# Consistency for $L_2$Boosting and Matching Pursuit with Trees and Tree-type Basis Functions

Peter Bühlmann
ETH Zürich, Switzerland

October 2002

**Abstract**

We present new consistency results in regression and classification for $L_2$Boosting, a powerful variant of boosting with the squared error loss function. For any dimension of the predictor, a square-integrable regression or an arbitrary conditional probability function, potentially discontinuous, can be consistently estimated with $L_2$Boosting using tree-type learners. We also discuss close connections to matching pursuits for basis functions in signal processing and demonstrate differences between tree and rectangle indicator basis functions. Depending on the signal to noise ratio, one of them will be better than the other and we thus get additional flexibility to tune boosting to high or low noise problems.

**Key Words:** basis selection, Bayes risk consistency, boosting, multiple decision trees, nonparametric classification, nonparametric regression

**Heading:** Consistency of Boosting

## 1 Introduction

Freund and Schapire's (1996) AdaBoost algorithm for classification has attracted much attention in the machine learning community (Schapire, 2002 and the references therein) as well as in related areas in statistics (Breiman, 1998; Friedman et al., 2000), mainly because of its good empirical performance in a variety of data sets. Boosting methods have been originally introduced as multiple prediction schemes, averaging estimated predictions from re-weighted data. Later Breiman (1998, 1999) then noted first that the AdaBoost algorithm can be viewed as a gradient descent optimization technique in function space. This important insight opened a new perspective, namely to use boosting methods in other contexts than classification. Friedman (2001) developed boosting methods for regression. He also considers the case where boosting is implemented as an optimization with the squared error loss function: this is what we call $L_2$Boosting. It is essentially the same as Mallat and Zhang's (1993) matching pursuit algorithm in signal processing.

Theoretical results for the AdaBoost and other boosting algorithms include VC-type bounds for the generalization error in classification (cf. Koltchinskii and Panchenko, 2002) or consistency for approximating the theoretically optimal Bayes procedure (Jiang, 2000; Lugosi and Vayatis, 2001; Mannor et al., 2002). For $L_2$Boosting in regression, only very few theoretical results exist, maybe because regression is in a sense more difficult than classification (cf. Devroye et al., 1996). Bühlmann and Yu (2001) have shown that $L_2$Boosting with smoothing spline learners for one-dimensional curve estimation achieves the minimax mean squared error rate asymptotically. Although this is the only asymptotic result about optimal minimax error rates in the boosting literature, it only deals (in theory) with one-dimensional curve estimation.

We present here consistency results for $L_2$Boosting in regression and classification with regression tree learners and versions thereof, where the nonparametric regression or conditional probability function $f(x) : \mathbb{R}^p \to \mathbb{R}$ allows for any dimension $p$ of the predictor. This boosting method is Friedman's MART procedure (Multiple Additive Regression Trees), and we thus prove here its consistency. Our consistency result for classification is comparable to Jiang's (2000) theoretical work on AdaBoost. However, we do not need any smoothness assumption on the underlying conditional probability functions whereas Jiang does. In particular, we allow for jumps which is necessary for classification problems with Bayes error zero. Because we consider regression trees as learners in $L_2$Boosting, we need an assumption requiring a growing number of observations in every terminal node in the tree as sample size increases. While such a requirement is very plausible for regression trees, we show here also how it turns out to become unnecessary for classification tree learners. But we argue in section 4 that classification tree learners can be very inefficient for the more ambitious goal of estimating the conditional probability functions instead of just classification. All of the other theoretical consistency results on some of the boosting algorithms consider the simpler case with classification tree learners (Breiman, 2000; Jiang, 2000; Lugosi and Vayatis, 2001), and they bypass the problem about very unbalanced trees; an exception is here the analysis by Mannor et al. (2002), but related to our restriction, they also require some regularity in terms of the magnitude of the coefficients in the convex combination of the base learner.

We also point out that our analysis is for a computationally efficient boosting algorithm which has to be regularized via early stopping of the iterations in the gradient descent optimization technique, like the original AdaBoost algorithm. This is computationally attractive since the minimum of the estimated test set error, e.g. by cross-validation, can be found effectively by running the boosting algorithm once, even not until numerical convergence. More recent works (cf. Lugosi and Vayatis, 2001; Mannor et al., 2002) have considered regularization of boosting by constraining the magnitude of the coefficients in the convex combination of the base learner. When doing so, the major computational advantages of boosting is lost: for every candidate regularization parameter, a regularized boosting algorithm has to be run until numerical convergence, and thus to find the best regularization, we need to run regularized boosting very many times. The theoretical analysis of computationally efficient versions of boosting, which seems more difficult than for regularized boosting, has otherwise only been considered by Jiang (2000) for the AdaBoost algorithm.

We also include in section 4 new insights for more fundamental understanding of the role of the base learner, here always in terms of tree-type learners. A connection to the matching pursuit algorithm in signal processing (Mallat and Zhang, 1993) is given, which is exactly equal to $L_2$Boosting if the learner is an estimated projection onto a basis function from a typically over-complete library of functions. This helps to identify cases, where projection onto single rectangular indicator functions is better than regression tree learners and vice-versa.

## 2 $L_2$Boosting and Matching Pursuit algorithm

Consider data $(X_1, Y_1), \ldots, (X_n, Y_n)$ with $X_i \in \mathbb{R}^p$ and $Y_i \in \mathbb{R}$ for regression or $Y_i \in \{0, 1\}$ for 2-class classification. The goal is to estimate the conditional mean $f(x) = \mathbb{E}[Y|X = x]$: for classification, this is just the conditional probability function $f(x) = \mathbb{P}[Y = 1|X = x]$.

We have a base procedure, called also the base learner, which fits an arbitrary response vector $U = (U_1, \ldots, U_n)' \in \mathbb{R}^n$ by least squares:

$$\hat{g}(\cdot) = \mathrm{argmin}_{g \in \mathcal{G}} \sum_{i=1}^{n} (U_i - g(X_i))^2, \tag{2.1}$$

where minimization is over a chosen class $\mathcal{G}$ of real-valued functions. We consider here most often the case of regression trees. A $d$ terminal nodes regression tree function is of the form

$$g_{tree}(x) = g_{\mathcal{P};\theta}(x) = \sum_{j=1}^{d} \theta_j \mathbf{1}_{[x \in \mathcal{R}^{(j)}]},\ \cup_{j=1}^{d} \mathcal{R}^{(j)} = \mathbb{R}^p,\ \mathcal{R}^{(j)} \cap \mathcal{R}^{(k)} = \emptyset\ (j \neq k), \tag{2.2}$$

where $\mathcal{P} = \{\mathcal{R}^{(j)}; j = 1, \ldots d\}$ denotes a partition and $\theta_j \in \mathbb{R}$ are coefficients. The partition $\mathcal{P}$ is representable in form of a binary tree (see Breiman et al., 1984), and each partition cell is a rectangle of the form $\mathcal{R} = (a_1, b_1] \times \cdots \times (a_p, b_p]$ in $\mathbb{R}^p$ with $\infty \leq a_j < b_j \leq \infty$. When fitting the response variables $U = (U_1, \ldots, U_n)'$ by least squares,

$$\hat{\theta}_j = \sum_{i=1}^{n} U_i \mathbf{1}_{[X_i \in \hat{\mathcal{R}}^{(j)}]} / \sum_{i=1}^{n} \mathbf{1}_{[X_i \in \hat{\mathcal{R}}^{(j)}]}\ (j = 1, \ldots, d), \tag{2.3}$$

and the $\hat{\mathcal{R}}^{(j)}$'s are from recursive binary splitting with the squared error loss function, see Breiman et al. (1984).

## $L_2$Boosting algorithm

*Step 1 (initialization).* Start with $F_n^{(0)} \equiv 0$. Set $r = 0$.

*Step 2.* Compute residuals $U_i = Y_i - \hat{F}_n^{(r)}(X_i)$ $(i = 1, \ldots, n)$ and fit the real-valued base procedure to the current residuals by least squares, as in (2.1),

$$\hat{g}_n^{(r+1)}(\cdot) = \hat{g}_{n;U}^{(r+1)}(\cdot)$$

where the notation emphasizes the dependence on the current residuals $U = (U_1, \ldots, U_n)'$. Update

$$\hat{F}_n^{(r+1)}(\cdot) = \hat{F}_n^{(r)}(\cdot) + \hat{g}_n^{(r+1)}(\cdot).$$

*Step 3 (iteration).* Increase the iteration index $r$ by one and repeat Step 2 until $r$ reaches a pre-specified stopping value $m$.

$L_2$Boosting is thus nothing else than repeated least squares fitting of residuals (cf. Friedman, 2001). With $m = 2$, it has already been proposed by Tukey (1977) under the name "twicing".

For continuous response variables $Y_i \in \mathbb{R}$, a regression estimate for $f(x) = \mathbb{E}[Y|X = x]$ is directly given by the $L_2$Boosting-estimate $\hat{F}_n^{(m)}(\cdot)$. For a two-class problem with $Y \in \{0, 1\}$, a classifier under equal misclassification costs is given by

$$\mathbf{1}_{[\hat{F}_n^{(m)}(\cdot) > 1/2]}.$$

Empirically, $L_2$Boosting has been demonstrated to be very competitive in both regression and classification, particularly if the predictor is high-dimensional, cf. Friedman (2001) and Bühlmann and Yu (2001).

We can view the $L_2$Boosting algorithm in terms of fitting functions from $\mathbb{R}^p \to \mathbb{R}$ from a given class,

$$\mathcal{G} = \{g_{\gamma;\theta}(\cdot); \gamma \in \Gamma\},$$

where $\Gamma$ is discrete, indexing the family of functions $\mathcal{G}$, and $\theta = \theta_\gamma$ denotes the unknown parameters for each function $g_\gamma$. For example with tree functions in (2.2), $\Gamma$ is the set of all rectangle partitions $\mathcal{P} = \{\mathcal{R}^{(j)};\ j = 1, \ldots, d\}$ (with corner points of $\mathcal{R}^{(j)}$ between observed predictor variables) which can be represented by a binary tree with $d$ terminal nodes, and $\theta = \theta_{\mathcal{P}}$ are the unknown coefficients $\theta_1, \ldots, \theta_d$. The $L_2$Boosting algorithm then selects in every iteration step (Step 2) the function $g_{\hat{\gamma};\theta}(\cdot)$ such that

$$\hat{\gamma} = \text{argmin}_{\gamma \in \Gamma} \sum_{i=1}^{n} (U_i - g_{\gamma;\hat{\theta}}(X_i))^2 \tag{2.4}$$

where $\hat{\theta} = \hat{\theta}_\gamma$ is the least squares estimate (for given $\gamma$). The distinction between selecting a function with discrete $\hat{\gamma}$ in (2.4) and parameter estimation $\hat{\theta} = \hat{\theta}_\gamma$ will become more important in section 4. The selection aspect in (2.4) also connects to the familiar setting of matching pursuit, namely basis selection from an over-complete library, see Mallat and Zhang (1993).

## 3 Consistency

Our proof for consistency of $L_2$Boosting with regression trees requires only few and simple regularity conditions. We distinguish between nonparametric regression and classification. For the former, we assume the data generating model

$$Y_i = f(X_i) + \varepsilon_i,\ X_i \in \mathbb{R}^p,\ f : \mathbb{R}^p \to \mathbb{R}\ \ (i = 1, \ldots, n), \tag{3.1}$$

where $X_1, \ldots, X_n$ are i.i.d., $\varepsilon_1, \ldots, \varepsilon_n$ are i.i.d. and independent from $\{X_s; 1 \le s \le n\}$, satisfying $\mathbb{E}[\varepsilon_i] = 0$. For two-class classification with $Y_i \in \{0, 1\}$ we assume

$$(X_1, Y_1), \ldots, (X_n, Y_n) \text{ i.i.d.},\ X_i \in \mathbb{R}^p,\ Y_i \in \{0, 1\}. \tag{3.2}$$

For regression, we impose a moment condition.

(A) In case of the regression model (3.1), $\sigma_\varepsilon^2 = \mathbb{E}|\varepsilon_i|^2 < \infty$ and $\mathbb{E}|f(X_i)|^2 < \infty$.

Furthermore, when using regression tree functions as base procedure, we require:

(B) Fitted tree functions with $d$ terminal nodes are restricted to partitions $\{\mathcal{R}^{(j)};\ j = 1, \ldots, d\}$ such that there are at least $N_{node}$ observations in every partition cell $\mathcal{R}^{(j)}$ $(j = 1, \ldots, d)$ satisfying $n^{-2/3} N_{node} \to \infty$.

Condition (B) requires a reasonable amount of observations per terminal node in a fitted tree, in order to guarantee sufficiently fast convergence of estimated means in terminal nodes. The value $N_{node}$ equals the "minbucket" parameter in the function `rpart` from the statistical software package $R$, publicly available at `http://www.r-project.org/`. It is quite natural to have such a constraint when boosting regression trees. In case of boosting $\pm 1$-trees (see (4.2)), i.e. classification trees, this restriction is not necessary anymore, see also Remark 2 below for further discussion.

**Theorem 1** *Consider model (3.1) for regression satisfying assumption (A). Then, for $L_2$Boosting with regression trees having $d = p + 1$ terminal nodes, satisfying (B), and with boosting iterations $m_n \to \infty$, $m_n = o(\log(n))$,*

$$\mathbb{E}_X |\hat{F}_n^{(m_n)}(X) - f(X)|^2 = o_P(1)\ \ (n \to \infty),$$

*where $X$ is a new predictor variable, independent from but with the same distributions as the data.*

A proof is given in section 6. The restriction about the increase of $m_n$ as $n \to \infty$ is probably far from the fastest possible whose exploration is beyond our scope. But our result is still a bit more general than Jiang's (2000) for consistency of AdaBoost, where only the non-constructive existence of a sufficiently slowly increasing $m_n$ is used to prove consistency.

Our assumption in (A) allows for non-smooth regression functions $f(\cdot)$ and for arbitrary predictors $X$, including finite categorical variables which are very common when modeling with trees.

**Corollary 1** *Consider the setting of Theorem 1 but using $L_2$Boosting with regression trees having $d < p + 1$ terminal nodes satisfying (B). Then,*

$$\mathbb{E}_X|\hat{F}_n^{(m_n)}(X) - f^*(X)|^2 = o_P(1) \ (n \to \infty),$$

*where*

$$f^* = argmin_{g \in T_d} \mathbb{E}|f(X) - g(X)|^2,$$

*and $T_d$ is the space of functions with additive and interaction terms up to order $d-1$:*
$T_d = \{g \in L_2(P);$
$g(x) = \sum_{j=1}^{p} g_j(x) + \sum_{j_1,j_2=1}^{g} g_{j_1 j_2}(x_{j_1}, x_{j_2}) + \ldots + \sum_{j_1,\ldots,j_{d-1}=1}^{p} g_{j_1 \ldots j_{d-1}}(x_{j_1}, \ldots, x_{j_{d-1}})$

A proof is given in the Appendix. In case of $d = 2$, the Corollary says that $L_2$Boosting with regression stumps consistently estimates the best approximating additive (in the predictor variables) function $f^*$.

**Theorem 2** *Consider model (3.2) for two-class classification. Then, for $L_2$Boosting with regression trees having $d = p+1$ terminal nodes, satisfying (B), and with boosting iterations $m_n \to \infty$, $m_n = o(\log(n))$,*

$$\mathbb{E}_X|\hat{F}_n^{(m_n)}(X) - f(X)|^2 = o_P(1) \ (n \to \infty), \quad f(x) = \mathbb{P}[Y = 1|X = x],$$

*and*

$$|\mathbb{P}_{(X,Y)}[Y \neq \mathbf{1}_{[\hat{F}_n^{(m_n)}(X) > 1/2]}] - Bayes\ risk| = o_P(1) \ (n \to \infty),$$

*where $(X, Y)$ are new variables, independent from but with the same distribution as the data.*

Proof: The first assertion follows exactly as in the proof of Theorem 1, although the equivalent regression model $Y_i = f(X_i) + \varepsilon_i$ has heteroscedastic errors $\varepsilon_i = Y_i - f(X_i)$ (in Lemma 2, one cannot isolate $\sigma_\varepsilon^2$ from $\|R_n^{m_n} f\|_2^2$ but this cancels out later in (6.6)). It is well known (cf. Theorem 2.3 of Devroye et al., 1996) that the misclassification risk is upper bounded by 2 times the $L_1$-norm and hence by 2 times the $L_2$-norm. $\square$

Theorem 2 establishes the Bayes risk consistency of $L_2$Boosting with decision trees having $p + 1$ terminal nodes. There is no further assumption besides the i.i.d. structure of the data pairs in (3.2). In particular, we allow for non-smooth conditional probability functions $p(\cdot)$ and thus classification problems having Bayes error zero; for AdaBoost, Jiang (2000) does not cover that case and he also assumes a probability density for the predictor.

**Remark 1.** Theorems 1, 2 and Corollary 1 also hold for learners with rectangle indicator basis functions (see (4.1)) which are sometimes better than trees, as discussed in section 4.

**Remark 2.** Theorems 1, 2 and Corollary 1 also hold for $\pm 1$-trees (see (4.2)) without imposing condition (B). This, because $\pm 1$-trees are complete in $L_2(P)$ (Breiman, 2000) and any $\pm 1$-tree satisfies $\|g_{\pm 1-tree}\|_2 = n^{-1} \sum_{i=1}^{n} g_{\pm 1-tree}(X_i)^2 = 1$ which simplifies the proofs in section 6 considerably. Thus consistency can be achieved without assumption (B). However, we do not advise to use $\pm 1$-trees in the regression context nor in $L_2$Boost or LogitBoost (Friedman et al, 2000) for classification problems where conditional probabilities are the target. For further discussion, see section 4.5.3.

## 4 Trees, $\pm 1$-trees and rectangle indicator basis functions

We develop here more fundamental understanding how $L_2$Boosting depends on the kind of base procedure. We focus exclusively on tree-type base procedures which are still the most popular so far. Already there, we will see that trees can be decomposed which may be advantageous in some cases. As in section 2, we always denote by $\mathcal{R} = (a_1, b_1] \times \cdots \times (a_p, b_p]$ with $\infty \leq a_j < b_j \leq \infty$ a rectangle in $\mathbb{R}^p$. By computational reasons, we typically constrain $\mathcal{R}$ to be representable as a terminal node in a binary tree, see Breiman et al. (1984).

When taking linear combinations of $\ell$ regression tree functions in (2.2), as in boosting, we fit $\ell \cdot d$ coefficients, see (2.3), and it is therefore not a linear combination of $\ell$ basis functions $g_k$, $\sum_{k=1}^{\ell} \beta_k g_k(\cdot)$ $(\beta_k \in \mathbb{R})$, involving $\ell$ parameters $\beta_k$ only. Thus, regression tree functions as in (2.2) are not basis functions in the process of $L_2$Boosting.

Tree functions in (2.2) have related cousins which are basis functions. One of them are indicators of rectangles

$$g_{rect}(x) = g_{\mathcal{R}}(x) = \mathbf{1}_{[x \in \mathcal{R}]}. \tag{4.1}$$

The other basis functions are $\pm 1$-trees

$$\begin{aligned} &g_{\pm 1-tree}(x) = g_{\mathcal{P},\eta}(x) = \sum_{j=1}^{d} \eta_j \mathbf{1}_{[x \in \mathcal{R}^{(j)}]}, \;\; \eta_j \in \{-1, +1\}, \\ &\mathcal{P} = \{\mathcal{R}^{(j)}; j = 1, \ldots, d\}, \; \cup_{j=1}^{d} \mathcal{R}^{(j)} = \mathbb{R}^p, \; \mathcal{R}^{(j)} \cap \mathcal{R}^{(k)} = \emptyset \; (j \neq k). \end{aligned} \tag{4.2}$$

When fitting response variables $U = (U_1, \ldots, U_n)' \in \mathbb{R}^n$, we view the functions in (4.1) or (4.2) as single basis functions and multiply them by the single coefficient

$$\hat{\beta} = \sum_{i=1}^{n} Y_i g_{rect}(X_i) / \sum_{i=1}^{n} g_{rect}(X_i)^2 \text{ for (4.1)}, \tag{4.3}$$

$$\hat{\beta} = n^{-1} \sum_{i=1}^{n} Y_i g_{\pm 1-tree}(X_i) \text{ for (4.2), respectively.} \tag{4.4}$$

Note that $n^{-1} \sum_i g_{\pm 1-tree}(X_i)^2 = 1$.

### 4.1 A population version in Hilbert space: Matching Pursuit

To understand differences among various tree-type learners in boosting, we focus first on a population version of $L_2$Boosting. It has an elegant formalization in the Hilbert space $L_2(P) = \{f; \|f\|_2^2 = \int f(x)^2 dP(x) < \infty\}$ with inner product $\langle f, g \rangle = \int f(x) g(x) dP(x)$. Here, the probability measure $P$ is generating the predictor $X$ in model (3.1) or (3.2).

We closely follow the analysis of Mallat and Zhang's (1993) matching pursuit algorithm. Consider first for simplicity the rectangle indicator basis function $g_{\mathcal{R}}(\cdot)$ as in (4.1). Define the following sequence of remainder functions, called matching pursuit algorithm:

$$\begin{aligned} &R^0 f = f, \\ &R^m f = R^{m-1} f - \frac{\langle R^{m-1} f, g_{\mathcal{R}_m} \rangle}{\|g_{\mathcal{R}_m}\|_2^2} g_{\mathcal{R}_m}, \; m = 1, 2, \ldots \end{aligned} \tag{4.5}$$

where $\mathcal{R}_m$ is chosen as

$$\mathcal{R}_m = \mathrm{argmax}_{\mathcal{R}} \frac{|\langle R^{m-1}f, g_{\mathcal{R}}\rangle|}{\|g_{\mathcal{R}}\|_2^2}. \tag{4.6}$$

It follows that

$$f = \sum_{j=0}^{m-1} \frac{\langle R^j f, g_{\mathcal{R}_{j+1}}\rangle}{\|g_{\mathcal{R}_{j+1}}\|_2^2} g_{\mathcal{R}_{j+1}} + R^m f,$$

and

$$\|R^m f\|_2^2 = \|R^{m-1}f\|_2^2 - \frac{|\langle R^{m-1}f, g_{\mathcal{R}_m}\rangle|^2}{\|g_{\mathcal{R}_m}\|_2^2}. \tag{4.7}$$

Thus, $\|R^m f\|_2$ is monotonely decreasing, and since the family $\{g_{\mathcal{R}}\}$ is complete in $L_2(P)$ (by requiring $p+1$ terminal nodes in the binary tree construction), convergence to zero as $m \to \infty$ follows from Mallat and Zhang (1993) . In case of regression tree functions (2.2), convergence of the population version holds as well (one rectangle indicator basis function is one tree function with $d-1$ parameters $\theta_j$ equal to 0).

From the pure approximation point of view in the population setting, the rectangle indicator basis functions are the most flexible for matching pursuit. To elaborate this, instead of choosing a whole partition $\{\mathcal{R}^{(j)};\ j = 1, \ldots, d\}$ as in (2.2) or (4.2) once, based on the *same* remainder function, it chooses $d$ rectangles one at a time such that in every of the $d$ matching pursuit (or boosting) iterations, the *most current* remainder function is matched best.

For example, two matching pursuit steps with rectangle indicator basis functions, when starting from the remainder $R^m f$, then look like

$$\begin{aligned}
R^{m+1}_{rect} f &= R^m f - \frac{\langle R^m f, g_{\mathcal{R}_{m+1}}\rangle}{\|g_{\mathcal{R}_{m+1}}\|_2^2} g_{\mathcal{R}_{m+1}}, \\
R^{m+2}_{rect} f &= R^{m+1}_{rect} f - \frac{\langle R^{m+1}_{rect} f, g_{\mathcal{R}_{m+2}}\rangle}{\|g_{\mathcal{R}_{m+2}}\|_2^2} g_{\mathcal{R}_{m+1}} \\
&= R^m f - \frac{\langle R^m f, g_{\mathcal{R}_{m+1}}\rangle}{\|g_{\mathcal{R}_{m+1}}\|_2^2} g_{\mathcal{R}_{m+1}} - \frac{\langle R^{m+1}_{rect} f, g_{\mathcal{R}_{m+2}}\rangle}{\|g_{\mathcal{R}_{m+2}}\|_2^2} g_{\mathcal{R}_{m+2}},
\end{aligned}$$

where $\mathcal{R}_{m+j}$ $(j = 1, 2)$ are chosen as in (4.6); in particular, $\mathcal{R}_{m+2}$ depends on the most current remainder function $R^{m+1}_{rect} f$ and *not* directly on $R^m f$. On the other hand with regression stumps, i.e. a tree in (2.2) with $d = 2$, one population $L_2$Boosting step also subtracts a linear combination of two indicator functions from the remainder

$$R^{m+1}_{stumps} f = R^m f - \frac{\langle R^m f, g_{\mathcal{R}^{(1)}_{m+1}}\rangle}{\|g_{\mathcal{R}^{(1)}_{m+1}}\|_2^2} g_{\mathcal{R}^{(1)}_{m+1}} - \frac{\langle R^m f, g_{\mathcal{R}^{(2)}_{m+1}}\rangle}{\|g_{\mathcal{R}^{(2)}_{m+1}}\|_2^2} g_{\mathcal{R}^{(2)}_{m+1}},$$

where $\mathcal{R}^{(2)}_{m+1} = \mathbb{R}^p \setminus \mathcal{R}^{(1)}_{m+1}$ is just the complement of $\mathcal{R}^{(1)}_{m+1}$ and the whole partition $\{\mathcal{R}^{(1)}_{m+1}, \mathcal{R}^{(2)}_{m+1}\}$ is determined by $R^m f$. Thus, two steps of matching pursuit with rectangle indicator basis functions in (4.1) is more greedy than doing one matching pursuit with one of the tree functions in (2.2): the former pursues maximal norm reduction in every step as described by (4.6) and ((4.7).

Like tree functions in (2.2), the $\pm 1$-trees in (4.2) are determining the partition $\{\mathcal{R}_{m+1}^{(j)}; j = 1, \dots d\}$ with one remainder function (denoted by $R^m f$ above) but their associated parameter is only one-dimensional (see (4.3)) which makes a fitted $\pm 1$-tree more parsimonious, but less flexible, than a regression tree.

Instead of classical binary trees, we can also consider $k$-ary (regression) trees where $k > 2$ branches grow out from every internal node in the tree, all of them corresponding to the same predictor variable. For example, a 3-ary tree could involve 3 out-growing branches which are determined by $x_{\hat{\iota}} \le \hat{\kappa}_1, \hat{\kappa}_1 < x_{\hat{\iota}} \le \hat{\kappa}_2, \hat{\kappa}_2 < x_{\hat{\iota}}$ using an estimated component $\hat{\iota} \in \{1, \dots, p\}$ and two estimated split points $\hat{\kappa}_1, \hat{\kappa}_2$. $k$-ary trees are still yielding functions of the form (2.2) but with partitions restricted to be representable as a $k$-ary tree. We will discuss below which types of trees or rectangle indicators are most useful and when.

## 4.2 Finite samples and stochastic noise

In case of finite samples and stochastic noise, selection of the best element in (2.4) from the base procedure has a cost and adds variance to the procedure.

It is known that, if the underlying regression function and the density of the predictor $X$ are smooth, the estimated split point $\hat{\kappa}$ in a tree has cube-root convergence rate $n^{-1/3}$ (Bühlmann and Yu, 2002) which is slower than $n^{-1/2}$ for the estimated parameters in (2.3), (4.3) or (4.4). Therefore, tree-type procedures with an overall low number of selections in (2.4), but possibly fitting a function with more parameters for each selected member in (2.4), can become better than fitting basis functions many times, each involving only a one-dimensional parameter. For example, pursuing the strategy with rectangle indicator basis functions in (4.1) can have high variance due to selection of $d$ rectangles in (2.4) one at a time: if every rectangle is constructed by say 2 split points and splitting variables (second order interactions), we would estimate $2d$ split points (e.g. for $d = 3$ equal to 6). On the other hand with regression trees in (2.2) or $\pm 1$-trees in (4.2), we only select one partition $\mathcal{P}$ which amounts to at most $d - 1$ estimated split points (e.g. for $d = 3$ equal to 2 which is three times less than with rectangle indicator basis functions). Particularly in high noise settings (and small to moderate sample size), the cost of selection in (2.4) (in every $L_2$Boosting iteration) comes into play, more precisely the slow $n^{-1/3}$ convergence rate. And vice versa, in low noise settings, or in case of very large sample size, the variance in estimating split points becomes negligible relative to "bias" in terms of non-greedy proceeding (systematically different from the maximal norm reduction in one step as in (4.7)), and $L_2$Boosting with rectangle indicator basis functions in (4.1) can be better than using trees.

We give now some summarizing heuristics why some of the tree-type functions are almost never appropriate and when some of them are better than others.

**Regression tree functions in (2.2).** As outlined above, they can be most efficient in high noise or small sample size settings. See also Figures 1-3.

**Rectangle indicator basis functions in (4.1).** As outlined above, they can be most efficient in low noise or large sample size settings. See also Figures 1-3.

**$\pm$1-trees in (4.2).** They are generally expected to be inferior than regression trees in (2.2) for the task of regression or estimating conditional probabilities in classification. The cost, or variability, to estimate the partition $\mathcal{P} = \{\mathcal{R}^{(j)}; j = 1, \ldots, d\}$ is the same as for trees. Since the variance contribution for selecting such a partition is larger than estimating the parameters in a tree as in (2.3), at least asymptotically, we should choose the more flexible tree-function involving $d$ parameters rather than just one parameter as in (4.4) and the signs $\eta_j$ in the $\pm 1$-tree. See also Figure 3.

**k-ary trees with k > 2.** $k$-ary trees ($k > 2$) are generally expected to be inferior than regression trees. $k$-ary trees ($k > 2$) also require the selection of a partition $\mathcal{P}$, as with regression trees or $\pm 1$-trees and estimating $d$ parameters as with regression trees. Having the same complexity as trees in terms of parameters and partitions, regression trees offer more flexibility. For example, a 3-ary tree with $d = 3$ nodes is a special type of a binary tree with $d = 3$ nodes. See also Figures 1-3.

## 4.3 Complementariness of tree functions and complementary rectangle functions

A tree function in (2.2) makes full use of complementariness in the following sense. For every estimated split point, say $\hat{\kappa}$, we always use both rectangles, characterized by $x_{\hat{\iota}} \le \hat{\kappa}$ *and* the complementary relation $x_{\hat{\iota}} > \hat{\kappa}$, where $\hat{\iota}$ is a selected component $\in \{1, \ldots, p\}$.

We can always select instead of a single rectangle indicator basis functions, say $\mathbf{1}_{[\cdot \in \mathcal{R}]}$, also in addition its complement $\mathbf{1}_{[\cdot \in \mathcal{R}^C]}$, where $\mathcal{R}^C = \mathbb{R}^p \setminus \mathcal{R}$. This gives rise to the complementary pair of rectangle functions,

$$g_{\text{C-}rect}(x) = \theta_1 \mathbf{1}_{[x \in \mathcal{R}]} + \theta_2 \mathbf{1}_{[x \in \mathcal{R}^C]}, \tag{4.8}$$

where $\theta_j \in \mathbb{R}$ are coefficients determined by the response variable, analogously as in (2.3). With one split point for the rectangle construction, the complementary rectangle functions coincide with stumps, i.e. trees with $d = 2$. But differences start to emerge when using rectangles which arise with more than one split point.

Complementary rectangle functions in (4.8) are somewhere between trees in (2.2) and rectangle indicator basis functions in (4.1). Depending on signal to noise level, among other things as seen also from Figure 1, one of them will sometimes be better than the other. Among the tree-type functions we have just discussed, none of them will be best overall. We did argue why $\pm 1$-trees in (4.2) and $k$-ary trees are expected to be rarely better than regression trees; but ranking $L_2$Boosting with regression trees, with complementary rectangle functions or with rectangle indicator basis functions is generally impossible. See also Figure 3.

## 4.4 Shrinking estimates

Shrinkage is another way to influence the bias-variance behavior of the base procedure. Instead of using a fitted function $\hat{g}(\cdot)$, generally as in (2.1), we can use $\nu \hat{g}(\cdot)$ $(0 < \nu \le 1)$ with shrinkage factor $\nu$. Friedman (2001) demonstrates empirically that shrinkage "never" hurts but sometimes improves considerably; Bühlmann and Yu's (2001) result about asymptotic optimality of $L_2$Boosting with smoothing splines for one-dimensional curve estimation also holds for the shrunken base procedure with any $0 < \nu \le 1$.

We found empirically, that the effect of shrinkage is rather unrelated from the effect of choosing a tree or a rectangle indicator basis function. The heuristics that rectangle indicator basis functions in (4.1) are better than trees in (2.2) in low noise (large sample size) problems is not directly affected by shrinkage.

## 4.5 Numerical examples

We focus here mainly on the case of $L_2$Boosting where the base procedure involves one or two selected predictor variables. For the former, we consider stumps, i.e. a tree in (2.2) with $d = 2$, or the following versions thereof: the rectangle indicator basis function in (4.1) involving only one split point $\kappa$ for the rectangle $\mathcal{R} = \mathbb{R} \times \mathbb{R} \cdots \times (a, b] \times \mathbb{R} \times \cdots \times \mathbb{R}$ with $(a, b] = (\infty, \kappa]$ or $(a, b] = (\kappa, \infty]$ for one of the axis in $\mathbb{R}^p$, or the $\pm 1$-stumps as in (4.2), or a 3-ary tree with $d = 3$ (i.e. only one variable will be used for splitting). All of such tree-type learners yield an estimated function of one predictor variable and hence, by linear combination of such estimates, we obtain an estimated function which is additive in the predictor variables. In section 4.5.3, we also look at tree-type learners which involve two split points and at most two predictor variables, yielding $L_2$Boosting function estimates which include additive and second-order interaction effects.

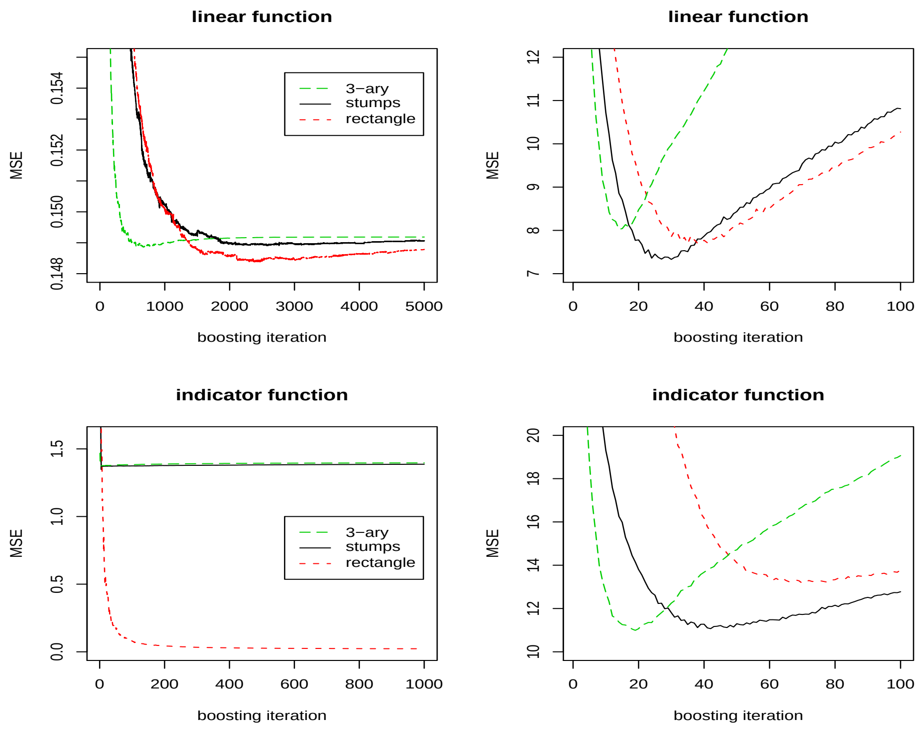

Figure 1: MSE of $L_2$Boosting with stumps (solid line), rectangle indicator basis function (short dashed line) as in (4.1) and 3-ary tree with 3 nodes (long dashed line). Top: linear function $f(x) = 5x$; Bottom: indicator function $f(x) = 10.29 \cdot \mathbf{1}_{[-0.5 \le x \le 0.5]}$. Left: no shrinkage ($\nu = 1$) and noise variance $\sigma_\varepsilon^2 = 0.01$. Right: shrinkage with factor $\nu = 0.1$ and noise variance $\sigma_\varepsilon^2 = 10$. Sample size $n = 100$.

### 4.5.1 One-dimensional functions in low and high noise settings

We consider here simulated data from the following nonparametric regression model:

$$Y_i = f(X_i) + \varepsilon_i,\ \varepsilon_i \sim \mathcal{N}(0, \sigma_\varepsilon^2),$$
$$f(x) = 5x \text{ or } f(x) = 10.29 \cdot \mathbf{1}_{[-0.5 \le x \le 0.5]}, \tag{4.9}$$

where $X_1, \dots, X_n$ are i.i.d. $\sim \mathcal{N}(0,1)$, $\varepsilon_1, \dots, \varepsilon_n$ i.i.d., independent from $\{X_s; 1 \le s \le n\}$. We always choose sample size $n = 100$. The two different functions $f(\cdot)$ are scaled such that the signal to noise ratios are equal.

Figure 1 shows the mean squared error $\mathbb{E}[(\hat{F}_n^m(X) - F(X))^2]$ (MSE) behavior of $L_2$Boosting with different base procedures, based on 10 independent realizations from model (4.9). In case of the rectangle indicator basis function, we see a substantial gain in the low noise situation. In case of the linear function, both stumps and rectangles perform about equally. We add here that shrinkage for stumps did not help in the lower left panel of Figure 1.

### 4.5.2 Additive function with many non-effective variables

We also consider simulated data from an additive regression model:

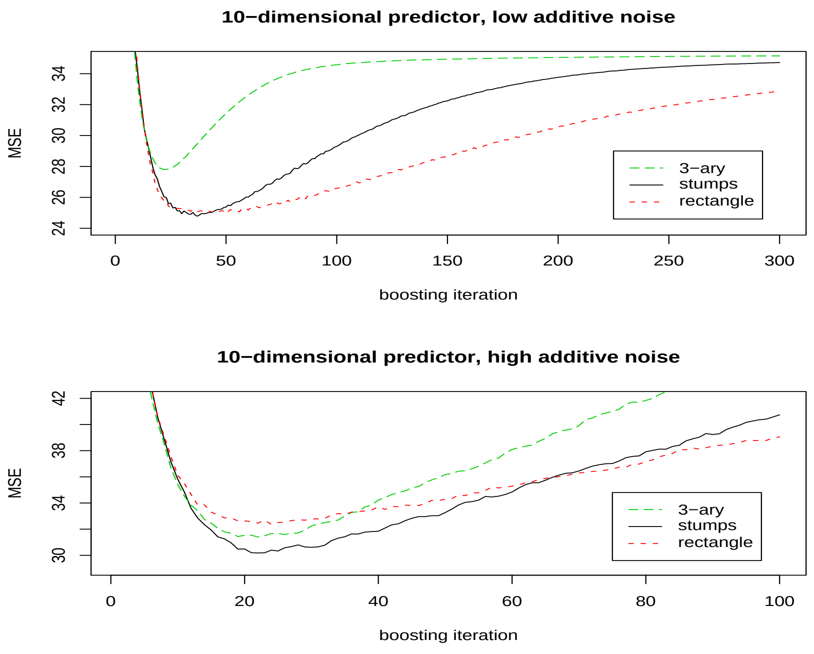

Figure 2: MSE of $L_2$Boosting with stumps (solid line), rectangle basis indicator function (short dashed line) as in (4.1) and 3-ary tree with 3 nodes (long dashed line), always with shrinkage factor $\nu = 0.1$. Indicator function $f(x)$ as in (4.10). Top: noise variance $\sigma_\varepsilon^2 = 0.01$; Bottom: noise variance $\sigma_\varepsilon^2 = 100$. Sample size $n = 100$.

$$Y_i = f(X_i) + \varepsilon_i,\ \varepsilon_i \sim \mathcal{N}(0, \sigma_\varepsilon^2),$$
$$f(x) = 10.29(\mathbf{1}_{[-0.5 \le x_1 \le 0.5]} + \mathbf{1}_{[-0.5 \le x_2 \le 0.5]})/\sqrt{2}, \tag{4.10}$$

where $X_1, \ldots, X_n$ are i.i.d. $\sim \mathcal{N}_{10}(0, I)$, and the $\varepsilon_i$'s as in (4.9). Sample size is again $n = 100$. The function $f(\cdot)$ involves only 2 of the 10 available predictor variables. The signal to noise ratio in (4.9) and (4.10) are the same.

Figure 2 shows the mean squared error $\mathbb{E}[(\hat{F}_n^m(X) - F(X))^2]$ behavior of $L_2$Boosting with different base procedures, based on 10 independent realizations from model (4.10). The advantage of rectangle indicator basis functions over stumps is lost here, even for the low additive noise case, see Figure 2. The number of effective predictor variables, namely 2, is small in comparison to $p = 10$ and the variance when making a choice in (2.4) is of course also depending on the number of predictor variables. Thus, what seems to be a low noise problem in the upper panel of Figure 2 is in fact only low additive error noise but overall a high noise problem due to the many non-effective predictor variables.

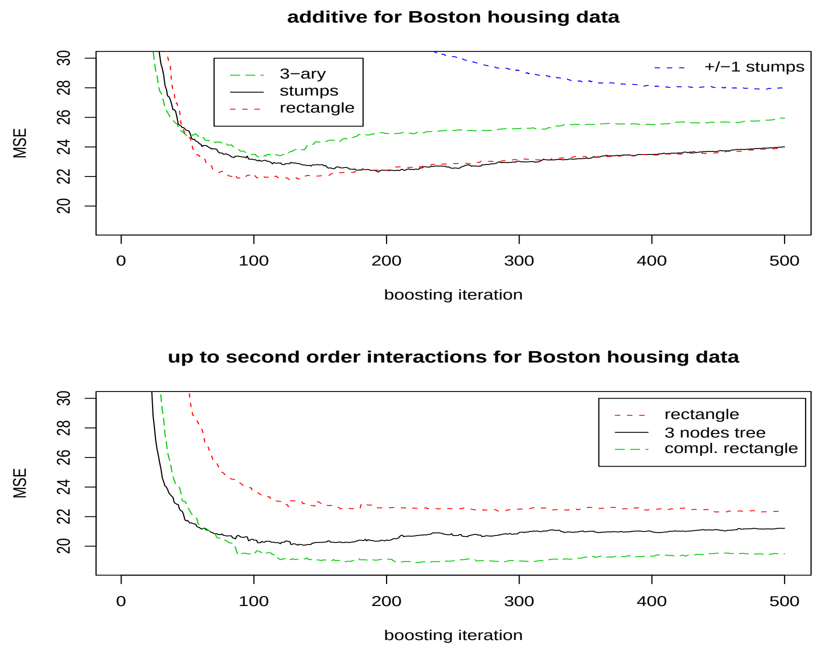

Figure 3: Estimated MSE $\mathbb{E}[(\hat{F}_n^m(X) - Y)^2]$, via 5-fold cross-validation, for Boston housing data. Top: additive $L_2$Boosting fits where every tree or rectangle involves only one predictor variable. $L_2$Boosting with stumps (solid line), rectangle indicator basis function (short dashed line) as in (4.1) and 3-ary tree with 3 nodes (long dashed line); in the upper right hand corner, also with $\pm 1$-stumps (dashed line) as in (4.2). Bottom: $L_2$Boosting fits where every tree or rectangle involves one or two predictor variable (second order interactions). $L_2$Boosting with 3 nodes regression tree (solid line), rectangle indicator basis function (short dashed line) as in (4.1) and complementary rectangle indicators (long dashed line) as in (4.8). Always using shrinkage factor $\nu = 0.1$.

### 4.5.3 Real dataset

We consider here the dataset about Boston housing prices with $p = 13$ predictor variables, available at http://lib.stat.cmu.edu/datasets/boston . Sample size is $n = 506$ and we always estimate mean squared error by 5-fold cross-validation. We force $L_2$Boosting to produce additive functions fits or estimates including additive and second-order interaction effects: this can be realized by using rectangles or tree which involve only one predictor variable or two variables in case of interaction effects. Particularly the method including second but not higher order interaction effects is popular for many practical problems.

Results are displayed in Figure 3: for comparison, the estimated MSE for a linear model is 37.1 so that $L_2$Boosting improves by more than 40%. In case of additive modelling, for this real dataset, we see a slight advantage of rectangle indicator basis functions over stumps. The top panel of Figure 3 also demonstrates that $\pm 1$-trees can be very inefficient for estimating the regression function. Generalizing, we expect that $\pm 1$-trees are inefficient for estimating the conditional probability function $\mathbb{P}[Y = 1|X = x]$ in 2-class classification. An improvement can be achieved by modelling including second order interactions. We see an advantage (5.8 % when stopping optimally) of complementary rectangle basis functions over classical regression trees with 3 terminal nodes. The rectangle indicator basis functions are not best anymore. It is interesting to note that in both, pure additive and interaction modelling, the classical regression tree learner could be slightly improved by using a more parsimonious tree-type function.

## 5 Conclusions

We present here consistency results for $L_2$Boosting with regression trees and related learners, both in regression and classification. It is the first consistency result for boosting-estimates of a nonparametric regression function $f(\cdot) = \mathbb{E}[Y|X = \cdot] : \mathbb{R}^p \to \mathbb{R}$ with arbitrary $p$. In the classification problem, our theory and assumptions can be compared to other theoretical analyses of boosting algorithms. Table 5.1 gives a comparative overview.

| reference | algorithm | $f(\cdot)$ | predictor | learner |
|---|---|---|---|---|
| current | $L_2$Boosting | arbitrary | arbitrary | $\mathbb{R}$-valued regression tree and versions |
| Jiang | AdaBoost | smooth | density on compact support | $\pm 1$-tree |
| Lugosi & Vayatis | regularized boosting | arbitrary | arbitrary | $\pm 1$-valued with finite VC-dim. |
| Mannor et al. | regularized boosting | arbitrary | arbitrary | general with condition on Rademacher complexity |

Table 5.1: Assumptions for consistency of different boosting methods in classification. In contrast to all others, our analysis for $L_2$Boosting also covers regression.

Besides our analysis, only Jiang (2000) also looks at non-regularized boosting which is computationally much more attractive, see also section 1, but we require clearly less than he does. In particular we allow for $f(\cdot) = \mathbb{P}[Y = 1|X = \cdot]$ having jumps which is necessary for Bayes error zero, for categorical predictors, and for $\mathbb{R}$-valued regression trees which are more efficient for estimating $f(\cdot)$ (but not necessarily better for classification). Even when comparing to regularized boosting, which seems simpler to analyze, our assumptions are competitive.

Finally, we discuss in section 4 the fundamental difference between a tree learner and projection onto basis functions, the latter being the common setting for matching pursuit (Mallat and Zhang, 1993) in signal processing. Depending on the signal to noise ratio, one will be better than the other. This adds flexibility to choose among different tree-type base learners.

## 6 Proofs

It is notationally more convenient to work with basis functions in (4.1), but the analysis for tree functions in (2.2) is analogous. We also consider the regression problem only in Theorem 1 and Corollary 1. From section 4.1 we know that the population version of $L_2$Boosting converges to the underlying target function $f(\cdot)$.

We consider here also a semi-population version of $L_2$Boosting: similarly as in (4.5), we define the sequence of remainder functions

$$\begin{aligned} R_n^0 f &= f, \\ R_n^m f &= R_n^{m-1} f - \frac{\langle R_n^{m-1} f, g_{\hat{\mathcal{R}}_m} \rangle}{\|g_{\hat{\mathcal{R}}_m}\|_2^2} g_{\hat{\mathcal{R}}_m}, \ m = 1, 2, \ldots \end{aligned} \tag{6.1}$$

where $\hat{\mathcal{R}}_m$ is chosen as in (2.4) which implies that $R_n^m f$ is random.

### 6.1 Asymptotic analysis as sample size increases

We show here how the effect of estimation errors can be controlled. The key ingredients are uniform bounds with respect to the rectangles $\mathcal{R}$ in (4.1).

Analogously to (6.1), define the $n \times 1$ vectors

$$\begin{aligned} \hat{R}_n^0 f &= Y = (Y_1, \ldots, Y_n)^T, \\ \hat{R}_n^m f &= \hat{R}_n^{m-1} f - \frac{\sum_{i=1}^n \hat{R}_n^{m-1}(X_i) g_{\hat{\mathcal{R}}_m}(X_i)}{\sum_{i=1}^n g_{\hat{\mathcal{R}}_m}(X_i)^2} g_{\hat{\mathcal{R}}_m}, \ m = 1, 2, \ldots \end{aligned}$$

With some abuse of notation, we denote by $f, R_n^m f, \hat{R}_n^m f$ either functions from $\mathbb{R}^p \to \mathbb{R}$ or $n \times 1$ vectors, evaluated at the observed predictors.

**Lemma 1** *Assume assumption (A). Then:*

*(i)* $\sup_{\mathcal{R}} |n^{-1} \sum_{i=1}^n g_{\mathcal{R}}(X_i)^2 - \|g_{\mathcal{R}}\|_2^2| = \xi_{n,1} = O_P(n^{-1/2})$,

*(ii)* $\sup_{\mathcal{R}} |n^{-1} \sum_{i=1}^n f(X_i) g_{\mathcal{R}}(X_i) - \langle f, g_{\mathcal{R}} \rangle \,| = \xi_{n,2} = O_P(n^{-1/2})$,

*(iii)* $\sup_{\mathcal{R}} |n^{-1} \sum_{i=1}^n \varepsilon_i g_{\mathcal{R}}(X_i)| = \xi_{n,3} = O_P(n^{-1/2})$,

Proof: The Donsker property holds for empirical processes with indicator functions indexed by hyper-rectangles $\mathcal{R} = (-\infty, a_1] \times \cdots \times (-\infty, a_p] \subseteq \mathbb{R}^p$, see van der Vaart and Wellner (1996, p.129). By taking differences of hyper-rectangles $\mathcal{R} = (-\infty, a_1] \times \cdots \times (-\infty, , a_p] \subseteq \mathbb{R}^p$ we can obtain any hyper-rectangle $\mathcal{R} \subseteq \mathbb{R}^p$. Therefore, assertions (i)-(iii) follow. $\square$

**Lemma 2** *Assume assumptions (A) and (B). Then, for $m = m_n \to \infty,\ m_n = o(\log(n))$,*

$$\|\hat{R}_n^{m_n} f\|_2^2 = \|R_n^{m_n} f\|_2^2 + \sigma_\varepsilon^2 + o_P(1).$$

Proof: Consider first $m = 1$. Then, by definition,

$$\hat{R}_n^1 f = f + \varepsilon - \frac{\langle f, g_{\hat{\mathcal{R}}_1}\rangle}{\|g_{\hat{\mathcal{R}}_1}\|_2^2} g_{\hat{\mathcal{R}}_1} + \Delta_{1,n} = R_n^1 f + \varepsilon + \Delta_{1,n},$$

$$\Delta_{1,n} = \gamma_{1,n} g_{\hat{\mathcal{R}}_1},\ \gamma_{1,n} = \left( \frac{\langle f, g_{\hat{\mathcal{R}}_1}\rangle}{\|g_{\hat{\mathcal{R}}_1}\|_2^2} - \frac{n^{-1}\sum_{i=1}^n Y_i g_{\hat{\mathcal{R}}_1}(X_i)}{n^{-1}\sum_{i=1}^n g_{\hat{\mathcal{R}}_1}(X_i)^2} \right)$$

The scalar random variable $\gamma_{1,n}$ can then be analyzed via Taylor expansion and using Lemma 1. The general bound in (6.4) below also applies to $\gamma_{1,n}$.

Now, we proceed iteratively. Denote by $\Delta_{k,n} = \gamma_{k,n} g_{\hat{\mathcal{R}}_k}$. Then,

$$\begin{aligned}
\hat{R}_n^j f &= \hat{R}_n^{j-1} f - \frac{n^{-1}\sum_{i=1}^n (\hat{R}_n^{j-1} f)_i g_{\hat{\mathcal{R}}_j}(X_i)}{n^{-1}\sum_{i=1}^n g_{\hat{\mathcal{R}}_j}(X_i)^2} g_{\hat{\mathcal{R}}_j} \\
&= \varepsilon + R_n^{j-1} f + \Delta_{1,n} + \ldots + \Delta_{j-1,n} - \frac{n^{-1}\sum_{i=1}^n (\hat{R}_n^{j-1} f)_i g_{\hat{\mathcal{R}}_j}(X_i)}{n^{-1}\sum_{i=1}^n g_{\hat{\mathcal{R}}_j}(X_i)^2} g_{\hat{\mathcal{R}}_j} \\
&= \varepsilon + R_n^j f + \Delta_{1,n} + \ldots + \Delta_{j,n}, \\
\gamma_{j,n} &= \frac{\langle R_n^{j-1} f, g_{\hat{\mathcal{R}}_j}\rangle}{\|g_{\hat{\mathcal{R}}_j}\|_2^2} - \frac{n^{-1}\sum_{i=1}^n (\hat{R}_n^{j-1} f)_i g_{\hat{\mathcal{R}}_j}(X_i)}{n^{-1}\sum_{i=1}^n g_{\hat{\mathcal{R}}_j}(X_i)^2} =: \frac{u_0}{v_0} - \frac{U_n}{V_n}. \qquad (6.2)
\end{aligned}$$

Next, we control the random variable $\gamma_{j,n}$. Since

$$\hat{R}_n^{j-1} f = \varepsilon + R_n^{j-1} f + \Delta_{1,n} + \ldots + \Delta_{j-1,n},$$

we expand $U_n$ in (6.2) into $j+1$ summands, and

$$\begin{aligned}
U_n - u_0 &= n^{-1}\sum_{i=1}^n \varepsilon_i g_{\hat{\mathcal{R}}_j}(X_i) + \left( n^{-1}\sum_{i=1}^n (R_n^{j-1} f)_i g_{\hat{\mathcal{R}}_j}(X_i) - \langle R_n^{j-1} f, g_{\hat{\mathcal{R}}_j}\rangle \right) \\
&+ \sum_{k=1}^{j-1} n^{-1}\sum_{i=1}^n \Delta_{k,n}(X_i) g_{\hat{\mathcal{R}}_j}(X_i) =: W_{n,1} + W_{n,2} + W_{n,3}. \qquad (6.3)
\end{aligned}$$

Denote in the sequel by $\xi_n = \max_{1\le i\le 3} \xi_{n,i}$ the maximum of the quantities in Lemma 1. Expanding $R_n^{j-1} f$ into $j$ summands as in (6.1), we get

$$|W_{n,1}| \le C\xi_n,$$
$$|W_{n,2}| \le C\xi_n j.$$

Bounding the third term in (6.3) can be done as follows:

$$|W_{n,3}| \le \sum_{k=1}^{j-1} \|\Delta_{k,n}\|_\infty n^{-1}\sum_{i=1}^n g_{\hat{\mathcal{R}}_j}(X_i) = \sum_{k=1}^{j-1} |\gamma_{k,n}| n^{-1}\sum_{i=1}^n g_{\hat{\mathcal{R}}_j}(X_i)^2,$$

because $g_{\hat{\mathcal{R}}_j}$ is an indicator function. Therefore, using Lemma 1 again, we get the bound for (6.3),

$$\begin{aligned} |U_n - u_0| &\leq \xi_n(j+1) + \sum_{k=1}^{j-1} |\gamma_{k,n}|(\|g_{\hat{\mathcal{R}}_j}\|_2^2 + \xi_n) \\ &\leq \xi_n(j+1) + \sum_{k=1}^{j-1} |\gamma_{k,n}|\|g_{\hat{\mathcal{R}}_j}\|_2^2 O_P(1), \end{aligned}$$

since $\|g_{\hat{\mathcal{R}}_j}\|_2^{-2} = o(n^{1/3})$ due to assumption (B). Using a Taylor expansion for $\gamma_{j,n}$ in (6.2), we get

$$\begin{aligned} |\gamma_{j,n}| &= \left|\frac{u_0 - U_n}{v_0} + (\frac{1}{v_0} - \frac{1}{V_n})U_n\right| \\ &\leq \|g_{\hat{\mathcal{R}}_j}\|_2^{-2}\xi_n(j+1) + \sum_{k=1}^{j-1} |\gamma_{k,n}|O_P(1) + \xi_n O_P(\|g_{\hat{\mathcal{R}}_j}\|_2^{-4})|U_n| \\ &\leq \|g_{\hat{\mathcal{R}}_j}\|_2^{-2}\xi_n(j+1) + \sum_{k=1}^{j-1} |\gamma_{k,n}|O_P(1) + \xi_n O_P(\|g_{\hat{\mathcal{R}}_j}\|_2^{-3}), \end{aligned} \tag{6.4}$$

because

$$|U_n| \leq (n^{-1}\sum_{i=1}^{n}(\hat{R}_n^{j-1}f)_i^2)^{1/2}(\sum_{i=1}^{n} g_{\hat{\mathcal{R}}_j}(X_i)^2)^{1/2} \leq O_P(1)\|g_{\hat{\mathcal{R}}_j}\|_2,$$

since $n^{-1}\sum_{i=1}^{n}(\hat{R}_n^{j-1}f)_i^2 \leq n^{-1}\sum_{i=1}^{n} Y_i^2 = O_P(1)$ and $n^{-1}\sum_{i=1}^{n} g_{\hat{\mathcal{R}}_j}(X_i)^2 \leq \|g_{\hat{\mathcal{R}}_j}\|_2^2 + \xi_n \leq O_P(\|g_{\hat{\mathcal{R}}_j}\|_2^2)$ since $\|g_{\hat{\mathcal{R}}_j}\|_2^{-2} = o(n^{1/3})$ due to assumption(B). A crude upper bound for (6.4) is then

$$|\gamma_{j,n}| \leq O_P(\xi_n) max_{\hat{\mathcal{R}}}\|g_{\hat{\mathcal{R}}}\|_2^{-3}(j+2) + \sum_{k=1}^{j-1} |\gamma_{k,n}|O_P(1).$$

(The $O_P(1)$ term does not depend on $j$). Recursive use of this bound then yields

$$|\gamma_{j,n}| \leq O_P(\xi_n) max_{\hat{\mathcal{R}}}\|g_{\hat{\mathcal{R}}}\|_2^{-3}2^j.$$

Hence for $\Delta_{j,n} = \gamma_{j,n} g_{\hat{\mathcal{R}}_j}$,

$$\|\sum_{j=1}^{m} \Delta_{j,n}\|_2 \leq \sum_{j=1}^{m} \|\Delta_{j,n}\|_2 \leq \sum_{j=1}^{m} |\gamma_{j,n}| \leq O_P(\xi_n) max_{\hat{\mathcal{R}}}\|g_{\hat{\mathcal{R}}}\|_2^{-3} \sum_{j=1}^{m} 2^j = o_P(1), \tag{6.5}$$

if $m = m_n = o(\log(n))$. Therefore, using formula (6.2) and (6.5) we arrive at

$$\begin{aligned} \|\hat{R}_n^m f\|_2^2 &\leq \sigma_\varepsilon^2 + \|R_n^m f\|_2^2 + \|\sum_{j=1}^{m} \Delta_{j,n}\|_2^2 + 2\|R_n^m f + \varepsilon\|_2 \, \|\sum_{j=1}^{m} \Delta_{j,n}\|_2 \\ &= \sigma_\varepsilon^2 + \|R_n^m f\|_2^2 + o_P(1). \end{aligned}$$

□

**Lemma 3** *Assume assumptions (A) and (B). Then, for any $\varepsilon > 0$, there exists $m_0 = m_0(\varepsilon)$ and $n_0 = n_0(\varepsilon)$ such that*

$$\mathbb{P}[\|R_n^{m_0} f\|_2 \leq \varepsilon] \geq 1 - \varepsilon \text{ for all } n \geq n_0(\varepsilon).$$

Proof: Lemma 1 implies that in the first boosting step, the empirical scaled inner product in (2.4) is close to the population quantity in (4.6) (not necessarily the maximizers). This holds for finitely many further iterations. Moreover, the first population maximizers $\mathcal{R}_1, \ldots, \mathcal{R}_{m_0}$ in (4.6) all satisfy $\|g_{\mathcal{R}_j}\|_2^{-2} = o(n^{1/3})$ $(j = 1, \ldots, m_0)$ as $n$ gets large ($m_0$ finite) so that they belong to the class of trees (rectangles) satisfying assumption (B). Hence, formula (11) in Mallat and Zhang (1993) holds in probability for the empirically chosen $\hat{\mathcal{R}}_1, \ldots, \hat{\mathcal{R}}_{m_0}$ ($m_0$ finite) and their result then implies the existence of an $m_0(\varepsilon)$ such that the assertion of the Lemma holds. $\square$

*Proof of Theorem 1:*
Let $X, Y$ be a new test observation, independent from the (training) data $\mathcal{D} = (X_i, Y_i)_{i=1}^n$. Lemma 2 and the monotone decay of $\|R_n^m f\|_2$ as $m \to \infty$ then imply, for any $\varepsilon > 0$,

$$\begin{aligned} \mathbb{E}_X|\hat{F}_n^{m_n}(X) - f(X)|^2 &= \mathbb{E}|\hat{F}_n^{m_n}(X) - Y|^2 - \sigma_\varepsilon^2 = \|\hat{R}_n^{m_n} f\|_2^2 - \sigma_\varepsilon^2 \\ \leq \ & \|R_n^{m_n} f\|_2^2 + o_P(1) \leq \|R_n^{m_0} f\|_2^2 + o_P(1) \leq \varepsilon O_P(1) + o_P(1), \end{aligned} \tag{6.6}$$

where the last inequality is due to Lemma 3. $\square$

*Proof of Corollary 1:*
The population version of $L_2$Boosting is converging to the best approximating function $f^*$ in the space spanned by regression trees with $d$ terminal nodes, see Mallat and Zhang (1993). Clearly, this space coincides with $T_d$ described in the Corollary. Finally, the estimation errors can be controlled analogously to the proof of Theorem 1. $\square$

## References

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

[2] Breiman, L. (1999). Prediction games & arcing algorithms. Neural Computation **11**, 1493-1517.

[3] Breiman, L. (2000). Some infinity theory for predictor ensembles. Tech. Report 579, Dept. of Statist., Univ. of Calif., Berkeley.

[4] Breiman, L., Friedman, J.H., Olshen, R.A. & Stone, C.J. (1984). *Classification and Regression Trees.* Wadsworth, Belmont (CA).

[5] Bühlmann, P. and Yu, B. (2001). Boosting with the $L_2$-loss: regression and classification. Preprint, ETH Zürich.

[6] Bühlmann, P. and Yu, B. (2002). Analyzing bagging. To appear in Ann. Statist. **30**.

[7] Devroye, L., Györfi, L., and Lugosi, G. (1996). *A probabilistic theory of pattern recognition.* Springer, New York.

[8] 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.

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

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

[11] Jiang, W. (2000). Process consistency for AdaBoost. Preprint, available at http://neyman.stats.northwestern.edu/jiang/boosting.html

[12] Koltchinskii, V. and Panchenko, D. Empirical margin distributions and bounding the generalization error of combined classifiers. Ann. Statist. **30**, 1–50.

[13] Lugosi, G. and Vayatis, N. (2001). On the Bayes-risk consistency of boosting methods. Preprint, available at http://www.econ.upf.es/~lugosi/pre.html

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

[15] Mannor, S., Meir, R. and Zhang, T. (2002). The consistency of greedy algorithms for classification. To appear in COLT (fifteenth annual conference on computational learning theory).

[16] Schapire, R. E. (2002). The boosting approach to machine learning: an overview. In MSRI Workshop on Nonlinear Estimation and Classification (D. D. Denison, M. H. Hansen, C. C. Holmes, B. Mallick and B. Yu, Eds.). Springer, New York.

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

[18] van der Vaart, A.W. and Wellner, J.A. (1996). *Weak Convergence and Empirical Processes: With Applications to Statistics.* Springer, New York.