Boosting, Model Selection, Lasso and Nonnegative Garrote

Peter Bühlmann
ETH Zürich

Bin Yu
University of California, Berkeley

January 10, 2005

Abstract

We study L_2Boosting and propose a new alternative version which employs model-selection criteria (MS-L_2Boosting). For the special case of an orthogonal linear model, we give an algorithmical equivalence of L_2Boosting to the Lasso and of one new MS-L_2Boosting to Breiman's nonnegative garrote estimator.

The connection to model selection criteria is based on a reasonable and computable definition of degrees of freedom for L_2Boosting. Consequently, we estimate the stopping iteration for L_2Boosting using a model selection criterion. This could result in a very substantial computational saving over a cross-validation tuned boosting. Moreover, we use model selection criteria for our new MS-L_2Boosting which proceeds by a stagewise reduction of a penalized (via model selection criteria) squared error. The model selection criteria explicitly considered are AIC_c, BIC, FPE, and $gMDL$ which has a data-dependent penalty and bridges between AIC and BIC. We also show how L_2- and MS-L_2Boosting can be used in the general nonparametric setting. This is in contrast to the Lasso or the nonnegative garrote estimator which are restricted to a (generalized) linear model or basis expansion using a fixed dictionary. Finally, simulation studies are carried out to compare different model selection criteria and illustrate the effectiveness of the model-selection stopped L_2Boosting and the new MS-L_2Boosting.

1 Introduction

Since its inception in a practical form in Freund and Schapire (1996), boosting has obtained and maintained its outstanding performance in numerous empirical studies both in the machine learning and statistics literatures. The gradient descent view of boosting as articulated in Breiman (1998) and Friedman et al. (2000) provides a springboard for the understanding of boosting to leap forward and at the same time serves as the base for new variants of boosting to be generated. In particular, the L_2Boosting (Friedman, 2001) takes the simple form of refitting a base learner to residuals of the previous iteration. It coincides with Tukey's (1977) twicing at its second iteration and reproduces matching pursuit of Mallat and Zhang (1993) when applied to a dictionary or collection of fixed basis functions. Bühlmann and Yu (2003) investigated L_2Boosting for linear base learners (base procedures) and showed that in such cases the variance or complexity of the boosted procedure is bounded and increases at an increment which is exponentially diminishing as iterations run – this special case calculation implies that the resistance to the over-fitting behavior of boosting could be due to the fact that the complexity of boosting increases at an extremely slow pace.

Consistency results for boosting-type algorithms include Mannor et al. (2002), Jiang (2004), Lugosi and Vayatis (2004), and Zhang and Yu (2003). Jiang (2004) and Zhang and Yu (2003) look at consistency achieved by early-stopping, and the others study consistency achieved by regularization through some penalty or constraint. Only Jiang (2004) considers the original AdaBoost algorithm for classification. All other authors consider versions of boosting with either ℓ_1-constraints for the boosting aggregation coefficients or a (numerically) relaxed version of boosting which is somewhere in between an ℓ_1-constrained and an early-stopped boosting. In the case of the plain L_2Boosting algorithm without further modifications, which is in our experience the easiest to use in practice, stronger results exist: Bühlmann and Yu (2003) showed that when using a smoothing spline base procedure, the method is asymptotically minimax optimal for the toy problem of one-dimensional curve estimation. Moreover, Bühlmann (2004) proved consistency of L_2Boosting for very high-dimensional linear models using a componentwise least squares regression base procedure (see section 2.3). The model dimension is allowed to grow as fast as $O(\exp(n^{1-\xi}))$ $(\xi > 0)$ as sample size $n \to \infty$, but the underlying true regression function is assumed to be sparse (it has a bounded ℓ_1-norm for its coefficients).

Recently Efron et al. (2004) made an intriguing connection between L_2Boosting and Lasso (Tibshirani, 1996) which is an ℓ_1-penalized least squares method. They consider a version of L_2Boosting, called forward stagewise least squares (denoted in the sequel by FSLR) and they show that for many cases, FSLR with infinitesimally small step-sizes produces a set of solutions which coincides with the set of Lasso solutions when varying the regularization parameter in Lasso. Furthermore, Efron et al. (2004) proposed the least angle regression (LARS) algorithm which is a clever computational short-cut for FSLR and Lasso. However, as Efron et al. (2004, sec. 8) write, their LARS procedure is not directly applicable to more general base procedures or learners such as Friedman's (2001) MART algorithm, which is the same as the L_2Boosting method when using regression trees as base learners. Thus, the connections between Lasso, FSLR and LARS are not understood for all cases.

The purpose of this paper is two-fold. First, we provide further analysis and understanding of L_2Boosting and its connections to FSLR and Lasso in the regression setting. Second we propose a new alternative version of boosting which is based on model-selection criteria (MS-L_2Boosting) where AIC_c, BIC, FPE, and $gMDL$ are explicitly considered. The key to connect to various model selection criteria is the notion of degrees of freedom for L_2Boosting. A reasonable definition has been already proposed in Bühlmann (2004), namely the trace of the boosting operator which is easily computable and can be used whenever the base procedure in boosting involves a linear fitting of a response vector to some (data-) selected subset of basis functions or some subset of predictor variables. Consequently, we can estimate the stopping iteration for L_2Boosting using a model selection criterion such as AIC_c, BIC, FPE, and $gMDL$ (cf. section 3.1) [the special case of AIC_c in the context of boosting for linear models has already been considered in Bühlmann (2004)], and as a new main contribution, we propose in section 3.2 the novel MS-L_2Boosting method as a natural and interesting alternative to L_2Boosting. The computational savings using model-selection tuned boosting instead of a cross-validation choice can become very substantial.

For the special case of an orthonormal linear model, we give an algorithmical equivalence of L_2Boosting to the Lasso or soft-thresholding in Section 2.3, and of the new MS-L_2Boosting based on FPE to Breiman's (1995) nonnegative garrote estimator in Section 3.3. The former result implies some asymptotic minimax property of L_2Boosting, for the somewhat special case of an orthonormal linear model. Although such a result about soft-thresholding may be expected from some soft-threshold property of the LARS algorithm in Efron et al. (2004), we give here a rigorous analysis for L_2Boosting which indicates the role of the step-size in boosting more clearly. In particular, this enables us to see some distinct properties of L_2Boosting and FSLR, at least from a theoretical and conceptual point of view.

We also emphasize here that L_2- and MS-L_2Boosting generalize to the general nonparametric setting, as demonstrated in section 4.2. This is in contrast to the Lasso or the nonnegative garrote estimator which are restricted to a (generalized) linear model or basis expansion using a fixed dictionary. Finally, numerical studies are carried out in section 4 to compare different model selection criteria and illustrate the effectiveness of the model-selection stopped L_2Boosting and the new MS-L_2Boosting. In our simulations, $gMDL$ performs the best overall, possibly because it bridges AIC and BIC (cf. Hansen and Yu, 2001). Thus we recommend to use $gMDL$ to stop L_2Boosting or run MS-$L2$Boosting and then choose the best of the two since they can be compared based on the $gMDL$ score. These methods do not rely on cross-validation and hence could bring substantial computational savings.

2 Boosting with the squared error loss: L_2Boosting

We assume that the data are realizations from

$$(X_1, Y_1), \ldots, (X_n, Y_n),$$

where $X_i \in \mathbb{R}^p$ denotes a p-dimensional predictor variable and $Y_i \in \mathbb{R}$ a univariate response. In the sequel, we denote by $x^{(j)}$ the jth component of a vector $x \in \mathbb{R}^p$. We usually assume that the pairs (X_i, Y_i) are i.i.d. or from a stationary process. The goal is to estimate the regression function $F(x) = \mathbb{E}[Y|X = x]$ which is well known to be the (population) minimizer of the expected squared error loss $\mathbb{E}[(Y - F(X))^2]$.

The boosting methodology in general builds on a user-determined base procedure or learner and uses it repeatedly on modified data which could be outputs from the previous iterations. The final boosted procedure takes the form of linear combinations of the base procedures. For L_2Boosting, one simply fits the base procedure to the original data to start with, then uses the residuals from the previous iteration as the new response vector and refits the base procedure, and so on. As we will see in section 2.2, L_2Boosting is a "constrained" minimization of the empirical squared error risk $n^{-1} \sum_{i=1}^{n} (Y_i - F(X_i))^2$ (with respect to $F(\cdot)$) which yields an estimator $\hat{F}(\cdot)$. The regularization of the empirical risk minimization comes in implicitly by the choice of a base procedure and by algorithmical constraints such as early stopping or penalty barriers.

2.1 Base procedures which do variable selection

To be more precise, a base procedure is in our setting a function estimator based on the data $\{(X_i, U_i);\ i = 1, \ldots, n\}$, where $U_1, \ldots, U_n$ denote some (pseudo-) response variables which are not necessarily the original $Y_1, \ldots, Y_n$. We denote the base procedure function estimator by

$$\hat{g}(\cdot) = \hat{g}_{(\mathbf{X},\mathbf{U})}(\cdot), \tag{1}$$

where $\mathbf{X} = (X_1, \ldots, X_n),\ \mathbf{U} = (U_1, \ldots, U_n)$.

Many base procedures involve some variable selection. That is, only some of the components of the p-dimensional predictor variables X_i are actually contributing in (1). In fact, almost all of the successful boosting algorithms in practice involve base procedures which do variable selection: examples include decision trees (cf. Freund and Schapire (1996), Breiman (1998), Friedman et al. (2000), Friedman (2001)), componentwise smoothing splines which involve selection of the best single predictor variable (cf. Bühlmann and Yu (2003)) or componentwise linear least squares in linear models with selection of the best single predictor variable (cf. Mallat and Zhang (1993), Bühlmann (2004)).

It is useful to represent the base procedure estimator (at the observed predictors X_i) as a hat-operator, mapping the (pseudo-) response to the fitted values:

$$\mathcal{H} : \mathbf{U} \mapsto (\hat{g}_{(\mathbf{X},\mathbf{U})}(X_1), \ldots, \hat{g}_{(\mathbf{X},\mathbf{U})}(X_n)),\ \mathbf{U} = (U_1, \ldots, U_n).$$

If the base procedure selects from a set of predictor variables, we denote the selected predictor variable index by $\hat{\mathcal{S}} \subset \{1, \ldots, p\}$, where $\hat{\mathcal{S}}$ has been estimated from a specified set Γ of subsets of variables. To emphasize this, we write for the hat operator of a base procedure

$$\mathcal{H}_{\hat{\mathcal{S}}} : \mathbf{U} \mapsto (\hat{g}_{(\mathbf{X}_{\hat{\mathcal{S}}},\mathbf{U})}(X_1), \ldots, \hat{g}_{(\mathbf{X}_{\hat{\mathcal{S}}},\mathbf{U})}(X_n)),\ \mathbf{U} = (U_1, \ldots, U_n). \tag{2}$$

The examples below illustrate this formalism.

Componentwise linear least squares in linear model (cf. Mallat and Zhang, 1993; Bühlmann, 2004)
We select only single variables and $\Gamma = \{1, 2, \ldots, p\}$. The selector $\hat{\mathcal{S}}$ chooses the predictor variable which reduces residual sum of squares most when using least squares fitting:

$$\hat{\mathcal{S}} = \arg\min_{1 \le j \le p} \sum_{i=1}^{n} (U_i - \hat{\gamma}_j X_i^{(j)})^2,\ \hat{\gamma}_j = \frac{\sum_{i=1}^{n} U_i X_i^{(j)}}{\sum_{i=1}^{n} (X_i^{(j)})^2}\ (j = 1, \ldots, p).$$

The base procedure is then

$$\hat{g}_{(\mathbf{X}_{\hat{\mathcal{S}}},\mathbf{U})}(x) = \hat{\gamma}_{\hat{\mathcal{S}}} x^{(\hat{\mathcal{S}})},$$

and its hat operator is given by the matrix

$$\mathcal{H}_{\hat{\mathcal{S}}} = \mathbf{X}^{(\hat{\mathcal{S}})} (\mathbf{X}^{(\hat{\mathcal{S}})})^T,\ \mathbf{X}^{(j)} = (X_1^{(j)}, \ldots, X_n^{(j)})^T.$$

Boosting with this base procedure yields a linear model with model selection and parameter estimates which are shrunken towards zero. More details are given in sections 2.2 and 2.3.

Componentwise smoothing spline (cf. Bühlmann and Yu, 2003)
Similarly to a componentwise linear least squares fit, we select only one single variable at a time from $\Gamma = \{1, 2, \ldots, p\}$. The selector $\hat{\mathcal{S}}$ chooses the predictor variable which reduces residual sum of squares most when using a smoothing spline fit. That is, for a given smoothing spline operator with fixed degrees of freedom `d.f.` (which is the trace of the corresponding hat matrix)

$$\hat{\mathcal{S}} = \arg\min_{1 \le j \le p} \sum_{i=1}^{n} (U_i - \hat{g}_j(X_i^{(j)}))^2,$$

$$\hat{g}_j(\cdot) \text{ is the fit from the smoothing spline to } \mathbf{U} \text{ versus } \mathbf{X}^{(j)} \text{ with } \texttt{d.f}.$$

Note that we use the same degrees of freedom `d.f.` for all j's. The hat-matrix corresponding to $\hat{g}_j(\cdot)$ is denoted by $\mathcal{H}_j$ which is symmetric; the exact from is not of particular interest here but is well known, cf. Green and Silverman (1994). The base procedure is

$$\hat{g}_{(\mathbf{X},\mathbf{U})}(x) = \hat{g}_{\hat{\mathcal{S}}}(x^{(\hat{\mathcal{S}})}),$$

and its hat operator is then given by a matrix $\mathcal{H}_{\hat{\mathcal{S}}}$. Boosting with this base procedure yields an additive model fit based on selected variables (cf. Bühlmann and Yu, 2003).

Pairwise thin plate splines
Generalizing the componentwise smoothing spline, we select pairs of variables and $\Gamma = \{(j,k);\ 1 \le j < k \le p\}$. The selector $\hat{\mathcal{S}}$ chooses the two predictor variables which reduce residual sum of squares most when using thin plate splines with two arguments:

$$\hat{\mathcal{S}} = \arg\min_{1 \le j < k \le p} \sum_{i=1}^{n} (U_i - \hat{g}_{j,k}(X_i^{(j)}, X_i^{(k)}))^2,$$

$$\hat{g}_{j,k}(\cdot,\cdot) \text{ is an estimated thin plate spline based on } \mathbf{U} \text{ and } \mathbf{X}^{(j)}, \mathbf{X}^{(k)} \text{ with } \texttt{d.f.},$$

where the degrees of freedom `d.f.` is the same for all $j < k$. The hat-matrix corresponding to $\hat{g}_{j,k}$ is denoted by $\mathcal{H}_{j,k}$ which is symmetric; again the exact from is not of particular interest but can be found in Green and Silverman (1994). The base procedure is

$$\hat{g}_{(\mathbf{X},\mathbf{U})}(x) = \hat{g}_{\hat{\mathcal{S}}}(x^{(\hat{\mathcal{S}})}),$$

where $x^{(\hat{\mathcal{S}})}$ denotes the 2-dimensional vector corresponding to the selected pair in $\hat{\mathcal{S}}$, and the hat operator is then given by a matrix $\mathcal{H}_{\hat{\mathcal{S}}}$. Boosting with this base procedure yields a nonparametric fit with second order interactions based on selected pairs of variables; more details are given in section 4.2.

In all the examples above, the selector is given by

$$\hat{\mathcal{S}} = \arg\min_{\mathcal{S} \in \Gamma} \sum_{i=1}^{n} (U_i - (\mathcal{H}_{\mathcal{S}} \mathbf{U})_i)^2 \tag{3}$$

Also (small) regression trees can be cast into this framework. For example for stumps, $\Gamma = \{(j, c_{j,k});\ j = 1, \ldots, p,\ k = 1, \ldots, n-1\}$, where $c_{j,1} < \ldots < c_{j,n-1}$ are the mid-points between (non-tied) observed values $X_i^{(j)}$ $(i = 1, \ldots, n)$. That is, Γ denotes the set of selected single predictor variables and corresponding split-points. The parameter values for the two terminal nodes in the stump are then given by ordinary least squares which implies a linear hat matrix $\mathcal{H}_{(j,c_{j,k})}$. Note however, that for mid-size or large regression trees, the optimization over the set Γ is usually not done exhaustively.

2.2 L_2Boosting and Forward Stagewise Linear Regression (FSLR)

L_2Boosting is nothing else than repeated fitting of residuals with the base procedure $\hat{g}_{(\mathbf{X},\mathbf{U})}(\cdot)$. Its derivation from a more general functional gradient descent algorithm using the squared error loss has been described by many authors, cf. Friedman (2001).

L_2Boosting

Step 1 (initialization). Given data $\{(X_i, Y_i); i = 1, \ldots, n\}$, fit the base procedure

$$\hat{F}_1(\cdot) = \hat{g}_{(\mathbf{X},\mathbf{Y})}(\cdot).$$

Set $m = 1$.

Step 2. Increase m by 1.
Compute residuals $U_i = Y_i - \hat{F}_{m-1}(X_i)$ $(i = 1, \ldots, n)$ and fit the base procedure to the current residuals. The fit is denoted by $\hat{f}_m(\cdot) = \hat{g}_{(\mathbf{X},\mathbf{U})}(\cdot)$.
Update

$$\hat{F}_m(\cdot) = \hat{F}_{m-1}(\cdot) + \nu \hat{f}_m(\cdot),$$

where $0 < \nu \leq 1$ is a pre-specified step-size parameter.

Step 3 (iteration). Repeat Steps 2 and 3 until some stopping value for the number of iterations is reached.

With $m = 2$ (one boosting step), L_2Boosting has already been proposed by Tukey (1977) under the name "twicing". The number of iterations is the main tuning parameter for L_2Boosting, whereas the choice for the step-size ν is much less crucial as long as ν is small, which is justified by our theory in section 2.3. We usually use $\nu = 0.1$. The number of boosting iterations may be estimated by cross-validation. As a computationally much more efficient alternative, we will develop in section 3.1 an approach which allows to use some model selection criteria to bypass cross-validation.

Example: L_2Boosting with componentwise linear least squares
Using the componentwise linear least squares base procedure from section 2.1, L_2Boosting estimates in every iteration m a selector $\hat{\mathcal{S}}_m$ and a corresponding regression coefficient $\hat{\gamma}_{\hat{\mathcal{S}}_m}$ so that the updating function equals $\hat{f}_m(x) = \hat{\gamma}_{\hat{\mathcal{S}}_m} x^{(\hat{\mathcal{S}}_m)}$ (the notation does not reflect that

$\hat{\gamma}_{\hat{\mathcal{S}}_m}$ also depends on the current residual in iteration m, besides the selector $\hat{\mathcal{S}}_m$). After m boosting iterations, we have a linear model fit

$$\hat{F}_m(x) = \sum_{j=1}^{p} \hat{\beta}^{(m)}_{boost,j} x^{(j)},$$
$$\hat{\beta}^{(m)}_{boost,j} = \sum_{r=1;\hat{\mathcal{S}}_r=j}^{m} \nu\hat{\gamma}_{\hat{\mathcal{S}}_r}. \tag{4}$$

Some of the coefficients $\hat{\gamma}_{m;j}$ may be zero, saying that variable selection has been in action; others are non-zero and can be viewed as shrunken ordinary least squares estimates. For $\nu = 1$, this L_2Boosting is also known in signal processing as matching pursuit (Mallat and Zhang, 1993).

L_2Boosting with componentwise linear least squares is related to forward stagewise linear regression (FSLR), as pointed out by Efron et al. (2004). FSLR differs from L_2Boosting with componentwise linear least squares in the update of the new estimate $\hat{F}_m$: instead of using

$$\hat{F}_m(x) = \hat{F}_{m-1}(x) + \nu\hat{\gamma}_{\hat{\mathcal{S}}_m} x^{(\hat{\mathcal{S}}_m)},$$

where $\hat{\gamma}_{\hat{\mathcal{S}}_m}$ is the least squares estimate when fitting the current residuals against the best predictor variable $x^{(\hat{\mathcal{S}}_m)}$, FSLR updates

$$\hat{F}_{m;FSLR}(x) = \hat{F}_{m-1;FSLR}(x) + \nu\,\mathrm{sign}(\hat{\gamma}_{\hat{\mathcal{S}}_m}) x^{(\hat{\mathcal{S}}_m)}.$$

Note that this description of FSLR is equivalent to the one in Efron et al. (2004). In our limited experience, FSLR has about the same prediction accuracy as L_2Boosting with componentwise linear least squares. However, we give here three reasons to favor boosting over FSLR. First, the update in FSLR is not scale-invariant: the parameter ν is more crucial for FSLR than in boosting since the boosting update is on the scale of the current residuals via the magnitude of the least squares estimate $\hat{\gamma}_{\hat{\mathcal{S}}_m}$. Second, FSLR can get stuck: it can happen that after some iterations, the algorithm alternates by selecting the same predictor variables with alternating signs in the update. Third, the number of boosting iterations can be estimated via degrees of freedom of the boosting operator as to be described in section 3.1. Defining reasonable degrees of freedom which are simple to compute seems not straightforward for FSLR. This has also been pointed out by Efron et al. (2004; comment after formula (4.10)), and they suggest the computationally intensive bootstrap to cope with this problem.

These issues also reflect the fact that FSLR and L_2Boosting with componentwise linear least squares are *not* the same algorithm even though their prediction performances are often comparable. In the case of orthogonal predictor variables and using small step-sizes ν, FSLR and L_2Boosting are both equivalent to the soft-threshold estimator, but the role of the step-size is somewhat different; the details are given in section 2.3. We also would like to point out that Efron et al. (2004) do not explicitly advocate to use FSLR in practice: they rather focus on the more interesting LARS algorithm which recently has been extended to the generalized linear model framework (Madigan and Ridgeway, 2004). On the other hand, the fixed amount up-date in FSLR does have its advantages over L_2Boosting when applying the gradient descent idea to a general objective function which is the sum of a convex loss and a convex penalty function (Zhao and Yu, 2004). And in this case the fixed amount up-date does not get stuck because of a new reserve boosting step.

2.3 L_2Boosting and soft-thresholding in the orthogonal case

In this section, we establish rigorously the equivalence of L_2Boosting with componentwise linear least squares and soft-thresholding in the orthogonal case. A similar result is also shown for FSLR. While the latter result is implicitly present in Efron et al. (2004), our proofs bring out explicitly the different roles that the step-sizes play in L_2Boosting and FSLR.

Consider a linear model with n orthonormal predictor variables. Let $\mathbf{x}^{(j)} = (x_1^{(j)}, \dots, x_n^{(j)})^T$ be the $n \times 1$ vector of the jth predictor variable and

$$\begin{aligned} Y_i &= \sum_{j=1}^{n} \beta_j x_i^{(j)} + \varepsilon_i, \; i = 1, \dots, n, \\ \sum_{i=1}^{n} x_i^{(j)} x_i^{(k)} &= \delta_{jk}, \end{aligned} \tag{5}$$

where δ_{jk} denotes the Kronecker symbol, and $\varepsilon_1, \dots, \varepsilon_n$ are i.i.d. random variables with $\mathbb{E}[\varepsilon_i] = 0$ and $\mathrm{Var}(\varepsilon_i) = \sigma_\varepsilon^2 < \infty$. We assume here the predictor variables as fixed and non-random. Using the standard regression notation, we can re-write model (5) as

$$\mathbf{Y} = \mathbf{X}\beta + \varepsilon, \quad \mathbf{X}^T\mathbf{X} = \mathbf{X}\mathbf{X}^T = I, \tag{6}$$

with the $n \times n$ design matrix $\mathbf{X} = (x_i^{(j)})_{i,j=1,\dots,n}$, the parameter vector $\beta = (\beta_1, \dots, \beta_n)^T$, the response vector $\mathbf{Y} = (Y_1, \dots, Y_n)^T$ and the error vector $\varepsilon = (\varepsilon_1, \dots, \varepsilon_n)^T$. The predictors could also be basis functions $g_j(t_i)$ at observed values t_i with the property that they build an orthonormal system.

The soft-threshold estimator for the unknown parameter vector β, is

$$\hat{\beta}_{soft,j} = \begin{cases} Z_j - \lambda, & \text{if } Z_j \geq \lambda, \\ 0, & \text{if } |Z_j| < \lambda, \\ Z_j + \lambda, & \text{if } Z_j \leq -\lambda. \end{cases} \qquad \text{where } Z_j = (\mathbf{X}^T\mathbf{Y})_j. \tag{7}$$

If the threshold is chosen as

$$\lambda = \lambda_n = \sqrt{2\log(n)}\sigma_\varepsilon,$$

the soft-threshold estimator is asymptotically (near) minimax in a variety of settings, cf. Donoho and Johnstone (1994).

We now present a result saying that for orthogonal predictors, L_2Boosting with componentwise linear least squares, yielding coefficient estimates $\hat{\beta}_{boost,j}^{(m)}$ as in (4), is equivalent to the soft-threshold estimator. We will briefly discuss this property also for FSLR.

Theorem 1. *Consider the model in (5) and a threshold $\lambda_n = a_n\sigma_\varepsilon$ in (7) for any sequence $(a_n)_{n\in\mathbb{N}}$. For L_2Boosting with componentwise linear least squares and using a step-size ν, as described in section 2.2, there exists a boosting iteration m, typically depending on λ_n, ν and the data, such that*

$$\hat{\beta}^{(m)}_{boost,j} = \hat{\beta}_{soft,j} \text{ in (7) with threshold of the form } \lambda_n(1 + e_j(\nu)), \text{ where}$$
$$\max_{1\le j\le n} |e_j(\nu)| \le \nu/(1-\nu) \to 0 \ (\nu \to 0).$$

A proof is given in section 6. We emphasize that the sequence $(a_n)_{n\in\mathbb{N}}$ can be arbitrary: in particular, λ_n does not need to depend on sample size n and can be arbitrary. For the special orthogonal case, Theorem 1 explains the role of the small step size ν. It governs the (relative) errors for approximating any reasonable value of the threshold in soft-thresholding. These approximation errors $e_j(\nu)$ are due to the discreteness of boosting when doing an entire additional iteration. But they can be made as small as desired, by choosing a small ν; the cost for this is only computational, but there is no cost in terms of an increased variance of the estimator. Theorem 1 also establishes minimax optimality of L_2Boosting, via the equivalence to soft-thresholding, for the special case of an orthogonal linear model.

A result similar to Theorem 1 also holds for the FSLR algorithm: for step-size ν, there exists an m, depending on λ_n, ν and the data, such that

$$\hat{\beta}^{(m)}_{FSLR,j} = \hat{\beta}_{soft,j} \text{ in (7) with threshold of the form } \lambda_n\Big(1 + \frac{e_j(\nu)}{\lambda_n}\Big),$$
$$\max_{1\le j\le n} |e_j(\nu)| \le 3\nu/2 \to 0 \ (\nu \to 0). \tag{8}$$

A proof is given in section 6. The result is also implicitly present in Efron et al. (2004), but our analysis allows to discuss in more details the role of the step-size. We see from (8) that in order to get the soft-threshold estimator with a good approximation for the parameter λ_n, we have to choose a step-size $\nu = \nu_n$, depending on λ_n, such that $\nu_n = o(\lambda_n)$ $(n \to \infty)$. With this choice, the threshold in (8) exhibits an approximation error $e_j(\nu_n)/\lambda_n = o(1)$ $(n \to \infty)$. Thus, if we want to accurately approximate the soft-threshold estimator with a small threshold λ_n, we have to take an even smaller step-size $\nu = \nu_n = o(\lambda_n)$ for FSLR. This is in contrast to L_2Boosting where the approximation errors are bounded by $\nu/(1-\nu)$, *regardless* of the magnitude of the threshold λ_n. In other words, the step-size ν in L_2Boosting controls the approximation error uniformly over all threshold values λ, in contrast to FSLR where the approximation error depends not only on ν but on λ as well. It implies, that L_2Boosting is rather insensitive to the choice of the step-size (if chosen reasonably small such as $\nu = 0.1$), whereas in FSLR, the step-size is a much more critical tuning parameter. We exploit here from another angle that the step-size ν in FSLR is not scale invariant and remains fixed even if the (current) residuals have already a small norm.

3 L_2Boosting and model selection

3.1 Stopping in L_2Boosting using model selection criteria

Using the notation as in (2), the L_2Boosting operator in iteration m is easily shown to be (cf. Bühlmann and Yu, 2003)

$$\mathcal{B}_m = I - (I - \nu\mathcal{H}_{\hat{\mathcal{S}}_m}) \cdots (I - \nu\mathcal{H}_{\hat{\mathcal{S}}_1}), \tag{9}$$

where $\hat{\mathcal{S}}_m$ denotes the selector in iteration m. Moreover, if all the $\mathcal{H}_{\mathcal{S}}$ are linear (i.e. the hat matrix), as in all the examples given in section 2.1, L_2Boosting has an approximately linear representation, where only the data-driven selector $\hat{\mathcal{S}}$ brings in some additional nonlinearity. In particular, the boosting operator has a corresponding matrix-form when using in (9) the hat-matrices for $\mathcal{H}_{\mathcal{S}}$. This suggests that we may use some model selection criteria for estimating the optimal number of boosting iterations if we ignore the nonlinearity. Moreover, this will also allow us to construct an interesting new version of boosting as described in detail in section 3.2.

For a linear regression model with a known noise variance σ_ε^2, the AIC model selection criterion (Akaike 1973; 1974) estimates the prediction or generalization error of a sub-model M_k of dimension k as follows:

$$AIC(M_k) = RSS(M_k) + 2k\sigma_\varepsilon^2,$$

where $RSS(M_k)$ denotes the residual sum of squares in the sub-model M_k using least squares estimation. In general, a final prediction error (or model selection) FPE_α criterion (Akaike, 1970; Shibata, 1981) takes the form

$$FPE_\alpha(M_k) = RSS(M_k) + \alpha k\sigma_\varepsilon^2.$$

Apparently, $\alpha = 2$ gives AIC and $\alpha = \log(n)$ gives BIC (Schwartz, 1978).

In the case of L_2Boosting, the sub-models correspond to the boosting operators $\mathcal{B}_m$ and for the dimensionality of a model, we propose to use

$$\mathrm{trace}(\mathcal{B}_m) = \mathrm{trace}(I - (I - \nu\mathcal{H}_{\hat{\mathcal{S}}_m}) \cdots (I - \nu\mathcal{H}_{\hat{\mathcal{S}}_1})).$$

This is as in Bühlmann (2004) and a standard way of defining degrees of freedom, cf. Green and Silverman (1994). An estimate for the boosting iteration number is then

$$\hat{m} = \arg\min_m FPE_\alpha(\mathcal{B}_m) = \arg\min_m \{\sum_{i=1}^n (Y_i - (\mathcal{B}_m\mathbf{Y})_i)^2 + \alpha\mathrm{trace}(\mathcal{B}_m)\sigma_\varepsilon^2\}.$$

In practice, the noise variance σ_ε^2 is rarely known. But we can use versions of FPE_α for the case of unknown noise variance. AIC and BIC take on different forms for a linear model M_k with dimension k and using least squares estimation:

$$AIC(M_k) = \log(RSS(M_k)/n) + 2k/n,$$
$$BIC(M_k) = \log(RSS(M_k)/n) + \log(n)k/n.$$

Empirical studies have shown that a corrected AIC, denoted by AIC_c (Sugiura, 1978; Hurvich and Tsai, 1989; Hurvich et al., 1998), has often a better finite-sample performance than AIC and is more widely used in practice today. We adopt it here in place of AIC:

$$AIC_c(M_k) = \log(RSS(M_k)/n) + \frac{1 + k/n}{1 - (k+2))/n}.$$

A minimum description length criterion, $gMDL$, (cf. Hansen and Yu, 2001) bridges the AIC and BIC worlds in the sense that it mimics the performance of the best of the two when the true model is finite-dimensional or infinite-dimensional (Speed and Yu, 1993). It takes the form

$$gMDL(M_k) = \log(S(M_k)) + \frac{k}{n}\log(F(M_k)),$$
$$S(M_k) = \frac{RSS(M_k)}{n-k},\ F(M_k) = \frac{\sum_{i=1}^n Y_i^2 - RSS(M_k)}{kS(M_k)}.$$

This criterion measures the code length needed based on a mixture code to transmit the response vector based on model M_k which balances the fit and a data-driven complexity of the model.

All the above model selection criteria depend only on RSS and the dimension k (and the sample size n). We denote the criteria when the noise variance is unknown by $C_A(RSS, k)$ for AIC, $C_{A_c}(RSS, k)$ for AIC_c, $C_B(RSS, k)$ for BIC and $C_{gM}(RSS, k)$ for $gMDL$. We then propose to estimate the stopping iteration as

$$\hat{m} = \arg\min_m C(RSS_m, \text{trace}(\mathcal{B}_m)), \tag{10}$$

where RSS_m is the residual sum of squares after m boosting iterations and $C(\cdot,\cdot)$ represents any of the four model selection criteria above. If the minimizer is not unique, we use the minimal m which minimizes the criterion. When using (10) with any of the four model selection criteria, boosting can now be run without tuning any parameter (we typically do not tune over the step-size ν but rather take a value like $\nu = 0.1$). The amount of computational savings over some cross-validation scheme can be very substantial.

3.2 Boosting of model selection criteria: MS-L_2Boosting

Since the model selection criteria are sensible ways to evaluate a fitted model, a natural idea is to use them as the objective function to boost, instead of the squared error loss. MS-L_2Boosting, where MS stands for model selection, is a boosting-type algorithm, where in every iteration step, a model selection criterion is minimized (instead of the residual sum of squares as in L_2Boosting). The motivation is to minimize a generalization (or out-of-sample) performance measure in every step. In our framework, MS-L_2Boosting minimizes in the mth step a model selection criterion $C(RSS_m, \text{trace}(\mathcal{B}_m))$, as described above, including the FPE-type criteria

$$C_\gamma(RSS_m, \text{trace}(\mathcal{B}_m)) = RSS_m + \gamma\text{trace}(\mathcal{B}_m). \tag{11}$$

By formula (9), the trace of the boosting operator is

$$\mathrm{trace}(\mathcal{B}_m) = \mathrm{trace}(I - (I - \nu \mathcal{H}_{\tilde{\mathcal{S}}_m}) \cdots (I - \nu \mathcal{H}_{\tilde{\mathcal{S}}_1})).$$

We emphasize here that the estimated selectors $\tilde{\mathcal{S}}_j$ are different from (3), as described more clearly below. For $\mathcal{B}$, a (boosting) operator mapping the response vector $\mathbf{Y}$ to the fitted variables, and a model selection criterion $C(RSS, k)$, we use the following objective function to boost:

$$T(\mathbf{Y}, \mathcal{B}) = C\left(\sum_{i=1}^{n} (Y_i - (\mathcal{B}\mathbf{Y})_i)^2, \mathrm{trace}(\mathcal{B})\right). \tag{12}$$

The algorithm is then as follows.

MS-L_2Boosting

Step 1 (initialization). Given data $\{(X_i, Y_i); i = 1, \ldots, n\}$, fit an initial weak learner

$$\hat{F}_1(\cdot) = \hat{g}_{(\mathbf{X},\mathbf{Y})}(\cdot),$$

where $\hat{g}_{(\mathbf{X},\mathbf{Y})}(\cdot)$ is defined in (1). Set $m = 1$.

Step 2. Increase m by 1.
Search for the best selector

$$\begin{aligned} \tilde{\mathcal{S}}_m &= \mathrm{argmin}_{\mathcal{S} \in \Gamma} T(\mathbf{Y}, \mathrm{trace}(\mathcal{B}_m(\mathcal{S}))), \\ \mathcal{B}_m(\mathcal{S}) &= I - (I - \mathcal{H}_{\mathcal{S}})(I - \nu \mathcal{H}_{\tilde{\mathcal{S}}_{m-1}}) \cdots (I - \nu \mathcal{H}_{\tilde{\mathcal{S}}_1}) \end{aligned}$$

Fit the residuals $U_i = Y_i - \hat{F}_{m-1}(X_i)$ with the base procedure using the selected $\tilde{\mathcal{S}}_m$ which yields a function estimate

$$\hat{f}_m(\cdot) = \hat{g}_{\tilde{\mathcal{S}};(\mathbf{X},\mathbf{U})}(\cdot),$$

where $\hat{g}_{\mathcal{S};(\mathbf{X},\mathbf{U})}(\cdot)$ corresponds to the hat operator $\mathcal{H}_{\mathcal{S}}$ from the base procedure.

Step 3 (update). Update,

$$\hat{F}_m(\cdot) = \hat{F}_{m-1}(\cdot) + \nu \hat{f}_m(\cdot).$$

Step 4 (iteration). Repeat Steps 2 and 3 for a large number of iterations M.

Step 5 (stopping). Estimate the stopping iteration by

$$\hat{m} = \arg\min_{1 \le m \le M} T(\mathbf{Y}, \mathrm{trace}(\mathcal{B}_m)), \quad \mathcal{B}_m = I - (I - \nu \mathcal{H}_{\tilde{\mathcal{S}}_m}) \cdots (I - \nu \mathcal{H}_{\tilde{\mathcal{S}}_1}).$$

The final estimate is $\hat{F}_{\hat{m}}(\cdot)$.

The only difference with boosting is that the selection in Step 2 yields a different $\tilde{\mathcal{S}}_m$ than in (3). While $\hat{\mathcal{S}}_m$ in (3) minimizes residual sum of squares, the selected $\tilde{\mathcal{S}}_m$ in MS-L_2Boosting minimizes a model selection criterion over all possible selectors (but keeping the step-size ν fixed). In particular, this means that MS-L_2Boosting can not be represented anymore as a linear combination of base procedures since the selector $\tilde{\mathcal{S}}_m$ depends not only on the current residuals $\mathbf{U}$ but also explicitly on all previous boosting iterations through $\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2, \ldots, \tilde{\mathcal{S}}_{m-1}$ via the trace of $\mathcal{B}_m(\mathcal{S})$. With a slight abuse of terminology, we still use wordings such as "MS-L_2Boosting with componentwise linear least squares", meaning that the selector $\hat{\mathcal{S}}$ in the componentwise linear lest squares procedure as described in section 2.1 has to be replaced by $\tilde{\mathcal{S}}$.

3.3 MS-L_2Boosting and nonnegative garrote in the orthogonal case

MS-L_2Boosting based on C_γ as in (11) enjoys a surprising equivalence to the nonnegative garrote estimator in an orthogonal linear model.

The nonnegative garrote estimator has been proposed by Breiman (1995) for a linear regression model to improve over subset selection. It shrinks each ordinary least squares (OLS) estimated coefficient by a nonnegative amount whose sum is subject to an upper bound constraint (the garrote). For a given response vector $\mathbf{Y}$ and a design matrix $\mathbf{X}$ (cf. (6)), the nonnegative garrote estimator takes the form

$$\hat{\beta}_{gar,j} = c_j \hat{\beta}_{OLS,j}$$

so that

$$\sum_{i=1}^{n} (Y_i - (\mathbf{X}\hat{\beta}_{gar})_i)^2 \text{ is minimized, subject to } \; c_j \geq 0, \; \sum_{j=1}^{p} c_j \leq s, \tag{13}$$

for some $s > 0$. In the orthonormal case from (5), since the OLS estimator is simply $\hat{\beta}_{OLS,j} = (\mathbf{X}^T\mathbf{Y})_j = Z_j$, the nonnegative garrote minimization problem becomes finding c_j's such that

$$\sum_{j=1}^{n} (Z_j - c_j Z_j)^2 \text{ is minimized, subject to } \; c_j \geq 0, \; \sum_{j=1}^{n} c_j \leq s.$$

Introducing a Lagrange multiplier $\tau > 0$ for the sum constraint gives the dual optimization problem: minimizing

$$\sum_{j=1}^{n} (Z_j - c_j Z_j)^2 + \tau \sum_{j=1}^{n} c_j, \quad c_j \geq 0 \text{ for } j = 1, ..., n. \tag{14}$$

This minimization problem has an explicit solution (Breiman, 1995):

$$c_j = (1 - \lambda/|Z_j|^2)^+, \; \lambda = \tau/2,$$

where $u^+ = \max(0, u)$. Hence $\hat{\beta}_{gar,j} = (1 - \lambda/|Z_j|^2)^+ Z_j$ or equivalently,

$$\hat{\beta}_{gar,j} = \begin{cases} Z_j - \lambda/|Z_j|, & \text{if } \operatorname{sign}(Z_j) Z_j^2 \geq \lambda, \\ 0, & \text{if } Z_j^2 < \lambda, \\ Z_j + \lambda/|Z_j|, & \text{if } \operatorname{sign}(Z_i) Z_j^2 \leq -\lambda. \end{cases}, \qquad \text{where } Z_j = (\mathbf{X}^T\mathbf{Y})_j. \tag{15}$$

We show in Figure 1 the nonnegative garrote threshold function in comparison to hard- and soft-thresholding. Hard-thresholding, corresponding to subset selection, is using ordinary least squares if $|Z_j|$ is larger than the threshold while the nonnegative garrote shrinks the OLS estimator a bit and soft-thresholding, corresponding to the Lasso, even more. Therefore, for the same amount of "complexity" or "degrees of freedom" (which is in case of hard-thresholding the number of ordinary least squares estimated variables), hard-thresholding or subset selection will typically select the fewest number of variables (non-zero coefficient estimates) while the nonnegative garrote will include some more variables and the soft-thresholding will be the least sparse in terms of the number of selected variables; the reason is that for the non-zero coefficient estimates, the shrinkage effect, which is slight in the nonnegative garotte and stronger for soft-thresholding, causes fewer degrees of freedom for every selected variable. This observation can also be compared with some numerical results in Figures 3 and 5 and Table 2.

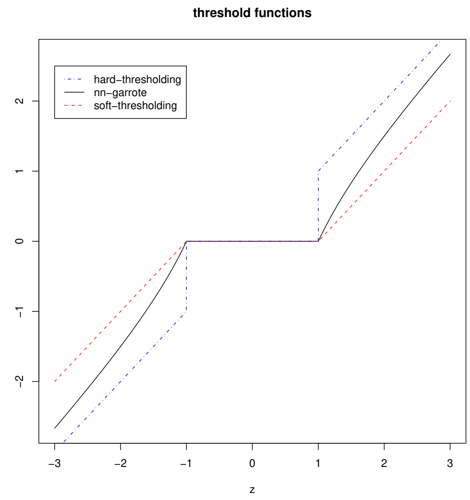

Figure 1: Threshold functions for subset selection or hard-thresholding (dashed-dotted line), nonnegative garrote (solid line) and lasso or soft-thresholding (dashed line).

The following result shows the equivalence of the nonnegative garrote and MS-L_2Boosting with componentwise linear least squares, yielding coefficient estimates $\hat{\beta}^{(m)}_{ms-boost,j}$, analogous to (4).

Theorem 2. *Consider the model in (5) and any sequence $(\gamma_n)_{n\in\mathbb{N}}$. For MS-$L_2$Boosting with componentwise linear least squares, based on C_{γ_n} as in (11) and using a step-size ν, as described in section 3.2, we have*

$$\hat{\beta}^{(\hat{m})}_{ms-boost,j} = \hat{\beta}_{gar,j} \text{ in (15) with parameter } \lambda_n = \frac{1}{2}\gamma_n(1 + e_j(\nu)),$$
$$\max_{1\le i\le n} |e_j(\nu)| \le \nu/(1-\nu) \to 0 \ (\nu \to 0).$$

A proof is given in section 6. Note that the sequence $(\gamma_n)_{n\in\mathbb{N}}$ can be arbitrary and does not need to depend on n (and likewise for the corresponding λ_n). Theorems 1 and 2 yield interesting interpretations of L_2Boosting and MS-L_2Boosting with C_γ as soft-threshold and nonnegative garrote estimators for the orthogonal case.

We briefly discuss now the choice of $\gamma = \gamma_n$ in (11) using some model selection criteria. We restrict ourselves to the case of known noise variance and consider the BIC criterion

$$\gamma_n = \log(n)\sigma_\varepsilon^2,$$

which yields, by Theorem 2, the equivalence to the nonnegative garrote parameter $\lambda_n \sim \gamma_n/2 = \log(n)\sigma_\varepsilon^2/2$. Some consistency results are known for the nonnegative garrote if $\lambda_n = C\log(n)\sigma_\varepsilon^2$ for some constant $C > 0$, cf. Mohammadi and van de Geer (2002). This indicates consistency of the BIC-driven MS-L_2Boosting.

3.4 Selecting between L_2Boosting and MS-L_2Boosting

There will be no overall superiority of either MS-L_2- or L_2Boosting; the same is true when comparing the Lasso with the nonnegative garrote estimator in a linear model with $p < n$. But it is straightforward to do a data-driven selection between L_2Boosting and MS-L_2Boosting, once we have decided upon the model selection criteria (i.e. AIC_c, BIC, $gMDL$ or FPE). That is, we can simply choose the boosting method which has the smaller final model selection score at the stopped MS-L_2- or L_2Boosting iteration.

However, we recommend to use the $gMDL$ model selection score (called $gMDL$-L_2Boosting) because it makes a good compromise between AIC and BIC as demonstrated in Hansen and Yu (2001). $gMDL$-L_2Boosting also shows very good overall performances in our numerical comparisons of section 4.

4 Numerical results

We investigate and compare L_2Boosting with model-selection based stopping rules, MS-L_2Boosting methods and other methods against each other. The step-sizes in both boosting methods were always chosen as $\nu = 0.1$. The simulation setups are based on some high-dimensional linear models and one nonparametric model. Except for one real data set, all our comparisons and results are based on 50 independent model simulations.

4.1 High-dimensional linear models

Consider the model

$$\begin{aligned} &Y = 1 + 5X_1 + 2X_2 + X_9 + \varepsilon, \\ &X = (X_1, \ldots, X_{p-1}) \sim \mathcal{N}_{p-1}(0, \Sigma),\ \varepsilon \sim \mathcal{N}(0,1), \end{aligned} \tag{16}$$

where ε is independent from X. The sample size is chosen as $n = 50$ and the predictor-dimension is $p \in \{50, 100\}$. For the covariance structure of the predictor X, we consider $\Sigma = I_{p-1}$ and $\Sigma_{ij} = 0.8^{|i-j|}$, respectively.

Case with $\Sigma = I_{p-1}$.
Results about the squared error and the number of selected variables are reported in Figures 2 and 3. MS-L_2Boosting with the $gMDL$ criterion for (12) is clearly the overall best.

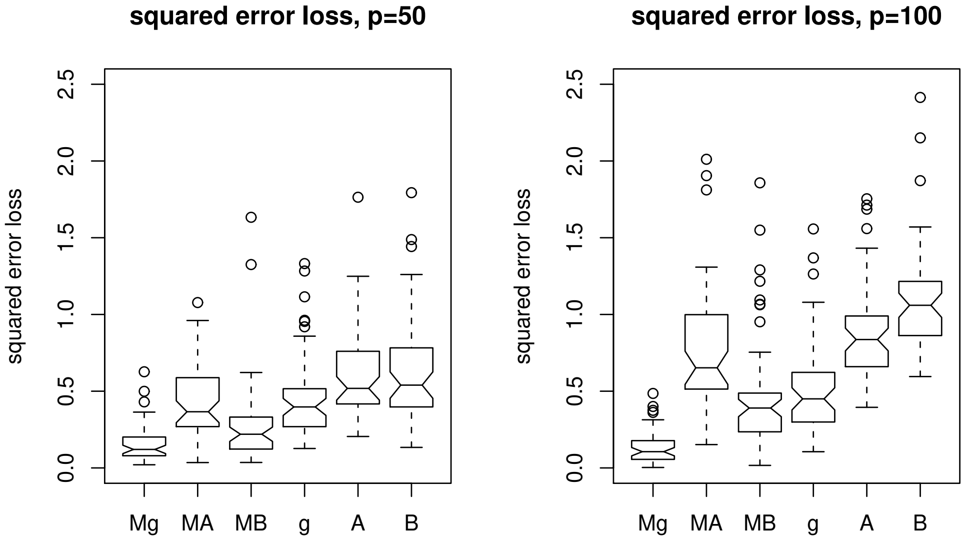

Figure 2: Squared error losses for model (16) with $\Sigma = I_{p-1}$. MS-L_2Boosting with $gMDL$ (Mg), with AIC_c (MA) and with BIC (MB); L_2Boosting stopped with $gMDL$ (g), with AIC_c (A) and with BIC (B). Sample size is $n = 50$.

We notice that L_2Boosting with BIC stopping selects very many variables, in particular more than stopping with AIC_c; in particular, the BIC stopping for (10) deteriorated in the case where $p = 100$. Stopping the (MS-) L_2Boosting iterations with $gMDL$ works very satisfactorily: the mean squared error for the $gMDL$-stopped algorithms is essentially equal to the minimal mean squared error over the (MS-) L_2Boosting iterations, see Table 1. Only for $p = 100$ and in case of L_2Boosting, we pay a slight price for estimating the stopping parameter. Stopping (MS-) L_2Boosting with the AIC_c or BIC criterion was found to be less accurate when calibrated against the minimal (MS-) L_2Boosting performance.

dimension	MS-L_2Boost with $gMDL$	minimal MS-L_2Boost with $gMDL$
$p = 50$	0.16 (0.018)	0.16 (0.018)
$p = 100$	0.14 (0.015)	0.14 (0.015)
	L_2Boost, $gMDL$-stopped	minimal L_2Boost
$p = 50$	0.46 (0.041)	0.46 (0.036)
$p = 100$	0.52 (0.043)	0.48 (0.045)

Table 1: Mean squared error for (MS-) L_2Boosting using the estimated stopping iteration $\hat{m}$ and using the oracle m which minimizes the mean squared error. Model (16) with $\Sigma = I_{p-1}$. Estimated standard errors are given in parentheses. Sample size is $n = 50$.

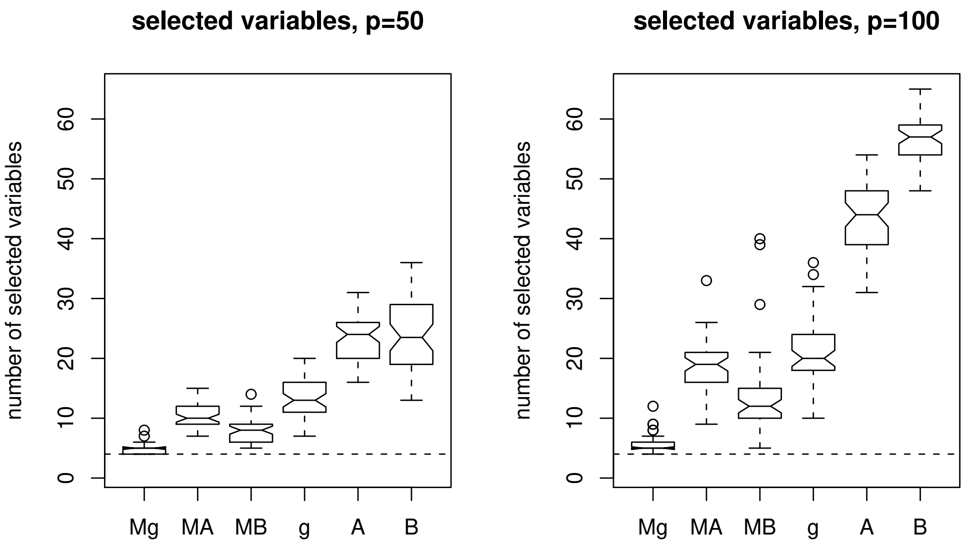

Figure 3: Number of selected variables for model (16) with $\Sigma = I_{p-1}$. MS-L_2Boosting with $gMDL$ (Mg), with AIC_c (MA) and with BIC (MB); L_2Boosting stopped with $gMDL$ (g), with AIC_c (A) and with BIC (B). The horizontal dashed line indicates the number of true effective variables which is 4. Sample size is $n = 50$.

We also evaluate the behavior about under- and overestimation of the true model. Table 2 summarizes the result. Again, MS-L_2Boosting with the gMDL criterion works best and yields for this simulation model a remarkably good variable (or feature) selection method.

method	MgMDL	MAIC$_c$	MBIC	gMDL	AIC$_c$	BIC
$p = 50$						
non-selected T	0	0	0	0	0	0
selected F	1	6.58	3.96	9.68	19.44	19.58
$p = 100$						
non-selected T	0	0	0	0	0	0
selected F	1.78	14.62	9.46	17.2	39.92	52.86

Table 2: Model (16) with $\Sigma = I_{p-1}$. Expected number of non-selected true effective variables (non-selected T) which is in the range of $[0, 4]$, and expected number of selected non-effective (false) variables (selected F) which is in the range of $[0, p-4]$. Methods: MS-L_2Boosting with $gMDL$ (MgMDL), with AIC_c (MAIC$_c$) and with BIC (MBIC); L_2Boosting stopped with $gMDL$ (gMDL), with AIC_c (AIC$_c$) and with BIC (BIC). Sample size is $n = 50$.

Case with $\Sigma = [0.8^{|i-j|}]_{i,j=1,\ldots,p-1}$.
Results about the squared error and the number of selected variables are reported in

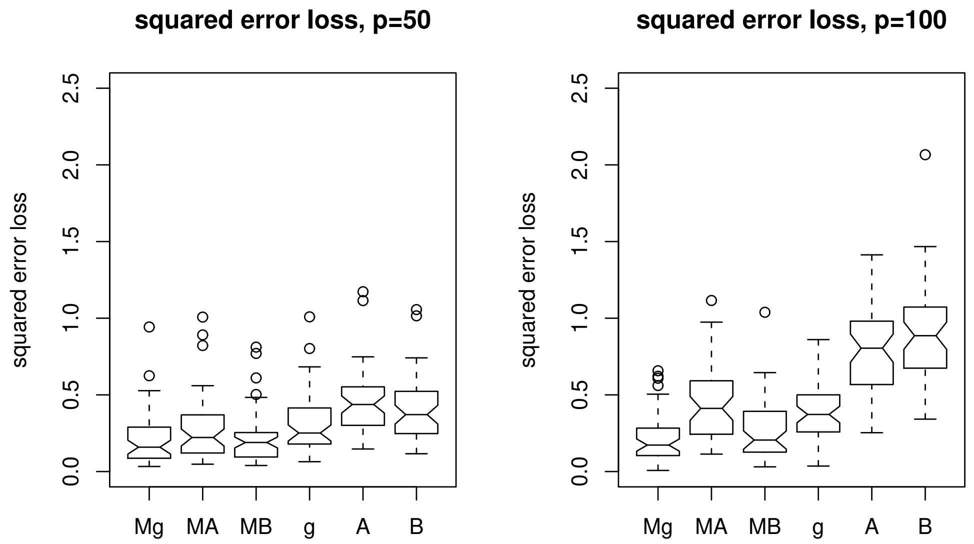

Figure 4: Squared error losses for model (16) with $\Sigma = [0.8^{|i-j|}]_{i,j=1,\ldots p}$. MS-$L_2$Boosting with $gMDL$ (Mg), with AIC_c (MA) and with BIC (MB); L_2Boosting stopped with $gMDL$ (g), with AIC_c (A) and with BIC (B). Sample size is $n = 50$.

Figures 4 and 5.

The conclusions are similar to the results for the case with $\Sigma = I_{p-1}$. Also, the under- and overfitting behavior of the different methods in terms of false positives and negatives is similar to the case with $\Sigma = I_{p-1}$ and we do not report these additional results here. Next we give a case where L_2Boosting is better than MS-L_2Boosting.

A favourable example for boosting.
Consider the model

$$Y = \sum_{j=1}^{p} \frac{1}{5}\beta_j X_j + \varepsilon,$$
$$X_1, \ldots, X_p \sim \mathcal{N}_p(0, I_p),\ \varepsilon \sim \mathcal{N}(0,1), \tag{17}$$

where $\beta_1, \ldots, \beta_p$ are fixed values from i.i.d. realizations of the double-exponential density $p(x) = \exp(-|x|)/2$. The magnitude of the coefficients $|\beta_j|/5$ is chosen to vary the signal to noise ratio from model (16), making it about 5 times smaller than for (17). Since Lasso (coinciding with L_2Boosting in the orthogonal case) is the maximum a-posteriori method when the coefficients are from a double-exponential distribution and the observations from a Gaussian distribution, as in (17), we expect L_2Boosting to be better than MS-L_2Boosting

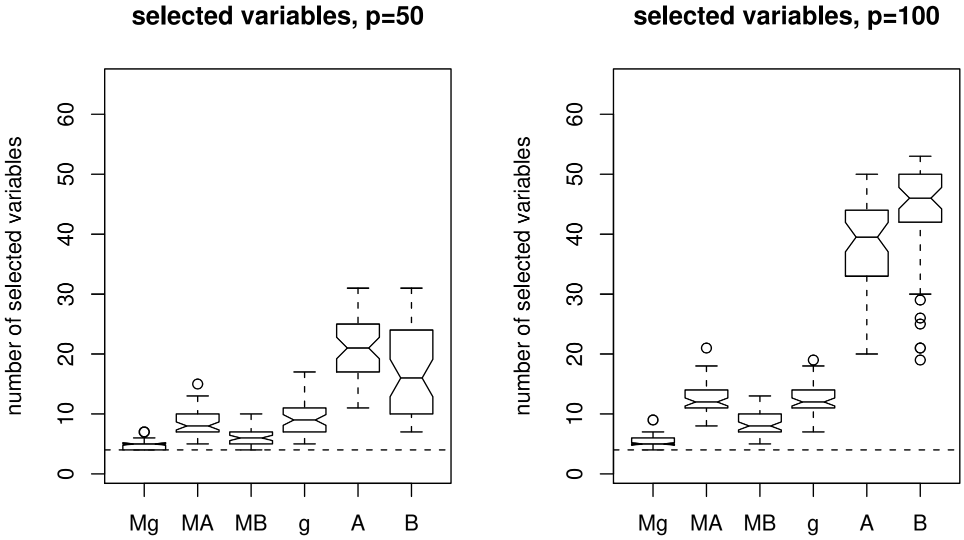

Figure 5: Number of selected variables for model 16 with $\Sigma = [0.8^{|i-j|}]_{i,j=1,...p}$. MS-$L_2$Boosting with $gMDL$ (Mg), with AIC_c (MA) and with BIC (MB); L_2Boosting stopped with $gMDL$ (g), with AIC_c (A) and with BIC (B). The horizontal dashed line indicates the number of true effective variables which is 4. Sample size is $n = 50$.

for this example. The squared error performance is given in Figure 6, supporting our expectations.

4.1.1 Data-driven choice between L_2- and MS-L_2Boosting: gMDL-L_2Boosting

We illustrate here the gMDL-L_2Boosting proposal from section 3.4 with the gMDL model selection score to choose in a data-driven way between MS-L_2- and L_2Boosting. As an illustration, we consider again the models in (16) and (17) with $p = 50$ and $n = 50$. Figure 7 displays the results. The data-driven selected boosting, choosing between L_2- and MS-L_2Boosting, performs somewhere in the middle between the better and the worse of the two boosting algorithms, but closer to the best performer in each situation: for model (17), there is essentially no degraded performance when estimating the better of the two boosting algorithms (when comparing to the better of the two boosting algorithms which is only known for simulated datasets).

4.1.2 Comparison to the nonnegative garrote

We compare here MS-L_2Boosting with the nonnegative garrote estimator from Breiman (1995), defined in (13), which can be used for the case where $p \leq n$.

We consider the model as in (16) with $p - 1 = 10$ and correlated design $\Sigma = [0.8^{|i-j|}]_{i,j=1,...p-1}$ but with $\varepsilon \sim \mathcal{N}(0, 4^2)$ and $n = 50$. The high noise variance is cho-

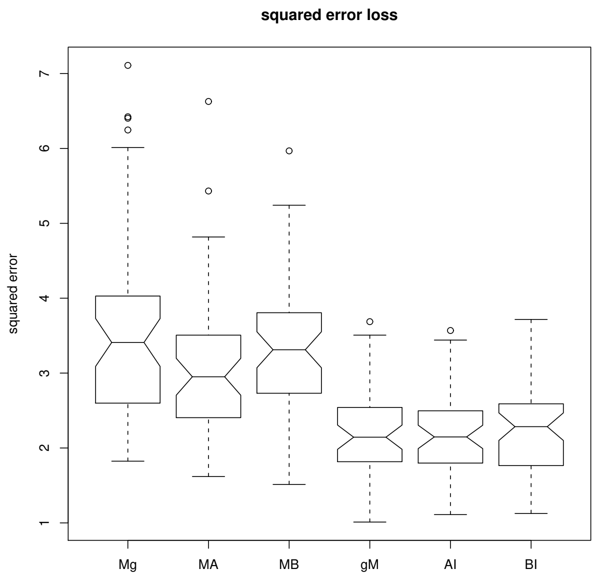

Figure 6: Squared error losses for the model in (17) with $p = 50$. MS-L_2Boosting with $gMDL$ (Mg), with AIC_c (MA) and with BIC (MB); L_2Boosting stopped with $gMDL$ (g), with AIC_c (A) and with BIC (B). Sample size is $n = 50$.

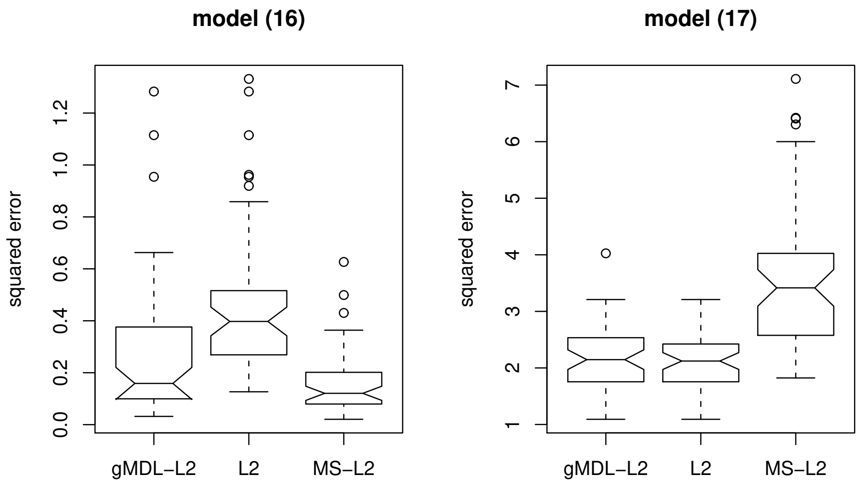

Figure 7: Squared error losses for the model in (16) and (17) with $p = 50$. Data-driven choice between L_2- and MS-L_2Boosting with gMDL (gMDL-L2), L_2Boosting with $gMDL$ (L2) and MS-L_2Boosting with $gMDL$ (MS-L2). Sample size is $n = 50$.

.

sen to make the full least squares estimator unreliable enough (although $p-1=10$ is not very large), calling for some regularization.

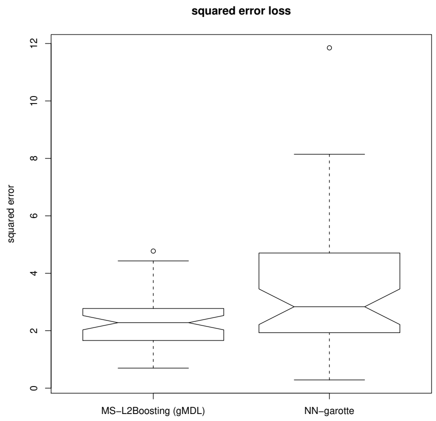

Figure 8: Squared error losses MS-L_2Boosting with the $gMDL$ criterion (left) and for nonnegative garrote (right) for the model as in (16) with $p-1=10$ and correlated design $\Sigma = [0.8^{|i-j|}]_{i,j=1,\ldots p-1}$ but with $\varepsilon \sim \mathcal{N}(0,4^2)$; sample size $n=50$.

The predictive performance when simulating 50 times over the model is described in Figure 8. MS-L_2Boosting with the $gMDL$ criterion works better than the nonnegative garrote whose penalty parameter has been tuned by 10-fold cross-validation (which makes the procedure computationally quite expensive). It is unclear why, for this simulation example, the nonnegative garotte is worse: it could be that another cross-validation tuning procedure than 10-fold would yield better results.

Ozone example with interactions terms.
We also consider a real data set about ozone concentration in the Los Angeles basin. There are $p=8$ meteorological predictors and a real-valued response about daily ozone concentration. As in Breiman (1995), we constructed second-order interaction and quadratic terms after having centered the original predictors. We then obtain a model with $p=45$ predictors (including an intercept) and a response. We used 10-fold cross-validation to estimate out-of-sample squared error, and we also used an internal 10-fold cross-validation to estimate the penalty parameter in the nonnegative garrote method.

The cross-validated squared error is 16.26 for the nonnegative garrote, and 16.52 for MS-L_2Boosting using the $gMDL$ penalty. When scaling the predictor variables (and their interactions) to zero mean and variance one, the performances were very similar: again 16.26 for the nonnegative garrote and 16.81 for MS-L_2Boosting using the $gMDL$ penalty. Thus, the performance of both methods is essentially the same. Note that MS-L_2Boosting does not need a (internal) cross-validation for tuning. We also tried the MS-L_2Boosting with the AIC_c penalty using the non-scaled predictors: the cross-validated squared error was 16.14. Our results are comparable to the analysis of bagging in Breiman (1996) which yielded a cross-validated squared error of 18.8 for bagging trees based on the original eight predictors (We were unable to reproduce some results which were nearly as good as reported in Breiman (1995): maybe this is due to some standardization of the response variables in Breiman (1995)).

When running MS-L_2Boosting with the $gMDL$ criterion on the whole dataset, 10 terms (out of 45) have been selected, including an intercept. Then, an estimate for the error variance is $n^{-1}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 = 15.56$ and the goodness of fit equals $R^2 = \sum_{i=1}^{n}(\hat{Y}_i - \overline{Y})^2 / \sum_{i=1}^{n}(Y_i - \overline{Y}_i)^2 = 0.71$.

4.2 Nonparametric function estimation with second-order interactions

Consider the Friedman #1 model (Friedman, 1991),

$$
\begin{aligned}
&Y = 10\sin(\pi X_1 X_2) + 20(X_3 - 0.5)^2 + 10X_4 + 5X_5 + \varepsilon, \\
&X \sim \text{Unif.}([0,1]^p),\ \varepsilon \sim \mathcal{N}(0,1),
\end{aligned}
\tag{18}
$$

where ε is independent from X. The sample size is chosen as $n = 50$ and the predictor dimension $p \in \{10, 20\}$ which is still large relative to n for a nonparametric problem.

(MS-) L_2Boosting with a pairwise thin plate spline, which selects the best pair of predictor variables yielding lowest residual sum of squares (when having the same degrees of freedom for every thin plate spline), yields a second-order interaction model; see also section 2.1. We demonstrate in Figure 9 the effectiveness of this procedure, also in comparison with the MARS (Friedman, 1991) fit constrained to second-order interaction terms. MS-L_2Boosting is better than L_2Boosting, and the AIC_c is better than the $gMDL$ criterion. Note that with the parameter choices as in Friedman (1991), the signal to noise ratio is very high which is the reason why we see hardly any overfitting of boosting until iteration 500. If we use a lower signal to noise ratio by choosing $\varepsilon \sim \mathcal{N}(0, 4^2)$ in (18), we see more clear overfitting and the stopping rules via model selection are close to the minimum over all considered boosting iterations: see Figure 10. The mean squared error for MARS (restricted to 2nd-order interactions) for the higher noise case with $\varepsilon \sim \mathcal{N}(0, 4^2)$ is 24.11 and thus much worse than any of the boosting methods.

We also examined for some of the settings the performance when using 10-fold cross-validation for stopping the MS-L_2Boosting with $gMDL$. The mean squared errors in comparison to the stopping rule from the $gMDL$-score directly (as in Step 5 of the algorithm in section 3.2) are as follows:

Var(ε)	MS-L_2Boost ($gMDL$)	MS-L_2Boost ($gMDL$) with 10-fold CV-stopping
1	3.71	12.59
4^2	11.70	15.15

For this specific example, 10-fold cross-validation is clearly worse than using the $gMDL$-score for stopping, in particular in the low noise case. One should be cautious though in generalizing this finding to other problems.

The result that the AIC_c is better than the $gMDL$ criterion for (MS-) L_2Boosting (in the low and higher noise case) is somewhat different than for the parametric model in (16) which involves only very few effective predictors and where the MSE ratio of AIC_c over $gMDL$ can be as large as 5; in the case of model (17) where all predictors are effective, we also found that MS-L_2Boosting with the AIC_c is better than using the $gMDL$ criterion.

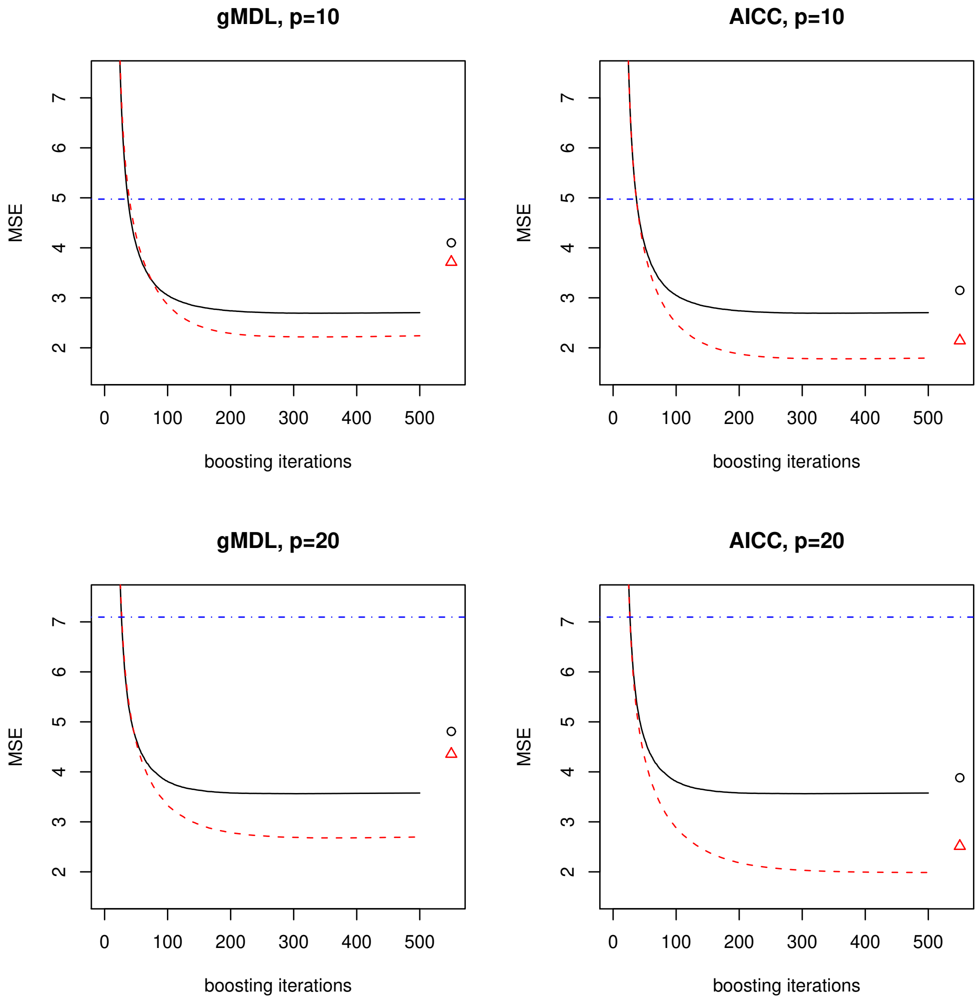

Figure 9: Mean squared errors for the nonparametric Friedman #1 model in (18) with $p \in \{10, 20\}$. (MS-) L_2Boosting are used with componentwise two-dimensional thin plate splines having `d.f.` $= 5$. Left panel: MS- and L_2Boosting with $gMDL$. Right panel: MS- and L_2Boosting with AIC_c. Upper panel: $p = 10$. Lower panel: $p = 20$. Solid lines correspond to L_2Boosting, dashed lines to MS-L_2Boosting. The circle indicates the performance for stopped L_2Boosting iterations, the triangle for stopped MS-L_2Boosting iterations. The horizontal line indicates the performance of MARS restricted to 2nd-order interactions.

However, the improvements here of AIC_c over $gMDL$ are not very substantial (MSE ratio of $gMDL$ to AIC_c close to 1 and well below 2), so we nevertheless think that the $gMDL$ criterion is a good overall choice. Also, in such high-dimensional nonparametric settings, the (MS-) L_2Boosting is clearly better than the more classical MARS fit, while all of the methods share the same simplicity of interpretation as second-order interaction models.

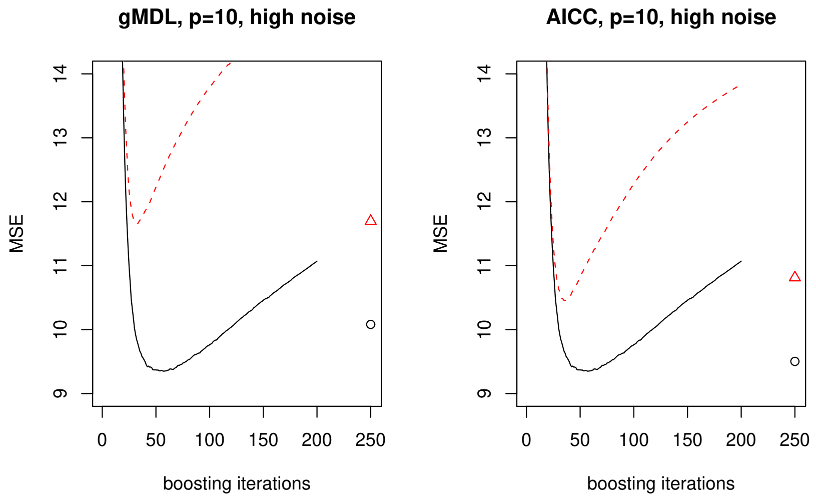

Figure 10: Mean squared errors for the higher noise, nonparametric Friedman #1 model in (18) with $p = 10$ and $\varepsilon \sim \mathcal{N}(0, 4^2)$. (MS-) L_2Boosting are used with componentwise two-dimensional thin plate splines having `d.f.` $= 5$. Left panel: MS- and L_2Boosting with $gMDL$. Right panel: MS- and L_2Boosting with AIC_c. Solid lines correspond to L_2Boosting, dashed lines to MS-L_2Boosting. The circle indicates the performance for stopped L_2Boosting iterations, the triangle for stopped MS-L_2Boosting iterations.

5 Conclusions

We study L_2Boosting and propose a new alternative version which is based on model-selection criteria (MS-L_2Boosting) where AIC_c, BIC, FPE, and $gMDL$ are explicitly considered. For the special case of an orthonormal linear model, we give an algorithmical equivalence of L_2Boosting to the Lasso or soft-thresholding, and of the new MS-L_2Boosting based on FPE to Breiman's (1995) nonnegative garrote estimator. This establishes some asymptotic minimax optimality for L_2Boosting in the special orthogonal case, and it is useful to get some insight into what L_2Boosting and MS-L_2Boosting do. There is no general superiority of one method over the other, very much as the comparison of the Lasso with the nonnegative garrote does not lead to an overall, general preference. While the Lasso and in particular the nonnegative garrote estimator are restricted to (generalized) linear models or basis expansions with a fixed dictionary, the (MS-) L_2Boosting easily generalizes to more general nonparametric settings: we show some examples for fitting a nonparametric function by (MS-) L_2Boosting allowing for second-order interactions.

The boosting approach automatically comes with a reasonable notion for degrees of freedom, namely the trace of the boosting operator trace$(\mathcal{B}_m)$ which is well defined for the cases where the base procedure in boosting involves linear fitting of the response vector to some (data-) selected subset of basis functions or of some predictor variables. This implies a direct, fast computable estimate of the out-of-sample error via some model selection criteria, and in turn, this out-of-sample error estimate allows for computationally attractive methods to stop the L_2Boosting iterations and to design our new MS-L_2Boosting. When using a model selection criterion as described in section 3.1, (MS-) L_2Boosting can be run without tuning any parameter (we typically do not tune over the step-size ν but rather use a value such as $\nu = 0.1$). The amount of computational savings over some cross-validation scheme may become very substantial. The numerical studies compare three model selection criteria AIC_c, BIC and $gMDL$ when used with L_2Boosting and MS-L_2Boosting, and we recommend gMDL-L_2Boosting to be used in practice as an automated boosting method.

Finally, (MS-) L_2Boosting is a much more efficient way to do variable selection in a linear model than some exhaustive classical BIC optimization. Instead of searching over 2^p sub-models, we can do (MS-) L_2Boosting which is computationally feasible for thousands of predictor variables. The idea of replacing a combinatorial optimization by a convex minimization problem is also present in the Lasso and the nonnegative garrote estimators, although the latter is restricted to the case $p \leq n$. Unlike (MS-) L_2Boosting with its computationally efficient model selection criteria, it seems that for the nonnegative garrote (but to a lesser extent for the Lasso, cf. Efron et al. (2004)), we have to rely on some cross-validation scheme for estimating the out-of-sample error and tuning the methods.

6 Proofs

Proof of Theorem 1. We represent the componentwise linear least squares base procedure as a hat operator $\mathcal{H}_{\hat{\mathcal{S}}}$ with $\mathcal{H}_j = \mathbf{x}^{(j)}(\mathbf{x}^{(j)})^T$, where $\mathbf{x}^{(j)} = (x_1^{(j)}, \ldots, x_n^{(j)})^T$; see also section 2.1. The L_2Boosting operator in iteration m is then given by the matrix

$$\mathcal{B}_m = I - (I - \nu\mathcal{H}_1)^{m_1}(I - \nu\mathcal{H}_2)^{m_2} \cdots (I - \nu\mathcal{H}_n)^{m_n},$$

where m_i equals the number of times that the ith predictor variable has been selected during the m boosting iterations; and hence $m = \sum_{i=1}^{n} m_i$. The derivation of the formula above is straightforward because of the orthogonality of the predictors $\mathbf{x}^{(j)}$ and $\mathbf{x}^{(k)}$ which implies the commutation $\mathcal{H}_j\mathcal{H}_k = \mathcal{H}_k\mathcal{H}_j$. Moreover, $\mathcal{B}_m$ can be diagonalized

$$\mathcal{B}_m = \mathbf{X}D_m\mathbf{X}^T \text{ with } \mathbf{X}^T\mathbf{X} = \mathbf{X}\mathbf{X}^T = I,\ D_m = \mathrm{diag}(d_{m,1}, \ldots, d_{m,n}),\ d_{m,i} = 1 - (1-\nu)^{m_i}.$$

Therefore, the residual sum of squares in the mth boosting iteration is:

$$RSS_m = \|\mathbf{Y} - \mathcal{B}_m\mathbf{Y}\|^2 = \|\mathbf{X}^T\mathbf{Y} - \mathbf{X}^T\mathcal{B}_m\mathbf{Y}\|^2 = \|Z - D_mZ\|^2 = \|(I - D_m)Z\|^2,$$

where $Z = \mathbf{X}^T\mathbf{Y}$.

The RSS_m decreases monotonously in m. Moreover, the amount of decrease $RSS_m - RSS_{m+1}$ is decaying monotonously in m, because L_2Boosting proceeds to decrease the RSS as much as possible in every step (by selecting the most reducing predictor $\mathbf{x}^{(j)}$) and due to the structure of $(1 - d_{m,i}) = (1-\nu)^{m_i}$. Thus, every stopping of boosting with an iteration number m corresponds to a tolerance δ^2 such that

$$\begin{aligned} &RSS_k - RSS_{k+1} > \delta^2,\ k = 1, 2, ..., m-1, \\ &RSS_m - RSS_{m+1} \leq \delta^2, \end{aligned} \tag{19}$$

that is, the iteration number m corresponds to a numerical tolerance where the difference $RSS_m - RSS_{m+1}$ is smaller than δ^2.

Since L_2Boosting changes only one of the summands in RSS_m in the boosting iteration $m+1$, the criterion in (19) implies that for all $i \in \{1, \ldots, n\}$

$$\begin{aligned} &((1-\nu)^{2(m_i-1)} - (1-\nu)^{2m_i})Z_i^2 > \delta^2, \\ &((1-\nu)^{2m_i} - (1-\nu)^{2(m_i+1)})Z_i^2 \leq \delta^2. \end{aligned} \tag{20}$$

If $m_i = 0$, only the second line in the above expression is relevant. The L_2Boosting solution with tolerance δ^2 is thus characterized by (20).

Let us first, for the sake of insight, replace the "$\leq$" in (20) by "$\approx$": we will deal later in which sense such an approximate equality holds. If $m_i \geq 1$, we get

$$((1-\nu)^{2m_i} - (1-\nu)^{2(m_i+1)})Z_i^2 = (1-\nu)^{2m_i}(1-(1-\nu)^2)Z_i^2 \approx \delta^2,$$

and hence

$$(1-\nu)^{m_i} \approx \frac{\delta}{\sqrt{1-(1-\nu)^2}|Z_i|}. \tag{21}$$

In case where $m_i = 0$, we obviously have that $1-(1-\nu)^{m_i} = 0$. Therefore,

$$\begin{aligned} &\hat{\beta}^{(m)}_{boost,i} = \hat{Z}_i = d_{m,i} = (1-(1-\nu)^{m_i})Z_i \approx Z_i - \frac{\delta}{\sqrt{1-(1-\nu)^2}|Z_i|}Z_i \quad \text{if } m_1 \geq 1, \\ &\hat{\beta}^{(m)}_{boost,i} = 0 \quad \text{if } m_i = 0. \end{aligned}$$

Since $m_i = 0$ happens only if $|Z_i| \leq \frac{\delta}{\sqrt{1-(1-\nu)^2}}$, we can write the estimator as

$$\hat{\beta}^{(m)}_{boost,i} \approx \begin{cases} Z_i - \lambda, & \text{if } Z_i \geq \lambda, \\ 0, & \text{if } |Z_i| < \lambda, \\ Z_i + \lambda, & \text{if } Z_i \leq -\lambda. \end{cases} \tag{22}$$

where $\lambda = \frac{\delta}{\sqrt{1-(1-\nu)^2}}$ (note that m is connected to δ, and hence to λ via the criterion in (19)). This is the soft-threshold estimator with threshold λ, as in (7). By choosing $\delta = a_n\sigma_\varepsilon\sqrt{1-(1-\nu)^2}$, we get the desired threshold $\lambda = \lambda_n = a_n\sigma_\varepsilon$.

We will now deal with the approximation in (21). By the choice of δ two lines above, we would like that

$$(1-\nu)^{m_i} \approx a_n\sigma_\varepsilon/|Z_i|.$$

As we will see, this approximation is accurate when choosing ν small. We only have to deal with the case where $|Z_i| > a_n\sigma_\varepsilon$; if $|Z_i| \le a_n\sigma_\varepsilon$, we know that $m_i = 0$ and $\hat{\beta}_i = 0$ exactly, as claimed in the right hand side of (22). Denote by

$$V_i = V(Z_i) = \frac{a_n\sigma_\varepsilon}{|Z_i|} \in (0,1).$$

(The range $(0,1)$ holds for the case we are considering here). According to the stopping criterion in (20), the derivation as for (21) and the choice of δ, this says that

$$\begin{aligned}(1-\nu)^{m_i} &> V_i,\\ (1-\nu)^{m_i+1} &\le V_i,\end{aligned} \tag{23}$$

and hence

$$\begin{aligned}\Delta(\nu, V_i) &\stackrel{\text{def}}{=} ((1-\nu)^{m_i} - V_i) \le ((1-\nu)^{m_i} - (1-\nu)^{m_i+1})\\ &= \frac{\nu}{1-\nu}(1-\nu)^{m_i+1} \le \frac{\nu}{1-\nu}V_i,\end{aligned}$$

by using (23). Thus,

$$\begin{aligned}&(1-\nu)^{m_i} = V_i + ((1-\nu)^{m_i} - V_i) = V_i(1 + \Delta(\nu, V_i)/V_i) = V_i(1 + e_i(\nu)),\\ &|e_i(\nu)| = |\Delta(\nu, V_i)/V_i| \le \nu/(1-\nu).\end{aligned} \tag{24}$$

Thus, when multiplying with $(-1)Z_i$ and adding Z_i,

$$\begin{aligned}\hat{\beta}^{(m)}_{boost,i} &= (1-(1-\nu)^{m_i})Z_i = Z_i - Z_iV_i(1+e_i(\nu))\\ &= \text{soft-threshold estimator with threshold } \lambda_n = a_n\sigma_\varepsilon(1+e_i(\nu)),\end{aligned}$$

where $\max_{1\le i\le n}|e_i(\nu)| \le \nu/(1-\nu)$ as in (24). □

Proof of formula (8). The residual sum of squares, denoted by RSS_m for FSLR iteration m, decays monotonously as a function of m (unless $m_i\nu > |Z_i|$ for some i which we ignore since the algorithm would be stopped before this happens), and also the difference $RSS_m - RSS_{m+1}$ is decaying monotonously in m. Thus, as for L_2Boosting, every stopping of FSLR with an iteration number m corresponds to a tolerance δ^2 such that (19) from the proof of Theorem 1 holds.

The residual sum of squares is

$$RSS_m = \|\mathbf{Y} - \mathbf{X}\hat{\beta}^{(m)}_{FSLR}\|^2 = \|Z - \hat{\beta}^{(m)}_{FSLR}\|^2, \quad Z = \mathbf{X}^T\mathbf{Y}.$$

The correlations (or coefficient estimates $\hat{\gamma}$) at FSLR iteration m are

$$\hat{c} = \mathbf{X}^T(\mathbf{Y} - \mathbf{X}\hat{\beta}^{(m)}_{FSLR}) = Z - \hat{\beta}^{(m)}_{FSLR}.$$

Thus, $\text{sign}(\hat{c}_i) = \text{sign}(Z_i)$ (since we would stop FSLR before $\hat{\beta}_{FSLR,i}$ would take the opposite sign of Z_i) and

$$\hat{\beta}^{(m)}_{FSLR,i} = m_i\nu\,\text{sign}(Z_i),$$

where m_i denotes the number of times the ith predictor has been selected during the first m FSLR iterations, i.e. $\sum_{i=1}^{p} m_i = m$. Therefore,

$$RSS_m = \sum_{i=1}^{n} (Z_i - m_i \nu \, \mathrm{sign}(Z_i))^2.$$

Proceeding analogously as in the proof of Theorem 1, by noting that

$$\begin{aligned}
&(Z_i - m_i \nu \, \mathrm{sign}(Z_i))^2 - (Z_i - (m_i+1)\nu \, \mathrm{sign}(Z_i))^2 = 2\nu |Z_i| - 2m_i\nu^2 - \nu^2 \\
&= 2\nu \, \mathrm{sign}(Z_i)(Z_i - m_i \nu \, \mathrm{sign}(Z_i) - \nu \, \mathrm{sign}(Z_i)/2).
\end{aligned}$$

we get instead of formula (20),

$$\begin{aligned}
&2\nu \, \mathrm{sign}(Z_i)(Z_i - (m_i - 1)\nu \, \mathrm{sign}(Z_i) - \nu \, \mathrm{sign}(Z_i)/2) > \delta^2, \\
&2\nu \, \mathrm{sign}(Z_i)(Z_i - m_i \nu \, \mathrm{sign}(Z_i) - \nu \, \mathrm{sign}(Z_i)/2) \le \delta^2.
\end{aligned}$$

Rewriting this leads to

$$\begin{aligned}
\text{if } \mathrm{sign}(Z_i) = 1: \quad & (m_i - 1)\nu \, \mathrm{sign}(Z_i) < Z_i - \mathrm{sign}(Z_i)(\delta^2/(2\nu) + \nu/2), \\
& m_i \nu \, \mathrm{sign}(Z_i) \ge Z_i - \mathrm{sign}(Z_i)(\delta^2/(2\nu) + \nu/2), \\
\text{if } \mathrm{sign}(Z_i) = -1: \quad & (m_i - 1)\nu \, \mathrm{sign}(Z_i) > Z_i - \mathrm{sign}(Z_i)(\delta^2/(2\nu) + \nu/2), \\
& m_i \nu \, \mathrm{sign}(Z_i) \le Z_i - \mathrm{sign}(Z_i)(\delta^2/(2\nu) + \nu/2). \qquad (25)
\end{aligned}$$

Replacing first the "$\ge$" and "$\le$" in (25, 2nd and 4th line) by "$\approx$", we get approximately the claimed soft-threshold estimator:

$$\begin{aligned}
&\hat{\beta}^{(m)}_{FSLR,i} = m_i \nu \, \mathrm{sign}(Z_i) \approx Z_i - sign(Z_i)(\delta^2/(2\nu) + \nu/2) \\
&= \text{soft-threshold estimator in (7) with threshold } \delta^2/(2\nu) + \nu/2.
\end{aligned}$$

Choosing $\delta^2 = \lambda_n 2\nu$, this yields the approximate threshold $\lambda_n(1 + \nu/(2\lambda_n))$.

Dealing with the approximation "$\approx$", observe that in (25, with $\mathrm{sign}(Z_i) = 1$), $(m_i - 1)\nu \, \mathrm{sign}(Z_i)$ has a strict "$<$", and $m_i \nu \, \mathrm{sign}(Z_i)$ has a "$\ge$" relation (and vice versa for $\mathrm{sign}(Z_i) = -1$). Therefore, the approximation error is bounded by ν. This yields an exact relation to the soft-threshold estimator with the threshold of the form

$$\lambda_n\left(1 + \frac{\nu}{2\lambda_n} + \frac{\tilde{e}_i(\nu)}{\lambda_n}\right), \quad \max_{1 \le i \le n} |\tilde{e}_i(\nu)| \le \nu.$$

But this can be rewritten as $\lambda_n(1 + e_i(\nu)/\lambda_n)$ with $e_i(\nu)$ uniformly bounded by $3\nu/2$ which completes the proof. $\square$

Proof of Theorem 2. The proof is based on similar ideas as for Theorem 1. The MS-L_2Boosting in iteration m aims to minimize

$$MSB_m = RSS_m + \gamma_n \mathrm{trace}(\mathcal{B}_m) = \|\mathbf{Y} - \mathbf{X}\hat{\beta}^{(m)}_{ms-boost}\|^2 + \gamma_n \mathrm{trace}(\mathcal{B}_m).$$

When using the orthogonal transformation by multiplying with $\mathbf{X}^T$, the criterion above becomes

$$MSB_m = \|Z - \hat{\beta}^{(m)}_{ms-boost}\|^2 + \gamma_n \mathrm{trace}(\mathcal{B}_m),$$

where $\mathrm{trace}(\mathcal{B}_m) = \sum_{i=1}^{n}(1-(1-\nu)^{m_i})$. Moreover, we run MS-$L_2$Boosting until the stopping iteration m satisfies the following:

$$\begin{aligned} &MSB_k - MSB_{k+1} > 0,\ k = 1, 2, \ldots, m-1, \\ &MSB_m - MSB_{m+1} \le 0. \end{aligned} \tag{26}$$

It is straightforward to see for the orthonormal case, that such an m coincides with the definition for $\hat{m}$ in section 3.2. Since MS-L_2Boosting changes only one of the summands in RSS and the trace of $\mathcal{B}_m$, the criterion above implies that for all $i = 1, \ldots, n$, using the definition of MSB,

$$\begin{aligned} &(1-\nu)^{2(m_i-1)} Z_i^2 (1-(1-\nu)^2) - \gamma_n \nu (1-\nu)^{m_i-1} > 0, \\ &(1-\nu)^{2m_i} Z_i^2 (1-(1-\nu)^2) - \gamma_n \nu (1-\nu)^{m_i} \le 0. \end{aligned} \tag{27}$$

Note that if $|Z_i|^2 \le \gamma_n\nu/(1-(1-\nu)^2)$, then $m_i = 0$. This also implies uniqueness of the iteration m such that (26) holds or of the m_i such that (27) holds.

Similarly to the proof of Theorem 1, we look at this expression first in terms of an approximate equality to zero, i.e. ≈ 0. We then immediately find that

$$(1-\nu)^{m_i} \approx \frac{\gamma_n \nu}{(1-(1-\nu)^2)|Z_i|^2}.$$

Hence,

$$\begin{aligned} \hat{\beta}^{(m)}_{ms-boost,i} &= (\mathbf{X}^T \mathcal{B}_m \mathbf{Y})_i = (\mathbf{X}^T \mathbf{X} D_m \mathbf{X}^T \mathbf{Y})_i = (D_m Z)_i = (1-(1-\nu)^{m_i}) Z_i \\ &\approx Z_i - \frac{\gamma_n \nu Z_i}{(1-(1-\nu)^2)|Z_i|^2} = Z_i - sign(Z_i)\frac{\gamma_n}{2-\nu}\frac{1}{|Z_i|}. \end{aligned}$$

The right-hand side is the nonnegative garrote estimator as in (15) with threshold $\gamma_n/(2-\nu)$.

Dealing with the approximation "$\approx$" can be done similarly as in the proof of Theorem 1. We define here

$$V_i = V(Z_i) = \frac{\gamma_n \nu}{(1-(1-\nu)^2)|Z_i|^2}.$$

We then define $\Delta(\nu, V_i)$ and $e_i(\nu)$ as in the proof of Theorem 1, and we complete the proof as for Theorem 1. □

Acknowledgments. B. Yu would like to acknowledge gratefully the partial supports from NSF grants FD01-12731 and CCR-0106656 and ARO grant DAAD19-01-1-0643, and the Miller Research Professorship in Spring 2004 from the Miller Institute at University of California at Berkeley. Both authors thank Dr. David Mease and Professor Leo Breiman for their very helpful comments and discussions on the paper.

References

[1] Akaike, H. (1970). Statistical predictor identification. Ann. Inst. Statist. Math. **22** 202–217.

[2] Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In *2nd International Symposium on Information Theory*, (eds. P.N. Petrov, F. Csàki), pp. 267–81. Akademiai Kiàdo, Budapest.

[3] Akaike, H. (1974). A new look at the statistical model identification. IEEE Trans. Automat. Control **19**, 716–723.

[4] Breiman, L. (1995). Better subset regression using the nonnegative garrote. Technometrics **37**, 373–384.

[5] Breiman, L. (1996). Bagging predictors. Machine Learning **24**, 123–140.

[6] Breiman, L. (1998). Arcing classifiers. Ann. Statist. **26**, 801–849 (with discussion).

[7] Bühlmann, P. (2004). Boosting for high-dimensional linear models. Preprint.

[8] Bühlmann, P. and Yu, B. (2003). Boosting with the L_2loss: regression and classification. J. Amer. Statist. Assoc. **98**, 324–339.

[9] Donoho, D.L. and Johnstone, I.M. (1994). Ideal spatial adaptation by wavelet shrinkage. Biometrika **81**, 425–455.

[10] Efron, B., Hastie, T., Johnstone, I. and Tibshirani, R. (2004). Least angle regression. Ann. Statist. **32**, 407–499 (with discussion).

[11] Freund, Y. and Schapire, R.E. (1996). Experiments with a new boosting algorithm. In Machine Learning: Proc. Thirteenth International Conference, pp. 148–156. Morgan Kauffman, San Francisco.

[12] Friedman, J.H. (1991). Multivariate adaptive regression splines (with Discussion). Ann. Statist. **19**, 1–141 (with discussion).

[13] Friedman, J.H. (2001). Greedy function approximation: a gradient boosting machine. Ann. Statist. **29**, 1189–1232.

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

[15] Green, P.J. and Silverman, B.W. (1994). *Nonparametric Regression and Generalized Linear Models*. Chapman and Hall.

[16] Hansen, M. and Yu, B. (2001). Model selection and minimum description length principle. J. Amer. Statist. Assoc. **96**, 746–774.

[17] Hurvich, C.M., Simonoff, J.S. and Tsai, C.-L. (1998). Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion. J. Roy. Statist. Soc., Ser. B, **60**, 271–293.

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

[19] Jiang, W. (2004). Process consistency for AdaBoost. Ann. Statist. **32**, 13–29 (disc. pp. 85–134).

[20] Lugosi, G. and Vayatis, N. (2004). On the Bayes-risk consistency of regularized boosting methods. Ann. Statist. **32**, 30–55 (disc. pp. 85–134).

[21] Madigan, D. and Ridgeway, G. (2004). Discussion on "Least angle regression" (Efron, B., Hastie, T., Johnstone, I. and Tibshirani, R.). Ann. Statist. **32**, 465–469.

[22] Mallat, S and Zhang, Z. (1993). Matching pursuits with time-frequency dictionaries. IEEE Trans. Signal Proc. **41**, 3397–3415.

[23] Mannor, S., Meir, R. and Zhang, T. (2002). The consistency of greedy algorithms for classification. In *Computational Learning Theory: 15th Annual Conference on Computational Learning Theory, COLT 2002, Proceedings* (eds. J. Kivinen, R.H. Sloan), pp. 319–333. Lecture Notes in Computer Science **2375**. Springer.

[24] Mohammadi, L. and van de Geer, S.A. (2002). On nonnegative garrote estimator in a linear regression model. Preprint.

[25] Schwartz, G. (1978). Estimating the dimension of a model. Ann. Statist. **6**, 461–464.

[26] Shibata, R. (1981). An optimal selection of regression variables. Biometrika **68**, 45–54.

[27] Sugiura, N. (1978). Further analyses of the data by Akaike's information criterion and the finite corrections. Comm. Statist. **A7**, 13–26.

[28] Speed, T. and Yu, B. (1994). Model selection and prediction: normal regression. Ann. Inst. Statist. Math. **45**, 35–54.

[29] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. J. Roy. Statist. Soc., Ser. B, **58**, 267–288.

[30] Tukey, J.W. (1977). *Exploratory data analysis.* Addison-Wesley, Reading, MA.

[31] Zhang, T. and Yu, B. (2003). Boosting with early stopping: convergence and consistency. To appear in Ann. Statist. Available from http://www.stat.berkeley.edu/users/binyu/publications.html

[32] Zhao, P. and Yu, B. (2004). Boosted lasso and reverse boosting. Tech. Report, Dept. Statist., UC Berkeley.