Equalization of Nonlinear Time-Varying Channels Using Type-2 Fuzzy Adaptive Filters

Qilian Liang and Jerry M. Mendel, *Fellow, IEEE*

***Abstract*—This paper presents a new kind of adaptive filter: type-2 fuzzy adaptive filter (FAF); one that is realized using an unnormalized type-2 Takagi–Sugeno–Kang (TSK) fuzzy logic system (FLS). We apply this filter to equalization of a nonlinear time-varying channel and demonstrate that it can implement the Bayesian equalizer for such a channel, has a simple structure, and provides fast inference. A clustering method is used to adaptively design the parameters of the FAF. Two structures are used for the equalizer: transversal equalizer (TE) and decision feedback equalizer (DFE). A new decision tree structure is used to implement the decision feedback equalizer, in which each leaf of the tree is a type-2 FAF. This DFE vastly reduces computational complexity as compared to a TE. Simulation results show that equalizers based on type-2 FAFs perform much better than nearest neighbor classifiers (NNC) or equalizers based on type-1 FAFs.**

***Index Terms*—Channel equalization, decision feedback equalizer, decision tree, interval type-2 TSK fuzzy logic systems, time-varying channels, type-2 fuzzy adaptive filters.**

I. Introduction

FILTERS are information processors. A type-1 fuzzy adaptive filter (FAF), which can process numerical data and linguistic information in a natural form, i.e., as fuzzy IF-THEN rules and input–output (I/O) data pairs, was proposed and applied to nonlinear channel equalization in [33]. Wang and Mendel demonstrated that by incorporating some linguistic descriptors (fuzzy terms) about the channel into the FAF, its adaptation speed could be greatly improved and its bit error rate (BER) could be made close to the BER of the optimal equalizer. Since then, type-1 FAFs have been extensively used in signal processing and communications. For example, in channel equalization, Sarwal and Srinath [25] observed that a linear transversal filter requires a much larger training set to achieve the same error rate as achieved by a fuzzy logic equalizer. Lee [16] proposed a complex FAF for QAM constellation channel equalization, Patra and Mulgrew [23] used an FAF to implement a Bayesian equalizer, and also used it to eliminate cochannel interference [24].

Quite often, the information to be processed by a FAF is uncertain due to uncertain linguistic knowledge and uncertain numerical values. For example, in IF-THEN rules concerning fuzzy concepts such as *slowly time-varying*, *moderately time-varying*, or *rapidly time-varying*, experts may not agree on how to represent these linguistic labels using fuzzy membership functions. For example, experts frequently assign different intervals to the same label. So information coming from experts contains linguistic uncertainty. As another example, in mobile communications, the mappings between input and output data pairs are uncertain due to the channel dynamics. This numerical data uncertainty causes type-1 FAF and other nonlinear filters to perform poorly.

Linguistic and numerical uncertainties require new filters to handle them. In this paper, we propose a type-2 FAF to do this in which antecedent or consequent membership functions are type-2 fuzzy sets.

The concept of type-2 fuzzy sets was introduced by Zadeh [34] as an extension of the concept of an ordinary fuzzy set, i.e., a type-1 fuzzy set. Type-2 fuzzy sets have grades of membership that are themselves fuzzy [6]. A type-2 membership grade can be any subset in $[0,1]$—the *primary membership* and corresponding to each primary membership, there is a *secondary membership* (which can also be in $[0,1]$) that defines the possibilities for the primary membership. Type-2 fuzzy sets allow us to handle linguistic uncertainties, as typified by the adage "words can mean different things to different people [20]."

Karnik and Mendel ([10]–[14]) established a complete type-2 fuzzy logic system (FLS) theory to handle linguistic and numerical uncertainties. A type-2 FLS includes fuzzifier, rule base, fuzzy inference engine, and output processor. The output processor includes a type-reducer and a defuzzifier; it generates a type-1 fuzzy set output from the type reducer and a crisp number from the defuzzifier. A type-2 FLS (just as a type-1 FLS) is characterized by IF-THEN rules, but its antecedent or consequent sets are now type-2. General type-2 FLSs are computationally intensive because type-reduction is very intensive. Things simplify a lot when secondary membership functions (MFs) are interval sets (in this case, the secondary memberships are either zero or one). A theory and design methodology for interval type-2 Mamdani FLSs is given in [17] and a comparable theory and design methodology for normalized output interval type-2 TSK FLSs is given in [18].

Recently, it has been shown [30], [31] that normalizing the output of a TSK FLS, which increases complexity, is unnecessary in some cases. In this paper, we therefore propose a type-2 FAF, which is an unnormalized output interval type-2 TSK FLS, and we apply this type-2 FAF to equalization of time-varying channels.

Most of the work that has been done in the area of adaptive equalization over the past few decades has focused on time-invariant channels. In today's communication environment, e.g., mobile communication, the channels are time-varying because of fading. The classical equalizers do not perform well for

Manuscript received August 31, 1999; revised May 19, 2000.

The authors are with the Signal and Image Processing Institute, Department of Electrical Engineering Systems, University of Southern California, Los Angeles, CA 90089-2564 USA (e-mail: mendel@sipi.usc.edu).

Publisher Item Identifier S 1063-6706(00)08465-4.

rapidly fading channels. Cowan and Semnani [5] approached this problem by increasing the number of equalizer taps and choosing the coefficients from different ranges of values according to the amplitude of the distorted signals. Their method requires choosing a large number of coefficients and switching thresholds.

We interpret the time-varying nature of a channel as uncertainties in its coefficients; this interpretation matches the reason of existence for a type-2 FAF and motivates us to use a type-2 FAF as an adaptive equalizer for time-varying channels.

There are, however, two types of adaptive equalization: sequence estimation and symbol detection [2]. Sequence estimation has very high computation complexity, because channel estimation is needed. Symbol detection is essentially a classification problem in which the input base-band signal is mapped onto a feature space determined by the direct interpretation of a known training sequence, e.g., in [27], a nearest neighbor rule is used to classify the distorted signal and, in [22], a systematic feature space partitioning method is proposed to divide the entire feature space into two decision regions using a set of hyperplanes. In the classifier-based approach, channel estimation is unnecessary, which tremendously simplifies this approach. *In this paper, we focus on the classifier approach to adaptive equalization, and use an unnormalized output interval type-2 TSK FAF (type-2 FAF, for short) as the adaptive equalizer*.

In Section II, we provide some preliminaries that are needed for the rest of this paper, i.e., we review an unnormalized output type-1 TSK FLS and the extension principle, introduce the meet and addition operations for interval sets and summarize the concept of upper and lower membership functions (MFs) of a type-2 fuzzy set. Our main results for a type-2 FAF are given in Section III. In Section IV, we explain how a type-2 FAF can be applied to time-varying channel equalization and how it can be used as a transversal equalizer (TE). In Section V, we design a decision feedback equalizer (DFE) using a decision tree and a collection of type-2 FAFs and compare its performance with a nearest neighbor classifier and another DFE designed using a decision tree and a collection of type-1 FAFs. Conclusions and future research directions are given in Section VI.

In this paper, A is a type-1 fuzzy set and the membership grade of $x \in X$ in A is $\mu_A(x)$, which is a crisp number in $[0,1]$. A type-2 fuzzy set in X is $\tilde{A}$ and the membership grade of $x \in X$ in $\tilde{A}$ is $\mu_{\tilde{A}}(x)$, which is a type-1 fuzzy set in $[0,1]$. The elements of the domain of $\mu_{\tilde{A}}(x)$ are called *primary memberships* of x in $\tilde{A}$ and the memberships of the primary memberships in $\mu_{\tilde{A}}(x)$ are called *secondary memberships* of x in $\tilde{A}$. The latter defines the possibilities for the primary membership. $\mu_{\tilde{A}}(x)$, can be represented, for each $x \in X$, as $\mu_{\tilde{A}}(x) = \int_u f_x(u)/u, u \in J \subseteq [0,1]$; when the secondary MFs are type-1 interval sets, we call the type-2 set an *interval type-2 set*. $\sqcap$ denotes *meet* operation and $\sqcup$ denotes *join* operation. Meet and join are defined and explained in great detail in [12].

II. Preliminaries

The type-2 FAF developed in this paper is based on an unnormalized output interval type-1 TSK FLS and is obtained by applying Zadeh's [34] extension principle to the latter and by analyzing the interval-set nature of the resulting output. In this section, we provide a review of background materials that let us explain how to do this.

A. Unnormalized Output Type-1 TSK FLS

A type-1 TSK FLS, is described by fuzzy IF-THEN rules which represent I/O relations of a system. The most widely used type-1 TSK FLS (the one we direct our attention at) is a first-order type-1 TSK FLS. It has a rule base of M rules, each having p antecedents, where the ith rule R^i is expressed as

$$R^i\text{: IF } x_1 \text{ is } F_1^i \text{ and } x_2 \text{ is } F_2^i \text{ and } \ldots \text{ and } x_p \text{ is } F_p^i$$
$$\text{THEN } y^i = c_0^i + c_1^i x_1 + c_2^i x_2 + \cdots + c_p^i x_p$$

in which $i = 1, 2, \ldots, M$; c_j^i $(j = 0, 1, \ldots, p)$ are the consequent parameters; y^i is the output of the ith IF-THEN rule; and, F_k^i $(k = 1, 2, \ldots, p)$ are type-1 fuzzy sets. Given an input $(x_1, x_2, \ldots, x_p)$, the final output of the unnormalized first-order type-1 TSK model is inferred as [30], [31]

$$y = \sum_{i=1}^{M} f^i y^i \tag{1}$$

where f^i are rule firing strengths defined as

$$f^i = \mathcal{T}_{k=1}^{p} \mu_{F_k^i}(x_k) \tag{2}$$

and $\mathcal{T}$ denotes a t-norm.

When Gaussian MFs and product t-norm are used, i.e.,

$$\mu_{F_k^i}(x_k) = \exp\left[-\frac{1}{2}\left(\frac{x_k - m_k^i}{\sigma_k^i}\right)^2\right] \tag{3}$$

then (1) can be expressed as

$$y = \sum_{i=1}^{M} y^i \prod_{k=1}^{p} \exp\left[-\frac{1}{2}\left(\frac{x_k - m_k^i}{\sigma_k^i}\right)^2\right]. \tag{4}$$

Observe that (4) is identical to the output formula for a radial basis function (RBF) network [2] when Gaussian MFs are used as the RBFs. This kind of RBF network has been applied to Bayesian equalization [2]–[4]. Later in this paper, the unnormalized output type-1 TSK FLS in (4) will be used as a type-1 FAF equalizer and its performance will be compared with that of a type-2 FAF.

B. Extension Principle

The extension principle [34] allows the domain of definition of a mapping or a relation to be extended from points in the universe of discourse U to fuzzy subsets of U. When we need to extend an operation of the form $f(\theta_1, \ldots, \theta_n)$ to an operation $f(A_1, \ldots, A_n)$, we will not extend the individual operations, like multiplication, addition, etc., involved in f; rather, we will use the following definition [12]:

$$f(A_1, \ldots, A_n) = \int_{\theta_1} \cdots \int_{\theta_n} \mu_{A_1}(\theta_1) \star \cdots \star \mu_{A_n}(\theta_n) / f(\theta_1, \ldots, \theta_n) \tag{5}$$

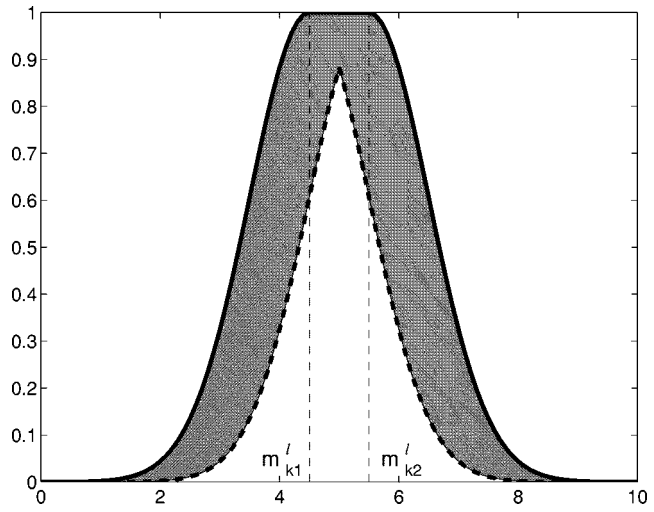


Fig. 1. The type-2 MF for Example 1. The thick solid lines denote upper MFs and the thick dashed lines denote lower MFs. The shaded regions are the footprints of uncertainty for interval secondaries. The centers of Gaussian MFs vary from 4.5 to 5.5.

where $\theta_i \in A_i$ for $i = 1, \ldots, n$, and $\star$ denotes a t-norm. For example, if $f(\theta_1, \theta_2) = [\theta_1\theta_2]/[\theta_1 + \theta_2]$, we write the extension of f to type-1 sets A_1 and A_2 as

$$f(A_1, A_2) = \int_{\theta_1} \int_{\theta_2} \mu_{A_1}(\theta_1) \star \mu_{A_2}(\theta_2) \Big/ \frac{\theta_1\theta_2}{\theta_1 + \theta_2} \qquad (6)$$

where $\theta_i \in \mu_{A_i}$ for $i = 1, 2$.

C. Meet and Addition for Interval Sets

The membership grade of $x \in X$ in a type-2 fuzzy set $\tilde{A}$ is $\mu_{\tilde{A}}(x)$; it is a type-1 fuzzy set in $[0, 1]$. In an interval type-2 fuzzy set (see Fig. 1), $\mu_{\tilde{A}}(x)$ is a type-1 interval set. The *meet* and *addition* operations, which will be needed to implement a type-2 FAF, can be greatly simplified for interval type-1 sets, by using (Fig. 1).

Theorem 1 (Meet of Interval Sets Under Minimum or Product t-Norms): The *meet* under minimum or product t-norms of n interval type-1 sets $F_1, \ldots, F_n$, $\sqcap_{i=1}^n F_i$, having domains $[l_1, r_1], \ldots, [l_n, r_n]$, respectively, where $[l_i, r_i] \subseteq [0, 1]$ $i = 1, 2, \ldots, n$, is an interval set with domain $[(l_1 \star l_2 \star \cdots \star l_n), (r_1 \star r_2 \star \cdots \star r_n)]$.

For the proof of this theorem see [13].

Theorem 2 (Addition of Interval Type-1 Sets): Given n interval type-1 sets $F_1, \ldots, F_n$, with domains $[l_1, r_1], \ldots, [l_n, r_n]$, respectively, where $[l_i, r_i] \subseteq [0, 1]$ $i = 1, 2, \ldots, n$, their linear combination $\sum_{i=1}^n \alpha_i F_i$, where α_i $(i = 1, \ldots, n)$ is a crisp constant, is also an interval type-1 set $[\sum_{i=1}^n \alpha_i l_i, \sum_{i=1}^n \alpha_i r_i]$.

The proof of Theorem 2 is given in [12].

Observe from Theorems 1 and 2 that the meet and addition operations of interval sets are determined just by the two end points of each interval set. In a type-2 FAF, the two end points are associated with two type-1 MFs to which we refer as *upper* and *lower MFs*.

D. Upper and Lower MFs of Type-2 MFs

For convenience in defining the upper and lower MFs of a type-2 MF, we first give the definition of *footprint of uncertainty of a type-2 MF*.

Definition 1 (Footprint of Uncertainty of a Type-2 MF): Uncertainty in the primary membership grades of a type-2 MF consists of a bounded region that we call the *footprint of uncertainty* of a type-2 MF (e.g., see Fig. 1). It is the union of all primary membership grades.

Definition 2 (Upper and Lower MFs): An upper MF and a lower MF are two type-1 MFs, which are bounds for the footprint of uncertainty of an interval type-2 MF. The upper MF is a subset that has the maximum membership grade of the footprint of uncertainty and the lower MF is a subset which has the minimum membership grade of the footprint of uncertainty.

We use an overbar (underbar) to denote the upper (lower) MF. For example, the upper and lower MFs of the interval type-2 fuzzy set $\mu_{\tilde{F}_k^l}(x_k)$ (used in the next section) are $\bar{\mu}_{F_k^l}(x_k)$ and $\underline{\mu}_{F_k^l}(x_k)$ and $\mu_{\tilde{F}_k^l}(x_k)$ can be expressed as

$$\mu_{\tilde{F}_k^l}(x_k) = \int_{w^l \in \left[\underline{\mu}_{F_k^l}(x_k), \bar{\mu}_{F_k^l}(x_k)\right]} 1/w^l. \qquad (7)$$

Example 1: Gaussian Primary MF with Uncertain Mean: Consider the case of a Gaussian primary MF having a fixed standard deviation σ_k^l and an uncertain mean that takes on values in $[m_{k1}^l, m_{k2}^l]$, i.e.,

$$\mu_k^l(x_k) = \exp\left[-\frac{1}{2}\left(\frac{x_k - m_k^l}{\sigma_k^l}\right)^2\right], \quad m_k^l \in \left[m_{k1}^l, m_{k2}^l\right] \qquad (8)$$

where $k = 1, \ldots, p; p$ is the number of antecedents $l = 1, \ldots, M$ and M is the number of rules. The upper MF, $\bar{\mu}_k^l(x_k)$, is (see Fig. 1)

$$\bar{\mu}_k^l(x_k) = \begin{cases} \mathcal{N}\left(m_{k1}^l, \sigma_k^l; x_k\right), & x_k < m_{k1}^l \\ 1, & m_{k1}^l \le x_k \le m_{k2}^l \\ \mathcal{N}\left(m_{k2}^l, \sigma_k^l; x_k\right), & x_k > m_{k2}^l \end{cases} \qquad (9)$$

where, for example, $\mathcal{N}(m_{k1}^l, \sigma_k^l; x_k) \triangleq \exp(-(1/2)(x_k - m_{k1}^l/\sigma_k^l)^2)$. The lower MF $\underline{\mu}_k^l(x_k)$ is (see Fig. 1)

$$\underline{\mu}_k^l(x_k) = \begin{cases} \mathcal{N}\left(m_{k2}^l, \sigma_k^l; x_k\right), & x_k \le \frac{m_{k1}^l + m_{k2}^l}{2} \\ \mathcal{N}\left(m_{k1}^l, \sigma_k^l; x_k\right), & x_k > \frac{m_{k1}^l + m_{k2}^l}{2}. \end{cases} \qquad (10)$$

□

We use the results of this example later in Sections IV-E and V.

III. Type-2 FAF

Our type-2 FAF for channel equalization is obtained by generalizing the unnormalized output type-1 TSK FLS to a type-2 TSK FLS. For equalization, the antecedents of the type-1 TSK FLS are generalized to type-2 fuzzy sets, whereas the consequent is unchanged (i.e., it is a crisp number). Note that this is not the only way to generalize the Section II-A TSK FLS to a type-2 TSK FLS, e.g., another generalization is from type-1

antecedents and type-0 consequent to type-2 antecedents and type-1 consequent. The reason we use a type-0 consequent in our FAF for equalization is because the consequent is determined by the channel state category, which, as explained in Section IV-B, is a crisp value.

In a type-2 FAF with a rule base of M rules, where each rule has p antecedents, the ith rule R^i is denoted as

$$R^i\text{: IF } x_1 \text{ is } \tilde{F}_1^i \text{ and } x_2 \text{ is } \tilde{F}_2^i \text{ and } \dots \text{ and } x_p \text{ is } \tilde{F}_p^i$$
$$\text{THEN } y^i = c_0^i + c_1^i x_1 + c_2^i x_2 + \cdots + c_p^i x_p$$

where $i = 1, 2, \dots, M$; c_j^i $(j = 0, 1, \dots, p)$ are the consequent parameters that are crisp numbers; y^i is an output from the ith IF-THEN rule, which is a crisp number and the $\tilde{F}_k^i$ $(k = 1, 2, \dots, p)$ are type-2 fuzzy sets. Given an input $\mathbf{x} = [x_1, x_2, \dots, x_p]^T$, the firing strength of the ith rule is ([10]–[12], [14])

$$F^i = \mu_{\tilde{F}_1^i}(x_1) \sqcap \mu_{\tilde{F}_2^i}(x_2) \sqcap \cdots \sqcap \mu_{\tilde{F}_p^i}(x_p). \tag{11}$$

The final output of the type-2 FAF is obtained by applying the *extension principle* to (1), as described in Section II-B, i.e.,

$$Y(F^1, \dots, F^M) = \int_{f^1} \cdots \int_{f^M} \mathcal{T}_{i=1}^M \mu_{F^i}(f^i) \Big/ \sum_{i=1}^M f^i y^i \tag{12}$$

where M is the number of rules fired, $f^i \in F^i$, and $\mathcal{T}$ indicates the chosen t-norm. Y is called an *extended weighted average*; it reveals the uncertainty at the output of a type-2 FLS due to antecedent uncertainties and is itself a type-1 fuzzy set.

Here, we focus on the very practical case when interval type-2 sets are used in the antecedents, which means $\mu_{\tilde{F}_k^i}(x_k)$ $(k = 1, \dots, p)$ is an interval set, and we denote

$$\mu_{\tilde{F}_k^i}(x_k) = \left[\underline{\mu}_{\tilde{F}_k^i}(x_k), \bar{\mu}_{\tilde{F}_k^i}(x_k)\right] \triangleq \left[\underline{f}_k^i, \bar{f}_k^i\right]. \tag{13}$$

Our type-2 FAF is then computed using results in the following.

Theorem 3:

1) In an interval type-2 FAF with meet under minimum or product t-norm, the firing strength in (11) for rule R^i is an interval set $F^i = [\underline{f}^i, \bar{f}^i]$, where $(i = 1, \dots, M)$

$$\underline{f}^i = \underline{\mu}_{\tilde{F}_1^i}(x_1) \star \cdots \star \underline{\mu}_{\tilde{F}_p^i}(x_p) = \mathcal{T}_{k=1}^p \underline{f}_k^i \tag{14}$$

and

$$\bar{f}^i = \bar{\mu}_{\tilde{F}_1^i}(x_1) \star \cdots \star \bar{\mu}_{\tilde{F}_p^i}(x_p) = \mathcal{T}_{k=1}^p \bar{f}_k^i. \tag{15}$$

2) The extended weighted average $Y(F^1, \dots, F^M)$ is also an interval set $[y_l, y_r]$ where

$$y_r = \sum_{i=1}^M \bar{f}^i y^i \tag{16}$$

$$y_l = \sum_{i=1}^M \underline{f}^i y^i \tag{17}$$

and

$$y^i = c_0^i + c_1^i x_1 + c_2^i x_2 + \cdots + c_p^i x_p. \tag{18}$$

3) The defuzzified output of our type-2 FAF is

$$y = \sum_{i=1}^M y^i (\underline{f}^i + \bar{f}^i)/2. \tag{19}$$

Proof:

1) Based on (11), (13), and Theorem 1, we see that F^i is an interval set. Applying Theorem 1 to (11), we obtain (14) and (15).
2) Because F^i $(i = 1, 2, \dots, M)$ are interval type-1 sets, i.e., $\mu_{F^i}(f^i) = 1$, (12) simplifies to

$$Y(F^1, \dots, F^M) = \int_{f^1} \cdots \int_{f^M} 1 \Big/ \sum_{i=1}^M f^i y^i = [y_l, y_r]. \tag{20}$$

Because $f^i \in F^i = [\underline{f}^i, \bar{f}^i]$ and y^i is a crisp value, applying Theorem 2 to (20), we obtain (16) and (17).
3) Because Y is an interval set, we defuzzify it using the average of y_l and y_r. Hence, the defuzzified output of the type-2 FAF is $y = (y_l + y_r/2)$ which is easily shown to be given by (19).

□

IV. Transversal Equalizer for Time-Varying Channels Using a Type-2 FAF

A. Introduction

Chen *et al.* [2] used an RBF network to implement a Bayesian equalizer for a time-invariant channel, and demonstrated that it has an identical structure to the optimal Bayesian symbol decision equalizer. Although they provided a decision-directed clustering algorithm to track the changes of channel states for a time-varying channel, no equalizer was designed for such channels. Patra and Mulgrew [23] observed that the Bayesian decision solution can be represented using a normalized formula which has an identical structure to a type-1 fuzzy filter, and used it to again design a Bayesian equalizer for a time-invariant channel. In this section, we explain why a type-2 FAF is needed for equalization of a time-varying channel, and design a transversal equalizer using a type-2 FAF.

B. Preliminaries for Channel Equalization

The block diagram of a baseband communication system that is subject to intersymbol interference (ISI) and additive Gaussian noise (AGN) is shown in Fig. 2, where $s(k)$ is the symbol to be transmitted, $e(k)$ is the noise, the channel order is n $(n + 1$ taps), and time-varying tap coefficients are $a_i(k)$ $(i = 0, 1, \dots, n)$. Hence, $r(k)$ can be represented [8] as

$$r(k) = \sum_{i=0}^n a_i(k) s(k - i) + e(k). \tag{21}$$

Here we assume that $s(k)$ is binary, i.e., it is either $+1$ or -1 with equal probability.

If a fuzzy filter (such as in [33]) is used as a TE, as shown in Fig. 3, its antecedents are $r(k), r(k-1), \dots, r(k-p+1)$, where

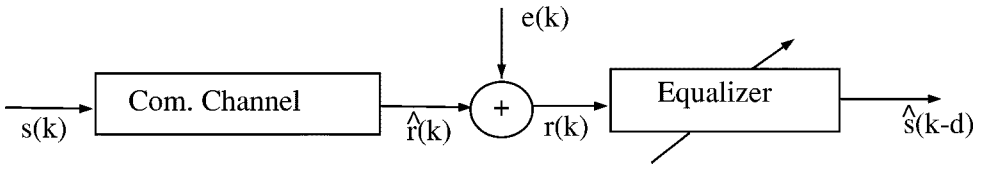


Fig. 2. Block diagram of a base-band communication system subject to ISI and AGN.

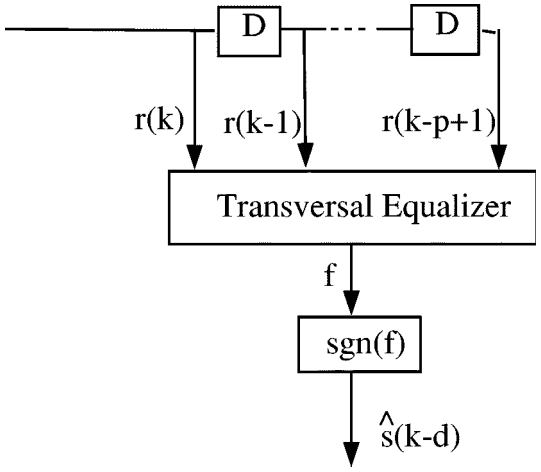


Fig. 3. The structure of a TE with p taps. In this paper, the parameters of the TE are determined by clustering the training sequence.

p is the equalizer order (i.e., number of taps in the equalizer). We denote

$$\mathbf{r}(k) \triangleq [r(k), r(k-1), \ldots, r(k-p+1)]^T. \tag{22}$$

Observe from (21), that $\mathbf{r}(k)$ depends on the channel input sequence $\mathbf{s}(k)$ (an $(n+p)\times 1$ vector), where

$$\mathbf{s}(k) = [s(k), s(k-1), \ldots, s(k-n-p+1)]^T. \tag{23}$$

Because $s(k)$ can be $+1$ or -1, there are $n_s = 2^{n+p}$ combinations of the channel input sequence. In Fig. 2, the noise-free signal is $\hat{r}(k)$, where

$$\hat{r}(k) = \sum_{i=0}^{n} a_i(k) s(k-i). \tag{24}$$

We let

$$\hat{\mathbf{r}}(k) \triangleq [\hat{r}(k), \ldots, \hat{r}(k-p+1)]^T \tag{25}$$

where $\hat{\mathbf{r}}(k)$ is called *channel state* [2]. Observe from (24) and (25) that each of the $n_s = 2^{n+p}$ combinations of the channel input sequence $\mathbf{s}(k)$ generates one $\hat{\mathbf{r}}(k)$, which we denote as $\hat{\mathbf{r}}_i(k)$, where $\hat{\mathbf{r}}_i(k) = [\hat{r}_i(k), \ldots, \hat{r}_i(k-p+1)]^T$. Hence, each channel state has a probability of occurence equal to $1/n_s$.

A correct decision by the equalizer occurs if

$$\hat{s}(k-d) = s(k-d) \tag{26}$$

where $\hat{s}(k-d)$ is the decision output of the equalizer and d is a decision delay. Based on the category of $s(k-d)$ (i.e., ± 1), the channel states, $\hat{\mathbf{r}}(k)$, can be partitioned into two classes [2]

$$R^+ = \{\hat{\mathbf{r}}(k) \,|\, s(k-d) = 1\} \tag{27}$$

$$R^- = \{\hat{\mathbf{r}}(k) \,|\, s(k-d) = -1\}. \tag{28}$$

The number of elements in R^+ and R^- are denoted n_s^+ and n_s^-, respectively. Because $s(k-d)$ has equal probability to be $+1$ or -1 so $n_s^+ = n_s^- = n_s/2 = 2^{n+p-1}$. The channel states in R^+ and R^- are denoted $\hat{\mathbf{r}}_i^+$ $(i = 1, \ldots, n_s^+)$ and $\hat{\mathbf{r}}_i^-$ $(i = 1, \ldots, n_s^-)$, respectively.

Chen *et al.* [2] have shown that the decision output of a Bayesian equalizer can be expressed as

$$\hat{s}(k-d) = \operatorname{sgn}(f(\mathbf{r}(k))) = \begin{cases} 1 & f(\mathbf{r}(k)) \geq 0 \\ -1 & f(\mathbf{r}(k)) < 0 \end{cases} \tag{29}$$

where $f(\mathbf{r}(k))$ is given by [3]

$$f(\mathbf{r}(k)) = \sum_{i=1}^{n_s^+} \left(2\pi\sigma_e^2\right)^{-m/2} \exp\left[\frac{-\|\mathbf{r}(k) - \hat{\mathbf{r}}_i^+\|^2}{2\sigma_e^2}\right] - \sum_{i=1}^{n_s^-} \left(2\pi\sigma_e^2\right)^{-m/2} \exp\left[\frac{-\|\mathbf{r}(k) - \hat{\mathbf{r}}_i^-\|^2}{2\sigma_e^2}\right] \tag{30}$$

in which σ_e denotes the standard deviation (std) of the Gaussian additive noise $e(k)$. Because only the sign of $f(\mathbf{r}(k))$ is used to make the decision in (29), the scaling term $(2\pi\sigma_e^2)^{-m/2}$ in (30) can be ignored [23].

Let

$$w_i \triangleq \begin{cases} 1 & \hat{\mathbf{r}}(k) \in R^+ \\ -1 & \hat{\mathbf{r}}(k) \in R^- \end{cases}; \tag{31}$$

then (30) can be expressed again as

$$f(\mathbf{r}(k)) = \sum_{i=1}^{n_s} w_i \exp\left[\frac{-\|\mathbf{r}(k) - \hat{\mathbf{r}}_i\|^2}{2\sigma_e^2}\right]. \tag{32}$$

Based on properties of the squared norm and the exponential function, (32) can be rewritten as (33), shown at the bottom of the page, where * denotes a conjugate. For the binary (2-PAM)

$$f(\mathbf{r}(k)) = \sum_{i=1}^{n_s} \prod_{l=0}^{p-1} w_i \exp\left\{-\frac{[r(k-l) - \hat{r}_i(k-l)]^* [r(k-l) - \hat{r}_i(k-l)]}{2\sigma_e^2}\right\} \tag{33}$$

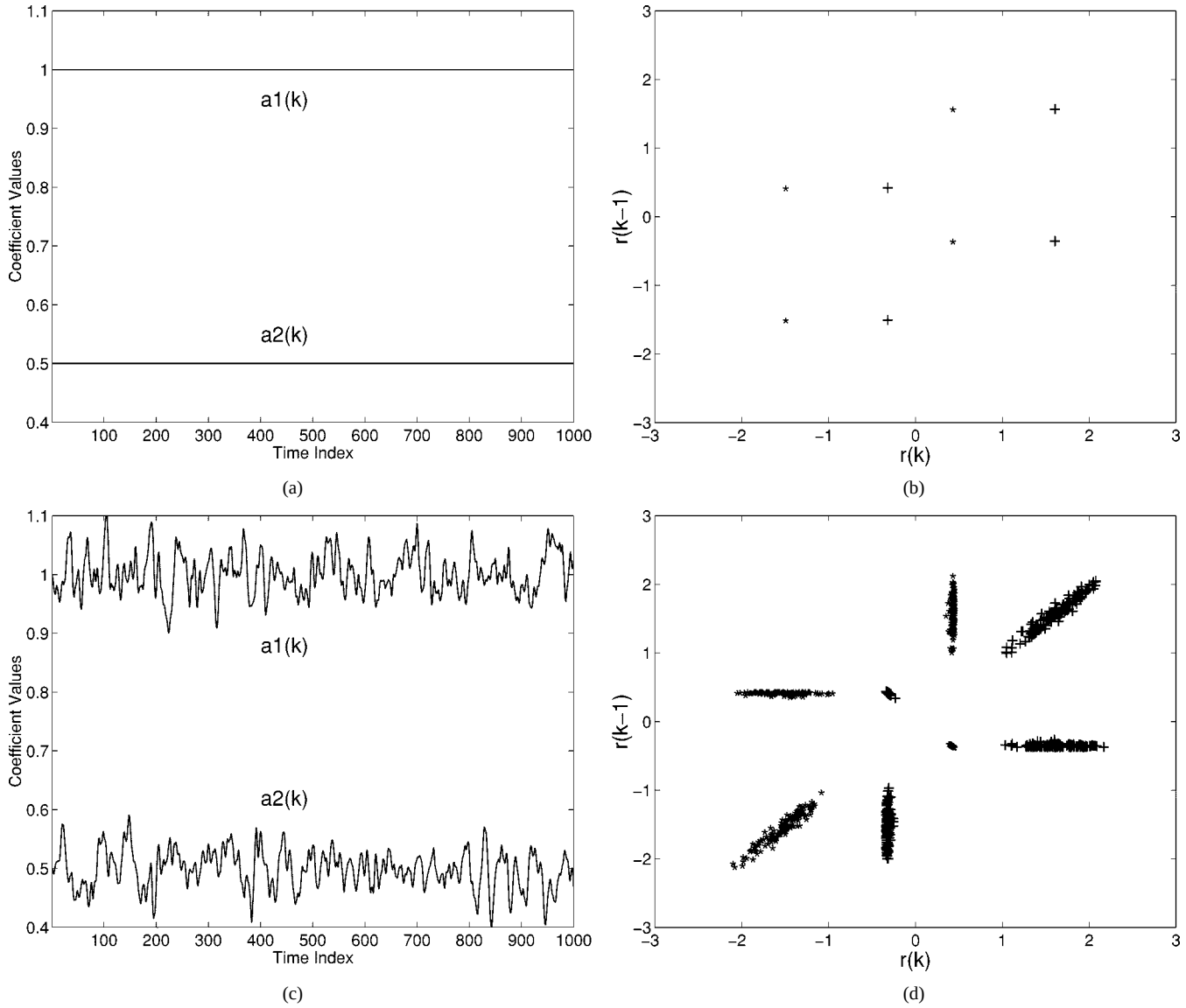


Fig. 4. For the channel in (35) (a) time-invariant channel with $a_1 = 1$ and $a_2 = 0.5$. (b) Channel states (noise free) of time invariant channel, where $*$ denotes the category $\hat{\mathbf{r}}(k)$ is $+1$ and $+1$ denotes the category $\hat{\mathbf{r}}(k)$ is -1. (c) Example of a time-varying channel with $\beta = 0.1$. (d) Channel states (noise free) of the time-varying channel in (c).

input sequence, $r(k-l)$ and $\hat{r}_i(k-l)$ are real so the conjugate operation can be ignored, i.e.,

$$f(\mathbf{r}(k)) = \sum_{i=1}^{n_s} \prod_{l=0}^{p-1} w_i \exp\left[-\frac{1}{2}\left(\frac{r(k-l) - \hat{r}_i(k-l)}{\sigma_e}\right)^2\right]. \tag{34}$$

Observe that (34) is identical to (4). Hence, an unnormalized output type-1 TSK FLS can be used to implement a Bayesian equalizer for a time-invariant channel.

Why not just implement (34) directly without connecting it to fuzzy logic? Equation (34) is based on a probability model, Gaussian distribution, whereas (4) is model free. As noted in [21], a shortcoming to model-based statistical signal processing is "... the assumed probability model for which model-based statistical signal processing results will be good if the data agrees with the model, but may not be so good if the data does not."

C. Why a Type-2 FAF Is Needed for Time-Varying Channel Equalization

Equation (34) has been derived for time-invariant channels. For a time-varying channel, the channel's coefficients, a_i $(i = 0, 1, \ldots, n)$, are uncertain. In [33], for example, the following nonlinear time-invariant channel model was used:

$$r(k) = a_1 s(k) + a_2 s(k-1) - 0.9[a_1 s(k) + a_2 s(k-1)]^3 + e(k) \tag{35}$$

where $a_1 = 1$ and $a_2 = 0.5$, as shown in Fig. 4(a). The channel states are plotted in Fig. 4(b) from which we observe that they are eight individual points.

In the rest of this section, we illustrate the design of a type-2 FAF for this channel, but we focus on the case when the channel is time-varying, i.e., when a_1 and a_2 in (35) are time-varying coefficients, each simulated, as in [5], by using a second-order Markov model in which a white Gaussian noise source drives a second-order Butterworth low-pass filter (LPF). In our

TABLE I
CHANNEL STATES FOR TIME-VARYING CHANNEL MODEL (35) WITH BINARY SYMBOLS: $d = 0$ AND $p = 2$

$s(k)$	$s(k-1)$	$s(k-2)$	$\hat{r}(k)$	$\hat{r}(k-1)$
1	1	1	$a_1(k) + a_2(k) - 0.9[a_1(k) + a_2(k)]^3$	$a_1(k) + a_2(k) - 0.9[a_1(k) + a_2(k)]^3$
1	1	-1	$a_1(k) + a_2(k) - 0.9[a_1(k) + a_2(k)]^3$	$a_1(k) - a_2(k) - 0.9[a_1(k) - a_2(k)]^3$
1	-1	1	$a_1(k) - a_2(k) - 0.9[a_1(k) - a_2(k)]^3$	$-a_1(k) + a_2(k) - 0.9[-a_1(k) + a_2(k)]^3$
1	-1	-1	$a_1(k) - a_2(k) - 0.9[a_1(k) - a_2(k)]^3$	$-a_1(k) - a_2(k) - 0.9[-a_1(k) - a_2(k)]^3$
-1	1	1	$-a_1(k) + a_2(k) - 0.9[-a_1(k) + a_2(k)]^3$	$a_1(k) + a_2(k) - 0.9[a_1(k) + a_2(k)]^3$
-1	1	-1	$-a_1(k) + a_2(k) - 0.9[-a_1(k) + a_2(k)]^3$	$a_1(k) - a_2(k) - 0.9[a_1(k) - a_2(k)]^3$
-1	-1	1	$-a_1(k) - a_2(k) - 0.9[-a_1(k) - a_2(k)]^3$	$-a_1(k) + a_2(k) - 0.9[-a_1(k) + a_2(k)]^3$
-1	-1	-1	$-a_1(k) - a_2(k) - 0.9[-a_1(k) - a_2(k)]^3$	$-a_1(k) - a_2(k) - 0.9[-a_1(k) - a_2(k)]^3$

simulations below, we used the function *butter*, provided by the Matlab Signal Processing Toolbox, to generate a second-order lowpass digital Butterworth filter with cutoff frequency 0.1; then the function *filter* was used to generate a colored Gaussian sequence, which was then used as a time-varying channel coefficient. Note that we centered $a_1(k)$ about 1 and $a_2(k)$ about 0.5. The input to the Butterworth filter was a white Gaussian sequence with standard deviation (std) β.

So that readers may replicate our simulations, we provide the source code for the time-varying coefficients with length 1000

```
[B,A] = butter(2,0.1);  % B (numerator) and
        A (denominator) of LPF
a1 = 1 + filter(B,A,beta*randn(1,1000))
a2 = 0.5 + filter(B,A,beta*randn(1,1000)).
```

Realizations of the time-varying coefficients and channel states are plotted in Figs. 4(c) and (d), respectively, for $\beta = 0.1$. Observe, that the channel states are now eight clusters instead of eight individual points. These clusters illustrate that $\hat{\mathbf{r}}_i$ is uncertain for all $i = 1, \ldots, 8$. From Table I, we see there are eight channel states and that $\mathbf{s}(k)$ determines which cluster $\hat{\mathbf{r}}(k)$ belongs to. Note that clusters $[\hat{r}(k), \hat{r}(k-1)]$ in the first four rows in Table I have category $+1$ [determined by $s(k-d) = s(k)$ based on (27) and (28)] and clusters in the last four rows have category -1. This [see (27), (28), and (31)] establishes the value of w_i in (34) as 1 or -1.

D. Designing the Type-2 FAF

In our type-2 FAF design, there are eight rules (each rule corresponds to one channel state), where the lth rule R^l is expressed as

$$R^l\text{: IF } r(k) \text{ is } \tilde{F}_1^l \text{ and } r(k-1) \text{ is } \tilde{F}_2^l \text{ THEN } y^l = w_l$$

where $\tilde{F}_1^l$ and $\tilde{F}_2^l$ are type-2 Gaussian MFs with uncertain means (as in Example 1), and w_l is a crisp value of $+1$ or -1 as determined by (31). For rule l, the range of the mean of antecedent $\tilde{F}_1^l$ ($\tilde{F}_2^l$) corresponds to the horizontal (vertical) projection of the lth cluster in Fig. 4(d). Observe from this rule that the consequent is a constant (i.e., it does not depend on $r(k)$ and $r(k-1)$). Hence, the consequent is a special case of the consequent in Section III.

We used (19) to compute the output of the type-2 FAF, where $y^l = w_l$ ($l = 1, \ldots, 8$) equals 1 or -1, $\underline{f}^l$ is obtained from (14), and $\overline{f}^l$ is obtained from (15). As in (8), we chose

$$\mu_{\tilde{F}_k^l}(x_k) = \exp\left[-\frac{1}{2}\left(\frac{x_k - m_k^l}{\sigma_e}\right)^2\right] \qquad m_k^l \in [m_{k1}^l, m_{k2}^l] \quad (36)$$

(see Fig. 1) and $k = 1, 2$. In order to specify the MFs $\tilde{F}_1^l$ and $\tilde{F}_2^l$, we need to specify their parameters, namely, $[m_{k1}^l, m_{k2}^l]$ and σ_e. Below, we let $\mathbf{m_1^l} = [m_{11}^l, m_{21}^l]^T$ and $\mathbf{m_2^l} = [m_{12}^l, m_{22}^l]^T$.

We used a clustering approach to estimate $\mathbf{m_1^l}$ and $\mathbf{m_2^l}$, because it is computationally simple [2]. Here we briefly summarize this approach. Suppose the number of training prototypes, $(\mathbf{s}(k), \mathbf{r}(k))$, is N. As we illustrated in Table I, $\mathbf{s}(k)$ determines which cluster $\mathbf{r}(k)$ belongs to; so the N $\mathbf{r}(k)$ are classified into $n_s = 2^{n+p}$ clusters, where, in our example, $2^{p+n} = 2^{2+1} = 8$. Suppose N_l training prototypes belong to the lth cluster $l = 1, \ldots, 8$ and the mean and std of these $\mathbf{r}(k)$, $k = 1, \ldots, N_l$ are denoted $\mathbf{m_r^l}$ (2×1 vector) and $\sigma_\mathbf{r}^l$ (2×1 vector), respectively. We let

$$\mathbf{m_1^l} \triangleq \mathbf{m_r^l} - \sigma_\mathbf{r}^l \quad (37)$$

$$\mathbf{m_2^l} \triangleq \mathbf{m_r^l} + \sigma_\mathbf{r}^l \quad (38)$$

where $l = 1, 2, \ldots, 8$. Doing this assumes that each cluster is centered at $\mathbf{m_r^l}$. Consequently, $[m_{11}^l, m_{12}^l]$ is the range of the mean of the type-2 antecedent Gaussian MF $\mu_{\tilde{F}_1^l}$ and $[m_{21}^l, m_{22}^l]$ is the range of the mean of $\mu_{\tilde{F}_2^l}$. For our type-1 FAF design, we used $\mathbf{m_r^l}$ (i.e., $[m_{r1}^l, m_{r2}^l]^T$) as the centers of its type-1 Gaussian antecedent MFs.

To complete the specification of the MFs in (36), we also need to estimate the std of the noise σ_e. In [4], it is shown that equalizer performance is not very sensitive to the value of σ_e. In our simulations, we assumed that the value of σ_e is known exactly. In all the simulations that follow, we fixed SNR and computed the std $\sigma_{\hat{r}}$ of $\hat{r}(k)$ in the combined training and testing sequence. Then, based on the fact that

$$\text{SNR} = 10 \log_{10} \frac{\sigma_{\hat{r}}^2}{\sigma_e^2} \quad (39)$$

we computed σ_e as

$$\sigma_e = \sigma_{\hat{r}} / 10^{\text{SNR}/20}. \quad (40)$$

E. Simulations

We compared our type-2 FAF with an unnormalized type-1 FAF (the latter is identical to an RBF network [2] in its output formula) and a nearest neighbor classifier (NNC) [27] for equalization of the time-varying channel in (35). The nearest-neighbor (NN) rule and its extension the K-NN algorithm [7] (if the number of training prototypes is N, then $K = \sqrt{N}$ is the optimal choice for K) are nonparameteric classification algorithms, that have been extensively applied to many pattern recognition problems. Recently, Savazzi *et al.* [27] applied a NNC, which used the K-NN algorithm to channel equalization for mobile radio communications and achieved good performance. NNC also belongs to sequence detection approach so we compare our type-2 FAF to NNC in channel equalization.

In our simulations, we chose the number of taps of the equalizer p equal to the number of taps of the channel, $n+1$, i.e., $p = n+1$. The number of rules equals the number of clusters, i.e., 2^{p+n}. The coefficients in (35) were chosen as described in Section IV-C.

To have $\sqrt{N}$ be an odd integer (as required by a NNC), we chose the number of training prototypes $N = 121$, which means $K = 11$. We used a sequence $s(k)$ of length 1000 for our experiments. The first 121 symbols were used for training, and the remaining 879 were used for testing. The training sequence established the parameters of the antecedent MFs, as described in Section IV-D. After training, the parameters of the type-1 and type-2 FAFs were fixed and then testing was performed.

In our first experiment, we fixed SNR at 20 dB and ran simulations for eight different β ranging from $\beta = 0.04$ to $\beta = 0.32$, with step size 0.04 (0.04:0.04:0.32) and we set $d = 0$. We performed 100 Monte Carlo (MC) simulations for each β value, where in each realization the channel coefficients and additive noise were uncertain. In Figs. 5(a) and (b), we plot the mean values and std of BER for the 100 MC realizations.

In a second experiment, we fixed $\beta = 0.1$ and ran simulations for five different SNR values ranging from SNR $= 15$ dB to SNR $= 25$ dB (15:2.5:25). We again performed 100 MC simulations for each SNR value. In Figs. 6(a) and (b), we plot the mean values and std of BER for the 100 MC realizations, respectively.

Observe the following from these figures.

1) In terms of the mean values of BER, the type-2 fuzzy filter performs much better than both the NNC and type-1 FAF (Figs. 5(a) and 6(a)).
2) When SNR $= 20$ dB, the NNC performs better than the type-1 FAF when $\beta \geq 0.12$ and the type-1 FAF peforms better than NNC when $\beta < 0.12$, but the the type-2 FAF always performs better than the NNC [Fig. 5(a)].
3) In terms of the std of BER, the type-2 FAF is more robust to the additive Gaussian noise than the other two equalizers and the type-1 FAF is more robust than the NNC [Figs. 5(b) and 6(b)].

These observations suggest that a type-2 FAF (as designed above) holds promise as a very good TE for time-varying channels. Unfortunately, though, the number of rules for such an equalizer is $n_s = 2^{n+p}$ (recall that $n+1$ is the number of channel taps and p is the number of antecedents), e.g., for $n = 4, p = 5$, we need 512 rules. This causes huge computational complexity when the channel order is high. Next, we use a DFE to tremendously reduce the number of rules in a FAF.

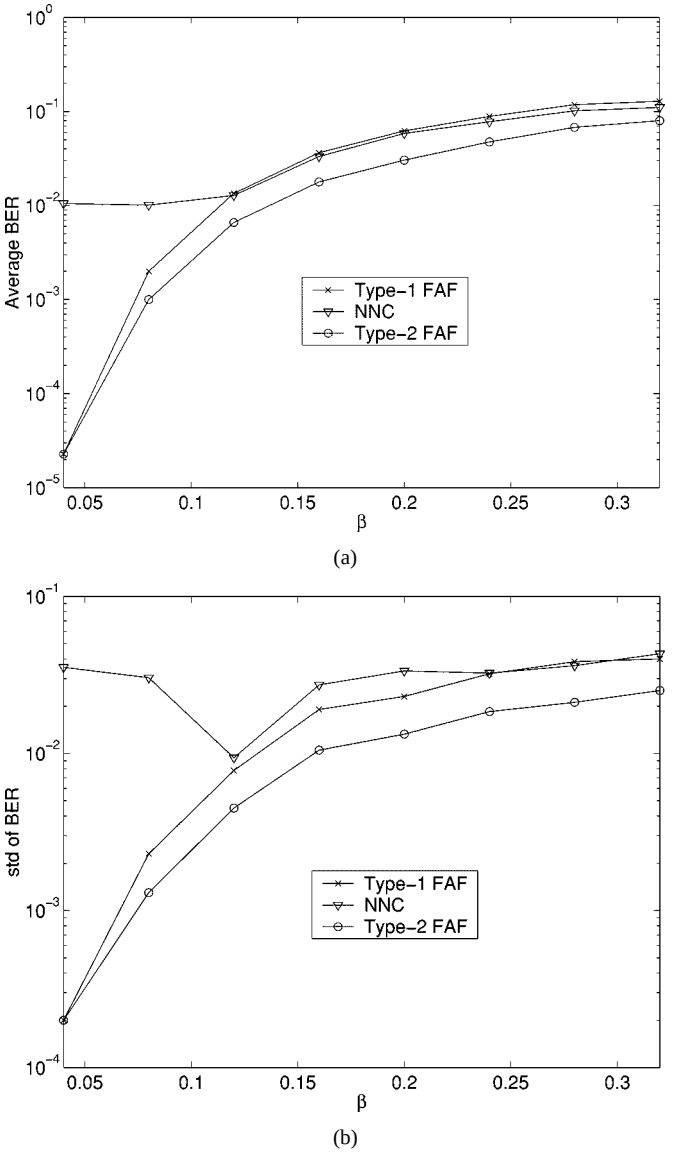
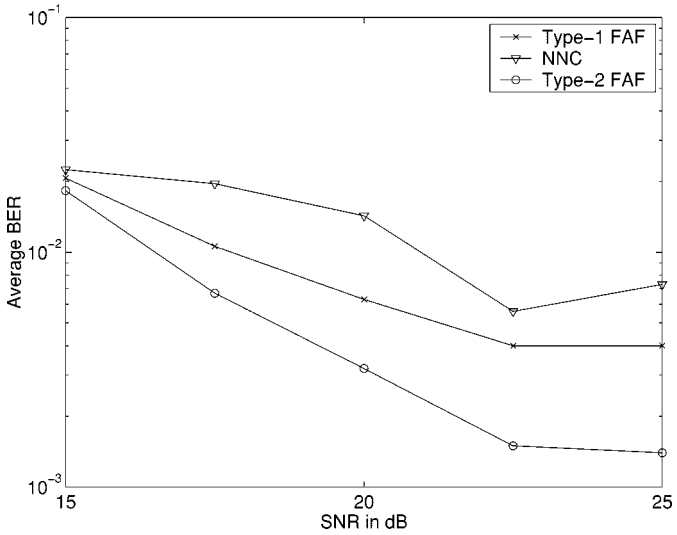


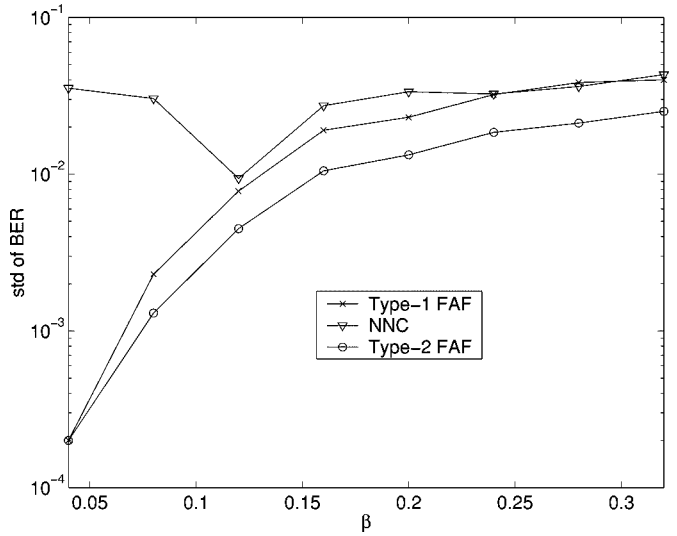
Fig. 5. Performance of type-1 FAF, nearest neighbor classifier (NNC), and type-2 FAF versus β when SNR $= 20$ dB and the number of training prototypes is 121. (a) Average BER. (b) STD of BER for 100 Monte Carlo realizations.

V. DFE for Time-Varying Channels Using a Decision Tree and Type-2 FAFs

It is well known that a DFE can reduce computational complexity and improve equalization performance [3] as compared to a TE. In addition, a DFE can be used to increase the channel bit-rate capacity of next-generation wireless mobile communication systems [1]. Fig. 7 shows the structure of a DFE having p feedforward taps and q feedback taps. In this section, we propose a new architecture to implement a FAF-based DFE, one that avoids the rule explosion generated with a TE.



(a)



(b)

Fig. 6. Performance of type-1 FAF, nearest neighbor classifier (NNC), and type-2 FAF versus SNR when $\beta = 0.1$ and the number of training prototypes is 121. (a) Average BER. (b) STD of BER for 100 Monte Carlo realizations.

A. An Architecture to Eliminate Rule Explosion in a FAF

We follow the channel state analysis used in [3] and, for illustrative purposes, use the following channel model to explain how a DFE can eliminate rule explosion:

$$r(k) = a_1(k)s(k) + a_2(k)s(k-1) + a_3(k)s(k-2) + e(k). \quad (41)$$

Assume a decision delay of unity $(d = 1)$ and two equalizer feedforward taps $(p = 2)$; then, there are $n_s = 2^{2+2} = 16$ channel states, which are enumerated in Table II in which the channel states $[\hat{r}(k), \hat{r}(k-1)]$ have category $+1$ or -1 [determined by $s(k-1)$ according to (27) and (28)]. If there are two feedback taps $(q = 2)$, then $\hat{s}(k-2)$ and $\hat{s}(k-3)$ are fed back to decide $\hat{s}(k-1)$ (see Fig. 7).

Observe, for example, in the first four rows of Table II, when $\hat{s}(k-2) = 1$ and $\hat{s}(k-3) = 1$ (shown using boldfaced numbers) we only need to use four channel states (four rules) to decide the value of $\hat{s}(k-1)$, which means 16 rules have been reduced to four. This motivates us to use a decision tree and four four-rule FAFs to implement a DFE for this channel (see Fig. 8).

The structure of a general DFE is specified by the decision delay d and number of channel taps $n+1$. d is chosen by the designer and increasing d improves performance, but it is required that $d \leq n$ [3]. It has been shown [3] that choosing the number of feedforward taps as $p = d+1$ (reducing d reduces the number of antecedents) and the number of feedback taps as $q = n$ is sufficient for a DFE to achieve all the performance potential (i.e., a DFE with $p = d+1$ has the same performance as DFEs with $p > d+1$ taps) for a given d and n. In Fig. 9, we depict a general structure for a DFE; it consists of a decision tree and 2^q FAFs, where each FAF has only 2^p rules. Observe that only one FAF is activated at a time to obtain the value of $\hat{s}(k-d)$. This structure reduces the number of FAF rules a lot, and makes it easy to design each of the FAFs, e.g., if a channel has five taps $(n = 4)$, delay $d = 4$, and we choose $p = d+1 = 5$ and $q = n = 4$, then we only need to design $2^q = 16$ FAFs, each having $2^p = 32$ rules (although there are 16 FAFs, only one is activated at any time). In contrast, as noted at the end of Section IV, if we use a TE with $p = 5$, we need $2^{n+p} = 512$ rules. The rule reduction ratio is $2^{n+p-p} : 1 = 2^n : 1$, i.e., 16 : 1. Although the number of rules in one FAF has been tremendously reduced, the total number of rules in all FAFs in a DFE is still the same as that of a TE, but it is much more difficult to compute one 512-rule FAF than to compute one 32-rule FAF.

Observe that the architecture of Fig. 9 is also applicable for other linear/nonlinear filter-based DFEs, e.g., a neural network (NN)-based DFE, where "FAF" can be substituted by "NN."

B. Designing a DFE Based on Type-2 FAFs

We used the following nonlinear time-varying channel in our simulations of a DFE:

$$\begin{aligned} r(k) = {} & a_1(k)s(k) + a_2(k)s(k-1) + a_3(k)s(k-2) \\ & - 0.7[a_1(k)s(k) + a_2(k)s(k-1) \\ & + a_3(k)s(k-2)]^3 + e(k) \end{aligned} \quad (42)$$

where nominal values for the channel coefficients are $a_1 = 0.3482, a_2 = 0.8704$, and $a_3 = 0.3482$. This channel's linear part $r(k) = 0.3482s(k) + 0.8704s(k-1) + 0.3482s(k-2) + e(k)$ has been studied in [3], [15], and [23]. A nonlinear channel model like (42) is frequently encountered in data transmission over digital satellite links, especially when the signal amplifiers operate in their high-gain limits [15]. We used the method given in Section IV-C to simulate the time-varying natures of $a_1(k), a_2(k)$ and $a_3(k)$.

We assumed a decision delay d of one. Since channel order $n = 2$ (we assumed that n is known), then it is sufficient to design a DFE with $p = 2$ and $q = 2$, which means the decision tree has $2^2 = 4$ leaves (FAFs), and each leaf (FAF) has $2^2 = 4$ rules. Since $p = 2$, the channel state analysis of (42) is very similar to that of (41) (Table II). To conserve space, we refer to Table II in analyzing the channel states of (42).

Designing the rules in each of the four FAFs is the same as that of designing a transversal fuzzy equalizer. As shown in

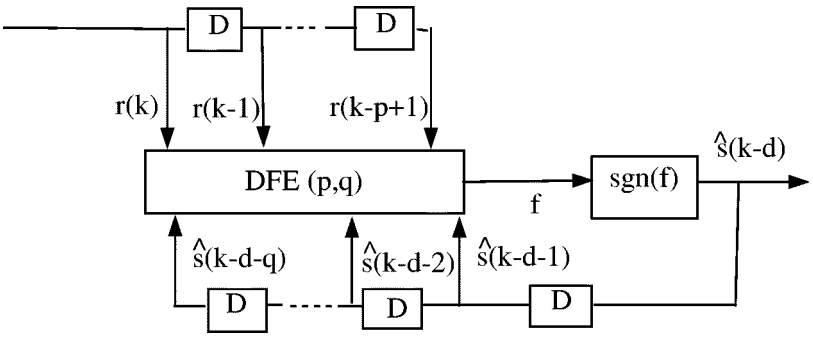


Fig. 7. The structure of a DFE with p feedforward taps and q feedback taps. In this paper, the parameters of the DFE are determined by clustering the training sequence.

TABLE II
CHANNEL STATES FOR TIME-VARYING CHANNEL MODEL (41) WITH BINARY SYMBOLS: $d = 1, p = 2$

$s(k)$	$s(k-1)$	$s(k-2)$	$s(k-3)$	$\hat{r}(k)$	$\hat{r}(k-1)$
1	1	**1**	**1**	$\mathbf{a_1(k)+a_2(k)+a_3(k)}$	$\mathbf{a_1(k)+a_2(k)+a_3(k)}$
-1	1	**1**	**1**	$\mathbf{-a_1(k)+a_2(k)+a_3(k)}$	$\mathbf{a_1(k)+a_2(k)+a_3(k)}$
1	-1	**1**	**1**	$\mathbf{a_1(k)-a_2(k)+a_3(k)}$	$\mathbf{-a_1(k)+a_2(k)+a_3(k)}$
-1	-1	**1**	**1**	$\mathbf{-a_1(k)-a_2(k)+a_3(k)}$	$\mathbf{-a_1(k)+a_2(k)+a_3(k)}$
1	1	1	-1	$a_1(k)+a_2(k)+a_3(k)$	$a_1(k)+a_2(k)-a_3(k)$
-1	1	1	-1	$-a_1(k)+a_2(k)+a_3(k)$	$a_1(k)+a_2(k)-a_3(k)$
1	-1	1	-1	$a_1(k)-a_2(k)+a_3(k)$	$-a_1(k)+a_2(k)-a_3(k)$
-1	-1	1	-1	$-a_1(k)-a_2(k)+a_3(k)$	$-a_1(k)+a_2(k)-a_3(k)$
1	1	-1	1	$a_1(k)+a_2(k)-a_3(k)$	$a_1(k)-a_2(k)+a_3(k)$
-1	1	-1	1	$-a_1(k)+a_2(k)-a_3(k)$	$a_1(k)-a_2(k)+a_3(k)$
1	-1	-1	1	$a_1(k)-a_2(k)-a_3(k)$	$-a_1(k)-a_2(k)+a_3(k)$
-1	-1	-1	1	$-a_1(k)-a_2(k)-a_3(k)$	$-a_1(k)-a_2(k)+a_3(k)$
1	1	-1	-1	$a_1(k)+a_2(k)-a_3(k)$	$a_1(k)-a_2(k)-a_3(k)$
-1	1	-1	-1	$-a_1(k)+a_2(k)-a_3(k)$	$a_1(k)-a_2(k)-a_3(k)$
1	-1	-1	-1	$a_1(k)-a_2(k)-a_3(k)$	$-a_1(k)-a_2(k)-a_3(k)$
-1	-1	-1	-1	$-a_1(k)-a_2(k)-a_3(k)$	$-a_1(k)-a_2(k)-a_3(k)$

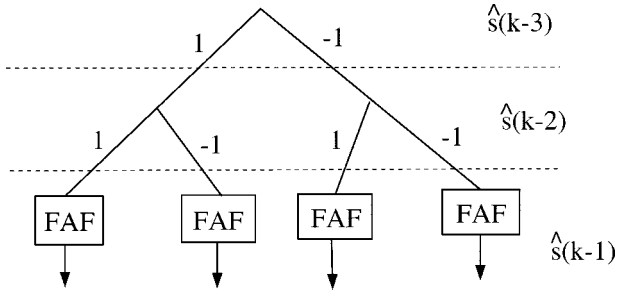


Fig. 8. The architecture of a DFE for channel (41), where decision delay $d = 1$, and DFE parameters $p = 2$ and $q = 2$. This DFE consists of a decision tree and four FAFs, and each FAF has four rules.

Table II and Fig. 8, in our type-2 FAF DFE design, there are a total of 16 rules and the lth rule, R^l is expressed as

$$R^l\text{: IF } r(k) \text{ is } \tilde{F}_1^l \text{ and } r(k-1) \text{ is } \tilde{F}_2^l \text{ THEN } y^l = w_l$$

where we assume that $\tilde{F}_1^l$ and $\tilde{F}_2^l$ are type-2 Gaussian MFs with uncertain means (as in Example 1), and w_l is a crisp value of $+1$ or -1 as determined by (31).

In the training period, $s(k-3)$ and $s(k-2)$ determine four rules belonging to one of the four FAFs (see Fig. 8 and Table II divided by horizontal double lines); so 16 rules have been divided into four groups, and $s(k)$ and $s(k-1)$ determine one channel state in each FAF (see Table II). We use (37) and (38) to set the antecedent MF parameters, and the consequent parameter $w_l = s(k-1)$.

Generally speaking, in the training period, suppose the number of training prototypes, $(\mathbf{s}(k), \mathbf{r}(k))$ is N, where $\mathbf{s}(k)$ and $\mathbf{r}(k)$ are defined by (23) and (22). Since $p = d+1$ and $q = n$, (23) can be rewritten as

$$\mathbf{s}(k) = [s(k), s(k-1), \ldots, s(k-d-q)]^T. \qquad (43)$$

A general description for designing a FAF DFE based on N training prototypes for a channel with $n+1$ taps and a decision delay d (which determine $p = d+1$ and $q = n$) is as follows.

1) Based on the values of $[s(k-d-q), \ldots, s(k-d-1)]$, a branch and its corresponding leaf (FAF) in the decision tree (see Fig. 9) is chosen.

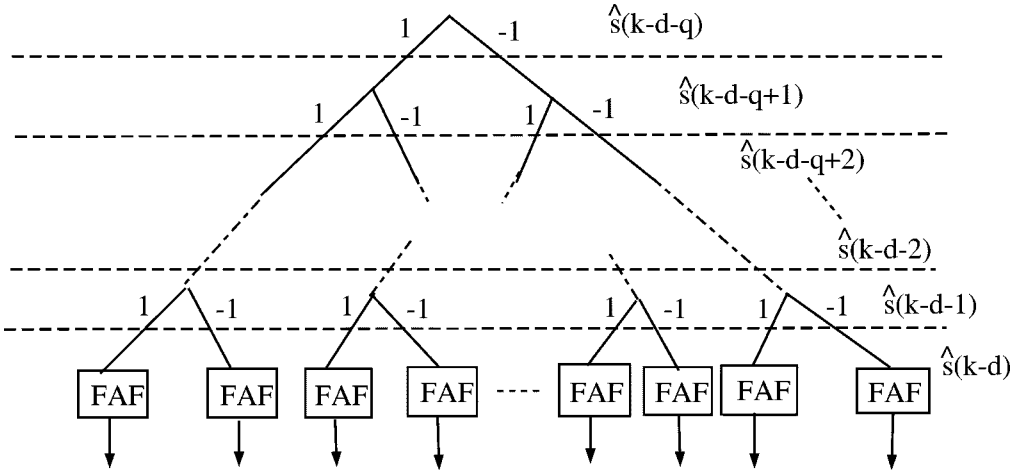


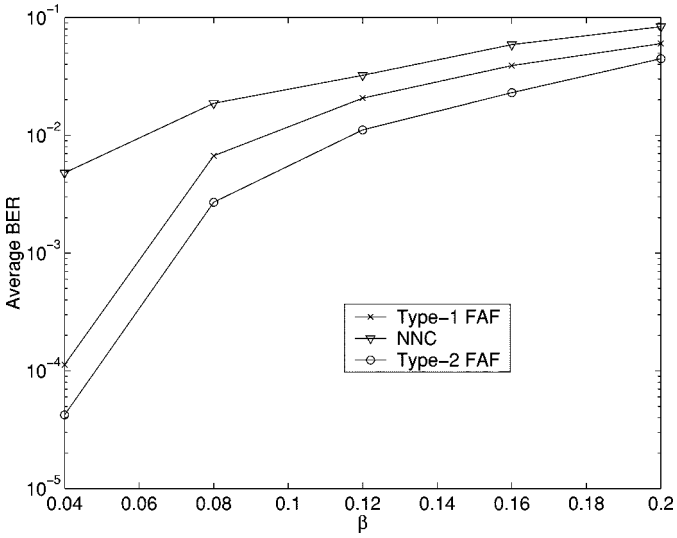
Fig. 9. For a channel with $n+1$ taps and a decision delay of d, the general architecture of a DFE with p $(p=d+1)$ feedforward taps and q $(q=n)$ feedback taps. The DFE consists of a decision tree and 2^q FAFs, where each FAF has 2^p rules.

2) In the chosen FAF, design 2^p rules, which means 2^p clusters are needed. Based on $[s(k-d),\ldots,s(k)]$, we know which cluster $\mathbf{r}(k)$ belongs to (see Table II) and $s(k-d)$ determines the cluster category $+1$ or -1.
3) Repeat steps 1) and 2) until all N training prototypes have been clustered.
4) Suppose in the ith FAF, $(i=1,\ldots,2^q)$, there are N_l training prototypes belonging to the lth cluster, $(l=1,\ldots,2^p)$ and the mean and std of these $\mathbf{r}(k)$ [$p\times 1$ vector, see (22)] $(k=1,\ldots,N_l)$, are denoted $\mathbf{m}_{\mathbf{r}}^l$ ($p\times 1$ vector) and $\sigma_{\mathbf{r}}^l$ ($p\times 1$ vector), respectively. Then we can use (37) and (38) to obtain the parameters $\mathbf{m}_{\mathbf{1}}^l$ and $\mathbf{m}_{\mathbf{2}}^l$, where $\mathbf{m}_{\mathbf{1}}^l=[m_{11}^l,m_{21}^l,\ldots,m_{p1}^l]^T$ and $\mathbf{m}_{\mathbf{2}}^l=[m_{12}^l,m_{22}^l,\ldots,m_{p2}^l]^T$; so $[m_{j1}^l,m_{j2}^l]$ $(j=1,2,\ldots,p)$ is the range of the type-2 antecedent Gaussian MF, $\mu_{\tilde{F}_j^l}$, in the ith FAF.
5) After the training period, the parameters of every FAF are fixed. In the testing period, for every $\mathbf{r}(k)$, use Theorem 3 to compute the defuzzified output of the activated FAF $f(\mathbf{x})$ and then use (29) to obtain the output of the DFE $\hat{s}(k-d)$.

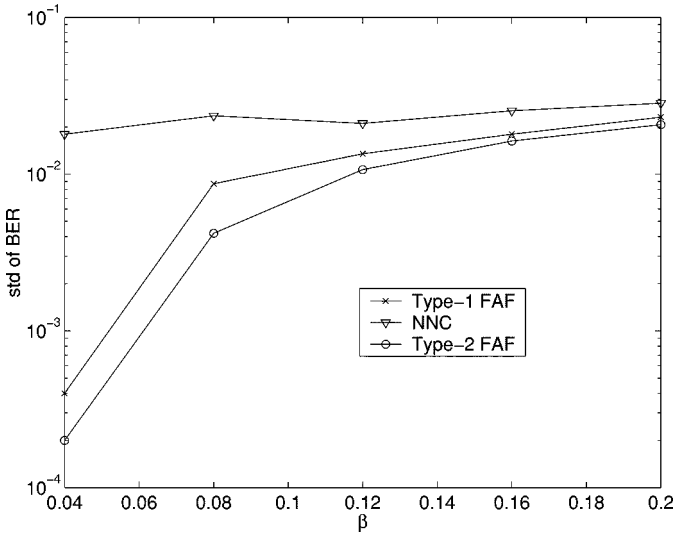
C. Simulations

Simulations were performed for channel (42) in which we used a 1000 symbol sequence $s(k)$. The first 289 symbols were for training and the remaining 711 symbols were for testing. After training, the parameters in all four fuzzy filters were fixed and then testing was performed.

In our first experiment, we fixed SNR at 20 dB and ran simulations for five different β ranging from $\beta=0.04$ to $\beta=0.2$ (0.04:0.04:0.20). We performed 100 MC simulations for each β value. In Fig. 10(a) and (b), we plot the mean values and std of BER for the 100 MC realizations. In a second experiment, we fixed $\beta=0.1$ and ran simulations for seven different SNRs ranging from SNR $=15$ dB to SNR $=30$ dB (15:2.5:30). We again performed 100 MC simulations for each SNR value. In Fig. 11(a) and (b), we plot the mean values and std of BER for these 100 MC realizations.

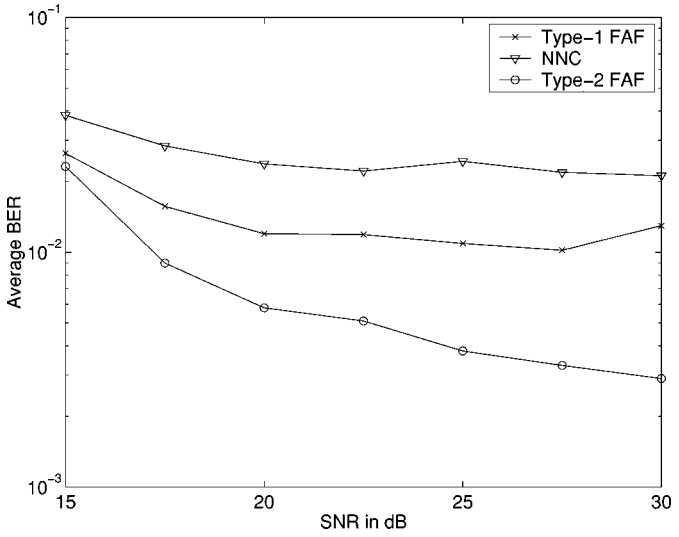


(a)

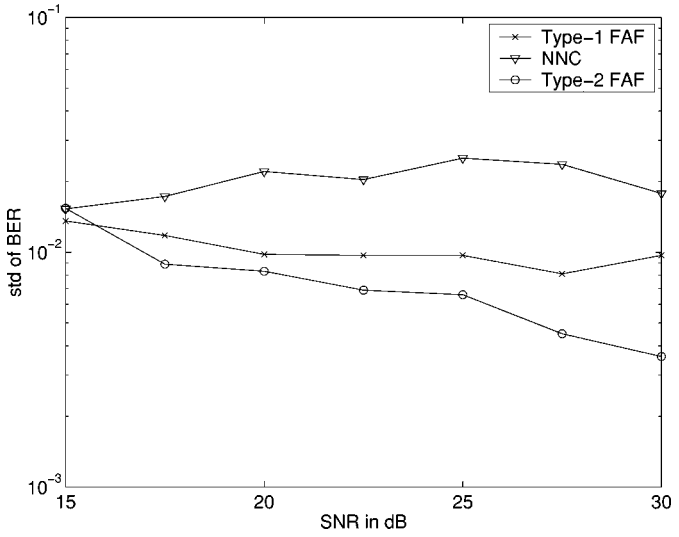


(b)

Fig. 10. Performance of type-1 FAF-based DFE, nearest neighbor classifier (NNC), and type-2 FAF-based DFE versus β when SNR $=20$ dB and the number of training prototypes is 289. (a) Average BER. (b) STD of BER in 100 Monte Carlo realizations.



(a)



(b)

Fig. 11. Performance of type-1 FAF-based DFE, nearest neighbor classifier (NNC), and type-2 FAF-based DFE versus SNR when $\beta = 0.1$ and the number of training prototypes is 289. (a) Average BER. (b) STD of BER in 100 Monte Carlo realizations.

From the mean and std values of BER, we see that the DFE based on four type-2 FAFs performs much better than the NNC and the DFE based on four type-1 FAFs (each is an unnormalized type-1 TSK FLS). The NNC cannot work well in such a complicated channel because there are 16 channel states and a NNC typically needs more training prototypes than we have used.

VI. Conclusions and Future Works

We have proposed a new unnormalized output type-2 TSK FAF, one that handles numerical and linguistic uncertainties. We applied this type-2 FAF to equalization where the channel is nonlinear and time-varying. Theoretical analysis shows that this type-2 FAF can exactly implement a Bayesian equalizer for time-varying channels. Its structure is simple, its inference is fast, and it is model free.

We used our type-2 FAF to implement a TE and also used a decision tree and more than one type-2 FAF to implement a DFE. The number of rules in each FAF of the DFE is tremendously reduced. In fact, for a channel with $n + 1$ taps, the rule reduction ratio is $2^n : 1$. This architecture is also applicable for other linear/nonlinear filter-based DFEs.

Simulation results showed that both the type-2 FAF TE and DFE performed better than either a type-1 FAF or a nearest neighbor classifier. Since no tuning procedure was used in the design of either type-2 FAF-based equalizer, real-time information processing is guaranteed.

Although a FAF has been extensively used for channel equalization, a training sequence is needed for all approaches, including the ones described in this paper. A challenge is to develop a type-2 FAF for adaptive equalization which does not need any training sequence, i.e., for blind equalization, first proposed by Sato [26]. The constant modulus algorithm (CMA) proposed by Godard [9] is emerging as the recognized standard blind equlization algorithm and is already appearing in real systems [32]. We are exploring the design of FAF-based blind equalizers for both time-invariant and time-varying channels.

References

[1] S. Ariyavisitakul, N. Sollenberger, and L. Greenstein, "Tap-selectable decision-feedback equalizer," *IEEE Trans. Commun.*, vol. 45, pp. 1497–1500, Dec. 1997.

[2] S. Chen, B. Mulgrew, and S. McLaughlin, "A clustering technique for digital communications channel equalization using radial basis function network," *IEEE Trans. Neural Networks*, vol. 4, pp. 570–579, July 1993.

[3] ——, "Adaptive Bayesian equalizer with decision feedback," *IEEE Trans. Signal Processing*, vol. 41, pp. 2918–2927, Sept. 1993.

[4] S. Chen, S. McLaughlin, B. Mulgrew, and P. M. Grant, "Adaptive Bayesian decision feedback equalizer for dispersive mobile radio channels," *IEEE Trans. Commun.*, vol. 43, pp. 1937–1956, May 1995.

[5] C. F. N. Cowan and S. Semnani, "Time-variant equalization using a novel nonlinear adaptive structure," *Int. J. Adaptive Contr. Signal Processing*, vol. 12, no. 2, pp. 195–206, 1998.

[6] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*. New York: Academic, 1980.

[7] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[8] G. B. Giannakis and C. Tepedelenlioglu, "Basis expansion models and diversity techniques for blind identification and equalization of time-varying channel," *Proc. IEEE*, vol. 86, pp. 1969–1986, Oct. 1998.

[9] D. Godard, "Self-recovering equalization and carrier tracking in two-dimensional data communication systems," *IEEE Trans. Commun.*, vol. 28, pp. 1867–1875, Nov. 1980.

[10] N. N. Karnik and J. M. Mendel, "Introduction to type-2 fuzzy logic systems," in *Proc. IEEE FUZZ Conf.*, Anchorage, AK, May 1998.

[11] ——, "Type-2 fuzzy logic systems: Type-reduction," in *Proc. IEEE SMC Conf.*, San Diego, CA, Oct. 1998.

[12] (1998) Introduction to type-2 fuzzy logic systems. Univ. Southern California. [Online]. Available: http://sipi.usc.edu/~mendel/report

[13] ——, "Operations on type-2 fuzzy sets," Fuzzy Sets Syst., to be published.

[14] N. N. Karnik, J. M. Mendel, and Q. Liang, "Type-2 fuzzy logic systems," *IEEE Trans. Fuzzy Syst.*, vol. 7, Oct. 1999.

[15] G. Kechriotis, E. Zervas, and E. Manolakis, "Using recurrent neural networks for adaptive communication channel equalization," *IEEE Trans. Neural Networks*, vol. 5, pp. 2670–2678, Mar. 1994.

[16] K. Y. Lee, "Complex fuzzy adaptive filters with LMS algorithm," *IEEE Trans. Signal Processing*, vol. 44, pp. 424–429, Feb. 1996.

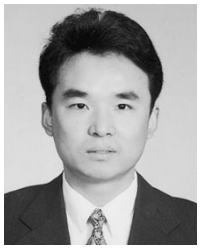
[17] Q. Liang and J. M. Mendel, "Interval type-2 fuzzy logic systems: Theory and design," IEEE Trans. Fuzzy Syst., to be published.

[18] ——, "Interval type-2 TSK fuzzy logic systems," IEEE Trans. Syst., Man, Cybern., to be published.

[19] J. M. Mendel, "Fuzzy logic systems for engineering: A tutorial," *Proc. IEEE*, vol. 83, pp. 345–377, Mar. 1995.

[20] ——, "Computing with words when words can mean different things to different people," in *Int. ICSC Congress Computat. Intell.: Methods Applicat. 3rd Annu. Symp. Fuzzy Logic Applicat.*, Rochester, NY, June 1999.

[21] ——, "Uncertainty, fuzzy logic, and signal processing," *Signal Processing*, vol. 80, no. 6, pp. 913–933, June 2000.
[22] J. Moon and T. Jeon, "Sequence detection for binary ISI channels using signal space partitioning," *IEEE Trans. Commun.*, vol. 46, pp. 891–901, July 1998.
[23] S. K. Patra and B. Mulgrew, "Efficient architecture for Bayesian equalization using fuzzy filters," *IEEE Trans. Circuits Syst. II*, vol. 45, pp. 812–820, July 1998.
[24] ——, "Fuzzy implementation of Bayesian equalizer in the presence of intersymbol and co-channel interference," *Proc. Inst. Elect. Eng. Commun.*, vol. 145, pp. 323–330, Oct. 1998.
[25] P. Sarwal and M. D. Srinath, "A fuzzy logic system for channel equalization," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 246–249, May 1995.
[26] Y. Sato, "A method of self-recovering equalization for multilevel amplitude-modulation systems," *IEEE Trans. Commun.*, vol. 23, pp. 679–682, June 1975.
[27] P. Savazzi, L. Favalli, E. Costamagna, and A. Mecocci, "A suboptimal approach to channel equalization based on the nearest neighbor rule," *IEEE J. Selected Areas Commun.*, vol. 16, pp. 1640–1648, Dec. 1998.
[28] M. Sugeno and G. T. Kang, "Structure identification of fuzzy model," *Fuzzy Sets Syst.*, vol. 28, pp. 15–33, 1988.
[29] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. 15, pp. 116–132, Jan./Feb. 1985.
[30] K. Tanaka, M. Sano, and H. Watanabe, "Modeling and control of carbon monoxide concentration using a neuro-fuzzy technique," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 271–279, Aug. 1995.
[31] K. Tanaka and M. Sugeno, "Introduction to fuzzy modeling," in *Fuzzy Systems Modeling and Control*, H. T. Nguyen and M. Sugeno, Eds. Boston, MA: Kluwer, 1998, pp. 63–89.
[32] J. R. Treichler, I. Fijalkow, and C. R. Johnson Jr., "Fractionally spaced equalisers: How long should they be?," *IEEE Signal Processing Mag.*, vol. 13, pp. 65–81, May 1996.
[33] L.-X. Wang and J. M. Mendel, "Fuzzy adaptive filters, with application to nonlinear channel equalization," *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 161–170, Aug. 1993.
[34] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning—I," *Inform. Sci.*, vol. 8, pp. 199–249, 1975.



Qilian Liang received the B.S. degree from Wuhan University, China, in 1993, the M.S. degree from Beijing University of Posts and Telecommunications, China, in 1996, and the Ph.D. degree from University of Southern California, Los Angeles, in 2000, all in electrical engineering.

He was a Research Assistant in the Department of Electrical Engineering-Systems, University of Southern California, Los Angeles, from September 1997 to May 2000. His research interests include wireless communications, broad-band networks, fuzzy logic systems, and video traffic classification. He is currently with Hughes Network Systems, San Diego, CA.

Jerry M. Mendel (S'59–M'61–SM'72–F'78) received the Ph.D. degree in electrical engineering from the Polytechnic Institute of Brooklyn, Brooklyn, NY.

Currently, he is Professor of Electrical Engineering and Associate Director for Education of the Integrated Media Systems Center, University of Southern California, Los Angeles, where he has been since 1974. He has published over 380 technical papers and is the author and/or editor of seven books. His present research interests include type-2 fuzzy logic systems and their applications to a wide range of problems.

Dr. Mendel is a Distinguished Member of the IEEE Control Systems Society. He was President of the IEEE Control Systems Society in 1986. Among his awards are the 1983 Best Transactions Paper Award of the IEEE Geoscience and Remote Sensing Society, the 1992 Signal Processing Society Paper Award, a 1984 IEEE Centennial Medal, and an IEEE Third Millenium Medal.