Predictive Modular Fuzzy Systems for Time Series Classification

Vas. Petridis and Ath. Kehagias
Div. of Electronics and Comp. Eng., Dept. of Electrical Eng.
Aristotle University of Thessaloniki
Thessaloniki 540 06, Greece
e-mail: kehagias@egnatia.ee.auth.gr

This paper appeared in IEEE Trans. on Fuzzy Systems

Abstract

We introduce the so called *PREdictive MOdular Fuzzy System* (PREMOFS) which performs time series classification. A PREMOFS consists of (a) a bank of prediction modules and (b) a fuzzy decision module. It is assumed that the time series is generated by a source belonging to a finite search set (universal set); then the classification problem is to select the source that best represents the observed data. Classification is based on a membership function which is updated recursively, according to the predictive accuracy of each model. Two algorithms are presented for updating the membership function: the first is based on sum/product fuzzy inference; the second on max/min fuzzy inference. In short, PREMOFS is a fuzzy *modular* system which classifies *time series* to one of a *finite* number of classes, using the full set of past data (*without preprocessing*)to perform a *recursive, competitive* computation of membership function, based on *predictive* accuracy. Convergence proofs are given for both PREMOFS algorithms; in both cases the membership grade tends to one for the source that best predicts the observed data and to less than one for the remaining sources; hence correct classification is guaranteed. Simulation results are also presented: PREMOFS are applied to signal detection, system identification and phoneme classification tasks.

1 Introduction

This paper treats the following *time series classification* problem. A source $S(\theta_k)$, generates time series X_1, X_2, X_3, ... ; here parameter θ_k takes values in the *finite* set $\Theta = \{\theta_1, ..., \theta_K\}$. The source that generates the time series must be determined. A typical example involves a speech time series X_1, X_2, ... and phoneme labels θ_k belonging to the phoneme set Θ= { [ah], [oo],... }; the phoneme [ah] corresponds to X_1, X_2,... generated by S([ah]). Problems of this type arise in phoneme recognition [6, 30], EEG/ECG diagnosis [9, 42, 43], polygraph test analysis [24], radar classification [10] and various general tasks [26, 29, 41]. System identification [27, 37] may also be viewed as a problem of this type.

The term "*dynamic pattern* classification" will be used as a synonym of "time series classification"; similarly for the terms "class" and "source". The term "dynamic pattern" is used in contradistinction to "*static pattern* classification", where a (finite dimensional) feature vector must be assigned to one of a set of possible classes. Problems of static pattern classification appear frequently in the pattern recognition literature and they can be solved by fuzzy methods, e.g. see [2, 8, 17, 28, 35, 40]; other treatments of the problem use statistical [7] or neural [12, 16, 38] methods. Of course, the above cited references are only a small sample of the vast pattern classification literature.

Several characteristics distinguish dynamic pattern classification from the static one. If the feature vector is taken to be the observations of the time series X_1, X_2, ... then the dimensionality of the problem increases without bound. Also, it is usually required that the classification is updated on-line as new data become available. The usual approach to these issues is what we call *staticizing* the problem. Namely, the time series is viewed in portions (usually through a sliding window); each portion is preprocessed and a static feature vector is extracted; finally, the preprocessed feature vector is used for static pattern classification. This approach, used in most of the time series applications cited above, may result in loss of important correlations across several time steps.

In this paper time series classification is carried out by the so called *PREdictive MOdular Fuzzy System* (PREMOFS). PREMOFS operates in two phases. In the off-line phase, a predictor is trained for each candidate source / class (using data from that particular source). In the on-line phase, at time steps t = 1, 2, ... , each predictor produces an estimate of the next observation X_t, using raw data X_{t-1}, X_{t-2}, ... When the new observation X_t becomes available, the respective prediction error is computed for each predictor; this error is used to update the respective source membership grade in a recursive manner. At time t the time series is classified to the source / class with maximum membership grade. In short, *PREMOFS* is a *modular* system which classifies *time series* to one of a *finite* number of classes, using the full set of past data *without preprocessing* to perform a *recursive* computation of membership function which is based on *predictive accuracy*.

The inspiration for PREMOFS comes from the *Partition Algorithm*, a method originally applied to system identification and control [13, 22, 23, 36]. We have used the Partition Algorithm as the basis for developing the PREMONNS (PREdictive Modular Neural Networks) family of neural classification algorithms [18, 31, 32]; these are close relatives of PREMOFS. Both the Partition Algorithm and the original PREMONN [31] are based on a Bayesian probabilistic point of view. More general versions of PREMONN's [18, 32] do not depend on a probabilistic interpretation, but still use a probabilistic convergence analysis. On the other hand, PREMOFS makes no use of probabilistic assumptions whatosever and can be based either on sum/product or on max/min inference, as will be explained in Section 3. In addition to these "direct" relatives of PREMOFS, related ideas appearing in the fuzzy and neural literature are discussed below.

In a sense, all fuzzy systems are implicitly modular, since they implement a combination of various rules, one module corresponding to each rule. However this does not necessarily result in a *modular training algorithm* such as the one employed by PREMOFS. By this we mean a method whereby substitution of one part of the classification system does not require retraining the whole system. Methods for modular training of fuzzy classifiers [11, 26, 29, 33, 39] are usually applied to static or staticized problems, they involve approximate (rather than exact) optimization and they do not use a recursive classification update. Modular architectures also appear in the neural literature; for instance in [14, 15] under the name of *local experts*. Once again, training is modular but approximate, and is performed off-line.

Several fuzzy clustering algorithms use new data to recursively update some classification criterion [2, 11, 26, 29, 39, 40]. In fact, some of these algorithms *discover a previously unknown* partition of the feature space; this is beyond the scope of PREMOFS. However, in such algorithms recursion is usually employed to improve the estimates of class boundaries; once this is achieved (i.e. when the recursion converges) every new exemplar presented is classified independently of the previous ones. PREMOFS, on the other hand, must learn the classification rules off-line; but every time step of on-line classification depends on previous classification decisions. In short, on-line recursion provides a link between past and currect clasifications. This underscores the dynamic nature of PREMOFS and dynamic pattern classification in general; in statistical terms, useful information is extracted from the correlation between successive observations of the time series. Finally, to the best of our knowledge, PREMOFS is unique among fuzzy time series classifiers in using raw data (without preprocessing) and in classifying according to prediction error.

The paper is organized as follows: in Section 2 classification by prediction is explained in detail ; in Section 3 two alternative methods of fuzzy inference are presented and used to develop the sum/product PREMOFS and the max/min PREMOFS; in Section 4 convergence of the PREMOFS algorithms is proved; in Section 5 some general issues are discussed; in Section 6 PREMOFS is applied to logistic detection, AC motor resistance estimation and phoneme classification; In Section 7 conclusions are summarized.

2 Classification by Prediction

We are given a *finite* crisp set $\Theta = \{\theta_1, ..., \theta_K\}$, a source $S(\theta_k)$ (where θ_k is a parameter taking values in Θ) and a time series X_t, t=1, 2, ... , generated by $S(\theta_k)$, where X_t takes values in R^n, for some n. Θ will be henceforth called the *source set*, *search set* or, in fuzzy set theory terms, the *universal set*. It is required to identify the source that generates the time series; i.e. to find the "true" value of θ_k.

Consider a variable Z, taking values in the crisp set Θ, and say that the time series is generated by source $S(Z)$; e.g. when $Z = \theta_1$, the time series X_1, X_2, ... is generated by $S(\theta_1)$. Also take an estimate of Z; it is called $\hat{Z}_t$ and takes values in Θ. The time index t is used because $\hat{Z}_t$ is updated by a fuzzy decision rule at every time step: if *at time* t we believe X_1, ... X_t to have been generated by, say, $S(\theta_1)$, then $\hat{Z}_t = \theta_1$. As more observations become available, $\hat{Z}_t$ may change with time. The selection of $\hat{Z}_t$ at time t is based on the following process of fuzzy inference. Consider the attribute: *"source $S(Z)$ has been active from time s up to time t"*. A crisp set of elements that satisfy this attribute, must include exactly one member of Θ; this is so because it has been assumed that the time series is generated by a single source. However, we propose to use a *fuzzy* set:

$$A(s,t) = \{(Z, \mu_{A(s,t)}(Z)) | Z \in \Theta\}. \tag{1}$$

The fuzzy set $A(s,t)$ consists of the crisp set Θ (the set of possible values of the source parameter) and the membership function $\mu_{A(s,t)}(Z)$; for a given Z, $\mu_{A(s,t)}(Z)$ is the membership grade of the attribute *"source $S(Z)$ has been active from time s to time t"*. Obviously $A(s,t)$ has a time dependence on times s and t. Now, consider θ_k (the k-th member of Θ): $\mu_{A(1,t)}(\theta_k)$ is the membership grade of *"source $S(\theta_k)$ has been active from time 1 to time t"*, or equivalently, *"observations X_1, X_2, ... , X_t have been generated by source $S(\theta_k)$"*. This particular quantity is used very frequently in this paper; for economy of space we use the alternative notation

$$\mu_t^k \doteq \mu_{A(1,t)}(\theta_k); \tag{2}$$

but the reader should keep in mind that μ_t^k depends on θ_k. It is emphasized that μ_t^k may change with time, depending on new observations; what is required here is to provide and justify a method for updating μ_t^k at every time step. This will be derived presently; but first note that, for a given time t, it is natural to set

$$\hat{Z}_t = \arg\max_{\theta_k \in \Theta} \mu_{A(1,t)}(\theta_k), \tag{3}$$

or, using a simpler notation,

$$\hat{Z}_t = \arg\max_{\theta_k \in \Theta} \mu_t^k \tag{4}$$

but the reader should keep in mind that $\hat{Z}_t$ takes values in the set Θ. In other words the time series is classified to the source $S(\hat{Z}_t)$ which achieves maximum membership grade.

To implement recursive computation of μ_t^k the PREMOFS algorithm is developed. Its first phase is an off-line process: for each source /class in the search set (i.e. for every value θ_k, k= 1, 2, ... K) labelled data are collected and a predictor trained; the predictor has the general form

$$\hat{X}_t^k = f(X_{t-1}, ..., X_{t-M}; \theta_k). \tag{5}$$

Here M is the *order* of the predictor (number of past time series values used). This is a general form, encompassing fuzzy, linear, sigmoid (neural) and many other predictor types. θ_k appears in eq.(5) because $\hat{X}_t^k$ approximates X_t *when the time series is generated by $S(\theta_k)$*. The second phase is on-line: for t=1, 2, ... and $k = 1, 2, ..., K$ define the one-step prediction error

$$e_t^k \doteq X_t - \hat{X}_t^k \tag{6}$$

or, more generally, the N-step prediction error

$$E_t^k \doteq [X_t - \hat{X}_t^k, \quad X_{t-1} - \hat{X}_{t-1}^k, \quad ..., \quad X_{t-N} - \hat{X}_{t-N}^k]. \tag{7}$$

Note that in eq.(7) E_t^k depends on N, the number of backward steps used, but this dependence is suppressed for brevity. For $N > 0$, E_t^k is a vector of size $(N+1) \cdot n$; for $N = 0$, $E_t^k = e_t^k$, which can be a vector (if $n > 1$) or a scalar. The idea on which PREMOFS is based is this: when the time series is generated by source $S(\theta_m)$, $|E_t^m|$ should be smaller than $|E_t^k|$, $k \neq m$, "on the average" (where $|\cdot|$ denotes Euclidean norm). Based on this idea and keeping track of predictor errors over long time intervals, a reasonable membership function will be computed in a recursive manner. For instance, the following update could be used

$$\mu_t^k = \mu_{A(1,t)}(\theta_k) = \mu_{A(1,t-1)}(\theta_k) \textbf{ AND } \mu_{A(t-1,t)}(\theta_k). \tag{8}$$

Eq.(8) means: the membership grade of the attribute "the complete X_1, ... , X_t observation has been generated by source / class $S(\theta_k)$" is the same as the membership grade of the conjuncted attributes "the complete X_1, ... , X_{t-1} observation has been generated by source / class $S(\theta_k)$" **AND** "the X_t observation has been generated by source / class $S(\theta_k)$". Thus, μ_t^k is computed in terms of $\mu_{A(1,t-1)}(\theta_k)$ and $\mu_{A(t-1,t)}(\theta_k)$. The latter can be computed as a function of the form

$$\mu_{A(t-1,t)}(\theta_k) = e^{-\left(\frac{|E_t^k|}{\sigma}\right)^2}. \tag{9}$$

Any function can be used in place of the exponential, as long it is a decreasing function of the Euclidean norm of the error. In eq.(9) the membership grade is expressed in terms of predictive accuracy: when $| E_t^k |$ is large, $\mu_{A(t-1,t)}(\theta_k) = e^{-\left(\frac{|E_t^k|}{\sigma}\right)^2}$ is small. Now eqs.(8), (9) result in the following recursive equation:

$$\mu_t^k = \mu_{t-1}^k \textbf{ AND } e^{-\left(\frac{|E_t^k|}{\sigma}\right)^2}. \tag{10}$$

The implementation of the **AND** conjunction in (10) has not yet been specified; several options are available and will be discussed presently. At any rate, (10) shows that when $| E_t^k |$ is large, then $e^{-\left(\frac{|E_t^k|}{\sigma}\right)^2}$ (and consequently $\mu_{A(t-1,t)}(\theta_k)$) is small; this implies that $\mu_t^k = \mu_{A(1,t-1)}(\theta_k)$ **AND** $e^{-\left(\frac{|X_t - \hat{X}_t^k|}{\sigma}\right)^2}$ is also small. In fact, a little reflection shows that eq.(10) results in a decreasing sequence of membership grades μ_t^k. This may result in various implementation problems (e.g. numerical underflow), so a normalized form will be used in what follows.

$$\mu_t^k = \frac{\mu_{t-1}^k \textbf{ AND } e^{-\left(\frac{|E_t^k|}{\sigma}\right)^2}}{\textbf{OR}_{l=1}^{K}\left(\mu_{t-1}^l \textbf{ AND } e^{-\left(\frac{|E_t^l|}{\sigma}\right)^2}\right)}. \tag{11}$$

The previous comments about the influence of $| E_t^k |$ on μ_t^k apply to eq.(11) as well, but now *relative*, not *absolute*, magnitude of $| E_t^k |$ influences μ_t^k, since the computation of membership grades is *competitive*. Hence, a large $| E_t^k |$ does *not* necessarily imply small membership grade μ_t^k; the value of μ_t^k may be large if $| E_t^l | > | E_t^k |$ for $l \neq k$, that is, if other predictors perform even worse.

Eqs.(5), (7), (9), (11) and (4) constitute the PREMOFS classification algorithm. This algorithm can be implemented in a modular manner, with K *prediction modules* implementing eq.(5) and a *decision module* implementing eq.(11). Each of these modules computes independently of each other; in addition, any module can be substituted without having to retrain the remaining modules. Eq.(11) updates μ_t^k using *the entire history* X_1, X_2, ... , X_t. The prediction error E_t^k has an explicit time dependence of $N+1$ steps back: t, $t-1$, ... , $t-N$ (see eq.(7)). However, μ_{t-1}^k depends on E_{t-1}^k (which depends on time steps $t-1$, $t-2$, ... , $t-N-1$), on μ_{t-2}^l, $l = 1, 2, ..., K$ and so on. Hence, μ_t^k implicitly depends on the entire X_1, ... , X_t series. This is useful to capture correlations between different parts of a time series; when such correlations exist they furnish information which is important for the classification task (this is an important difference between static and dynamic pattern classification). On the other hand, the update method should provide a *forgetting* mechanism, so that the distant past does not influence current classification decisions excessively. Such a mechanism is provided in the PREMOFS algorithm, by the use of a threshold; this issue is discussed in 5.1.

It has been established that PREMOFS is a method for *modular*, *recursive* update of class membership grades, depending on class predictive accuracy. The computation of membership grades is *competitive*: the classes compete for a share of the membership grade, and receive as much of it as is warranted by their respective predictive accuracy. Note that the form of the decision module has not yet been specified; this will depend on the implementation of the fuzzy **AND** and **OR** inference, to be discussed in the next section.

3 Modes of Fuzzy Inference

The form of the PREMOFS decision module depends on the implementation of the fuzzy **AND** and **OR** in eq.(11). In fuzzy set theory there are two standard ways to implement such logical operators [3]: **AND** is implemented by a product and **OR** is implemented by a sum; alternatively **AND** is implemented by a minimum and **OR** is implemented by a maximum. Two "hybrid" combinations are also possible: **AND** is implemented by a product and **OR** is implemented by a maximum; **AND** is implemented by a minimum and **OR** is implemented by a sum. In this paper only the first two cases are dealt with, yielding respectively the sum/product PREMOFS and the max/min PREMOFS; but it should be obvious how to obtain the corresponding max/product PREMOFS and sum/min PREMOFS.

The sum/product PREMOFS algorithm is obtained by combining eqs.(4), (5), (6) and (11) (with N=1, i.e. $E_t^k = e_t^k$), replacing all **AND** 's by products and all **OR** 's by sums.

Sum/Product PREMOFS Algorithm

For $k = 1, ..., K$ and for t=1,2,...

$$0 < \mu_0^k < 1 \tag{12}$$

$$\hat{X}_t^k = f(X_{t-1}, ..., X_{t-M}; \theta_k). \tag{13}$$

$$e_t^k \doteq X_t - \hat{X}_t^k. \tag{14}$$

$$\mu_t^k = \frac{\mu_{t-1}^k \cdot e^{-\left(\frac{|e_t^k|}{\sigma}\right)^2}}{\sum_{l=1}^K \mu_{t-1}^l \cdot e^{-\left(\frac{|e_t^l|}{\sigma}\right)^2}} \tag{15}$$

$$\hat{Z}_t = \arg\max_{\theta_k \in \Theta} \mu_t^k \tag{16}$$

The max/min PREMOFS algorithm is obtained by combining eqs.(4), (5), (7) and (11) , replacing all **AND** 's by min's and all **OR** 's by max's.

Max/Min PREMOFS Algorithm

For $k = 1, ..., K$ and for t=1,2,...

$$0 < \mu_0^k \leq 1 \tag{17}$$

$$\hat{X}_t^k = f(X_{t-1}, ..., X_{t-M}; \theta_k). \tag{18}$$

$$E_t^k \doteq [X_t - \hat{X}_t^k, \quad X_{t-1} - \hat{X}_{t-1}^k, \quad ..., \quad X_{t-N} - \hat{X}_{t-N}^k] \tag{19}$$

$$\mu_t^k = \frac{\mu_{t-1}^k \wedge e^{-\left(\frac{|E_t^k|}{\sqrt{N}\sigma}\right)^2}}{\bigvee_{l=1}^K \left(\mu_{t-1}^l \wedge e^{-\left(\frac{|E_t^l|}{\sqrt{N}\sigma}\right)^2}\right)} \tag{20}$$

$$\hat{Z}_t = \arg\max_{\theta_k \in \Theta} \mu_t^k \tag{21}$$

It has already been mentioned that the N-time-steps error E_t^k is more general than the one-time-step error e_t^k. It is easy to design a sum/product PREMOFS algorithm that uses E_t^k rather than e_t^k; we chose to present the particular forms of sum/product and max/min PREMOFS algorithms because these are the ones that were experimentally found to work best.

The normalized form of equations (15) and (20) ensures that in both algorithms we have $0 \leq \mu_t^k \leq 1$ for all t and K. In the case of the sum/product PREMOFS $\sum_{k=1}^K \mu_t^k = 1$ for every t and in the case of the max/min PREMOFS $\bigvee_{k=1}^K \mu_t^k = 1$ for every t. Hence the normalization ensures that at least one μ_t^k never becomes too small. In fact, under appropriate conditions, one μ_t^k will tend to one, as will be seen in the next section.

4 Convergence of Membership Grades

It has already been stated informally that the time series should be classified to the source / class that best predicts the observations. This requirement is now stated in an exact manner. It is immediately recognized that a "predictive accuracy" criterion cannot be based on the instantaneous predictive error e_t^k or E_t^k for any fixed value of t, because even the most accurate predictor (corresponding to the best class) may perform poorly at any particular time step. What matters is *average* performance.

In a probabilistic context one would describe average behavior using *expected values*; but here we want to avoid probabilistic considerations. A purely phenomenological approach will be followed, based on the *Cesaro average* ([4], p.572). For a sequence x_1, x_2, ... we say that $x_t \to x$ in the *Cesaro sense* iff

$$\lim_{t \to \infty} \frac{x_1 + x_2 + ... + x_t}{t} = x. \tag{22}$$

In short, $\{x_t\}_{t=1}^{\infty}$ tends to x *on the average.* It is easy to prove that convergence in the usual sense implies convergence in the Cesaro sense, but not conversely ([4], p.572).

Now convergence results are given for the two PREMOFS algorithms: if the m-th predictor is best (i.e. the limit of Cesaro average prediction error is smallest for the m-th predictor), then in the case of sum/product PREMOFS $\lim_{t\to\infty}\mu_t^m = 1$ and $\lim_{t\to\infty}\mu_t^k = 0$, $k \neq m$, while in the case of max/min PREMOFS $\lim_{t\to\infty}\mu_t^m = 1 > \limsup_{t\to\infty}\mu_t^k$, $k \neq m$. [1] In both cases, the time series is classified to the "best" source.

4.1 Convergence of the sum/product PREMOFS Algorithm

First, for $k = 1, 2, ..., K$, the following quantity is defined

$$D^k \doteq \lim_{t\to\infty} \frac{| X_1 - \hat{X}_1^k |^2 + | X_2 - \hat{X}_2^k |^2 + ... + | X_t - \hat{X}_t^k |^2}{t}. \tag{23}$$

It is emphasized that the k superscript is not an exponent, but denotes the error of the k-th predictor. Existence of the limit is discussed in the next paragraph. In other words D^k is the limit of the Cesaro average of the squared one-step prediction errors $| e_t^k |^2 = | X_t - \hat{X}_t^k |^2$. D^k is a meaningful index of the predictive accuracy of the k-th predictor: the smallest D^k corresponds to the predictor that is best in the sense of having smallest average square error. The following theorem is proved regarding convergence of the sum/product PREMOFS algorithm, as summarized in eqs.(12) - (16) (the proof is presented in the Appendix). [2]

Theorem 1 *If the following conditions hold:*

A1 $0 < \mu_0^k < 1$ *for* $k = 1, 2, ..., K$,

A2 *there is some* m *such that* $D^m < D^k$, *for all* $k \neq m$,

then $\lim_{t\to\infty}\mu_t^m = 1$ *and for* $k \neq m$ $\lim_{t\to\infty}\mu_t^k = 0$.

Condition **A1** says that initial values μ_0^k must be strictly between zero and one; this can always be satisfied, since we choose μ_0^k. Condition **A2** says that the quantities D^k must exist for all k and an m must exist such that D^m is minimum (namely *there is* a "best" class). (In a probabilistic context, this condition would hold for an ergodic time series; but our formulation avoids any reference to probabilistic concepts.) If the above conditions are satisfied, convergence to the "best" class is guaranteed and, in the limit, the largest membership grade is attained by the m-th class which minimizes prediction error (in the sense of limit Cesaro average); this also ensures that $\lim_{t\to\infty}\hat{Z}_t = \theta_m$.

[1] A given sequence a_t may be divergent; still it always has at least one subsequence converging to some limit. Roughly speaking, $\limsup a_t$ is the largest limit attainable by some subsequence of a_t. In other words, in the long run a_t will never exceed $\limsup a_t$ by much.

[2] Here (and in what follows) *average* value means the time average, such as taken in eq.(23); this should be distinguished from the probabilistic *ensemble average*, usually called *mean* or *expected value* and denoted by $E(\cdot)$. For an *ergodic* time series, time and ensemble averages are equal; however, wanting to avoid probabilistic assumptions, we only use time averages.

4.2 Convergence of the max/min PREMOFS Algorithm

First define D_t^k, as

$$D_t^k \doteq \frac{| X_{t-N} - \hat{X}_{t-N}^k |^2 + ... + | X_{t-1} - \hat{X}_{t-1}^k |^2 + | X_t - \hat{X}_t^k |^2}{\sigma^2 \cdot (N+1)}. \tag{24}$$

(In fact D_t^k also depends on N; for notational simplicity this dependence is not denoted explicitly.) D_t^k is an *approximation* to the previously defined D^k: in most practical situationsD_t^k becomes independent of N, for large enough N. The following theorem is proved, regarding convergence of the max/min PREMOFS algorithm, as summarized in eqs.(17) - (21) (the proof is presented in the Appendix).

Theorem 2 *If the following conditions hold:*

B1 $0 < \mu_0^k \leq 1$ *for* $k = 1, 2, ..., K$,

B2 *there are numbers* m, A, B, C *such that for a given* N, **for all** t *and for all* $k \neq m$

$$0 < A < D_t^m < B < C < D_t^k, \tag{25}$$

then $\lim_{t\to\infty} \mu_t^m = 1$ *and for* $k \neq m$ $\limsup_{t\to\infty} \mu_t^k < 1$.

The quantities D_t^k are not actual limits, but they approximate the limits D^k for large N. Condition **B2** requires that the error of the m-th predictor is smaller than that of all other predictors; **B2** is similar to **A2** in requiring the existence of a "best" class. Compared to the sum/product PREMOFS, the convergence condition is more complicated to state (and the theorem harder to prove) for a technical reason: the min operator cannot be concatenated in the same manner as the product operator and it is necessary to operate with "finite approximations" of the limit of the Cesaro average (i.e. the D_t^k's). This becomes apparent in the proof of the theorem. It should be mentioned that in practice a value of N around 50 is sufficient to ensure convergence.

5 General Considerations

The mathematical analysis of the previous section guarantees convergence to the best class, as long as the assumptions of Theorems 1 and 2 are satisfied. This analysis has been complemented by informal analysis, verified by running a large number of classification experiments. Some experiments are reported in Section 6. In this section some general conclusions are presented.

5.1 Decision Module

Given a bank of prediction modules we must reach a classification decision. This will be influenced by several factors.

A. Form of Decision Module. First, there is a choice regarding the implementation of the **AND** and **OR** operations; as already mentioned, we have experimented with sum/product and max/min modules. Second, in eqs.(15), (20), the membership function is a Gaussian or Radial Basis Function (RBF) , i.e. an exponential function of the negative squared prediction error. This type of membership function is prevalent in fuzzy sets literature [19, 34]. However, any other standard form of membership function (triangular, square etc.) may be used, as long as it is a decreasing function of the norm of the prediction error.

B. Robustness of Decision. Classification performance depends not on *absolute*, but on *relative* predictive accuracy, because of the competitive computation of membership grade (see eqs.(15) and(20)). The time series will be classified to a source even if it performs poorly, but consistently better than its competitors. This results in considerable robustness to prediction error. Good predictors generally result in superior classification performance, but, within certain bounds, high prediction error does not affect classification too much. This is verified by the experiments of Section 6. On the other hand, since *some* classification must always take place, if no good class is available, the "least bad" one will be chosen, which may at times lead to wrong conclusions. But, as discussed in 6.4, examination of membership grade profiles offers a useful diagnostic in such cases.

C. Scale Factor σ. The σ parameter appearing in eqs.(15), (20) influences the speed of convergence to correct classification. In the proof of Theorem 1, it is shown that convergence rate of the μ_t^k's depends on $(D^k - D^l)/\sigma^2$, for $k, l = 1, \dots, K$. If predictor k has large D^k and predictor l has small D^l, then $(D^k - D^l)/\sigma^2$ is large, and convergence is fast (see eq.(44)). However, all other things being equal, when σ is large, $(D^k - D^l)/\sigma^2$ becomes small and convergence is slow. A similar, but more complicated argument applies to max/min PREMOFS as well. From a different point of view, a large error means that less information is obtained per observation, so more observations are necessary to reach a certain level of confidence. This can be controlled by the choice of σ, which can be estimated during the predictor training phase, by the usual statistical estimator: $\sigma \simeq s$, $s = \sqrt{\frac{\sum_{t=1}^{T}|e_t|^2}{T}}$. It is sometimes advantageous to modify this estimate: for faster classification, σ must be decreased, as experiments indicate.

D. Threshold h. So far it has been assumed that the whole time series is generated by a single source; in other words, a *stationary* time series has been assumed. In many cases this assumption is not true. For instance, in a speech time series each phoneme is generated by a different source and different sources are active over different time intervals, introducing nonstationarity of the time series. This situation is termed *source switching* and presents the following difficulty. Suppose that a source, say $S(\theta_1)$, has been active for a long time; then by the convergence theorems presented earlier, μ_t^1 will be close or equal to 1 and μ_t^k, $k \neq 1$ may be close to 0 (especially in the case of sum/product PREMOFS). In fact, due to numerical underflow the μ_t^k's may become *equal* to 0. From eq.(15) or eq.(20) it is obvious that in such a case μ_t^k will remain 0 even in case source $S(\theta_k)$ is "switched on" at a later time and a false classification will result. To solve this problem, when μ_t^k falls below a prespecified threshold h, it is reset to h. Such thresholding amounts to introducing a forgetting factor: even if a sequence of observations is very unlikely to have been generated by a particular source, that source is never completely removed from consideration; if it predicts well a later sequence of observations, μ_t^k can recover a high value. An alternative way of looking at the use of h is this: whenever one or more μ_t^k's fall below h (because the corresponding predictors perform poorly) the algorithm is restarted using new μ_0^k's, obtained from resetting the corresponding μ_t^k's to h. Under this interpretation, the prior estimate of the membership grade for each source never goes below h.

Both σ and h are related to a trade-off between speed and accuracy. A large σ value slows the PREMOFS algorithm down and assigns little importance to individual errors; small σ speeds the PREMOFS up, but also assigns more importance to instantaneous fluctuations and makes the PREMOFS more prone to instantaneous classification errors. Conversely, a large h minimizes the importance of past performance and speeds up the response of the PREMOFS to source switchings, at the cost of spurious false classifications; small h results in large recovery time between source switchings.

E. Initial values μ_0^k. It can be seen from the convergence Theorems 1 and 2 that strict positivity is the only restriction placed on the μ_0^k's to ensure convergence of the membership functions. This theoretical conclusion, is also supported by our experiments: classification performance is more or less independent of the initial values μ_0^k.

5.2 Prediction Modules

Let us now consider the prediction modules.

A. Form of Prediction Module. Several different types of predictors can be used for the PREMOFS prediction modules: linear, sigmoid (neural), polynomial or RBF (radial basis functions) predictors are some possibilities. These are called *black-box* models or predictors and do *not* take into account the mechanism that generates a particular time series. On the other hand, if such a mechanism is known (except for some parameter values) a so called *structured* model or predictor may be used, e.g. a logistic-form predictor for a logistic time series. The distinction between black-box and structured models is well established in the system identification literature. Black-box predictors are often used in fuzzy set methods of identification; consider for instance [27] and especially [37]. It must be mentioned that PREMOFS can employ different types of predictors within the same predictor bank, e.g. a linear, a sigmoid and a polynomial predictor.

B. Noise Robustness of Prediction. Predictors of the form (5) use past observations as input. This makes prediction accuracy depend on the level of noise present in the observations. In certain cases, a predictor performs very well with noise-free input, but performance deteriorates rapidly with increasing noise level. This phenomenon has been observed in the logistic time series experiments presented in 6.1, 6.2, 6.3. Three types of predictors are used: sigmoid, linear and logistic-structured. The logistic predictor gives zero prediction error when noise-free input is used, but the error becomes quite large when the input is noisy. This is due to the nonlinear, chaotic nature of the logistic mapping: small input variations may result in large output variations [1], (pp.226, 244). The sigmoid predictor is even more sensitive to noise. On the other hand, the linear predictor error is relatively insensitive to noise, never becoming too large or too small. [3] This, combined with the decision robustness discussed in 5.1, results in superior, noise-robust classification, as will be seen in 6.1, 6.2, 6.3. The noise robustness

[3]The linear predictor uses past observations of the logistic time series in a linear combination; by the linearity or superposition principle, this results in separate components for the signal and noise parts of the time series. Regarding the noise part, the total noise term will have the form

$$w_1 n_{t-1} + w_2 n_{t-2} + ... + w_M n_{t-M}. \qquad (26)$$

observed in the logistic experiments is an illustration of a general principle: a combination of decision and prediction robustness makes PREMOFS classification noise-robust.

C. Predictor Order M**.** This is related to the number of predictor parameters, to be determined in the training phase. The main consideration is to avoid overfitting. We have used reasonably low order predictors, trying to enhance robustness rather than accuracy of prediction.

D. Prediction Error Order N**.** Sum/product PREMOFS uses one-step error e_t^k; while max/min PREMOFS uses N-step error E_t^k. Hence, for the max/min PREMOFS, N is another parameter to be chosen. A theoretical consideration in the choice of N, is that it should be large enough to justify the approximation of D^k by D_t^k (see Theorem 2). On the other hand, our experience indicates that a large value of N results in a sluggish system; a small value in an unstable one. Usually N is taken around 50, to obtain satisfactory results.

E. Modularity. The predictors employed by PREMOFS are independent of each other, in the sense that if one of them is removed from the system the remaining predictors need not be reconfigured or retrained. For instance, if it is decided to replace the sigmoid predictor of a particular source, by a linear predictor for the same source, predictors of the other sources remain unchanged. Similarly, the decision module can be changed without affecting the prediction modules, e.g. change from sum/product to max/min decision. This property of modularity is useful in reducing development time.

F. Parallelism. Modularity also introduces parallelism. Predictors can operate in parallel and send their predictions concurrently to the decision module. Hence execution time is independent of the number of classes in the classification problem and new classes / predictors can be added as the need arises. In contrast, it is a well known fact that *lumped* classifiers scale badly with the number of classes. E.g. in neural classifiers, as the number of parameters increases training may take exponentially long time and/or fail to achieve a good solution [12]. Similar considerations apply for fuzzy classifiers. Hence, modular classifiers, where fixed size modules are trained piecewise, are highly desirable. PREMOFS falls in this category. Clearly, both classifier size and training time scale linearly with the number of sources / classes that must be learned.

6 Experiments

Several sets of experiments are presented in this section, dealing with various time series classification tasks. The following considerations apply to all the experiment sets presented.

Classification performance is measured by a figure of merit $c = N_1/N_0$, where N_1 is the number of time steps at which the source of the time series is correctly identified, and N_0 is the total number of time steps.

Since $E(n_t)= E(n_{t-1})= \dots =E(n_{t-M})$, the expectation of the above expression is equal to

$$E(n_t) \cdot (w_1 + w_2 + \dots + w_M) \tag{27}$$

Since the terms $w_1, \dots , w_M$ will in general have different signs, their summation usually yields a small net result; in short, even if the individual noise terms may have large values, the total noise term will usually be small (barring exceptional cases). Hence a linear predictor has higher noise robustness than a nonlinear one. This principle has been verified in our logistic experiments and appears to have general application for many types of time series.

Two different decision modules have been used: a sum/product module and a max/min module. Depending on the type of predictors used (linear, sigmoid, structured) several different combinations of prediction / decision modules are obtained; the value of c is listed for each combination. The σ, h, M and μ_0^k parameters were chosen as follows: σ was always set equal to the standard deviation of prediction error, computed during the training phase of each predictor; h was set to 0.01; M was chosen for each predictor used, so as to get good prediction while avoiding overfitting; μ_0^k was always taken to be $1/K$. No attempt is made to fine tune these parameters; the values chosen were found to work quite well. Experimenting with other values we have found that PREMOFS is characterized by considerable robustness to parameter choice.

In every experiment set, classification experiments are repeated with the original time series being mixed with additive white noise, uniformly distributed in some interval $[-\frac{A}{2}, \frac{A}{2}]$, for various A values (i.e. various noise levels). The strength of noise signal relative to that of the time series is shown by listing a signal to noise ratio, i.e. signal energy ($\sum_t X_t^2$) divided by noise energy ($\sum_t n_t^2$). This yields a signal-to-noise ratio denoted by SNR. Training was performed on noise-free data.

6.1 Logistic and noise, black box predictor, complete search set

The first example concerns a case of signal detection: a signal must be discriminated from white noise. The signal used is the *logistic* time series (see [1]) generated by the following difference equation

$$X_t = \alpha \cdot X_{t-1} \cdot (1 - X_{t-1}) \qquad t = 1, 2, ... \tag{28}$$

Here α plays the role of source parameter Z. A test time series has been generated by running a logistic with α=3.9 for 950 time steps, then appending 950 time steps of white noise, uniformly distributed in the [0,1] interval. The task is to classify each step of the time series to logistic (with α= 3.9) or noise. This is a rather hard task, as the logistic with $\alpha = 3.9$ is a chaotic time series, similar to white noise, both statistically ([25], p.110) and visually (in Fig.1 observe the time series at SNR=6.00).

It is assumed that a part of the time series has been generated by a logistic, but the exact value of α is not known; it is conjectured that it lies in the interval [3.6,3.9]. Hence, in the off-line phase a bank of four predictors has been trained on noise-free versions of logistic time series, generated by the following values of α: 3.6, 3.7, 3.8, 3.9. An additional predictor is trained on white noise data. [4] Of course the train data are different from the test data; this holds true for this as well as all subsequent experiments. Hence a bank of five predictors is used in the on-line phase of PREMOFS.

[4]Does a predictor learn something when trained on white noise data? Consider the optimal predictor (of any form) X_t^* for white noise uniformly distributed in [0,1]: it is the conditional expectation $X_t^*= E(X_t|X_{t-1}, X_{t-2}, ..., X_1)$, which, because of the independence of white noise, equals the unconditional expectation $E(X_t)$= 0.5 (in other words a constant predictor) (for more details see [5], p.64). Now consider a *linear* predictor of the form $\hat{X}_t= w_1 X_{t-1}+ w_2 X_{t-2}+ ...+ w_M X_{t-M}$, for some M. When trained, $\hat{X}_t$ should approximate the optimal predictor X_t^*= 0.5. As $\hat{X}_t$ depends on a number of variables $(X_{t-1}, X_{t-2}, ...X_{t-M})$, it cannot be constant; but if a large M is used and if w_1, ..., w_M are approximately equal, then the predictor $\hat{X}_t$ essentially takes an average of past values $X_{t-1}, X_{t-2}, ...X_{t-M}$, each approximately equally weighted, hence it approximates $E(X_t)$. This is exactly what we found to happen with the linear predictor of white noise trained for this experiment, in contrast to the linear predictors of logistic time series, where w_j values vary considerably, reflecting different correlations between X_t, X_{t-1}, X_{t-2} etc. A similar, but more complicated comparison pertains to the sigmoid predictors.

Two separate banks of predictors have been used: 12-th order linear predictors, in one case, 18-5-1 sigmoid feedforward neural networks (i.e. 18 input, 5 hidden, one output neurons), in the other case. In the case of max/min PREMOFS, N (error order) was taken equal to 50. In all, there are four combinations of predictor and decision modules, so the classsification experiment is repeated four times, once for each combination.

In Table I the c values are listed for four different PREMOFS implementations: sigmoid predictor with max/min decision, sigmoid predictor with sum/product decision, linear predictor with max/min decision and linear predictor with sum/product decision. Fig.2 is a plot of the evolution of membership grades as computed by the max/min method; Fig.3 is a plot of the evolution of membership grades as computed by the sum/product method. Both cases pertain to sigmoid predictor experiments and a noise level SNR=12.50. A similar plot is presented in Fig.4 using a linear predictor, sum/product decision and noise level SNR=3.33. Figures 2 - 4 and Table I indicate that the logistic/noise time series is generally classified at a high level of accuracy. For the noise free case, sigmoid predictor and sum/product decision give the best results with a classification figure of merit c over 0.95. Linear predictors, on the other hand, are more robust to noise. Max/min and sum/product decision perform almost equally well.

6.2 Logistic and noise, black box predictor, incomplete search set

This set of experiments again concerns signal detection. The setup is identical to the previous experiment set, except in that it is *incorrectly* assumed that the logistic α lies in the interval [3.5,3.8]; it is intended to investigate the behavior of the PREMOFS when the true model is not included in the search set. Accordingly, the predictors are trained on α values 3.5, 3.6, 3.7, 3.8 and on white noise. For the logistic part of the time series, classification will be considered correct if the closest available source is chosen, namely α=3.8. For the noise part there is no change. Experimental results (on a new test time series) are summarized in Table II; these support the same conclusions as in section 6.1. Namely, sigmoid predictors are more accurate at low noise levels but linear predictors are more robust to noise. In addition it is evident that even when the true class does not appear in the search set, PREMOFS yields quite accurate classification in the sense that the closest class available is selected.

6.3 Logistic and logistic, black box predictor, complete search set

This set of experiments can be considered as an exercise in black-box system identification. Again, a 1900 steps time series is used; the first 950 steps have been generated by a logistic with $\alpha = 3.6$ and the last 950 steps by a logistic with $\alpha = 3.7$. The search set is the same as in 6.1; The task is to find the correct value of α. The results of this experiment set (on a new test time series) are summarized in Table III; they are similar to those of the previous experiment sets, except that in this task the sigmoid predictors are found to be quite sensitive to noise. In Fig.5 the membership grades evolution is presented for the combination of linear predictor and max/min decision, while in Fig.6 it is presented for the combination of linear predictor and sum/product decision; in both cases noise level is SNR=5.00.

6.4 Logistic and noise, structured predictor, complete search set

In this set of experiments, a signal detection task is again considered. Two points are investigated: the use of structured predictors, and performance with a large search set. A 1900 steps time series is used; the first 950 steps of it has been generated by a logistic with $\alpha = 3.9$ and the last 950 steps by white noise uniformly distributed in [0,1]. The difference between this and previous experiments is in the predictor modules and search set. Namely, nine *structured* logistic predictors are used, having the same form as eq.(28) and α= 3.1, 3.2, ... , 3.9. The noise predictor is $\hat{X}_t$= 0.5; it is structured in the sense that prior information about the time series is used. As remarked in footnote 10, this is the optimal predictor for uniform white noise in the interval [0,1]. In the case of max/min PREMOFS, N (error order) was taken equal to 50. This experiment set also illustrates the use of different forms of predictor modules (the form of the predictor equation is different for the logistic and noise predictors). The results of this experiment set are summarized in Table IV.

In terms of noise robustness, the performance of structured logistic predictors is somewhere between that of linear and sigmoid predictors used in the previous experiment sets. Namely, structured logistic predictors perform very well up to relatively high noise level, but then performance gradually drops off. We attribute this phenomenon to the chaotic nature of the logistic mapping, which is known to be very sensitive to small deviations in initial conditions [1]. Hence, a relatively small level of noise in the observation of past X_t values can lead all predictors to almost equally large prediction errors; this cannot be compensated by the competitive nature of PREMOFS computations, since all predictors are about equally bad. Regarding the use of a larger predictor bank, while in principle this might lead to a deterioration of classification performance (more predictors/ classes are competing at every time step), it is observed that PREMOFS performs quite satisfactorily. Finally, as in previous experiment sets, the sum/product decision module generally performs better than the max/min one, but the distinction is not so sharp.

6.5 Noise, structured predictor, incomplete search set

This experiment set is concerned with a time series which is completely mismatched to the available predictors. A 2000 steps time series is used, consisting solely of white noise, uniformly distributed in [0,1]. The search set consists of *ten* logistic predictors with α= 3.1, 3.2, ... , 3.9, 4.0. In the case of max/min PREMOFS, N (error order) was taken equal to 50. The point of the experiment set is to determine whether PREMOFS will provide some indication that the time series observed bears no relation to the available classes. According to Theorems 1 and 2, classification will converge to the class with minimum average square prediction error. It can be proved that this is the predictor with α= 3.1, but also that prediction errors are very close for all predictors (proofs are omitted, for lack of space). Hence, $\alpha = 3.1$ is taken to be the "best" class. Classification results are presented in Table V.

Results of Table V indicate, on the one hand, that PREMOFS classifies to the class with minimum average square prediction error, as expected by the theoretical analysis. On the other hand, this result may mislead the user into believing that a logistic time series with α= 3.1 is observed, while in fact the time series is white noise. However, PREMOFS does give an implicit warning signal, through the profile of the membership grades. In Fig.7, the irregular form and rapid variation of the membership grade profile is due to large prediction errors, which are almost uniformly spread among all predictors. Since all predictors are about equally bad, the competitive function of PREMOFS does not give a clear advantage to any predictor. The membership grade profiles of Fig.7 should be compared to that of Figs.3, 4, where a single predictor has a clear advantage over all others. Such irregular profiles can serve as a useful diagnostic for a reevaluation of the search set.

6.6 Motor, structured predictor, complete search set

In this experiment set a more realistic example of system identification is considered, involving measurements generated from simulation of an AC induction motor. The operation of the AC induction motor is described (in discrete time) by the following nonlinear state equations.

$$Y(t) = \delta t \cdot A^{-1} \cdot \{B \cdot Y(t-1) + U(t)\} \tag{29}$$

$$w(t) = w(t-1) + \delta t \cdot \left\{ \frac{3P^2}{2J} \left(i_{qs}(t-1) i_{dr}(t-1) - i_{ds}(t-1) i_{qr}(t-1) \right) - T_L(t) \right\}, \tag{30}$$

where $Y(t)$= $[i_{qs}(t), i_{ds}(t), i_{qr}(t), i_{dr}(t)]$, $U(t)$= $[V_{qs}(t), V_{ds}(t), 0, 0]$ and

$$A = \begin{bmatrix} L_s & 0 & L_0 & 0 \\ 0 & L_s & 0 & L_0 \\ -L_0 & 0 & L_s & 0 \\ 0 & -L_0 & 0 & L_s \end{bmatrix} \quad B = \begin{bmatrix} R_s & 0 & 0 & w(t-1)L_s \\ 0 & R_s & w(t-1)L_0 & 0 \\ 0 & -w(t-1)L_0 & R_s & w(t-1)L_0 \\ -w(t-1)L_s & 0 & 0 & R_r \end{bmatrix}. \tag{31}$$

Here $i_{qs}(t)$, $i_{ds}(t)$ are stator currents, $i_{qr}(t)$, $i_{dr}(t)$ are rotor currents, $w(t)$ is angular velocity, $V_{qs}(t)$, $V_{ds}(t)$ are stator voltages and $T_L(t)$ is torque. δt is the integration step; R_s, R_r are stator and rotor resistances, L_s, L_r, L_0 are stator, rotor and mutual inductances; J is moment of inertia and P is number of pole pairs. $Y(t)$ and $w(t)$ are the states; $U(t)$ and $T_L(t)$ are the inputs. All the parameters are known, except for R_r, which depends on operating conditions. However, is necessary for the determination of the motor time constant T_r, which in turn is necessary for efficient and economic angular velocity control. Hence this problem is a typical example of on-line parameter estimation. Various methods have been proposed to solve this problem [20, 21]. To apply PREMOFS to the R_r estimation task, the problem must be converted to one of time series classification. We measure $i_{qs}(t)$, $i_{ds}(t)$; the vector sequence $[i_{qs}(t), i_{ds}(t)]$ (t=1, 2, ...) is considered to be the time series X_t (t=1, 2, ...) which may have been generated by any of a number of sources, each with a particular value R_r; in other words R_r plays the role of the source parameter Z. The parameter estimation task has been converted to the task of classifying the observable time series to one of a number of classes; correct classification (at a given time step) will yield the (approximately) correct value of R_r.

The AC induction motor is simulated, and the stator current observations mixed with additive noise at various noise levels. Each simulation is run for 10000 time steps, each step corresponding to 0.5 milliseconds of real time; hence the operation of the motor is simulated for a 5 second time interval. Input is a three phase AC voltage of 220 Volts RMS value and torque T_L=1.5 N·m. The actual motor has the following parameters: R_s=11.58 Ohm, L_s =0.071 Henry, L_r=0.072 Henry, L_0=0.069 Henry,

J=0.089 kg·m^2 , B=0 Nt·sec/m, P=2. To simulate the effect of R_r variation, *ten* R_r values are used: from time t=0.0 to 0.5 seconds R_r= 4.9 Ohm, from 0.5 to 1.0 seconds R_r=5.9 Ohm and so on until the value 13.9 Ohm. We use a bank of ten prediction modules (K=10), tuned to R_r values of 5, 6,..., 14 Ohm. When actual R_r value is 4.9, the best estimate is 5 Ohm; similarly for R_r=5.9, 6.9, ...

For the integration of eqs.(29), (30) the time step was δt=0.5 msec. However, several sampling times were used for the observation of the stator current, namely δs= 0.5, 1, 2, 3 msec. Larger sampling time implies that less information is obtained about the operation of the motor and fewer comparisons are performed between the true system and the predictors. Presumably, this makes the identification task harder; on the other hand it ameliorates computational load and makes the method more attractive for on-line application. In the case of max/min PREMOFS, N (error order) was taken equal to 10. The classification results for various noise levels and sampling times are presented in Table VI. It can be seen that PREMOFS has good performance for quite high noise levels.

6.7 Speech, black box predictor, complete search set

The last example is a case of phoneme classification, involving real world data. The time series used are two utterances of the word "one", one for training and one for testing. Both have been sampled initially at 10 KHz, then subsampled at 2 KHz. Three sources (phonemes [oo], [ah] and [nn]) are contained in the data. Three linear prediction modules of the form $\hat{y}_t^k = w_1^k \cdot y_{t-1} + ... + w_{18}^k \cdot y_{t-18}$, $k = 1, 2, 3$ have been trained, using data from the first utterance of the three phonemes. In the case of max/min PREMOFS, N (error order) was taken equal to 10. The predictors are combined into a sum/product PREMOFS and a max/min PREMOFS and several classification experiments are run, using the second utterance of "one" with additive noise superposed at various levels. The noise free speech time series is plotted in Fig.8. In Fig.9 the evolution of membership grades is presented for the noise-free case.

In Fig.8 two transition regions of the time series can be observed, between time steps 250 and 300, and 550 and 600, where the time series does not correspond to any phoneme. It could be said that *two* sources/phonemes are active over each transition region ([oo] and [ah] for the first region, [ah] and [nn] for the second). This is well reflected in the respective portions of membership function profiles, as displayed in Fig.9. In the transition regions the membership grades of the two sources fluctuate widely before they settle into steady state. Corresponding to this phenomenon of concurrently active sources, classification results can be computed in two different ways. Namely, the transition regions can be included in the computation of c, or removed from consideration; the second option is more realistic, since the transition regions cannot be properly classified to any phoneme. Classification results presented in Table VII, are computed in both ways: the second and third columns of the table give classification results on the complete time series, while the fourth and fifth columns give classification results with the transition regions deleted.

7 Conclusion

We have presented a Predictive Modular Fuzzy System for classification of time series to one of a finite number of classes. Two varieties of PREMOFS were presented, depending on the method used to implement the fuzzy AND and OR operators (max/min and sum/product). PREMOFS has the following characteristics.

1. Classification is based on membership function of the entire time series to a universal set of sources / classes. The computation of the membership function is based on the *predictive* accuracy of each source for the observed time series.

2. PREMOFS is *modular* and *parallelizable*. This results in good scaling properties and high execution speed. The max/min PREMOFS can be implemented by very simple hardware.

3. The computation is *recursive* and thus well suited to on-line classification. *No preprocessing* of data is required.

4. The assignment of membership grades is *competitive*: what matters is not how *well* a particular source fits the data, but how much *better* than other sources. As a result, PREMOFS is highly robust to noise and prediction error.

Convergence of classification to the correct class has been mathematically proved for both varieties of PREMOFS. A number of experiments have been presented which corroborate our mathematical results. Theory and experiment indicate that PREMOFS is an efficient, simple and hence attractive method for dynamic pattern classification. Many possible variations of PREMOFS have been discussed only briefly. Such variations can be created by changing either the prediction or the decision modules. In addition, there is a number of parameters (σ, h, M, N, μ_0^k) which can be used to optimize performance.

Several issues merit further research. We have dealt here exclusively with the case of a finite source set. This is an important case, but it is also desirable to extend the use of PREMOFS to infinite (especially continuous) source sets. One method to achieve this involves a system of progressively narrowing grids. At every classification *epoch* a fixed grid with a finite number of nodes is used; PREMOFS finds the "best" node; then a narrower grid is built around this node and a new epoch is started. A desirable result would be to prove *consistency* of this method, in the limit: as the grid size goes to zero, so does the error in parameter estimate. Another way to deal with an infinite source set uses adaptive generation and training of source predictors. At any given time only a finite number of these are actively considered (i.e. their μ_t^k values updated); but if some μ_t^k becomes too small, the respective θ_k is discarded and replaced by a new one in the search set. In this way we have a finite active source set, but a potentially infinite total source set.

A Appendix

Here we present the proofs of Theorems 1 and 2 of Section 4.
Proof of Theorem 1: Take any $k \neq m$; from eq.(15) we have

$$\frac{\mu_t^k}{\mu_t^m} = \frac{\mu_{t-1}^k}{\mu_{t-1}^m} \cdot \frac{e^{-\left(\frac{|e_t^k|}{\sigma}\right)^2}}{e^{-\left(\frac{|e_t^m|}{\sigma}\right)^2}} \Rightarrow$$

(repeating the argument for $s = t-1,\ t-2,\ \dots\ ,1$)

$$\frac{\mu_t^k}{\mu_t^m} = \frac{\mu_0^k}{\mu_0^m} \cdot \frac{e^{-\left(\frac{\sum_{s=1}^t |e_t^k|^2}{\sigma^2}\right)}}{e^{-\left(\frac{\sum_{s=1}^t |e_t^m|^2}{\sigma^2}\right)}} \Rightarrow \sqrt[t]{\frac{\mu_t^k}{\mu_t^m}} = \sqrt[t]{\frac{\mu_0^k}{\mu_0^m}} \cdot \frac{e^{-\left(\frac{\sum_{s=1}^t |e_t^k|^2}{t}\frac{1}{\sigma^2}\right)}}{e^{-\left(\frac{\sum_{s=1}^t |e_t^m|^2}{t}\frac{1}{\sigma^2}\right)}}. \tag{32}$$

Using the limit of Cesaro average of $|\ e_s^k\ |^2$, $|\ e_s^m\ |^2$ and the continuity of the exponential function, it is concluded that for every $\epsilon > 0$ there is some t_ϵ such that for all $t > t_\epsilon$ we have

$$\sqrt[t]{\frac{\mu_t^k}{\mu_t^m}} < \sqrt[t]{\frac{\mu_0^k}{\mu_0^m}} \cdot e^{-\left(\frac{D^k - D^m - \epsilon}{\sigma^2}\right)}. \tag{33}$$

However, since by assumption **A2** D^k is strictly larger than D^m, one can find some ϵ small enough that the exponent above is negative; this in turn implies that the exponential in (33) equals some ρ, with $0 < \rho < 1$. Substitute ρ in (33) and raise to the t-th power; for every $t > t_\epsilon$ to get

$$\frac{\mu_t^k}{\mu_t^m} < \frac{\mu_0^k}{\mu_0^m} \cdot \rho^t. \tag{34}$$

Since $\rho < 1$, taking the limit as t goes to infinity

$$\lim_{t\to\infty} \frac{\mu_t^k}{\mu_t^m} = 0 \quad \forall k \neq m;$$

this, combined with the fact that $\sum_{l=1}^K \mu_t^l = 1$ gives the required conclusion of the theorem:

$$\lim_{t\to\infty} \mu_t^m = 1 \quad \text{and for } \ k \neq m \lim_{t\to\infty} \mu_t^k = 0 \tag{35}$$

and the proof is concluded. •

Proof of Theorem 2: Multiplying eq.(25) by -1 and exponentiating, one gets

$$1 > \alpha > a_t^m \geq \beta > \gamma \geq a_t^k > 0 \quad k \neq m; \tag{36}$$

$\alpha = e^{-A}$, $\beta = e^{-B}$, $\gamma = e^{-C}$, $a_t^l = e^{-D_t^l}$, for $l = 1, 2, ..., K$; update equation (20) becomes

$$\mu_t^k = \frac{\mu_{t-1}^k \wedge a_t^k}{\bigvee_{l=1}^K (\mu_{t-1}^l \wedge a_t^l)}. \tag{37}$$

Suppose that for some time s we have $\mu_s^m < a_s^m$; then, since $\mu_s^m \wedge a_s^m$ is the minimum of μ_s^m, a_s^m, we must have $\mu_s^m \wedge a_s^m = \mu_s^m$ and eq.(37) becomes

$$\mu_{s+1}^m = \frac{\mu_s^m}{\bigvee_{l=1}^K (\mu_s^l \wedge a_s^l)}. \tag{38}$$

On the other hand, for $l = 1, 2, ..., K$

$$\mu_s^l \wedge a_s^l \leq a_s^l \Rightarrow \bigvee_{l=1}^{K} (\mu_s^l \wedge a_s^l) \leq \bigvee_{l=1}^{K} a_s^l = a_s^m \tag{39}$$

where the last maximum equals a_s^m by (36). Use (39) in the right hand side denominator of (38):

$$\mu_{s+1}^m = \mu_s^m \cdot \frac{1}{\bigvee_{l=1}^{K} (\mu_s^l \wedge a_s^l)} \geq \mu_s^m \cdot \frac{1}{a_s^m} > \mu_s^m \cdot \frac{1}{\alpha}. \tag{40}$$

The last inequality follows from (36). Now, applying (40) n times we get

$$\mu_{s+n}^m > \mu_t^m \cdot (\frac{1}{\alpha})^n \tag{41}$$

and taking n large enough, μ_{s+n}^m will get larger than a_{s+n}^m, which is bounded above. In short: for some t_0

$$\mu_{t_0}^m \geq a_{t_0}^m \Rightarrow \mu_{t_0}^m \wedge a_{t_0}^m = a_{t_0}^m \geq \beta; \tag{42}$$

at the same time, for $k \neq m$

$$\mu_{t_0}^k \wedge a_{t_0}^k \leq a_{t_0}^k \leq \gamma. \tag{43}$$

Combining (42) and (43) (and using the assumption $\beta > \gamma$) it is concluded that

$$\bigvee_{l=1}^{K} (\mu_{t_0}^l \wedge a_{t_0}^l) = \bigvee_{l=1}^{K} a_{t_0}^l = a_{t_0}^m \Rightarrow \mu_{t_0+1}^m = \frac{a_{t_0}^m}{a_{t_0}^m} = 1. \tag{44}$$

But then $1 = \mu_{t_0+1}^m \geq a_{t_0+1}^m$ and the argument can be repeated from (42) down. From this follows that there is some t_0 such that **for all** $t \geq t_0$ we have $\mu_t^m = 1$. This yields the first part of the theorem. Similarly, one sees that **for all** $t \geq t_0$ and $k \neq m$

$$\mu_{t+1}^k \leq \frac{a_t^k}{a_t^m} < \frac{\gamma}{\beta} < 1; \tag{45}$$

taking lim sup on both sides of (45) we obtain the second part of the theorem; the proof is concluded. •

It is worth emphasizing that, if the conditions of Theorem 2, (in particular **B2**) are satisfied exactly, then μ_t^m will increase monotonically and will achieve the value 1 in a finite number of steps; then it will never decrease. μ_t^m temporarily becomes different from one, during a source switch, when it starts from a small initial value μ_0^m; then it will increase monotonically until it becomes one. This is an immediate consequence of the proof presented above. On the other hand, in realistic experiments, e.g. in the simulations of Section 6, **B2** may be temporarily violated, especially if N (the E_t^k error order) is small and the observations are very noisy. In such cases temporary decreases of μ_t^m may be observed. However, if the assumptions of Theorem 2 are satisfied, the proof of the theorem also shows that $\mu_t^m = 1$ is a stable equilibrium point in the following sense: if $\mu_t^m = 1$ then $\mu_{t+1}^m = 1$; if $\mu_t^m < 1$ then $\mu_{t+n}^m = 1$ for some finite n. Therefore, if **B2** is temporarily violated by some perturbations, but is later restored, the system will return to equilibrium in a finite number of steps.

References

[1] D.K. Arrowsmith and C.M.Place, *An Introduction to Dynamical Systems*, Cambridge Un. Press, Cambridge, 1990.

[2] E. Backer and A.K. Jain, “A Clustering Performance Measure Based on Fuzzy Set Decomposition”, *IEEE Trans. on Pattern Anal. and Mach. Intel.*, vol. 3,no.1, pp.66-75, January 1981.

[3] J.C. Bezdek and J.D. Harris “Fuzzy Partitions and Relations: an Axiomatic Basis for Clustering”, *Fuzzy Sets and Systems*, vol.1, pp. 111-127, 1978.

[4] P. Billingsley, *Probability and Measure*, Wiley, New York, 1986.

[5] P.J. Brockwell and R.A. Davis, *Time Series: Theory and Methods*, Springer, New York, 1987.

[6] B.B. Devi and V.V.S. Sarma, “A Fuzzy Approximation Scheme for Sequential Learning in Pattern Recognition”, *IEEE Trans. on Sys. Man and Cyb.*, vol. SMC-16, no.5, pp.668-679, September 1986.

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

[8] J.C. Dunn, “A Fuzzy Relative of the ISODATA Process and its Use in Detecting Compact Well-Separated Clusters”, *J. of Cybernetics*, vol.3, no.3, pp.32-57, 1973.

[9] I. Gath and A.B. Geva, “Unsupervised Optimal Fuzzy Clustering”, *IEEE Trans. on Pattern Anal. and Mach. Intel.*, vol. 11,no.7, pp.773-781, July 1989.

[10] S. Haykin and D. Cong, “Classification of radar clutter using neural networks”, *IEEE Trans. on Neural Networks*, vol.2, pp. 589-600, November 1991.

[11] R.J. Hathaway and J.C. Bezdek, “Switching Regression Models and Fuzzy Clustering”, *IEEE Trans. on Fuzzy Systems*, vol. 1, no.3, pp.195-204, August 1993.

[12] J. Hertz, A. Krogh and R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, 1991.

[13] C. G. Hilborn and D.G. Lainiotis, “Unsupervised Learning Minimum Risk Pattern Classification for Dependent Hypotheses and Depedent Measurements”, *IEEE Trans. on Systems Science and Cybernetics*, vol.5, pp. 109-115, April 1969.

[14] R.A. Jacobs, M.I. Jordan, S.J. Nowlan and G.E. Hinton, “Adaptive mixtures of local experts”, *Neural Computation*, vol.3, pp.79-87, 1991.

[15] M.I. Jordan and R.A. Jacobs, “Hierarchical Mixtures of Experts and the EM Algorithm”, *Neural Computation*, vol.6, pp. 181-214, 1994.

[16] F. Kanaya and S. Miyake, "Bayes statistical behavior and valid generalization of pattern classifying neural networks", *IEEE Trans. on Neural Networks*, vol.2, pp. 471-475, July 1991.

[17] N. Karayannis, "MECA: Maximum Entropy Clustering Algorithm", in *Proc. of the 3rd IEEE Conf. on Fuzzy Systems*, pp.630-635, 1994.

[18] A. Kehagias and V. Petridis, "Predictive Modular Neural Networks for Time Series Classification", to appear in *Neural Networks*.

[19] H.M. Kim and J.M. Mendel, "Fuzzy Basis Functions: Comparison with Other Basis Functions", *IEEE Trans. on Fuzzy Systems*, vol.3, no.2, pp. 158-168, May 1995.

[20] R. Krishnan and F.C. Doran, "Study of parameter sensitivity in high-performance and inverter-fed induction motor drive systems", *IEEE Trans. on Ind. Appl.*, vol. 23, no.4, pp.263-265, July/August 1987.

[21] R. Krishnan and F.C. Doran, "A review of parameter sensitivity and adaptation in indirect vector controlled induction motor drive", *IEEE Trans. on Power Electr.*, vol.6, no.4, pp. 695-703, October 1991.

[22] D.G. Lainiotis, "Optimal adaptive estimation: structure and parameter adaptation", *IEEE Trans. on Automatic Control*, vol.16, pp. 160-170, April 1971.

[23] D.G. Lainiotis and K.N. Plataniotis, "Adaptive Dynamic Neural Network Estimation", in *Proc. of IJCNN 1994*, vol.6, pp. 4736-4745, 1994.

[24] S. Layeghi et al., "Pattern Recognition of the Polygraph Using Fuzzy Classification", in *Proc. of the 3rd IEEE Conf. on Fuzzy Systems*, pp.1825-1829, 1994.

[25] E. Levin, "Hidden control neural architecture modeling of nonlinear time varying systems and its applications", *IEEE Trans. on Neural Networks*, vol.4, no.1, pp.109-116, January 1993.

[26] C.T. Lin and C.S.G. Lee, "Reinforcement Structure/Parameter Learning for Neural Network-Based Fuzzy Logic Control Systems", *IEEE Trans. on Fuzzy Systems*, vol.2, no.1, pp.46-63, February 1994.

[27] Y. Lin and G.A. Cunningham III, "A new approach to fuzzy-neural system modelling", *IEEE Trans. on Fuzzy Systems*, vol.3, no.2, pp. 190-199, May 1995.

[28] K. Nozaki et al, "Selecting Fuzzy Rules with Forgetting in Fuzzy Classification Systems", in *Proc. of the 3rd IEEE Conf. on Fuzzy Systems*, pp.618-623, 1994.

[29] Y.M. Park, U. Moon and K.Y. Lee, "A Self-organizing Fuzzy Logic Controller for Dynamic Systems using a FARMA Model", *IEEE Trans. on Fuzzy Systems*, vol. 3, no.1, pp.75-82, February 1995.

[30] S.K. Pal and D.D. Majumder, "Fuzzy Sets and Decision making Approaches in Vowel and Speaker Recognition", *IEEE Trans. on Sys. Man and Cyb.*, pp.625-629, August 1977.

[31] V. Petridis and A. Kehagias, "A Recursive Network for Time Series Classification", *Neural Computation*, vol.8, pp.357-372, 1996.

[32] V. Petridis and A. Kehagias, "Recursive Modular Neural Networks and the Partition Algorithm for Time Series Classification", *IEEE Trans. on Neural Networks*, vol.7, no.1, pp.73-86, January 1996.

[33] S.J. Qin and G. Borders, "A Multi-Region Fuzzy Logic Controller for Nonlinear Process Control", *IEEE Trans. on Fuzzy Systems*, vol. 2, no.1, pp.74-81, February 1994.

[34] R. Rovatti, R. Guerrieri and G. Baccarani, "An enhanced two-level Boolean synthesis methodology for fuzzy rules minimization", *IEEE Trans. on Fuzzy Systems*, vol.3, no.3, pp. 288-300, August 1995.

[35] E.H. Ruspini, "A New Approach to Clustering", *Inform. and Control*, vol.15, no.1, pp.22-32, July 1969.

[36] F.L. Sims, D.G. Lainiotis and D.T. Magill, "Recursive algorithm for the calculation of the adaptive Kalman filter coefficients", *IEEE Trans. on Automatic Control*, vol.14, pp.215-218, April 1969.

[37] M. Sugeno and T. Yasukawa, "A Fuzzy logic-based approach to qualitative modeling", *IEEE Trans. on Fuzzy Systems*, vol.1, no.3, pp. 7-32, February 1993.

[38] E.A. Wan, "Neural network classification: a Bayesian interpretation", *IEEE Trans. on Neural Networks* vol.1, pp.303-304, December 1990.

[39] L.X. Wang and J.M. Mendel, "Fuzzy Adaptive Filters, with Application to Nonlinear Channel equalization", *IEEE Trans. on Fuzzy Systems*, vol. 1, no.3, pp.161-170, August 1993.

[40] M.P. Windham, "Cluster Validity for the Fuzzy c-Means Clustering Algorithm", *IEEE Trans. on Pattern Anal. and Mach. Intel.*, vol.4, no.4, pp.357-363, July 1982.

[41] A. Zardecki, "Fuzzy Control for Forecasting and Pattern Recognition in a Time Series", in *Proc. of the 3rd IEEE Conf. on Fuzzy Systems*, pp.1815-1819, 1994.

[42] L.H. Zetterberg et al., "Computer analysis of EEG signals with parametric models", *Proc. of the IEEE*, vol.69, pp.451-461, April 1981.

[43] K. Zhu et al., "Training neural networks for ECG feature Recognition", in *Proceedings of the IJCNN*, ed. M. Caudill. Washington, 1989.

SNR	Sig-Sum/Pro	Sig-Max/Min	Lin-Sum/Pro	Lin-Max/Min
Noise Free	0.952	0.886	0.905	0.911
25.00	0.948	0.858	0.909	0.911
12.50	0.934	0.821	0.904	0.921
8.33	0.893	0.759	0.903	0.910
6.00	0.835	0.728	0.899	0.915
5.00	0.822	0.649	0.906	0.901
4.33	0.855	0.630	0.900	0.922
3.50	0.748	0.604	0.929	0.916
3.33	0.616	0.516	0.931	0.942
2.80	0.546	0.532	0.927	0.919
2.50	0.557	0.450	0.893	0.841

TABLE I

Classification accuracy results for logistic-to-noise time series, black box predictor, complete search set.

SNR	Sig-Sum/Pro	Sig-Max/Min	Lin-Sum/Pro	Lin-Max/Min
Noise Free	0.962	0.887	0.955	0.953
25.00	0.960	0.877	0.958	0.951
12.50	0.959	0.858	0.955	0.956
8.33	0.957	0.843	0.963	0.959
6.00	0.936	0.821	0.959	0.954
5.00	0.950	0.785	0.951	0.920
4.33	0.942	0.763	0.968	0.945
3.50	0.950	0.718	0.970	0.917
3.33	0.682	0.506	0.966	0.935
2.80	0.711	0.514	0.951	0.897
2.50	0.311	0.453	0.961	0.940

TABLE II

Classification accuracy results for logistic-to-noise time series, black box predictor, incomplete search set.

SNR	Sig-Sum/Pro	Sig-Max/Min	Lin-Sum/Pro	Lin-Max/Min
Noise Free	0.509	0.686	0.945	0.900
25.00	0.931	0.889	0.943	0.925
12.50	0.919	0.927	0.945	0.905
8.33	0.615	0.779	0.946	0.863
6.00	0.423	0.422	0.925	0.852
5.00	0.005	0.098	0.918	0.849
4.33	0.006	0.051	0.911	0.813
3.50	0.019	0.040	0.820	0.650
3.33	0.001	0.004	0.805	0.599
2.80	0.005	0.006	0.875	0.684
2.50	0.003	0.002	0.560	0.520

TABLE III

Classification accuracy results for logistic-to-logistic time series, black box predictor, complete search set.

SNR	sum/pro	max/min
Noise Free	0.960	0.959
25.00	0.928	0.962
12.00	0.944	0.952
8.33	0.968	0.951
6.00	0.971	0.932
5.00	0.976	0.895
4.33	0.968	0.842
3.50	0.505	0.541
3.33	0.505	0.527
2.80	0.509	0.509

TABLE IV

Classification accuracy results for logistic-to-noise time series, structured predictor, complete search set.

SNR	sum/pro	max/min
Noise Free	0.984	0.916
20.00	0.974	0.901
10.00	0.984	0.884
6.66	0.985	0.850
5.00	0.984	0.859
4.00	0.968	0.834
3.33	0.984	0.837
2.85	0.984	0.923
2.50	0.984	0.889
2.22	0.985	0.876

TABLE V

Classification accuracy results for noise time series, structured predictor, incomplete search set.

SNR	$\delta s=$	0.0005 sec	$\delta s=$	0.0010 sec	$\delta s=$	0.0020 sec	$\delta s=$
	sum/product	max/min	sum/product	max/min	sum/product	max/min	sum/product
Noise Free	0.951	0.979	0.950	0.979	0.916	0.964	0.870
20.00	0.949	0.979	0.947	0.979	0.916	0.966	0.870
10.00	0.947	0.969	0.940	0.965	0.908	0.960	0.855
6.66	0.943	0.958	0.929	0.953	0.894	0.946	0.816
5.00	0.934	0.953	0.917	0.949	0.866	0.936	0.822
4.00	0.905	0.945	0.889	0.930	0.820	0.912	0.801
3.33	0.881	0.934	0.887	0.913	0.808	0.908	0.720
2.50	0.870	0.926	0.853	0.890	0.796	0.874	0.687

TABLE VI

Classification accuracy results for motor currents time series, structured predictor, complete search set.

SNR	Lin-Sum/Pro	Lin-Max/Min	Lin-Sum/Pro	Lin-Max/Min
	Complete TS	Complete TS	Deleted Trans.	Deleted trans.
Noise Free	0.816	0.807	0.912	0.903
33.30	0.838	0.810	0.913	0.900
16.60	0.832	0.781	0.916	0.878
12.50	0.821	0.785	0.912	0.818
8.30	0.798	0.726	0.835	0.775
6.25	0.748	0.603	0.827	0.692
5.00	0.688	0.638	0.775	0.715

TABLE VII

Classification accuracy results for speech time series, black box predictor, complete search set.

Figure 1: Plot of logistic time series, mixed with noise at SNR=6.00.

Figure 2: Plot of evolution of membership grades for logistic-to-noise time series at SNR=12.5, max/min decision module, sigmoid predictor. Thick "—" line is membership grade of source with $\alpha = 3.9$; thick "- . -" line is membership grade of white noise source; the remaining membership grades are denoted by thin "- -" lines. Note that the maximum membership grade always equals 1.

Figure 3: Plot of evolution of membership grades for logistic-to-noise time series at SNR=12.5, sum/product decision module, sigmoid predictor. Thick "- . -" line is membership grade of source with $\alpha = 3.9$; thick "—" line is membership grade of white noise source; the remaining membership grades are denoted by thin "- -" lines.

Figure 4: Plot of evolution of membership grades for logistic-to-noise time series at SNR=3.33, sum/product decision module, linear predictor. Thick "- . -" line is membership grade of source with $\alpha = 3.9$; thick "—" line is membership grade of white noise source; the remaining membership grades are denoted by thin "- -" lines.

Figure 5: Plot of evolution of membership grades for logistic-to-logistic time series at SNR=5.00, max/min decision module, linear predictor. Thick "—" line is membership grade of source with $\alpha = 3.6$; thick "- . -" line is membership grade of source with $\alpha = 3.7$; the remaining membership grades are denoted by thin "- -" lines. Note that the maximum membership grade always equals 1.

Figure 6: Plot of evolution of membership grades for logistic-to-logistic time series at SNR=5.00, sum/product decision module, linear predictor. Thick "—" line is membership grade of source with $\alpha = 3.6$; thick "- . -" line is membership grade of source with $\alpha = 3.7$; the remaining membership grades are denoted by thin "- -" lines.

Figure 7: Plot of evolution of membership grade for white noise time series at SNR=25, sum/product decision module, structured predictor. Thick "—" line is membership grade of source with $\alpha = 3.1$; the remaining membership grades are denoted by thin "- -" lines. Note the fluctuations in membership grade, indicating that the true source is not in the search set.

Figure 8: Plot of noise-free speech time series.

Figure 9: Plot of evolution of membership grade for noise-free speech time series, sum/product decision module, linear predictor. Thick "—" line is membership grade of phoneme [oo]; thick "- -" line is membership grade of phoneme [ah]; thick "..." line is membership grade of phoneme [n].

Figures for the paper "Predictive Modular Fuzzy Systems for Time Series Classification"

by V. Petridis and Ath. Kehagias

Fig.1

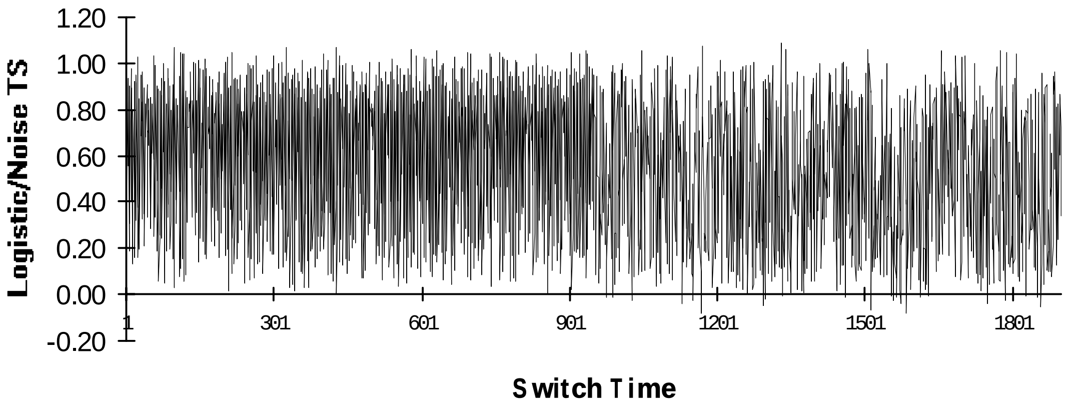


Fig.2

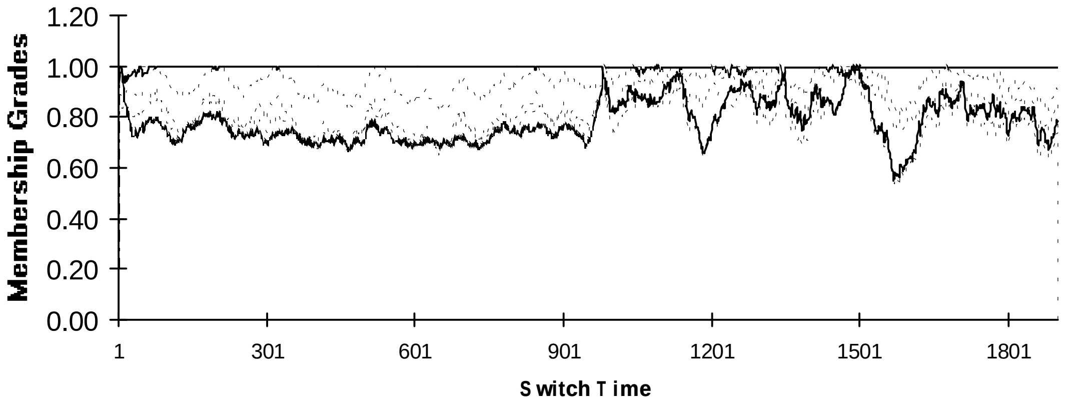


Fig.3

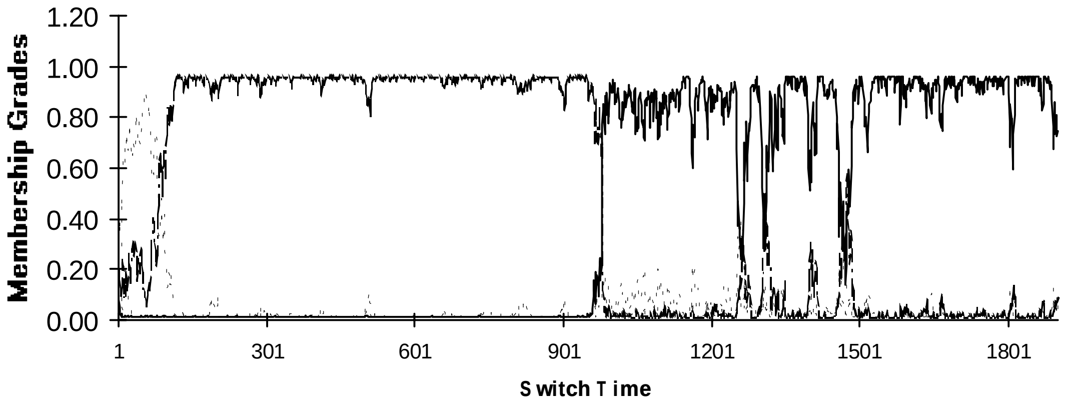

1.20
1.00
0.80
0.60
0.40
0.20
0.00
Membership Grades
1
301
601
901
1201
1501
1801
Switch Time

Fig.4

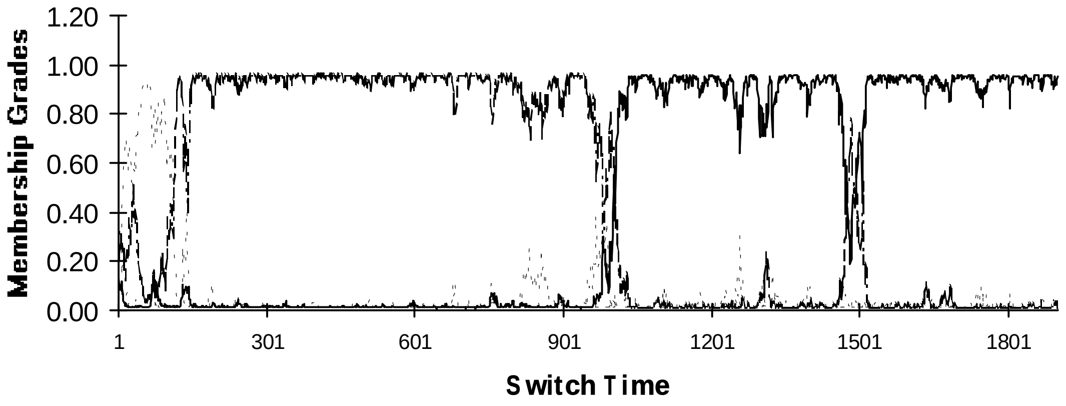
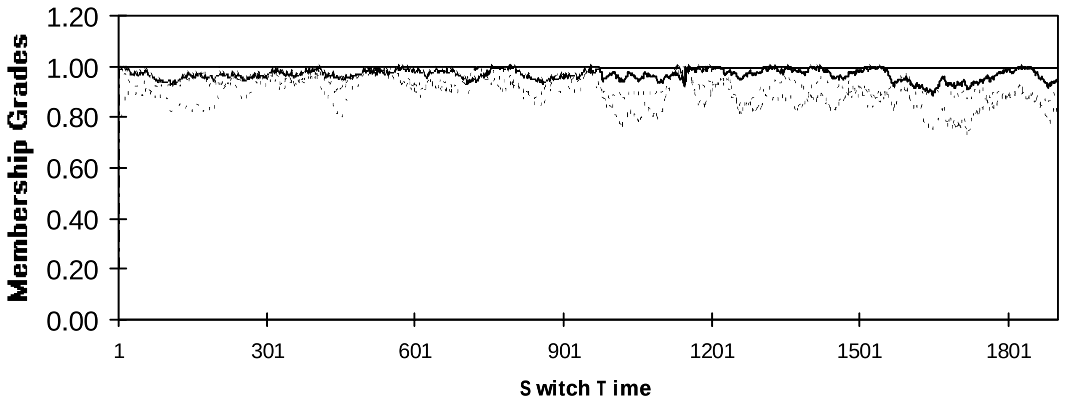
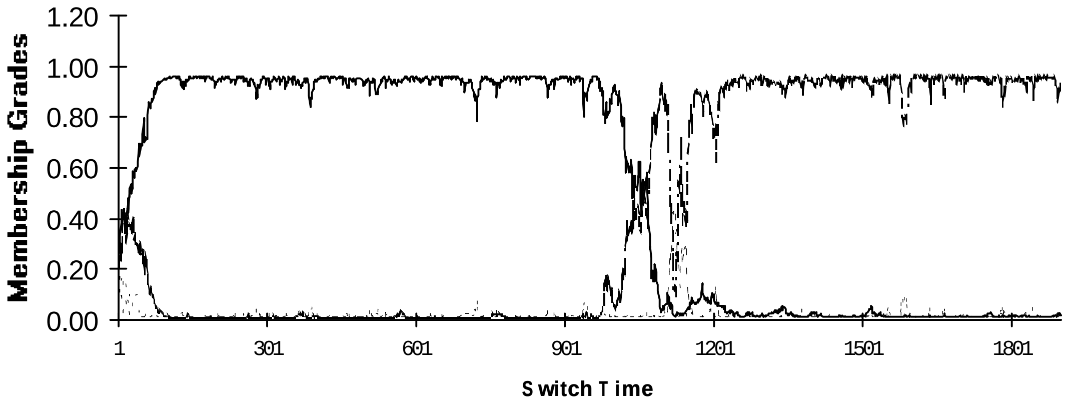

1.20
1.00
0.80
0.60
0.40
0.20
0.00
Membership Grades
1
301
601
901
1201
1501
1801
Switch Time

Fig. 5



1.20
1.00
0.80
0.60
0.40
0.20
0.00
Membership Grades
1
301
601
901
1201
1501
1801
Switch Time

Fig.6



1.20
1.00
0.80
0.60
0.40
0.20
0.00
Membership Grades
1
301
601
901
1201
1501
1801
Switch Time

Fig.7

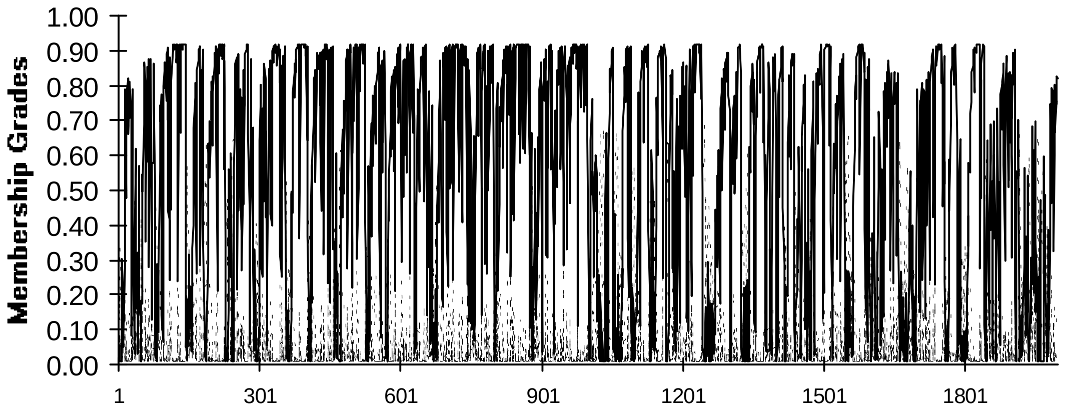

Membership Grades
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
1
301
601
901
1201
1501
1801

Fig.8

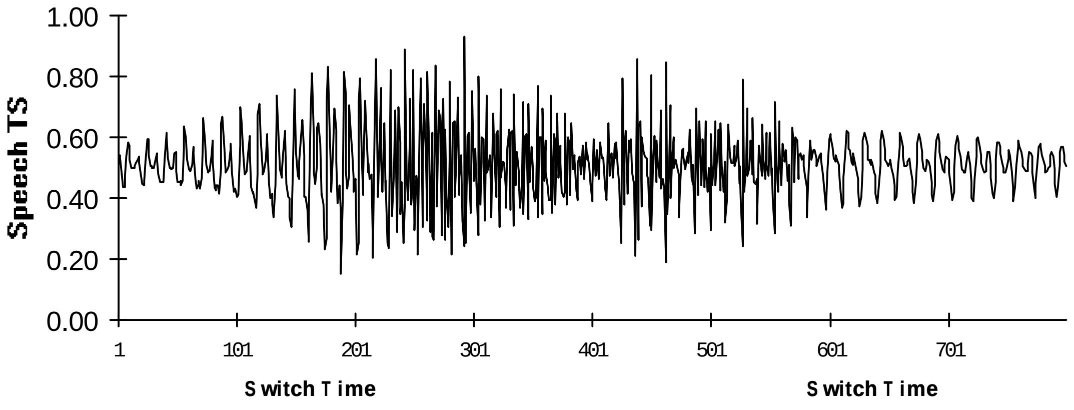
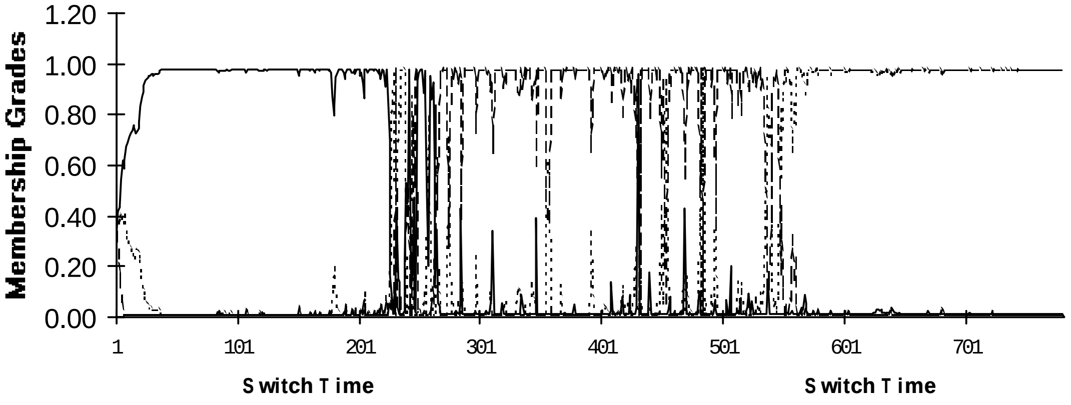

Speech TS
1.00
0.80
0.60
0.40
0.20
0.00
1
101
201
301
401
501
601
701
Switch Time
Switch Time

Fig.9



1.20
1.00
0.80
0.60
0.40
0.20
0.00
Membership Grades
1
101
201
301
401
501
601
701
Switch Time
Switch Time