Automatic Clustering-Based Identification of Autoregressive Fuzzy Inference Models for Time Series

Federico Montesino Pouzols*,[2]

Department of Information and Computer Science, Aalto University School of Science and Technology P.O. Box 15400, FI-00076 Aalto, Espoo, Finland.

Angel Barriga Barros

Department of Electronics and Electromagnetism, University of Seville Avda. Reina Mercedes s/n, E-41012 Seville, Spain.

Abstract

We analyze the use of clustering methods for the automatic identification of fuzzy inference models for autoregressive prediction of time series. A methodology that combines fuzzy methods and residual variance estimation techniques is followed. A nonparametric residual variance estimator is used for a priori input and model selection. A simple scheme for initializing the widths of the input membership functions of fuzzy inference systems is proposed for the Improved Clustering for Function Approximation algorithm (ICFA), previously introduced for initializing RBF networks. This extension to the ICFA algorithm is shown to provide the most accurate predictions among a wide set of clustering algorithms. The method is applied to a diverse set of time series benchmarks. Its advantages in terms of accuracy and computational requirements are shown as compared to least-squares support vector machines (LS-SVM), the multilayer perceptron (MLP) and two variants of the extreme learning machine (ELM).

Key words: Times series prediction, fuzzy inference, clustering, fuzzy clustering, supervised learning

1. Introduction

Time series prediction and analysis techniques find applications in virtually all areas of natural and social sciences as well as in engineering. In the time series prediction field, obtaining accurate predictions is not the only major goal. Understanding the behavior of time series and gaining insight into their underlying dynamics is a highly desired capability of time series prediction methods [50].

In the past, conventional statistical techniques such as AR and ARMA models have been extensively used for forecasting [6]. However, these techniques have limited capabilities for modeling time series data. More advanced nonlinear methods including artificial neural networks and other computational intelligence based techniques have been frequently applied with success [8].

Fuzzy logic based modeling techniques are appealing because of their interpretability and potential to address a broad spectrum of problems. In particular, fuzzy inference systems exhibit a combined description and prediction capability as a consequence of their rule-based structure [27, 49].

In practice, one finds two problems when building a fuzzy model for a time series: choosing variables or inputs to the inference system, and identifying the structure of the system (linguistic labels and rule base). Once these steps have been accomplished, the fuzzy model can be tuned through supervised learning techniques.

The first problem can be addressed by means of a priori feature selection techniques based on nonparametric residual variance estimation [19]. These techniques also provide

*Corresponding author. Phone: +358-9-470-25237. FAX: +358-9-470-23277.

Email addresses: fedemp@cis.hut.fi (Federico Montesino Pouzols), barriga@us.es (Angel Barriga Barros)

[1]The first author is supported by a Marie Curie Intra-European Fellowship for Career Development grant (agreement PIEF-GA-2009-237450) within the European Community´s Seventh Framework Programme (FP7/20072013). Most of this work was done while the first author was with the Microelectronics Institute of Seville, IMSE-CNM, CSIC – Scientific Research Council. C. Americo Vespucio s/n. Parque Tecnologico Cartuja. E-41092 Seville, Spain.

[2]This work was supported in part by project TEC2008-04920/MICINN from the Spanish Ministry of Science and Innovation, as well as project P08-TIC-03674 and grants IAC07-I-0205:33080 and IAC08-II-3347:56263 from the Andalusian regional Government.

an estimate of the error of the most accurate nonlinear model that can be built without overfitting.

The second problem can be addressed by data-driven techniques for identification of fuzzy systems from numerical examples. Two approaches are often distinguished in the literature: structure-oriented and clustering-based. It has been shown that certain types of fuzzy inference models identified using clustering methods rank among the best models in international times series competitions [31, 21], while yielding compact, interpretable models. In this paper we analyze the use of clustering-based identification methods for fuzzy inference systems in the context of time series prediction applications. Different clustering methods are used to perform the identification stage of an automatic methodology framework previously proposed for the design of fuzzy inference systems as autoregressors for time series prediction [33, 31, 32]. This methodology uses fuzzy clustering techniques and nonparametric residual variance estimation techniques in an intertwined manner. This way, the number of clusters that minimizes the generalization error is derived automatically from an a priori nonparametric residual variance estimate.

With these premises we propose a methodology for building compact, interpretable yet highly accurate fuzzy inference models. The proposed method, implemented in the Xfuzzy environment for the design of fuzzy inference systems [51, 34, 35], is applied to five datasets coming from diverse real-world time series applications. The results will be compared against least-squares support vector machines (LS-SVM) [45], a well established method in the field of time series prediction, that has been shown to be highly accurate. For further comparison, we will also show the results obtained using optimally-pruned extreme learning machine (OP-ELM) [44] and standard Extreme Learning Machine (ELM) models [17].

This article is organized as follows. The next section outlines a nonparametric residual variance estimation method that will be used for both variable and proper model complexity selection. In section 3 we define fuzzy inference autoregressors and describe how these are identified using clustering methods. Section 4 presents the stages of the methodology used to build fuzzy inference models. In section 5 we briefly illustrate the methodology through examples and analyze experimental results. Finally, section 5.8 further discusses

experimental results.

2. Nonparametric Residual Variance Estimation: Delta Test

Nonparametric residual variance estimation (NRVE or nonparametric noise estimation, NNE) is a well-known technique in statistics and machine learning, finding many applications in nonlinear modeling [19]. NRVE methods can be applied to recurrent problems such as variable and model structure selection. These methods are not however in widespread use in the machine learning community as most work has been done to date within the statistics community.

Delta Test (DT), introduced for time series in 1994 [38], is a NRVE method, i.e., it estimates the lowest mean square error (MSE) that can be achieved by a model without overfitting the training set [19]. Given N multiple input-single output pairs, $(\bar{x}_i, y_i) \in \mathbb{R}^M \times \mathbb{R}$, the theory behind the DT method considers that the mapping between $\bar{x}_i$ and y_i is given by the following expression:

$$y_i = f(\bar{x}_i) + r_i,$$

where f is an unknown perfect fitting model and r_i is the noise. DT is based on hypothesis coming from the continuity of the regression function. When two inputs x and x' are close, the continuity of the regression function implies that the corresponding outputs, $f(x)$ and $f(x')$ will be close enough. When this implication does not hold, it is due to the influence of the noise.

Let us denote the first nearest neighbor of the point $\bar{x}_i$ in the set $\{\bar{x}_1, \ldots, \bar{x}_N\}$ by $\bar{x}_{NN(i)}$. Then the DT, δ, is defined as follows:

$$\delta = \frac{1}{2N} \sum_{i=1}^{N} \left| y_{NN(i)} - y_i \right|^2,$$

where $y_{NN(i)}$ is the output corresponding to $\bar{x}_{NN(i)}$. For a proof of convergence, refer to [24, 23]. DT is an unbiased and asymptotically perfect estimator with a relatively fast convergence [23] and is useful for evaluating nonlinear correlations between two random

variables, namely, input-output pairs. DT can be seen as part of a more general NRVE framework known as the Gamma Test [19]. Despite the simplicity of DT, it has been shown to be a robust method in real world applications [24]. This method will be used in the next sections for a priori input selection, an application initially proposed formally in [11].

3. Fuzzy Inference Systems as Autoregressors

The methodology proposed in this paper is intended to apply to crisp time series, i.e. those time series consisting of crisp values, as opposed to other kinds of values, such as interval and fuzzy values. That is, we propose here an automatic methodology to perform autoregressive prediction of crisp time series by means of fuzzy inference systems using nonparametric residual variance estimation [33]. We will call fuzzy autoregressors those autoregressors implemented as fuzzy inference systems. This is not to be confused with what is usually called fuzzy regression in the literature [7].

Consider a discrete time series as a vector, $\bar{y} = y_1, y_2, \ldots, y_{t-1}, y_t$, that represents an ordered set of values, where t is the number of values in the series. The problem of predicting one future value, y_{t+1}, using an autoregressive model (autoregressor) with no exogenous inputs can be stated as follows:

$$\hat{y}_{t+1} = f_r(y_t, y_{t-1}, \ldots, y_{t-M+1}),$$

where $\hat{y}_{t+1}$ is the prediction of model f_r and M is the number of inputs to the regressor, i.e., the regressor size.

Predicting the first unknown value requires building a model, f_r, that maps regressor inputs (known values) into regressor outputs (predictions). When a prediction horizon higher than 1 is considered, the unknown values can be predicted following two main strategies: recursive and direct prediction.

The recursive strategy applies the same model recursively, using predictions as known data to predict the next unknown values. For instance, the third unknown value is predicted as follows:

$$\hat{y}_{t+3} = f_r(\hat{y}_{t+2}, \hat{y}_{t+1}, y_t, y_{t-1}, \ldots, y_{t-M+3}).$$

Recursive prediction is the most simple and intuitive strategy and does not require any additional modeling after an autoregressor for 1 step ahead prediction is built. However, recursive prediction suffers from accumulation of errors. The longer the prediction term is, the more predictions are used as inputs. In particular, for prediction horizons greater than the regressor size, all inputs to the model are predictions.

Direct prediction requires that the process of building an autoregressor be applied for each unknown future value. Thus, for a maximum prediction horizon H, H direct models are built, one for each prediction horizon h:

$$\hat{y}_{t+h} = f_h(y_t, y_{t-1}, \ldots, y_{t-M+1}), \text{ with } 1 \leq h \leq H.$$

While building a prediction system through direct prediction is more computationally intensive (as many times as values are to be predicted) it is also straightforward to parallelize. As opposed to recursive prediction, direct prediction does not suffer from accumulation of prediction errors.

In this paper, we follow the direct prediction strategy. In order to build each autoregressor, a fuzzy inference system is defined as a mapping between a vector of crisp inputs and a crisp output. Let us rename the inputs $y_t, y_{t-1}, \ldots, y_{t-M+1}$ as $y_1, \ldots, y_M$ for simplicity. This way, assuming all (M) inputs are used, the fuzzy autoregressor for prediction horizon h can be expressed as a set of N_h fuzzy rules of the following form:

$$R_i^h : \text{IF } y_1 \text{ is } L_1^{i,h} \text{ AND } y_2 \text{ is } L_2^{i,h} \text{ AND } \ldots \text{ AND } y_M \text{ is } L_M^{i,h} \text{ THEN } \hat{y}_{t+h} \leftarrow \mu_{R_i^h},$$

where $i = 1, \ldots, N_h$, and the fuzzy sets $L_j^{i,h} \in \{L_{j,k}^h\}, k = 1 \ldots, n_j^h, j = 1, \ldots, M$, with n_j^h being the number of linguistic labels defined for the jth input variable. $L_j^{i,h}$ are the fuzzy sets representing the linguistic terms used for the jth input in the ith rule of the fuzzy model for prediction horizon h. $\mu_{R_i^h}$ are the consequents of the rules and can take different forms. For example, in a system with two inputs, if $L_1^{i,h}$ is renamed LOW_1 and $L_2^{i,h}$ is renamed $HIGH_2$, the ith rule for horizon 1, R_i^1, would have the following form:

$$\text{IF } y_t \text{ was } LOW_1 \text{ AND } y_{t-1} \text{ was } HIGH_2 \text{ THEN } \hat{y}_{t+h} \leftarrow \mu_{R_i^h}.$$

Depending on the fuzzy operators, inference model and type of membership functions (MFs) employed, the mapping between inputs and outputs can have different formulations. In principle, the methodology proposed in this paper can be applied for any combination of types of MFs, operators and inference model, but the selection can have a significant impact on practical results.

As a concrete implementation for this paper, we use the minimum as T-norm for conjunction operations, Gaussian MFs for inputs, singleton outputs, and product inference of rules. Defuzzification is performed using the fuzzy mean method, i.e., zero-order Takagi-Sugeno systems [46, 37, 27] are defined. Thus, the result of the inference process is a weighted average of the singleton consequents. This inference scheme was chosen in order to keep systems as simple and interpretable as possible. In particular, the use of singleton outputs simplifies both the interpretation of rules and its local optimization.

Therefore, in this particular case a fuzzy autoregressor for prediction horizon h can be formulated as follows:

$$\mathcal{F}_h(\bar{y}) = \frac{\sum_{i=1}^{N_h} \left(\mu_{R_i^h} \cdot \min_{1 \leq j \leq M} \mu_{L_j^{i,h}}(y_j) \right)}{\sum_{i=1}^{N_h} \min_{1 \leq j \leq M} \mu_{L_j^{i,h}}(y_j)},$$

where N_h is the number of rules in the rule base for horizon h, $\mu_{R_i^h}$ are singleton output values, and $\mu_{L_j^{i,h}}$ are Gaussian MFs for the inputs. Thus, each fuzzy set defined for the input linguistic terms, $L_{j,k}^h$ (for horizon h, and the kth linguistic term defined for the jth input), is characterized by an MF having the following form:

$$\mu_{L_{j,k}^h} = exp\left[-(y_j - c_{k,j,h})^2 / 2\sigma_{k,j,h}^2\right],\ k = 1 \ldots, n_j^h,\ j = 1, \ldots, M,\ h = 1, \ldots, H, \qquad (1)$$

where $c_{k,j,h}$ and $\sigma_{k,j,h}$ are scalar values and represent the centers and widths of the inputs MFs, respectively.

Fuzzy inference systems of the class being designed here are universal approximators [52, 18]. Thus, for a sufficiently large number of rules and MFs, any input-output mapping should be approximated with arbitrary accuracy.

3.1. Clustering-Based Identification of Fuzzy Inference Systems

Different approaches to the identification of fuzzy inference systems from numeric data have been proposed in the literature [29, 40, 2, 20, 41]. Roughly, two classes of methods can be distinguished: structure-oriented and clustering-based.

In this paper we focus on the clustering-based class of methods and specially on those methods that follow an offline approach, as opposed to evolving methods such as DENFIS [20] or eTS [2], which are more suitable for adaptive, online learning. The following clustering algorithms are compared in this paper for the purposes of identifying fuzzy inference systems: the method based on subtractive clustering (SC) proposed by Chiu [9], the Gath-Geva (GG) [1, 13], Gustafson-Kessel (GK) [16], Hard and Fuzzy C-means (HCM and FCM, respectively) [10] clustering algorithms, and the Improved Clustering for Function Approximation (ICFA) algorithm [15], originally proposed for initializing radial basis function neural networks (RBFNNs) for regression problems. We will pay special attention to the use of the ICFA algorithm for the identification of fuzzy inference systems since it is tailored for modeling input-output patterns as opposed to traditional clustering algorithms.

The first step for clustering-based identification of fuzzy inference systems within the methodology proposed is to apply a clustering algorithm on the input patterns for each prediction horizon h. Once this process finishes, Q^h, $h = 1, \ldots, H$, clusters have been identified. Then, the structure of the corresponding fuzzy inference systems has to be defined. In general, fuzzy rules can be interpreted as joint constraints [42] rather than implication rules. Thus, it is sensible to define a fuzzy rule from each cluster identified. This is the most frequent approach in the literature. This way, the clusters and their corresponding rules are considered as prototypes or models of the whole input pattern sequence.

Let us consider as above a multiple scalar input, single scalar output case where the input patterns to the clustering algorithm consist of M inputs and one output. For every prediction horizon h, let us denote the clusters identified by $\bar{c}_k^h$, $i = 1, \ldots, Q^h$. Let every cluster have the following general form:

$$\bar{c}_k^h : (c_{k,1}^h, \ldots, c_{k,M+1}^h), \text{ with } k = 1, \ldots, Q^h,$$

where the $c^h_{k,M+1}$ correspond to the outputs (y_{t+h}) of fuzzy inference models whereas the $c^h_{k,1}, \ldots, c^h_{k,M}$ correspond to the inputs ($y_1, \ldots, y_M$) to the fuzzy model. For each cluster, a matching rule is generated with the following form:

$$R^h_k: \text{IF } y_1 \text{ is } L^h_{1,k} \text{ AND } y_2 \text{ is } L^h_{2,k} \text{ AND } \ldots \text{ AND } y_M \text{ is } L^h_{M,k} \text{ THEN } \hat{y}_{t+h} \leftarrow c_{k,M+1}, \quad k = 1, \ldots, Q^h,\ Q^h = N^h = n^h_j,$$

where a set of input linguistic terms is created $\{L^h_{j,k}\}, k = 1 \ldots, n^h_j,\ j = 1, \ldots, M,$. These linguistic terms are defined by Gaussian MFs, $\mu_{L^h_{j,k}}$, as in equation 1. The output membership functions are defined as singleton functions centered at the corresponding element of the cluster centers, $c^h_{k,M+1}$. The centers of the input Gaussian MFs for the jth input and kth rule ($c_{k,j,h}$ in equation 1) are set to the jth elements of the corresponding clusters $\bar{c}^h_k$.

When inference systems are identified with clustering methods following this approach, the number of linguistic terms defined for every input variable, $n^h_j, j = 1, \ldots, M$, is equal to the number of clusters identified, Q^h, which in turn is equal to the number of rules identified, N^h. Hence, Q^h different membership functions are generated for each input and output variable, and Q^h rules are generated for horizon h.

The way the widths of the input Gaussian MFs ($\sigma_{k,j,h}$ in equation 1) are set depends on the clustering algorithm used. In the case of the subtractive clustering algorithm, the width are set as a constant value proportional to the range of the input and a neighborhood radius parameter commonly set to 0.1. For the Hard C-means, Fuzzy C-means, Gath-Geva and Gustafson-Kessel algorithms the widths are set as a function of the membership degrees of the input patterns to the clusters.

3.2. ICFA-based Identification

The ICFA clustering algorithm was originally proposed as an improvement to the CFA algorithm [14], for initializing RBF networks, using the k-nearest neighbors algorithm with $k = 1$ for setting the radii. In this paper we will analyze two variants of the ICFA algorithm extended for the initialization of fuzzy inference systems.

The first variant, that we will call ICFA, sets the widths of the input MFs using the k-nearest neighbors algorithm with $k = 1$ as well. However, distances are computed on a

per-input basis and thus the widths for a certain rule derived from a certain cluster can potentially be set from different neighbors.

The second variant of ICFA, ICFA$_f$, is a simple generalization of the original ICFA proposal where all the widths for a certain rule are set to a value inversely proportional to the average weighting parameter w. In the ICFA algorithm, the parameter w measures the difference between the estimated output of a center and the output value corresponding to an input vector. The parameter is used to weight the distances between input patterns and cluster centers. It is defined for every input pattern and cluster as follows:

$$w_{ik} = |F(\bar{y}_i) - o_k|,\ i = 1,\ \ldots,\ N, \tag{2}$$

where N is the number of input patterns, $F(\bar{y}_i)$ are the outputs corresponding to the ith input pattern, and o_k are the estimated outputs for the clusters $\bar{c}_k$. For better readability, the details of the ICFA algorithm are omitted here, refer to appendix A and [15] for a complete definition of how the o_k are computed and a proof of convergence.

Thus, in the ICFA$_f$ variant, the widths are set as follows:

$$w_k = \frac{1}{N} \sum_{i=1}^{i=N} w_{ik}$$

$$\sigma_{k,j} = \frac{1}{w_k},\ j = 1,\ \ldots,\ M$$

The rationale behind this second extended version of ICFA for fuzzy inference systems can be explained as follows. The w parameter is defined as a measure of the dispersion of the actual output values with respect to the output values of clusters. The higher the average w, the more distant output values are on average to the cluster. Thus, clusters with a higher average w should be considered as more local prototypes, i.e., the widths of the MFs should be reduced. In the experimental section it will be shown that this variant improves the accuracy of fuzzy inference systems as compared to the first ICFA extension.

4. Methodology for Time Series Prediction with Fuzzy Inference Systems

The problem of building a regressor can be precisely stated as that of defining a proper number and configuration of MFs and building a fuzzy rule base from a data set comprising

t samples from a time series such that the fuzzy systems $\mathcal{F}_h(\bar{y})$ closely predict the hth next values of the time series. The error metric to be minimized is the mean squared error (MSE).

In this paper, we follow a methodology in which a fuzzy inference system is defined for each prediction horizon throughout the stages shown in figure 1. These stages and how they are specifically implemented in this paper are detailed in the following subsections.

4.1. Variable Selection

In the time series prediction field, the application of variable selection methods has been shown to provide several advantages, such as reducing the model complexity and increasing the accuracy of predictions [43]. A proper choice of input variables can alleviate the curse of dimensionality problem while all the relevant data are still available for building a model. As first step in the methodology, DT estimates are employed so as to perform an a priori selection of the optimal subset of inputs from the initial set of M inputs, given a maximum regressor size M.

Variable selection requires a selection criterion. We use the result of the DT applied to a particular variable selection as as a measure of the goodness of the selection. The input selection that minimizes the DT estimate, and thus the achievable MSE, is chosen for the next stages. This way, the set of selected variables is the one that represents the mapping between inputs and outputs in the most deterministic manner.

NRVE based selection can be classified into the set of model independent methods for input selection. These methods select inputs a priori, i.e., the selection stage is based only on the dataset and does not require to build models. Thus, the computational cost of DT based selection is lower than that of the model dependent cases, in which input selection is addressed as a generalization error minimization problem, using leave-one-out, bootstrap or other resampling techniques [24]. Note that an exhaustive evaluation of the DT for all the possible input selections can become unaffordable for high dimensional problems. This problem is outside the scope of this paper. In these cases, other, more sophisticated, approximate search algorithms can be employed.

4.2. System Identification and Optimization

Usually, defining a fuzzy inference model from data requires two steps: the identification of the structure and the optimization of parameters [18, 40]. The identification and tuning stage of our methodology comprises three substages, see figure 1, that are performed iteratively and in a coordinated manner. The whole process is driven by the third (complexity selection) substage, until a system that satisfies a training error condition derived from the DT estimate is constructed.

4.2.1. Substage 2.1: System identification

In this substage, the structure of the inference system (linguistic labels and rule base) is defined by means of an automatic fuzzy systems identification algorithm for fuzzy inference systems. When clustering algorithms are employed for identification, the process is as described in section 3.1. The set of inputs is fixed after the previous (variable selection) stage. The proper number of clusters is found as follows.

The identification substage, as well as the next (tuning) substage are iteratively performed for increasing numbers of clusters (or increasing model complexity). As described in section 3.1, clustering algorithms usually need the number of clusters to be identified as an a priori parameter. The performance the clustering algorithms in practical forecasting applications is hence highly sensitive to this parameter. Here we overcome this limitation by using the DT estimates as an estimation of the optimal training error of the system. Since the training error is non-monotonic with the number of clusters, an exploration has to be performed. Systems are explored in an increasing order of complexity, from the lowest possible number of clusters up to a maximum specified as complexity boundary.

This iterative identification process for increasing number of clusters stops when a system is built such that the training error is lower than the DT estimate. The selection is made in the third stage by comparing the error after the next (optimization) stage.

4.2.2. Substage 2.2: System Tuning

We consider an additional step for local optimization in the methodology as a substage separated from the identification substage.

A number of supervised learning and optimization methods have been compared for this study, including gradient descent, probabilistic, second order, and conjugate gradient methods. The Resilient Propagation (Rprop) [39, 26] gradient descent method was selected as the most accurate alternative. The tuning process is driven by the normalized MSE.

All the parameters of the membership functions of every input and output are adjusted using the algorithm implementation in the Xfuzzy development environment [35], i.e., self-tuning inference systems are defined.

4.2.3. Substage 2.3: Complexity Selection

The last step in the process of identifying and tuning fuzzy autoregressors consists in selecting the proper complexity of the (estimated) best autoregressor. The iterative identification and tuning stage stops when a system is built such that its training error is equal to or lower than the DT estimate or a threshold based on the DT estimate. Since identification and tuning iterations are performed for an increasing number of clusters, the system with the lowest number of clusters that satisfies the DT based error condition is selected.

It should be noted that the methodology described does not require a validation stage and thus the whole available data set can be used as training data.

5. Experimental Results

In this section, the results obtained for 5 diverse datasets are analyzed. In general, no pre-processing steps are taken and thus the methodology described above is directly applied to the datasets. This way we analyze the capabilities of the methods considered in this paper to directly model real-world data without any expert intervention or use of pre-processing techniques. In order to study the performance of both short- and long-term models, prediction horizons ranging from 1 through 50 are considered, i.e., 50 models are built for every time series. First, we describe the datasets. Then we analyze the accuracy of different modeling alternatives. Finally, we analyze computational requirements and further discuss the results obtained.

5.1. ESTSP 2007 Competition Dataset

We first consider the data set from the competition of the first European Symposium on Time Series Prediction (ESTSP 2007) [12]. This data set, see figure 2, consists of 875 samples of weekly temperatures of the El Niño-Southern Oscillation phenomenon. In this section we analyze the original ESTSP 2007 series split into two subsets: a training set (first 475 samples) and a second set (last 400 samples) that will be used for test. We will call this series ESTSP07. A maximum regressor size $M = 10$ was chosen for this series.

5.2. Sunspot Numbers

The series of sunspot numbers (Sunspots henceforth) is a periodic measurement of the sunspot activity as a function of the number of spots visible on the face of the Sun and the number of groups into which they cluster. We analyzed the series of monthly averaged sunspot numbers covering from January 1749 through December 2007, as provided by the National Geographical Data Center from the US National Oceanic and Atmospheric Administration[3]. The series was split into a set of 2108 values for training and a set of 1000 values for testing, as shown in figure 3. Given the yearly periodicity of the series, a maximum regressor size of 12 was defined.

5.3. Poland Electricity Benchmark

This time series (PolElec henceforward) represents the normalized average daily electricity demand in Poland in the 1990´s. The benchmark consists of a training set of 1400 samples, shown in figure 4(a), and a test set of 201 samples, shown in figure 4(b), available from [47]. It has been shown that the essential dynamics of this time series is nearly linear [22]. Besides the yearly periodicity, a clear weekly periodicity can be seen on smaller time scales (see figure 4(b)). In this case, a maximum regressor size of 14 was chosen to better capture the weekly periodicity [22].

[3]The series used here can be obtained from http://www.ngdc.noaa.gov/stp/SOLAR/ftpsunspotnumber.html. The International Sunspot Number is produced by the Solar Influence Data Analysis Center (SIDC) at the Royal Observatory of Belgium [48].

5.4. Dataset 1 of the ESTSP 2008 Competition

Here we consider the first dataset from the ESTSP 2008 competition (ESTSP08-1) [12, 21]. This series is part of a multidimensional time series of monthly averages of different chemical descriptors of a certain area of the Baltic Sea. The series is made of 354 samples and spans for 29 years and a half. In this case, the first two thirds of the series were selected as training set while the final third was selected as test set, as shown in figure 5. Thus, only 236 samples are available for training. In addition, a clear increasing trend in the training set and change in dynamics can be observed. This fact limits to a great extent the number of training samples useful for predicting the test set. A maximum regressor size of 12 was selected in this case.

5.5. Dataset 2 of the ESTSP´08 Competition

The second dataset from the ESTSP 2008 competition (ESTSP08-2) [12, 21] is a univariate time series consisting of 1300 samples that describe the daily average amount of traffic in a data network, see figure 6. The first two thirds of the series were selected for training whereas the last third was selected as test set. For this series the maximum regressor size was fixed to 14.

5.6. Results

Let us illustrate the application of the methodology followed in this paper through a few examples. In the first stage, given a maximum regressor size, a subset of variables is selected. Figure 7 shows the total amount of variables selected for two of the series studied. In every case, the selection stage is performed for each prediction horizon (1 through 50). It can be seen that the DT based variable selection leads to a significant decrease of the complexity of the fuzzy inference systems in terms of number of inputs.

As second stage, once input variables have been selected, an iterative identification and tuning process is carried out in three substages, as shown in figure 1. In the first substage (identification) a clustering algorithm is applied to the training set in order to identify inference systems. These systems are then tuned in the second substage through supervised

learning using the Rprop algorithm. The process is repeated for increasing numbers of clusters (or fuzzy rules), starting from 1.

Within this iterative process, in the third substage (complexity selection) the DT estimate is used to check whether the best possible approximation has been achieved, i.e., the right compromise between model complexity and training error has been found. Figure 8 shows the normalized DT (NDT) estimates as well as the training and test errors for two example series: ESTSP07 and PolElec. Again, horizons 1 through 50 are considered. Note training and test errors are normalized against the variance of the training and test datasets, respectively. In figure 9, two examples of the prediction results are shown for the ESTSP07 and ESTSP08-2 series.

A number of supervised learning algorithms were tested for the tuning substage. For this study, we used the implementations in the Xfuzzy environment [35]. Among them, we distinguish four classes of methods: gradient descent [26], conjugate gradient, second order or quasi-Newton [3], and algorithms with no derivatives. All the results given in this section have been obtained using a method belonging to the conjugate gradient class: Resilient Propagation (Rprop) [39, 26], which provided the best results on average The following parameters were employed for the Rprop method: 0.1 as initial update, 1.5 as increase factor, and 0.5 as decrease factor. It should be noted though that similar results can be achieved with different alternatives. In particular, the Levenberg-Marquardt (L-M) [3] method yields errors approximately 1% higher on average, while the Scaled Conjugated Gradient (SCG) method [30], yields errors approximately 2% higher on average.

In table 1, the accuracy of the different clustering alternatives considered is compared. The normalized mean squared error (NMSE) is used as error metric. The default parameters of the implementation in version 3.3 of the Xfuzzy design environment [51, 36] were used. In particular, the ICFA algorithm was applied with fuzziness index $h = 2.0$ and 0.01 as threshold to decide whether the centers have moved significantly. The table lists results for the two variants of the ICFA algorithm that were described in section 3.1. Table 2 shows the corresponding values of standard deviation of the square errors. In general, lower average square error values correspond to lower standard deviation values. Comparatively

the performance of the clustering methods analyzed in terms of standard deviation of the square error closely resembles the performance in terms of average error

From the table it is clear that the SC method is the least accurate while ICFA$_f$ is the most accurate. For the purposes of comparison with other results in the literature we note the similarity between the SC method and the subtractive clustering based ANFIS. Note however that the tuning algorithm used in this paper is Rprop, which in our experience outperforms the hybridization of gradient descent and least squares optimization proposed originally in the ANFIS method [18].

The ICFA algorithm is the most accurate by a slight but consistent difference. There is however one exception. In the case of the ESTSP08-1 dataset the FCM algorithm is the most accurate, followed by HCM. The ESTSP08-2 series was chosen as a representative case of series for which only a reduced number of useful training data is available. In these cases, both the HCM and FCM methods are more robust.

It can be observed that a proper initialization of the widths of the input membership functions is a key factor to obtaining a better performance with the ICFA algorithm.

5.7. Comparison with Other Modeling Approaches

Let us now analyze the accuracy of the clustering-based fuzzy models described as compared to alternative modeling approaches. As before, no pre-processing steps are performed. Table 3 shows the test errors for four modeling techniques: Multilayer perceptron (MLP), least-squares support vector machines LS-SVM, the extreme learning machine (ELM) and the optimally-pruned ELM (OP-ELM). These modeling techniques were applied using the same input selection scheme as before.

The MLP [5] is a well known, widely used modeling method with universal approximation capability and good generalization potential. LS-SVM [45] is a well established method in the field of time series prediction, that has been shown to be highly accurate. The extreme learning machine (ELM) [17] is a simple yet effective learning algorithm for training single-hidden-layer feed-forward artificial neural networks with random hidden nodes. The optimal-pruned extreme learning machine (OP-ELM) [44] is a methodology based on the ELM, that

has been shown to produce models competitive against well-known, accurate techniques, such as LS-SVM and the MLP, while being significantly faster.

An overall comparison of the results listed in tables 3 and 1 shows that fuzzy models outperform the three alternative methods. As an exception, in the case of the Sunspots series fuzzy models are less accurate than the other alternatives analyzed.

For this study, standard two-layer MLP models were built for the same training subset, following a 10-fold cross-validation strategy in order to perform model selection for a maximum of 40 hidden units. The implementation in the standard Neural Networks Matlab Toolbox was used. As for LS-SVM models, the following options were chosen: Radial Basis Function (RBF) kernels, grid search as optimization routine and cross-validation as cost function, see [45] for a detailed specification of these and other options. The optimized C version of the LS-SVMlab1.5 Matlab/C toolbox [25] was employed. Regarding ELM, the implementation by Zhu and Huang available from `http://www3.ntu.edu.sg/home/egbhuang` was used with sigmoid functions and standard options. OP-ELM models were built using the OP-ELM toolbox [28] with the following configuration options: a combination of linear, Gaussian and sigmoid kernels, using a maximum of 100 neurons. This way, the results presented here can be compared with those of other studies that also analyzed MLP, LS-SVM and OP-ELM models using the same implementations [28]. In all cases, data are normalized before modeling.

In the ELM_{best} method shown in the table, ELM models are built for 100 different numbers of neurons between 1 and 100. The model that yields the lowest test error is selected. Thus, ELM_{best} can be regarded as a reference of what could be achieved with standard ELM models. It should be observed that, in practice, results from ELM models can be expected to be worse as a consequence of the limitations of the particular model selection scheme applied.

5.8. Discussion

The methodology applied in this paper leverages on a robust technique for NRVE and input selection as well as the optimization of models through supervised learning. This

allows for the identification of compact yet highly accurate inference systems at a reasonable computational cost.

A tool, xftsp [34], has been developed that implements the methodology proposed in this paper and provides support for the identification and tuning algorithms included in the Xfuzzy environment [51]. Xfuzzy is conceived as a development environment for fuzzy inference systems that covers the whole design process, from initial specification using a high level language to implementation as software or hardware.

The Xfuzzy environment covers the following stages in the design flow of fuzzy inference systems: description, tuning, verification and synthesis. A number of standalone tools implement these stages. The link among all these tools is the use of a common specification language, XFL3, and a common software component for the definition of fuzzy inference systems using XFL3.

Within the description stage, Xfuzzy includes graphical tools for defining fuzzy systems in a visual manner. Tools for simulation, monitoring and graphical representation of the system behavior are provided for the verification stage. The tuning stage encompasses tools for identification, supervised learning and simplification tasks. Finally, the synthesis stage includes tools for generating high-level language descriptions for software and hardware implementations. Software implementations can be automatically generated for languages such as C and Java, whereas hardware implementations are generated in the form of synthesizable VHDL descriptions. Each tool can be executed whether as an independent program or as part of a global environment. Interactive usage is under a graphical user interface that ties together the whole set of tools.

xftsp can be run whether as a standalone console tool or within the Xfuzzy environment. The design of the xftsp tool allows for the use of the wide set of tools available in the Xfuzzy environment for complementary tasks such as visualization, simplification and code generation. Refer to [34] for further details on the design of xftsp and how it fits in the overall architecture of Xfuzzy. For a complete description of the Xfuzzy environment refer to [35, 51, 36].

This Java based implementation of the methodology presented here is consistently be-

tween 1 and 2 orders of magnitude faster than the optimized implementation of LS-SVM used for this study. Table 4 shows the time required to build models using the aforementioned four modeling methods for a subset of the time series considered in this paper. Roughly, OP-ELM models are faster than fuzzy models by an order of magnitude, while fuzzy models are faster than MLP models by an order of magnitude, and faster than LS-SVM models by two orders of magnitude.

In practice systems built using the proposed method have a very low number of rules while attaining a high accuracy for a number of time series benchmarks [33]. In the case of clustering-based identification methods analyzed here, the number of clusters required to model time series rarely exceeds 10. Table 5 shows the average number of clusters identified for the different clustering algorithms analyzed. Despite the methodology yields highly accurate models, the compactness of these is remarkable as well.

The proposed method has a fundamental advantage over usual prediction techniques. Each fuzzy rule can be interpreted as a linguistic map between regions of interest of the input and output domain. The method yields compact rulebases made of rules of simple structure. This way, fuzzy inference models can be identified in a fully automatic manner yet analyzed off-line by experts in order to extract linguistic knowledge. Linguistic interpretation opens new possibilities such as the use of CAD tools that implement interactive techniques to ease the visualization and analysis of fuzzy inference systems [4].

Let us consider one particular example for the purposes of illustrating the way the proper number of clusters is automatically selected and how the membership functions are adjusted. For the 1 step ahead model of the ESTSP07 series three input variables are selected, y_t, y_{t-2}, and y_{t-7}, in order to model y_{t+1}.

Figure 10 shows the training and test errors for different numbers of clusters. Besides ICFA$_f$ the next three best options, GG, FCM and HCM are considered. It should be noted that, as can be seen in the example as well as in table 5, in general better accuracy comes at the cost of higher system complexity in terms of clusters and rules. For ICFA$_f$ the model with 9 clusters is selected. For GG the model automatically selected has 6 clusters. In both cases the selection is perfect. Both the FCM and HCM models have 5 clusters. In the case

of the FCM clustering method the model with 5 clusters is selected instead of the model with 6 clusters, which is the most accurate. In the case of the HCM method, the model with 7 clusters would have been slightly more accurate than the model selected. Finally, figure 11 shows as an example the shape of the membership functions for the model built using the ICFA$_f$ clustering algorithm.

6. Conclusion

We have described an automatic method for long-term time series prediction by means of fuzzy inference systems. Regressive inference systems are identified by clustering methods using nonparametric residual variance estimates together with a local optimization algorithm in order to set the proper number of rules and configuration of membership functions. The following conclusions can be drawn from the experiments performed:

- We have compared the performance of different clustering alternatives for initializing the centers and widths of the membership functions of fuzzy inference systems for time series prediction.
- This comparison has been made on a diverse set of time series benchmarks. in the context of a methodology that improves interpretability and accuracy of models by performing an initial input selection stage.
  - It has been shown that the proper number of clusters can be defined in a robust manner by using a nonparametric residual variance estimator.
  - This leads to the identification of remarkably compact rulebases.
- A simple scheme for initializing the widths of the input membership functions has been proposed for the ICFA algorithm. This scheme has been shown to yield the most accurate results among a diverse set of clustering algorithms. In addition, fuzzy inference systems initialized using this variant of the ICFA algorithm have been shown to provide overall better results than other modeling techniques such as MLP, LS-SVM and OP-ELM.

A. Convergence of the ICFA Algorithm

The ICFA clustering algorithm was originally proposed by Guillén et al. [15] as an improvement to the CFA algorithm [14]. This appendix explains how the centers are computed in the ICFA algorithm and how convergence is guaranteed, reproducing the equations developed in [15] with slight notational changes.

For the ICFA$_f$ variant introduced in this paper, the average weighting parameter is used, as explained in section 3.2. This parameter is based on the weighting parameter w_{ik} defined in the ICFA algorithm (see equation 2). For this definition, normalized functions are assumed and the way in which the weighting parameter is used guarantees the convergence of the algorithm, as explained below.

The ICFA algorithm is targeted at functional approximation problems and pursues the goal of making the centers closer to the areas of higher variability of the target function. To this end, the Euclidean distance is adjusted by the weighting parameter as follows:

$$D_{kiW} = ||y_i - c_k||w_{ik}^2.$$

Let us call N the number of samples or input-output pairs, and Q the number of clusters identified, denoted as Q_h for different prediction horizons h in section 3.1. The distortion function to be minimized by the algorithm is defined as

$$J_h(U, C, W) = \sum_{i=1}^{N} \sum_{k=1}^{Q} u_{ki}^h D_{kiW},$$

where h is the fuzziness index (for which, as in this paper, a common value is 2), U is the matrix of membership values, u_{ki}, C is the matrix of cluster centers, $\bar{c}_k$, and W is the matrix of weighting parameters, w_{ik}. The above distortion function is subject to the following constraints:

$$\sum_{k=1}^{Q} u_{ki} = 1, \quad \forall i = 1, \ldots, N,$$

$$0 < \sum_{i=1}^{N} u_{ki} < N, \quad \forall k = 1, \ldots, Q.$$

In the approach proposed in [15], the distortion function is minimized as in the original CFA proposal, using the Picard iteration algorithm. On each iteration the three following parameters are computed in this order: the membership degrees, the positions of the centers and the expected outputs for the centers. These parameters are computed as follows:

$$u_{ki} = \left(\sum_{j=1}^{Q} \left(\frac{D_{kiW}}{D_{jiW}} \right)^{1/(h-1)} \right)^{-1},$$

$$\bar{c}_k = \frac{\sum_{i=1}^{N} u_{ki}^h \bar{y}_i w_{ik}^2}{\sum_{i=1}^{N} u_{ki}^h w_{ik}^2},$$

$$\bar{o}_k = \frac{\sum_{i=1}^{N} u_{ki}^h F(\bar{y}_i) d_{ik}^2}{\sum_{i=1}^{N} u_{ki}^h w_{ik}^2},$$

where d_{ki} is the Euclidean distance between the input data $\bar{y}_i$ and centers $\bar{c}_k$, as defined above. These equations are derived in [15] by setting the derivative of the distortion function with respect to the parameters to be optimized, i.e., U, C and O, equal to zero. The distortion function is extended by Lagrange multipliers to incorporate the two constraints above. This way, convergence is guaranteed, as opposed to the original CFA algorithm. Furthermore, ICFA requires only one update step per iteration of the algorithm. In addition, a migration stage is performed after the centers have been identified, refer to [15] for a complete description of the ICFA algorithm.

References

[1] J. Abonyi, R. Babuska, F. Szeifert, Modified Gath-Geva Fuzzy Clustering for Identification of Takagi-Sugeno Fuzzy Models, IEEE Transactions on Systems, Man and Cybernetics, Part B 32 (5) (2002) 612–621.

[2] P. P. Angelov, D. P. Filev, An approach to online identification of takagi-sugeno fuzzy models, IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics 34 (1) (2004) 484–498.

[3] R. Battiti, First and Second Order Methods for Learning: Between Steepest Descent and Newton's Method, Neural Computation 4 (2) (1992) 141–166.

[4] I. Baturone, F. J. Moreno-Velo, A. Gersnoviez, A CAD Approach to Simplify Fuzzy System Descriptions, in: 15th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'06), 2006.

[5] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1996, ISBN: 0198538642.

[6] G. Box, G. M. Jenkins, G. Reinsel, Time Series Analysis: Forecasting & Control, Prentice Hall; 3rd edition, 1994, ISBN: 0130607746.

[7] Y.-H. O. Chang, B. M. Ayyub, Fuzzy regression methods - a comparative assessment, Fuzzy Sets and Systems 119 (2) (2001) 187–203.

[8] C. Chatfield, The Analysis of Time Series. An Introduction, CRC Press, 2003, Sixth edition, ISBN: 1-58488-317-0.

[9] S. L. Chiu, A Cluster Estimation Method with Extension to Fuzzy Model Identification, in: IEEE Conference on Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence, Orlando, FL, USA, 1994.

[10] J. V. de Oliveira, W. Pedrycz (eds.), Advances in Fuzzy Clustering and its Applications, John Wiley & Sons, Ltd., West Sussex, England, 2007, ISBN: 978-0-470-02760-8.

[11] E. Eirola, E. Liitiäinen, A. Lendasse, F. Corona, M. Verleysen, Using the delta test for variable selection, in: M. Verleysen (ed.), ESANN 2008, European Symposium on Artificial Neural Networks, Bruges (Belgium), 2008.

[12] ESTSP: European Symposium on Time Series Prediction (Jun. 2009).
URL `http://www.estsp.org`

[13] I. Gath, A. B. Geva, Unsupervised Optimal Fuzzy Clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (7) (1989) 773–780.

[14] J. González, I. Rojas, H. Pomares, J. Ortega, A. Prieto, A new clustering technique for function aproximation, IEEE Transactions on Neural Networks 13 (1) (2002) 132–142.

[15] A. Guillén, J. González, I. Rojas, H. Pomares, L. J. Herrera, O. Valenzuela, A. Prieto, Using fuzzy logic to improve a clustering technique for function approximation, Neurocomputing 70 (16–18) (2007) 2853–2860.

[16] E. E. Gustafson, W. C. Kessel, Fuzzy Clustering with a Fuzzy Covariance Matrix, in: 17th Symposium on Adaptive Processes, 1978 IEEE Conference on Decision and Control , San Diego, CA, 1978.

[17] G.-B. Huang, Q. Y. Zhu, C. K. Siew, Extreme learning machine: Theory and applications, Neurocomputing 70 (1–3) (2006) 489–501.

[18] J.-S. R. Jang, C.-T. Sun, E. Mizutani, Neuro-Fuzzy and Soft Computing A Computational Approach to Learning and Machine Intelligence, Prentice Hall, Upper Saddle River, New Jersey, 1997, ISBN 0-13-261066-3.

[19] A. J. Jones, New Tools in Non-linear Modelling and Prediction, Computational Management Science 2 (1) (2004) 109–149.

[20] N. K. Kasabov, Q. Song, DENFIS: Dynamic Evolving Neural-Fuzzy Inference System and Its Application for Time-Series Prediction, IEEE Transactions on Fuzzy Systems 10 (2) (2002) 144–154.

[21] A. Lendasse (ed.), 2nd European Symposium on Time Series Prediction (ESTSP´08), 2008 (Sep. 2008).

[22] A. Lendasse, J. Lee, V. Wertz, M. Verleyssen, Forecasting Electricity Consumption using Nonlinear Projection and Self-Organizing Maps, Neurocomputing 48 (1) (2002) 299–311.

[23] E. Liitiäinen, A. Lendasse, F. Corona, Bounds on the mean power-weighted nearest neighbour distance, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 464 (2097) (2008) 2293–2301.

[24] E. Liitiäinen, M. Verleysen, F. Corona, A. Lendasse, Residual variance estimation in machine learning, Neurocomputing 72 (16–18) (2009) 3692–3703.

[25] Least Squares - Support Vector Machines Matlab/C Toolbox (Apr. 2008).
URL `http://www.esat.kuleuven.ac.be/sista/lssvmlab`

[26] G. D. Magoulas, M. N. Vrahatis, G. S. Androulakis, Improving the Convergence of the Backpropagation Algorithm Using Learning Rate Adaptation Methods, Neural Computation (1999) 1769–1796.

[27] J. M. Mendel, Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions, Prentice Hall PTR, 2001, ISBN: 0130409693.

[28] Y. Miche, A. Sorjamaa, A. Lendasse, OP-ELM: Theory, Experiments and a Toolbox, in: 18th International Conference on Artificial Neural Networks (ICANN), vol. 5163 of Lecture Notes in Computer Science, Prague, Czech Republic, 2008.

[29] S. Mitra, Y. Hayashi, Neuro-fuzzy rule generation: survey in soft computing framework, IEEE Transactions on Neural Networks 11 (3) (2000) 748–768.

[30] M. F. Møller, A scaled conjugate gradient algorithm for fast supervised learning, Neural Networks 6 (4) (1993) 525–533.

[31] F. Montesino-Pouzols, A. Barriga, Regressive fuzzy inference models with clustering identification: Application to the ESTSP08 competition, in: 2nd European Symposium on Time Series Prediction, Porvoo, Finland, 2008.

[32] F. Montesino-Pouzols, A. Lendasse, A. Barriga, Autoregressive time series prediction by means of fuzzy inference systems using nonparametric residual variance estimation, Fuzzy Sets and Systems, in press.

[33] F. Montesino-Pouzols, A. Lendasse, A. Barriga, Fuzzy Inference Based Autoregressors for Time Series Prediction Using Nonparametric Residual Variance Estimation, in: 17th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'08), IEEE World Congress on Computational Intelligence, Hong Kong, China, 2008.

[34] F. Montesino-Pouzols, A. Lendasse, A. Barriga, xftsp: a Tool for Time Series Prediction by Means of Fuzzy Inference Systems, in: 4th IEEE International Conference on Intelligent Systems (IS´08), Varna, Bulgaria, 2008.

[35] F. J. Moreno-Velo, I. Baturone, A. Barriga, S. Sánchez-Solano, Automatic Tuning of Complex Fuzzy

Systems with Xfuzzy, Fuzzy Sets and Systems 158 (18) (2007) 2026–2038.

[36] F. J. Moreno-Velo, I. Baturone, S. Sánchez-Solano, A. Barriga, Rapid Design of Fuzzy Systems With Xfuzzy, in: 12th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'03), St. Louis, MO, USA, 2003.

[37] H. T. Nguyen, N. R. Prasad (eds.), Fuzzy Modeling and Control: Selected Works of M. Sugeno, CRC Press, Inc, 1999.

[38] H. Pi, C. Peterson, Finding the embedding dimension and variable dependencies in time series, Neural Computation 6 (3) (1994) 509–520.

[39] M. Riedmiller, Advanced supervised learning in multi-layer perceptrons - from backpropagation to adaptive learning algorithms, Computer Standards and Interfaces 16 (3) (1994) 265–278.

[40] I. Rojas, H. Pomares, J. Ortega, A. Prieto, Self-Organized Fuzzy System Generation from Training Examples, IEEE Transactions on Fuzzy Systems 8 (1) (2000) 23–36.

[41] H.-J. Rong, N. Sundararajan, G.-B. Huang, P. Saratchandran, Sequential Adaptive Fuzzy Inference System (SAFIS) for nonlinear system identification and prediction, Fuzzy Sets and Systems 157 (9) (2006) 1260–1275.

[42] L. Rutkowski, Flexible Neuro-Fuzzy Systems. Structures, Learning and Performance Evaluation, Kluwer Academic Publishers, Boston, MA, USA, 2004, ISBN: 1-4020-8042-5.

[43] A. Sorjamaa, J. Hao, N. Reyhani, Y. Ji, A. Lendasse, Methodology for Long-Term Prediction of Time Series, Neurocomputing 70 (16–18) (2007) 2861–2869.

[44] A. Sorjamaa, Y. Miche, R. Weiss, A. Lendasse, Long-Term Prediction of Time Series using NNE-based Projection and OP-ELM, in: 2008 International Joint Conference on Neural Networks (IJCNN 2008), IEEE World Congress on Computational Intelligence, Hong Kong, China, 2008.

[45] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, Least Squares Support Vector Machines, World Scientific, Singapore, 2002, ISBN: 981-238-151-1.

[46] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Transactions on Systems, Man, and Cybernetics 15 (1) (1985) 116–132.

[47] Time Series Datasets. Time Series Prediction and Chemoinformatics Group (Nov. 2009).
URL `http://www.cis.hut.fi/projects/tsp`

[48] R. A. M. Van der Linden, the SIDC Team, Online Catalogue of the Sunspot Index, RWC Belgium, World Data Center for the Sunspot Index, Royal Observatory of Belgium, years 1748-2007, http://sidc.oma.be/html/sunspot.html (Jan. 2008).

[49] L. X. Wang, The WM Method Completed: A Flexible System Approach to Data Mining, IEEE Transactions on Fuzzy Systems 11 (6) (2003) 768–782.

[50] A. Weigend, N. Gershenfeld, Times Series Prediction: Forecasting the Future and Understanding the

Past, Addison-Wesley Publishing Company, 1994, ISBN: 0201626020.

[51] The Xfuzzy Development Environment: Fuzzy Logic Design Tools (Jun. 2009).
URL https://forja.rediris.es/projects/xfuzzy,http://www.imse-cnm.csic.es/Xfuzzy

[52] H. Ying, Sufficient Conditions on Uniform Approximation of Multivariate Functions by General Takagi-Sugeno Fuzzy Systems with Linear Rule Consequent, IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans 28 (4) (1998) 515–520.

List of Tables

Table 1: Accuracy comparison of different clustering-based identification algorithms for fuzzy inference systems. The table shows the average normalized square test errors (NMSE for the test series). Errors are averaged for prediction horizons 1 through 50. In all cases the Rprop optimization algorithm was used. The lowest errors are highlighted in bold face for each series.

Series	SC	HCM	FCM	GK	GG	ICFA	ICFA$_f$
ESTSP07	0.3243	0.2983	0.2911	0.2974	0.2873	0.2714	**0.2639**
PolElec	0.2645	0.2428	0.2481	0.2463	0.2550	0.2424	**0.2418**
Sunspots	1.0541	0.9550	0.9552	0.9486	0.9570	0.9334	**0.9214**
ESTSP08-1	0.7677	0.5887	**0.5567**	0.6746	0.6872	0.6607	0.6221
ESTSP08-2	1.0526	0.9125	0.8932	0.9257	0.9197	0.8777	**0.8564**

Table 2: Standard deviation of normalized square test errors. Comparison of the variability in the accuracy of different clustering-based identification algorithms for fuzzy inference systems. Standard deviations are averaged for prediction horizons 1 through 50. The standard deviation values shown correspond to the average values shown in table 1. The lowest deviations are highlighted in bold face for each series.

Series	SC	HCM	FCM	GK	GG	ICFA	ICFA$_f$
ESTSP07	0.3579	0.3185	0.3236	0.3345	0.3145	0.3065	**0.2685**
PolElec	0.3997	0.3828	0.3830	0.3802	0.3883	0.3966	**0.3764**
Sunspots	0.9600	0.8521	0.8496	0.9037	0.8848	0.8392	**0.7446**
ESTSP08-1	1.3825	1.0531	**1.0323**	1.0988	1.2238	1.3668	1.1524
ESTSP08-2	1.6312	1.4888	1.4715	1.4724	1.4617	1.5608	**1.4388**

Table 3: Accuracy of alternative modeling techniques. Average test errors for prediction horizons 1 through 50 are shown. The lowest errors for each series are highlighted in bold face.

Series	MLP	LS-SVM	ELM_{best}	OP-ELM
ESTSP07	**0.424**	0.472	0.478	0.602
PolElec	0.741	0.416	0.429	**0.398**
Sunspots	**0.665**	0.898	0.908	0.833
ESTSP08-1	1.379	2.084	1.948	**1.131**
ESTSP08-2	2.173	**0.927**	2.315	1.111

Table 4: Run time (in seconds) required to build models for prediction horizons 1-50. All tests were run on the same system, with no significant competing load. Processing time was measured using the standard getrusage UNIX system call, accounting for both user and system space tasks.

Series	LS-SVMlab1.5	MLP (NN Toolbox)	Xfuzzy 3.3 (ICFA$_f$)	OP-ELM Toolbox
ESTSP07	$3.27 \cdot 10^5$	$1.20 \cdot 10^4$	$7.41 \cdot 10^2$	$7.84 \cdot 10^1$
PolElec	$8.34 \cdot 10^5$	$2.33 \cdot 10^4$	$1.50 \cdot 10^3$	$2.81 \cdot 10^2$
Sunspots	$2.42 \cdot 10^5$	$3.34 \cdot 10^4$	$1.09 \cdot 10^3$	$5.20 \cdot 10^2$
ESTSP08-1	$6.34 \cdot 10^3$	$9.19 \cdot 10^3$	$2.81 \cdot 10^2$	$3.14 \cdot 10^1$
ESTSP08-2	$2.34 \cdot 10^5$	$7.34 \cdot 10^3$	$4.00 \cdot 10^3$	$1.25 \cdot 10^2$

Table 5: Average and standard deviation of the number of clusters (rules) identified for models built following the proposed methodology. The numbers shown are averaged for horizons 1 through 50.

Series	SC	HCM	FCM	GK	GG	ICFA	ICFA$_f$
ESTSP07	8.7 ± 3.5	3.7 ± 1.0	4.0 ± 1.0	4.3 ± 1.5	3.7 ± 1.0	5.1 ± 1.4	5.9 ± 2.4
PolElec	5.3 ± 4.9	2.4 ± 1.6	2.3 ± 1.7	3.4 ± 5.3	2.6 ± 1.5	3.7 ± 2.9	3.0 ± 4.7
Sunspots	1.5 ± 0.8	1.4 ± 0.5	1.4 ± 0.6	1.8 ± 3.3	1.4 ± 0.6	1.9 ± 1.8	1.8 ± 1.4
ESTSP08-1	3.2 ± 1.3	2.1 ± 0.3	2.0 ± 0.3	2.2 ± 0.6	2.1 ± 0.3	9.8 ± 6.5	2.2 ± 0.8
ESTSP08-2	5.2 ± 3.3	4.6 ± 2.2	5.0 ± 1.9	11 ± 6.0	4.6 ± 1.4	7.2 ± 3.2	7.4 ± 6.1

List of Figures

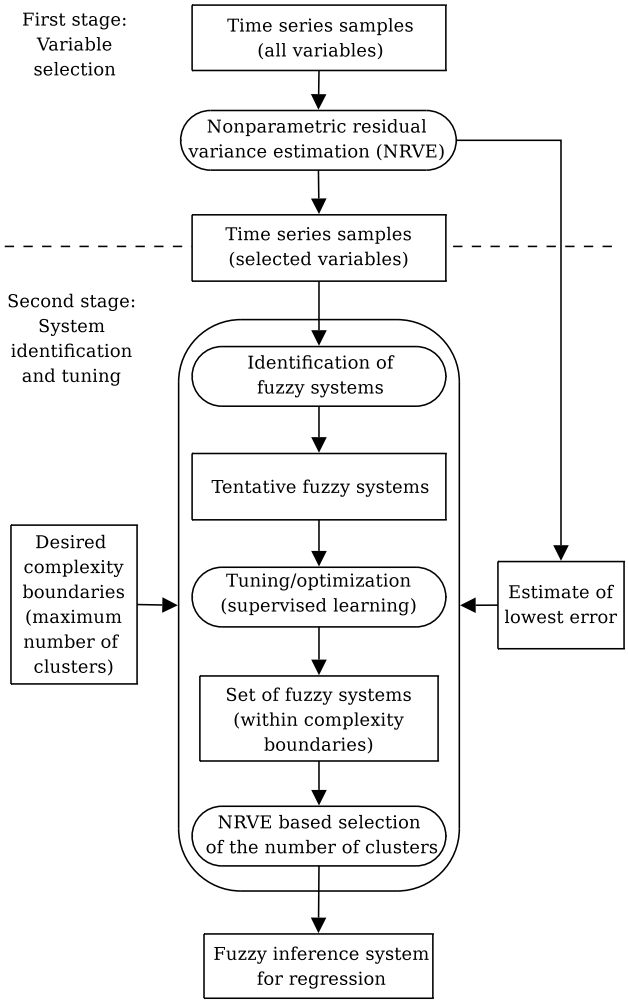
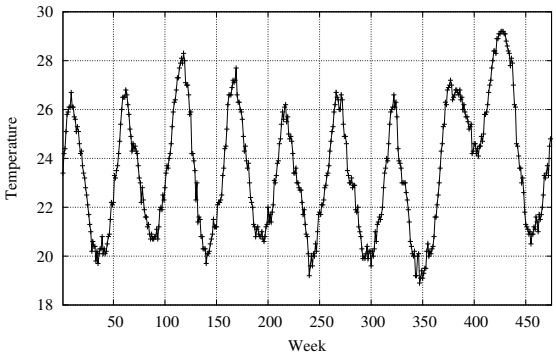
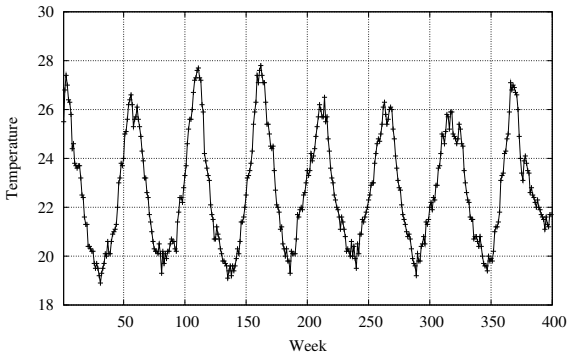


Figure 1: Methodology for time series prediction using clustering-based identification methods.

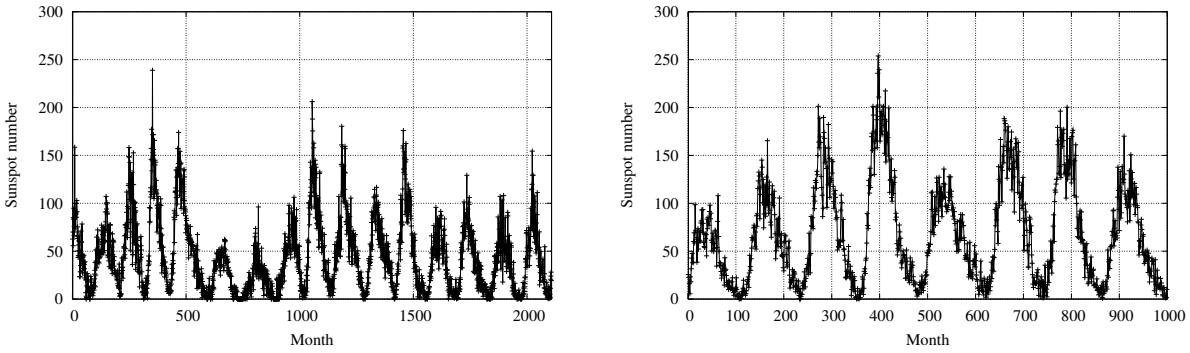


(a) Training series (475 samples)



(b) Test series (400 samples)

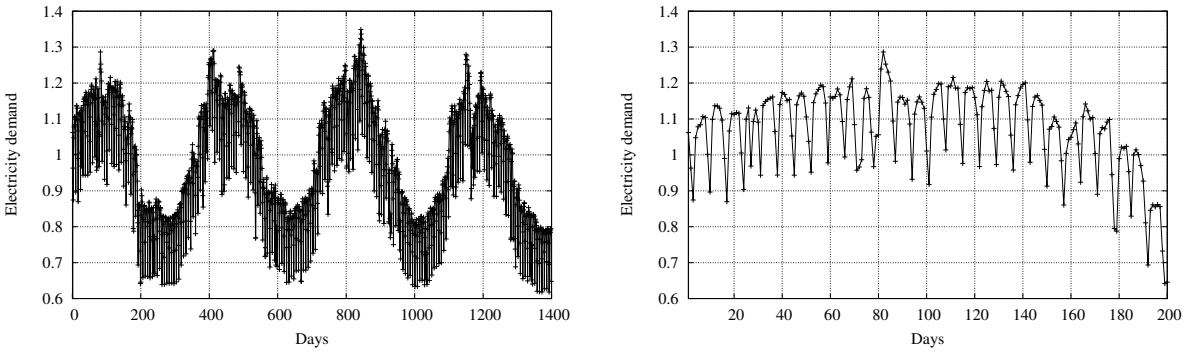
Figure 2: ESTSP07: Dataset from the ESTSP 2007 competition (875 samples).



(a) Training series (2108 samples)

(b) Test series (1000 samples)

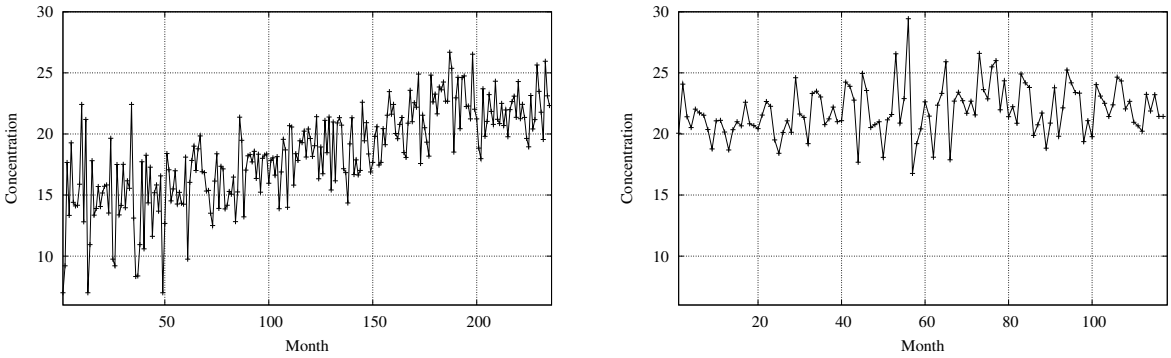
Figure 3: Sunspots: Monthly averages of the Sunspot number series (3198 samples).



(a) Training series (1400 samples)

(b) test series (201 samples)

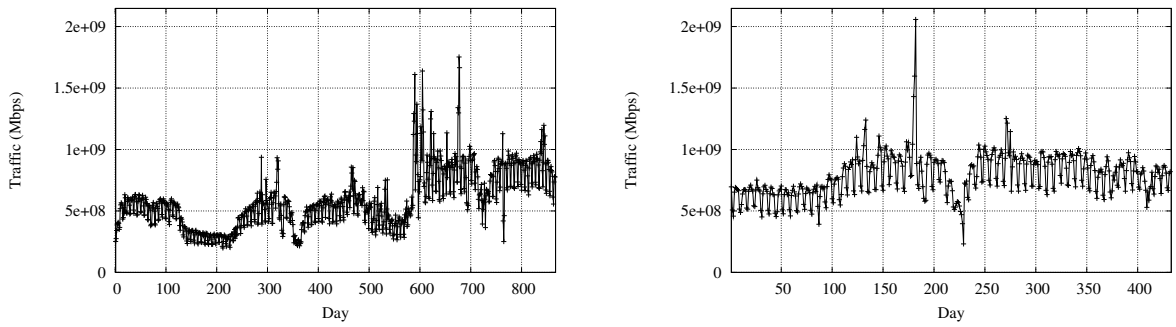
Figure 4: PolElec: 1601 samples of electricity consumption.



(a) Training series (236 samples).

(b) Test series (118 samples)

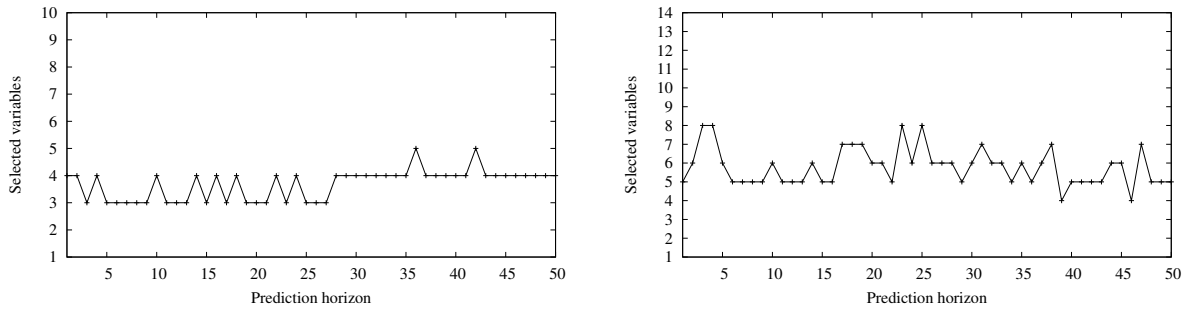
Figure 5: ESTSP08-1: Dataset 1 from the ESTSP 2008 competition (354 samples).



(a) Training series (867 samples)

(b) Test series (433 samples)

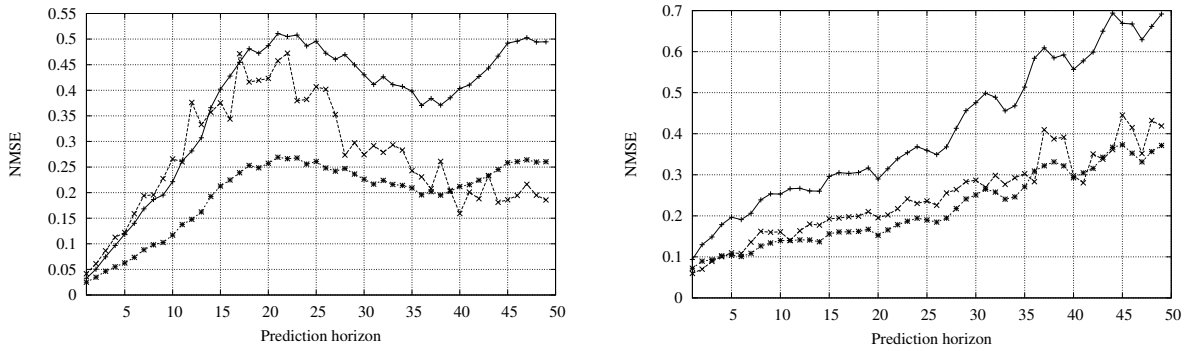
Figure 6: ESTSP08-2: Dataset 2 from the ESTSP 2008 competition (1300 samples).



(a) ESTSP07 Dataset

(b) PolElec Dataset

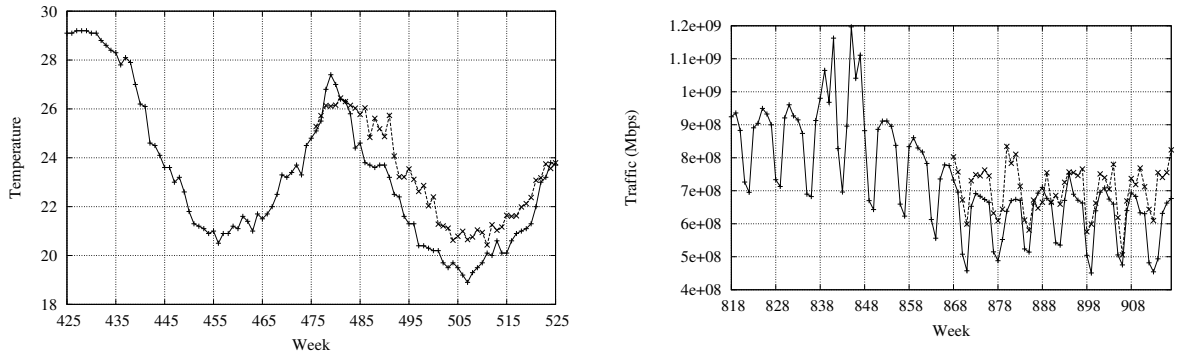
Figure 7: Number of selected variables for horizons 1 through 50 for the ESTSP07 and PolElec datasets.



(a) ESTSP07 Dataset

(b) PolElec Dataset

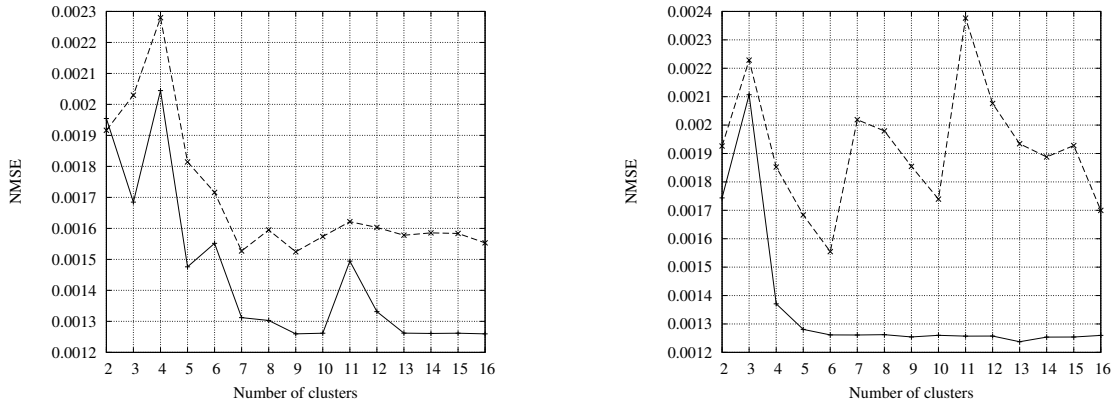
Figure 8: NDT estimates ($*$), training ($+$) and test ($\times$) errors of fuzzy autoregressors for the ESTSP07 and PolElec time series. DT based selection of inputs, ICFA$_f$-based identification and optimization with Resilient Propagation.



(a) ESTSP07 dataset

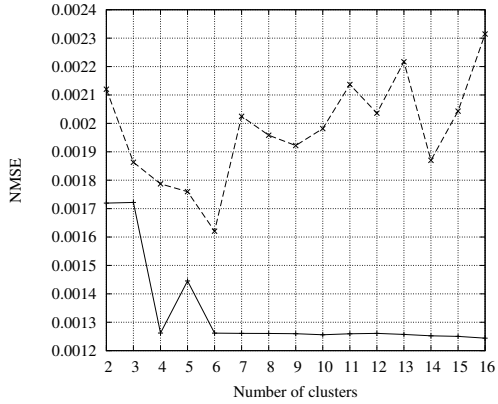
(b) ESTSP08-2 dataset

Figure 9: Prediction of 50 values after the training set for the ESTSP07 and ESTSP08-2 series. Continuous line (+): actual time series (last 50 training samples and first 50 test samples). Dashed line (×): predictions. DT based selection of inputs, ICFA$_f$-based identification and optimization with Resilient Propagation.

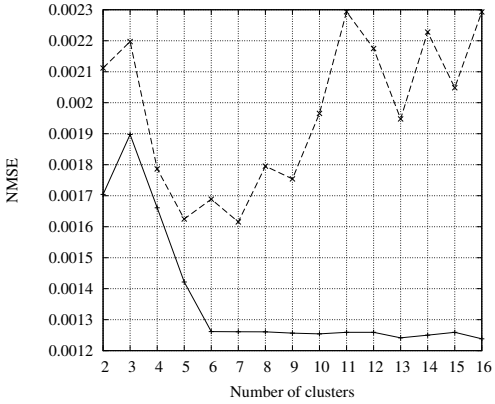


(a) ICFA (9)

(b) GG (6)

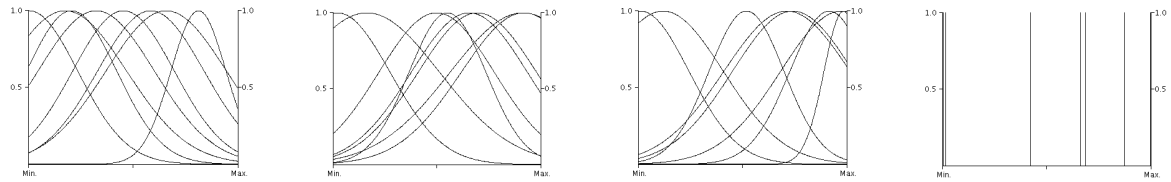


(c) FCM (5)



(d) HCM (5)

Figure 10: Training and test errors as a function of the number of clusters for the 1 step ahead model of the ESTSP07 series. The plots show training errors (continuous line) and test errors (dashed line). The number of clusters selected automatically within the proposed methodology for each clustering method is specified between parenthesis in the respective subheadings.



(a) y_{t-7} (b) y_{t-2} (c) y_t (d) Output

Figure 11: Membership functions of the inputs and output for the 1 step ahead model of the ESTSP07 series. ICFA$_f$ clustering.