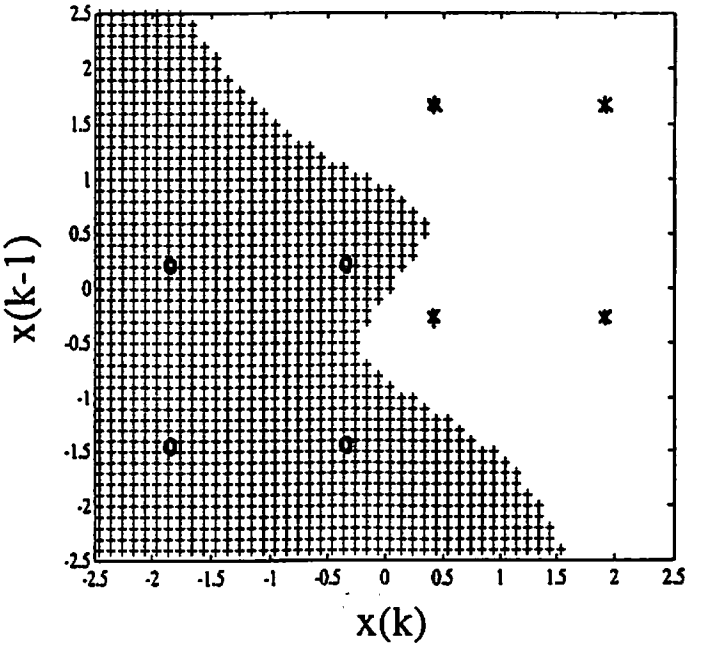

Figure 12: Optimal decision region for the channel (28), Gaussian white noise with variance $\sigma_e^2 = 0.2$, and equalizer order $n = 2$ and lag $d = 1$.

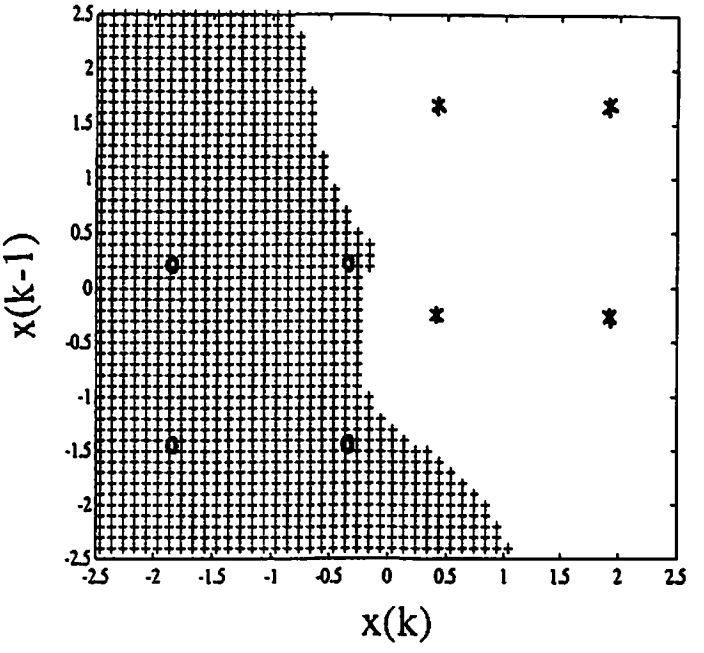

Figure 13: Decision region of the RLS fuzzy adaptive filter for the case of Example 5 when the adaptation stopped at $k = 20$.

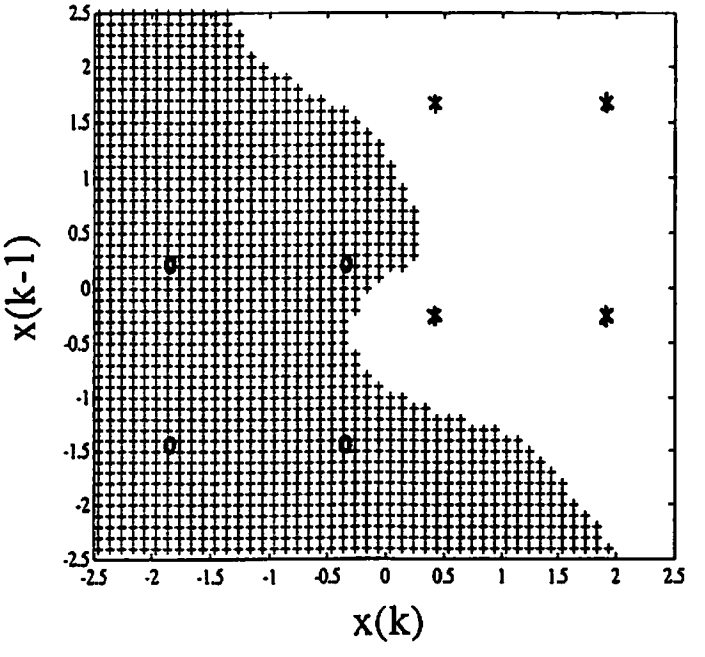

Figure 14: Decision region of the RLS fuzzy adaptive filter for the case of Example 5 when the adaptation stopped at $k = 50$.

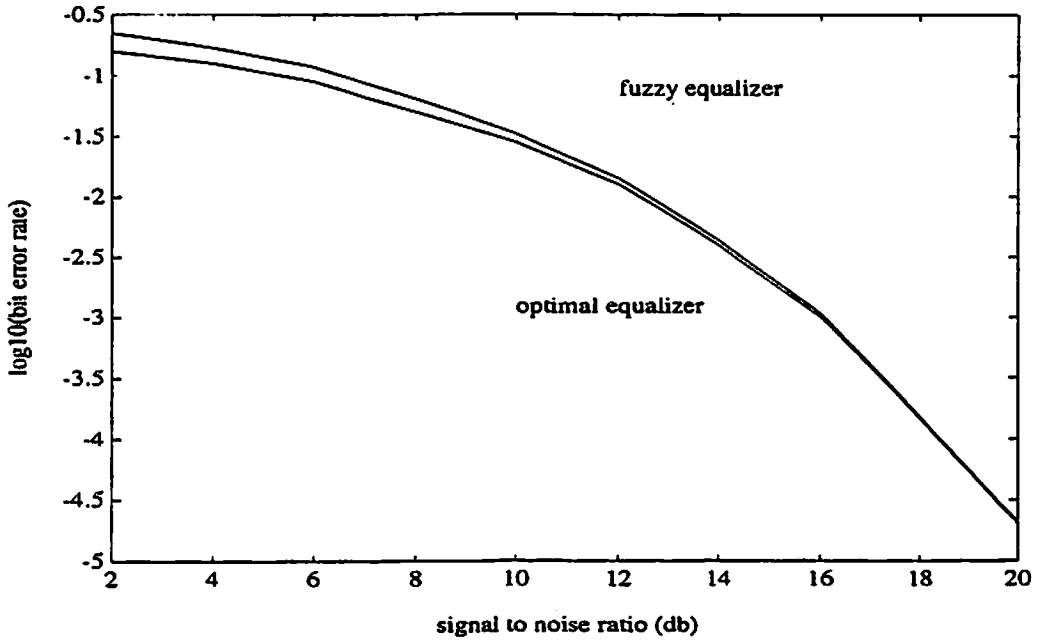

Figure 15: Comparison of bit error rates achieved by the optimal and fuzzy equalizers (Example 6).