A COMBINATORIAL APPROACH TO PIECEWISE LINEAR TIME SERIES ANALYSIS

MARCELO C. MEDEIROS, ALVARO VEIGA, AND MAURICIO G.C. RESENDE

ABSTRACT. Over recent years, several nonlinear time series models have been proposed in the literature. One model that has found a large number of successful applications is the threshold autoregressive model (TAR). The TAR model is a piecewise linear process whose central idea is to change the parameters of a linear autoregressive model according to the value of an observable variable, called the *threshold variable*. If this variable is a lagged value of the time series, the model is called a self-exciting threshold autoregressive (SETAR) model. In this paper, we propose a heuristic to estimate a more general SETAR model, where the thresholds are multivariate. We formulate the task of finding multivariate thresholds as a combinatorial optimization problem. We develop an algorithm based on a Greedy Randomized Adaptive Search Procedure (GRASP) to solve the problem. GRASP is an iterative randomized sampling technique that has been shown to quickly produce good quality solutions for a wide variety of optimization problems. The proposed model performs well on both simulated and real data.

1. INTRODUCTION AND PROBLEM DESCRIPTION

The most frequently used approaches to time series model building assume that the data under study are generated from a linear Gaussian stochastic process (Box et al., 1994). One of the reasons for this popularity is that linear Gaussian models provide a number of appealing properties, such as physical interpretations, frequency domain analysis, asymptotic results, statistical inference, and many others that nonlinear models still fail to produce consistently. Despite those advantages, it is well known that real-life systems are usually nonlinear, and certain features, such as limit-cycles, asymmetry, amplitude-dependent frequency responses, jump phenomena, and chaos cannot be correctly captured by linear statistical models.

Over recent years, several nonlinear time series models have been proposed both in classical econometrics (Tong, 1990; Granger and Teräsvirta, 1993; van Dijk et al., 2000) and in machine learning theory (Zhang et al., 1998; Kuan and White, 1994; Leisch et al., 1999). One model that has found a large number of successful applications is the threshold autoregressive model (TAR), proposed by Tong (1978) and Tong and Lim (1980). The TAR model is a piecewise linear process whose central idea is to change the parameters of a linear autoregressive model according to the value of a single observable variable, called the *threshold variable*. If this variable is a lagged value of the time series, the model is called a self-exciting threshold autoregressive (SETAR) model.

In this paper, we propose a heuristic to estimate SETAR models with thresholds defined by more than one variable. This is a generalization of the procedures described in Tong

and Lim (1980) and Tsay (1989), where the switching mechanism is controlled by a single threshold variable.

Multivariate thresholds are useful in describing complex nonlinear behavior and allow for different sources of nonlinearity. Several papers concerning multiple threshold variables have appeared in the literature during the past years. However, they assumed that the threshold was controlled by known linear combination of individual variables. See, for example, Tiao and Tsay (1994) where the thresholds are controlled by two lagged values of a transformed US GNP series reflecting the situation of the economy. In the present framework, we adopt a less restrictive formulation, assuming that the linear combination of variables is unknown and is jointly estimated with the other parameters of the model. An alternative approach is the Adaptive Spline AutoRegressive (ASTAR) model proposed by Lewis and Stevens (1991), which is based on Multivariate Adaptive Regression Splines (MARS) (Friedman, 1991).

We formulated the task of finding multivariate thresholds as a combinatorial optimization problem. Combinatorial optimization is a field of applied mathematics that treats a special type of mathematical optimization problem where the set of feasible solutions is finite. We developed an algorithm based on a Greedy Randomized Adaptive Search Procedure (GRASP), proposed by Feo and Resende (1989) (see also Feo and Resende (1995) and Resende (1999)), to solve the problem.

The paper is organized as follows. Section 2 gives a general description of threshold models. Section 3 presents the proposed procedure. Section 4 deals with the specification of the model. Section 5 briefly describes the GRASP methodology and presents its application to our particular problem. Section 6 presents some numerical examples illustrating the performance of the proposed model. Section 7 shows an application with a real data set. Concluding remarks are made in Section 8.

2. THRESHOLD AUTOREGRESSIVE MODELS

The Threshold AutoRegressive (TAR) model was first proposed by Tong (1978) and further developed by Tong and Lim (1980) and Tong (1983). The main idea of the TAR model is to describe a given stochastic process by a piecewise linear autoregressive model, where the determination of whether each of the models is active or not depends on the value of a known variable.

A time series y_t is a *threshold process* if it follows the model

$$y_t = \alpha_0 + \sum_{j=1}^{p} \alpha_j y_{t-j} + \sum_{i=1}^{h} \left[\lambda_{0i} + \sum_{j=1}^{p} \lambda_{ji} y_{t-j} \right] I_i(q_t) + \varepsilon_t, \tag{1}$$

where $\varepsilon_t \sim \text{NID}(0, \sigma^2)$. The terms $\alpha_0, \dots, \alpha_p$, and $\lambda_{0i}, \dots, \lambda_{pi}$, $i = 1, \dots, h$, are real coefficients. $I_i(\cdot)$ is an indicator function, defined by

$$I_i(q_t) = \begin{cases} 1, & \text{if } q_t \geq r_i; \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

where $\{r_1, \dots, r_h\}$ is a linearly ordered subset of the real numbers, such that $-\infty < r_1 < r_2 < \dots < r_h < \infty$. Usually, the variance of the error term is allowed to change according to the regime.

The model can be rewritten in vector notation as

$$y_t = \boldsymbol{\alpha}' \mathbf{z}_t + \sum_{i=1}^{h} \boldsymbol{\lambda}_i' \mathbf{z}_t I_i(q_t) + \varepsilon_t, \tag{3}$$

where $\boldsymbol{\alpha}' = [\alpha_0, \dots, \alpha_p]$, $\boldsymbol{\lambda}_i' = [\lambda_{0i}, \dots, \lambda_{pi}]$, and $\mathbf{z}_t' = [1, y_{t-1}, \dots, y_{t-p}]$.

Model (3) is composed of $l = h + 1$ linear autoregressive models of order p, AR(p), each of which will be active or not depending on the value of q_t.

The choice of the threshold variable q_t, which determines the dynamics of the process, allows a number of possible situations. An important case is when q_t is replaced by y_{t-d}, where the model becomes the self exciting threshold autoregressive model

$$y_t = \boldsymbol{\alpha}' \mathbf{z}_t + \sum_{i=1}^{h} \boldsymbol{\lambda}_i' \mathbf{z}_t I_i(y_{t-d}) + \varepsilon_t, \tag{4}$$

denoted by the acronym SETAR(l). The scalar d is known as the *delay parameter* or the *length of the threshold*.

Due to the discontinuity at each threshold, the derivative based optimization techniques cannot be applied to estimate the parameters of model (4). However, once the locations of the thresholds are determined, the least squares algorithm can be used to estimate each of the l linear models. Tong and Lim (1980) proposed a grid search based on Akaike's information criterion (Akaike, 1974) to specify the model and to estimate the parameters. Tsay (1989) proposed a simple model building procedure based on the residuals of an arranged autoregression. He suggested a simple statistic to test for the threshold nonlinearity and to specify the threshold variable. He also proposed graphical techniques to identify the number and the candidate locations of the thresholds. Both methodologies consider only thresholds controlled by a single lagged observation of the time series.

3. THE MULTIVARIATE THRESHOLD AUTOREGRESSIVE MODEL

3.1. **Model presentation.** As stated in Section 2, the dynamics of a SETAR model are controlled by a partition of the real line $\mathbb{R}$ induced by the parameters r_i, $i = 1, \dots, h$. However, in a more general situation, it will be interesting to consider a partition of a q-dimensional space, say $\mathbb{R}^q$. This paper proposes a procedure to estimate SETAR models with evolution controlled by a partition of a multidimensional space induced by h separating hyperplanes.

Consider a q-dimensional Euclidean space and a point $\mathbf{x}$ in that space. A hyperplane is defined by

$$\mathbb{H} = \{\mathbf{x} \in \mathbb{R}^q \,|\, \boldsymbol{\omega}'\mathbf{x} = \beta\}, \tag{5}$$

where $\boldsymbol{\omega}$ is a q-dimensional parameter vector and β is a scalar parameter. Figure 1 shows an example in $\mathbb{R}^2$. The direction of $\boldsymbol{\omega}$ determines the orientation of the hyperplane and the scalar term β determines the position of the hyperplane in terms of its distance from the origin.

A hyperplane induces a partition of the space into two regions defined by the halfspaces

$$\mathbb{H}^+ = \{\mathbf{x} \in \mathbb{R}^q \,|\, \boldsymbol{\omega}'\mathbf{x} \geq \beta\} \tag{6}$$

and

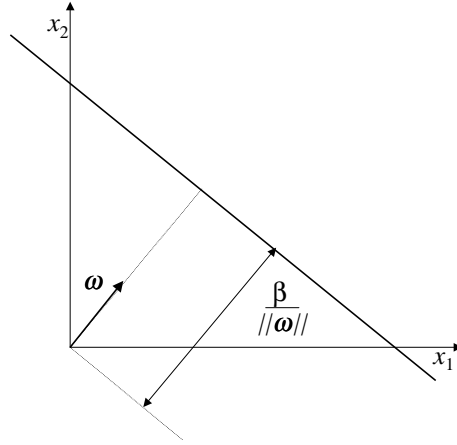$$\mathbb{H}^- = \{\mathbf{x} \in \mathbb{R}^q \,|\, \boldsymbol{\omega}'\mathbf{x} < \beta\}. \tag{7}$$

FIGURE 1. Hyperplane defined by $\boldsymbol{\omega}'\mathbf{x} = \beta$ in $\mathbb{R}^2$.

With h hyperplanes, a q-dimensional space will be split into several polyhedral regions. Each region is defined by the nonempty intersection of the halfspaces (6) and (7) of each hyperplane.

For a given hyperplane, defined by $\boldsymbol{\omega}$ and β, denote by $I_{\boldsymbol{\omega},\beta}(\mathbf{x})$ an indicator function

$$I_{\boldsymbol{\omega},\beta}(\mathbf{x}) = \begin{cases} 1, & \text{if } \boldsymbol{\omega}'\mathbf{x} \geq \beta; \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

The main idea of the proposed procedure is to use (8) to create a multidimensional threshold structure. Suppose that a q-dimensional space is spanned by q lagged values of a given stochastic process y_t, say $\mathbf{x}'_t = [y_{t-1}, \cdots, y_{t-q}]$, and suppose we have h functions $I_{\boldsymbol{\omega}_i,\beta_i}(\mathbf{x}_t)$, $i = 1, \ldots, h$, each of which defines a threshold. Now consider

$$y_t = \boldsymbol{\alpha}'\mathbf{z}_t + \sum_{i=1}^{h} \boldsymbol{\lambda}'_i\mathbf{z}_t I_{\boldsymbol{\omega}_i,\beta_i}(\mathbf{x}_t) + \varepsilon_t. \tag{9}$$

Equation (9) represents a SETAR model with multivariate thresholds, hereafter denoted by the acronym SEMTAR(h) – *Self-Exciting Multivariate Threshold AutoRegressive*. The maximum number of polyhedral regions $M(h,q)$ created by the hyperplanes is defined by the following recursive formula

$$M(h,q) = M(h-1,q) + M(h-1,q-1), \tag{10}$$

where $M(1,q) = 2$ and $M(h,1) = h+1$ are the boundary conditions.

Although a model with a large number of hyperplanes is difficult to interpret, in most practical situations we expect to have only a small number of separating hyperplanes.

Note that model (9) is, in principle, neither globally nor locally identified. There are two characteristics of the model which cause the non-identifiability: the first is due to symmetries in the model architecture. The likelihood function of the model will be unchanged if we permute the indicator functions, resulting in $h!$ possibilities for each coefficient of the model. The characteristic is the relationship $I_{\boldsymbol{\omega}_i,\beta_i}(\mathbf{x}_t) = 1 - I_{-\boldsymbol{\omega}_i,-\beta_i}(\mathbf{x}_t)$. The third characteristic is the mutual dependence of the parameters $\boldsymbol{\lambda}_i$ and $\boldsymbol{\omega}_i$, $i = 1, \ldots, h$. If all

the elements of $\boldsymbol{\lambda}_i$ equal zero, the corresponding $\boldsymbol{\omega}_i$ can assume any value without affecting the value of the likelihood function. On the other hand, if $\boldsymbol{\omega}_i = \mathbf{0}$, then $\boldsymbol{\lambda}_i$ can take any value.

The first problem is solved by imposing the restrictions $\beta_1 \leq \ldots \leq \beta_h$. The second problem can be circumvented, for example, by imposing the restriction $\omega_{1i} = 1$, $i = 1, \ldots, h$. To remedy the third problem, it is necessary to ensure that the model contains no irrelevant hyperplanes. This is solved with the techniques described in Section 4.3.

3.2. **Estimation of the parameters.** Rewrite model (9) as

$$\mathbf{y} = \mathbf{Z}\boldsymbol{\theta} + \boldsymbol{\varepsilon}, \tag{11}$$

where $\mathbf{y}' = [y_1, y_2, \ldots, y_T]$, $\boldsymbol{\varepsilon}' = [\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_T]$, $\boldsymbol{\theta}' = \left[\boldsymbol{\alpha}', \boldsymbol{\lambda}_1', \ldots, \boldsymbol{\lambda}_h'\right]$,

$$\mathbf{Z} = \begin{pmatrix} \mathbf{z}_1' & \tilde{\mathbf{z}}_{1,1}' & \cdots & \tilde{\mathbf{z}}_{h,1}' \\ \mathbf{z}_2' & \tilde{\mathbf{z}}_{1,2}' & \cdots & \tilde{\mathbf{z}}_{h,2}' \\ \mathbf{z}_3' & \tilde{\mathbf{z}}_{1,3}' & \cdots & \tilde{\mathbf{z}}_{h,3}' \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{z}_T' & \tilde{\mathbf{z}}_{1,T}' & \cdots & \tilde{\mathbf{z}}_{h,T}' \end{pmatrix},$$

and $\tilde{\mathbf{z}}_{i,t} = I_{\boldsymbol{\omega}_i, \beta_i}(\mathbf{x}_t)\mathbf{z}_t$, $i = 1, \ldots, h$.

Once the parameters $\boldsymbol{\omega}_i$ and β_i have been determined, the parameter vector $\boldsymbol{\theta}$ can be estimated by

$$\hat{\boldsymbol{\theta}} = \left(\mathbf{Z}'\mathbf{Z}\right)^{-1}\mathbf{Z}'\mathbf{y}. \tag{12}$$

The covariance matrix of $\hat{\boldsymbol{\theta}}$ conditional on $\boldsymbol{\omega}_i$ and β_i, $i = 1, \ldots, h$, can be estimated as

$$\hat{\boldsymbol{\Sigma}} = \hat{\sigma}^2 \left(\mathbf{Z}'\mathbf{Z}\right)^{-1}, \tag{13}$$

where $\hat{\sigma}^2$ is the estimated variance of the residuals. We can thus use (13) to test restrictions in the autoregressive parameters such as equality of models in different regimes.

The problem now is to estimate parameters $\boldsymbol{\omega}_i$ and β_i, $i = 1, \ldots, h$. As stated earlier in this section, these parameters define a hyperplane in a q-dimensional space. If we have N observations of $\mathbf{x}_t$, we must consider hyperplanes that separate the observed points. Of course, these hyperplanes are not unique. All that matters is how the points are partitioned. In that sense, we only need to consider the hyperplanes defined by combinations of these points. Consider χ the set of observations of $\mathbf{x}_t$, then the set of possible hyperplanes is defined by

$$\Upsilon = \{(\boldsymbol{\omega}_i, \beta_i) \mid \boldsymbol{\omega}_i'\mathbf{x}_t = \beta_i, i = 1, \ldots, h, \mathbf{x}_t \in \chi\}. \tag{14}$$

Hence, if we have N points, there are $N!/(n!(N-n)!)$ possible hyperplanes to search. One way would be to search all the possible combinations of hyperplanes and choose the combination that minimizes the sum of squared errors. Of course, for most practical problems this is infeasible. In Section 5 we propose a procedure based on GRASP that, given $\mathbf{x}_t$, is able to choose the set of h hyperplanes with small cost. In the next section, we discuss the specification of $\mathbf{x}_t$ and the selection of h.

4. MODEL SPECIFICATION

In this section, a specific-to-general specification strategy is developed. From equation (9) two specification problems require special care. The first is variable selection, that is, the correct selection of elements of $\mathbf{z}_t$ and $\mathbf{x}_t$. The problem of selecting the right subset of variables is very important because selecting a too small subset leads to misspecification,

whereas choosing too many variables aggravates the "curse of dimensionality." The second problem is the selection of the correct number of separating hyperplanes. The specification procedure as a whole may be viewed as a sequence consisting of the following three steps:

1. Specifying a linear autoregressive model.
2. Testing linearity and selecting the elements of $\mathbf{x}_t$.
3. Determining the number of hyperplanes.

4.1. **Specification of the autoregression.** The elements of $\mathbf{z}_t$ are determined with the use of an information criterion such as the AIC (Akaike, 1974) or the SBIC (Schwarz, 1978). Given a candidate set of lags, $\Omega = \{1, \dots, p_{max}\}$, we have to estimate several linear models and select the variables that minimize the information criteria. If we test each possible combination of lags, we would need to estimate $\sum_{i=1}^{p_{max}} p_{max}!/(i!(p_{max} - i)!)$ linear models. If p_{max} is very large, it is not reasonable to test every possible combination. In that case, the practitioner may only estimate p_{max} autoregressive models ranging from an AR(1) to an AR(p_{max}) model. However, in most practical situations we expect a low value for p_{max}.

The main drawback of this approach is that when the true data generating process is nonlinear, the algorithm tends to select more lags than necessary. Nevertheless, this does not pose any problem because after estimating the parameters of the nonlinear model, we can test the null hypothesis that an autoregressive parameter is zero.

4.2. **Testing linearity and specifying the threshold variables.** In practical nonlinear time series modeling, testing linearity plays an important role. In the context of model (9), testing linearity has two objectives. The first is to verify if a linear model is able to adequately describe the data generating process. If this is true, it is not necessary to fit a nonlinear model. The second is to determine the elements of $\mathbf{x}_t$.

The linearity test used in this paper is the F-test, proposed by Tsay (1989), based on the recursive least squares estimates of the parameters of an arranged autoregression.

Suppose we have the AR(p) model for y_t. We refer to $[y_t, 1, y_{t-1}, \dots, y_{t-p}]$ as a *case* of data. An arranged autoregression is an autoregression with the cases reordered, based on the values of a particular variable. In the framework of the SETAR model, arranged autoregression becomes useful if we reorder the cases according to the threshold variable y_{t-d}. Tsay (1989) proposed running the linearity for different values of d, and choosing the one that minimize the p-value of the test.

In the present framework it is, in principle, impossible to reorder the autoregression because we do not know the linear combination of threshold variables. To circumvent this problem, we adopt the following heuristic. Set $\mathbf{x}_t$ equal to each possible subset of the elements of $\mathbf{z}_t$, reorder the autoregression according to the first principal component of $\mathbf{x}_t$ and run the linearity test. Then, select the threshold variables that minimize the p-value of the test. Of course, the model builder can also use the other principal components to reorder the autoregression.

4.3. **Determining the number of hyperplanes.** In real applications, the number of separating hyperplanes is not known and should be estimated from the data. One possible solution is to start estimating a model with only one hyperplane, and then continue adding one hyperplane at a time to the model until the value of the SBIC (or the AIC) of the fitted model is not further improved. The SBIC and the AIC are defined as

$$\text{SBIC}(h) = \ln(\hat{\sigma}^2) + \frac{\ln(T)}{T} \times [h \times (p + q + 1) + p], \tag{15}$$

$$\text{AIC}(h) = \ln(\hat{\sigma}^2) + 2 \times \frac{[h \times (p + q + 1) + p]}{T}, \tag{16}$$

where $\hat{\sigma}^2$ is the estimated residual variance. This means that to choose a model with h separating hyperplanes, we need to estimate $h + 1$ models.

5. A GRASP FOR PIECEWISE LINEAR MODELS

A GRASP (Feo and Resende, 1989, 1995; Resende, 1999) is a multi-start iterative randomized sampling technique, with each GRASP iteration consisting of two phases, a construction phase and a local search phase. The best overall solution is kept as the result.

The construction phase of GRASP is essentially a randomized greedy algorithm, where a feasible solution is iteratively constructed, one element at a time. At each construction iteration, the choice of the next element to be added to the solution is determined by ordering all candidate elements in a candidate list with respect to a greedy function. This function measures the (myopic) benefit of selecting each element. The heuristic is adaptive because the benefits associated with every element are updated at each iteration of the construction phase to reflect the changes brought on by the selection of the previous element. The probabilistic component of a GRASP is characterized by randomly choosing one of the best candidates in the list, but not necessarily the top candidate. The list of best candidates is called the *restricted candidate list* (RCL). This choice technique allows for different solutions to be obtained at each GRASP iteration, but does not necessarily compromise the power of the adaptive greedy component of the method.

As is the case for many deterministic methods, the solutions generated by a GRASP construction are not guaranteed to be locally optimal with respect to simple neighborhood definitions. Hence, it is almost always beneficial to apply a local search to attempt to improve each constructed solution. Normally, a local optimization procedure, such as a two-exchange, is employed. While such procedures can require exponential time from an arbitrary starting point, empirically their efficiency significantly improves as the initial solution improves. Through the use of customized data structures and careful implementation, an efficient construction phase can be created which produces good initial solutions for efficient local search. The result is that often many GRASP solutions are generated in the same amount of time required for the local optimization procedure to converge from a single random start. Furthermore, the best of these GRASP solutions is generally significantly better than the solution obtained by a local search from a random starting point.

Figure 2 illustrates a generic GRASP in pseudo-code. The GRASP takes as input parameters for setting the maximum number of GRASP iterations and the seed for the random number generator. The GRASP iterations are carried out in lines 1–5. Each GRASP iteration consists of the construction phase (line 2), the local search phase (line 3) and, if necessary, the incumbent solution update (line 4).

In the framework of the piecewise linear time series modeling problem, we built a GRASP to estimate the separating hyperplanes of model (9). The greedy function proposed orders the possible hyperplanes with respect to the mean squared error (MSE) of the fitted model. As the number of hyperplanes is not known in advance, we build an outer loop where at each iteration we increase the number of hyperplanes by a unit. We stop when the SBIC (or the AIC) of the model is not further improved. The estimation is carried out in a static way, where we keep the estimates of the previous hyperplanes fixed and estimate only the last one. This is done to speed up the estimation process and does not have an adverse effect on the quality of the solution. Figure 4 shows the main loop in pseudo-code.

```
procedure grasp( )
1      do k = 1, ⋯ , MaxIter →
2            ConstructGreedyRandomizedSolution( );
3            LocalSearch( );
4            UpdateSolution( );
5      end do;
6      return(BestSolutionFound);
end grasp;
```

FIGURE 2. Generic GRASP.

```
procedure main( )
1      BestSolutionFound=OLS;
2      h = 0;
3      RandomSelectCases( );
4      while SBIC is improved →
5            h = h + 1;
6            Gplts( );
7            UpdateSolution( );
8            ComputeSBIC( );
9      end do;
10     return(BestSolutionFound);
end gptls;
```

FIGURE 3. Main loop procedure.

```
procedure gplts( )
1      do k = 1, ⋯ , MaxIter →
2            ConstructGreedyRandomizedSolution( );
3            RotationLocalSearch( );
4            TranslationLocalSearch( );
5            UpdateSolution( );
6      end do;
7      return(BestSolutionFound);
end gplts;
```

FIGURE 4. GRASP procedure.

Because the number of possible hyperplanes can grow very fast as a function of the number of observed points, before we start the outer loop we randomly select a subset of the cases and then generate the set of all possible hyperplanes, denoted by C.

We next describe each one of the components in detail.

5.1. **Construction phase.** In the construction phase each possible hyperplane is ordered according to the mean squared error (MSE) of the fitted model. To capture the adaptive

```
procedure ConstructGreedyRandomizedSolution( )
1      ℍ̄ = max{MSE(ℍ) | ℍ ∈ C};
2      ℍ_ = min{MSE(ℍ) | ℍ ∈ C};
3      RCL = {ℍ ∈ C | MSE(ℍ) ≤ ℍ_ + α(ℍ̄ − ℍ_)};
4      SelectHyperAtRandom(RCL);
5      s = s ⋃{ℍ};
6      AdaptCost( );
end ConstructGreedyRandomizedSolution;
```

FIGURE 5. Construction phase - Static estimation.

```
procedure ConstructGreedyRandomizedSolution( )
1      s = ∅;
3      do k = 1, ⋯ , MaxHyperSolution →
4            ℍ̄ = max{SSE(ℍ) | ℍ ∈ C};
5            ℍ_ = min{SSE(ℍ) | ℍ ∈ C};
6            RCL = {ℍ ∈ C | SSE(ℍ) ≤ ℍ_ + α(ℍ̄ − ℍ_)};
7            SelectHyperAtRandom(RCL);
8            s = s ⋃{ℍ};
9            AdaptCost( );
9      end do;
end ConstructGreedyRandomizedSolution;
```

FIGURE 6. Construction phase - Full estimation.

component, at each time a hyperplane is chosen, the remaining hyperplanes are reordered to reflect the benefits of the selection of the previous ones.

The random component of this GRASP sequentially selects, at random, the hyperplanes from the restricted candidate list (RCL) until the maximum number of hyperplanes is reached. In the case of static estimation, this means that only one hyperplane has to be chosen at each iteration. Otherwise, the construction phase selects h hyperplanes. Let $\alpha \in [0, 1]$ be a given parameter and MSE($\mathbb{H}$) the cost of selecting a given hyperplane from the set of all possible alternatives C, then the RCL is defined as

$$\text{RCL} = \{\mathbb{H} \in C \mid \text{MSE}(\mathbb{H}) \leq \underline{\mathbb{H}} + \alpha(\overline{\mathbb{H}} - \underline{\mathbb{H}})\} \tag{17}$$

where $\underline{\mathbb{H}} = \min\{\text{MSE}(\mathbb{H}) \mid \mathbb{H} \in C\}$ and $\overline{\mathbb{H}} = \max\{\text{MSE}(\mathbb{H}) \mid \mathbb{H} \in C\}$.

In this implementation at each GRASP iteration the parameter α is randomly chosen from a uniform distribution between 0 and 1. Figures 5 and 6 illustrate the construction phase of the GRASP for both static and full estimation. In the case of static estimation, the construction procedure receives as parameters the previous chosen hyperplanes that compose the solution s, the RCL parameter α and the subset C of candidate hyperplanes. Otherwise, only α and the subset C of candidate hyperplanes are passed to the construction procedure.

5.2. **Local search.** For a given problem, a local search algorithm works in a iterative fashion by successively replacing the current solution by a better solution in the neighborhood of the current solution with respect to some cost function. It terminates when there is no

```
procedure LocalSearch( )
1     do s is not locally optimal in N(s) →
2           FindBetterSolution( );
3           ReplaceSolution( );
4     end do;
end LocalSearch;
```
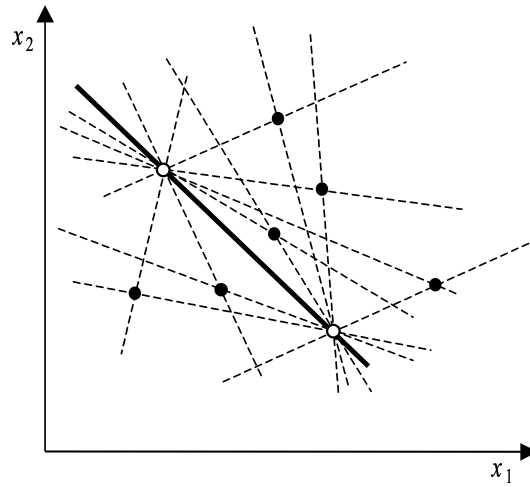
FIGURE 7. Local search phase.

FIGURE 8. Rotation local search.

better solution in the neighborhood. Figure 7 shows the pseudo-code of a general local search.

The local search implemented in this GRASP is a two-exchange local search, where a hyperplane that is in the solution set is replaced by another hyperplane that is not in the solution set. The local search is divided into two main blocks. The first rotates each hyperplane in the solution set. The rotation is carried out by substituting a point that generate the hyperplane by another point from the sample. Figure 8 illustrates the procedure in $\mathbb{R}^2$. The initial hyperplane (bold line) is defined by two points (white circles). The rotation local search consists of substituting the original points, one at a time, by all the other points in the sample (black circles), defining new hyperplanes. The possible hyperplanes are shown by the dashed lines. We finally select the hyperplane with the smallest cost.

The second block translates each hyperplane. This is accomplished by substituting, for each hyperplane, β_i by the elements of the projection $\boldsymbol{\omega}_i'\mathbf{x}_t$, $i = 1, \ldots, h$ and choosing the hyperplane with the smallest cost. See Figure 9 for details. The initial hyperplane is represented by the bold line. The grey circles are the projection of the points in the sample (black circles) in the direction of the vector $\boldsymbol{\omega}$. The hyperplanes generated by the translation local search are represented by the dashed lines.

Note that in the local search we consider all the points in the sample, and not only the subset of points that initially generated the hyperplanes.
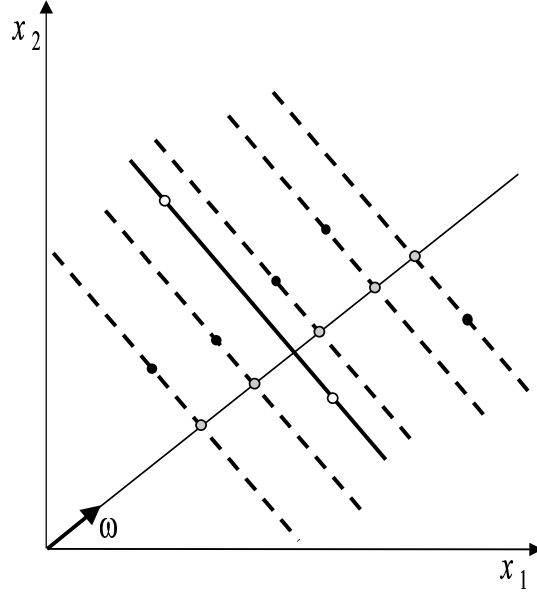
FIGURE 9. Translation local search.

6. MONTE-CARLO EXPERIMENT

In this section, we report the results of a simulation study designed to study the behavior of the proposed algorithms. The experiments were done on a Pentium II 266 MHz computer with 128 Mbytes of RAM. All the algorithms were programmed in MatLab.

We simulated several models, discarding the first 500 observations to avoid any initialization effects. The first 2 models are variations of the following basic model

$$y_t = 0.5 + 0.8y_{t-1} - 0.2y_{t-2} + (-0.5 - 1.2y_{t-1} +$$
$$0.7y_{t-2})I_{\boldsymbol{\omega}_1,\beta_1}(\mathbf{x}_t) + \varepsilon_t, \quad \varepsilon_t \sim \text{NID}(0,1^2), \quad (18)$$

where $\mathbf{x}_t = [y_{t-1}, y_{t-2}]'$.
- Model I: $\boldsymbol{\omega}_1 = [1,0]'$ and $\beta_1 = 0.5$
- Model II: $\boldsymbol{\omega}_1 = [1,-1]'$ and $\beta_1 = 1$

The last 2 models are based on the following specification

$$y_t = 0.5 + 0.8y_{t-1} - 0.2y_{t-2} + (-0.5 - 1.2y_{t-1} + 0.7y_{t-2})I_{\boldsymbol{\omega}_1,\beta_1}(\mathbf{x}_t)$$
$$+ (1.5 + 0.6y_{t-1} - 0.3y_{t-2})I_{\boldsymbol{\omega}_2,\beta_2}(\mathbf{x}_t) + \varepsilon_t, \quad \varepsilon_t \sim \text{NID}(0,1^2), \tag{19}$$

where $\mathbf{x}_t = [y_{t-1}, y_{t-2}]$.
- Model III: $\boldsymbol{\omega}_1 = [1,-1]'$, $\boldsymbol{\omega}_2 = [1,1]'$, $\beta_1 = -0.5$, and $\beta_2 = 1.6$.
- Model IV: $\boldsymbol{\omega}_1 = [1,-1]'$, $\boldsymbol{\omega}_2 = [1,-1]'$, $\beta_1 = -1$, and $\beta_2 = 1$.

6.1. **Estimation algorithm.** To evaluate the performance of the estimation algorithm in small samples, we simulated 100 replications of the last three models, each with 100 and 300 observations. We estimated the parameters for each replication, with $\mathbf{z}_t$ and $\mathbf{x}_t$ correctly specified and assuming that the number of hyperplanes was known. The GRASP is

TABLE 1. Mean and standard deviation of the estimates of the parameters based on 10 GRASP iterations over 100 replications of the models. True values between parentheses.

	100 observations					
	Model II		Model III		Model IV	
Parameter	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
$\hat{\omega}_{11}$	1 (1)	–	1 (1)	–	1 (1)	–
$\hat{\omega}_{21}$	–	–	1 (1)	–	1 (1)	–
$\hat{\omega}_{12}$	−0.9959 (−1)	0.3400	0.0200 (−1)	3.1300	−0.9914 (−1)	0.2538
$\hat{\omega}_{22}$	–	–	2.3800 (1)	26.1600	−0.9519 (−1)	0.2886
$\hat{\beta}_1$	0.8641 (1)	0.4212	−1.0757 (−0.5)	3.3367	−0.9375 (−1)	1.0216
$\hat{\beta}_2$	–	–	3.7917 (1.6)	12.7735	1.1140 (1)	0.6558

	300 observations					
	Model II		Model III		Model IV	
Parameter	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
$\hat{\omega}_{11}$	1 (1)	–	1 (1)	–	1 (1)	–
$\hat{\omega}_{21}$	–	–	1 (1)	–	1 (1)	–
$\hat{\omega}_{12}$	−0.9973 (−1)	0.0711	−0.5572 (−1)	3.5463	−1.0028 (−1)	0.0482
$\hat{\omega}_{22}$	–	–	0.6772 (1)	2.1848	−0.9996 (−1)	0.0413
$\hat{\beta}_1$	0.9946 (1)	0.0668	−0.3749 (−0.5)	0.4538	−0.9940 (−1)	0.1771
$\hat{\beta}_2$	–	–	2.0941 (1.6)	1.5065	0.9971 (1)	0.0991

based on the static estimation, where each hyperplane is estimated one at a time. Tables 1 and 2 show the mean and the standard deviation of the estimates based respectively on 10 and 20 GRASP iterations. The true value of the parameters are shown between parentheses. Table 3 shows solution quality. For Models II, III, and IV, the table shows the minimum, maximum, and mean (over 100 runs) MSE of the best solution found. If the models are correctly estimated we expect that the MSE of all the fitted models is around one, because in all the simulated models the error term is a normally distributed independent random variable with zero mean and variance one. Table 4 shows solution times. For Models II, III, and IV, the table shows the minimum, maximum, and mean (over 100 runs) total running time.

Observing Tables 1 and 2 we can see that the estimates of the hyperplanes in Models II and IV are rather precise in all cases considered. The performance is improved when we increase the sample size. The results presented in Table 3 show that the MSE of the fitted models is around one (as expected). Considering the solution times, the speed of the algorithm can be increased (up to 30%) if the code is implemented in C. Compiling the Matlab code is also a possibility.

6.2. **Model selection tests.**

6.2.1. *Variable selection.* Table 5 shows the results of the variable selection procedure based on 1000 replications of each model. The selection was made among the first five lags of y_t. The column C indicates the relative frequency of correctly selecting the elements of

TABLE 2. Mean and standard deviation of the estimates of the parameters based on 20 GRASP iterations over 100 replications of the models. True values between parentheses.

| | 100 observations | | | | | |
| | Model II | | Model III | | Model IV | |
Parameter	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
$\hat{\omega}_{11}$	1 (1)	–	1 (1)	–	1 (1)	–
$\hat{\omega}_{21}$	–	–	1 (1)	–	1 (1)	–
$\hat{\omega}_{12}$	−0.9588 (−1)	0.2966	−1.0185 (−1)	5.9265	−0.9984 (−1)	0.9845
$\hat{\omega}_{22}$	–	–	1.9920 (1)	22.8731	−1.0090 (−1)	0.5848
$\hat{\beta}_1$	0.8561 (1)	0.3825	−2.5887 (−0.5)	11.5999	−1.0922 (−1)	1.5785
$\hat{\beta}_2$	–	–	3.7791 (1.6)	13.3591	1.4848 (1)	3.0249

| | 300 observations | | | | | |
| | Model II | | Model III | | Model IV | |
Parameter	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
$\hat{\omega}_{11}$	1 (1)	–	1 (1)	–	1 (1)	–
$\hat{\omega}_{21}$	–	–	1 (1)	–	1 (1)	–
$\hat{\omega}_{12}$	−1.0167 (−1)	0.0975	0.1061 (−1)	1.2249	−0.9984 (−1)	0.0470
$\hat{\omega}_{22}$	–	–	−0.0426 (1)	2.3131	−1.0001 (−1)	0.0477
$\hat{\beta}_1$	0.9826 (1)	0.1413	−0.4099 (−0.5)	0.6778	−0.9720 (−1)	0.2025
$\hat{\beta}_2$	–	–	1.8328 (1.6)	0.7254	0.9974 (1)	0.0891

TABLE 3. Solution quality based on 100 runs.

| | 100 observations | | | | | | | |
| | 10 GRASP iterations | | | | 20 GRASP iterations | | | |
Model	Min.	Max.	Mean	Std. Dev.	Min	Max.	Mean	Std. Dev.
II	0.5907	1.2096	0.9015	0.1174	0.6091	1.4795	0.8836	0.1416
III	0.5487	1.1549	0.8054	0.1310	0.5560	1.1887	0.8020	0.1293
IV	0.5879	1.2260	0.8911	0.1224	0.5723	1.2814	0.8453	0.1359

| | 300 observations | | | | | | | |
| | 10 GRASP iterations | | | | 20 GRASP iterations | | | |
Model	Min.	Max.	Mean	Std. Dev.	Min	Max.	Mean	Std. Dev.
II	0.7889	1.1957	0.9692	0.0860	0.8135	1.3008	0.9799	0.0755
III	0.7863	1.2179	0.9727	0.0968	0.7602	1.2617	0.9743	0.0882
IV	0.8069	1.1805	0.9630	0.0795	0.7558	1.1996	0.9694	0.0845

$\mathbf{z}_t$. The columns U and O indicate, respectively, the relative frequency of underfitting and overfitting the dimension of $\mathbf{z}_t$.

Observing Table 5, we can see that the SBIC strongly underfits the models while the AIC tends to select more lags than necessary. With the exception of Model III, the performance of the variable selection procedure is improved, as it should, when we increase the sample size.

TABLE 4. Solution times based on 100 runs (in minutes).

	100 observations							
	10 GRASP iterations				20 GRASP iterations			
Model	Min.	Max.	Mean	Std. Dev.	Min	Max.	Mean	Std. Dev.
II	0.4880	0.9355	0.6797	0.0832	0.9198	2.0119	1.3360	0.1597
III	1.2488	1.7941	1.4720	0.1058	2.2209	3.4525	2.8549	0.1965
IV	1.2930	1.8525	1.5309	0.1072	1.9211	3.4680	2.9214	0.2481
	300 observations							
	10 GRASP iterations				20 GRASP iterations			
Model	Min.	Max.	Mean	Std. Dev.	Min	Max.	Mean	Std. Dev.
II	2.8259	4.8039	3.8499	0.4224	5.9120	10.8464	7.9806	0.9111
III	7.3928	10.9424	8.7800	0.7447	14.7106	22.9912	17.7978	1.2776
IV	7.6323	12.4441	9.3979	0.8337	15.1007	22.2525	18.5962	1.3210

TABLE 5. Relative frequency of selecting correctly the variables of the model at sample sizes of 100 and 300 observations based on 1000 replications among the first 5 lags.

	100 observations					
	C		U		O	
Model	SBIC	AIC	SBIC	AIC	SBIC	AIC
I	0.0280	0.0780	0.5970	0.4230	0.3750	0.4990
II	0.0430	0.1250	0.6320	0.3510	0.3250	0.5240
III	0.5490	0.3450	0.1330	0.0460	0.3180	0.6090
IV	0.1210	0.2740	0.6800	0.2420	0.1990	0.4840
	300 observations					
	C		U		O	
Model	SBIC	AIC	SBIC	AIC	SBIC	AIC
I	0.0960	0.2370	0.6910	0.2820	0.2130	0.4810
II	0.2780	0.2430	0.3930	0.0930	0.3290	0.6640
III	0.5090	0.2000	0.0010	0	0.4900	0.8000
IV	0.4190	0.4340	0.4370	0.0460	0.1440	0.5200

6.2.2. *Linearity tests and threshold variable selection.* In this section, we show results concerning the power of the linearity test and the selection of the threshold variables. Figures 10 and 11 show the size-power curve based on, respectively, 100 and 300 observations of 1000 replications of each one of the models using different lags of y_t as threshold variables. In power simulations we assume that $\mathbf{z}_t$ is correctly specified and we also tested the ability of the linearity test to identify the correct set of elements of $\mathbf{x}_t$. We expect that when $\mathbf{x}_t$ is correctly defined, the power increases. Table 6 shows the minimum, maximum, mean, and standard deviation of the F-statistic for each of the models based on 1000 replications. For each model, the first line indicates the results setting $\mathbf{x}_t = y_{t-1}$, the second line refers to $\mathbf{x}_t = y_{t-2}$, and the third line concerns $\mathbf{x}_t = [y_{t-1}, y_{t-2}]'$. As we can see, the F-statistic is a useful tool to identify the threshold variables.

6.2.3. *Selecting the number of hyperplanes.* Here we present the results concerning the selection of the number of hyperplanes. Table 7 shows the results using the SBIC and the AIC to select the number of hyperplanes. For models II, III, and IV, the table shows the

(a) Model I

(b) Model II
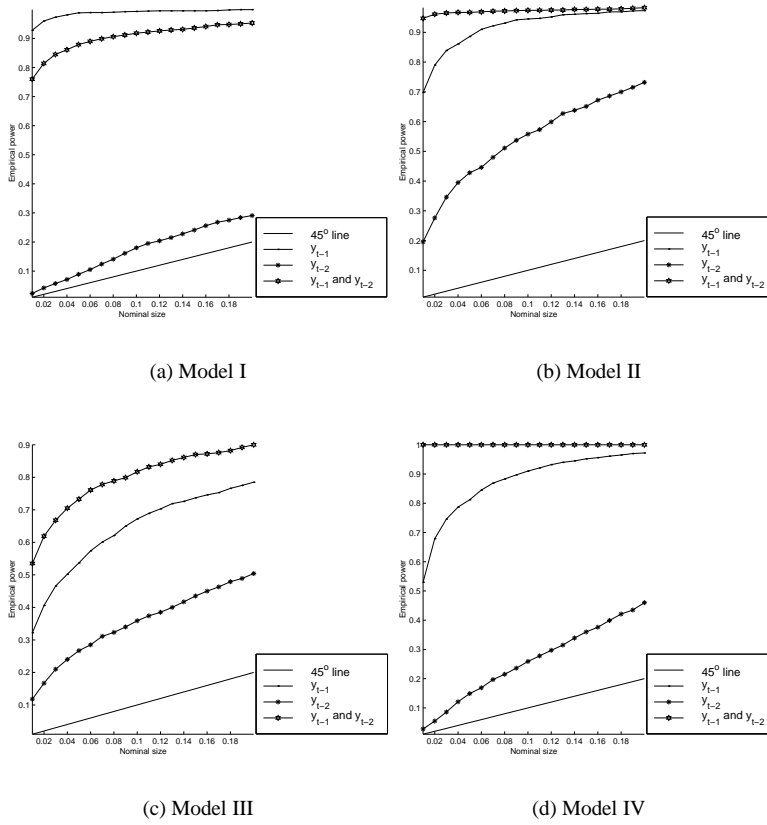
(c) Model III

(d) Model IV

FIGURE 10. Power-size curve of the linearity tests at sample size of 100 observations based on 1000 replications.

TABLE 6. Results of the linearity test over 1000 replications of each model.

	F-statistic							
	100 observations				300 observations			
Model	Min.	Max.	Mean	Std. Dev.	Min	Max.	Mean	Std. Dev.
I	1.4259	27.7415	**8.8306**	3.8273	10.1949	52.1402	**25.1747**	6.3151
	0.0209	7.9549	1.2486	1.0375	0.0135	12.8424	1.5743	1.3101
	0.0142	25.1537	6.6123	3.5042	1.6805	38.5694	19.3797	5.5554
II	0.3635	18.8093	5.7503	2.8052	5.3294	37.9985	15.7418	4.7390
	0.0255	10.7039	2.7226	1.6893	1.0323	20.1099	6.6388	2.8829
	0.0484	28.0097	**10.5415**	4.4804	14.1164	57.5102	**31.4149**	7.0637
III	0.1182	18.1091	3.5234	2.5165	0.6327	36.5506	8.3836	4.3773
	0.0226	16.1936	2.1128	1.8736	0.0494	26.6049	4.3220	3.2329
	0.1513	15.7074	**4.7543**	2.8791	2.6213	28.8574	**13.0340**	4.5432
IV	0.3207	14.8957	4.5251	2.0066	4.6570	21.6156	12.2263	2.9986
	1.0746×10^{-4}	6.6403	1.6398	1.0258	0.2203	10.4931	3.6886	1.5507
	6.4409	32.8783	**16.3745**	4.5413	27.8204	77.7052	**48.1796**	7.3982

(a) Model I

(b) Model II

(c) Model III

(d) Model IV
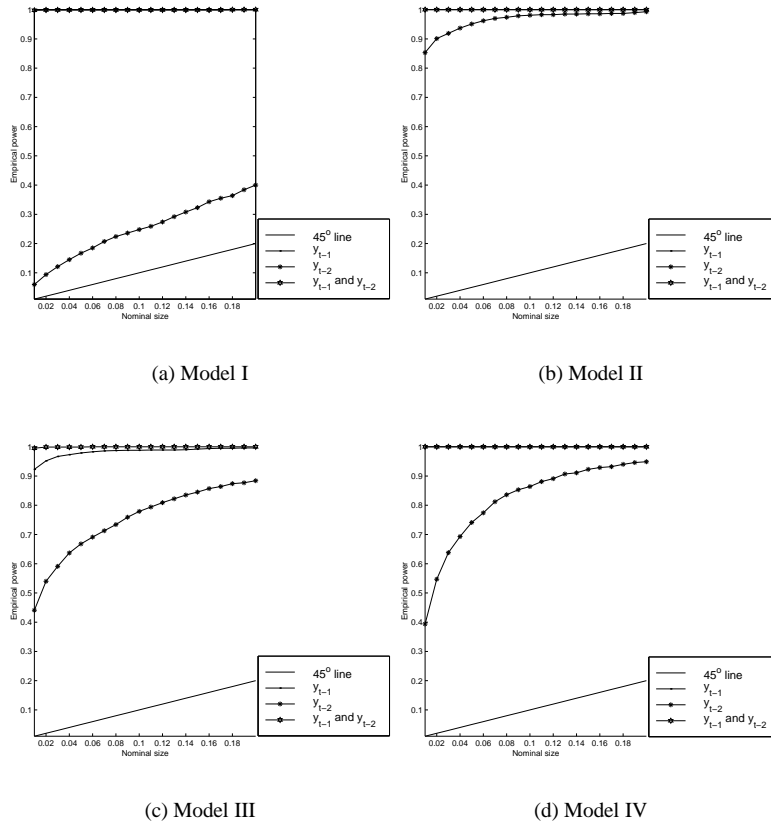
FIGURE 11. Power-size curve of the linearity tests at sample size of 300 observations based on 1000 replications.

TABLE 7. Frequency of of accepting the second hyperplane (over 100 runs). The hyperplanes are estimated with 10 GRASP iterations.

| | 100 observations | | 300 observations | |
Model	SBIC	AIC	SBIC	AIC
II	0	0.8200	0	0.95
III	0.0500	0.9200	0.8600	1
IV	0.3300	1	1	1

frequency (over 100 runs) of selecting the second hyperplane based on 10 GRASP iterations. As we can see, the AIC overfits the model (model II). In that sense we recommend the use of the SBIC to select the number of hyperplanes.

7. EXAMPLES

In this section we present an illustration of the modeling techniques discussed in this paper.

FIGURE 12. Time series generated by model (20)

7.1. **Example 1: Simulated data.** The first time series is generated by the following data generating process.

$$y_t = 1.5 + 0.5y_{t-1} - 0.8y_{t-2} + 0.5y_{t-3} +$$
$$(-4.0 - 0.4y_{t-1} + 0.5y_{t-2} - 0.2y_{t-2})I_{\boldsymbol{\omega}_1,\beta_1}(\mathbf{x}_t) +$$
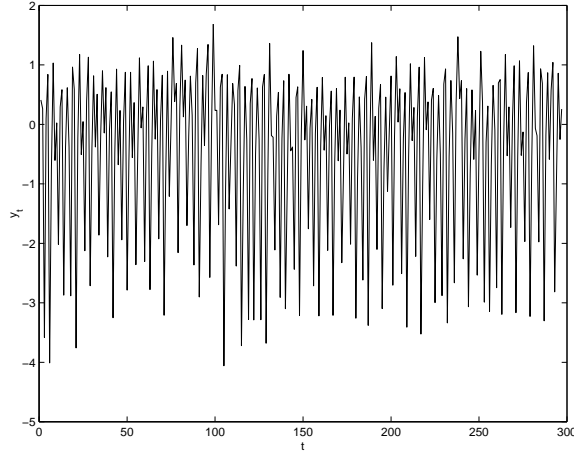$$(3 + 0.2y_{t-1} + 0.2y_{t-2} + 0.1y_{t-3})I_{\boldsymbol{\omega}_2,\beta_2}(\mathbf{x}_t) + \varepsilon_t,$$
$$\varepsilon_t \sim \text{NID}(0, 0.25^2), \quad (20)$$

where $\mathbf{x}_t = [y_{t-1}, y_{t-2}]$, $\boldsymbol{\omega}_1 = [1, -1]'$, $\boldsymbol{\omega}_2 = [1, -1]'$, $\beta_1 = -1$, and $\beta_2 = 1$. The time series is illustrated in Figure 12.

To estimate the parameters we ran the GRASP described in Section 5 with 30 iterations and using 50 sample points to generate the initial hyperplanes. The estimated residual standard deviation is $\hat{\sigma} = 0.251$, $\hat{\boldsymbol{\omega}}_1 = [1, -1.185]'$, $\hat{\boldsymbol{\omega}}_2 = [1, -1.069]'$, $\hat{\beta}_1 = -1.114$, and $\hat{\beta}_2 = 1.022$. Figure 13 shows the scatter plot of the transition variables, the true (dashed lines) and estimated (solid lines) hyperplanes. As we can see, the algorithm has successfully estimated the separating hyperplanes.

7.2. **Example 2: Simulated data.** The second time series is generated by the following data generating process.

$$y_t = -0.2 + 0.5y_{t-1} - 0.8y_{t-2} + 0.5y_{t-3} +$$
$$(0.8 - 0.4y_{t-1} + 0.5y_{t-2} - 0.2y_{t-3})I_{\boldsymbol{\omega}_1,\beta_1}(\mathbf{x}_t) +$$
$$(-0.4 + 0.2y_{t-1} + 0.2y_{t-2} + 0.1y_{t-3})I_{\boldsymbol{\omega}_2,\beta_2}(\mathbf{x}_t) + \varepsilon_t,$$
$$\varepsilon_t \sim \text{NID}(0, 0.25^2), \quad (21)$$

where $\mathbf{x}_t = [y_{t-1}, y_{t-2}]$, $\boldsymbol{\omega}_1 = [1, 0]'$, $\boldsymbol{\omega}_2 = [1, -1]'$, $\beta_1 = 0$, and $\beta_2 = 0$. The time series is illustrated in Figure 14.

The estimated residual standard deviation is $\hat{\sigma} = 0.25$, $\hat{\boldsymbol{\omega}}_1 = [1, 0.015]'$, $\hat{\boldsymbol{\omega}}_2 = [1, -1.037]'$, $\hat{\beta}_1 = 0.013$, and $\hat{\beta}_2 = 0.015$. Figure 15 shows the scatter plot of the transition variables, the true (dashed lines) and estimated (solid lines) hyperplanes. The algorithm has successfully estimated the hyperplanes.
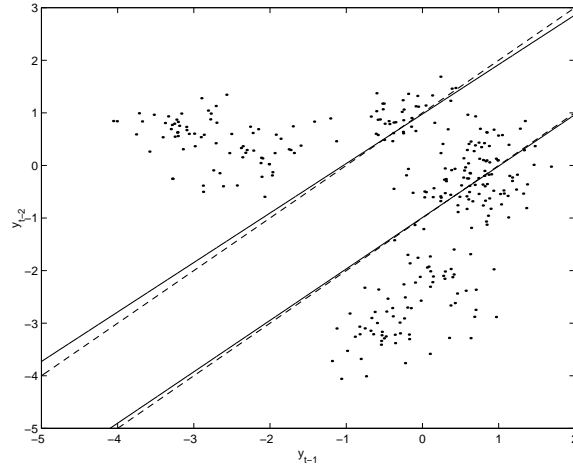
FIGURE 13. Estimated hyperplanes
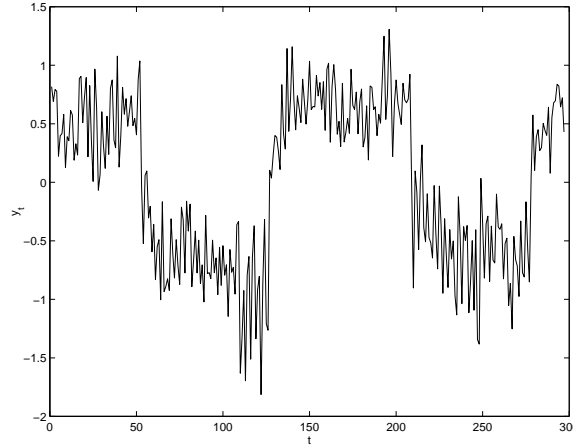
FIGURE 14. Time series generated by model (21)

7.3. **Example 3: Annual sunspot numbers.** In this example we consider the annual sunspot numbers over the period 1700–1998. The observations for the period 1700–1979 were used to estimate the model and the remaining were used to forecast evaluation. We adopted the same transformation as in Tong (1990), $y_t = 2\left[\sqrt{(1+N_t)} - 1\right]$, where N_t is the sunspot number. We selected lags 1, 2, and 9 using SBIC, among the first 12 lags. Linearity was rejected and the p-value of the linearity test was minimized (0.0044) with lags 1 and 2 as threshold variables.

The final estimated model is

$$
\begin{aligned}
y_t = &-\underset{(0.504)}{0.727} + \underset{(0.105)}{0.829}y_{t-1} - \underset{(0.106)}{0.039}y_{t-2} + \underset{(0.030)}{0.162}y_{t-9} \\
&+ \left(\underset{(0.874)}{3.605} + \underset{(0.140)}{0.267}y_{t-1} - \underset{(0.146)}{0.451}y_{t-2} + \underset{(0.054)}{0.035}y_{t-9}\right) \times I_{\hat{\boldsymbol{\omega}},\hat{\beta}}(\mathbf{x}_t) + \hat{\varepsilon}_t,
\end{aligned}
\tag{22}
$$

FIGURE 15. Estimated hyperplanes

where $\mathbf{x}_t = [y_{t-1}, y_{t-2}]'$, $\hat{\boldsymbol{\omega}} = [1, -1.021]'$, and $\hat{\beta} = 0.216$. The estimated in-sample residual standard deviation is $\hat{\sigma}_\varepsilon = 1.916$. It is important to notice that the estimated linear combination of threshold variables is almost the first difference of y_{t-1}.

We continue considering the out-of-sample performance of the estimated model. We compare our results with the ones obtained by the SETAR model fitted by Tong (1990,p. 420) and the model estimated by Chen (1995), a Threshold Autoregression System with Open-loop (TARSO) (Tong, 1990,p. 101) where the threshold variable is a nonlinear function of lagged values of the time series. The estimated model is

$$
y_t = \begin{cases}
\underset{(0.866)}{0.490} + \underset{(0.098)}{1.453}\,y_{t-1} - \underset{(0.165)}{0.790}\,y_{t-2} + \underset{(0.176)}{0.300}\,y_{t-3} - \underset{(0.125)}{0.150}\,y_{t-4} + \\
\qquad\qquad\qquad\qquad\qquad\qquad \underset{(0.056)}{0.217}\,y_{t-8}, \text{if } q_t < 0; \\
\underset{(0.649)}{0.133} + \underset{(0.060)}{1.010}\,y_{t-1} - \underset{(0.068)}{0.255}\,y_{t-2} + \underset{(0.061)}{0.036}\,y_{t-7} - \underset{(0.097)}{0.158}\,y_{t-8} + \\
\qquad\qquad\qquad\qquad\qquad\qquad \underset{(0.066)}{0.295}\,y_{t-9}, \text{otherwise,}
\end{cases} \tag{23}
$$

where $q_t = (y_{t-1} - 10)^2 - 10y_{t-3} - 113$.

Table 8 shows the results of the one-step ahead forecast computed by the SETAR model estimated in Tong (1990,p. 420), the TARSO fitted by Chen (1995), and model (22). The table shows the one-step ahead forecasts, their root mean square errors, and mean absolute errors for the transformed annual number of sunspots for the period 1980-1998. Both the root mean squared errors (RMSE) and the mean absolute errors (MAE) of the SEMTAR with variables selected by SBIC are lower than the ones of the SETAR and TARSO specifications.

8. CONCLUSIONS

In this paper we consider a generalization of the SETAR model to deal with a flexible specification of the threshold variables. We propose a heuristic to estimate SETAR models with thresholds defined by more than one variable. A model specification procedure based on statistical inference is developed and the results of a simulation experiment showed that the proposed methodology works well. A GRASP has been developed to estimate the

TABLE 8. One-step ahead forecasts, their root mean square errors, and mean absolute errors for the annual number of sunspots for the period 1980-1998.

Year	Observation	SETAR Model		TARSO model		SEMTAR Model	
		Forecast	Error	Forecast	Error	Forecast	Error
1980	154.6	160.96	-6.36	134.33	20.27	148.40	6.20
1981	140.4	137.21	3.19	125.39	15.01	117.75	22.65
1982	115.9	99.04	16.86	99.30	16.60	100.45	15.45
1983	66.6	75.96	-9.36	85.03	-18.43	82.78	-16.18
1984	45.9	35.66	10.24	41.16	4.74	44.42	1.47
1985	17.9	24.22	-6.32	29.82	-11.92	30.82	-12.92
1986	13.4	10.72	2.68	9.76	3.64	14.11	-0.72
1987	29.4	20.11	9.29	16.54	12.86	16.89	12.50
1988	100.2	54.49	45.71	66.44	33.76	67.57	32.63
1989	157.6	155.72	1.88	121.84	35.76	153.13	4.46
1990	142.6	156.39	-13.78	152.47	-9.87	164.12	-21.51
1991	145.7	93.25	52.44	123.71	21.99	117.47	28.23
1992	94.3	111.27	-16.97	115.98	-21.68	111.25	-16.95
1993	54.6	67.77	-13.17	69.22	-14.62	69.87	-15.27
1994	29.9	27.03	2.87	35.74	-5.84	37.34	-7.44
1995	17.5	18.36	-0.87	18.91	-1.41	20.33	-2.83
1996	8.6	18.04	-9.44	11.64	-3.04	14.43	-5.83
1997	21.5	12.31	9.17	11.82	9.68	12.82	8.68
1998	64.3	46.70	17.60	58.54	5.76	59.76	4.54
RMSE			18.71		16.94		15.28
MAE			13.06		14.05		12.45

parameters of the model. Both the simulation study and the real examples suggest that the theory developed here is useful and the proposed model thus seems to be a useful tool for the time series practitioner. Finally, the results presented here can be easily generalized into a regression framework with exogenous variables.

ACKNOWLEDGMENTS

REFERENCES

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.

G. E. P. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis, Forecasting and Control*. Prentice Hall, San Francisco, third edition, 1994.

R. Chen. Threshold variable selection in open-loop threshold autoregressive models. *Journal of Time Series Analysis*, 16(5):461–481, 1995.

T.A. Feo and M.G.C. Resende. A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8:67–71, 1989.

T.A. Feo and M.G.C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

J. H. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19(1):1–142, 1991.

C. W. J. Granger and T. Teräsvirta. *Modelling Nonlinear Economic Relationships*. Oxford University Press, Oxford, 1993.

C. M. Kuan and H. White. Artificial neural networks: An econometric perspective. *Econometric Reviews*, 13:1–91, 1994.

F. Leisch, A. Trapletti, and K. Hornik. Stationarity and stability of autoregressive neural network processes. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11. MIT Press, USA, 1999.

P. A. W. Lewis and J. G. Stevens. Nonlinear modeling of time series using multivariate adaptive regression splines. *Journal of the American Statistical Association*, 86:864–877, 1991.

M.G.C. Resende. Greedy randomized adaptive search procedures (GRASP). In C. Floudas and P.M. Pardalos, editors, *Encyclopedia of Optimization*. Kluwer Academic Publishers, 1999. To appear.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

G. C. Tiao and R. S. Tsay. Some advances in non-linear and adaptive modelling in time-series. *Journal of Forecasting*, 13:109–131, 1994.

H. Tong. On a threshold model. In C. H. Chen, editor, *Pattern Recognition and Signal Processing*, Amsterdam, 1978. Sijthoff and Noordhoff.

H. Tong. *Threshold Models in Non-linear Time Series Analysis*, volume 21 of *Lecture Notes in Statistics*. Springer-Verlag, Heidelberg, 1983.

H. Tong. *Non-linear Time Series: A Dynamical Systems Approach*, volume 6 of *Oxford Statistical Science Series*. Oxford University Press, Oxford, 1990.

H. Tong and K.S. Lim. Threshold autoregression, limit cycles and cyclical data (with discussion). *Journal of the Royal Statistical Society, Series B*, 42:245–292, 1980.

R.S. Tsay. Testing and modeling threshold autoregressive processes. *Journal of the American Statistical Association*, 84:431–452, 1989.

Dick van Dijk, Timo Teräsvirta, and Philip Hans Franses. Smooth transition autoregressive models - a survey of recent developments. Working Paper Series in Economics and Finance 380, Stockholm School of Economics, 2000.

G. Zhang, B.E. Patuwo, and M.Y. Hu. Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, 14:35–62, 1998.

(M. C. Medeiros) DEPARTMENT OF ECONOMICS, CATHOLIC UNIVERSITY OF RIO DE JANEIRO, RIO DE JANEIRO, RJ, BRAZIL.
E-mail address: mcm@econ.puc-rio.br

(A. Veiga) DEPARTMENT OF ELECTRICAL ENGINEERING, CATHOLIC UNIVERSITY OF RIO DE JANEIRO, RIO DE JANEIRO, RJ, BRAZIL.
E-mail address: alvf@ele.puc-rio.br

(M. G. C. Resende) INFORMATION SCIENCES RESEARCH CENTER, AT&T LABS RESEARCH, FLORHAM PARK, NJ 07932 USA.
E-mail address: mgcr@research.att.com