LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

INSTITUT FÜR STATISTIK

Nivien Shafik & Gerhard Tutz

Boosting Nonlinear Additive Autoregressive Time Series

Technical Report Number 006, 2007
Department of Statistics
University of Munich

http://www.stat.uni-muenchen.de

Boosting Nonlinear Additive Autoregressive Time Series

Nivien Shafik & Gerhard Tutz
Ludwig-Maximilians-Universität München
Akademiestraße 1, 80799 München

2nd December 2007

Abstract

Within the last years several methods for the analysis of nonlinear autoregressive time series have been proposed. As in linear autoregressive models main problems are model identification, estimation and prediction. A boosting method is proposed that performs model identification and estimation simultaneously within the framework of nonlinear autoregressive time series. The method allows to select influential terms from a large numbers of potential lags and exogenous variables. The influence of the selected terms is modelled by an expansion in basis function allowing for a flexible additive form of the predictor. The approach is very competitive in particular in high dimensional settings where alternative fitting methods fail. This is demonstrated by means of simulations and two applications to real world data.

Key words: Nonlinear time series, semi-parametric model, splines, lag selection, variable selection, boosting

1 Introduction

There is a large variety of nonlinear time series, and even when one restricts consideration to autoregressive models, there will remain many, such as threshold-, functional coefficient- and additive models. Threshold processes can be understood as a special case of functional coefficient (FCAR) models and functional coefficient processes can be interpreted as a special case of additive models. In the following we will consider the nonlinear additive autoregressive (NAARX) model with exogenous variables, which is the most general among these.
In time series analysis with the focus on autoregressive models one faces in particular three problems: model identification, i.e. lag selection, model estimation and prediction. Many methods have been proposed to cope with these problems. In

the following an overview over methods that have been developed more recently is given. A nonparametric method for conditional heteroscedastic autoregressive time series was proposed by Tschernig & Yang (2000). Lag selection was done via a nonparametric version of the final prediction error, whose asymptotic analog is then penalized to reduce overfitting. They gave two alternative estimators for the final prediction error: one is based on a local linear and the other on the Nadaraya-Watson estimate. A problem with their approach is that it can suffer from the curse of dimensionality. The method proposed here avoids that problem by fitting an additive structure that is approximated by an expansion in basis functions.

Chen & Tsay (1993) used the backfitting procedures ACE and BRUTO to identify the significant lags and the method of best subset regression to determine the final model. BRUTO is based on a modified general cross validation (GCV) criterion (see Hastie (1989)). Lewis & Stevens (1991) propose multivariate adaptive regression splines (MARS) for fitting threshold autoregressive functions. MARS uses a basis expansion with products of univariate splines (see Friedman (1991)). Model selection in the forward step is done via the residual squared error, in the backward step a modified GCV criterion is used. MARS has been extended to the multivariate case (POLYMARS) and applied to general nonlinear time series by De Gooijer & Ray (2003).

Polynomial splines were investigated by Huang & Shen (2004), where a B-spline basis is used for the estimation of the regression function of a functional coefficient model. The selection of the significant lags is done via a forward and a backward selection step by adding and deleting that component that minimizes the mean squared error. From these candidate sets they choose the one that minimizes a criterion like Akaike's information criterion (AIC; see Akaike (1974)), the corrected version of the AIC, the AIC_{corr} (see Hurvich & Tsai (1989)), or the Bayes information criterion (BIC), also known as Schwarz's information criterion (see Schwarz (1978)). The same implementation was used by Huang & Yang (2004) for NAARX processes. They used additive spline fitting and the BIC to estimate the conditional mean.

In the present paper we propose boosting procedures for the fitting of NAARX processes. Boosting was originally developed for classification purposes, but was adapted to regression problems later (for the original boosting algorithm AdaBoost see Freund & Schapire (1996); for AdaBoost and regression see Friedman, Hastie & Tibshirani (2000)). An overview on boosting methods is given in Bühlmann & Hothorn (2007). Boosting is an ensemble method that uses a combination of "rule of thumbs", i.e. weak learners, in order to minimize the expected loss (for an overview see Freund & Schapire (1999)). Boosting algorithms differ mainly in their choice of the loss function and the weak learner. Bühlmann & Yu (2003) proposed a boosting procedure based on squared error loss (hence the name L_2Boost), while AdaBoost was minimizing exponential loss. Sometimes sparser solutions than the ones resulting from L_2Boost are needed. This implies a penalized loss function, for example the well known model selection criteria AIC_{corr} and BIC can be applied (for SparseL_2Boost see Bühlmann & Yu (2006)). An alternative to SparseL_2Boost

is TwinBoosting from Bühlmann (2007) where the boosting algorithm is done twice. In the first run it selects some candidate variables, while the second one is only done with the preselected variables from the first run. Besides computational inefficiency TwinBoosting has the disadvantage of a more complicated hat matrix, which makes it difficult to calculate the model selection criteria, where the trace of the hat matrix is needed.
Boosting approaches to the fitting of time series have been given recently by Lutz & Bühlmann (2006). The authors restricted consideration to ordinary linear (vector-) autoregressive processes, which are fitted within a linear model framework. For the NAARX processes considered here nonlinear fitting procedures are needed. Therefore we propose to use a spline basis expansion in each smooth component within an additive modeling approach.
The paper is organized as follows. In section 2 we introduce the model for the NAARX process as used in this paper. estimation procedure via penalized least squares and the boosting algorithm is given in section 3. Simulated data are examined in section 4, where the results are judged by mean squared model error (MSE), one-step ahead prediction error (OSPE) and hit- versus false alarm rate. In section 5 two real data example are given. In the second example exogenous variables are included.

2 The Model

Let a time series (Y_t, X_t), be generated by a nonlinear additive autoregressive model of order p with exogenous variables (NAARX(p)), i.e.

$$\begin{aligned} Y_t &= f_1(X_{t1}) + \cdots + f_p(X_{tp}) + \epsilon_t \\ &= \sum_{i=1}^{p} f_i(X_{ti}) + \epsilon_t, \qquad t \in \mathbb{N}. \end{aligned}$$

The variable $X_t = (X_{t1}, \ldots, X_{tp})^\top$ consists of exogenous and/or lagged values of the time series, the error term ϵ_t is i.i.d. with $\mathbb{E}(|\epsilon_t|) < \infty$, and ϵ_1 admits a positive and continuous probability density function. The functions f_i denote mappings from $\mathbb{R}$ into $\mathbb{R}$ for all $i = 1, \ldots, p$. Furthermore, we will assume that each of these functions f_i may be expanded in m basis functions ϕ_{ij}. Hence f_i can be written as

$$f_i(x) = \sum_{j=1}^{m} \alpha_{ij}\phi_{ij}(x) \tag{1}$$

with coefficients α_{ij}. Then the time series Y_t has the form

$$Y_t = \sum_{i=1}^{p}\sum_{j=1}^{m} \alpha_{ij}\phi_{ij}(X_{ti}) + \epsilon_t.$$

Estimating the conditional mean of Y_t, $\mu_t = \mathbb{E}(Y_t|X_t = x)$, is equal to estimating $\sum_{i=1}^{p} f_i(X_{ti})$ and therefore the coefficients α_{ij}. In the following it is assumed that the time series is stationary. Conditions are found in Tong (1990).

3 Estimation

Let T points of the univariate time series Y_t be observed. Since the nonlinear functional terms are approximated by an expansion in basis functions, estimation refers to the coefficients α_{ij}. In contrast to Huang & Yang (2004) estimation is based on the penalized least squares method, i.e. the function to be minimized is given by

$$PRSS(f, \lambda) = \sum_{t=1}^{T} \left(Y_t - \sum_{i=1}^{p} f_i(x_{ti}) \right)^2 + \sum_{i=1}^{p} J(f_i, \lambda_i)$$

where $(x_t)_{t=1,\ldots,T}$ represents the data with $x_t = (x_{t1}, \ldots, x_{tp})^\top$ and $\lambda = (\lambda_1, \ldots, \lambda_p)^\top$ the vector of penalties. There are various ways to choose a penalty function $J(f_i, \lambda_i)$. Hastie, Tibshirani & Friedman (2001) use different λ_i for each f_i and the integral of the second derivatives of the functions f_i, i.e.

$$J(f_i, \lambda_i) = \lambda_i \int_{\mathbb{R}} \left(\frac{\partial^2 f_i(t_i)}{\partial t_i^2} \right)^2 dt_i.$$

When using B-splines on a equally spaced grid (see e.g. Eilers & Marx (1996)) a simple rescaled discrete approximation to the upper penalty term is given by

$$J(f_i, \lambda_i) = \lambda_i \sum_{j=k+1}^{m} (\Delta^k \alpha_{ij})^2,$$

where Δ is the difference operator given by $\Delta^1 \alpha_{ij} = \alpha_{ij} - \alpha_{ij-1}$ and $\Delta^k \alpha_{ij} = \Delta^1 \Delta^{k-1} \alpha_{ij}$. In the following, we will choose $k = 2$, which results in $J(f_i, \lambda_i) = \lambda_i \sum_{j=3}^{m} (\alpha_{ij} - 2\alpha_{ij-1} + \alpha_{ij-2})^2$. For $\lambda = 0$ one obtains the ordinary least squares method, for $k = 0$ the method is equivalent to ridge regression. When B-splines are used as basis functions the method is referred to as P-splines for penalized B-Splines. It should be noted that the selection of smoothing parameters λ_i raises severe problems. Since the number of autoregressive terms is unknown one has to investigate varying number of autoregressive terms. That, however, makes several optimization procedures necessary. While optimization with respect to λ_i is easily done for one dimension it is highly problematic for three or more influential terms. The high computational burden usually makes selection impossible. Therefore one often assumes one common smoothing parameter λ with the consequence that varying curvature of the underlying functions f_i is simply ignored. As outlined in the next section, boosting techniques avoids the problem of differing smoothing parameters by updating selected influential terms.

3.1 Simultaneous Fitting and Selection by Boosting Techniques

For given number of autoregressive terms and given smoothing parameters minimization of $PRSS(f, \lambda)$ is straightforward (see Eilers & Marx (1996)). However, the big disadvantage is that no components are selected. The procedure proposed here is componentwise boosting, which means that in an iterative procedure only one component of the additive term is updated within one step. For the fit of one component within the procedure a "weak" learner is used. A weak learner is a fitting procedure, that uses a small number of degrees of freedom. When fitting is steered by the smoothing parameter λ, a weak learner is obtained by choosing λ very large. A nice feature of the iterative boosting procedure is that one global (and large) tuning parameter λ may be used for all of the components. If a component has stronger curvature, then it is chosen more often in the componentwise updating procedure. Moreover, componentwise boosting has a built-in selection device, it automatically selects relevant covariates (see also Tutz & Binder (2006)).
In the following the fitting procedure is described in detail. The initial value of the α_{ij}s is set to zero, which leads to a fitting of the original time series Y_t in the first step, while in the following ones only the residuals $Y_t - \hat{f}^{(k-1)}(x_t)$ are of interest. In each iteration we will start with the fitting of the nonlinear part, which is, due to the basis expansion, equal to adjusting the coefficients α_{ij}. Since in each iteration only one f_{i^*} is fitted, only $(\alpha_{i^*j})_j$ change, while all the other $(\alpha_{i'j})_j$ for $i' \neq i^*$ remain unchanged. The component to be updated, f_{i^*}, is chosen via the minimization of the BIC. The number of the refitting steps for components is denoted by K. The choice of K is considered separately in the next section.
Since the steps within the boosting procedure are based on the updating of one simple component, we consider briefly the fitting for that case. For given smoothing parameter the m-dimensional vector of coefficients for the ith component $\boldsymbol{\alpha}^{(i)}= (\alpha_{i1}, \ldots, \alpha_{im})^\top$ is obtained by solving

$$\left(\Phi^{(i)}\right)^\top \tilde{\mathbf{Y}}_\mathbf{t} = \left(\left(\Phi^{(i)}\right)^\top \Phi^{(i)} + \lambda D_2^\top D_2\right) \hat{\boldsymbol{\alpha}}^{(i)}$$

with $\Phi^{(i)}$ being the $T \times m$-matrix of basis functions of each x_{ti}, i.e.

$$\Phi^{(i)} = \begin{pmatrix} \phi_{i1}(x_{1i}) & \ldots & \phi_{im}(x_{1i}) \\ \vdots & \vdots & \vdots \\ \phi_{i1}(x_{Ti}) & \ldots & \phi_{im}(x_{Ti}) \end{pmatrix},$$

$\tilde{\mathbf{Y}}_\mathbf{t} = (Y_1, \ldots, Y_T)^\top$ and D_2 being the matrix representation of the difference operator Δ^2, given by

$$D_2 = \begin{pmatrix} 1 & -2 & 1 & 0 & \ldots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & 1 & -2 & 1 \end{pmatrix}.$$

Hence the estimate is given by

$$\hat{\boldsymbol{\alpha}}^{(i)} = \left[\left(\Phi^{(i)}\right)^\top \Phi^{(i)} + \lambda D_2^\top D_2 \right]^{-1} \left(\Phi^{(i)}\right)^\top \tilde{\mathbf{Y}}_\mathbf{i}.$$

and the corresponding hat matrix has the form

$$H^{(i)}(\lambda, \hat{\boldsymbol{\alpha}}^{(i)}) = \Phi^{(i)} \left[\left(\Phi^{(i)}\right)^\top \Phi^{(i)} + \lambda D_2^\top D_2 \right]^{-1} \left(\Phi^{(i)}\right)^\top.$$

The boosting procedure for fitting and selection of components has the following form.

Initialization:
Set $\alpha_{ij}^{(0)} = 0$ for $i = 1, \ldots, p$, $j = 1, \ldots, m$, hence $\hat{f}^{(0)} = 0$. Choose K large.

Iteration:
For $k = 1, 2, \ldots, K$

Computation of the current residuals
For $t = 1, \ldots, T$ compute the residuals

$$r_t^{(k)} = Y_t - \hat{f}^{(k-1)}(x_t).$$

Fitting of the base learner
For $i = 1, \ldots, p$ fit the model

$$r_t^{(k)} = \sum_{j=1}^{m} \alpha_{ij} \phi_{ij}(x_{ti}) + \epsilon_t, \qquad t = 1, \ldots, T,$$

by penalized least squares yielding the penalized estimate

$$\hat{\boldsymbol{\alpha}}^{(i)} = \left(\left[\left(\Phi^{(i)}\right)^\top \Phi^{(i)} + \lambda D_2^\top D_2 \right]^{-1} \left(\Phi^{(i)}\right)^\top (r_1^{(k)}, \ldots, r_T^{(k)})^\top \right),$$

where $\left[\left(\Phi^{(i)}\right)^\top \Phi^{(i)} + \lambda D_2^\top D_2 \right]^{-1} \left(\Phi^{(i)}\right)^\top$ is the hat matrix from above.

Selection step
Take that component i^* that minimizes the BIC (see section 3.2 for a formula of the BIC).

Update of the estimator
The estimator can be written in two different ways. The first one is $\hat{f}^{(k)}(x_t) = \hat{f}^{(k-1)}(x_t) + \sum_{j=1}^{m} \hat{\alpha}_{i^*j} \phi_{i^*j}(x_{ti^*})$ and the second one is via the coefficients, i.e.

$$\alpha_{ij}^{(k)} = \begin{cases} \alpha_{ij}^{(k-1)} + \hat{\alpha}_{ij} & \text{for } i = i^* \\ \alpha_{ij}^{(k-1)} & \text{otherwise,} \end{cases}$$

for $j = 1, \ldots, m$, yielding $\hat{f}^{(k)} = \sum_{i=1}^{p} \sum_{j=1}^{m} \alpha_{ij}^{(k)} \phi_{ij}(x_{ti})$.

3.2 Stopping Criterion for the Iterations

Finding a good number of iterations K can be done via many different criteria, e.g. AIC, corrected AIC_{corr} and BIC or via GCV. The last one is not recommended, since it is computational very expensive. We will use the BIC unlike Lutz & Bühlmann (2006), who used AIC_{corr}. Therefore, we need the hat matrix of the smoother, which uses the hat-matrices for one step of the procedure given in section 3.1. Bühlmann & Yu (2003) showed that the L_2Boost hat matrix in the kth boosting iteration is

$$\mathcal{H}_k = I - \prod_{l=1}^{k} (I - H^{(i^*,l)}(\lambda, \boldsymbol{\alpha}^{i^*}))(I - H^{(0)})$$

where $H^{(i^*,l)}(\lambda, \boldsymbol{\alpha}^{i^*})$ denotes the hat matrix $H^{(i^*)}(\lambda, \boldsymbol{\alpha}^{i^*})$ in the lth iteration, where the $(\alpha_{i^*j})_j$s are updated, and $H^{(0)}$ is $\frac{1}{T}\mathbf{1}^\top\mathbf{1}$.
The degrees of freedom are given by the trace of the L_2Boost hat matrix $tr(\mathcal{H}_k)$. Hence the BIC that is used for stopping is given by

$$BIC(k) = \frac{Dev^{(k)}}{\hat{\sigma}^2} + \log(T)(1 + tr(\mathcal{H}_k)),$$

where $\hat{\sigma}^2 = Dev^{(K)}/(T - tr(\mathcal{H}_K))$ is the estimator of the largest model, $\tilde{\mathbf{Y}} = (Y_1, \ldots, Y_T)^\top$ as before and $Dev^{(k)}$ denotes the deviance after the kth iteration. The estimate for the numbers of iterations is that k that minimizes the BIC. Within the selection step the BIC is computed for the components with the hat matrices given by $H^{(i)}(\lambda, \boldsymbol{\alpha}^i)$. One selects that component that has minimal BIC.

4 Simulation Study

4.1 Setup and Implementation

In the following we will consider eight different processes, AR1-AR4 and NLAR1-NLAR4 (see Table 1). The models AR1-AR3 and NLAR1-NLAR3 were also used by Tschernig & Yang (2000) and Huang & Yang (2004) and are stationary processes. AR4 and NLAR4 were added in order to investigate the estimation procedures when more than two significant lags are in the data generating model. To start in the stationarity distribution we generated 400+T observations and discarded the first 400, with T being 100, 200 and 500. Since displaying all results would go beyond the scope of this paper, we restrict presentation to sample size of 100 (or even less), where the methods differed the most. The processes AR1-AR4 are linear autoregressive processes of order two, three or ten and hence are mainly differing in their lag vector. The other processes, NLAR1-NLAR4, are general nonlinear autoregressive processes of order two and seven, as well as functional coefficient processes of order ten (FCAR(10)).

Model	Function
AR1	$Y_t = 0.5Y_{t-1} + 0.4Y_{t-2} + 0.1\epsilon_t$
AR2	$Y_t = -0.5Y_{t-1} + 0.4Y_{t-2} + 0.1\epsilon_t$
AR3	$Y_t = -0.5Y_{t-6} + 0.5Y_{t-10} + 0.1\epsilon_t$
AR4	$Y_t = -0.8Y_{t-1} - 0.4Y_{t-2} + 0.25Y_{t-3} + 0.1\epsilon_t$
NLAR1	$Y_t = -0.4(3 - Y_{t-1}^2)/(1 + Y_{t-1}^2) +$ $0.6\,(3 - (Y_{t-2} - 0.5)^3)\,/\,(1 + (Y_{t-2} - 0.5)^4) + 0.1\epsilon_t$
NLAR2	$Y_t = \left(0.4 - 2\exp(-50Y_{t-6}^2)\right) Y_{t-6} + \left(0.5 - 0.5\exp(-50Y_{t-10}^2)\right) Y_{t-10} +$ $0.1\epsilon_t$
NLAR3	$Y_t = \left(0.4 - 2\cos(40Y_{t-6})\exp(-30Y_{t-6}^2)\right) Y_{t-6} +$ $\left(0.55 - 0.55\sin(40Y_{t-10})\exp(-10Y_{t-10}^2)\right) Y_{t-10} + 0.1\epsilon_t$
NLAR4	$Y_t = 0.9\sin((\pi/8)Y_{t-4}) - 0.75\sin((\pi/8)Y_{t-5}) + 0.52\sin((\pi/8)Y_{t-6}) +$ $0.38\sin((\pi/8)Y_{t-7}) + 0.1\epsilon_t$

Table 1: Specification of processes

We will compare our boosting procedure to MARS, BRUTO and the algorithm of Huang & Yang (2004). The latter used a stepwise procedure based on spline estimation. Hence they had a basis expansion equivalent to (1) but estimated the coefficients via OLS. For variable selection they proposed a procedure consisting of three stages (one forward, one backward and one final selection) where in the first two stages they added or deleted in each step the variable with the smallest mean squared error. In the final selection step they chose from all the candidate models of the forward and backward stage the model with the smallest BIC.
Identification of lags is evaluated by hit- and false alarm rate, which are plotted in a way similar to the ROC curve with hits being the proportion of correctly identified lags and false alarms being the proportion of falsely selected lags over 100 simulation runs. The goodness-of-fit is judged by the MSE

$$MSE = \frac{1}{T}\sum_{t=1}^{T}(\mu_t - \hat{f}(x_t))^2$$

where μ_t is the conditional mean of Y_t. The performance concerning prediction is quantified by the OSPE

$$OSPE = (Y_{t+1} - \hat{f}(x_{t+1}))^2.$$

The implementation is done in `R` (available on http://www.r-project.org/). For all algorithms and processes we allowed to choose from lags one to ten and ran each setting 100 times.

Boosting: For the boosting algorithm we used the R (R Development Core Team,2007) implementation `GAMBoost` which is based on Tutz & Binder (2006),

added the selection via the BIC and the pruning step, used cubic P-Splines whose knots are equally spaced and set the penalty λ equal to 4000.

MARS and BRUTO: MARS and BRUTO are provided in the `mda` package of R. Both procedures have a tuning parameter for the cost per degree of freedom (in the first case it is `cost` and in the second `penalty`). We examined the default two, which is similar to AIC, and $\log(T)$, which corresponds to BIC. Since $\log(T)$ performs ways better we will present only the results of this setting. The parameter `nk` of MARS specifying the maximum number of model terms allowed was erased from the default 21 to 131, hence allowing terms, i.e. products, of length two.

Algorithm of Huang & Yang (2004): For the algorithm of Huang & Yang (2004) we equated `S_max` (the maximal number of variables that are allowed in the model) with `d` (the total number of candidate variables to be selected from), hence just needed the backward stage. Moreover, we chose cubic regression splines and used the `mgcv` package of R for the GAM fitting.

4.2 Results on Lag Selection

For the evaluation of the performance concerning the selection of lags we examine the hit and false alarm rate of each process and each sample size. We added an alternative procedure, referred to as SparseBoost(cut), which uses a pruning step at the end of the algorithm. More specific, the pruning step that has been applied computes the final estimate by

$$\alpha_{ij} = \begin{cases} \alpha_{ij} & \text{if} \quad \frac{\sum_{j=1}^{m} |\alpha_{ij}|}{\sum_{i=1}^{p} \sum_{j=1}^{m} |\alpha_{ij}|} \geq \frac{1}{\mathtt{d}} \\ 0 & \text{otherwise.} \end{cases}$$

Figure 1 shows the hit and false alarm rates for sample size 100 and processes AR3, AR4, NLAR3 and NLAR4, since they are the most complex ones. The more the dots are in the upper left corner (this means having a high hit and a low false alarm rate) the better the performance of the selection procedure. The procedures are denoted by SparseBoost for the boosting procedure and SparseBoost(cut) for the the boosting procedure with a pruning step. It is seen that the pruning step reduces the false alarm rate slightly. Comparison to the competitors shows that the boosting procedure is never dominated by one of the alternative procedures, where dominance means that performance is better in both dimensions, hit and false alarm rate. MARS does very poor in all settings, having a low hit and a high false alarm rate.

4.3 Results on Estimation and Prediction

Besides identifying the relevant lags, the accuracy of estimation and prediction plays an important role in modeling time series. Therefore, we compare the MSE and the

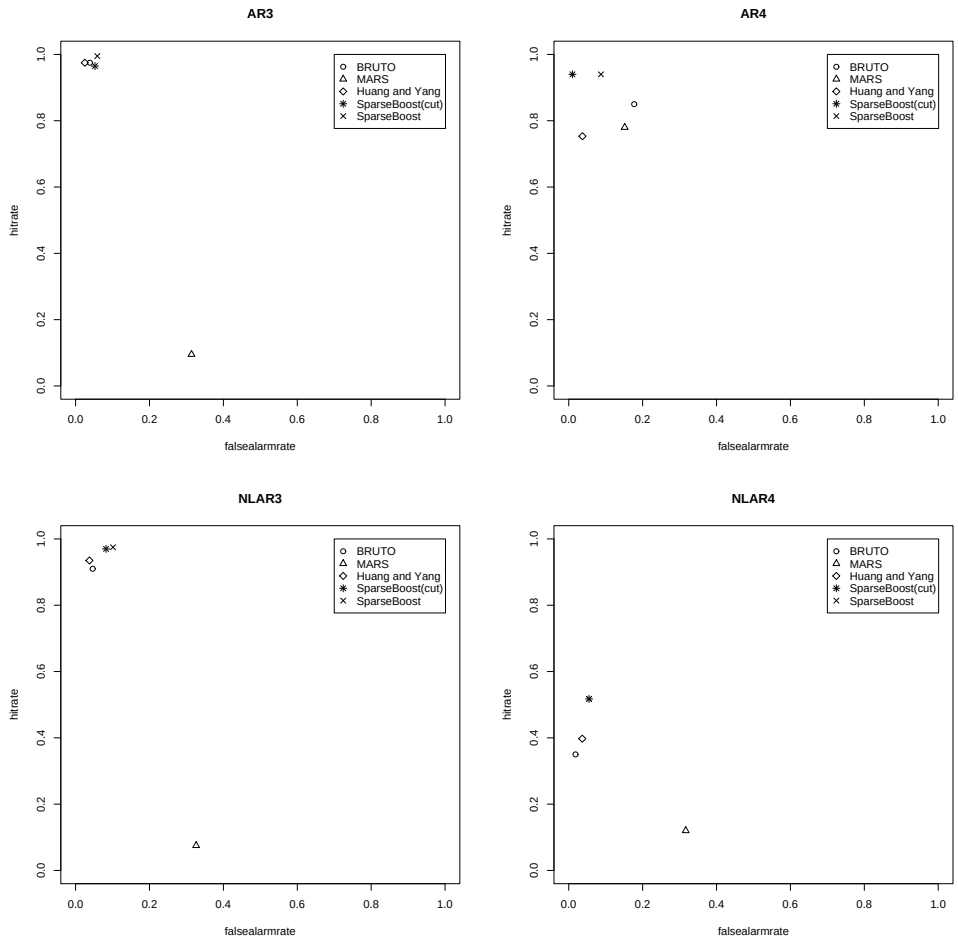

Figure 1: Hit versus false alarm rate for processes AR3, AR4, NLAR3 and NLAR4

OSPE of the proposed estimators. Table 2 shows the mean MSE for all procedures with sample size 100. The best performer is given in boldface. It is seen that SparseBoost is the best performer in five of the eight considered time series, although it should be noted that the differences between methods are small.

In the box plots of the prediction error given in Figure 2 the algorithm of Huang & Yang (2004) is denoted by HaY. Here we again display only the results of the processes AR3, AR4, NLAR3 and NLAR4 and sample size 100. Given the variability of the performance the difference in median is rather weak. In particular MARS shows large variability for linear processes but small variability for nonlinear processes. The number of outliers for the procedures does not vary strongly across procedures.

Model	BRUTO	MARS	Huang and Yang	SparseBoost	data
AR1	2.69	2.69	2.69	**2.64**	0.87
AR2	2.94	2.94	2.95	**2.86**	0.01
AR3	2.61	2.63	2.65	**2.59**	0.01
AR4	4.90	4.91	4.90	**4.72**	0.00
NLAR1	**42.05**	43.55	42.75	42.47	24.64
NLAR2	**0.91**	0.98	0.97	0.96	0.68
NLAR3	6.69	6.80	6.78	**6.68**	4.14
NLAR4	**0.39**	0.46	0.44	0.46	0.02

Table 2: Mean MSE for all processes (in percent)

5 Applications

In the following two univariate real data time series are modeled as nonlinear autoregressive processes. Both are concerned with modeling unemployment data. The first application does not use exogenous variables, while the second includes exogenous variables. The data sets were also used by Stoffer & Shumway (2006), the second was further examined by Young & Pedregal (1999). Stoffer & Shumway (2006) used the latter data set for analysis in the frequency domain, while Young & Pedregal (1999) treated it in the time domain. Both data sets may be downloaded from the homepage of the book by Stoffer & Shumway (2006)(`unemp.dat`, `econ5.dat`). The first consists of 372 monthly measures of the Federal Reserve Board unemployment index, while the second consists of 161 quarterly measures of the unemployment rate, along with observations of US GNP, consumption, governmental spending and private investment. The seasonal component has been removed from both data sets. To compare the fitting of the different algorithms we will consider GCV

$$GCV = \frac{1}{T}\sum_{t=1}^{T}\left(\frac{Y_t - \hat{f}(x_t)}{1 - \frac{1}{T}df}\right)^2$$

where df are the degrees of freedom. The prediction power is evaluated by the iterated mean squared prediction error (MSPE)

$$MSPE = \frac{1}{20}\sum_{t=T+1}^{T+20}\left(Y_t - \hat{f}(x_t)\right)^2.$$

Besides BRUTO, MARS, the algorithm of Huang & Yang (2004) and SparseBoost, we will examine the boosting procedure with a pruning step, i.e. SparseBoost(cut).

5.1 Unemployment Index

In the univariate case without exogenous variables only the Federal Reserve Board unemployment index (and ten lags thereof) is modelled. As the time series is more

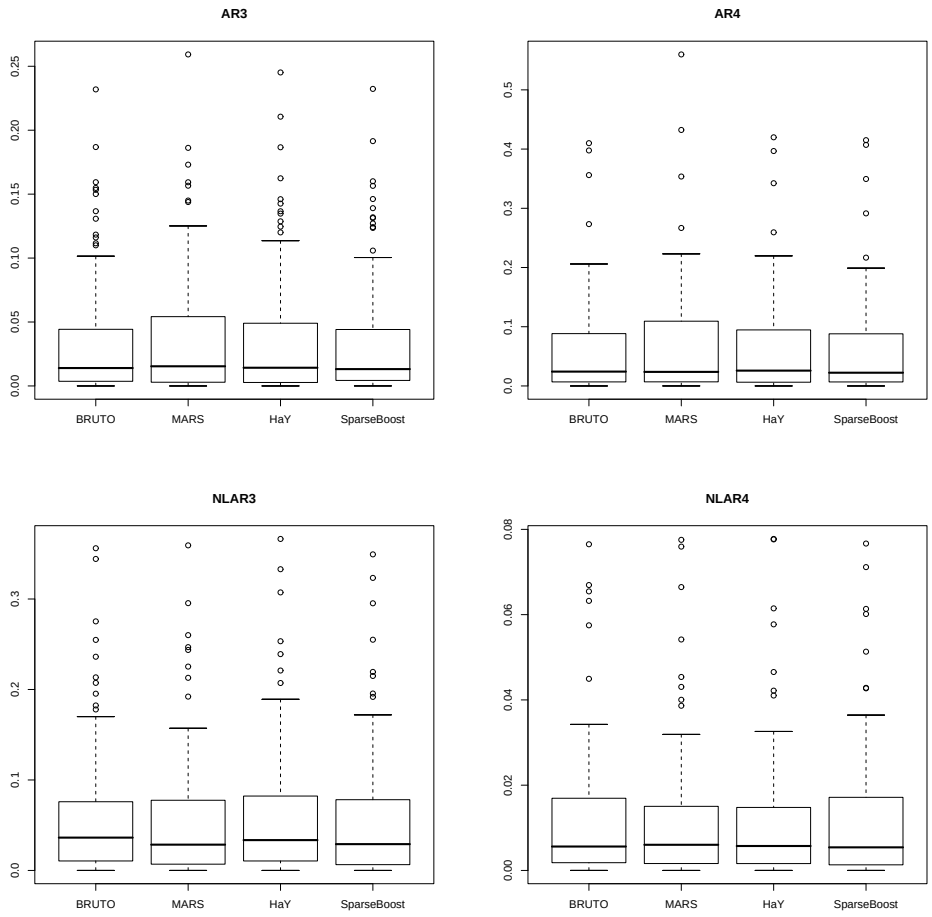

Figure 2: Boxplot of the OSPE for processes AR3, AR4, NLAR3 and NLAR4

wiggly in the first past, we will examine only the last part of it. To get a sample size between 100 and 200, we will consider 131 observations, from the 200th to the 330th observation. Additional 20 observations are used for prediction. Table 3 shows the lags that were selected as well as GCV and MSPE of each procedure. In terms of GCV and MSPE the boosting methods do best. MARS selected almost all of the possible ten lags, but omitted lag seven, which was selected by all the others and seems not to be neglectable (see also Figure 4).
In Figure 3 the 32 last observations which are used for fitting, i.e. from 100th to the 131th of the chosen observations, are given together with the fitted values. When comparing the approximations of Huang & Yang (2004) and SparseBoost one sees that in particular in the last observations the method of Huang & Yang tends to predict a rather flat curve while SparseBoost varies in accordance with the data.
In addition the autocorrelation function (ACF) of the time series and the estimated functions are given in Figure 4. The values of the autocorrelation functions are given

criterion	BRUTO	MARS	Huang and Yang
selected variables	5, 7, 10	1, 2, 3, 4, 5, 6, 9, 10	5, 7, 8, 9, 10
GCV*100	70.06	81.66	66.89
MSPE	362,550.38	362,222.63	369,364.75

criterion	SparseBoost(cut)	SparseBoost
selected variables	2, 5, 7, 8, 10	2, 5, 7, 8, 10
GCV*100	0.03	**0.02**
MSPE	**344,264.75**	354,846.54

Table 3: Selected variables, GCV and MSPE for all methods

in pairs. The first one is the estimate based on the time series, the second one is the autocorrelation function of the fitted model. One can see that SparseBoost and SparseBoost(cut) are the only ones that show the same sign as the time series for every lag.

5.2 Unemployment Rate with Exogenous Variables

To get a stationary time series Young & Pedregal (1999) advice to analyze relatively measured time series, for example governmental spending/US GNP. The unemployment rate as considered here is already a relative measure. We consider the same model as Young & Pedregal (1999) who built a model for the unemployment rate depending on the unemployment rate, governmental spending/US GNP and private investment/US GNP. In their data-based mechanistic modelling approach they use a special form of the recursive least squares method and fixed interval smoothing for parameter estimation.

We allow ten lags for the unemployment rate as well as for governmental spending/US GNP and private investment/US GNP. For this setting the algorithm of Huang & Yang (2004) would have to deal with 1,073,741,824 different possible models and for every one of them a GAM has to be fitted and the mean squared error calculated! Therefore the algorithm of Huang & Yang (2004) fails and is omitted in the comparison.

We will examine the last 150 observations, 20 of these are used for prediction. Variables one to ten characterize the lags of the unemployment rate, eleven to 21 characterize governmental spending/US GNP and its ten lags, and 22 to 32 characterize private investment/US GNP and its ten lags. Table 4 shows the selected variables, GCV and MSPE. BRUTO failed completely. SparseBoost methods show the smallest GCV and both boosting methods perform best in terms of MSPE.

As far as the numbers of selected lags are concerned SparseBoost and SparseBoost(cut) differ strongly. While SparseBoost selected six variables, SparseBoost(cut) just chose two of them. Figure 5 shows the estimates of the six smooth components selected by SparseBoost. It is seen that the lags one and 23 are indeed far away

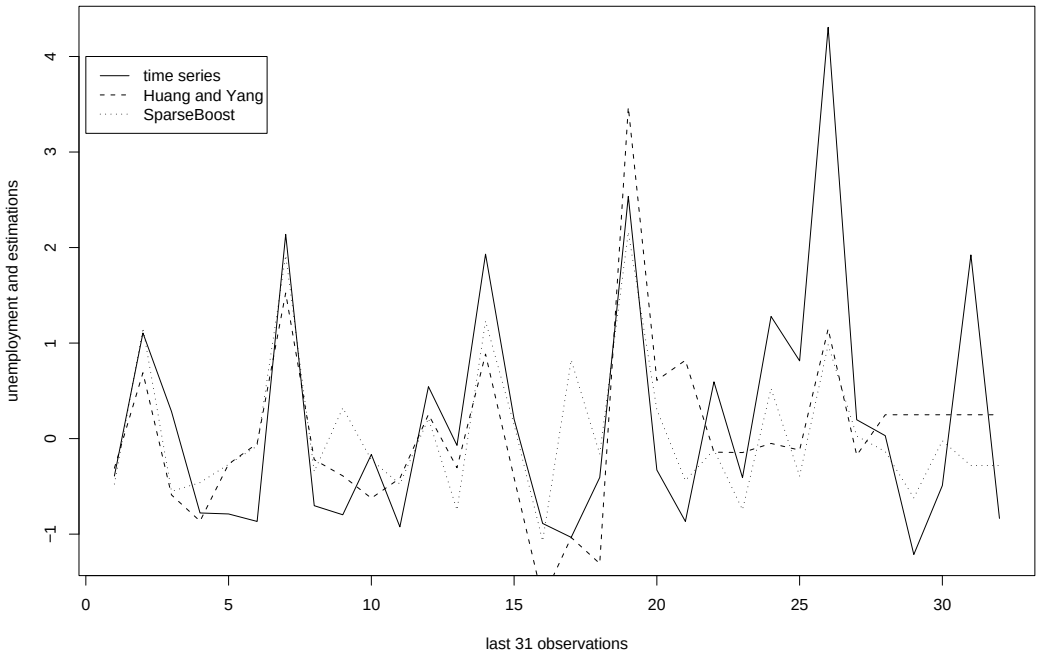

Figure 3: Plot of the end of the unemployment data time series together with the estimates of Huang and Yang and SparseBoost

from zero. The influence of lag one seems to be nearly linear, a conclusion that has also been found by Young & Pedregal (1999).

6 Conclusion

Due to the penalized estimation procedure boosting performs better than the algorithm of Huang & Yang (2004) at the boundary of the data. Generally speaking, boosting is especially made for complex settings. This can be seen from Figure 1 where the hit- and false alarm rates are displayed and boosting is ways better than the other methods in the more complex settings, i.e. the processes with more than two significant variables (AR4 and NLAR4). Here in particular MARS did a very poor job. Also in the application to data sets boosting performs better in the high dimensional setting with exogenous variables, where BRUTO and the algorithm of Huang & Yang (2004) failed completely.

criterion	BRUTO	MARS
selected variables	5, 9, 16, 22, 28	1, 2, 4, 5, 6, 9, 10
GCV*100	$9.15*10^{40}$	18.20
MSPE	$6.49*10^{78}$	4.12

criterion	SparseBoost(cut)	SparseBoost
selected variables	1, 23	1, 15, 22, 23, 24, 25
GCV*100	1.87	**0.00**
MSPE	155.38	**3.96**

Table 4: Selected variables, GCV and MSPE for BRUTO, MARS and the boosting algorithms

References

Akaike, H. (1974). A new look at statistical model identification. *IEEE Transactions on Automatic Control*.

Bühlmann, P. (2007). Twin boosting: Improved feature selection and prediction. *TAS* **34**, 559–583.

Bühlmann, P. and Hothorn, T. (2007). Boosting algorithms: regularization, prediction and model fitting (with discussion). To appear in Statistical Science.

Bühlmann, P. and Yu, B. (2003). Boosting with the L2 loss: Regression and classification. *Journal of the American Statistical Association* **98**, 324–339.

Bühlmann, P. and Yu, B. (2006). Sparse boosting. *Journal of machine learning research* **7**, 1001–1024.

Chen, R. and Tsay, R. S. (1993). Nonlinear additive arx-models. *JASA* **88**, 955–967.

De Gooijer, J. G. and Ray, B. K. (2003). Modeling vector nonlinear time series using polymars. *Computational Statistics and Data analysis* **42**, 73–90.

Eilers, P. H. C. and Marx, B. D. (1996). Flexible smoothing with B-splines and Penalties. *Statistical Science* **11**, 89–121.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. *Proceedings of the Thirteenth International Conference on Medicine Learning. Morgan Kaufmann, San Francisco*, 148–156.

Freund, Y. and Schapire, R. E. (1999). A short introduction to boosting. *journal of japanese society for artificial intelligence* **14**, 771–780.

Friedman, J. (1991). Multivariate adaptive regression splines. *TAS* **19**, 1–67.

Friedman, J. H., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting. *Ann. Statist.* **28**, 337–407.

Hastie, T. (1989). Flexible parsimonious smoothing and additive modeling: Discussion. *Technometrics* **31**, 23–29.

Hastie, T., Tibshirani, R., and Friedman, J. H. (2001). *The Elements of Statistical Learning* (2nd ed.). Springer-Verlag, New York, USA.

Huang, J. Z. and Shen, H. (2004). Functional-coefficient regression models for non-linear time series: A polynomial spline approach. *Scandinavian Journal of Statistics* **31**, 515–534.

Huang, J. Z. and Yang, L. (2004). Identification of non-linear additive autoregressive models. *JRSS* **B 66**, 463–477.

Hurvich, C. M. and Tsai, C.-L. (1989). Regression and time series model selection in small samples. *BMA* **76**, 297–307.

Lewis, P. and Stevens, J. (1991). Nonlinear modeling of time series using multivariate adaptive regression splines (mars). *JASA* **86**, 864–877.

Lutz, R. W. and Bühlmann, P. (2006). Boosting for high-multivariate responses in high-dimensional linear regression. *Statistica Sinica* **16**, 471–494.

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

Stoffer, D. S. and Shumway, R. H. (2006). *Time Series Analysis and its Applications. With R examples* (2nd ed.). New York: Springer.

Tong, H. (1990). *Nonlinear Time Series. A dynamical system approach*. Oxford: Oxford University Press.

Tschernig, R. and Yang, L. (2000). Nonparametric lag selection for time series. *JTSA* **21**, 457–487.

Tutz, G. and Binder, H. (2006). Generalized additive modeling with implicit variable selection by likelihood-based boosting. *Biometrics* **62**, 961–971.

Young, P. and Pedregal, D. (1999). Macro-economic relativity: government spending, private investment and unemployment in the usa 1948-1998. *Journal of structural change and economic dynamics* **10**, 359–380.

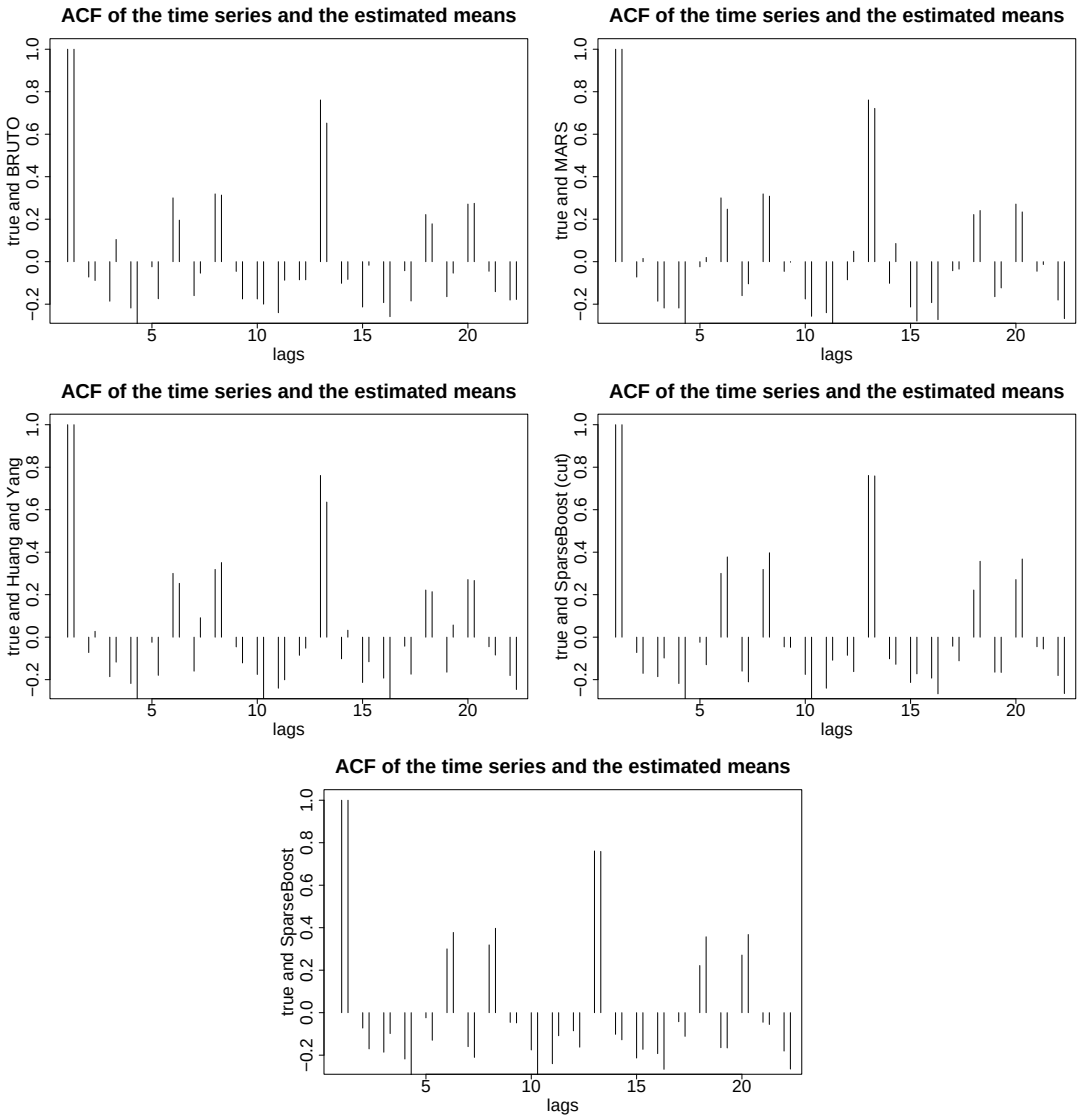

Figure 4: Plot of the autocorrelation functions of the unemployment data time series together with the estimates

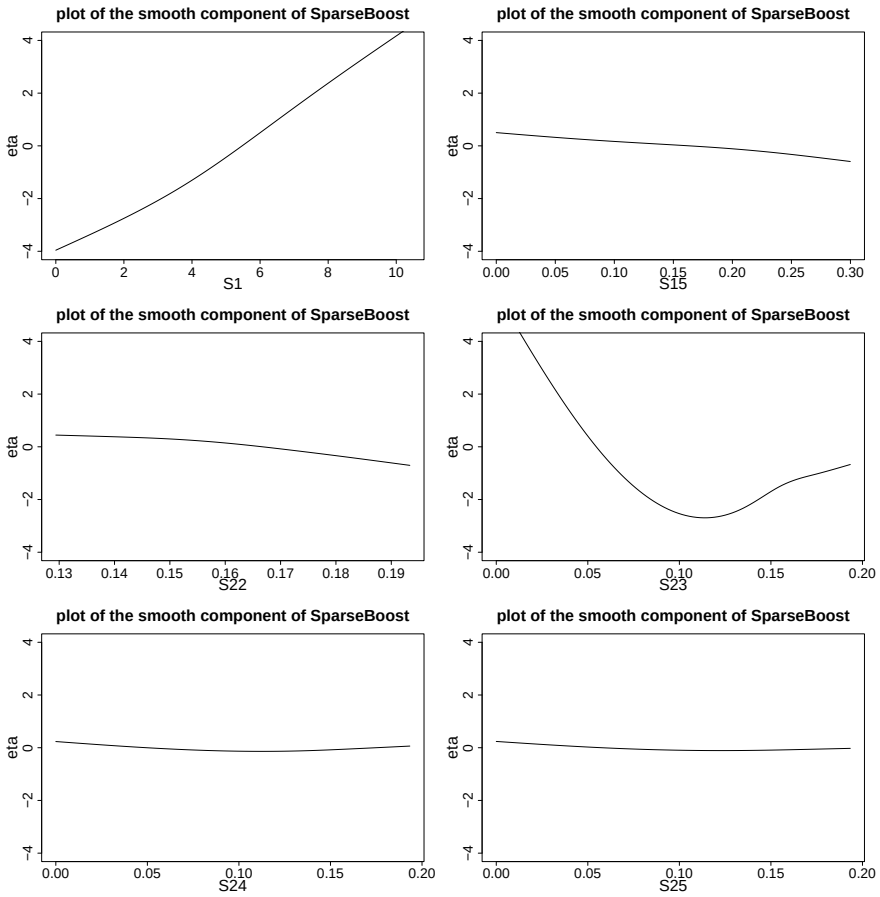

Figure 5: Plot of the smooth components of SparseBoost