A Constructive-Fuzzy System Modeling for Time Series Forecasting

Ivette Luna, Secundino Soares and Rosangela Ballini

***Abstract*— This paper suggests a constructive fuzzy system modeling for time series prediction. The model proposed is based on Takagi-Sugeno system and it comprises two phases. First, a fuzzy rule base structure is initialized and adjusted via the Expectation Maximization optimization technique (EM). In the second phase the initial system is modified and the structure is determined in a constructive fashion. This phase implements a constructive version of the EM algorithm, as well as adding and pruning operators. The constructive learning process reduces model complexity and defines automatically the structure of the system, providing an efficient time series model. The performance of the proposed model is verified for two series of the reduced data set at the Neural Forecasting Competition, for one to eighteen steps ahead forecasting. Results show the effectiveness of the constructive time series model.**

I. Introduction

Currently, there is a vast literature describing models in the field of time series for a variety of applications, such as filtering, prediction of future values, simulation or simply modeling to provide some kind of data description [1]. From all these tasks, time series prediction is a difficult one that involves different stages and very challenging problems like input selection, model generation and model evaluation.

On the one hand, it is common that techniques for building time serie models be based on linear strategies, considering only linear dependencies among variables, as the traditional Box & Jenkins methodology [2]. The drawback of applying this kind of modeling for nonlinear problems is that they are almost unable to capture nonlinear relationships among variables, resulting in models with low performance and poor prediction results.

On the other hand, non-linear modeling based on neural and fuzzy techniques have emerged as effective nonlinear predictor alternatives, since they are able to capture nonlinear input-output relationships. However, the determination of the optimal structure is a critical issue that cannot be addressed analytically. If some understanding of the problem or some information is available, and the complexity of the function to map is low, thus, the possibility of finding a topology close to optimal increases. When the complexity of the problem goes up, we have to use all the available methods, including techniques from the statistical theory field.

Time series modeling is a class of problem where the goal is to approximate an hypothetical function describing the behavior of a dynamic system on the basis of observations. The modeling task is then complicated by the ignorance of the optimal input space and all the model uncertainties tend to downgrade the prediction accuracy.

In this paper, the relevant inputs for building time series models are selected applying a combined methodology based on the False Nearest Neighborhood algorithm (**FNN**) [3] and the one proposed in [4], which is based on the Partial Mutual Information Criterion (**PMI**), originally proposed in [5]. This approach is applied because it is capable of dealing with non-linear relationship among variables involved, which is more appropriate for the problems under study.

It is well known that modeling methods are problem dependant and that any modeling methodology cannot pretend to be completely universal. At this paper, we suggest a time series model based on a constructive-fuzzy system modeling (**C-FSM**), which is built in two stages. First, an initial model structure based on two fuzzy rules is generated with available historical data. The parameters of this structure are adjusted via a classical Expectation Maximization (EM) algorithm [6]. In the second stage, the initial structure is modified and refined based on a constructive offline learning and on adding and pruning operators. Hence, the classical EM algorithm is adapted as a learning algorithm for fuzzy rule based systems. This algorithm develops parameters adjustment and an automatic structure selection simultaneously, which is a great advantage, when compared with other approaches of the literature.

The proposed approach is applied to forecast two series belonging to the reduced data set at the Neural Forecasting Competition. The constructive models are adjusted, analyzed and evaluated through a multiple steps ahead forecasting.

The paper is organized as follows. Section II introduces the structure of the **C-FSM**. Based on the proposed architecture, Section III details the constructive learning algorithm. Simulation results are shown in Section IV. Finally, some conclusions are presented in Section V.

II. Structure of the **C-FSM**

The structure of the **C-FSM** is composed by a set of M fuzzy rules, which are based on the first order Takagi-Sugeno (TS) fuzzy system [7], as illustrated in Fig. 1.

Let $\mathbf{x}^k = [x_1^k,\ x_2^k,\ \ldots,\ x_p^k] \in \mathbb{R}^p$ be the input vector at instant k, $k \in \mathbb{Z}_0^+$; $\hat{y}^k \in \mathbb{R}$ the output model, for its correspondent input $\mathbf{x}^k$.

The purpose is to build a nonparametric model, based on a fuzzy rules based system, dividing the global problem in subproblems with a lower complexity degree. This goal is achieved partitioning the input space into M subregions. Each partition is defined by its center $\mathbf{c}_i \in \mathbb{R}^p$ and its

Ivette Luna and Secundino Soares are with the Department of System Engineering, Faculty of Electrical Engineering, State University of Campinas-SP, CEP 13083-852, Brazil (phone: 0xx55 19 3521-3705; fax: 0xx55 19 3289-4313; email: {iluna,dino}@cose.fee.unicamp.br).

Rosangela Ballini is with the Department of Economic Theory, Institute of Economics, State University of Campinas-SP, CEP 13083-857, Brazil (phone: 0xx55 19 3521-5707; fax: 0xx55 19 3289-1512; email: ballini@eco.unicamp.br).

covariance matrix $\mathbf{V}_i \in \mathbb{R}^{p\times p}$. Hence, a fuzzy rule is assigned to each partition, and a local model is adjusted. As partitions are not roughly defined, every input pattern will have a membership degree g_i^k associated to each subregion.

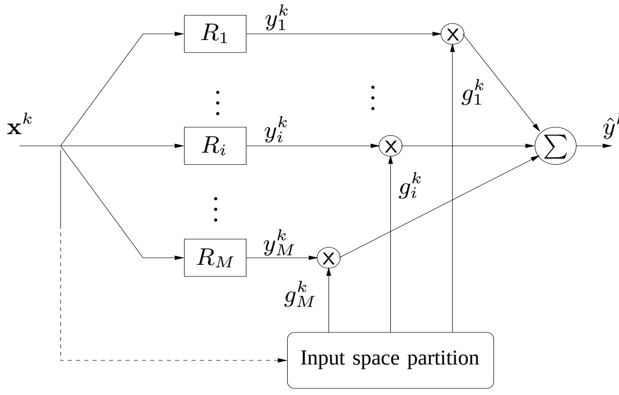

Fig. 1. A general structure of the **C-FSM**, composed by a total of M fuzzy rules.

The system emulates a fuzzy reasoning mechanism, encoding a fuzzy rules base in the form of **IF** <antecedent> **THEN** <consequent>. The antecedent part is represented by a subregion of the input space, and the consequent part is delineated by a local model. Therefore, the $i-$th fuzzy rule is defined as

$$R_i^k: \quad \textbf{IF} \quad < \mathbf{x}^k \text{ belongs to the "}i-\text{th" region with a membership degree } g_i^k > \textbf{THEN} \quad < \hat{y}^k = y_i^k > \quad (1)$$

where y_i^k, $i = 1,\dots,M$, is defined by a local linear model:

$$y_i^k = \phi^k \times \theta_i{}^T \quad (2)$$

which $\phi^k = [1\ x_1^k\ x_2^k\ \dots x_p^k] \in \mathbb{R}^{(p+1)}$ is a vector composed by the $k-$th input vector and a constant term; $\theta_\mathbf{i} = [\theta_{i0}\ \theta_{i1}\ \dots\ \theta_{ip}] \in \mathbb{R}^{(p+1)}$ is the coefficients vector for each local model. Membership degrees $g_i(\mathbf{x}^k) \in [0,1],\ i = 1,\dots,M$, are computed as:

$$g_i(\mathbf{x}^k) = g_i^k = \frac{\alpha_i \cdot P[\ i \mid \mathbf{x}^k\ ]}{\sum_{q=1}^{M} \alpha_q \cdot P[\ q \mid \mathbf{x}^k\ ]} \quad (3)$$

being α_i positive coefficients satisfying $\sum_{i=1}^{M} \alpha_i = 1$. These coefficients can be considered an indirect measure of relevance for each rule generated during the training process, so that, a higher value of α_i is, indicates that the respective rule is more relevant for the modeling task. Besides, $P[i \mid \mathbf{x}^k]$ in Eq.(3) is defined as

$$P[\ i \mid \mathbf{x}^k\ ] = \frac{1}{(2\pi)^{p/2}\det(\mathbf{V}_i)^{1/2}} \times \times\exp\left\{-\frac{1}{2}(\mathbf{x}^k - \mathbf{c}_i)\mathbf{V}_i^{-1}(\mathbf{x}^k - \mathbf{c}_i)^T\right\} \quad (4)$$

where $\det(\cdot)$ is the determinant function. Eq. (4) also represents the conditional probability of activating the $i-$th rule, given the input vector $\mathbf{x}^k$, as well as the parameters covariance matrix $\mathbf{V}_i$ and centers $\mathbf{c}_i$.

Indeed, from Eq.(3), g_i^k satisfies the unity condition:

$$\sum_{i=1}^{M} g_i^k = 1 \quad (5)$$

which is a condition necessary to consider g_i^k as a membership function in fuzzy theory [8].

The output model $\hat{y}^k$ is computed as a non-linear combination among the outputs of the local models y_i^k and theirs respective membership degrees g_i^k, that is:

$$\hat{y}^k = \sum_{i=1}^{M} g_i(\mathbf{x}^k)\ y_i^k \quad (6)$$

Eqs.(2)-(6), have been detailed for a single output. However, this model can be extended for modeling multiple outputs. Thus, the proposed model may be interpreted as an inference mechanism, codified into the model structure, where the model output $\hat{y}^k$ is calculated by means of an aggregated value of M local linear models y_i^k, according to Eq. (6). Notice that g_i^k relies on $\mathbf{x}^k$, $\mathbf{c}_i$ and $\mathbf{V}_i$.

Different papers in the literature have done interesting comparison studies among statistical methods and computational models based on neural networks and fuzzy systems (see [9] and [10]). Both, regression models and the **C-FSM** model proposed in this work, are used to model a relationship among variables, commonly represented as

$$\hat{y}^k = f[\ \mathbf{x}^k\ ] = f[\ x_1^k, x_2^k, \dots, x_p^k\ ]$$

where $f[\cdot]$ is the functional relationship between independent (or observable) variables $\mathbf{x}^k$ and the dependant variable y^k; $\hat{y}^k$ denotes the estimated value of y^k. Following this, the goal of the model **C-FSM** is to predict the single dependent variable y^k from its input variable $\mathbf{x}^k$, generally composed by lagged realizations of its output. This predicted value $\hat{y}^k$ is represented by the non-linear relationship defined by Eq.(6). In the case of the eleven time series from the reduced data set at the NN3 Competition, $\mathbf{x}^k$ is composed only by the past values of y^k, for one step ahead forecast.

A difference between these two approaches is that, regression models are defined by its coefficient vector, whereas the **C-FSM** is defined by a set of parameters, including centers $\mathbf{c}_i$, covariance matrices $\mathbf{V}_i$ and local models parameters θ_i, $i = 1,\dots,M$. Consequentely, while linear models have a closed form solution for the coefficients, the constructive one uses an iterative process. Despite this "extra cost", an advantage of the model proposed is that, applying the constructive learning, its structure and parameters are defined automatically, decreasing the difficulty of choosing parameters for building the time series models. The constructive learning is detailed as follows.

III. Constructive Learning of the C-FSM

This paper suggests a constructive offline learning for building a time series model. This algorithm determines automatically the number of fuzzy rules, as well as its internal parameters $\mathbf{c}_i$, $\mathbf{V}_i$ and θ_i, $i = 1, \ldots, M$, which define the entire structure of the model.

The learning algorithm comprises two stages: model initialization and adaptation.

A. First Stage: Model Initialization

First, an initial structure composed by two fuzzy rules ($M = 2$) is defined, and its parameters are adjusted via the traditional Expectation Maximization algorithm, originally proposed for mixture of experts models in [6]. The parameters of these rules are initialized as: $\mathbf{c}_i$ is equal to a pattern from the training set chosen aleatory; $\alpha_i = 1/M = 0.5$, while $\mathbf{V}_i = 10^{-4}\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Coefficients θ_i for local models are initialized with random values between $[0, 1]$ and the standard deviation $\sigma_i = 1.0$.

Based on the EM algorithm, model parameters are adjusted from a set of N input-output patterns, during an iterative sequence of EM steps. The goal of the EM algorithm is to find a set of model parameters, which will maximize the log-likelihood $\mathcal{L}$, of the observed values of y^k at each M step of the learning process. This objective function is defined by

$$\mathcal{L}(D,\ \boldsymbol{\Omega}) = \sum_{k=1}^{N} \ln\left(\sum_{i=1}^{M} g_i(\mathbf{x}^k,\ \mathbf{C}) \times P(y^k \mid \mathbf{x}^k,\ \theta_i)\right) \quad (7)$$

where $D = \{(\mathbf{x}^k, y^k) | k = 1, \ldots, N\}$, $\boldsymbol{\Omega}$ contains all model parameters and $\mathbf{C}$ contains just the antecedents parameters (centers and covariance matrices). However, for maximizing $\mathcal{L}(D,\ \boldsymbol{\Omega})$, it is necessary to estimate the missing data h_i^k during the E step. This missing data, according to mixture of experts theory, is known as the posterior probability of $\mathbf{x}^k$ belong to the active region of the $i-$th local model.

When the EM algorithm is adapted for adjusting fuzzy systems, h_i^k may also be interpreted as a posterior estimate of membership functions defined by Eq. (3), computed as

$$h_i^k = \frac{\alpha_i P(i \mid \mathbf{x}^k) P(y^k \mid \mathbf{x}^k,\ \theta_i)}{\sum_{q=1}^{M} \alpha_q P(q \mid \mathbf{x}^k) P(y^k \mid \mathbf{x}^k,\ \theta_q)} \quad (8)$$

for $i = 1, \ldots, M$. These estimates are called as "posterior", because these are calculated assuming $y^k,\ k = 1, \ldots, N$ as known. Moreover, conditional probability $P(y^k \mid \mathbf{x}^k,\ \theta_i)$ is defined by:

$$P(y^k \mid \mathbf{x}^k,\ \theta_i) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{[y^k - y_i^k]^2}{2\sigma_i^2}\right) \quad (9)$$

with the variance σ_i^2 estimated as:

$$\sigma_i^2 = \left(\sum_{k=1}^{N} h_i^k [y^k - y_i^k]^2\right) \Big/ \sum_{k=1}^{N} h_i^k \quad (10)$$

Hence, the EM algorithm for determining **C-FSM** parameters is summarized as:

1) E step: Estimate h_i^k via Eq. (8);
2) M step: Maximize Eq. (7) and update model parameters, with optimal values given as:

$$\alpha_i = \frac{1}{N} \sum_{k=1}^{N} h_i^k \quad (11)$$

$$\mathbf{c}_i = \left(\sum_{k=1}^{N} h_i^k \mathbf{x}^k\right) \Big/ \sum_{k=1}^{N} h_i^k \quad (12)$$

$$\mathbf{V}_i = \left(\sum_{k=1}^{N} h_i^k (\mathbf{x}^k - \mathbf{c}_i)'(\mathbf{x}^k - \mathbf{c}_i)\right) \Big/ \sum_{k=1}^{N} h_i^k \quad (13)$$

for $i = 1, \ldots, M$. Note that the covariance matrix $\mathbf{V}_i$ is a positive semidefined diagonal matrix. This condition suggests an alternative to simplify the problem and avoid unfeasible solutions. An optimal solution for θ_i is derived solving the following equation:

$$\sum_{k=1}^{N} \frac{h_i^k}{\sigma_i^2} \left(y^k - \phi^k \times \theta_i\right) \cdot \phi^k = 0 \quad (14)$$

where σ_i is the standard deviation for each local output y_i^k, with σ_i^2 defined by Eq.(10). After parameters adjustment, calculate the new value for $\mathcal{L}(D,\ \boldsymbol{\Omega})$.
3) If convergence is achieved, then stop the process, else return to step 1.

This optimization algorithm applied for model initialization, will also be the base for the structure adaptation, during the application of adding and pruning operators, as it will be detailed next.

B. Second Stage: Adaptation

The constructive part of the algorithm proposed consists in modifying the initial model structure. As it was mentioned before, it is based on the EM algorithm, detailed in Subsection III-A, and on adding and pruning operators. The adding operator consists in generating a new rule and incorporate it in the model structure. The pruning operator consists in eliminating an existing rule form the actual model. Thus, if there is some change in the actual structure some EM iterations will be applied, in order to become a new local optimal solution, taking into account restrictions and definitions given by Eqs. (8)-(14). In this way, model structure and parameters will be adjusted and selected automatically during the learning process.

1) Adding a new rule: The criteria proposed to judge whether to generate a new rule or not is called the if-part criterion. The if-part criterion evaluates if existing fuzzy rules can cover and cluster input vectors suitably. It means that, each input pattern must belong to a local rule with at least a minimum probability higher than a pre-defined threshold.

Assuming a normal input data distribution, with a confidence level of $\gamma\%$, we can construct a confidence interval $[\mathbf{c}_i - z_\gamma \sqrt{diag(\mathbf{V}_i)}, \mathbf{c}_i + z_\gamma \sqrt{diag(\mathbf{V}_i)}]$, where $diag(\mathbf{V}_i)$ is the main diagonal of the covariance matrix $\mathbf{V}_i$. In this paper, we get a confidence level of $\gamma = 72,86\%$ which requires a z_γ value of 1.1, from normal distribution table. Obviously,

$\gamma = 72,86\%$ is the middle chunk, leaving 13,57% probability excluded in each tail.

Suppose that the actual input pattern $\mathbf{x}^k$ has a conditional probability function $P[\ i \mid \mathbf{x}^k\ ] = P^k_{i^*}$, such that

$$P^k_{i^*} = \max\left(P[\ i \mid \mathbf{x}^k\ ]\right)_{i=1,\ldots,M}$$

if $P^k_{i^*}$ is lower than 0.1357, then this pattern does not belong to any active region of the fuzzy rules. Thus, the condition to generate a new rule may be written as:

$$P^k_{i^*} = \max\left(P[\ i \mid \mathbf{x}^k\ ]\right)_{i=1,\ldots,M} > 0.1357 \qquad (15)$$

If this condition is not satisfied, it means that there will be no rule that can cluster this input vector. Since the EM learning algorithm guarantees a local optimal solution, a new rule is added to the structure, so that the input space partition could be improved and a new optimal solution would be reached.

Suppose that the actual model is composed by a total of M fuzzy rules. If the condition defined by Equation (15) is not satisfied for one or more patterns, it means that these ones are not inside any active region of one of the M rules. Defining Ω as a set containing all input patterns that do not satisfy Eq. (15), a new rule will be generated, being its center initially estimated as:

$$\mathbf{c}_{M+1} = \frac{1}{N_\Omega}\sum_{t\in\Omega}\mathbf{x}^{\mathbf{t}} \qquad (16)$$

where N_Ω is the number of input patterns $\mathbf{x}^t$ in Ω. Indeed, all the other parameters for the new rule will be initialized as follows:

- $\sigma_{M+1} = 1.0$;
- $\theta_{M+1} = [\bar{y}\ 0\ \ldots\ 0]_{1\times(p+1)}$, where $\bar{y}$ is calculated being an average value of the outputs of all patterns in Ω.

In the case of α_i and $\mathbf{V}_i$, all these parameters were re-initialized for $i = 1,\ldots,M+1$ according to:

- $\mathbf{V}_i = 10^{-4}\mathbf{I}$, where $\mathbf{I}$ is a $p\times p$ identity matrix;
- $\alpha_i = 1/(M+1)$ and $M = M+1$;

It is necessary to re-initialize α_i and $\mathbf{V}_i$ in order to give the same importance to all rules during parameters adjustment and to achieve in that way a suitable input space partition, taking into account the new rule into the structure.

2) Pruning an existing rule: As it can be observed in Equation (11), α_i can be considered a measure of the importance that each fuzzy rule has for the corresponding topology, when compared to the other rules. It occurs because α_i is proportional to the sum of all posterior probabilities h^k_i over all the training data set. Thus, the more times the rule is strongly activated, the higher its corresponding α_i will be.

This characteristic will help to determine which rule to be pruned. Obviously, if it is necessary to prune some rule or local model, an acceptable candidate will be the one with the minimum value of α_i. A minimum value for α_i is defined, so that every rule with $\alpha_i < \alpha_{min}$ at each iteration is pruned and eliminated from the actual model structure.

After a modification in the structure is identified (adding or pruning), the model will be re-adjusted using the EM algorithm and Equations (11)–(10) for some iterations. The global process will just stop when convergence is achieved and there is no more evidence of changes on the model structure.

IV. Simulation Results

In this section, the proposed model is applied to build a time series model for two of the eleven time series composing the reduced data set at the Neural Forecasting Competition (NN3).

As it was mentioned in Section I, building a time series model implies a series of tasks to be solved. A strategy adopted in this work is composed by the next steps:

- Data analysis and pre-processing;
- Input selection;
- Model structure and parameter estimation;
- Model verification and evaluation.

A. Data analysis and pre-processing

Time series used in this paper for evaluating the **C-FSM** are series NN3_102 and NN3_104, which are part of the reduced data set at the Neural Forecasting Competition 2006/2007. Series NN3_102 is composed by a total of 126 monthly observations, from January 1979 to June 1989. Series NN3_104 is composed by 133 monthly observations, from January 1983 to July 1988. Both series have a seasonal component, that can be removed applying the following transformation on the original data:

$$z^k(m) = \frac{y^k(m) - \mu(m)}{\sigma(m)} \qquad (17)$$

where $z^k(m)$ is the stationary version of the time series $y^k,\ k = 1,\ldots,126$, at the $m-$th month, for a seasonal time series without a trend; $y^k(m)$ is the $k-$th observation, $\mu(m)$ is the monthly average value and $\sigma(m)$ is the monthly standard deviation.

A trend component may be avoided applying a first difference operation, several times as necessary. In order to determine the number of times that this operation needs to be applied, the unit root test was performed. This test, also known as the Augmented Dickey-Fuller test (ADF) [11] was applied with the help of the Econometric Views software [12]. The ADF test provides a *t*-statistic that determines the existence of unit roots in a regression model of the first difference of the series against the series lagged once, lagged difference terms and/or a constant and a time trend [12]

Therefore, evaluating the time series and comparing the obtained ADF *t*-statistic versus the reported critical values, the existence of unit roots was rejected. Then, there is no necessity to work with the integrated time series and we just remove the seasonal component. This procedure was followed for the entire reduced data set, observing that, for some series (ex. series NN3_101), the first difference transformation was enough to get its stationary version. These transformations will be applied for input selection and for adjusting the time series model.

B. Input selection

A suitable set of inputs is selected applying the False Nearest Neighborhood Algorithm - FNN [3] and the Partial Mutual Information Criterion - PMI [5].

The FNN algorithm is used to define an initial set of possible inputs and it is based on the reconstruction of the original space state of a dynamical system from the observed data (the time series). That is, a $d-$dimensional state space similar to the original and unknown $p-$dimensional state space is rebuilt.

For a given d, input patterns in $\mathbb{R}^d$ can appear as "neighbors" in the actual state space because of its projections, and not as a result of the system dynamic, whereas, for a higher d they move apart. Thus, to detect a suitable d, input patterns are built, for $d = 1, 2, 3 \ldots$ and so forth, asking at each step, how many patterns in the data set, as seen in dimension d, fail to remain close in dimension $d + 1$ [3]. Then d is increased until there are no more "false neighbors". Further details can be found in [3].

The FNN algorithm has shown as an interesting technique for input selection. However, this algorithm is not able to deal with redundant inputs, due to a selection based on consecutive lags. In order to verify redundancy and get a reduced number of input vectors, possible inputs defined by the FNN algorithm ($\mathbf{y}^{k-1},\ \mathbf{y}^{k-2}, \ldots, \mathbf{y}^{k-d}$) are evaluated via the Partial Mutual Information criterion (PMI) [5].

Partial mutual information is a measure of the partial or additional information that a new input can add to the existing prediction model. Given a dependent discrete variable Y (the output of the model), and an input X (the independent discrete variable), for a set of pre-existing inputs $\mathbf{Z}$, the discrete version of the PMI criterion is defined as:

$$\text{PMI} = \frac{1}{N} \sum_{i=1}^{N} \log_e \left[\frac{f_{X',Y'}(x_i', y_i')}{f_{X'}(x_i') f_{Y'}(y_i')} \right] \tag{18}$$

where:

$$x_i' = x_i - E(x_i|\mathbf{Z}) \tag{19}$$

and:

$$y_i' = y_i - E(y_i|\mathbf{Z}) \tag{20}$$

$E(\cdot)$ denotes the expectation function, x_i' and y_i' represent the residual components, corresponding to the $i-$th data pair sample, $i = 1, \ldots, N$, and $f_{X'}(x_i)$ $f_{Y'}(y_i)$ and $f_{X',Y'}(x_i, y_i)$ are the respective marginal and joint probability densities.

Using Eqs. (19) and (20) the resulting variables X' and Y' represent only the residual information in variables X and Y, since the effect of the existing predictors in $\mathbf{Z}$ have been taken into account [5]. As it can be noticed, good estimates of probability and density functions are necessary and quite relevant for calculating the PMI criterion. Some modifications were established in [4], so that some estimates were improved. Thus, expected values are approximated via the Nadaraya-Watson Estimator [13] and the city block kernel function, whereas marginal and joint probability functions are estimated via kernel functions, due to its efficiency and robustness, as shown in [5].

Advantages applying the PMI criterion for input selection are its capability of detecting all linear and non-linear relations among inputs and outputs variables and its independence of model structure (linear or non-linear). More details can be found in [3], [4], [5] and [14].

Eqs. (19) and (20) are applied iteratively, until the PMI value of a selected input be inferior to a confidence measure.

Confidence limit is calculated assuming independence among inputs and the output variable, generating p different arrangements of the independent variable by bootstrapping. The PMI value is calculated for each arrangement and a null test of hypothesis is built. The PMI value associated to the γth percentile will be equal to the confidence limit. That is, if the PMI value of the selected input is greater than the PMI value of the γth percentile sample, it means that there is some significant dependence between the inputs and the output variable and, hence, the null test of hypothesis will be rejected. In this work, we used $p = 100$ different arrangements of the independent variable and $\gamma = 95\%$.

Initial sets of possible inputs defined by the FNN algorithm for time series NN3_102 and NN3_104 were composed by the first nine and four lags ($d_{NN3\_102} = 9$, $d_{NN3\_104} = 4$), respectively.

Table I shows PMI results obtained for both series. In the case of the series NN3_102, the process stopped at the third step, since the PMI value obtained for lag 9 (a possible input) was inferior to the PMI confidence limit. In the same way, the process for series NN3_104 was concluded at the fourth step. Thus, a model for the NN3_102 time series will have as inputs lags 1 and 3, whereas a time series model for NN3_104 series will have as inputs lags 1, 2 and 3.

TABLE I
FNN-PMI RESULTS FOR SERIES NN3_102 AND NN3_104.

	NN3_102			NN3_104		
Step	**Lag**	**PMI**	**Thres.**	**Lag**	**PMI**	**Thres.**
1	1	0.7775	0.1585	1	0.3793	0.1411
2	3	0.2348	0.1010	3	0.1705	0.1058
3	**9**	**0.0744**	**0.0956**	2	0.0351	0.0303
4	-	-	-	**4**	**0.0249**	**0.0322**

C. Model structure and parameters estimation

After input selection, model structure is initialized, according to the approach detailed in Section III-A, being model structure and parameters defined automatically applying the constructive learning presented in Section III-B. The parameter α_{min} for both models (NN3_102 and NN3_104) was set as $\alpha_{min} = 0.025$. Data set was split into two groups, a training set (in sample data) and a testing set (out sample data). The training set was composed by the first 105 and 94 input-output patterns, for the NN3_102 and NN3_104 models, respectively. In both cases, the first 3 observations for both models were lost because of the form of the input patterns. The testing set was composed by the last 18 data pairs, corresponding to the last 18 observations of each time series.

D. Model verification and evaluation

Autocorrelation and cross correlation checks were applied for model verification. The first 20 lags of the residual

TABLE II

GLOBAL PREDICTION ERRORS FOR SERIES NN3_102 AND NN3_104 OF THE REDUCED DATA SET

Series	In sample 1 step ahead			Out of sample 1 step ahead		Out of sample 1 to 18 steps ahead		In sample 1 step ahead		
NN3_102	$k = 4, \dots, 108$			$k = 109, \dots, 126$		$k = 109, \dots, 126$		$k = 4, \dots, 126$		
NN3_104	$k = 4, \dots, 97$			$k = 98, \dots, 115$		$k = 98, \dots, 115$		$k = 4, \dots, 115$		
Series	M	**sMAPE** (%)	**MAE** (u)	**sMAPE** (%)	**MAE** (u)	**sMAPE** (%)	**MAE** (u)	M	**sMAPE** (%)	**MAE** (u)
NN3_102	3	3.41	179.36	4.72	287.98	11.40	658.05	3	2.95	168.28
NN3_104	3	10.98	438.27	6.75	334.93	12.32	612.32	8	8.89	386.79

autocorrelations and cross correlation coefficients were estimated. Based on this analysis for both models, there were no evidence of an inadequate modeling.

Symmetric mean absolute percentage error (**sMAPE** (%)) and mean absolute error (**MAE** (u)) are calculated for model evaluation, where u represents the time series unit.

Table II shows numerical results obtained for one step ahead forecasting for in sample data, as well as one to eighteen steps ahead forecasting for the out of sample data. This kind of evaluation is just for having some idea of model performance. Table II also shows the global errors for a one step ahead forecasting over the entire historical data. Predicted values of NN3_102 and NN3_104 are depicted in Fig. 2. Because it is necessary to use all data available for predicting the future 18 values of the time series, a second training was performed. These future values are also illustrated in Fig 2.

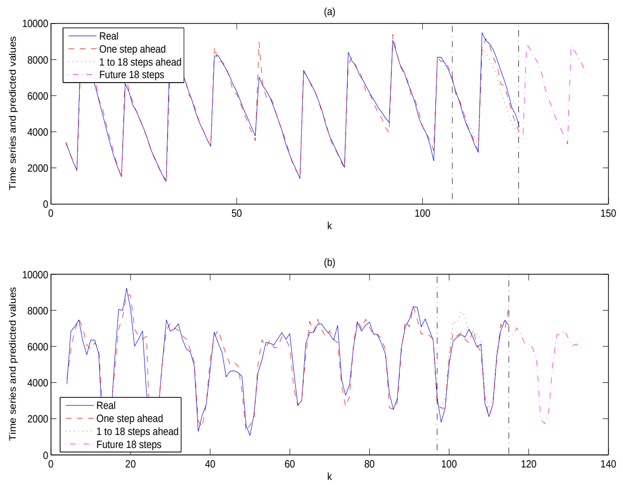

Fig. 2. Multiple steps ahead: (a) Predictions for series NN3_102. (b) Predictions for series NN3_104.

As we can observe in Fig. 2, both models captured the dynamic of the series, and the future eighteen value seem to follow historical data adequately. From Table II, we can also observe that global errors decrease when the number of training patterns increases, since the model has more information for a better adjustment. It is also interesting to realize that, for a different training set, we can get a different time series model, as occurred for time series NN3_104, where the first model obtained is composed by $M = 3$ fuzzy rules, being its training performed with input-output patterns from $k = 4, \dots, 97$. On the other hand, the second model for the same time series, with a training set composed by input-output patterns from $k = 4, \dots, 115$, is composed by a total of 8 fuzzy rules.

V. CONCLUSION

In this work, a constructive fuzzy system modeling was presented for building time series models. The constructive technique is based on the EM algorithm as well as in adding and pruning operations. These operators were applied to define automatically the model structure in an iterative batch learning process. The model was applied to forecast the NN3_102 and NN3_104 time series, which are part of the reduced data set at the NN3 Competition. Results showed an adequate modeling, as well as promising prediction results.

ACKNOWLEDGMENT

This work was partially supported by the Research Foundation of the State of São Paulo (FAPESP) and the Research and Projects Financing (FINEP).

REFERENCES

[1] P. J. Brockwell and R. A. Davis, *Introduction to time series and forecasting*, 2nd ed. Springer, 2002.

[2] G. Box, G. Jenkins, and G. C. Reinsel, *Time Series Analysis, Forecasting and Control*, 3rd ed. Oakland, California, EUA: Holden Day, 1994.

[3] H. Abarbanel and M. Kennel, “Local false nearest neighbors and dynamical dimensions from observed chaotic data,” *Physical Review E*, vol. 47, pp. 3057–3068, 1993.

[4] I. Luna, S. Soares, and R. Ballini, “Partial Mutual Information Criterion For Modelling Time Series Via Neural Networks,” *Proceedings of the 11th Information Processing and Management of Uncertainty International Conference*, July 2006.

[5] A. Sharma, “Seasonal to internannual rainfall probabilistic forecasts for improved water supply management: Part 1 – A strategy for system predictor identification,” *Journal of Hydrology*, no. 239, pp. 232–239, 2000.

[6] R. Jacobs, M. Jordan, S. Nowlan, and G. Hinton, “Adaptive Mixture of Local Experts,” *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[7] T. Takagi and M. Sugeno, “Fuzzy Identification of Systems and Its Applications to Modeling and Control,” *IEEE Transactions on Systems, Man and Cybernetics*, no. 1, pp. 116–132, January/February 1985.

[8] W. Pedrycz and F. Gomide, *An Introduction to Fuzzy Sets: Analysis and Design*. MIT Press, Cambridge, MA, 1998.

[9] B. Warner and M. Misra, “Understanding Neural Networks as Statistical Tools,” *The American Statistician*, vol. 50, no. 4, pp. 284–293, November 1996.

[10] R. S. Tsay, *Analysis of Financial Time Series*, 2nd ed. Wiley & Sons, 2005.

[11] J. D. Hamilton, *Time Series Analysis*. Princeton University Press, 1994.

[12] Quantitative Micro Software, “Econometric Views 2.0,” 1995.

[13] D. W. Scott, *Multivariate Density Estimation: Theory, Practice an Visualization*. John Wiley & Sons Inc., 1992.

[14] I. Luna, S. Soares, and R. Ballini, “A technique for identifying time series models,” *XVI Brazilian Automation Conference - CBA’06*, October 2006.