Additive Model Boosting: New Insights and Path(ologie)s

Rickmer Schulte[1,2], David Rügamer[1,2]
[1]Department of Statistics, LMU Munich, Munich, Germany
[2]Munich Center for Machine Learning, Munich, Germany
david.ruegamer@lmu.de

Abstract

Additive models (AMs) have sparked a lot of interest in machine learning recently, allowing the incorporation of interpretable structures into a wide range of model classes. Many commonly used approaches to fit a wide variety of potentially complex additive models build on the idea of boosting additive models. While boosted additive models (BAMs) work well in practice, certain theoretical aspects are still poorly understood, including general convergence behavior and what optimization problem is being solved when accounting for the implicit regularizing nature of boosting. In this work, we study the solution paths of BAMs and establish connections with other approaches for certain classes of problems. Along these lines, we derive novel convergence results for BAMs, which yield crucial insights into the inner workings of the method. While our results generally provide reassuring theoretical evidence for the practical use of BAMs, they also uncover some "pathologies" of boosting for certain additive model classes concerning their convergence behavior that require caution in practice. We empirically validate our theoretical findings through several numerical experiments.

1 INTRODUCTION

Additive models (AMs) are widely used in the statistics and machine learning community. Examples include interpretable boosting methods such as Lou et al. (2012); Nori et al. (2019) or the so-called neural additive models (NAMs; Agarwal et al., 2021; Radenovic et al., 2022). In order to maintain the interpretability of such models in the presence of a high-dimensional feature space, sparsity approaches like the Lasso (Tibshirani, 1996) have become indispensable. While methods such as the Lasso are well-understood from a theoretical point of view, their applicability is often restricted to a specific class of problems. An alternative approach that is applicable to a much larger class of AMs and can induce sparse structures even in the presence of complex feature effects is to boost additive models.

Boosting Additive Models Gradient boosting methods that use additive model components as base learners have been studied under different names and in different forms in recent years. Under the name of model-based boosting (Hothorn et al., 2010), this idea was proposed as an alternative optimization and selection routine for AMs. The resulting boosted additive models that we abbreviate with *BAMs*, allow the fitting of a large plethora of different model classes, including non-linear effects through regression spline representations (Schmid and Hothorn, 2008), time-varying or functional response models (Brockhaus et al., 2017), boosted densities (Maier et al., 2021) or shapes (Stöcker et al., 2023), and have been used for high-dimensional settings both in the context of classical generalized additive models (GAMs; Hothorn and Bühlmann, 2006), quantile regression (Fenske et al., 2011) and distributional regression (Mayr et al., 2012). Yet, most papers building on the seminal work of Bühlmann and Yu (2003) and Friedman (2001) do not provide theoretical guarantees but instead rely on the corresponding findings in simpler models.

Related Literature Previous work that investigated BAMs from a theoretical perspective make use of the connection to functional gradient descent (Friedman, 2001), which allows providing numerical convergence results of various boosting methods (Collins et al., 2002; Mason et al., 1999; Meir and Rätsch, 2003; Rätsch et al., 2001; Zhang and Yu, 2005). Most of these results, however, require rather strong assumptions about the objective function or consider modified versions of boosting by imposing certain restrictions on the step size and considered function space. This limits the

Preliminary work. Do not distribute.

applicability of results and does not generalize to more complex cases of BAMs. More recent investigations (Karimi et al., 2016; Freund et al., 2017; Locatello et al., 2018) exploit new theoretical insights derived from greedy coordinate descent routines, but are again restricted to specific BAM classes. Accelerated and randomized versions were investigated in Lu et al. (2020); Lu and Mazumder (2020). Statistical properties of BAMs that incorporate the iterative fitting in boosting have been investigated in Bühlmann and Hothorn (2007); Bühlmann and Yu (2003); Schmid and Hothorn (2008). However, as we show in our work, these results do not cover important theoretical aspects required for practical applications of BAMs.

Problem statement Although empirical results frequently show that more complex BAMs perform well in practice, there is a significant gap between the existing theory for simpler models and the application of these in more complex models. This gap raises the risk of researchers and practitioners using these models without fully understanding what is being optimized and what should be kept in mind when using them in practice.

Our Contributions In this work, we study BAMs from a theoretical perspective and find several connections to other prominent optimization methods. However, our investigations also uncover pathologies inherent to boosting certain additive model classes that have important implications for practical usage. In summary, our findings include:

1. Derivation of exact parameter paths for various L_2-Boosting variants.
2. Explicit characterization of the implicit regularization of several BAM classes, formally showing their difference to explicit regularized counterparts.
3. Convergence guarantees including a linear convergence rate for (greedy) block-wise boosting.
4. Specific convergence results for various spline and exponential family boosting models.

We further provide empirical experiments in Section 4 that confirm our theoretical findings.

2 Background

2.1 Notation

In this paper, we consider n observations $(y_i, x_i), i \in [n] := \{1, \dots, n\}$, that are the realizations of random variables from some joint distribution $\mathbb{P}_{yx}$. We denote the stacked outcome as $y = (y_1, \dots, y_n)^\top \in \mathbb{R}^n$ with $y_i \in \mathcal{Y} \subseteq \mathbb{R}$, and the feature matrix $X = (x_1^\top, \dots, x_n^\top)^\top \in \mathbb{R}^{n \times p}$ with rows $x_i \in \mathcal{X} \subseteq \mathbb{R}^p$. The jth feature in x_i potentially spans multiple columns p_j (e.g., for one-hot encodings) and is denoted with $x_{ij} \in \mathbb{R}^{p_j}$, $1 \leq p_j \leq p$. Stacking x_{ij} for all n observations yields $X_j \in \mathbb{R}^{n \times p_j}$. We will use the notation $\| \cdot \|$ to denote the L_2 norm, if not stated otherwise. For a matrix Q, $\lambda_{max}(Q)$ and $\lambda_{pmin}(Q)$ denote the largest and smallest-non zero eigenvalues of Q. Our goal is to learn or estimate a parametric model $f(x; \beta)$ that takes features (or predictors) x and given parameters (or weights) β produces a prediction. A subset of this parameter is denoted by $\beta_j \in \mathbb{R}^{p_j}$. We measure the goodness-of-fit of f using the loss function $\ell(\beta) := \ell(y, f(x, \beta)) : \mathbb{R}^p \to \mathbb{R}$ written as a function of the parameters β. When learning f, we will update the model iteratively. In this context, we denote the step size or learning rate with $\nu \in (0, 1]$, $\beta^{[k]}$ the value of β in the kth iteration $k \in \mathbb{N}_0$, and $f^{[k]} = f(x; \beta^{[k]})$. If not stated otherwise, we consider the problem

$$\arg\min_{\beta \in \mathbb{R}^p} \ \ell(\beta). \tag{1}$$

As we investigate boosting of AMs, we choose f to be an additive model as discussed in the following.

2.2 Additive Models

Given a pre-specified loss function, we make the optimization problem in (1) explicit by defining the model class to be a (generalized) additive model of the form

$$\mathbb{E}(Y|x) = h(f(x; \beta)) = h\left(\textstyle\sum_{j=1}^{J} f_j(x; \beta_j)\right), \tag{2}$$

with Y the random variable related to the observations y_i and f the additive predictor that is additive in the single predictor functions f_j with parameter $\beta_j \in \mathbb{R}^{p_j}$. Usually each component f_j encompasses only a subset of features $x_{\cdot j} \in \mathbb{R}^{p_j}$. For GAMs and BAMs, these functions are typically interpretable by nature (linear effects, univariate splines, low-dimensional interaction terms, etc.). h is a monotonic activation (or inverse-link) function, mapping the learned function values to the domain of Y. Examples for h are the sigmoid function for logistic additive models or $h(\cdot) = \exp(\cdot)$ for a Poisson (count) regression model. To ensure interpretability of f, we require the f_j functions to be elements of a vector space $\mathcal{F}$ with closure under addition and scalar multiplication. In other words, if $f_j, g \in \mathcal{F}$ and $c \in \mathbb{R}$, then $f_j + cg \in \mathcal{F}$.

Linearity More specifically, we consider the space of functions f_j that are *linear* in their parameters β_j. This includes linear feature effects, dummy-encoded binary or categorical variable effects, regression splines such as P-splines (Eilers and Marx, 1996), Kriging (Oliver and Webster, 1990), Markov random fields (Rue and Held, 2005), tree-stumps or built trees with leaf weights β_j, and transfer-learning neural basis functions as in

Agarwal et al. (2021) where β_j are the last layer's weights. For the ease of presentation, we will use

$$f_j(x; \beta_j) = x_{\cdot j}^\top \beta_j \tag{3}$$

and assume that the aforementioned basis transformations are already encoded in $x_{\cdot j}$.

2.3 Boosting Additive Models

In this work, we study a commonly used variant of the gradient boosting algorithm proposed in Friedman (2001). In Algorithm 1, we present this extension of gradient boosting (Bühlmann and Hothorn, 2007), adapted for the use case of BAMs with so-called base learners f_j as given in Eq. (3). While the original algorithm was presented as boosting in function space, the linearity of the base learners f_j in the parameters β_j also allows the interpretation of boosting in parameter space.

Algorithm 1 Gradient boosting of additive models

1: Initialize $f^{[0]}$. Set $k = 0$.
2: Given loss function $\ell(\cdot)$, compute the functional derivative evaluated at the current estimate $f^{[k]}$, i.e.,

$$\widetilde{y}_i^{[k]} := -\frac{\partial}{\partial f} \ell(y_i, f)\Big|_{f=f^{[k]}(x_i)} \quad i = 1, \ldots, n \tag{4}$$

Set $k = k + 1$.
3: a) For $j = 1, \ldots, J$, estimate:

$$\hat{\beta}_j = \underset{\beta_j \in \mathbb{R}^{p_j}}{\arg\min}\, n^{-1} \sum_{i=1}^{n} (\widetilde{y}_i^{[k]} - x_{ij}^\top \beta_j)^2 + \lambda_j \beta_j^\top P_j \beta_j$$

b) Select $\hat{\jmath} = \arg\min_{j \in [J]} n^{-1} \sum_{i=1}^{n} (\widetilde{y}_i^{[k]} - x_{ij}^\top \hat{\beta}_j)^2$
4: Update $f^{[k]}(x) = f^{[k-1]}(x) + \nu \cdot f_{\hat{\jmath}}(x; \hat{\beta}_{\hat{\jmath}})$ with $\nu \in (0, 1]$.
5: Repeat steps 2 – 4 until convergence or until a prespecified stopping criterion is met.

To use the algorithm as a fitting procedure for models described in Section 2.2, one needs to bring the response on the level of the additive predictor f. This can be done by defining the loss as $\ell(\beta) := \ell(h^{-1}(y), f(x; \beta))$ for GAMs. More generally, for boosting distributional regression models (Mayr et al., 2012), ℓ represents the negative log-likelihood of a parametric distribution and the distribution parameters of interest are each modeled via different additive predictors. Apart from different base learners and loss functions that can be used in Algorithm 1, fitting can be done *jointly* or in a *greedy block-wise* fashion, thereby resembling a wide variety of gradient boosting variants.

Note, that the original gradient boosting algorithm of Friedman (2001) incorporates an additional line search between step 3 and 4 of Algorithm 1. However, this is usually omitted in practice due to its computational costs while only yielding negligible differences in the estimated models, especially for small step sizes.

2.4 Joint or Block-wise Selection and Updates

The greedy base learner selection of Algorithm 1 chooses the best-performing additive component f_j at each step. As a single component f_j may comprise multiple columns of the design matrix X, e.g. for splines or categorical variables, we further introduce the notion of block-wise updates with blocks $b \in \mathcal{B} \subseteq [J]$. Without loss of generality, we assume that each block b corresponds to one set of parameters $\beta_j, j \in [J]$ with corresponding features matrices $X_b = X_j$.

Model Types In step 3a) each base learner is fitted against the current negative gradient $\tilde{y}$. Fitting may vary for each base learner. For penalized base learners, the minimization includes a penalty parameter $\lambda_j > 0$ and penalty matrix $P_j \in \mathbb{R}^{p_j \times p_j}$. For unpenalized base learners, we set $\lambda_j = 0$. Step b) then chooses the best-fitting base learner (in terms of the L_2 loss). In combination with early stopping, this can enable variable selection as some components f_j might not be selected. In contrast, *joint* updates fit all components simultaneously to the negative gradient such that all components are selected and updated in each step. This can be seen as a special case of the block-wise procedure with a single block, i.e., $J = 1$. We theoretically investigate both model types and BAMs with several different base learners and loss functions in Section 3.

3 THEORETICAL PROPERTIES OF BOOSTED ADDITIVE MODELS

As with other iterative methods such as the Lasso, the hope is that the boosting algorithm produces an interesting set of submodels among which we can choose the final model. However, whether the obtained paths of parameters $\beta^{[k]}$ are sensible and useful is largely determined by the convergence behavior of the method, which in parts is still poorly understood. Therefore, we investigate whether and under what conditions these paths converge and what solution they are converging to. This also requires studying the implicit regularization of the method and discussing to what extent the method can be related to explicitly regularized optimization problems.

3.1 Joint Updates

3.1.1 Warm Up: Linear Model Boosting

The simplest case of Algorithm 1 is boosting with linear learners, i.e., $f = X\beta$, and L_2 loss. Studying this algorithm is instructive for a better understanding of more complex versions. Boosting with a linear model learner aims to iteratively find the parameters $\beta \in \mathbb{R}^p$ that minimize the L_2 loss. In combination with

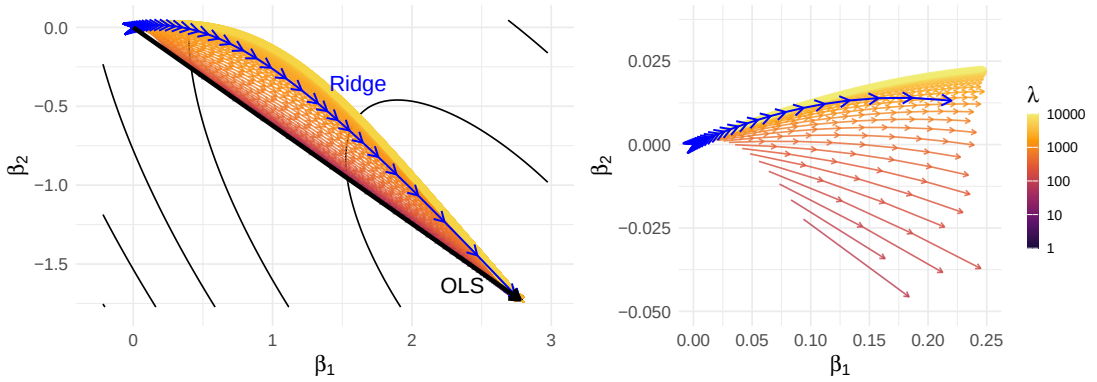

Figure 1: *Left: Paths of boosted linear models with ridge penalty (ridge boosting) for different penalty parameters λ (colored lines according to the legend) together with the ridge regression path (blue). Path of linear model boosting is the limiting case of ridge boosting with $\lambda = 0$ (black). Block contour lines represent the loss surface. Right: Same plot but zoomed in.*

joint updates, this is a particular variant of the well-known L_2-Boosting (Bühlmann and Yu, 2003). As the negative functional gradient at the current function estimate $\hat{f}^{[k]} = X\beta^{[k]}$ in (4) corresponds to the model residuals, L_2-Boosting corresponds to iterative least-squares fitting of the residuals (Bühlmann and Yu, 2003). In case X is of full column rank, a simple derivation shows that the parameters in each boosting iteration k can be written as

$$\beta^{[k]} = \left(\sum_{m=0}^{k-1} \nu(1-\nu)^m \right) \cdot \beta^{OLS} := \delta^{[k]} \cdot \beta^{OLS} \quad (5)$$

where $\nu \in (0, 1]$ denotes the step size and $\beta^{OLS} := (X^\top X)^{-1} X^\top y$ the OLS solution (cf. Appendix B.1). The shrinkage is determined by the factor $\delta^{[k]}$, which only depends on ν and k and is strictly decreasing in both. As the factor can be simplified to $\delta^{[k]} = 1 - (1-\nu)^k$, we get convergence to the respective OLS estimator in the limit ($\beta^{[k]} \to \beta^{OLS}$ as $k \to \infty$). This result is to be expected given that linear model boosting with joint updates can be seen to resemble Newton steps (with step size ν) for which convergence on quadratic problems is known (Boyd and Vandenberghe, 2004). The convergence of linear model boosting is depicted in Figure 1. The same figure also depicts the regularization path of ridge regression and the convergence paths of ridge boosting, a regularized version of linear model boosting. The explicit regularization of ridge regression and boosting's implicit regularization are discussed next.

3.1.2 Boosting with Quadratic Penalties

A common alternative to linear models in BAMs are regression splines, which allow for non-linear modeling of features. As an example for regression splines, we use P-splines (Eilers and Marx, 1996) which utilize B-splines (de Boor, 1978) as basis expansion of the predictor variables and penalize (higher-order) differences between adjacent weights β. For details on P-splines, we refer to Appendix C. Generally, splines can be formulated as an L_2 loss optimization problem with quadratic penalization term:

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2} \|y - X\beta\|^2 + \frac{\lambda}{2} \beta^\top P \beta, \quad (6)$$

where $X \in \mathbb{R}^{n \times p}$ is a matrix containing the evaluated basis functions, $\lambda > 0$ defines the regularization parameter, and $P \in \mathbb{R}^{p \times p}$ is a symmetric penalty matrix such as a second order difference matrix (Green and Silverman, 1994; Wahba, 1990). Boosting splines amounts to replacing y with the current negative gradient in (6) as shown in Algorithm 1. As regression splines use (penalized) linear effects of multiple basis functions to represent non-linear functions of single features, joint updates (of all basis functions together) naturally arise in their application. Their parameter paths and shrinkage can again be characterized exactly:

Proposition 1. *The estimates of L_2-Boosting with quadratic penalty and joint updates in iteration k are given by*

$$\beta^{[k]} = \left[\textstyle\sum_{m=0}^{k-1} \nu(I - \nu(X^\top X + \lambda P)^{-1} X^\top X)^m \right] \beta^{PLS}, \quad (7)$$

with step size $\nu \in (0, 1]$, $\lambda > 0$, P a symmetric penalty matrix, and the penalized least squares solution $\beta^{PLS} := (X^\top X + \lambda P)^{-1} X^\top y$. If X has full column rank,

$$\beta^{[k]} \overset{k \to \infty}{\longrightarrow} (X^\top X)^{-1} X^\top y = \beta^{OLS},$$

otherwise parameters converge to the min-norm solution $\beta^{[k]} \overset{k \to \infty}{\longrightarrow} X^+ y$, with X^+ being the pseudoinverse of X.

Remark 1. *The above result also holds for the special case of ridge boosting with $P = I$. Its parameter paths and convergence to the OLS are shown in Figure 1.*

The proof of Proposition 1 builds on the fact that the shrinkage in (7) can be recognized as a Neumann series. Interestingly, despite the explicit regularization, this series converges such that the shrinkage vanishes and

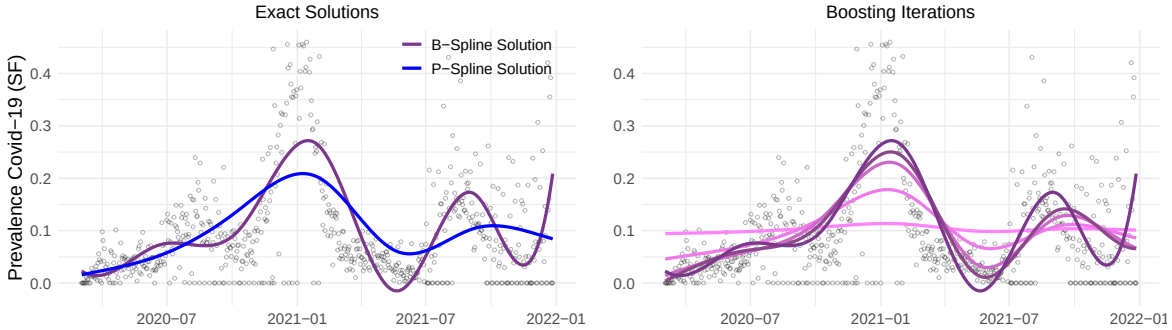

Figure 2: *Estimated logarithmic Covid-19 prevalence in San Francisco (SF) via BAMs. Left: Exact B-spline (purple) and P-spline solution (blue). Right: P-spline boosting iterates converging to the unpenalized (B-spline) solution (darkest color).*

the unpenalized fit is obtained in the limit. Figure 2 demonstrates the latter using the example of boosting P-splines to model Covid-19 prevalence in San Francisco. Details are discussed in Section 4.2.

The previous results show that L_2-Boosting (with and without penalty) induces an implicit shrinkage on the parameters. The next theorem shows that this implicit shrinkage can be characterized exactly. It demonstrates that the boosted parameters also correspond to the unique solution of an explicitly regularized problem.

Theorem 1. *Given full column rank matrix X, L_2-Boosting with quadratic penalty and joint updates* (7) *uniquely solves at each iteration $k \in \mathbb{N}$ the explicitly regularized problem*

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2}\|y - X\beta\|^2 + \frac{1}{2}\beta^\top \Gamma_k \beta, \tag{8}$$

with $\Gamma_k := (X^\top X)\ S_\lambda^{-1}\ [(I - \nu S_\lambda)^{-k} - I]^{-1}\ S_\lambda$ as penalty matrix and $S_\lambda := (X^\top X + \lambda P)^{-1} X^\top X$.

Remark 2. *The problem in* (8) *changes in each step k and generally does not correspond to the problem in* (6)*. Thus, Theorem 1 disproves a widespread interpretation of regularized spline boosting that the algorithm implicitly minimizes* (6) *for a specific $\lambda > 0$ in each iteration and thereby implicitly finds the model with optimal λ.*

Theorem 1 also allows us to further investigate the implicit shrinkage of linear model and ridge boosting by determining under what conditions their paths correspond to the solution of ridge regression.

Corollary 1. *The parameter paths of L_2-Boosting for linear models with joint updates correspond to the solutions of ridge regression with penalty parameter*

$$\tilde{\lambda}(k) := \sigma_X^2 \frac{(1-\nu)^k}{1-(1-\nu)^k} \tag{9}$$

if and only if $X^\top X = \sigma_X^2 I$ for some $\sigma_X^2 > 0$.

Remark 3. *The same holds for ridge boosting with a slightly different penalty parameter (cf. Appendix B.4). Importantly, these equivalences only hold in the case of isotropic features. The fact that paths of boosting and ridge regression differ in general is shown in Figure 1. Details about Figure 1 can be found in Appendix D.5.*

3.2 Block-wise Boosting

We now turn to the block-wise boosting variant and a second main result. As discussed in Section 2.4, the block-wise setting generalizes joint and component-wise updates, including both as special cases. Before deriving our main convergence results, we first relate this greedy update variant to a specific optimization procedure:

Proposition 2. *Boosting additive models with greedy block-wise updates and L_2 loss corresponds to optimizing AMs with greedy block coordinate descent (GBCD) and the Gauss-Southwell-Quadratic (GSQ) update scheme (21,25). In the component-wise case, it matches greedy coordinate descent (GCD) with Gauss-Southwell-Lipschitz (GSL) update scheme (23,24).*

Remark 4. *In case of the penalized L_2 loss* (6)*, block-wise boosting additive models can be seen as G(B)CD with GSQ (GSL) on the unpenalized L_2 loss. We give further details in Section 3.2.1. The connection in case of other loss functions is discussed in Section 3.2.3.*

While GBCD has a long-standing history in the optimization literature, the two mentioned update schemes of GSQ and GSL were only introduced recently and shown to be particularly efficient (Nutini et al., 2015, 2022). Thus, to the best of our knowledge, the described link between these update schemes and BAMs has not been established so far. Under certain conditions on the loss function, we will use the above equivalence to derive novel convergence guarantees for BAMs with block-wise updates, including a linear convergence rate. For this, we assume the problem to be μ-PL and L-smooth in the parameters $\beta \in \mathbb{R}^p$ (cf. Appendix A.1 for a definition). Requiring only the weaker condition of μ-PL instead of strong convexity allows us to study convergence also in cases where the optimal solution is not unique, e.g. boosting high-dimensional linear models with $n < p$.

Theorem 2. *If $\ell(\beta)$ is μ-PL and L-smooth with respect to the parameters β, block-wise boosting with step size ν converges with rate γ:*

$$\ell(\beta^{[k]}) - \ell^* \leq \underbrace{\left(1 - \nu(2-\nu)\frac{\mu}{L_{\mathcal{B}}|\mathcal{B}|}\right)}_{=:\gamma}^{k} \left(\ell(\beta^{[0]}) - \ell^*\right), \tag{10}$$

for $\nu \in (0, 1]$ s.t. $\nabla^2_{bb}\ell(\beta) \preceq \frac{1}{\nu} X_b^\top X_b \; \forall \beta \in \mathbb{R}^p$, $\forall b \in \mathcal{B}$. Above ℓ^* denotes the optimal loss, $|\mathcal{B}|$ the number of blocks, and $L_{\mathcal{B}}$ the largest Lipschitz constant of all blocks with $L_{\mathcal{B}} := \max_{b\in\mathcal{B}} L_b \leq L$.

Remark 5. *A distinct property of boosting additive models, that originates from the quadratic approximation of the negative functional gradient in the boosting algorithm, is the scaling of each gradient update by a particular (block) matrix. Independent of ℓ, this scaling matrix is the inverse of $\frac{1}{\nu} X_b^\top X_b$ and $\frac{1}{\nu}(X_b^\top X_b + \lambda P_b)$ for unpenalized and penalized base learner, respectively. Hence, for convergence guarantees as in Theorem 2, one requires $\frac{1}{\nu} X_b^\top X_b$ to provide a valid upper bound to the respective block of the Hessian for all $b \in \mathcal{B}$. For some BAM classes, this requires choosing ν sufficiently small. For others, this upper bound condition is naturally fulfilled. As quadratic problems are μ-PL and L-smooth (cf. Appendix A.1), we get the following corollary.*

Corollary 2. *If $\ell(\beta)$ is a quadratic problem with positive semi-definite Hessian Q, block-wise boosting converges for $\nu \in (0, 1]$:*

$$\ell(\beta^{[k]}) - \ell^* \leq \left(1 - \frac{\nu(2-\nu)}{|\mathcal{B}|}\frac{\lambda_{pmin}(Q)}{\lambda_{max}(Q)}\right)^k \left(\ell(\beta^{[0]}) - \ell^*\right). \tag{11}$$

We obtain several insights from Theorem 2 and Corollary 2 related to boosting's convergence speed. First, γ is monotonically decreasing in the step size ν and increasing in the number of blocks $|\mathcal{B}|$. Given the terms in γ, it is evident that $\gamma \in [0, 1)$, so progress is guaranteed in each step. Further, the last term of the rate in (11) is similar to the reciprocal of the condition number of the Hessian Q (using λ_{pmin} instead of λ_{min}). This matches the common notion that gradient methods converge faster for well-posed problems with condition numbers close to one (Boyd and Vandenberghe, 2004).

The convergence of component-wise L_2-Boosting follows immediately from (11), given that the component-wise procedure is just a special case of the block-wise counterpart with $|\mathcal{B}| = p$ and $L_{\mathcal{B}} = L_{CLS}$ where L_{CLS} is the component-wise Lipschitz constant. This result is reassuring as it matches convergence results of Freund et al. (2017) which consider the special case of component-wise L_2-Boosting with standardized predictors, i.e., $L_{CLS} = 1$.

3.2.1 Regression Splines

Following Remark 4 (with further discussion in Appendix B.5.3), we know that block-wise boosting with penalized loss differs from usual GBCD as it uses the gradient of the unpenalized problem. Similar to the case of joint updates, this means that boosting is neglecting any penalization in previous iterations. As a consequence, the loss in the convergence result of Theorem 2 corresponds to the unpenalized loss, leading to convergence to the unpenalized instead of the penalized fit. As practitioners are usually interested in the latter, this property might not be desirable. In this case, however, the previous results also provide a way forward by using GBCD instead of boosting as fitting routine of the penalized problem. As the result in Theorem 2 also holds for the usual GBCD, but in terms of the penalized loss, optimizing penalized regression splines with GBCD will converge to the penalized fit.

Corollary 3. *Using GBCD with GSQ update scheme to fit regression spline models of the form* (6) *that are μ-PL and L-smooth in their parameters converges to an optimal solution of the regression spline problem with rate stated in Theorem 2.*

A prominent alternative fitting procedure for regression splines or additive models is backfitting (Friedman and Stuetzle, 1981). While convergence guarantees for backfitting have been analyzed by Buja et al. (1989) and Ansley and Kohn (1994), to the best of our knowledge, no explicit convergence rate has been derived to this date, making the rate in Corollary 3 the first convergence rate for regression spline problems.

3.2.2 Cubic Smoothing Splines

L_2-Boosting with cubic smoothing splines (CSS) was proposed in the seminal work of Bühlmann and Yu (2003). As CSS penalize the second differences of the fitted function, this can be seen as a continuous generalization of boosting with regression splines using second-order difference penalties. Using previous findings, we can also establish the convergence of component-wise L_2-Boosting with multiple CSS (proof in Appendix B.8). Interestingly, the result in Proposition 3 holds independently of the selection rule (greedy, cyclic, random).

Proposition 3. *Let $f^{[k]}$ be the fitted function values of component-wise L_2-Boosting with cubic smoothing splines after k iterates and f^* be the saturated model (perfect fit). Then $f^{[k]} \to f^*$ for $k \to \infty$.*

3.2.3 Generalized Boosting Approaches

We now show how to extend previous results to models with exponential family distribution assumption. In

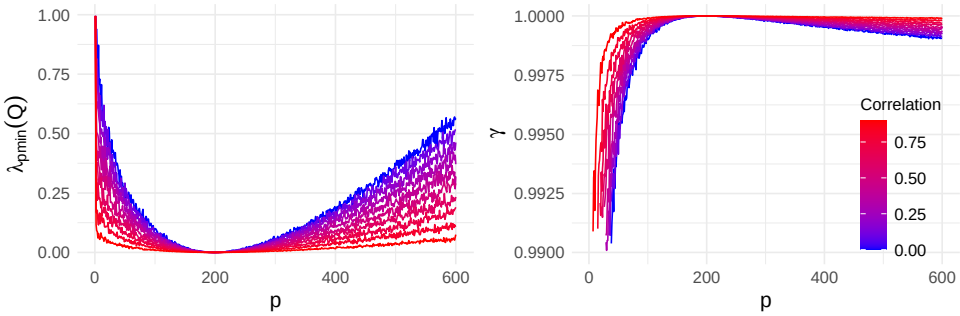

Figure 3: *Smallest non-zero eigenvalue of Q (left) and convergence rate γ as given in* (11) *(right) for component-wise boosting for a linear model with varying pairwise correlation ρ between predictor variables (color) for fixed $n = 200$, $\nu = 1$, and varying p (x-axis).*

this case, the density of the response can be written as

$$p_\theta(y) = \exp\left[\{y\theta - \varsigma(\theta)\}/\phi + c(\phi, y)\right], \qquad (12)$$

where the terms θ, $\varsigma(\theta)$, ϕ and $c(\phi, y)$ depend on the exponential family distribution (see, e.g., Fahrmeir et al., 2013; Wood, 2017). The loss function w.r.t. the functional f is then defined by

$$\ell(f) = \ell(\theta) = \{y\theta(f) - \varsigma(\theta(f))\}/\phi + c(\phi, y), \qquad (13)$$

where f is used to model the natural parameter θ. In the so called canonical link case, $f = \theta = h^{-1}(\mathbb{E}(Y|x))$, the log-likelihood in (13) can be shown to be strictly convex (cf. Appendix A.4). For other link functions this property cannot be guaranteed. As strictly convex problems are μ-PL on a compact set (Karimi et al., 2016), (13) is μ-PL as long as its parameters are bounded. In case the condition of L-smoothness is also fulfilled for the respective exponential family problem at hand, e.g., the logistic loss is known to be $1/4$-smooth, then the following proposition holds.

Proposition 4. *Block-wise boosting with exponential family loss ℓ corresponds to GBCD with GSQ update scheme as long as the Hessian upper bound condition is fulfilled. If ℓ is μ-PL and L-smooth w.r.t. the parameters $\beta \in \mathbb{R}^p$, the procedure converges with rate as given in Theorem 2.*

In Section 4.3, we investigate the examples of Binomial and Poisson boosting. For the latter, we show that a sufficiently small step size is essential to fulfill the upper bound condition and thus to obtain convergence. In contrast, convergence of Binomial boosting follows directly from Proposition 4 for arbitrary step size $\nu \in (0, 1]$, as the upper bound condition is naturally fulfilled for the Hessian Q^{log} of the logistic model given that $Q^{log}_{bb} \preceq \frac{1}{4} X_b^\top X_b \preceq X_b^\top X_b \; \forall b \in \mathcal{B}$.

Next to the derivation of Proposition 4 in Appendix B.9, we also discuss other loss functions outside the exponential family class, such as the L_1, Huber and Cox proportional hazards loss, in Appendix B.10.

3.2.4 Boosting distributions

When using BAMs for distributional regression, i.e., learning multiple parameters of a parametric distribution, convexity (and other) assumptions usually do not hold. While a general analysis is challenging, we here study the case of using BAMs to learn both the mean and scale parameters of a Gaussian distribution. To this end, we define the loss function

$$\ell(\beta, \xi) = \ell(y, (f_\psi, f_\sigma)) = -\log p_{\mathcal{N}(\psi,\sigma)}(y) \qquad (14)$$

as the negative log-likelihood of a Gaussian distribution and parameterize the distribution's mean $\psi = f_\psi(x; \beta)$ and standard deviation $\sigma = f_\sigma(z; \xi)$ both with individual functions $f_\psi(x; \beta) = x^\top \beta$ with parameters β and $f_\sigma(z; \xi) = \exp(z^\top \xi)$ with parameters ξ. Then the following holds:

Proposition 5. *The problem* (14) *is biconvex in (β, ξ).*

While general convergence guarantees for biconvex optimization are already challenging to obtain (Gorski et al., 2007), another issue that complicates convergence guarantees is that (14) is not L-smooth in the parameters ξ (cf. Appendix B.11). This can lead to convergence issues as demonstrated in Appendix D.4.

4 NUMERICAL EXPERIMENTS

In the following, we numerically demonstrate our theoretical findings.

4.1 Convergence Rates

In Figure 3, we investigate the influence of different degrees of pairwise correlations between predictors on the convergence rate γ and the μ-PL constant ($\lambda_{pmin}(Q)$) for a linear model with an increasing number of predictors. Generally, γ is close to one, reflecting slow convergence, consistent with the typical slow (over-) fitting behavior of boosting (Bühlmann and Hothorn, 2007). Lower pairwise correlations lead to faster convergence, but as the number of predictors increases, γ rises until $p = n$. In the underdetermined case ($n < p$), γ decreases slightly while staying near one, driven by

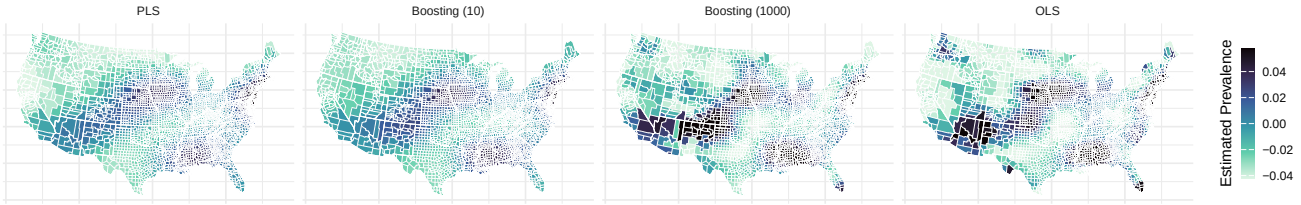

Figure 4: *Mean-centered spatial effect of Covid-19 prevalence in the United States obtained with BAMs. From left to right: Penalized least squares (PLS); early-stopped BAM; BAM with a large number of iterations; unpenalized least squares fit (OLS) to which boosting is converging to.*

changes in the smallest non-zero eigenvalue of the system. We further investigate the impact of different condition numbers on the linear convergence rate in Figure 6 (in Appendix D). In line with the theoretical results obtained from Theorem 2, the convergence is slower for higher condition numbers of the problem.

4.2 Convergence to the Unpenalized Model

Another implication of our analysis is the solution the boosting algorithm is converging to. As discussed in previous sections, this corresponds to the unpenalized solution for linear, ridge, (cf. Figure 1) and regression spline boosting (cf. Figure 2). This can have drastic effects in cases where penalized base learners are inevitable such as in spatial modeling. To demonstrate this, we again model the Covid-19 prevalence using BAMs, but now spatially over the entire US. Figure 4 shows the estimated spatial effect surface of the penalized fit (PLS), boosting after 10 and 1000 iterations, and the unpenalized fit (OLS). Results clearly indicate that boosting can roughly match the PLS estimation, but when running the algorithm further it will — despite explicit penalization — converge to the unpenalized model fit. In Appendix D.1, we demonstrate this phenomenon along with various other boosting applications such as boosting with ridge penalty and P-splines as well as the more complex setup of function-on-function regression boosting.

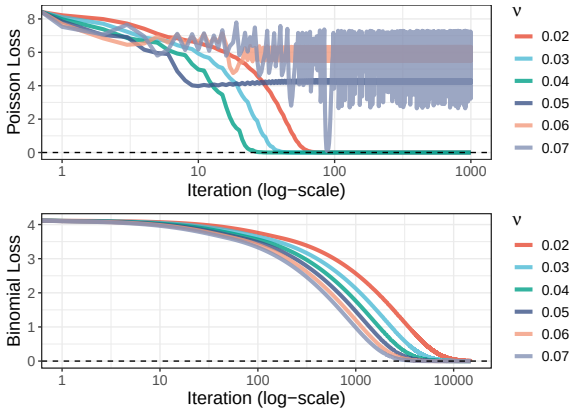

Figure 5: *Loss path for Poisson (top) and Binomial (bottom) BAMs with different learning rates (colors) showing potential convergence issues for Poisson BAMs.*

4.3 Exponential Family Boosting

Lastly, we demonstrate findings from Section 3.2.3 to show differences in convergence for different distribution families. To this end, we simulate a Poisson and Binomial GLM model and then run different BAMs with learning rates $\nu \in \{0.02, 0.03, \ldots, 0.07\}$ for both distributions for a maximum of 1000 iterations.

Results: As derived in Proposition 4, we observe convergence for Binomial BAMs for all learning rates in Figure 5, whereas for Poisson BAMs, half of the defined learning rates do not upper bound the Hessian correctly and result in oscillating parameter updates and non-convergence. We show the issue of non-convergence for Poisson BAMs also on observational data in Appendix D.2 and provide further details about the experiments in this and previous sections in Appendix D.

5 DISCUSSION

Boosted additive models (BAMs) are an indispensable toolbox in many applications. Understanding the inner workings of BAMs and their induced implicit regularization is key to insights into their theoretical properties. In this work, we characterize the implicit shrinkage of BAMs by relating them to solution paths of explicitly regularized problems. We further establish an important link between greedy block-wise boosting and greedy block coordinate descent with a particular update scheme. Using this equivalence, we derive novel convergence results for BAMs.

Our investigation also uncovers several pathologies of BAMs. We show that boosting penalized models neglects the penalization in previous steps and therefore converges to the unpenalized fit. For exponential family loss, we show that the implicit Hessian approximation in BAMs might induce non-convergence.

Limitations and future research While our work provides important theoretical insights into the inner workings of boosting additive models, proper statistical inference for BAMs that accounts for implicit regularization and model selection remains an open challenge. We hope our findings inspire continued research in this direction.

References

Agarwal, R., Melnick, L., Frosst, N., Zhang, X., Lengerich, B., Caruana, R., and Hinton, G. E. (2021). Neural additive models: Interpretable machine learning with neural nets. *Advances in Neural Information Processing Systems*, 34:4699–4711.

Ansley, C. F. and Kohn, R. (1994). Convergence of the backfitting algorithm for additive models. *Journal of the Australian Mathematical Society*, 57(3):316–329.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Brockhaus, S., Melcher, M., Leisch, F., and Greven, S. (2017). Boosting flexible functional regression models with a high number of functional historical effects. *Statistics and Computing*, 27:913–926.

Brockhaus, S., Rügamer, D., and Greven, S. (2020). Boosting functional regression models with fdboost. *Journal of Statistical Software*, 94(10):1–50.

Bühlmann, P. and Hothorn, T. (2007). Boosting Algorithms: Regularization, Prediction and Model Fitting. *Statistical Science*, 22(4):477 – 505.

Buja, A., Hastie, T., and Tibshirani, R. (1989). Linear smoothers and additive models. *The Annals of Statistics*, pages 453–510.

Bühlmann, P. and Yu, B. (2003). Boosting with the l2 loss: Regression and classification. *Journal of the American Statistical Association*, 98(462):324–339.

Chalvidal, M., Serre, T., and VanRullen, R. (2023). Learning functional transduction. In *Advances in Neural Information Processing Systems*, volume 36, pages 73852–73865. Curran Associates, Inc.

Collins, M., Schapire, R. E., and Singer, Y. (2002). Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 48:253–285.

de Boor, C. (1978). *A Practical Guide to Spline*, volume 27. Springer.

DiCiccio, T. J. and Efron, B. (1996). Bootstrap confidence intervals. *Statistical science*, 11(3):189–228.

Edmonson, J. H., Fleming, T. R., Decker, D. G., Malkasian, G. D., Jorgensen, E. O., Jefferies, J. A., Webb, M. J., and Kvols, L. K. (1979). Different chemotherapeutic sensitivities and host factors affecting prognosis in advanced ovarian carcinoma versus minimal residual disease. *Cancer Treat Rep*, 63(2):241–247.

Eilers, P. H. C. and Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. *Statistical Science*, 11(2):89 – 121.

Fahrmeir, L., Kneib, T., Lang, S., and Marx, B. (2013). *Regression: Models, Methods and Applications*. Springer.

Fenske, N., Kneib, T., and Hothorn, T. (2011). Identifying risk factors for severe childhood malnutrition by boosting additive quantile regression. *Journal of the American Statistical Association*, 106(494):494–510.

Freund, R. M., Grigas, P., and Mazumder, R. (2017). A new perspective on boosting in linear regression via subgradient optimization and relatives. *The Annals of Statistics*, 45(6):2328–2364.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232.

Friedman, J. H. and Stuetzle, W. (1981). Projection pursuit regression. *Journal of the American statistical Association*, 76(376):817–823.

Gorski, J., Pfeuffer, F., and Klamroth, K. (2007). Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical methods of operations research*, 66(3):373–407.

Green, P. J. and Silverman, B. W. (1994). *Nonparametric regression and generalized linear models*. Chapman & Hall/CRC Press.

Hofner, B., Mayr, A., and Schmid, M. (2016). gamboostLSS: An R Package for Model Building and Variable Selection in the GAMLSS Framework. *Journal of Statistical Software*, 74(1):1–31.

Hothorn, T. and Bühlmann, P. (2006). Model-based boosting in high dimensions. *Bioinformatics*, 22(22):2828–2829.

Hothorn, T., Bühlmann, P., Kneib, T., Schmid, M., and Hofner, B. (2010). Model-based boosting 2.0. *Journal of Machine Learning Research*, 11(71):2109–2113.

Karimi, H., Nutini, J., and Schmidt, M. (2016). Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*, pages 795–811. Springer.

Koenker, R. and Bassett Jr, G. (1982). Robust tests for heteroscedasticity based on regression quantiles. *Econometrica: Journal of the Econometric Society*, pages 43–61.

Locatello, F., Raj, A., Karimireddy, S. P., Rätsch, G., Schölkopf, B., Stich, S., and Jaggi, M. (2018). On matching pursuit and coordinate descent. In *International Conference on Machine Learning*, pages 3198–3207. PMLR.

Loprinzi, C. L., Laurie, J. A., Wieand, H. S., Krook, J. E., Novotny, P. J., Kugler, J. W., Bartel, J., Law, M., Bateman, M., and Klatt, N. E. (1994). Prospective evaluation of prognostic variables from

patient-completed questionnaires. north central cancer treatment group. *Journal of Clinical Oncology*, 12(3):601–607.

Lou, Y., Caruana, R., and Gehrke, J. (2012). Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 150–158.

Lu, H., Karimireddy, S. P., Ponomareva, N., and Mirrokni, V. (2020). Accelerating gradient boosting machines. In *International Conference on Artificial Intelligence and Statistics*, pages 516–526. PMLR.

Lu, H. and Mazumder, R. (2020). Randomized gradient boosting machine. *SIAM Journal on Optimization*, 30(4):2780–2808.

Maier, E.-M., Stöcker, A., Fitzenberger, B., and Greven, S. (2021). Additive density-on-scalar regression in bayes hilbert spaces with an application to gender economics. *arXiv preprint arXiv:2110.11771*.

Mason, L., Baxter, J., Bartlett, P., and Frean, M. (1999). Boosting algorithms as gradient descent. In Solla, S., Leen, T., and Müller, K., editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press.

Mayr, A., Fenske, N., Hofner, B., Kneib, T., and Schmid, M. (2012). Generalized additive models for location, scale and shape for high dimensional data—a flexible approach based on boosting. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 61(3):403–427.

Meir, R. and Rätsch, G. (2003). *An Introduction to Boosting and Leveraging*, volume 2600, pages 119–184. Springer.

Nesterov, Y. (2004). *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Nesterov, Y. (2012). Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362.

Nori, H., Jenkins, S., Koch, P., and Caruana, R. (2019). Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*.

Nutini, J., Laradji, I., and Schmidt, M. (2022). Let's make block coordinate descent converge faster: Faster greedy rules, message-passing, active-set complexity, and superlinear convergence. *Journal of Machine Learning Research*, 23(131):1–74.

Nutini, J., Schmidt, M., Laradji, I., Friedlander, M., and Koepke, H. (2015). Coordinate descent converges faster with the gauss-southwell rule than random selection. In *International Conference on Machine Learning*, pages 1632–1641. PMLR.

Oliver, M. A. and Webster, R. (1990). Kriging: a method of interpolation for geographical information systems. *International Journal of Geographical Information System*, 4(3):313–332.

Radenovic, F., Dubey, A., and Mahajan, D. (2022). Neural basis models for interpretability. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K., editors, *Advances in Neural Information Processing Systems*.

Rätsch, G., Mika, S., and Warmuth, M. K. K. (2001). On the convergence of leveraging. In Dietterich, T., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

Rue, H. and Held, L. (2005). *Gaussian Markov random fields: theory and applications*. Chapman and Hall/CRC.

Scheipl, F., Staicu, A.-M., and Greven, S. (2015). Functional additive mixed models. *Journal of Computational and Graphical Statistics*, 24(2):477–501.

Schmid, M. and Hothorn, T. (2008). Boosting additive models using component-wise p-splines. *Computational Statistics & Data Analysis*, 53(2):298–311.

Stöcker, A., Steyer, L., and Greven, S. (2023). Functional additive models on manifolds of planar shapes and forms. *Journal of Computational and Graphical Statistics*, 32(4):1600–1612.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288.

Wahba, G. (1990). *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, Philadelphia.

Wahltinez, O., Cheung, A., Alcantara, R., Cheung, D., Daswani, M., Erlinger, A., Lee, M., Yawalkar, P., Lê, P., Navarro, O. P., et al. (2022). Covid-19 opendata a global-scale spatially granular meta-dataset for coronavirus disease. *Scientific data*, 9(1):162.

Wood, S. N. (2017). *Generalized additive models: an introduction with R*. CRC press.

Zhang, T. and Yu, B. (2005). Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, 33(4):1538–1579.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] See Section 2 in particular Algorithm 1.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes] Analysis investigates several theoretical properties of an existing well-known algorithm (complexity of algorithm known from previous research).
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes] The code is supplied as supplementary material.
2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes] All assumptions related to each theoretical result are given along each theoretical result respectively.
   (b) Complete proofs of all theoretical results. [Yes] Proofs for all theoretical results are provided in the supplementary material.
   (c) Clear explanations of any assumptions. [Yes] Assumptions are discussed and explained either along the theoretical result itself or in a dedicated section.
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes] See supplementary material.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes] Details are provided in the supplementary material.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Not Applicable]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes] See Appendix E.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable] The paper does not use existing assets.
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable] The paper does not release new assets.
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

SUPPLEMENTARY MATERIAL

A Preliminaries

A.1 Function properties

To derive theoretical convergence results for gradient optimization methods, conditions such as strong convexity and L-smoothness have been considered. A common notion of smoothness used in convergence analysis (e.g., Boyd and Vandenberghe, 2004; Nesterov, 2012, 2004) is the following:

Definition 1. *A function $\ell : \mathbb{R}^p \to \mathbb{R}$ is L-smooth with $L > 0$, if $\forall \beta, \tilde{\beta} \in \mathbb{R}^d$:*

$$\|\nabla \ell(\beta) - \nabla \ell(\tilde{\beta})\| \leq L\|\beta - \tilde{\beta}\|.$$

L-smoothness is sometimes also referred to as L-Lipschitz continuous gradient $\nabla \ell$. From Definition 1 we can deduce that an L-smooth function ℓ fulfills $\forall \beta, \tilde{\beta} \in \mathbb{R}^p$

$$\ell(\tilde{\beta}) \leq \ell(\beta) + \langle \nabla \ell(\beta), \tilde{\beta} - \beta \rangle + \frac{L}{2}\|\tilde{\beta} - \beta\|^2. \tag{15}$$

By contrast, a function that is strongly convex, or more precisely μ-strongly convex with $\mu > 0$, fulfills $\forall \beta, \tilde{\beta} \in \mathbb{R}^p$

$$\ell(\tilde{\beta}) \geq \ell(\beta) + \langle \nabla \ell(\beta), \tilde{\beta} - \beta \rangle + \frac{\mu}{2}\|\tilde{\beta} - \beta\|^2. \tag{16}$$

For twice-differentiable objective functions, the conditions in (15) and (16) provide lower and upper bounds on eigenvalues of the Hessian, $\mu I \preceq \nabla^2 \ell(\beta) \preceq LI \, \forall \beta \in \mathbb{R}^p$, where I is the identity matrix. More recently, Karimi et al. (2016) showed that it is sufficient to consider the PL-inequality instead of strong convexity to derive convergence rates for iterative gradient methods.

Definition 2. *Karimi et al. (2016) A function $f : \mathbb{R}^p \to \mathbb{R}$ is μ-PL with some $\mu > 0$, if $\forall \beta \in \mathbb{R}^p$:*

$$\frac{1}{2}\|\nabla \ell(\beta)\|^2 \;\geq\; \mu \, (\ell(\beta) - \ell^*). \tag{17}$$

In Definition 2, ℓ^* denotes the optimal function value of the optimization problem in (1). The notation ℓ^* instead of $\ell(\beta^*)$ is used as the optimal function value is unique, whereas the solution β^* to the problem $\ell(\beta)$ with $\ell^* = \ell(\beta^*)$ does not have to be unique in case $\ell(\beta)$ is μ-PL. This is in contrast to strong convexity, where the solution β^* is guaranteed to be unique. Definition 2 is more general than strong-convexity and several important problems do not fulfill strong convexity but the more general PL-inequality.

Convex quadratic problems The above conditions can be shown to hold for the important class of convex quadratic problems, that can be written in the following form

$$\min_{\beta \in \mathbb{R}^p} \ell(\beta) = \min_{\beta \in \mathbb{R}^p} \frac{1}{2}\beta^\top Q \beta + q_1^\top \beta + q_0, \tag{18}$$

where Q denotes a symmetric positive semi-definite (p.s.d.) matrix. A simple derivation shows that the least squares and P-spline problem can be written in this form with Q corresponding to the Hessian ($Q_{LS} = X^\top X$ and $Q_{PLS} = X^\top X + \lambda P$). The problem in (18) is L-smooth with some Lipschitz constant $L \leq \lambda_{max}(Q)$, where $\lambda_{max}(Q)$ denotes the largest eigenvalue of Q.

Further, Freund et al. (2017); Karimi et al. (2016) showed that any problem that can be written in the form of (18) is μ-PL with $\mu = \lambda_{pmin}(Q)$, where $\lambda_{pmin}(Q)$ denotes the smallest non-zero eigenvalue of Q. For p.d. Q (all eigenvalues positive), we can even recover the stronger condition of strong convexity, as we get that $\mu = \lambda_{min}(Q) > 0$. This indeed makes a difference e.g. when considering the least squares problem in the overdetermined $n > d$-setting ($X^\top X$ p.d.) compared to the underdetermined setting with $n < d$ ($X^\top X$ only p.s.d.) as frequently encountered in high-dimensional statistics.

A different formulation of L-smoothness As we are dealing with matrix updates and not just single coordinate updates in our derivations, it will prove beneficial to state the L-smoothness condition in the form of

$$\|\nabla_b \ell(\beta + U_b \upsilon) - \nabla_b \ell(\beta)\|_{H_b^{-1}} \leq \|\upsilon\|_{H_b}, \tag{19}$$

where $\beta \in \mathbb{R}^p$, $H_b \in \mathbb{R}^{|b|\times|b|}$ and $\upsilon \in \mathbb{R}^{|b|}$. When considering twice-differentiable functions for ℓ, (19) essentially states that H_b must provide an upper bound with respect to the block of the Hessian belonging to the coordinates of block b, i.e., $\nabla^2_{bb}\ell(\beta) \preceq H_b$. Note that (19) contains the usual L-smoothness condition as described in (15) as a special case by choosing $H_b = L_b I$, with L_b being the Lipschitz constant for block b (Nutini et al., 2022). Conversely, (19) can be shown to hold, whenever the L-smoothness condition (15) is assumed to hold, as we have that $L_b < L$, $\forall b \in \mathcal{B}$.

A.2 Optimization routines

In the subsequent convergence analyses, we will consider both component-wise as well as block-wise gradient methods.

A.2.1 Greedy (block) coordinate descent

Greedy coordinate descent (GCD) with constant step size $\nu \in (0, 1]$ is an iterative method in which the update steps are performed component-wise as

$$\beta^{[k+1]} = \beta^{[k]} - \nu\nabla_{i_k}\ell\left(\beta^{[k]}\right)e_{i_k}, \tag{20}$$

where e_{i_k} is the unit vector corresponding to the variable $i_k \in \{1, \dots, p\}$ selected to be updated at step k.

Greedy block coordinate descent (GBCD) works similarly to GCD but considers blocks of variables instead of single variables to be updated in each step. Thus, the d variables are partitioned into disjoint blocks, where each block is indexed by $b \in \mathcal{B}$. Overall, one obtains a total number of $|\mathcal{B}|$ blocks, where $|\cdot|$ denotes the cardinality. Note that by considering a single coordinate per block, we recover GCD, which is why the latter can be seen as a special instance of GBCD. The block-wise updates in GBCD are of the form $\beta^{[k+1]} = \beta^{[k]} + \nu U_{b_k}\varkappa_{b_k}$, with step size $\nu \in (0, 1]$ and U_{b_k} a block-wise matrix with an identity matrix block for the selected block b_k at the current step k and else zeros. With $\varkappa_{b_k}$ we denote the direction in which the selected block will be updated. This direction can be chosen to correspond to the block-wise steepest descent direction and thus to the negative gradient with respect to the variables of the selected block $\varkappa_{b_k} = -\nabla_{b_k}\ell\left(\beta^{[k]}\right)$, or can be extended to a matrix update by scaling with matrix H_{b_k} to yield

$$\varkappa_{b_k} = -(H_{b_k})^{-1}\nabla_{b_k}\ell\left(\beta^{[k]}\right), \tag{21}$$

where H_{b_k} could correspond to the respective block of the Hessian or an upper bound of the latter (Nutini et al., 2022).

A.2.2 Update rules

Gauss-Southwell(-Lipschitz) A common selection strategy for the selection of the i_kth coordinate in GCD is to use the Gauss-Southwell (GS) selection rule

$$i_k = \arg\max_i |\nabla_i\ell(\beta^{[k]})|. \tag{22}$$

A more elaborated update routine, the so called Gauss-Southwell-Lipschitz (GSL) rule, was proposed by Nutini et al. (2015) and is given by

$$i_k = \arg\max_i \frac{|\nabla_i\ell(\beta^{[k]})|}{\sqrt{L_i}}. \tag{23}$$

The GSL rule not only takes the gradient into account but also the curvature along each component by scaling with respect to the Lipschitz constant L_i of the i-th component. This second-order information in the GSL rule can be incorporated into the update step (20) as well, yielding

$$\beta^{[k+1]} = \beta^{[k]} - \nu\frac{1}{L_{i_k}}\nabla_{i_k}\ell\left(\beta^{[k]}\right)e_{i_k}. \tag{24}$$

Gauss-Southwell-Quadratic Analogously to the block update in (21), a greedy block selection rule called Gauss-Southwell-Quadratic (GSQ) is defined by

$$b_k \quad = \quad \underset{b \in \mathcal{B}}{\arg\max} \quad \{\|\nabla_b \ell(\beta^{[k]})\|_{H_b{}^{-1}}\}, \tag{25}$$

where, $\|\cdot\|_H = \sqrt{\langle H\cdot, \cdot\rangle}$ denotes a general quadratic norm (a proper norm as long as H is a positive definite matrix) (Nutini et al., 2022).

A.3 Linear operators

We will prove Proposition 3 below using operator norms. For this, we will consider linear operators.

Properties 1. *Let $T : \mathbb{R}^n \to \mathbb{R}^n$ be a linear operator and $z, c \in \mathbb{R}^n$. Further let $T_1 : \mathbb{R}^n \to \mathbb{R}^n$ and $T_2 : \mathbb{R}^n \to \mathbb{R}^n$ be two operators. Then the following holds:*

1. *Operator Norm:* $\|T\|^* = \sup_{\|z\|=1} \|Tz\| = \sup_{\|z\|\neq 0} \|T\frac{z}{\|z\|}\|$;
2. $\|T\|\|c\| = \|c\|\|T\frac{c}{\|c\|}\| \leq \|c\| \sup_{\|c\|\neq 0} \|T\frac{c}{\|c\|}\| = \|c\|\|T\|^*$;
3. $\|T_1T_2\|^* = \sup_{\|z\|\neq 0} \frac{\|T_1T_2z\|}{\|z\|} = \sup_{\|z\|\neq 0} (\frac{\|T_1T_2z\|}{\|T_2z\|}\frac{\|T_2z\|}{\|z\|}) \leq \sup_{\|T_2z\|\neq 0} \frac{\|T_1T_2z\|}{\|T_2z\|} \sup_{\|z\|\neq 0} \frac{\|T_2z\|}{\|z\|} = \|T_1\|^*\|T_2\|^*$.

A.4 Exponential family

For any exponential family model, the density of the response can be written in the following form

$$p_\theta(y) = \exp\left[\{y\theta - \varsigma(\theta)\}/\phi + c(\phi, y)\right], \tag{26}$$

where the terms θ, $\varsigma(\theta)$, ϕ and $c(\phi, y)$ depend on the exponential family distribution considered (Fahrmeir et al., 2013; Wood, 2017). The parameter θ is called the canonical parameter and is often written as $\theta(\psi)$, due to its dependence on the conditional expectation of the outcome given features, $\mathbb{E}(Y|x) = \psi$. In this setup, one aims to estimate a function f, given some response function $h(\cdot)$, such that $h(f_i) = \mathbb{E}[Y_i] = \psi_i$. For generalized linear models (GLMs) f is linear in the estimated parameter β, as we assume $f = X\beta$. For each exponential family, there exists a unique canonical link function $g = h^{-1}$, such that $\theta_i = f_i$ (Fahrmeir et al., 2013). Choosing the canonical link has several theoretical benefits. First, the log-likelihood $\log(p_\theta(y))$ used to estimate the model, can be written both in terms of θ or f. Thus, the loss function is defined by

$$\ell(f) = \ell(\theta(f)) = \{y\theta(f) - \varsigma(\theta(f))\}/\phi + c(\phi, y). \tag{27}$$

Another key feature of the canonical link is that (27) is strictly convex in f. For other link functions this property cannot be guaranteed. We can make this notion more explicit by looking at the Hessian of (27). We do so by considering GLMs, for which (27) becomes a function of the parameter β. Using the canonical link, the Hessian of the log-likelihood simplifies and coincides with the Fisher information matrix (Wood, 2017). The latter (derived, e.g., in Fahrmeir et al., 2013) corresponds to

$$\nabla^2 \ell(\beta) = X^\top W X \tag{28}$$

with $W = \text{diag}(\ldots, \tilde{w}_i, \ldots)$ and

$$\tilde{w}_i = \frac{(h'(f_i))^2}{\varsigma''(\theta_i)\phi}. \tag{29}$$

By looking at the definition of the respective terms for different exponential family distributions, it becomes clear that $\varsigma''(\theta) > 0$ and $\phi > 0$ (Fahrmeir et al., 2013; Wood, 2017). With the canonical response function being strictly monotonic, the numerator must be greater than zero as well. Therefore, given the canonical link, the weights are positive and the Hessian positive definite. Thus, as long as X is full rank, the log-likelihood is strictly convex.

B Proofs and derivations

B.1 Derivation of Equation (5)

Derivation. As L_2-Boosting for linear models with joint updates corresponds to least squares fitting of residuals from the previous iteration, we can write the parameters at each iteration recursively (with $\beta^{[0]} := 0$):

$$
\begin{aligned}
\beta^{[1]} &= \beta^{[0]} + \nu(X^\top X)^{-1}X^\top y = \nu\beta^{OLS} \\
\beta^{[2]} &= \beta^{[1]} + \nu(X^\top X)^{-1}X^\top(y - X\beta^{[1]}) = (1-\nu)\beta^{[1]} + \nu\beta^{OLS} = [(1-\nu)\nu + \nu]\ \beta^{OLS} \\
\beta^{[3]} &= \beta^{[2]} + \nu(X^\top X)^{-1}X^\top(y - X\beta^{[2]}) = [(1-\nu)^2\nu + (1-\nu)\nu + \nu]\ \beta^{OLS} \\
&\vdots \\
\beta^{[k]} &= (1-\nu)\beta^{[k-1]} + \nu\beta^{OLS} \\
&= [\sum_{m=0}^{k-1} \nu(1-\nu)^m]\ \beta^{OLS} = \nu\left(\frac{1-(1-\nu)^k}{\nu}\right)\beta^{OLS} = \underbrace{\left(1-(1-\nu)^k\right)}_{:=\delta(\nu)^{[k]}}\beta^{OLS},
\end{aligned}
$$

where the form of the parameter at iteration k follows by induction. The last equality follows from the fact that we have a geometric series, which converges as $(1-\nu) < 1$ with a learning rate $\nu \in (0, 1)$. Using this notation, the fitted function at each iteration k can be seen as a linear smoother, by writing $f^{[k]} = \delta(\nu)^{[k]}X(X^\top X)^{-1}X^\top y$. For $\nu = 1$ we get convergence in one step. Moreover, we can observe that $\delta(\nu)^{[k]} \overset{k\to\infty}{\to} 1$ for arbitrary $\nu \in (0, 1)$, such that boosting paths converge to the respective ordinary least squares solution (OLS). □

B.2 Proof of Proposition 1

Proof. As L_2-Boosting for linear models with joint updates and quadratic penalization corresponds to penalized least squares fitting of residuals from the previous iteration, we can write the parameters at each iteration recursively (with $\beta^{[0]} := 0$):

$$
\begin{aligned}
\beta^{[1]} &= \beta^{[0]} + \nu(X^\top X + \lambda P)^{-1}X^\top y = \nu\beta^{PLS} \\
\beta^{[2]} &= \beta^{[1]} + \nu(X^\top X + \lambda P)^{-1}X^\top(y - X\beta^{[1]}) = (I - \nu(X^\top X + \lambda P)^{-1}X^\top X)\nu\beta^{PLS} + \nu\beta^{PLS} \\
\beta^{[3]} &= \beta^{[2]} + \nu(X^\top X + \lambda P)^{-1}X^\top(y - X\beta^{[2]}) \\
&= \left[(I - \nu(X^\top X + \lambda P)^{-1}X^\top X)^2\nu + (I - \nu(X^\top X + \lambda P)^{-1}X^\top X)\nu + \nu\right]\ \beta^{PLS} \\
&\vdots \\
\beta^{[k]} &= (I - \nu(X^\top X + \lambda P)^{-1}X^\top X)\beta^{[k-1]} + \nu\beta^{PLS} \\
&= [\sum_{m=0}^{k-1} \nu(I - \nu(X^\top X + \lambda P)^{-1}X^\top X)^m]\ \beta^{PLS}
\end{aligned}
$$

where the form of the parameter at iteration k follows by induction. If we let the number of iterations grow to infinity, we get

$$
\beta^{[k]} \overset{k\to\infty}{\longrightarrow} [\sum_{m=0}^{\infty} \nu(I - \nu(X^\top X + \lambda P)^{-1}X^\top X)^m]\beta^{PLS} = [\nu\sum_{m=0}^{\infty} T^m]\beta^{PLS},
$$

where the latter can be recognized as a Neumann series with operator $T := (I - \nu(X^\top X + \lambda P)^{-1}X^\top X)$ with $\nu \in (0, 1]$. In the following, we differentiate between two cases. First, we discuss the case of X having full column rank ($X^\top X$ is p.d.) and subsequently the rank deficient case where X has reduced column rank ($X^\top X$ is p.s.d.).

Full column rank case: The Neumann series is known to converge if $\|T\|^* < 1$, where $\|\cdot\|^*$ denotes the operator norm. To show that this is the case, we can first write

$$
\|T\|^* = \|I - \nu R\|^* = 1 - \nu\lambda_{min}(R),
$$

with $R := (X^\top X + \lambda P)^{-1}X^\top X$. The eigenvalues of R are the same as the eigenvalues of a hat matrix of a penalized regression spline, which are known to be bounded by zero and one (Schmid and Hothorn, 2008). Further,

as R is the product of two symmetric p.d. matrices (as $X^\top X$ was assumed to be p.d.), R is positive definite and thus $\lambda_{min}(R) > 0$. From this it follows that $\|T\|^* < 1$ and the Neumann series above converges as

$$\nu \sum_{m=0}^{\infty} T^m = \nu(I-T)^{-1} = \nu(I - (I - \nu(X^\top X + \lambda P)^{-1} X^\top X))^{-1} = (X^\top X)^{-1}(X^\top X + \lambda P)$$

Therefore:

$$\beta^{[k]} \overset{k\to\infty}{\longrightarrow} [\nu \sum_{m=0}^{\infty} T^m]\beta^{PLS} = (X^\top X)^{-1}(X^\top X + \lambda P)(X^\top X + \lambda P)^{-1} X^\top y = (X^\top X)^{-1} X^\top y$$

Rank-deficient case: Now consider the case that X has reduced column rank. This is a setting that often occurs when a P-spline basis expansion is used with multiple basis functions (determined by the number of knots). In the reduced rank case with $\text{rank}(X) = r < p$, the convergence of the Neumann series can no longer be guaranteed by the same arguments as above, as $\|T\|^* < 1$ may no longer hold. The largest eigenvalue might be one, due to the rank-deficiency. In the following, we consider $P = I$ (ridge boosting) w.l.o.g. as any p-spline can be rewritten in terms of a ridge penalty using a modified design matrix. To show convergence in this case, we first pre-multiply $(I - T)$ to the Neumann series above

$$(I-T)\nu \sum_{m=0}^{\infty} T^m = \nu \sum_{m=0}^{\infty} (I-T)T^m = \nu \sum_{m=0}^{\infty} T^m - T^{m+1} = \nu \left(I - \lim_{m\to\infty} T^m \right).$$

The last term can be shown to converge. Using a the full SVD $X = U\Sigma^{\frac{1}{2}} V^\top$, we can write

$$\begin{aligned}
I - \lim_{m\to\infty} T^m &= I - \lim_{m\to\infty} \left(I - \nu(X^\top X + \lambda P)^{-1} X^\top X \right)^m \\
&= I - \lim_{m\to\infty} \left(I - \nu V(\Sigma + \lambda I)^{-1} \Sigma V^\top \right)^m \\
&= I - V \lim_{m\to\infty} \left(I - \nu(\Sigma + \lambda I)^{-1}\Sigma \right)^m V^\top \\
&= I - V D_s^\perp V^\top = V D_s V^\top
\end{aligned}$$

with a diagonal matrix $D_s = \text{diag}(1_r^\top, 0_{p-r}^\top)$ containing r ones and $p - r$ zeros on the diagonal, and $D_s^\perp = I - D_s$. Given the previous display, the Neumann series above converges

$$\nu \sum_{m=0}^{\infty} T^m = \nu(I-T)^+ V D_s V^\top = (X^\top X)^+ (X^\top X + \lambda P) \cdot V D_s V^\top,$$

where $(X^\top X)^+$ denotes the Moore–Penrose generalized inverse of $X^\top X$ and where we used basic properties of the Moore–Penrose generalized inverse to simplify the expression. Therefore, in the case of rank deficiency, we get convergence of the parameters to the min-norm solution

$$\begin{aligned}
\beta^{[k]} \overset{k\to\infty}{\longrightarrow} [\nu \sum\nolimits_{m=0}^{\infty} T^m] \cdot \beta^{PLS} &= (X^\top X)^+ (X^\top X + \lambda P) \cdot V D_s V^\top \cdot (X^\top X + \lambda P)^{-1} X^\top y \\
&= (X^\top X)^+ \cdot V D_s (\Sigma + \lambda I)^{-1} (\Sigma + \lambda I) V^\top \cdot X^\top y \\
&= (X^\top X)^+ \tilde{V} \tilde{V}^\top X^\top y \\
&= (X^\top X)^+ X^\top y = X^+ y,
\end{aligned}$$

where we use in the second line again a full SVD with $X = U\Sigma^{\frac{1}{2}} V^\top$ and the fact that diagonal matrices commute, in the third line a compact SVD with $X = \tilde{U}\tilde{\Sigma}^{\frac{1}{2}} \tilde{V}^\top$ and the fact that $V D_s V^\top = \tilde{V}\tilde{V}^\top$, and in the forth the fact that $\tilde{V}\tilde{V}^\top$, the projection onto the row space of X, can be disregarded as $X\top y$ is already in the row space. Finally, $X^+ y$ denotes the well-known min-norm solution of the (underdetermined) least squares problem, i.e. when $n = r < p$. □

B.3 Proof of Theorem 1

Proof. From (7) in Proposition 1 we get an explicit expression for the parameters $\beta^{[k]}$ in each step k of penalized L_2-Boosting (with quadratic penalty) and joint updates. With $\beta^{PLS} := (X^\top X + \lambda P)^{-1} X^\top y$ this is:

$$\begin{aligned}\beta^{[k]} &= [\sum_{m=0}^{k-1} \nu(I - \nu(X^\top X + \lambda P)^{-1} X^\top X)^m] \; \beta^{PLS} \\ &= (X^\top X)^{-1}(X^\top X + \lambda P) \; [I - (I - \nu(X^\top X + \lambda P)^{-1} X^\top X)^k] \; \beta^{PLS}.\end{aligned}$$

In the second line, we used a telescope-sum argument. In order to find the explicit minimization problem that is implicitly solved by boosting in each step, we can derive the explicit problem for which the solution of the problem corresponds to the above parameter formula in each step. Considering quadratic minimization problems, we aim to find the penalty matrix Γ_k of the explicit minimization problem:

$$\ell_{\Gamma_k}(\beta) = \frac{1}{2}\|y - X\beta\|^2 + \frac{1}{2}\beta^\top \Gamma_k \beta, \tag{30}$$

for which the solution of the problem $\beta^*_{\Gamma_k} = (X^\top X + \Gamma_k)^{-1} X^\top y$ matches the above closed-form parameter expression of boosting (uniqueness of $\beta^*_{\Gamma_k}$ is guaranteed as X has full column rank). To match them, we equate the two and get

$$\begin{aligned}(X^\top X + \Gamma_k)^{-1} X^\top y \;&=\; (X^\top X)^{-1}(X^\top X + \lambda P) \; [I - (I - \nu(X^\top X + \lambda P)^{-1} X^\top X)^k] \; (X^\top X + \lambda P)^{-1} X^\top y \\ \Leftrightarrow \qquad (X^\top X + \Gamma_k)^{-1} \;&=\; (X^\top X)^{-1}(X^\top X + \lambda P) \; [I - (I - \nu(X^\top X + \lambda P)^{-1} X^\top X)^k] \; (X^\top X + \lambda P)^{-1}.\end{aligned}$$

Inverting both sides and subsequently rearranging terms, we obtain

$$\Gamma_k \;=\; -(X^\top X) + (X^\top X + \lambda P) \; [I - (I - \nu(X^\top X + \lambda P)^{-1} X^\top X)^k]^{-1} \; (X^\top X + \lambda P)^{-1}(X^\top X).$$

Again full column rank of X ensures the invertibility of the terms. Using the Woodbury matrix identity, we can rewrite the inverse such that certain terms cancel out

$$\begin{aligned}\Gamma_k \;&=\; -(X^\top X) + (X^\top X + \lambda P) \; (I + [(I - \nu(X^\top X + \lambda P)^{-1} X^\top X)^{-k} - I]^{-1}) \; (X^\top X + \lambda P)^{-1}(X^\top X) \\ &=\; (X^\top X + \lambda P) \; [(I - \nu(X^\top X + \lambda P)^{-1} X^\top X)^{-k} - I]^{-1} \; (X^\top X + \lambda P)^{-1}(X^\top X)\end{aligned}$$

Finally, we use the notation $S_\lambda := (X^\top X + \lambda P)^{-1} X^\top X$ to obtain the desired result

$$\Gamma_k \;=\; (X^\top X) \; S_\lambda^{-1} \; [(I - \nu S_\lambda)^{-k} - I]^{-1} \; S_\lambda.$$

□

B.4 Proof of Corollary 1

Proof. The connection of L_2-Boosting of linear models with joint updates to the solution paths of ridge regression under certain assumptions on the design matrix X as well as the differences between the two in general, can be seen as a direct application of Theorem 1. The same holds true for boosting linear models with ridge penalty (ridge boosting). To see this, we can fist note that Theorem 1 applies to linear model boosting and ridge boosting when choosing $\lambda = 0$ and $P = I$, respectively. In order to link the solution paths of ridge regression to the paths of boosting, we need to link the explicit regularization term Γ_k from (8) in Theorem 1, that is induced by the implicit minimization of boosting, to the penalization parameter $\tilde{\lambda}$ of ridge regression. We use different notations for the regularization terms in order to differentiate the explicit penalty in ridge regression from the penalty parameter λ used in ridge boosting. We first derive the connection for linear model boosting and subsequently for ridge boosting.

L_2-Boosting of linear models with joint updates For $\lambda = 0$ the quadratic penalty term Γ_k in (8) simplifies:

$$\Gamma_k \;=\; (X^\top X) \; [(I - \nu I)^{-k} - I]^{-1} = \; (X^\top X) \; \frac{(1-\nu)^k}{1 - (1-\nu)^k}.$$

Clearly, Γ_k can now match the scalar penalty parameter of ridge regression $\tilde{\lambda}$, *if and only if* $X^\top X = \sigma_X^2 I$ for an arbitrary $\sigma^2 > 0$ ($\Gamma_k = \tilde{\lambda}(k) \cdot I \Leftrightarrow X^\top X = \sigma_X^2 I$). With this condition on the design matrix, we get:

$$\Gamma_k = I \cdot \sigma_X^2 \frac{(1-\nu)^k}{1-(1-\nu)^k} \quad \Leftrightarrow \quad \tilde{\lambda}(k) := \sigma_X^2 \frac{(1-\nu)^k}{1-(1-\nu)^k}.$$

Thus, with the condition on the design matrix, the parameters of L_2-Boosting of linear models with joint updates at iteration $k \in \mathbb{N}_0$ correspond to the solution of ridge regression with penalty parameter $\tilde{\lambda}(k)$.

Ridge boosting with joint updates For $P = I$ and the penalty parameter $\lambda > 0$ of the ridge base learner, the quadratic penalty term Γ_k in (8) simplifies to:

$$\Gamma_k \;=\; (X^\top X + \lambda I)\,[(I - \nu(X^\top X + \lambda I)^{-1} X^\top X)^{-k} - I]^{-1} (X^\top X + \lambda I)^{-1} X^\top X.$$

Again, Γ_k can match the scalar penalty parameter of ridge regression $\tilde{\lambda}_{RB}$, *if and only if* $X^\top X = \sigma_X^2 I$ for an arbitrary $\sigma^2 > 0$ ($\Gamma_k = \tilde{\lambda}_{RB}(k) \cdot I \Leftrightarrow X^\top X = \sigma_X^2 I$). With this condition on the design matrix, we get:

$$\Gamma_k = I \cdot \sigma_X^2 \left[\left(1 - \nu \frac{\sigma_X^2}{\sigma_X^2 + \lambda} \right)^{-k} - 1 \right]^{-1} \quad \Leftrightarrow \quad \tilde{\lambda}_{RB}(k) := \sigma_X^2 \left[\left(1 - \nu \frac{\sigma_X^2}{\sigma_X^2 + \lambda} \right)^{-k} - 1 \right]^{-1}.$$

Similar as before, the parameters of ridge boosting with joint updates at iteration $k \in \mathbb{N}_0$ correspond to the solution of ridge regression with penalty parameter $\tilde{\lambda}_{RB}(k)$ only under the condition of isotropic features ($X^\top X = \sigma_X^2 I$). Otherwise, the implicit shrinkage of boosting for the two considered model classes cannot be matched to ridge regression in each step k of the procedure. Lastly, we can notice that the two regularization parameters $\tilde{\lambda}(k)$ and $\tilde{\lambda}_{RB}(k)$ that characterize the implicit shrinkage of boosting explicitly, decrease in the boosting steps k and converge to zero in the limit, in line with the results in (5) and Proposition 1. □

B.5 Derivation of Proposition 2

B.5.1 Equivalence to GBCD with GSQ update scheme

Derivation. The GSQ rule in block-wise L_2-Boosting for linear models can be recovered by examining the greedy selection of blocks at iteration k (given the current residuals $u^{[k]}$) in the boosting method:

$$\begin{aligned}
\hat{b}_k &= \arg\min_{b \in \mathcal{B}} \left\| u^{[k]} - X_b \hat{\beta}_b \right\|^2 \\
&= \arg\min_{b \in \mathcal{B}} \quad - {u^{[k]}}^\top X_b \left(X_b^\top X_b \right)^{-1} X_b^\top u^{[k]} \\
&= \arg\max_{b \in \mathcal{B}} \quad \sqrt{(\nabla_b \ell(\beta^{[k]}))^\top \left(H_b \right)^{-1} \nabla_b \ell(\beta^{[k]})} \\
&= \arg\max_{b \in \mathcal{B}} \quad \| \nabla_b \ell(\beta^{[k]}) \|_{H_b^{-1}} \quad \text{(GSQ rule)}.
\end{aligned} \tag{31}$$

Here, X_b corresponds to the b-th block of X, i.e., $b \in \mathcal{P}(\{1, \ldots, p\})$ with power set $\mathcal{P}$ with $\cup_{b \in \mathcal{B}} b = \{1, \ldots, p\}$ and $b_1 \cap b_2 = \emptyset \, \forall b_1, b_2 \in \mathcal{B}, b_1 \neq b_2$. After plugging in the OLS estimator for $\hat{\beta}_b$, we obtain the result by using the gradient and Hessian of the LS problem along the block b. Apart from recovering the GSQ rule (25), we further notice that the update step for this L_2-Boosting variant is

$$\beta^{[k+1]} = \beta^{[k]} + \nu U_{\hat{b}_k} \hat{\beta}_{\hat{b}_k}, \tag{32}$$

with step size $\nu \in (0, 1]$, $U_{\hat{b}_k}$ as defined in Appendix A.2.1, and $\hat{\beta}_{\hat{b}_k}$ as

$$\hat{\beta}_{\hat{b}_k} = \left(X_{\hat{b}_k}^\top X_{\hat{b}_k} \right)^{-1} X_{\hat{b}_k}^\top u^{[k]} = -(H_{\hat{b}_k})^{-1} \nabla_{\hat{b}_k} \ell \left(\beta^{[k]} \right). \tag{33}$$

Thus, the update is identical to the GSQ update step as defined in Appendix A.2.2 used for the GBCD routine as defined in Appendix A.2.1. Therefore, we have established the equivalence of block-wise L_2-Boosting and GBCD with GSQ-type selection and updates. □

B.5.2 Equivalence to GSL update scheme

Derivation. Similarly, we can investigate block-wise L_2-Boosting for the case with only a single predictor per block. The greedy selection of components in L_2-Boosting at iteration k (given the current residuals $u^{[k]}$) is

$$\begin{aligned}\hat{j}_k &= \arg\min_{1\leq j\leq d} \left\| u^{[k]} - X_j \hat{\beta}_j \right\|^2 = \arg\min_{1\leq j\leq d} - \frac{\left\| X_j^\top u^{[k]} \right\|^2}{X_j^\top X_j} \\ &= \arg\max_{1\leq j\leq d} \quad \frac{\left\| \nabla_j \ell(\beta^{[k]}) \right\|^2}{L_j} \quad \text{(GSL rule)}.\end{aligned}$$

Here, X_j corresponds to the column of X belonging to the single predictor with index j. As for the block-wise selection, we plug in the OLS estimator for $\hat{\beta}_b$ that minimizes the OLS problem for each predictor. Using the coordinate-wise gradient and Hessian, we can recover the GSL selection rule (23). As before, we can examine the update step for this L_2-Boosting variant, which is

$$\beta^{[k+1]} = \beta^{[k]} + \nu e_{\hat{j}_k} \hat{\beta}_{\hat{j}_k}, \tag{34}$$

with step size $\nu \in (0, 1]$, $e_{\hat{j}_k}$ as defined in Appendix A.2.1, and $\hat{\beta}_{\hat{j}_k}$ as

$$\hat{\beta}_{\hat{j}_k} = \frac{X_{\hat{j}_k}^\top u^{[k]}}{X_{\hat{j}_k}^\top X_{\hat{j}_k}} = -\frac{\nabla_{\hat{j}_k} \ell(\beta^{[k]})}{L_{\hat{j}_k}}. \tag{35}$$

This, again can be seen to be identical to the GSL update step as defined in (24). Therefore, we have also established the equivalence of component-wise L_2-Boosting for linear models and GCD with GSL-type selection and updates. □

B.5.3 Derivation for Remark 4

Derivation. For L_2-Boosting with penalized linear models, we can also write the selection and update steps in terms of the gradient and Hessian. A key insight is that compared to G(B)CD applied to penalized problems, these BAM variants neglect the penalization accumulated in previous boosting iterations. We demonstrate this along the example of BAMs with block-wise regression spline fitting, which uses the penalized loss in (6). In the derivation below each X_b corresponds to the columns of a single regression spline, e.g., a P-spline. In this case, the greedy selection at step k can be written as

$$\begin{aligned}\hat{b}_k &= \arg\min_{b\in\mathcal{B}} \left\| u^{[k]} - X_b \hat{\beta}_b \right\|^2 + \lambda \hat{\beta}_b^\top P_b \hat{\beta}_b \\ &= \arg\min_{b\in\mathcal{B}} - {u^{[k]}}^\top X_b \left(X_b^\top X_b + \lambda P_b \right)^{-1} X_b^\top u^{[k]} \\ &= \arg\max_{b\in\mathcal{B}} \|\nabla_b \ell_{LS}(\beta^{[k]})\|_{(H_b^{PLS})^{-1}} \quad \text{(GSQ rule)}.\end{aligned} \tag{36}$$

In the second line in (36), we plugged in the P-spline estimator for $\hat{\beta}_b$ and subsequently simplified terms. Importantly, in the fourth line, we recover the gradient of the unpenalized LS problem $\nabla_b \ell_{LS}$ as the algorithm does not account for penalization from previous steps. However, as the $H_b^{LS} \preceq H_b^{PLS}$, block-wise boosting with penalized L_2 loss can still be interpreted as GBCD with GSQ rule applied to the unpenalized L_2 loss. The same can be observed for the update step which is

$$\beta^{[k+1]} = \beta^{[k]} + \nu U_{\hat{b}_k} \hat{\beta}_{\hat{b}_k} \tag{37}$$

with step size $\nu \in (0, 1]$, $U_{\hat{b}_k}$ as defined in Appendix A.2.1, and $\hat{\beta}_{\hat{b}_k}$ as

$$\hat{\beta}_{\hat{b}_k} = - \left(H_{\hat{b}_k}^{PLS} \right)^{-1} \nabla_{\hat{b}_k} \ell^{LS}(\beta^k). \tag{38}$$

Note, that (36) and (38) scale the gradient with H_b^{PLS}, which is a block from the Hessian of the penalized problem. Hence, (36) is not equivalent to the update step that we have seen for L_2-Boosting (31). Further, in case GBCD with GSQ rule is applied to the same penalized problem in Eq. (6), the selection and updates steps would be identical to (36) and (38), with the important distinction that the gradient of the penalized problem is used. Hence, the two procedures would not be equivalent in this case. □

B.6 Proof of Theorem 2 and Corollary 2

Proof. In the following, we adopt some of the techniques used by Nutini et al. (2022), who showed convergence for GBCD with GSQ rule but without deriving an explicit convergence rate. First, we use the fact that the function $\ell(\beta)$ is assumed to be L-smooth in the parameters $\beta \in \mathbb{R}^p$. This guarantees that there exists a matrix H_b such that $\nabla^2_{bb}\ell(\beta) \preceq H_b \ \forall \beta \in \mathbb{R}^p, \ \forall b \in \mathcal{B}$. Equivalently, this also guarantees that there exists $\nu \in (0, 1]$ s.t. $\nabla^2_{bb}\ell(\beta) \preceq \frac{1}{\nu} X_b^\top X_b \ \forall \beta \in \mathbb{R}^p, \ \forall b \in \mathcal{B}$. In the following, the matrix H_b may therefore denote $\frac{1}{\nu} X_b^\top X_b$ or $\frac{1}{\nu}(X_b^\top X_b + \lambda P_b)$ for unpenalized and penalized base learners, respectively. The latter also provides a valid upper bound of the Hessian in case the former does, given that $0 \preceq \lambda P_b$. Using the matrix formulation of the L-smoothness condition (19), we can derive the following upper bound:

$$\begin{aligned} \ell(\beta^{[k+1]}) &\leq \ell(\beta^{[k]}) + \langle \nabla_{b_k} \ell(\beta^{[k]}), \beta^{[k+1]} - \beta^{[k]} \rangle + \frac{1}{2} \|\beta^{[k+1]} - \beta^{[k]}\|^2_{H_{b_k}} \\ &= \ell(\beta^{[k]}) - \nu(1 - \frac{\nu}{2}) \|\nabla_{b_k} \ell(\beta^{[k]})\|^2_{H_{b_k}{}^{-1}}, \end{aligned} \tag{39}$$

where the first inequality follows from the L-smoothness in (19) and for the second equality we used the GSQ related update (21) for $\beta^{[k+1]}$. We can rewrite the upper bound in (39) by using the norm

$$\|\vartheta\|_{\mathcal{B}} = \max_{b \in \mathcal{B}} \|\vartheta_b\|_{H_b{}^{-1}} \tag{40}$$

for some $\vartheta \in \mathbb{R}^p, \vartheta_b \in \mathbb{R}^{|b|}$, and some p.d. $H_b \in \mathbb{R}^{|b| \times |b|}$. Using this norm, we have that $\|\nabla \ell(\beta^{[k]})\|_{\mathcal{B}} = \|\nabla_{b_k} \ell(\beta^{[k]})\|_{H_{b_k}{}^{-1}}$, so that we can write (39) as

$$\ell(\beta^{[k+1]}) \leq \ell(\beta^{[k]}) - \nu(1 - \frac{\nu}{2}) \|\nabla \ell(\beta^{[k]})\|^2_{\mathcal{B}}. \tag{41}$$

Next we use the fact that the function $\ell(\beta)$ is assumed to be μ-PL in the parameters $\beta \in \mathbb{R}^p$. Instead of using Definition 2 in terms of the L_2-norm, we use the previously introduced norm in (40), for which it holds $\|\vartheta\|_2^2 \leq L_{\mathcal{B}} |\mathcal{B}| \|\vartheta\|^2_{\mathcal{B}}$ with $L_{\mathcal{B}} = \max_{b \in \mathcal{B}} \lambda_{max}(H_b)$. This inequality can be verified by

$$\|\vartheta\|_2^2 = \sum_b^{|\mathcal{B}|} \|\vartheta_b\|_2^2 \leq \sum_b^{|\mathcal{B}|} \lambda_{max}(H_b) \ \vartheta_b^\top H_b{}^{-1} \vartheta_b \leq L_{\mathcal{B}} \sum_b^{|\mathcal{B}|} \vartheta_b^\top H_b{}^{-1} \vartheta_b \leq L_{\mathcal{B}} \ |\mathcal{B}| \|\vartheta\|^2_{\mathcal{B}}. \tag{42}$$

Thus, we obtain the PL-inequality

$$\frac{1}{2} \|\nabla \ell(\beta)\|^2_{\mathcal{B}} \ \geq \ \frac{\mu}{L_{\mathcal{B}} |\mathcal{B}|} \ (\ell(\beta) - \ell^*), \tag{43}$$

where μ corresponds to the PL parameter of Definition 2 with L_2-norm. Lastly, by connecting the inequalities (41) and (43), and iterating over k iterations, we get our final convergence result

$$\ell(\beta^{[k]}) - \ell^* \leq \left(1 - \nu(2 - \nu) \frac{\mu}{L_{\mathcal{B}} |\mathcal{B}|}\right)^k \left(\ell(\beta^{[0]}) - \ell^*\right). \tag{44}$$

To show Corollary 2, we first note that for quadratic functions in particular, it holds that $\mu = \lambda_{pmin}(Q)$. Further we can use that $L_{\mathcal{B}} \leq \lambda_{max}(Q)$ to get

$$\ell(\beta^{[k]}) - \ell^* \leq \left(1 - \nu(2 - \nu) \frac{1}{|\mathcal{B}|} \frac{\lambda_{pmin}(Q)}{\lambda_{max}(Q)}\right)^k \left(\ell(\beta^{[0]}) - \ell^*\right). \tag{45}$$

□

B.7 Proof of Corollary 3

Proof. Corollary 3 essentially follows from Theorem 2. This, is because the proof of Theorem 2 in Appendix B.6 can be conducted analogously for GBCD with GSQ. The only difference compared to block-wise boosting is that GBCD with GSQ is using gradients of the penalized problem in the selection and update steps (see Appendix B.5.3). Therefore, under the assumptions of Corollary 3, i.e., that the regression spline is μ-PL and L-smooth in its parameters, Theorem 2 implies convergence to a solution of the regression spline with minimal loss. The two conditions of μ-PL and L-smoothness follow from the fact that regression spline problems can be written in quadratic form (cf. Appendix A.1). Note, in case the regression spline problem is not only μ-PL but strongly convex, GBCD with GSQ converges to the unique optimal solution. □

B.8 Derivation of Proposition 3

Derivation. Assume $d := |\mathcal{B}|$ cubic smoothing splines, each with operator $S_m : \mathbb{R}^n \to \mathbb{R}^n, (m \in [d])$, that maps the current residuals to fitted values. Each S_m has n eigenvalues $(\lambda_m)_i, i \in [n]$, with $0 < (\lambda_m)_i \leq 1 \, \forall i \in [n]$. Define the boosting operator in step k as $T^{[k]} := (I - S_{m_k})$, where m_k denotes the index of the selected cubic smoothing spline learner at step k. The boosting operator $T^{[k]} : \mathbb{R}^n \to \mathbb{R}^n$ with $k \in \mathbb{N}_0$ has n eigenvalues $(\tilde{\lambda}_m)_i$, $i \in [n]$, with $0 \leqslant (\tilde{\lambda}_m)_i < 1$ (Bühlmann and Yu, 2003). Define $\tilde{\lambda}_{max} := \max_{m \in [d]}(\max_{i \in [n]}((\tilde{\lambda}_m)_i))$, to be the largest eigenvalue of all d boosting operators. Further, the fitted values f^* of the saturated model match the observed values y exactly, i.e., $f^* = y$. Then it holds

$$
\begin{aligned}
\left\| y - f^{[k]} \right\| = \left\| f^* - f^{[k]} \right\| &= \| T^{[k]} \cdot \ldots \cdot T^{[1]} y \| \\
&\leq \| T^{[k]} \cdot \ldots \cdot T^{[1]} \|^* \| \, y \| \qquad \text{by Property 1, 2.} \\
&\leq \left\| T^{[k]} \right\|^* \cdot \ldots \cdot \left\| T^{[1]} \right\|^* \| y \| \quad \text{by Property 1, 3.} \\
&\leq \| y \| \underbrace{\underbrace{(\tilde{\lambda}_{max})^k}_{\in [0,1)}}_{\overset{k \to \infty}{\to} 0}
\end{aligned}
$$

Thereby, it follows that $f^{[k]} \to f^*$ for $k \to \infty$. □

B.9 Derivation of Proposition 4 and further remarks

Derivation. As described in Algorithm 1, in each iteration BAMs separately fit each base learner against the negative functional derivative of the loss function $\ell(\cdot)$ at the current function estimate $f^{[k]}$. In order to relate boosting for exponential families to GBCD, we establish the link between the negative functional derivative $\{\widetilde{y}_i\}_{i=1}^n$ and the gradient of the loss function with respect to the parameter β. Considering the negative log-likelihood instead of the log-likelihood in (13) and using the linearity of the base learners in their parameters ($f = X\beta$), one can do so by

$$
-\frac{\partial}{\partial \beta} \ell(\beta) = \frac{\partial}{\partial \beta} f(\beta) \; \widetilde{y} = X^\top \widetilde{y}. \tag{46}
$$

The block-wise LS base procedure of BAMs for exponential family loss can now be written as GBCD with GSQ. We derive this for the block-wise procedure. The component-wise procedure then follows as a special case. First, the block selection is done via

$$
\begin{aligned}
\hat{b}_k &= \underset{b \in \mathcal{B}}{\arg\min} \left\| \widetilde{y}^{[k]} - X_b \hat{\beta}_b \right\|^2 \\
&= \underset{b \in \mathcal{B}}{\arg\min} \quad - \widetilde{y}^{[k]^\top} X_b \left(X_b^\top X_b \right)^{-1} X_b^\top \widetilde{y}^{[k]} \\
&= \underset{b \in \mathcal{B}}{\arg\max} \quad \sqrt{(\nabla_b \ell(\beta^{[k]}))^\top \left(X_b^\top X_b \right)^{-1} \nabla_b \ell(\beta^{[k]})} \\
&= \underset{b \in \mathcal{B}}{\arg\max} \quad \| \nabla_b \ell(\beta^{[k]}) \|_{(X_b^\top X_b)^{-1}} \quad \text{(GSQ rule),}
\end{aligned} \tag{47}
$$

which is analogous to the derivation in (31) and corresponds to the GSQ rule. Similarly, the update is

$$
\hat{\beta}^{[k+1]} = \hat{\beta}^{[k]} + \nu U_{\hat{b}_k} \hat{\beta}_{\hat{b}_k} \tag{48}
$$

with

$$
\hat{\beta}_{\hat{b}_k} = \left(X_{\hat{b}_k}^\top X_{\hat{b}_k} \right)^{-1} X_{\hat{b}_k}^\top \widetilde{y}^{[k]} = - \left(X_{\hat{b}_k}^\top X_{\hat{b}_k} \right)^{-1} \nabla_{\hat{b}_k} \ell(\beta^{[k]}). \tag{49}
$$

Therefore, as long as $\frac{1}{\nu} X_b^\top X_b$ provides an upper bound to the respective blocks of the Hessian in (28) for all $b \in \mathcal{B}$, the selection and updates correspond to the GSQ rule and we get linear convergence of BAMs with block-wise updates for exponential family loss due to Theorem 2. Note that the derivation essentially builds on the linearity of the base learners in their parameters. Thus, the same could be done for any other model class that fulfills this condition. □

B.10 Remark on other Loss Functions

Robust loss functions such as the L_1 and Huber loss are frequently used alternatives to the L_2 loss for gradient boosting methods (Bühlmann and Hothorn, 2007). While the idea of greedy selection and updates would still be applicable in this case, we cannot obtain convergence results as stated in Theorem 2. The reason for this is that the gradient of the L_1-loss does not decrease in size as we reach the minimizer of the problem, which is why it cannot be μ-PL. The Huber loss is μ-PL only in a δ-neighborhood of the minimum. Thus, the convergence rate for the Huber loss cannot be globally linear.

Exponential family loss functions with canonical link such as the Binomial and Poisson loss were shown to be strictly convex in Appendix A.4 and thus μ-PL over any compact set. While the Binomial loss was also shown to be L-smooth ($L = 1/4$), the Poisson loss is non-globally Lipschitz continuous. However, the Poisson loss is still L-smooth over any compact set (but potentially with a large Lipschitz constant $L>0$).

Lastly, we consider the loss of the Cox Proportional Hazards (Cox PH) model that is frequently encountered in survival analysis. For the Cox PH model, the log partial likelihood is given by

$$\ell(\beta) = \sum_{i:C_i=1} \left(x_i^\top \beta - \log \sum_{j \in \tilde{R}(t_i)} \exp(x_j^\top \beta) \right)$$

and the gradient is given by

$$\nabla \ell(\beta) = \sum_{i:C_i=1} \left(x_i - \frac{\sum_{j \in \tilde{R}(t_i)} x_j \exp(x_j^\top \beta)}{\sum_{j \in \tilde{R}(t_i)} \exp(x_j^\top \beta)} \right) := \sum_{i:C_i=1} \left(x_i - \sum_{j \in \tilde{R}(t_i)} x_j \omega_j(\beta) \right),$$

where t_i the observed survival time, C_i is a event indicator ($C_i = 1$ if the event occurred and $C_i = 0$ if the data is censored), and $\tilde{R}(t_i)$ the risk set at time t_i (the set of individuals still at risk of the event at time t_i). First, we consider the μ-PL condition. Importantly, while the first term in the log partial likelihood is simply linear in β the second is a log-sum-exp function of the parameters β. Apart from pathological cases that can be ruled out in almost all modeling setups, such functions are known to be strictly convex. Thus, similar to the case of exponential family losses, the μ-PL condition is fulfilled for the Cox PH loss over any compact set. To show L-smoothness of the Cox PH loss, we consider the gradient of the loss, which only depends on β via the term $\omega_j(\beta)$. As the latter can be identified as a softmax function, which is known to be Lipschitz continuous, the L-smoothness of the Cox PH loss follows. Given the two fulfilled conditions, convergence guarantees from Theorem 2 apply. We demonstrate this using numerical experiments in Appendix D.3.

B.11 Proof of Proposition 5

Proof. We now prove that the distributional Gaussian regression is biconvex in (β, ξ). Given the negative log-likelihood $-\mathcal{L}$ of a Gaussian distribution with the mean $\psi_i = x_i^\top \beta$ and standard deviation $\sigma_i = \exp(z_i^\top \xi)$, and observed values y_i for $i = 1, \ldots, n$, we first derive the gradient and Hessian with respect to β and ξ. Note that x_i and z_i could contain the same or different features, and are thus not necessarily the same. The probability density function (PDF) of a Gaussian distribution is given by:

$$p_{\mathcal{N}(\psi_i, \sigma_i)}(y_i) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(y_i - \psi_i)^2}{2\sigma_i^2} \right).$$

The log-likelihood for n observations is:

$$\mathcal{L} = \sum_{i=1}^{n} \log p_{\mathcal{N}(\psi_i, \sigma_i)}(y_i).$$

Substituting the Gaussian PDF, we get:

$$\mathcal{L} = \sum_{i=1}^{n} \left[-\frac{1}{2} \log(2\pi) - \frac{1}{2} \log(\sigma_i^2) - \frac{(y_i - \psi_i)^2}{2\sigma_i^2} \right].$$

Simplifying, and noting that $\sigma_i = \exp(z_i^\top \xi)$ implies $\log(\sigma_i^2) = 2z_i^\top \xi$, we get:

$$-\mathcal{L} = \frac{n}{2}\log(2\pi) + \sum_{i=1}^{n}\left(z_i^\top \xi + \frac{(y_i - x_i^\top \beta)^2}{2\exp(2z_i^\top \xi)}\right).$$

Let $r_i = y_i - x_i^\top \beta$ and $\sigma_i = \exp(z_i^\top \xi)$. The derivative with respect to β is

$$\frac{\partial(-\mathcal{L})}{\partial \beta} = \sum_{i=1}^{n}\frac{\partial}{\partial \beta}\left(\frac{r_i^2}{2\sigma_i^2}\right) = \sum_{i=1}^{n}\left(\frac{1}{2\sigma_i^2}\right)(-2x_i r_i) = -\sum_{i=1}^{n}\frac{x_i r_i}{\sigma_i^2}.$$

The derivative with respect to ξ is

$$\frac{\partial(-\mathcal{L})}{\partial \xi} = \sum_{i=1}^{n}\left(\frac{\partial z_i^\top \xi}{\partial \xi} + \frac{\partial}{\partial \xi}\left(\frac{r_i^2}{2\sigma_i^2}\right)\right).$$

The first term is $\frac{\partial z_i^\top \xi}{\partial \xi} = z_i$, whereas the second term evaluates to

$$\frac{\partial}{\partial \xi}\left(\frac{r_i^2}{2\sigma_i^2}\right) = \frac{\partial}{\partial \xi}\left(\frac{r_i^2}{2\exp(2z_i^\top \xi)}\right) = -\frac{r_i^2}{2\exp(2z_i^\top \xi)} \cdot \frac{\partial \exp(2z_i^\top \xi)}{\partial \xi}.$$

As

$$\frac{\partial \exp(2z_i^\top \xi)}{\partial \xi} = 2z_i \exp(2z_i^\top \xi).$$

we have

$$\frac{\partial}{\partial \xi}\left(\frac{r_i^2}{2\sigma_i^2}\right) = -2z_i \cdot \frac{r_i^2}{2\sigma_i^2} = -z_i\frac{r_i^2}{\sigma_i^2}.$$

So, combining both terms:

$$\frac{\partial(-\mathcal{L})}{\partial \xi} = \sum_{i=1}^{n}\left(z_i - z_i\frac{r_i^2}{\sigma_i^2}\right) = \sum_{i=1}^{n} z_i\left(1 - \frac{r_i^2}{\sigma_i^2}\right).$$

For the Hessian we need to compute the second derivatives with respect to β and ξ. We start with β:

$$\frac{\partial^2(-\mathcal{L})}{\partial \beta \partial \beta^\top} = \sum_{i=1}^{n}\frac{\partial}{\partial \beta^\top}\left(-\frac{x_i r_i}{\sigma_i^2}\right) = \sum_{i=1}^{n}\frac{x_i x_i^\top}{\sigma_i^2}.$$

For ξ we have:

$$\frac{\partial^2(-\mathcal{L})}{\partial \xi \partial \xi^\top} = \sum_{i=1}^{n}\frac{\partial}{\partial \xi^\top}\left(z_i - z_i\frac{r_i^2}{\sigma_i^2}\right).$$

$$\frac{\partial}{\partial \xi^\top}\left(z_i - z_i\frac{r_i^2}{\sigma_i^2}\right) = \frac{\partial z_i}{\partial \xi^\top} - \frac{\partial z_i}{\partial \xi^\top}\frac{r_i^2}{\sigma_i^2} - z_i\frac{\partial}{\partial \xi^\top}\left(\frac{r_i^2}{\sigma_i^2}\right).$$

Since $\frac{\partial z_i}{\partial \xi^\top} = 0$ and noting that:

$$\frac{\partial}{\partial \xi^\top}\left(\frac{r_i^2}{\sigma_i^2}\right) = -2\frac{r_i^2}{\sigma_i^2}z_i^\top,$$

we get

$$\frac{\partial^2(-\mathcal{L})}{\partial \xi \partial \xi^\top} = \sum_{i=1}^{n}\left(2\frac{r_i^2}{\sigma_i^2}z_i z_i^\top\right).$$

For the mixed second derivatives with respect to β and ξ, we have

$$\frac{\partial^2(-\mathcal{L})}{\partial \beta \partial \xi^\top} = \sum_{i=1}^{n}\frac{\partial}{\partial \xi^\top}\left(-\frac{x_i r_i}{\sigma_i^2}\right),$$

where

$$\frac{\partial}{\partial \xi^\top}\left(-\frac{x_i r_i}{\sigma_i^2}\right) = -x_i r_i\frac{\partial}{\partial \xi^\top}\left(\frac{1}{\sigma_i^2}\right) = 2\frac{r_i}{\sigma_i^2} \cdot x_i z_i^\top.$$

So the mixed Hessian component is:

$$\frac{\partial^2(-\mathcal{L})}{\partial\beta\partial\xi^\top} = 2\sum_{i=1}^{n} \frac{r_i}{\sigma_i^2} \cdot x_i z_i^\top.$$

Combining everything, the Hessian matrix thus is

$$H = \begin{bmatrix} \frac{\partial^2(-\mathcal{L})}{\partial\beta\partial\beta^\top} & \frac{\partial^2(-\mathcal{L})}{\partial\beta\partial\xi^\top} \\ \frac{\partial^2(-\mathcal{L})}{\partial\xi\partial\beta^\top} & \frac{\partial^2(-\mathcal{L})}{\partial\xi\partial\xi^\top} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n \frac{x_i x_i^\top}{\exp(2z_i^\top\xi)} & 2\sum_{i=1}^n \frac{(y_i - x_i^\top\beta)x_i z_i^\top}{\exp(2z_i^\top\xi)} \\ 2\sum_{i=1}^n \frac{(y_i - x_i^\top\beta)z_i x_i^\top}{\exp(2z_i^\top\xi)} & 2\sum_{i=1}^n \frac{(y_i - x_i^\top\beta)^2 z_i z_i^\top}{\exp(2z_i^\top\xi)} \end{bmatrix}.$$

We can directly observe that for given $\hat{\beta}$, the lower diagonal block can be written as $Z^\top\Psi Z$ with $\Psi = \text{diag}(2r_i^2/\sigma_i^2)$ and $Z = (z_1, \ldots, z_n)^\top$, i.e., a positive semi-definite matrix. Clearly, due to the term $\exp(2z_i^\top\xi)$ on the diagonal, $Z^\top\Psi Z$ cannot be bounded for all ξ. Accordingly, (14) cannot be L-smooth in the parameters ξ. The upper diagonal block of H is a positive definite matrix since it can similarly be written as $X^\top\Sigma^{-1}X$ with $\Sigma = \text{diag}(\sigma_i^2)$. In general, however, the Hessian cannot be written as

$$H = \Lambda^\top\Sigma^{-1}\Lambda$$

since with $\Lambda = (X\ \sqrt{2}\tilde{Z})$ and $\tilde{Z} = Z\text{diag}(r_i)$ the factors 2 in front of the mixed terms cannot be matched. Hence, even if the diagonal blocks are positive definite, we can construct a counterexample, e.g., $n = 1$, $x_i = z_i = \beta = \xi = 1$ and $y_i = 2$, for which the matrix is indefinite. □

C Cubic smoothing splines and P-splines

Cubic Smoothing Splines. A *cubic smoothing spline* f as considered in Section 3.2.2 (for a twice differentiable function f), minimizes

$$\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda \int \left(\frac{\partial^2 f}{\partial x^2}\right)^2 dx, \tag{50}$$

for a given $\lambda \geq 0$ and where x_i denotes the i-th row of the $n \times p$ feature matrix X. Cubic smoothing splines aim to find an optimal function that best adjusts to the least squares problem while at the same time penalizing the squared second derivative of the function to induce smoothness on the solution. The trade-off between the two is controlled by the smoothness parameter λ. Cubic smoothing splines are considered by Bühlmann and Yu (2003) for component-wise L_2-Boosting.

P-splines. Although cubic smoothing splines are particularly interesting from a theoretical point of view, the integral for penalization imposes major computational disadvantages compared to other smoothing base learners. A natural alternative is to consider a discretized version of it. As such, Eilers and Marx (1996) proposed a specific type of penalized regression splines, which uses B-splines (de Boor, 1978) as basis expansions of the predictor variables and penalizes the higher-order differences between adjacent regression parameters of the B-spline. B-splines, also called basis splines, are a collection of piece-wise polynomials that are connected at specific points, called knots. For notational convenience, we elaborate on the basis expansion of a single variable via B-splines. The extension to multiple variables is straightforward. Consider $\{B_i^l(\cdot)\}_{i=1}^{\kappa+l-1}$ B-spline basis functions of order l defined at κ equidistant knot positions. When considering a P-spline for a certain feature Z_j ($j \in [p]$), this will induce an expanded feature matrix X (Fahrmeir et al., 2013), which is given by

$$X = \begin{pmatrix} B_1^l(Z_{1,j}) & \dots & B_{\kappa+l-1}^l(Z_{1,j}) \\ \vdots & & \vdots \\ B_1^l(Z_{n,j}) & \dots & B_{\kappa+l-1}^l(Z_{n,j}) \end{pmatrix}.$$

With this expanded feature matrix, the approximation of the cubic smoothing spline by a P-spline is given by

$$\|y - X\beta\|^2 + \lambda \beta^\top \underbrace{D_2^\top D_2}_{:=P} \beta, \tag{51}$$

where β now denotes an extended parameter vector corresponding to each piecewise polynomial in the B-spline, respectively. The matrix P corresponds to the second-order difference penalty matrix. The $p-2 \times p$ matrix D_2 and $p \times p$ matrix P are defined as

$$D_2 = \begin{pmatrix} 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \end{pmatrix}, \qquad P = \begin{pmatrix} 1 & -2 & 1 & & & & \\ -2 & 5 & -4 & 1 & & & \\ 1 & -4 & 6 & -4 & 1 & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & 1 & -4 & 6 & -4 & 1 \\ & & & 1 & -4 & 5 & -2 \\ & & & & 1 & -2 & 1 \end{pmatrix}.$$

Note that the penalty matrix P is rank deficient. A key property of the P-spline modeling approach in (51) is that the dimensionality of the penalization term is greatly reduced compared to the penalty in (50), as we consider a discretized version of it (Schmid and Hothorn, 2008). This is what gives P-splines a great computational advantage over cubic smoothing splines and thus makes it particularly interesting for applications such as L_2-Boosting, where (51) needs to be solved repeatedly.

D Additional experiments, experimental details and results

D.1 Convergence of penalized boosting to unpenalized model

Boosting with Ridge Penalty In our first experiment, we perform least-squares (LS) and ridge boosting on a two-dimensional problem $\beta \in \mathbb{R}^2$ to check whether both are converging against the same solution. As depicted in Figure 7, this is in fact the case, despite the exact ridge solution (without boosting) being far away from the LS solution.

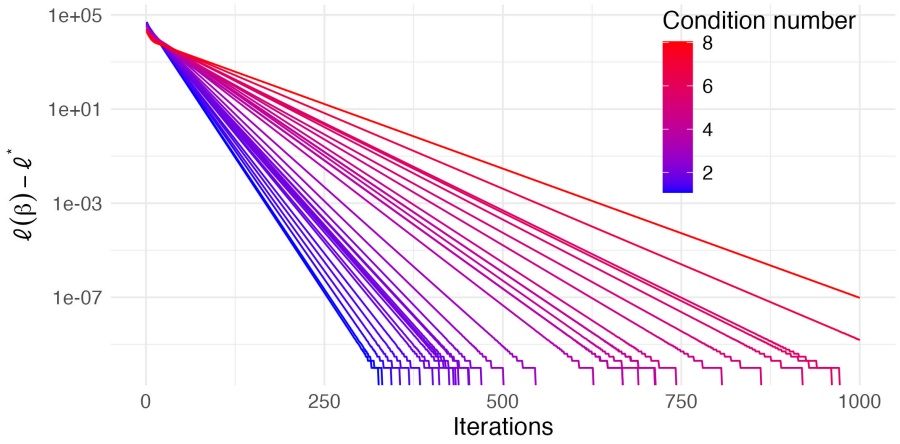

Figure 6: Linear convergence for different condition numbers (indicated by the color) for a linear model with two predictor variables. The condition numbers are induced by pairwise correlation of the predictor variables. Y axis on a logarithmic scale.

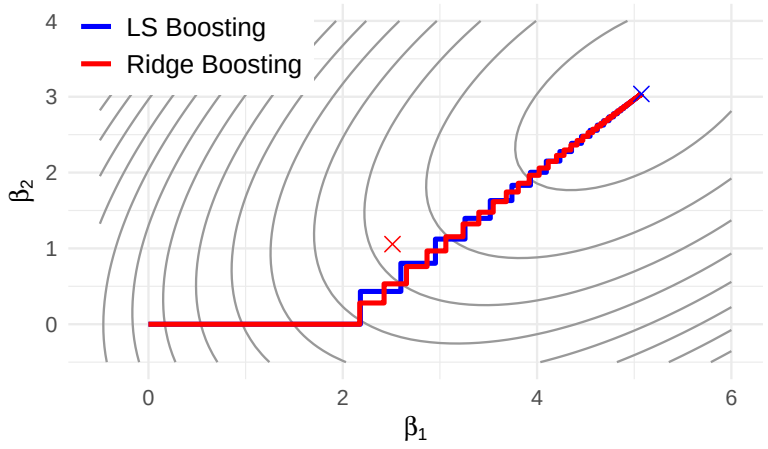

Figure 7: Parameter paths for linear model (blue line) and ridge (red line) boosting as well as their actual solutions (crosses in resp. colors). Linear model boosting is also known as Least Squares (LS) Boosting.

Boosting with P-Splines As a second example, we fit a univariate spline problem using unpenalized and penalized B-splines (B-/P-splines, resp.) using boosting. We do this both on observed data (Figure 2) and on synthetic data (Figure 8). The P-Spline in Figure 2 uses a B-Spline with 9 basis functions of degree 3 and second-order difference penalty with a penalty parameter λ of 1. The boosted version of the P-Spline are depicted in the left plot of Figure 2 at boosting iterations 1, 10, 30, 60 and 5000. Similarly, The P-Spline in Figure 8 uses the same type of B-Spline but with a penalty parameter λ of 10. The boosted version of the P-Spline in Figure 8 are shown at boosting iterations 1, 10, 30, 60 and 50000. Analogous to the result of ridge boosting, we can observe in both cases that boosting converges to the unpenalized B-spline solution despite iteratively solving a penalized LS criterion. Interestingly, due to the stage-wise fitting nature of boosting, the parameter and function path do not have to visit the penalized fit. Thus, stopping boosting iterations early does not guarantee that we can recover the penalized fit.

As a consequence, the common notion of boosting performing an implicit smoothing parameter selection might be flawed and the early stopped model might belong to a completely different function class. This, in turn, can have detrimental consequences for the statistical inference, likely providing false uncertainty statements.

The examples of ridge and P-spline boosting show that as the boosting iterations increase they converge toward the unpenalized fit (the OLS and the B-spline fit, respectively). This could raise the question of why to consider penalized base learners in boosting in the first place. First, convergence paths of penalized and unpenalized base learners differ and hence both variants explore different models. This can result in different final models in case boosting is stopped early. Second, certain base learners such as P-splines might not even work without

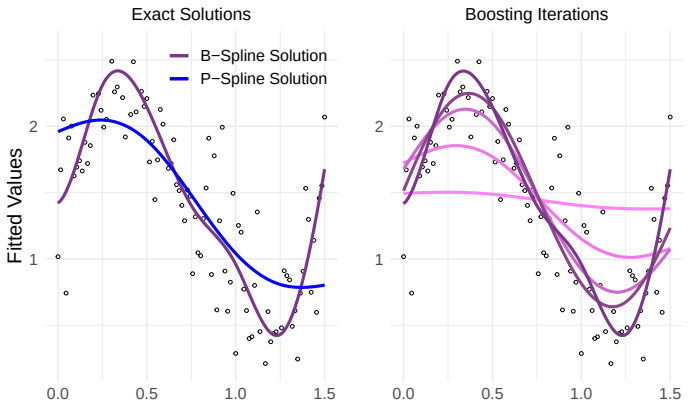

Figure 8: Functional Fitting via P-spline boosting on synthetic data. Left: Exact B-spline (purple) and P-spline solution (blue). Right: P-spline boosting iterates converging to the unpenalized (B-spline) solution (darkest color).

penalization, e.g., when $X^\top X$ is not of full rank and thus not invertible while $X^\top X + \lambda P$ is.

Models for Spatial Data For the spatial data application in Figure 4, we fit a tensor-product spline model defined by the row-wise Kronecker product of two marginal B-spline bases, each with 20 basis functions of degree 2 and first-order difference penalty. We fix an isotropic penalty for the PLS model with both smoothing parameters set to the value 5. For the boosting model, we take the knots and basis functions as defined by the PLS model and define a corresponding base-learner. The penalty strength is defined by letting the hat matrix of both P-spline dimensions have 4 degrees-of-freedom, from which the corresponding λ penalty values are computed. Note that these are not the effective degrees-of-freedom and the final penalization, but only define the base-learners a-priori flexibility.

Boosting function-on-function regression Lastly, we demonstrate the fact that BAMs are converging to the unpenalized model fit, along the more complex application of boosting of function-on-function regression. Modeling functional relationships has recently sparked renewed interest due to its connection to in-context learning of transformers (see, e.g., Chalvidal et al., 2023). We start by simulating a two-dimensional functional weight $\beta(s,t) = \sin(2|s-t|)\cos(2t)$. We then create a non-linear process $x(s), s \in [0,1]$ by spanning B-spline basis functions across the domain $[0,1]$ and drawing random basis parameters from a standard Gaussian distribution. Finally, the functional outcome is given by $y(t) = x(s)\beta(s,t) + \varepsilon(t), t \in [0,1]$, where $\varepsilon(t)$ is a white noise process.

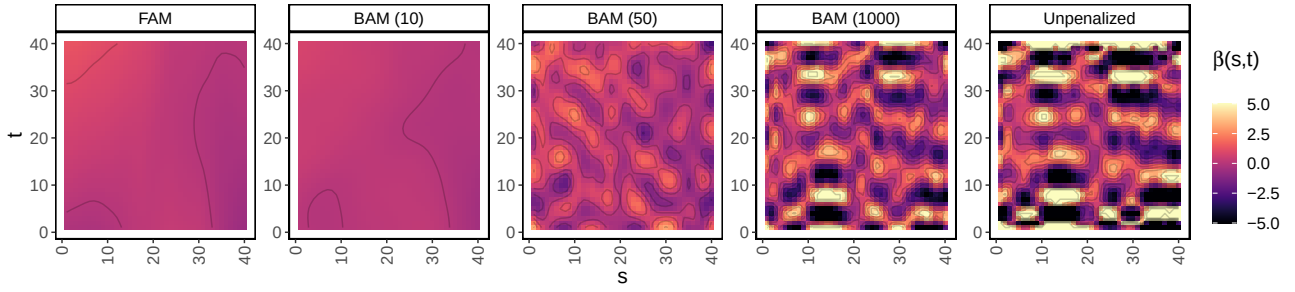

Figure 9: Exemplary estimated weight surfaces of a penalized functional regression (FAM; left) compared to BAM (second to fourth plot) with different numbers of iterations in brackets. For larger iterations, the estimated weight surface of BAM converges to the one of the unpenalized model (right plot).

After creating $n = 300$ pairs of functions, discretized to 40 time points, we fit a functional additive model (FAM; Scheipl et al., 2015), the pendant of a penalized spline regression for functional data, and compare it with BAMs for functional data (Brockhaus et al., 2020). As these models' feature matrix can be represented by a Kronecker product of evaluated basis functions and their penalization by a quadratic penalty with penalty matrix defined by a Kronecker sum, results from Section 3.1.2 apply.

Figure 9 shows the estimated weight surface $\beta(s,t)$ of FAM, boosting after 10, 50, and 1000 iterations, and the unpenalized fit. Results clearly indicate that boosting can roughly match the estimation of FAM, but when running the algorithm further will — despite explicit penalization — converge to the unpenalized model fit.

Data used for Figure 2 and Figure 4 In these two figures, we analyze a spatially granular data set of coronavirus disease spread from Wahltinez et al. (2022) with a particular focus on spatial and temporal effects. The Covid-19 infection prevalence is modeled spatially over the entire US in Fig. 4, and for San Francisco over time in Fig. 2. Both clearly demonstrate the convergence of penalized base learners to the unpenalized fit in accordance with Proposition 1.

D.2 Exponential family boosting

Simulation Details We simulate the data with $n = 100$ and $p = 2$ two features with feature effects $\beta = (3, -2)^\top$. The features are drawn from a multivariate Gaussian distribution with an empirical correlation of $\rho = 0.5$. The outcomes are finally generated by transforming the predictor $f = X\beta$ by the canonical link function h (exp-function in the Poisson case and the sigmoid function in the Binomial model) and drawing observations from the respective distributions with mean $h(f)$.

Binomial Boosting The plot for the binomial distribution depicting the convergence on the loss and parameter level for the experiment in Section 4.3 is given in Figure 10.

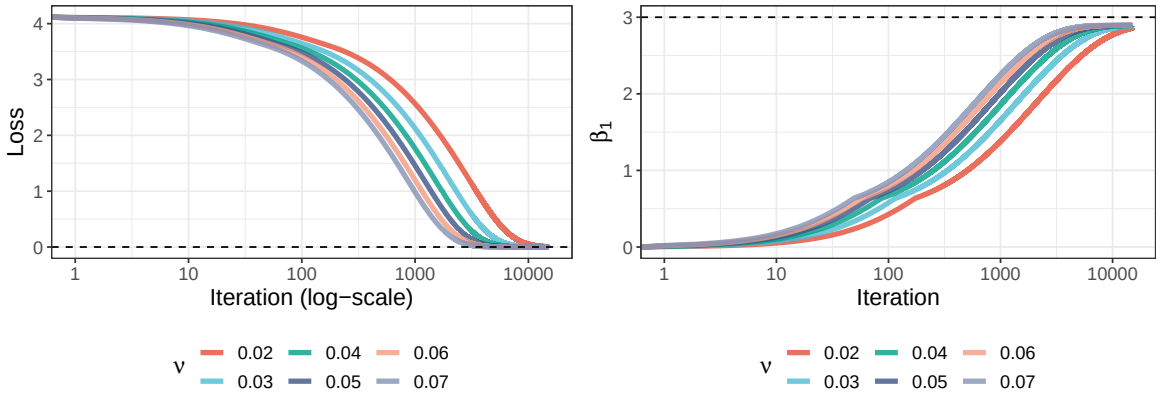

Figure 10: Left: Loss path for different learning rates (colors) showing convergence for BAMs for all rates. Right: corresponding β_1 parameters.

Poisson Boosting The plot for the Poisson distribution depicting the (non-)convergence on the loss and parameter level for the experiment in Section 4.3 is given in Figure 11.

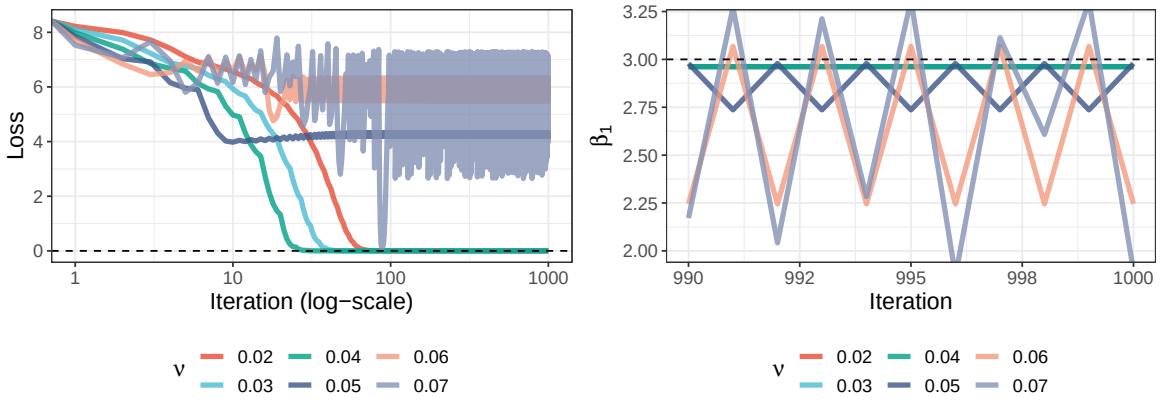

Figure 11: *Left: Loss path for different learning rates (colors) showing convergence for BAMs with the three smaller rates and non-convergence for the larger ones. Right: Last 10 updates of the corresponding β_1 parameter, depicting oscillating updates for $\nu \geq 0.05$ due to inaccurate curvature approximation.*

Poisson Boosting on observation data We also demonstrate the potential non-convergence of boosting Poisson models on observational data. Boosted poisson models were used to model the number of new Covid-19 cases in San Francisco given features such as temperature (Figure 12, top) and to model a person's health score given several features such as a cognition assessment of that person (Figure 12, bottom). While the former uses

the same data from Wahltinez et al. (2022) previous experiments, the latter uses the Health and Retirement Study (HRS) longitudinal data set (`https://hrs.isr.umich.edu`). Figure 12 depicts boosting's (non)-convergence both in terms of loss and parameters for different learning rates. This is in line with the findings of Figure 5 in the main text.

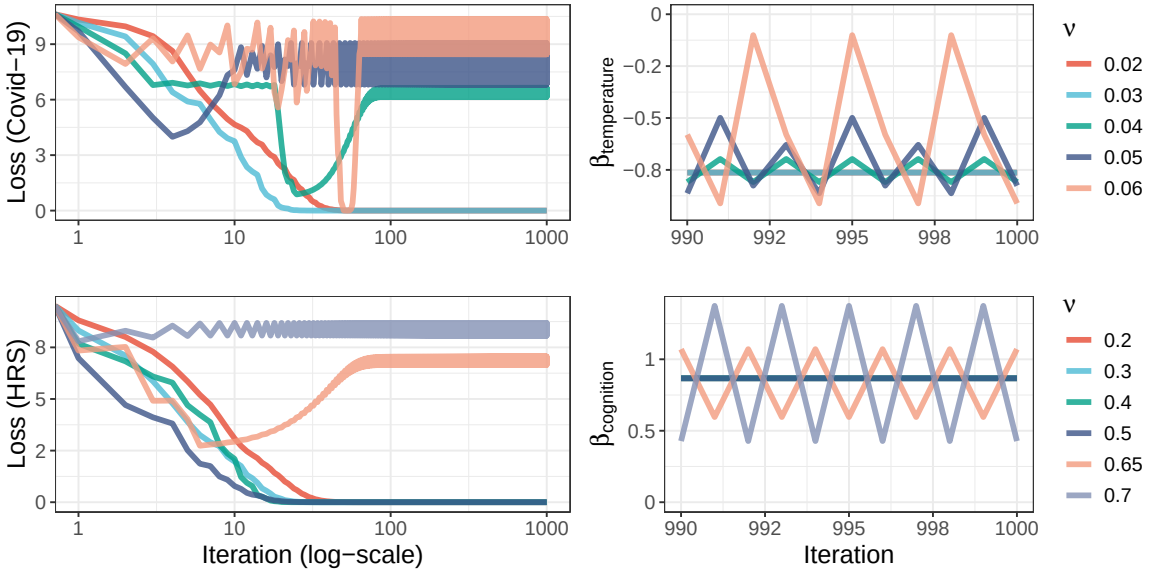

Figure 12: *Poisson Boosting with different learning rates. Top: Poisson Boosting of Covid-19 cases in San Francisco. Bottom: Poisson Boosting of health scores in a Health and Retirement Study (HRS). Non-convergence occurs for parameters of variables temperature (top right) and cognition assessment (bottom right).*

D.3 Cox Proportional Hazards Model Boosting

Cox Proportional Hazards (Cox PH) model boosting as mentioned in Section 3.2.3 extends the idea of gradient boosting to survival modeling. Models are again fitted using the corresponding boosting implementation of the `mboost` package. More specifically, we use boosted Cox PH models to model the survival data of patients with ovarian cancer and lung cancer, respectively. The data is provided in the *Ovarian* (Ovarian Cancer; Edmonson et al., 1979) and *Lung* (North Central Cancer Treatment Group Lung Cancer; Loprinzi et al., 1994) data sets. Figure 13 shows convergence with linear convergence rate (y-axis on log-scale) in the Cox PH loss[1] over the boosting iterations.

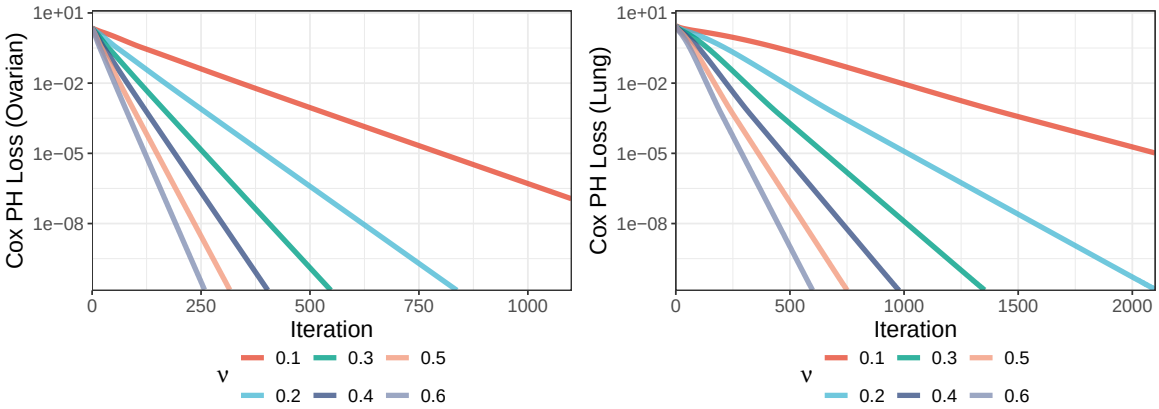

Figure 13: *Cox PH model boosting with different learning rates (y-axis on log-scale). Left: Cox PH Boosting of survival in patients with ovarian cancer (Ovarian). Right: Cox PH boosting of survival in patients with lung cancer (Lung).*

[1]For better readability, we show the difference to the optimal loss value

D.4 Distributional Boosting

Distributional boosting in the case of Gaussian distributions has been discussed in Section 3.2.4. Figure 14 and Figure 15 demonstrate the potential divergence of Gaussian distribution boosting along the example of the *Engel* (Koenker and Bassett Jr, 1982) and *CD4* (DiCiccio and Efron, 1996) data set, respectively. Both data sets show a heteroscedastic variance of the dependent variable over the covariates, making them a fitting example for distributional boosting. The *gamboostLSS* implementation (Hofner et al., 2016) is used to fit boosted models for the mean and variance parameters (characterizing the Gaussian distribution of the dependent variables) for each data set. The procedure uses greedy block/component-wise updates to build the mean and variance model and alternates updates between the two models in a cyclic fashion (as described in Section 2.3). In both Figs. 14 and 15 the loss with respect to the variance component (variance model) diverges already after a few boosting iterations for larger learning rates. As the loss of the variance model is not globally L-smooth, parameter updates can become unbounded for large learning rates and thus lead to divergence. The latter is depicted in the right plots of Fig. 14 and Fig. 15.

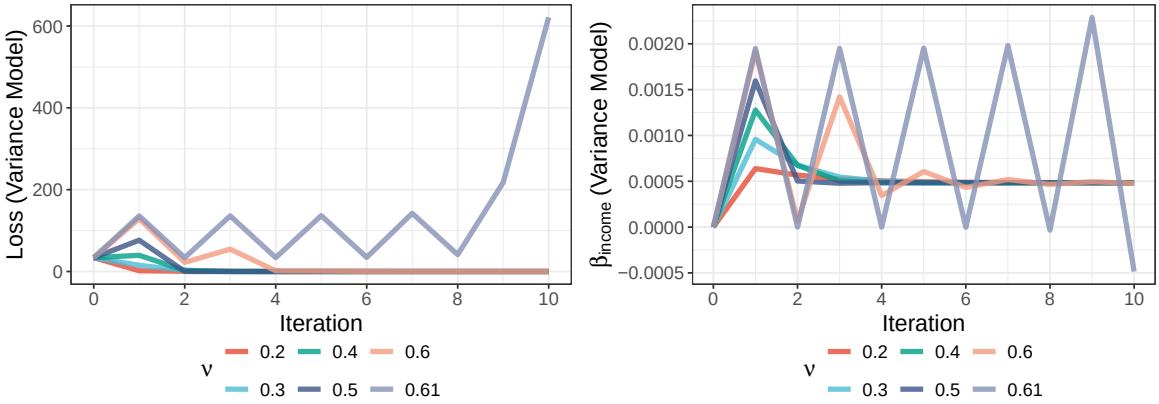

Figure 14: *Gaussian distribution boosting with different learning rates on the Engel data set. Left: Loss w.r.t. the variance component (Var. Model). Right: Estimated parameter of income variable in the variance model. Divergence occurs already after a few boosting iterations for the larger learning rates.*

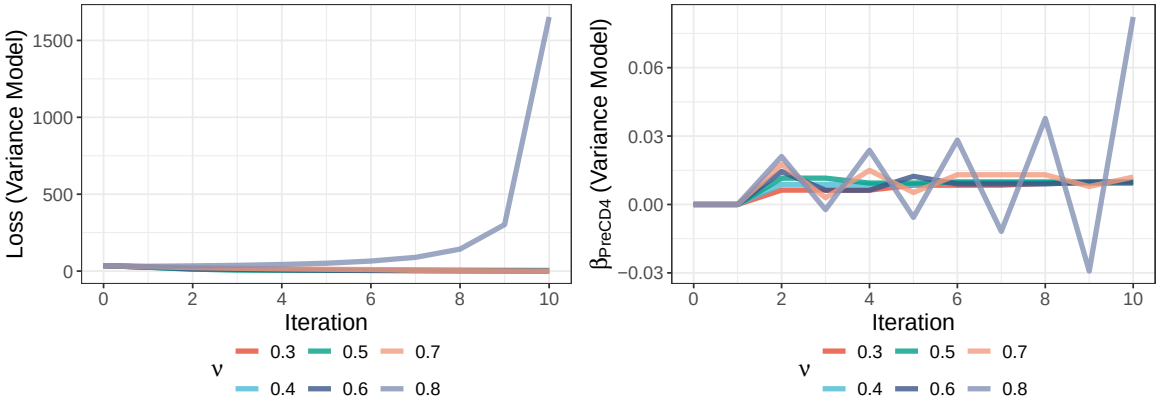

Figure 15: *Gaussian distribution boosting with different learning rates on the CD4 data set. Left: Loss w.r.t. the variance component (Var. Model). Right: Estimated parameter of the PreCD4 variable in the variance model. Divergence occurs already after a few boosting iterations for the larger learning rates.*

D.5 Path matching

In the case of joint updates, we simulate data and compare the ridge regression and BAM path. We therefore generate $n = 100$ samples with $p = 2$ predictors. The predictors $\mathbf{X}$ are generated with an empirical correlation of $\rho = 0.7$ by drawing the first predictor from a standard Gaussian distribution and defining the second predictor as a linear combination of the first and another independent normal random variable to induce correlation: $x_i = \rho x_1 + \sqrt{1-\rho^2} z_i$, where $z_i \sim \mathcal{N}(0, 1)$. The true parameter vector $\beta = (3, -2)^\top$. The response vector is then

generated by $y = X\beta + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$. We then perform ridge regression and Ordinary Least Squares (OLS) regression on the simulated data. We further run BAMs as implemented in the `mboost` package with penalized linear base learners and L_2 loss for different values of λ and track the parameter changes over iterations.

We find that it is possible to find specific ν and λ combinations such that the boosting path gets very close to the ridge regression path. This is depicted in Figure 1 (left) for $\nu = 0.1$ and maximum number of steps of 10000. However, when zooming in (right), we see that for this particular setting, the paths do not align.

E Computational environment

All computations were performed on a user PC with Intel(R) Core(TM) i7-8665U CPU @ 1.90GHz, 8 cores, and 16 GB RAM. Run times of each experiment do not exceed one hour.