ELSEVIER

Signal Processing 49 (1996) 73–83

SIGNAL PROCESSING

Competitive local linear modeling

Carlos J. Pantaleón-Prieto[a], Ignacio Santamaría-Caballero[a], Aníbal R. Figueiras-Vidal[b,*]

[a] *Departamento de Electrónica, ETSI Industriales y de Telecomunicación, University of Cantabria, Av Los Castros s.n., 39005 Santander, Spain*

[b] *Departamento de Señales, Sistemas y Radiocomunicaciones, ETSI Telecomunicación, UPM, Ciudad Universitaria, 28040 Madrid, Spain*

Received 15 October 1993; revised 2 March 1995 and 16 November 1995

Abstract

This paper describes a novel approach for nonlinear signal modeling and prediction. We propose a nonlinear extension of the conventional AR model by using several linear models, each covering a subset of the whole signal data. Considering that AR modeling is conducted by associating a delay vector with the future value to be predicted, we interpret the signal as a codebook of patterns having the desired predicted value as the associated output. The pattern groups associated with each AR model are obtained by competition among the linear models as they are trained: the competition gives us the AR models as well as a classification of the data patterns. The obtained data groups are the basis for estimating a segmentation model, i.e., a classifier that, given a new data pattern, indicates what linear model should be used. The combination of the classifier and the linear models constitutes the final system. Several practical applications show the advantages of our approach.

Zusammenfassung

Dieser Aufsatz beschreibt einen neuen Ansatz zur nichtlinearen Signalmodellierung und Prädiktion. Wir schlagen eine nichtlineare Erweiterung des konventionellen AR Modells vor, bei dem mehrere lineare Modelle verwendet werden, um jeweils eine Teilmenge der gesamten Signaldaten abzudecken. Betrachtet man die AR Modellierung als die Verbindung eines Verzögerungsvektors mit dem zukünftigen, vorherzusagenden Signalwert, so können wir das Signal als ein Codebuch von Mustern interpretieren, die den gewünschten Vorhersagewert als zugehörigen Ausgang besitzen. Die Mustergruppen, die zu jedem AR Modell gehören, werden aus dem Wettbewerb der linearen Modelle während ihres Trainings gewonnen: dieser Wettbewerb liefert uns sowohl die AR Modelle als auch eine Klassifikation der Datenmuster. Die so erhaltenen Datengruppen sind die Grundlage für die Schätzung eines Segmentierungsmodells, d.h. eines Klassifikators, der für ein neu eingegebenes Datenmuster angibt, welches lineare Modell benutzt werden soll. Die Kombination des Klassifikators und der linearen Modelle stellt das endgültige System dar. Mehrere praktische Anwendungen zeigen die Vorteile unseres Ansatzes.

Résumé

Cet article décrit une approche nouvelle pour la modélisation et la prédiction non linéaires des signaux. Nous proposons une extension non linéaire du modèle AR conventionnel en utilisant plusieurs modèles linéaires, chacun couvrant un sous-ensemble de l'ensemble des données. La mise en oeuvre de cette modélisation AR consiste à associer un

* Corresponding author. Tel.: + 34-1-3367226; fax: + 34-1-3367350; e-mail: anibal@gtts.ssr.upm.es.

SSDI 0165-1684(95)00147-6

vecteur d'état à la valeur future à prédire, et nous interprétons le signal comme dictionnaire de formes ayant la valeur prédite désirée comme sortie associée. Les groupes de formes associés à chaque modèle AR sont obtenus par compétition entre les modèles AR lors de leur apprentissage: cette compétition nous donne les modèles AR ainsi qu'une classification des formes de données. Les groupes de données obtenus forment la base pour l'estimation d'un modèle de segmentation, c.à.d. un classifieur qui, en présence d'une nouvelle forme de données, indique quel modèle linéaire devrait être utilisé. La combinaison du classifieur et des modèles linéaires constitue le système final. Plusieurs applications pratiques montrent les avantages de notre approche.

Keywords: Nonlinear modeling; Piecewise linear; Competitive training; Nonlinear classifiers

1. Introduction

Signal modeling is a very common problem in many areas of science and engineering, usually solved by means of a linear model excited by a Gaussian distributed white noise process. In many practical cases the assumptions made by the linear model are violated, and a potential benefit may be obtained by generalizing it. If we deal with linear systems with non-Gaussian inputs, cumulant-based approaches can be used, allowing us to model signals that may be generated by nonminimum phase systems or which may have added colored noise. On the other side, nonlinear models offer a much wider framework and an undoubtedly much more realistic approximation of the real word [17, 24].

Most nonlinear models are based on a nonlinear functional of a finite number of past values of the signal under analysis $y[n]$, leading to the so-called nonlinear autoregressive model (NLAR) [24]

$$y[n+1] = \boldsymbol{f}(\boldsymbol{y}[n]) + e[n], \tag{1}$$

where $\boldsymbol{y}[n] = \{y[n], y[n-1] \ldots y[n-p+1]\}$ is the delayed sample vector with length p, and $e[n]$ the error term.

From the signal history we can build a collection of data patterns (the delayed data vectors) with associated desired outputs (samples to be predicted). Under this approach, sometimes called the Codebook paradigm [21], the modeling problem is transformed into a functional approximation task from non uniformly spaced samples (the pairs {data patterns, sample to be predicted}). Most nonlinear models proposed in literature try to approximate the function $\boldsymbol{f}$ in different ways, following one of two lines: either they attempt to approximate all the data set with a unique (global) model, or several submodels are used (local models), each dealing with a subset of the whole data set.

When considering *global* nonlinear models, the first reference is the Volterra theory of nonlinear modeling [19]. This approach offers a complete and exciting theoretical framework for the study of nonlinear systems at the expense of a great complexity in the final results and a very difficult application to practical problems. A general modeling methodology is also proposed with the state-dependent models of Priestley [17], where a linear state-varying model is presented. Again, the generality of this method is opposed to its practical applicability.

More simple global models have been proposed, looking for a compromise between complexity and applicability: NARMAX models [4] are based on a polynomial approximation of the functional $\boldsymbol{f}$, while bilinear models [22] only consider cross input–output terms. Exponential autoregressive models (EAR) introduce nonlinearity by allowing the coefficients to depend exponentially on the data [14]. In [26] nonlinearity appears as a multiplicative state-dependent function of the noise input in an AR scheme. These last models, rather than searching for a general modeling capability, attempt to reproduce phenomena which are typically encountered in nonlinear time series, as limit cycles or sudden bursts, that cannot be generated by linear approaches.

Global models exhibit several disadvantages: when dealing with complex systems the model becomes complex as well and difficult to understand and analyze: issues such as stability or invertibility cannot, in general, be accounted for. This type of models may also give unpredictable behaviors for

those data groups poorly represented in the data used for estimation.

Local models, on the other side, try to approximate the functional f with several models rather than a unique one. Most general approaches for local linear modeling assume that the function f is smooth enough, so the value in one point can be estimated as an average of the values in its neighborhood. Differences arise from the way this locality is exploited. In nonparametric prediction [11], an average of the closest values weighted with an exponential function of the distances is used for prediction. This approach is similar to a Gaussian Radial Basis Function approximation of f, using a function for each data sample [13]. Codebook modeling, as well, uses several linear models covering neighborhoods of data patterns [21]. The Multipredictor formulation, inherited from that of iterated functions systems (IFS) [7], attempts to define a general framework for this type of modeling.

Of course, in many applications (such as coding, for example), a nonlinear representation with as few parameters as possible is needed: a solution is to improve one linear model by using two, three ..., as few of them as possible for a given performance. Threshold models [24] normally include two or three linear models, and the selection of the model to use with a particular data pattern is accomplished by a threshold applied to a delayed sample of the time series.

The problem of all local approaches is to find the optimum dimension of the local model (the 'bandwidth selection' problem in [11]). If many models are estimated, the number of parameters becomes enormous and, because of the few patterns used to estimate each model, the confidence in the models decreases, while if very few models are used we loose quality in our predictions. In any case, defining the size of the local model, as well as which data patterns are associated to it, remains an open problem.

In this paper, we will consider a novel local (linear) model. This approach considers several linear models, each of them associated to some data patterns. The classification of the data patterns in groups is done by a competition among the models as they are trained: starting from random coefficient linear models, we sequentially present each training pattern to every model; the one that better predicts the future value is adapted via an LMS algorithm; the rest of the models are left intact. When the training algorithm finishes, we have a group of linear models as well as a collection of data patterns associated with each model. For prediction purposes, however, we need to select what model to use when dealing with a new data pattern: the competition has classified the training data patterns, but we have to infer a segmentation law for the whole data space, to make prediction possible. The estimation of this segmentation model is quite dependent on the particular problem we are facing; in a general case, however, a classifier will do the task.

Section 2 describes the competitive estimation of our model. In Section 3 we address the problem of estimating the segmentation model. Section 4 deals with the computation of the number of models, and, finally, Section 5 presents several practical applications of the proposed method. Some conclusions close this paper.

2. Competitive estimation of piecewise-linear models

Competitive learning is a well-known neurocomputing paradigm. It can be stated as follows: we have a collection of vector observations $\{\boldsymbol{x}(t)\}$ and a set of reference vectors $\{\boldsymbol{m}_i(t)\}$ initialized at random. We iteratively choose one of the $\{\boldsymbol{x}(t)\}$ and compare it with all the reference vectors using some metric. The winner of this competition reduces its distance (in the reference metric) to the training vector. When stability is reached, every reference vector represents a group of the training data [9].

Competitive learning makes the reference vectors concentrate on a particular group of patterns. It does not seem too risky, therefore, to think of extending this idea to models, and thinking that *if several models compete for training patterns, each one will concentrate in some group of them that share some kind of similarity*. So, we can expect that a competition among linear models will produce a meaningful classification of the data patterns.

Given a signal $y[n]$, we construct vectors of the form $\boldsymbol{y}_k = (y[k], y[k-1], \ldots, y[k-q+1])^{\mathrm{T}}$ with desired response $\boldsymbol{d}_k = y[k+1]$, q being the length of the patterns. Our objective is to estimate a collection of N AR models that transform the data vectors into their desired response with minimum squared error.

If $\boldsymbol{a}_j = (a_{(0,j)}, a_{(1,j)}, \ldots, a_{(q-1,j)})^{\mathrm{T}}$ is the vector of coefficients of the AR model number j ($j = 1, \ldots, N$); and we call $\boldsymbol{a}_m$ the best performing model for pattern $\boldsymbol{y}_k$, we can define the error in pattern k as

$$e_k = d_k - \boldsymbol{a}_m^{\mathrm{T}} \boldsymbol{y}_k \tag{2}$$

and the following cost function:

$$E[(e_k)^2], \tag{3}$$

E being the statistical expectation operator. A standard minimization of this cost function is not possible, so we have to rely on iterative approaches. Our proposal is that a model competition will give a collection of linear systems that solve this problem adequately.

Model competition is conducted as follows: we iteratively select at random a data pattern from our codebook and choose the best performing model $\boldsymbol{a}_m$ for that data pattern, as the one that better matches the desired response. Only that model is trained, leaving the rest of the models intact. $\boldsymbol{a}_m$ is updated with an LMS algorithm: this algorithm updates the coefficients at every iteration making a stochastic estimation of the gradient

$$\boldsymbol{a}_m^{l+1} = \boldsymbol{a}_m^{l} + \alpha e^{l} \boldsymbol{y}_k, \tag{4}$$

the error in iteration l being defined as

$$e^{l} = d_k - (\boldsymbol{a}_m^{l})^{\mathrm{T}} \boldsymbol{y}_k \tag{5}$$

and α is a positive update parameter. If we denote by $\boldsymbol{y}^{(j)}$ the patterns belonging to model j, convergence is assured if the update parameter for the adaptation of model j is made smaller than 2 divided by the largest eigenvalue of matrix $\boldsymbol{R} = E[\boldsymbol{y}^{(j)}(\boldsymbol{y}^{(j)})^{\mathrm{T}}]$. Since the data clusters are not available before the competitive estimation is carried out, this value cannot be calculated and we have to rely on a trial and error selection of α; a common solution in many related problems [9]. The competition is pursued till some termination criterion is fulfilled: the mean squared error over the data set falling below a threshold or maintaining a constant value over several iterations of the algorithm. When stability is reached, we end with a group of data clusters and a model for each group.

Table 1
Linear model competition

(1) Form a codebook of pairs ($\{y[k], y[k-1] \ldots y[k-q+1]\}$; $y[k+1]$) from the signal history
(2) Select the number of models you wish to consider and initialize them at a random value
(3) Choose randomly a data vector and apply every model; the one that better matches the desired response is trained with that pattern with the LMS algorithm. The rest of the models are left intact
(4) Go to (3) till stability is reached

Table 1 resumes the proposed algorithm.

The estimated model is not yet valid for prediction, however, because we still do not know how to assign a model to a new data pattern. This classification problem will be addressed in the next section.

3. Segmentation model

As we said before, we need to build a classifier that divides the data space, so we can know which model to assign to any possible input pattern. We will call this classifier the *segmentation model*: we consider our model composed of a *segmentation model*, that classifies the data patterns in groups, and several linear *prediction models*, which, given a data pattern, predict the next signal sample. Several possible cases can be considered:

3.1. We have a parametric segmentation model

The classification which we have obtained can be used to confirm the validity of the segmentation model as well as to estimate its parameters. We can switch back and forth between the segmentation model and the prediction model until we obtain a reasonable solution; i.e., once we have adjusted the data to the segmentation model, we estimate

again the linear models in the obtained data clusters.

An example of this situation is quasistationary signal modeling. This type of signals are composed of segments of stationary behavior combined with abrupt changes in their statistical properties in the transitions between segments. In this case, the signal generation mechanism is a collection of linear models and the segmentation law is a time driven change of model. So, we have a parametric segmentation model: it is composed of the time instants where the transitions between models are placed and it can be estimated from the data clusters obtained in the model competition. In the section devoted to simulations we will extend this discussion.

3.2. We have several tentative segmentation models

The data clusters obtained in the model competition are used to decide which of these models is more adequate. Under this perspective, every possible parametric segmentation function can be tried and the one that gives the best performance (minimum error) is selected. At this step, everything considered in Section 3.1 can be applied.

An example of this situation can be found in competitive modeling of speech, as presented in the simulations section. Basically, we attempt to code speech windows by establishing a two AR model competition, and transmitting both AR models, as well as the classification obtained in the competition. To optimize the coding, we can have a codebook of sequences of model transitions, and transmit only the identifier of the one that gives the lowest error, as in the CELP coder, where a codebook of excitation vectors is used [8].

3.3. We do not have a segmentation model

In this case, we are faced with a pattern classification problem, and several alternatives can be used to solve it. In this work, we will analyze two basic approaches: neural networks and nearest-neighbor classifiers; but other possibilities can be considered [9].

A part of the literature on neural networks is devoted to solving the pattern classification problem. Neural networks are composed of many nonlinear computational elements (nodes) operating in parallel. These nodes are connected via weights that are adapted to improve performance. One of the most used neural architectures is the multilayer perceptron (MLP).

The MLP is a feedforward net with one or more layers of nodes between the input and the output nodes. Each layer has several neurons, and each neuron in a layer is connected to the neurons in the adjacent layers with different weights. The weights in this architecture are trained by presenting every input pattern to the network: according to the difference between the produced and target outputs, the weights are adjusted to reduce the overall error. The standard training algorithm is known as backpropagation [9].

The MLP architecture can be used to learn the classification resulting from our model competition: we can present every data pattern in the signal history and seek as the target output the associated model number. Using classical train and test sets, we can dedicate a group of data patterns to train the network and the rest to test it, i.e., to evaluate if the network will behave correctly with data patterns not used for training. If the trained MLP shows good performance with the training and test sets, our model is finished. Signal prediction is accomplished by presenting a data pattern to the MLP, obtaining the AR model number and applying it. Repeated predictions allow us to generate signals.

A nearest-neighbor classifier can be applied as well to our problem [13]. This classifier assigns to every pattern the same category as its closest training pattern. The notion of closeness is associated with some type of metric: in our case we will use the Euclidean distance, because of its simplicity and robustness. So, we build our codebook of training patterns and a new data pattern is assigned to the same class as its closest one in the codebook.

The use of one approach or the other depends, in a general case, on the characteristics of the problem we are facing. However, the data length is to be taken into account. If our codebook is very large, the search for the nearest neighbor will be very time

Table 2
Estimation of the segmentation model

(1) Is there a segmentation model?
NO
(2) Train a classifier with the data patterns and the desired classes
YES
(2) Estimate the segmentation model parameters with the classes obtained in the model competition

consuming, and this can be a serious drawback in some applications. On the other side, when very reduced data sets are used, the MLP has very serious training problems. Of course, a comparison of both approaches will always render the best decision.

The estimation of the segmentation model can be resumed as in Table 2.

4. Order selection

In this section, we will address the problem of defining criteria to select our model order, i.e., the number of models to use, as well as the number of parameters of each model. Akaike's AIC criterion [2] is widely accepted for model order selection; it has been shown, however, that the asymptotical distribution of the optimal AIC order is not consistent: a probability greater than 15% remains that an order will be selected that is higher than the true process order [20]. The minimum descriptive length (MDL) criterion [18], on the other side, achieves consistency, and several comparative simulations have shown its superior performance. The MDL is defined

$$\hat{p} = \arg\left\{\min_{k\in\{0,1,\ldots,P\}}\left(\ln\hat{\sigma}_k^2 + \frac{k\ln N}{N}\right)\right\}, \tag{6}$$

where $\hat{\sigma}_k^2$ is the maximum likelihood estimation of the noise variance, N the length of the signal data, k the number of parameters of the model and P the maximum considered order. This criterion is composed of two terms. The first one will decrease monotonically as k increases, but the second term will grow with k to account for the increase in variance due to the estimation of extra parameters. The selected order will be the one that better balances both factors.

In our nonlinear model, we establish competition among a known number of linear models (in the next paragraph we will consider how is the number of models selected) with a given order. When the classifier is developed, and we have the final groups of data patterns (the classifier may modify the data groups given by the model competition), we may reestimate the order (and the parameters) of every AR model using the MDL criterion described above. With this approach we obtain a nonlinear model with a fixed number of linear blocks (Table 3).

At this point, we are able to estimate our model with any number of linear components. The only question that remains open is how to choose the best performing architecture among the models with one, two, ..., linear blocks. The problem closely resembles the AR order estimation one, and, similarly, our proposed approach is to define a maximum number of AR models, Q, and estimate the competitive model for every order. Again we have to establish a criterion to compare the performance of several piecewise-linear models.

The problem of model selection among several nonlinear candidates is an open line of research [4], and very few guidelines are available. A common approach is cross-validation, where the available data is divided in two subsets: the first one is used to estimate the parameters of the model and the second one for checking the performance [9, 23].

In our approach however, we have relied on a much simpler, and heuristic approach, similar to the one used for comparing different threshold models in [23]: the model selected will be the one that minimizes a combination of the MDLs of the AR blocks of each piecewise linear model. One possibility is adding the MDL values of the linear models that compose each piecewise-linear model. A

Table 3
Local linear model estimation for a fixed number of AR models

(1) Model competition (Table 1)
(2) Estimate segmentation model (Table 2)
(3) Estimate an AR model for each resulting data group using an MDL order selection criterion

Table 4
Estimation of the optimum number of AR models

(1) Define a maximum number of linear models Q
(2) For every model order k ($k = 1 \ldots Q$)
  (a) Local linear model estimation for a fixed number of AR models (Table 3)
  (b) calculate the MDL of the local linear model
(3) The model with the lowest MDL is selected.

second one, slightly different, is to apply the MDL criterion to the whole nonlinear model: in this case $\hat{\sigma}_k^2$ is the mean square error of the predictions in *all* the data patterns, k the number of parameters in *all* the linear models, and N the number of data samples. If a parametric classifier is used, its number of independent parameters can be added to k; if a nonparametric one is applied, some measure of complexity has to be inferred. Nonetheless, if we keep the number of models low, we can simplify the problem by supposing that the complexity of the classifier remains fixed and there is no need to consider it. Table 4 summarizes the model-order estimation in the general case.

5. Simulation results

The proposed algorithm has been applied to several signal modeling and prediction problems. We will consider here three practical applications: segmentation of quasistationary signals, speech modeling, and time-series prediction.

5.1. Segmentation of quasistationary signals

In [15] linear model competition is applied to the segmentation of quasistationary signals: these signals are composed of segments of stationary behavior combined with abrupt changes in their statistical properties in the transitions between different segments. They appear frequently in fields like speech analysis, seismology, vibration analysis, and econometrics.

Quasistationary signals fit very well with our modeling technique. We have several linear models and we have a segmentation model, i.e., the AR models have to be produced sequentially in time. Our hypothesis is that, if a competition among AR models is established, each one will concentrate in a particular time segment of the signal.

The simulations reported in [15] confirm these ideas: when competition is established among randomly initialized models, clustering takes place and each model concentrates in a particular time segment. Some simulations have been conducted to evaluate the performance of model competition for the segmentation of quasistationary signals in comparison with other approaches: the results suggest that model competition shows a reasonable compromise between complexity and performance in this type of tasks. Other simulations compared the MSE in modeling a particular quasistationary signal of the proposed competitive model and a unique AR model. The competitive model clearly outperforms the AR model by reaching a close to optimum performance, in the sense that the MSE obtained in modeling is very close to the one we would have obtained if the transition instants were known and a linear model would have been estimated in each stationary segment.

The last comparison may seem unfair in the sense that is obvious that a model composed of several AR models will outperform a unique one: but this approach is normal in many signal processing tasks, as, for example, in speech processing, in which a signal window (where several models may be present) is coded with a unique AR model. In the next section we will further analyze this fact.

5.2. Speech modeling

Obtaining a good speech model is of great interest in applications like coding, synthesis, and recognition: it will allow the use of less data bandwidth to transmit a speech signal, to get a higher-quality synthetic speech, or to develop better recognition systems. In these fields, linear prediction schemes are the most used, with their nonlinear counterparts showing little improvement at normally a higher computational cost [8].

A common approach in speech processing, due to the nonstationary nature of this signal, is to work with windows of a few milliseconds, and

estimating an AR model for each of these data windows. As a preliminary study, we compare the performance of a unique and two competing AR models applied to this type of speech segments.

For our simulations we use 20 voiced segments of speech, each 240 samples long. They were taken from real continuous speech recorded with a sampling rate of 8 kHz and 16-bit resolution. Half of them are from a male speaker and the other half from a female one. The segments represent utterances of the 5 Spanish vowels 'a', 'e', 'i', 'o', 'u'.

To compare both proposed approaches (a unique AR model and two competing AR models) we use the prediction gain, defined as

$$10 \log \left(\frac{\sigma_s^2}{\sigma_e^2} \right), \tag{11}$$

with σ_s^2 being the variance of the speech signal and σ_e^2 the variance of the prediction error.

We obtained the prediction gain over the data set comparing a unique AR model, of order ranging from 6 to 38, with two competing AR models. The results are shown in Fig. 1 where models of the same number of parameters are compared (we show the prediction gain of a model of order N and, in the same column, the one obtained with two competing models of order $N/2$). It can be seen how the competitive model increases the prediction gain, in some cases by more than 3 dB.

Of course, if we are working in speech coding, this prediction improvement is obtained at the additional cost of a classification of the data patterns, a classification that has to be transmitted, increasing the necessary bandwidth. Nonetheless, as can be seen in Fig. 2, the model transitions follow a quite regular pattern, and it seems reasonable to expect that this information can be efficiently coded.

Fig. 3 shows a preliminary draft of a modified version of the well-known CELP coder [8]. The CELP structure is maintained, but we substitute a competitive piecewise-linear model to the conventional short-delay AR filter. Another codebook it is necessary to include the time segments where each model is active. To develop this codebook is probably necessary to use pitch information, due to the periodic nature of the model transitions, as can be seen in Fig. 2.

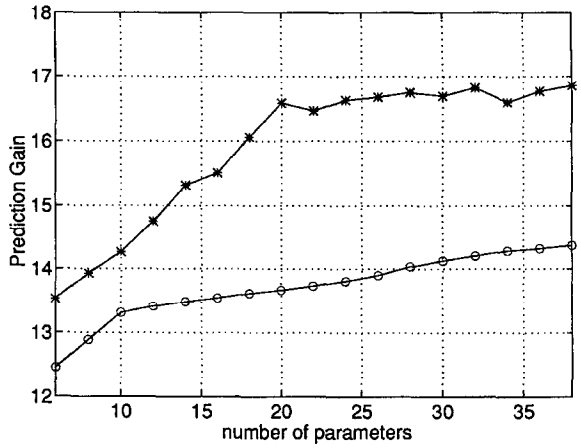

Fig. 1. Prediction gains with the same number of parameters (O) linear model and (*) competitive model.

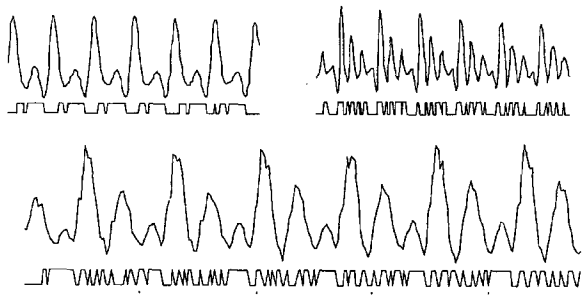

Fig. 2. Three examples of speech segments used in the simulations. Under every segment it can be seen a signal showing the time segments each model is active.

The suggested approach may introduce some additional improvements in the synthesized speech, noting that the change of model may serve as an adequate way of modeling phoneme transitions, usually problematic to account for in traditional approaches. To evaluate this improvement, we applied the algorithm proposed in [15] to speech segmentation.

We considered the Spanish sentence: "Diez días cabalgando en bici de montaña por el profundo sur marroquí al borde de los arenales del Sahara", sampled at 8 kHz and with 16-bit resolution. It is divided in 480 sample windows and, into each window, two AR models of order 10 compete, searching for a model transition. We accepted that a transition has taken place if we can divide the

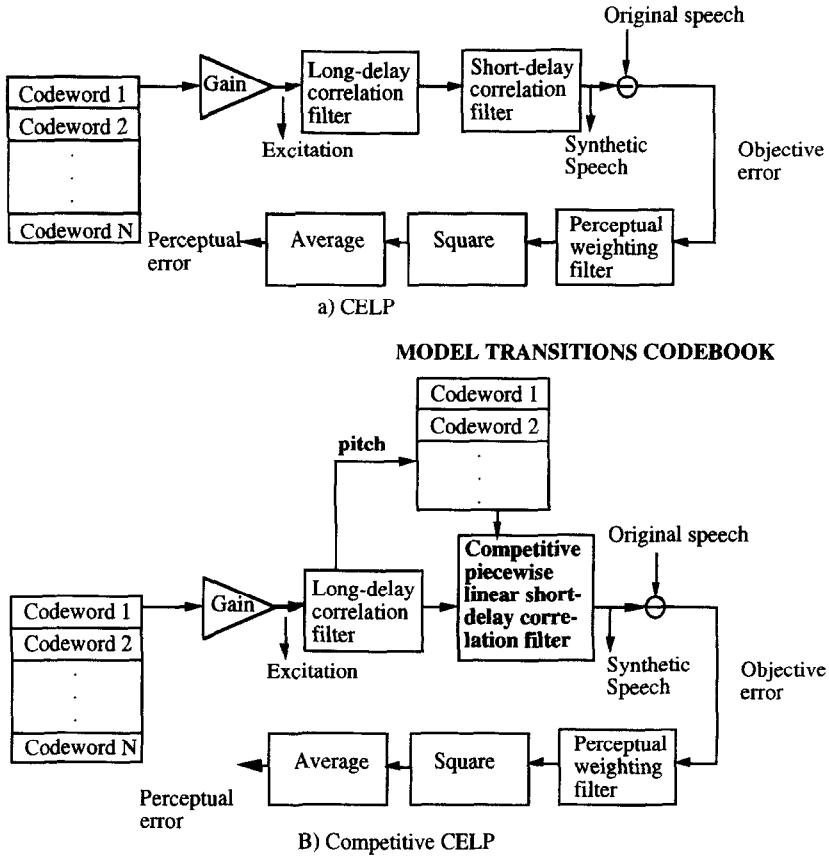

Fig. 3. Structure of (a) conventional CELP and (b) CELP with a piecewise linear short-delay filter.

time segments into two parts of a reasonable length (in our simulations more than 25 samples) and each part can be assigned to one of the models because it has won in, at least, 75% of the points that compose it [15].

The objective is not to evaluate the segmentation abilities of the proposed algorithm, but to know if the competitive distribution of models captures a model transition if it happens. We simulated 32 speech windows, and in 25 cases the results were correct: either there was a transition and it was detected or there was no transition and the algorithm guessed that transitions were not present. In the other 7 cases the scheme failed. In Fig. 4 we can see some examples of detected transitions. The competitive estimation of models captures, to some extent, the quasistationary nature of speech, giving reasonable hope that this approach will offer a good-quality synthesized speech.

As a conclusion, we can say that competitive modeling considerably improves the prediction gain of the conventional linear prediction schemes with the additional cost of transmitting the resulting classification. Besides, the two linear model approach allows us to model phoneme transitions within a window, a fact that may improve the final quality of the decoded speech.

5.3. *Time-series prediction*

Several time series have been widely accepted as presenting nonlinear features [24]; among them, one of the most studied is the Canadian-lynx time

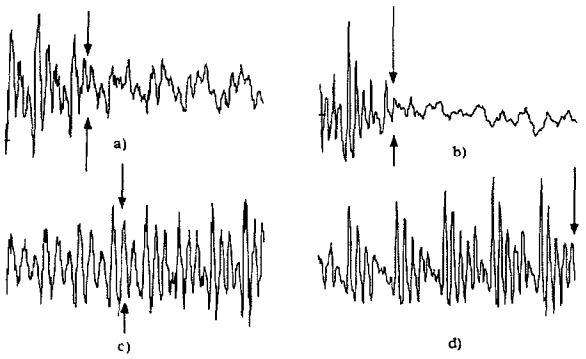

Fig. 4. Segmentation of speech. Cases (a), (b), (d) correct, (d) no transition detected, and (c) incorrect (no obvious transition).

series. In [16] we compared the performance of our piecewise linear model with the best performing among threshold models applied to this time series. In the next paragraph we will summarize the most important results. Other nonlinear models of this signal can be reviewed in [22].

The proposed modeling methodology was applied to the whole 114 point data set, using two models, with fixed orders 8 and 3 (the order of the linear components of the optimal threshold model [23]). Ten simulations of our model competition were carried out, obtaining two very similar classifications of data patterns, each five times: this fact suggests the existence of minimum error classification, and that competition leads to it. This quite surprising result is complemented with an important increase in performance: the residual variance was around 0.0136, opposed to a 0.0360 variance of the best performing threshold model. The same process, applied this time to the first 100 points of the time series, produced only one data pattern classification, rendering a residual variance of 0.0163 opposed to a 0.0405 variance of the threshold model.

If the competitive model is to be used for prediction purposes, the classifier that maps each data pattern into its associated prediction model needs to be estimated. Because no information is available about the structure of the segmentation model, we have to rely on general purpose classifiers: in the 114 point time series, after the model competition, we have 104 data patterns each with an associated-class (we predict from the 11th sample to compare adequately with known results concerning threshold models [23]). To build our classifier we took the first 88 patterns as the training set and the final 16 patterns as the test set. We trained MLP and nearest-neighbor arquitectures to classify the data patterns. We could not find an MLP arquitecture with a reasonable performance, probably due to the reduced data set. The nearest neighbor classifier (using patterns of length 3) offered the best results and was selected: the final system is composed of two linear models and a nearest-neighbor classifier that selects which model to use depending on the input pattern. Further simulations [16] evaluate the prediction capabilities and the long-term behavior of the estimated model, showing good comparative performance.

6. Conclusions

We have presented a new nonlinear time-series modeling methodology: considering that the prediction problem becomes the approximation of an unknown, nonlinear functional of the past samples of the data, we approximate this functional by a collection of linear functions, each covering groups of data patterns. We make no a priori assumption about the classification of these data patterns, but we establish a competition among linear models to obtain this segmentation. For prediction purposes, however, we have to be able to select what model to use when dealing with a new data pattern. Although we can find different situations, in the most general case a classifier can be trained to learn the resulting classification: we have analyzed MLP and nearest-neighbor approaches. An order estimation procedure completes the description of our algorithm.

The proposed method has been applied to several signal processing problems. In what concerns segmentation of piecewise stationary signals, we have shown how model competition is an efficient way of identifying the different data segments. When applied to speech modeling, our model achieves an important improvement over the conventional AR model, and its application to coding seems interesting. Finally, the proposed approach adequately models the Canadian-lynx time series, improving conventional techniques both in performance and computational cost.

Future lines of research include the development of a theoretical framework that proves that model

competition can be interpreted as an optimization approach; the analysis of the stability of the proposed model by using the Markov chains theory as in [24]; the comparison between the proposed hard-switching among models and soft-switching approaches (where more than one model can process the input pattern); and the application of this methodology to practical problems, with special interest in speech modeling and coding, where the proposed approach seems promising. In some applications where linear model competition will not work, it is interesting to further analyze other piecewise linear approaches, such as canonical piecewise linear models [10] or piecewise linear radial basis functions [6]. For more complex cases, approaches as modular neural networks [9] should be considered.

Acknowledgements

This work has been supported by CICYT grant TIC#92-0800-C05-01. We are grateful to the referees for numerous comments and suggestions which have greatly improved this paper.

References

[1] S. Ahalt, "Vector quantization using artificial neural network models", *Proc. 2nd Workshop Cost #229, Adaptive Algorithms and Non Classical Schemes.* Bayona (Spain), 13–15 March 1991, pp. 111–130.

[2] H. Akaike, "A new look at the statistical model identification", *IEEE Trans. Automat. Control*, Vol. 19, No. 6, December 1974, pp. 716–723.

[3] G. Box and G. Jenkins, *Time Series Analysis: Forecasting and Control*, Holden Day, San Francisco, 1976.

[4] S. Chen and S.A. Billings, "Model selection and validation methods for non-linear systems", *Internat. J. Control.* Vol. 45, No. 1, 1987, pp. 311–341.

[5] J. Cid-Sueiro and A.R. Figueiras-Vidal, "Digital equalization using modular neural networks: An overview", *Proc. 1995 Internat. Workshop on Digital Communications* (Invited paper), Viereggio, Italy, September 1995, To appear.

[6] N. Dyn, W.A. Light and E.W. Cheney, "Interpolation by piecewise-linear radial basis functions, I", *J. Aprox. Theory*, Vol. 59, 1989, pp. 202–223.

[7] G.C. Freeland and T.S. Durrani, "Multipredictor modeling with application to chaotic signals", *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process.*, Minneapolis. Vol. 3, 1993, pp. 133–136.

[8] S. Furui, *Digital Speech Processing, Synthesis, and Recognition*, Marcel-Dekker, New York, 1989.

[9] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Macmillan, New York, 1994.

[10] C. Kahlert and L.O. Chua, "A generalized canonical piecewise-linear representation", *IEEE Trans. Circuits Systems*, Vol. 37, No. 3, 1990, pp. 373–383.

[11] Y.K. Lee and D.H. Johnson, "Nonparametric prediction of non-gaussian time series", *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process.*, Minneapolis, Vol. 4, 1993, pp. 480–483.

[12] R.P. Lippman, "An introduction to computing with neural nets", *IEEE Acoustic Speech Signal Process. Mag.*, April 1987, pp. 4–22.

[13] O.J. Murphy, "Nearest neighbor pattern classification perceptrons", *Proc. IEEE*, Vol. 78, No. 10, 1990, pp. 1595–1598.

[14] T. Ozaki, "Non-linear time series models and dynamical systems", in: E.J. Hannan, P.R. Krishnaiah and M.M. Rao, eds., *Handbook of Statistics 5*, North-Holland, Amsterdam, 1985, pp. 25–85.

[15] C. Pantaleón-Prieto and A.R. Figueiras-Vidal, "Nonlinear time series modeling by competitive segmentation of state space", in: J. Mira, J. Cabestany and A. Prieto, eds., *New Trends in Neural Computation*, Springer, Berlin, 1993, pp. 525–531.

[16] C. Pantaleón-Prieto and A.R. Figueiras-Vidal, "Competitive nonlinear time series modeling and prediction", *Proc. Internat. Workshop Intelligent Signal Process. Commun. Systems, ISPACS-93,* Sendai, Japan, October, 1993, pp. 216–221.

[17] M.B. Priestley, *Non-Linear and Non-Stationary Time Series Analysis*, Academic Press, London, 1989.

[18] J.A. Riessanen, "Universal prior for the integers and minimum description length", *Ann. Statist.*, Vol. 11, 1983, pp. 417–431.

[19] M. Schetzen, "Nonlinear system modeling based on the Wiener theory", *Proc. IEEE*, Vol. 69, 1981, pp. 1557–1573.

[20] R. Shibata, "Selection of the order of an autoregressive model by Akaike's information criterion", *Biometrika*, Vol. 63, No. 1, 1976, pp. 117–126.

[21] A.C. Singer, G.W. Wornell and A.V. Oppenheim, "Codebook prediction: A nonlinear modeling paradigm", *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process.*, San Francisco, Vol. 5, 1992, pp. 325–328.

[22] T. Subba Rao, *An Introduction to Bispectral Analysis and Bilinear Time Series Models*, Lecture Notes in Statistics, No. 24, Springer, New York, 1984.

[23] H. Tong, *Threshold Models in Non-linear Time Series Analysis*, Lecture Notes in Statistics, No. 21, Springer, New York, 1983.

[24] H. Tong, *Non-Linear Time Series – A Dynamical System Approach*, Oxford Science Publications, Oxford, 1990.

[25] B. Townshend, "Nonlinear prediction of speech", in: M. Casdagli and S. Eubank, eds., *Nonlinear Modeling and Forecasting*, Addison-Wesley, Reading, MA, 1991, pp. 433–454.

[26] J.M. Vesin, "A nonlinear autoregressive signal model with state dependent gain", *Signal Processing*, Vol. 26, 1992, pp. 37–48.