The Path(ologie)s of Boosted Additive Models

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

Boosted additive models (BAMs) are a versatile toolbox to fit complex intelligible models, offering broader applicability than comparable methods due to their generality. By allowing the incorporation of different base learners as well as the use of different loss functions, BAMs cover a wide range of additive models using the idea of gradient boosting. While 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 their implicit regularization property. 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. In addition to results that provide reassuring evidence for the practical use of BAMs, we also identify a few "pathologies" in the algorithm and its convergence behavior that require caution when applying BAMs. We empirically validate our theoretical findings through several numerical experiments.

1 Intro

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. Examples include interpretable boosting methods such as [24, 34] or the so-called neural additive models (NAMs; [1, 38]). In order to maintain the interpretability of such models in the presence of a high-dimensional feature space, sparsity approaches like the Lasso [45] 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 [21], this idea was proposed as an alternative optimization and selection routine for AMs. These boosted additive models (BAMs) allow the fitting of a large plethora of different model classes, including non-linear effects through regression spline representations [43], time-varying or functional response models [5], boosted densities [27] or shapes [44], and have been used for high-dimensional settings both in the context of classical generalized additive models (GAMs; [20]), quantile regression [15] and distributional regression [29]. Yet, most papers building on the seminal work of [9] and [17] 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 [17], which allows providing numerical convergence

Preliminary work under review. Do not distribute.

results of various boosting methods [11, 28, 30, 39, 48]. Most of these 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 applicability of results to certain sub-classes of problems and does not generalize to more complex cases of BAMs. More recent investigations [16, 22, 23] exploit new theoretical insights derived from greedy coordinate descent routines, yet without deriving implications for BAMs directly. Accelerated and randomized versions were investigated in [25, 26]. Statistical properties of BAMs that incorporate the iterative fitting in boosting have been investigated in [7, 9, 43]. 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 certain BAM classes that have important implications for practical usage. In summary, our findings include 1) the parameter paths and convergence results for L_2-Boosting, 2) a formalization of the misalignment of direct optimization of Ridge and regression spline models compared to their boosted versions, 3) convergence guarantees including a linear convergence rate for block-wise boosting as well as 4) specific convergence results for various spline and exponential family boosting models. Empirical experiments in Section 4 confirm our theoretical findings, but also demonstrate failure cases in which BAMs should be treated with care.

2 Background

2.1 Notation

In this paper, we consider n pairs of observations $(y_i, x_i), i \in [n] := \{1, \ldots, n\}$, that are the realizations of some joint distribution $\mathbb{P}_{yx}$. We denote the stacked outcome as $y = (y_1, \ldots, y_n)^\top \in \mathcal{Y} \subseteq \mathbb{R}$ and the feature matrix $X = (x_1^\top, \ldots, x_n^\top)^\top \in \mathbb{R}^{n \times p}$ with rows $x_i \in \mathcal{X} \subseteq \mathbb{R}^p$. $X_j \in \mathbb{R}^{n \times p_j}, 1 \leq p_j \leq p$, refers to a subset of columns of X. 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 $f^{[k]} = f(\cdot; \beta^{[k]})$. If not stated otherwise, we consider the optimization problem

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

As we are studying BAMs, we will consider additive models for f 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 f to be a (generalized) additive model of the form

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

with Y the random variable related to the observations y_i and f_j being single predictor functions with parameter $\beta_j \in \mathbb{R}^{p_j}$ that form the additive model f. 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 $f_j + cg \in \mathcal{F}$. More specifically, the space considered in this work is 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 [13], Kriging [37], Markov random fields [40], tree-stumps or built trees with leaf weights β_j, and transfer-learning neural basis functions as in [1] where β_j are the last layer's weights.

2.3 Gradient boosting algorithm

In this work, we study Algorithm 1 as given in [7]. This algorithm is presented as a generalization of the original version from [17] and has since then been modified for the various use cases described in our introduction. Below, we use the notation $\widetilde{y}_i$ to denote the functional derivative (or Gâteaux derivative) of the loss function at the current function estimate. Note that this is only defined at the data points $i \in [n]$. The updates $g^{[k]}(\,\cdot\,;\hat{\beta})$ of the function estimate $f^{[k]}$ are parameterized by a parameter $\hat{\beta}$. This is to highlight the fact that gradient boosting can not only be seen as a method performing steps in function space but also in the parameter space.

Algorithm 1 Functional gradient boosting

```
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.,
       $$\widetilde{y}_i^{[k]} := -\frac{\partial}{\partial f}\ell(y_i, f)\Big|_{f=f^{[k]}(x_i)} \qquad i = 1, \dots, n \qquad (3)$$
   Set $k = k + 1$.
3: Fit base procedure to the negative gradient $\{\widetilde{y}_i^{[k]}\}_{i=1}^n$:
       $$\{\widetilde{y}_i^{[k]}, x_i\}_{i=1}^n \xrightarrow{\text{base procedure}} g^{[k]}(\,\cdot\,;\beta). \qquad (4)$$
4: Update $f^{[k]}(\cdot) = f^{[k-1]}(\cdot) + \nu g^{[k]}(\,\cdot\,;\hat{\beta})$ with $\nu \in (0, 1]$
5: Repeat steps 2 – 4 until convergence or until a pre-specified stopping criterion is met.
```

Apart from being compatible with different loss functions, Algorithm 1 gains its generality by incorporating a generic base procedure. The latter can include various base learners such as linear models or penalized regression splines, that can be updated jointly or in a greedy block-wise fashion, thereby resembling a wide variety of gradient boosting variants. The algorithm further replaces the line search from the original gradient boosting algorithm with updates of constant step size in step 4.

2.4 Boosting additive models

The functional gradient boosting algorithm can be adapted as a fitting and selection procedure for models described in Section 2.2. This can be done by defining the loss function as $\ell(\beta) := \ell(h^{-1}(y), f(x;\beta))$ for GAMs, or more generally, as the negative log-likelihood of a parametric distribution, where the parameter(s) of interest are modeled via $h(f(x;\beta))$. The definition of the base procedure g used in Algorithm 1, in turn, allows to change between a joint update of all components included in f and a block-wise selection of additive components.

Joint updates and block-wise selection Analyzing BAMs from a theoretical point of view is more straightforward when updating all additive terms in (2) simultaneously, i.e., $g = f$. The base procedure can, however, also be defined to greedily select the best-performing additive component f_j at each step. This corresponds to defining one base learner for every additive term $j \in [J] := \{1, \dots, J\}$ and greedily performing block-wise selection and updates. This changes step 3 of Algorithm 1 as described in the following:

```
3: a) For $j = 1, \dots, J$, find $\hat{\beta}_j = \arg\min_{\beta_j \in \mathbb{R}^{p_j}} n^{-1} \sum_{i=1}^n (\widetilde{y}_i^{[k]} - f_j(x_i;\beta_j))^2$
   b) Denote $j^* = \arg\min_{j \in [J]} n^{-1} \sum_{i=1}^n (\widetilde{y}_i^{[k]} - f_j(x_i;\hat{\beta}_j))^2$
   c) Set $g^{[k]}(\cdot;\hat{\beta}) = f_{j^*}(\cdot;\hat{\beta}_{j^*})$
```

In combination with early stopping, this can enable variable selection as not all additive components f_j are selected and thus can result in a sparse model. In general, these block-wise updates are performed by selecting blocks $b \in \mathcal{B}$, where each block b can be a subset of all base learners or even a subset of parameters within one base learner. To simplify the presentation, we assume that each block b corresponds to a set of parameters $\beta_j, j \in [J]$.

In contrast to routines like the Lasso, which requires a specific optimization routine for every new model class, the selection property of BAMs is generic and applicable for all model classes discussed in Section 2.2. Although this versatility of BAMs is beneficial in practice, it poses certain challenges when studying their properties from a theoretical perspective. In the following, we therefore analyze different subproblems of BAMs step by step.

3 Theoretical properties of BAMs

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 weights $\beta^{[k]}$ 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. BAMs with a linear model learner aim to iteratively find the parameters $\beta \in \mathbb{R}^p$ that minimize the L_2 loss. In the literature, this is known as L_2-Boosting [9], and the obtained BAM is linear in each feature. As the negative functional gradient at the current function estimate $\hat{f}^{[k]} = X\beta^{[k]}$ in (3) corresponds to the model residuals, L_2-Boosting corresponds to iteratively fitting the residuals with least squares [9]. The paths of weights and the induced shrinkage compared to the ordinary least squares (OLS) solution can be characterized exactly in this case.

Proposition 1. *If X is of full column rank, the estimates of L_2-Boosting with linear models and joint updates in iteration k are given by*

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

where $\nu \in (0,1]$ denotes the step size, $\beta^{OLS} := (X^\top X)^{-1}X^\top y$ the OLS solution, and $\delta^{[k]} := \left(1-(1-\nu)^k\right) \in [0,1)$ the shrinkage factor at iteration k. For arbitrary step size $\nu \in (0,1]$, it further holds

$$\beta^{[k]} \overset{k\to\infty}{\longrightarrow} \beta^{OLS}.$$

Proposition 1 shows that each iterate is the linear interpolation of the previous iterate and the OLS solution. As the shrinkage is determined by a single shrinkage factor, all parameters are shrinked equally. Further, the shrinkage factor $\delta^{[k]}$ is shown to depend on both ν and k and is strictly decreasing in both. As an immediate consequence, we get the convergence to the respective OLS estimator.

The previous result is to be expected as joint updates resemble Newton steps with step size ν. What is however less clear, is whether the weight paths correspond to the solutions of an explicitly regularized optimization problem such as Ridge regression or Lasso. That is to ask, what kind of problem is L_2-Boosting implicitly solving in this case. We obtain the following result:

Proposition 2. *For scaled orthogonal predictors, the weight paths of L_2-Boosting for linear models with joint updates correspond to the solutions of Ridge regression with varying regularization parameter $\lambda > 0$. That is $\forall \lambda > 0\ \exists\ k \in \mathbb{N}_0$ and $\nu \in (0,1)$ such that*

$$\beta^{Ridge}(\lambda) = \left(1-(1-\nu)^k\right) \beta^{OLS} := \beta^{[k]}(\nu). \quad (6)$$

Remark 1. *The regularization paths of L_2-Boosting and Ridge regression only match in the special case of scaled orthogonal predictors.*

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 [13] which utilize B-splines [12] 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 D. Boosting splines can be formulated as an L_2 loss optimization problem with quadratic penalization term:

$$\ell_P(\beta) = \frac{1}{2}\|y - X\beta\|^2 + \frac{\lambda}{2}\beta^\top P\beta, \tag{7}$$

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 [19, 46]. 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 weight paths and shrinkage can again be characterized exactly:

Proposition 3. *The estimates of L_2-Boosting with quadratic penalty and joint updates in iteration k are given by*

$$\beta^{[k]} = [\textstyle\sum_{m=0}^{k-1} \nu(I - \nu(X^\top X + \lambda P)^{-1}X^\top X)^m]\, \beta^{PLS}, \tag{8}$$

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, it holds that

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

Remark 2. *The above result also holds for the special case of Ridge regularization where $P = I$.*

Interestingly, the shrinkage in (8) can be recognized as a Neumann series and despite the explicit regularization converges to the unpenalized OLS fit in the limit. While Proposition 1 and Proposition 3 might seem similar at first, they imply different convergence paths, as the parameters are no longer equally shrunk when comparing explicit and implicit Ridge regularization (cf. Figure 8 in Appendix E.3).

Again, the general question arises as to whether these solution paths correspond to an explicitly regularized problem. A widespread interpretation of regularized spline boosting is that the algorithm implicitly minimizes (7) for a specific $\lambda > 0$ value in each iteration and thus implicitly finds the model with optimal λ. However, the next proposition states that the paths of the explicit and implicit regularized penalized least squares problem are in general not equivalent. This can be proven by matching the two approaches' hat matrix, i.e., the operator mapping y to f.

Proposition 4. *The hat matrix for penalized boosting in iteration k is $\mathcal{H}_\lambda^{[k]} = (I - (I - \mathcal{S}_\lambda)^{k+1})$ where $\mathcal{S}_\lambda = X(X^\top X + \lambda P)^{-1}X^\top$ denotes the usual hat matrix of a penalized regression model depending on λ. In general, we cannot find a $\tilde{\lambda} > 0$ for a given pair (λ, k) s.t. $S_{\tilde{\lambda}} = H_\lambda^{[k]}$.*

3.1.3 Boosting parameter flow

While the previous section studies BAMs for specific learning rates ν and proves the misalignment between boosting paths and Ridge regression, the previous results can also be confirmed when using an infinitesimally small learning rate by deriving the parameter flow of BAMs:

Lemma 1. *For BAMs with quadratic penalty, outcome y, predictor matrix X, and Moore-Penrose generalized inverse $X^\dagger = (X^\top X)^{-1}X^\top$, the parameter flow for $t \in \mathbb{R}_0^+$, subject to $\beta(0) = 0$, is*

$$\beta(t) = X^\dagger \left(I - \exp\left(-X(X^\top X + \lambda P)^{-1}X^\top \cdot t\right)\right) y. \tag{9}$$

Apart from providing the first parameter flow result for BAMs, the representation in Lemma 1 further characterizes the optimization problem that is implicitly solved by BAMs and explicitly shows that its convergence paths indeed differ from the paths of the explicitly minimized counterpart(Theorem 3 and Corollary 2 in Appendix C).

3.2 Block-wise boosting

We now turn to the block-wise boosting variant and our main results. The block-wise setting generalizes joint and component-wise updates, including both as special cases. To derive our main convergence results, we first characterize what optimization procedure is induced when using greedy updates:

Theorem 1. *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 (20,24). In the component-wise case, the optimization of BAMs matches greedy coordinate descent (GCD) with Gauss-Southwell-Lipschitz (GSL) update scheme (22,23).*

Remark 3. *The exact equivalence in Theorem 1 does no longer hold if the penalized L_2 loss* (7) *is used. However, even in this case, boosting additive models is a variant of G(B)CD with GSQ (GSL) which uses the gradient of the unpenalized problem together with the Hessian of the penalized problem in the selection and update steps.*

While GBCD has a long-standing history in the optimization literature, the two mentioned updates schemes of GSQ and GSL were only introduced recently and shown to be particularly efficient [35, 36]. We will use the above equivalence to derive convergence guarantees for BAMs with block-wise updates and L_2 loss, including the derivation of a convergence rate under certain conditions. 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 and relation to the class of convex quadratic problems). Using the condition of μ-PL instead of strong convexity allows studying convergence also in cases where the optimal solution is not unique.

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

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

where ℓ^* denotes the optimal loss, $\nu \in (0,1]$ the step size, $|\mathcal{B}|$ the number of blocks, and $L_{\mathcal{B}}$ the largest Lipschitz constant of all blocks with $L_{\mathcal{B}} := \max_{b\in\mathcal{B}} L_b \leq L$.

If $\ell(\beta)$ is a quadratic problem with positive semi-definite Hessian Q, the convergence rate γ of block-wise boosting admits

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

We obtain several insights from Theorem 2 related to boosting's convergence speed. First, γ is monotonically decreasing in the step size $\nu \in (0,1]$ 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 in γ can be identified as the reciprocal of the condition number of the Hessian Q. This matches the common notion that gradient methods converge faster for well-posed problems with condition numbers close to one [4].

The convergence of component-wise L_2-Boosting follows immediately from (10), 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 [16] which consider the special case of component-wise L_2-Boosting with standardized predictors, i.e., $L_{CLS} = 1$.

3.2.1 Regression splines

The previous results shows that greedy block-wise boosting differs from usual GBCD as it uses the gradient of the unpenalized problem (see Remark 3 and Appendix B.5.3). 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, BAMs might not be a favorable option. In this case, however, the previous results also provide a way forward by using

GBCD instead of boosting as fitting routine. 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. This is described in Corollary 1.

Corollary 1. *Using GBCD with GSQ update scheme to fit regression spline models of the form* (7) *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 [18]. While convergence guarantees for backfitting have been analyzed by [8] and [3], to the best of our knowledge, no explicit convergence rate has been derived to this date, making the rate in Corollary 1 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 [9]. 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. While [9] considered component-wise CSS in their experiments, they did not derive convergence results for the component-wise case. Using previous findings, the following Proposition 5 states the convergence of component-wise L_2-Boosting with multiple CSS (proof in Appendix B.8). Interestingly, Proposition 5 holds independently of the selection rule (greedy, cyclic, random, etc.).

Proposition 5. *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

From a statistical perspective, all previous results relate to boosting linear or additive models with L_2 loss, hence an (implicit) Gaussian distribution assumption. Our next contribution extends these results to models with exponential family distribution assumption. In this case, the density of the response can be written as

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

where the terms θ, $b(\theta)$, ϕ and $c(\phi, y)$ depend on the exponential family distribution (see, e.g., [14, 47]). The loss function w.r.t. the functional f is then defined by

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

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 [22], (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., logistic loss is known to be $1/4$-smooth, then the following proposition holds.

Proposition 6. *Optimization of BAMs with block-wise updates and exponential family loss ℓ corresponds to GBCD with GSQ update scheme as long as a sufficient upper bound of the Hessian is used in each step. If ℓ is μ-PL and L-smooth w.r.t. the parameters $\beta \in \mathbb{R}^p$, the procedure converges to an optimal solution with minimal loss.*

The derivation of the equivalence stated in Proposition 6 is given in Appendix B.9, where we also provide a remark about potential other losses outside the exponential family.

A distinct property of BAMs that distinguishes them from Newton-type methods is the use of $X_b^\top X_b$ instead of the actual Hessian block (27) to scale each gradient update. While replacing the true Hessian by a positive definite matrix is common practice and known under the term preconditioner methods [33], the convergence guarantee only holds if $X_b^\top X_b$ provides a valid upper bound to the respective block of the Hessian in (27) for all $b \in \mathcal{B}$. For this to hold in general, it is essential to choose the step-size ν to be small, as this inversely increases the size of $\frac{1}{\nu} X_b^\top X_b$. In Section 4.3, we show that the choice of step-size greatly impacts the convergence behavior for Poisson boosting. In Appendix E.2, we also investigate the example of boosting with logistic regression models where BAMs guarantee convergence due to Proposition 6, as for the Hessian Q_{log} of the logistic model, it holds $Q_{log} \preceq \frac{1}{4} X_b^\top X_b \preceq X_b^\top X_b \; \forall b \in \mathcal{B}$ for arbitrary step-size $\nu \in (0, 1]$.

4 Numerical experiments

In the following, we numerically demonstrate our theoretical findings.

4.1 Convergence rates

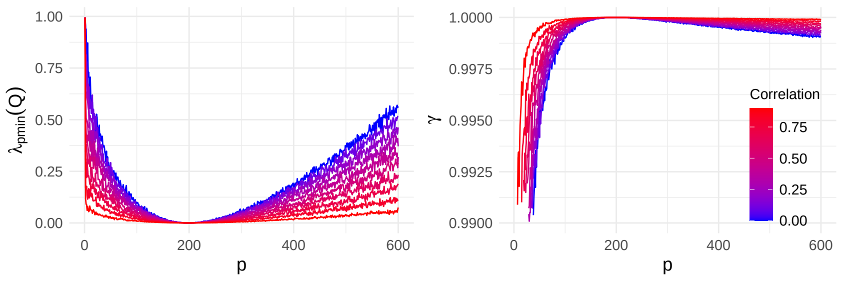

Figure 1: Smallest non-zero eigenvalue of Q (left) and convergence rate γ as given in (11) (right) for component-wise boosting for a linear model with varying pairwise correlation ρ between predictor variables (color) for fixed $n = 200$, $\nu = 1$, and varying p (x-axis).

In Figure 1, 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, the convergence rate is close to one, indicating relatively slow convergence behavior. An observation that matches the general notion of slow (over-) fitting behavior of boosting [7]. Convergence is generally faster (small γ values) for low pairwise correlation. For a growing number of predictors, γ steeply increases to a rate close to one until $p = n$. In the underdetermined case ($n < p$), the rate slowly decreases again while remaining close to one. This phenomenon is due to the smallest eigenvalue decreasing with the number of predictors while remaining above zero for full-rank tall matrices ($n > p$). By contrast, for flat matrices ($n < p$), more and more eigenvalues become zero, thus the smallest non-zero eigenvalue grows, resulting in an increase in convergence speed. We further investigate the impact of different condition numbers on the linear convergence rate in Figure 4 (in Appendix E). 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

To demonstrate empirically, that BAMs are converging to the unpenalized model fit, we run a more complex application of boosting a 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., [10]). 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. After creating $n = 300$ pairs of functions, discretized to 40 time points, we fit a functional additive model (FAM; [42]), the pendant of a penalized spline regression for functional data, and compare it with BAMs for functional data [6]. 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.

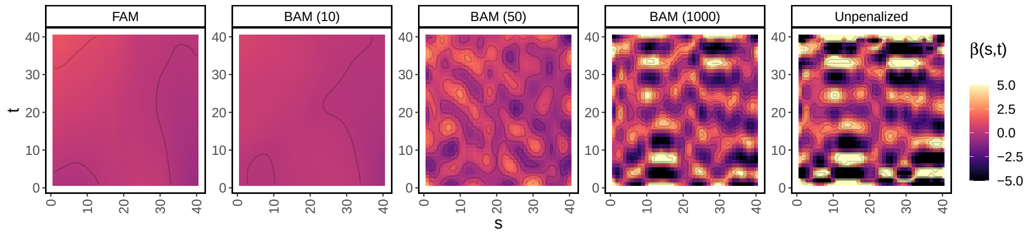

Figure 2: Exemplary estimated weight surfaces of a penalized functional regression (FAM; left) compared to BAM (second to fourth plot) with different numbers of iterations in brackets. For larger iterations, the estimated weight surface of BAM converges to the one of the unpenalized model (right plot).

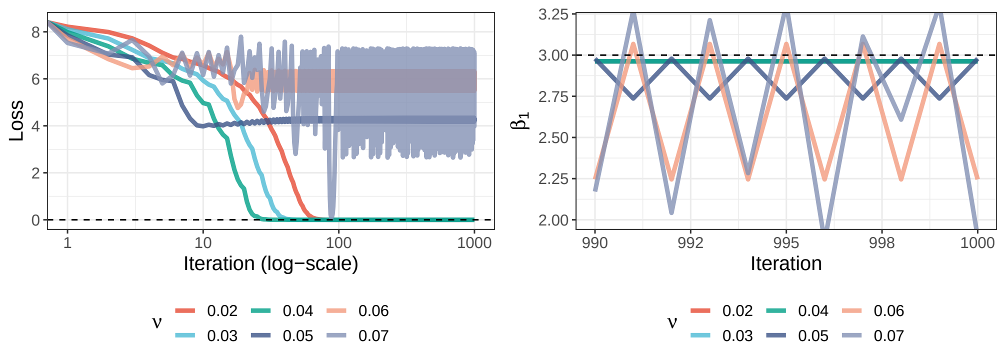

Figure 3: Left: Loss path for different learning rates (colors) showing convergence for BAMs with the three smaller rates and non-convergence for the larger ones. Right: Last 10 updates of the corresponding β_1 parameter, depicting oscillating updates for $\nu \geq 0.05$ due to inaccurate curvature approximation.

Results: Figure 2 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.

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 6, we observe convergence for Binomial BAMs for all different learning rates (see Figure 7 in Appendix E.2), 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 (see Figure 3).

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 show how to relate certain classes of BAM optimization to explicitly regularized problems, but also point out that this equivalence does not hold in general. 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 highlights 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 this work establishes important connections between convergence paths of BAMs and other well-known problems, it also demonstrates that the paths of boosted and explicitly regularized problems differ in general. This raises the question to what other explicit regularized problems the paths of these BAMs correspond to. Moreover, proper statistical inference for BAMs that accounts for the implicit regularization and model selection still remains an open challenge. Our work provides the foundational building blocks to address this challenge, paving the way for reliable statistical inference in the future.

References

[1] Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E Hinton. Neural additive models: Interpretable machine learning with neural nets. *Advances in Neural Information Processing Systems*, 34:4699–4711, 2021.

[2] Alnur Ali, J Zico Kolter, and Ryan J Tibshirani. A continuous-time view of early stopping for least squares regression. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1370–1378. PMLR, 2019.

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

[4] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[5] Sarah Brockhaus, Michael Melcher, Friedrich Leisch, and Sonja Greven. Boosting flexible functional regression models with a high number of functional historical effects. *Statistics and Computing*, 27:913–926, 2017.

[6] Sarah Brockhaus, David Rügamer, and Sonja Greven. Boosting functional regression models with fdboost. *Journal of Statistical Software*, 94(10):1–50, 2020.

[7] Peter Bühlmann and Torsten Hothorn. Boosting Algorithms: Regularization, Prediction and Model Fitting. *Statistical Science*, 22(4):477 – 505, 2007.

[8] Andreas Buja, Trevor Hastie, and Robert Tibshirani. Linear smoothers and additive models. *The Annals of Statistics*, pages 453–510, 1989.

[9] Peter Bühlmann and Bin Yu. Boosting with the l2 loss: Regression and classification. *Journal of the American Statistical Association*, 98(462):324–339, 2003.

[10] Mathieu Chalvidal, Thomas Serre, and Rufin VanRullen. Learning functional transduction. In *Advances in Neural Information Processing Systems*, volume 36, pages 73852–73865. Curran Associates, Inc., 2023.

[11] Michael Collins, Robert E Schapire, and Yoram Singer. Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 48:253–285, 2002.

[12] Carl de Boor. *A Practical Guide to Spline*, volume 27. Springer, 01 1978.

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

[14] Ludwig Fahrmeir, Thomas Kneib, Stefan Lang, and Brian Marx. *Regression: Models, Methods and Applications*. Springer, 01 2013.

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

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

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

[18] Jerome H Friedman and Werner Stuetzle. Projection pursuit regression. *Journal of the American statistical Association*, 76(376):817–823, 1981.

[19] Peter J. Green and Bernard Walter Silverman. *Nonparametric regression and generalized linear models*. Chapman & Hall/CRC Press, 1994.

[20] Torsten Hothorn and Peter Bühlmann. Model-based boosting in high dimensions. *Bioinformatics*, 22(22):2828–2829, 2006.

[21] Torsten Hothorn, Peter Bühlmann, Thomas Kneib, Matthias Schmid, and Benjamin Hofner. Model-based boosting 2.0. *Journal of Machine Learning Research*, 11(71):2109–2113, 2010.

[22] Hamed Karimi, Julie Nutini, and Mark Schmidt. 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, 2016.

[23] Francesco Locatello, Anant Raj, Sai Praneeth Karimireddy, Gunnar Rätsch, Bernhard Schölkopf, Sebastian Stich, and Martin Jaggi. On matching pursuit and coordinate descent. In *International Conference on Machine Learning*, pages 3198–3207. PMLR, 2018.

[24] Yin Lou, Rich Caruana, and Johannes Gehrke. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 150–158, 2012.

[25] Haihao Lu, Sai Praneeth Karimireddy, Natalia Ponomareva, and Vahab Mirrokni. Accelerating gradient boosting machines. In *International Conference on Artificial Intelligence and Statistics*, pages 516–526. PMLR, 2020.

[26] Haihao Lu and Rahul Mazumder. Randomized gradient boosting machine. *SIAM Journal on Optimization*, 30(4):2780–2808, 2020.

[27] Eva-Maria Maier, Almond Stöcker, Bernd Fitzenberger, and Sonja Greven. Additive density-on-scalar regression in bayes hilbert spaces with an application to gender economics. *arXiv preprint arXiv:2110.11771*, 2021.

[28] Llew Mason, Jonathan Baxter, Peter Bartlett, and Marcus Frean. Boosting algorithms as gradient descent. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

[29] Andreas Mayr, Nora Fenske, Benjamin Hofner, Thomas Kneib, and Matthias Schmid. 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, 2012.

[30] Ron Meir and Gunnar Rätsch. *An Introduction to Boosting and Leveraging*, volume 2600, pages 119–184. Springer, 01 2003.

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

[32] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2004.

[33] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

[34] Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*, 2019.

[35] Julie Nutini, Issam Laradji, and Mark Schmidt. 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, 2022.

[36] Julie Nutini, Mark Schmidt, Issam Laradji, Michael Friedlander, and Hoyt Koepke. Coordinate descent converges faster with the gauss-southwell rule than random selection. In *International Conference on Machine Learning*, pages 1632–1641. PMLR, 2015.

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

[38] Filip Radenovic, Abhimanyu Dubey, and Dhruv Mahajan. Neural basis models for interpretability. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[39] Gunnar Rätsch, Sebastian Mika, and Manfred K. K Warmuth. On the convergence of leveraging. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001.

[40] Havard Rue and Leonhard Held. *Gaussian Markov random fields: theory and applications*. Chapman and Hall/CRC, 2005.

[41] David Ruppert, Matt P Wand, and Raymond J Carroll. *Semiparametric regression*. Number 12. Cambridge university press, 2003.

[42] Fabian Scheipl, Ana-Maria Staicu, and Sonja Greven. Functional additive mixed models. *Journal of Computational and Graphical Statistics*, 24(2):477–501, 2015.

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

[44] Almond Stöcker, Lisa Steyer, and Sonja Greven. Functional additive models on manifolds of planar shapes and forms. *Journal of Computational and Graphical Statistics*, 32(4):1600–1612, 2023.

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

[46] G. Wahba. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, Philadelphia, 1990.

[47] Simon N Wood. *Generalized additive models: an introduction with R*. CRC press, 2017.

[48] Tong Zhang and Bin Yu. Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, 33(4):1538–1579, 2005.

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., [4, 31, 32]) 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 $\forall \beta, \tilde{\beta} \in \mathbb{R}^p$

$$\ell(\tilde{\beta}) \leq \ell(\beta) + \langle \nabla \ell(\beta), \tilde{\beta} - \beta \rangle + \frac{L}{2}\|\tilde{\beta} - \beta\|^2. \tag{14}$$

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

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

For twice-differentiable objective functions, the conditions in (14) and (15) 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, [22] showed that it is sufficient to consider the PL-inequality instead of strong convexity to derive convergence rates for iterative gradient methods.

Definition 2. *[22] A function $f : \mathbb{R}^p \to \mathbb{R}$ is μ-PL with some $\mu > 0$, if $\forall \beta \in \mathbb{R}^p$:*

$$\frac{1}{2}\|\nabla \ell(\beta)\|^2 \ \geq \ \mu \left(\ell(\beta) - \ell^*\right). \tag{16}$$

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 f 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

$$\min_{\beta \in \mathbb{R}^p} \ell(\beta) = \min_{\beta \in \mathbb{R}^p} \frac{1}{2}\beta^\top Q \beta + q_1^\top \beta + q_0, \tag{17}$$

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 (17) is L-smooth with some Lipschitz constant $L \leq \lambda_{max}(Q)$, where $\lambda_{max}(Q)$ denotes the largest eigenvalue of Q.

Further, [16, 22] showed that any problem that can be written in the form of (17) 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

$$\|\nabla_b \ell(\beta + U_b \upsilon) - \nabla_b \ell(\beta)\|_{H_b^{-1}} \leq \|\upsilon\|_{H_b}, \tag{18}$$

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, (18) 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., $\nabla^2_{bb}\ell(x) \preceq H_b$. Note that (18) contains the usual L-smoothness condition as described in (14) as a special case by choosing $H_b = L_b I$, with L_b being the Lipschitz constant for block b [35]. Conversely, (18) can be shown to hold, whenever the L-smoothness condition (14) 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

$$\beta^{[k+1]} = \beta^{[k]} - \nu \nabla_{i_k} \ell\left(\beta^{[k]}\right) e_{i_k}, \tag{19}$$

where e_{i_k} is the unit vector corresponding to the variable $i_k \in \{1, \ldots, p\}$ 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

$$\varkappa_{b_k} = -(H_{b_k})^{-1}\nabla_{b_k} f\left(\beta^{[k]}\right), \tag{20}$$

where H_{b_k} could correspond to the respective block of the Hessian or an upper bound of the latter [35].

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

$$i_k = \arg\max_i |\nabla_i \ell(\beta^{[k]})|. \tag{21}$$

A more elaborated update routine, the so called Gauss-Southwell-Lipschitz (GSL) rule, was proposed by [36] and is given by

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

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 (19) as well, yielding

$$\beta^{[k+1]} = \beta^{[k]} - \nu\frac{1}{L_{i_k}}\nabla_{i_k}\ell\left(\beta^{[k]}\right)e_{i_k}. \tag{23}$$

Gauss-Southwell-Quadratic Analogously to the block update in (20), a greedy block selection rule called Gauss-Southwell-Quadratic (GSQ) is defined by

$$b_k \quad = \quad \arg\max_{b\in\mathcal{B}} \quad \{\|\nabla_b \ell(\beta^{[k]})\|_{H_b{}^{-1}}\}, \tag{24}$$

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) [35].

A.3 Linear operators

We will prove Proposition 5 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. $\|T\|\|c\| = \|c\|\|T\frac{c}{\|c\|}\| \le \|c\|\sup_{\|c\|\neq 0}\|T\frac{c}{\|c\|}\| = \|c\|\|T\|^*$;
3. $\|T_1T_2\|^* = \sup_{\|z\|\neq 0}\frac{\|T_1T_2z\|}{\|z\|} = \sup_{\|z\|\neq 0}\left(\frac{\|T_1T_2z\|}{\|T_2z\|}\frac{\|T_2z\|}{\|z\|}\right) \le \sup_{\|T_2z\|\neq 0}\frac{\|T_1T_2z\|}{\|T_2z\|}\sup_{\|z\|\neq 0}\frac{\|T_2z\|}{\|z\|} = \|T_1\|^*\|T_2\|^*$.

A.4 Exponential family

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

$$p_\theta(y) = \exp\left[\{y\theta - b(\theta)\}/\phi + c(\phi, y)\right], \tag{25}$$

where the terms θ, $b(\theta)$, ϕ and $c(\phi, y)$ depend on the exponential family distribution considered [14, 47]. 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 $\theta_i = f_i$ [14]. 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

$$\ell(f) = \ell(\theta(f)) = \{y\theta(f) - b(\theta(f))\}/\phi + c(\phi, y). \tag{26}$$

Another key feature of the canonical link is that (26) 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 (26). We do so by considering GLMs, for which (26) 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 [47]. The latter [derived, e.g., in 14] corresponds to

$$\nabla^2 \ell(\beta) = X^\top W X \tag{27}$$

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

$$\tilde{w_i} = \frac{(h'(f_i))^2}{b''(\theta_i)\phi}. \tag{28}$$

By looking at the definition of the respective terms for different exponential family distributions, it becomes clear that $b''(\theta) > 0$ and $\phi > 0$ [14, 47]. 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 Proof of Proposition 1

Proof. 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 $\beta^{[0]} = 0$):

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

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). $\square$

B.2 Proof of Proposition 2

Proof. Consider the design matrix X with scaled orthogonal predictors. So X has orthogonal columns each with variance σ_X^2. Thus, $X^\top X = \sigma_X^2 I$. Then we can rewrite the Ridge estimator in terms of the OLS estimator:

$$\begin{aligned}
\beta^{Ridge}(\lambda) &= (X^\top X + \lambda I)^{-1}X^\top y = (\sigma_X^2 I + \lambda I)^{-1}X^\top y = (\frac{1}{\sigma_X^2 + \lambda})X^\top y \\
&= (\frac{1}{\sigma_X^2 + \lambda})(X^\top X)(X^\top X)^{-1}X^\top y = (\frac{\sigma_X^2}{\sigma_X^2 + \lambda})\beta^{OLS}
\end{aligned}$$

Given the above derivation for $\forall \lambda > 0 \; \exists \; k \in \mathbb{N}_0$ and $\nu \in (0,1)$ s.t.

$$\beta^{Ridge}(\lambda) = (\frac{\sigma_X^2}{\sigma_X^2 + \lambda})\beta^{OLS} = \left(1 - (1-\nu)^k\right)\beta^{OLS} = \beta^{[k]}(\nu). \tag{29}$$

A simple derivation shows that we can choose the learning rate as a function of k and λ by

$$\nu(\lambda, k) := 1 - \sqrt[k]{1 - \frac{\sigma_X^2}{\sigma_X^2 + \lambda}}$$

such that (29) is fulfilled. Note that $\nu(\lambda, k) \in (0,1)$. This also shows that the pair (k, ν) of boosting iterations and step size fulfilling the above condition is not unique. □

B.3 Proof of Proposition 3

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 $\beta^{[0]} = 0$):

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

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 $\nu \in (0,1]$. 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 [43]. 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:

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

□

B.4 Derivation of Proposition 4

Derivation. The hat matrix for penalized boosting has the following form after boosting iteration k [43]:

$$\mathcal{H}_\lambda^{[k]} = (I - (I - \mathcal{S}_\lambda)^{k+1}),$$

where $\mathcal{S}_\lambda = X(X^\top X + \lambda P)^{-1} X^\top$ denotes the usual hat matrix when solving the penalized regression spline (or, in general, the quadratic penalized) problem explicitly given a regularization parameter $\lambda > 0$.

The hat matrices $\mathcal{S}_\lambda$ and $\mathcal{H}_\lambda^{[k]}$ can be further investigated by rewriting them using a thin singular value decomposition (SVD) of the matrix X. We assume X to be of $\text{rank}(X) = p$ and consider the SVD, $X = UDV^\top$, where U is a $(n \times p)$ matrix, V a $(p \times p)$ matrix and $D = \text{diag}(\sigma_1, \ldots, \sigma_p)$ containing the eigenvalues of X. Note that we have $U^\top U = I_p$ and $V^\top V = I_p$. Then we get that:

$$\begin{aligned}
\mathcal{S}_\lambda &= UDV^\top (VDU^\top UDV^\top + \lambda P)^{-1} VDU^\top & (30)\\
&= UD(D^2 + \lambda V^{-1} P V^{-\top})^{-1} DU^\top & (31)\\
&= U(I_p + \lambda D^{-1} V^{-1} P V^{-\top} D^{-1})^{-1} U^\top & (32)\\
&= UG(\lambda)U^\top & (33)
\end{aligned}$$

Using this decomposition we can rewrite $\mathcal{H}_\lambda^{[k]}$:

$$\begin{aligned}
\mathcal{H}_\lambda^{[k]} &= I_n - (I_n - \mathcal{S}_\lambda)^{k+1}\\
&= I_n - (I_n - UG(\lambda)U^\top)^{k+1}\\
&= U(I_p - (I_p - G(\lambda))^{k+1})U^\top
\end{aligned}$$

The last line follows from the fact that $U^\top U = I$ and $G(\lambda)$ a symmetric matrix. Note that $\mathcal{H}_\lambda^{[k]}$ in the last equation is in general not equivalent to the hat matrix of the same penalized quadratic or regression spline problem

$$\mathcal{S}_{\tilde{\lambda}} = UG(\tilde{\lambda})U^\top,$$

for any regularization parameter $\tilde{\lambda} > 0$. That is, the fit of an implicitly regularized spline model via boosting does not need to correspond to a solution of the respective spline optimization problem (problem with explicit regularization). Hence, boosting regularized models cannot in general be seen as implicitly selecting the regularization parameter of the respective regularized problem. The latter is also demonstrated along the example of boosting Ridge penalized linear models in Fig. 8.

□

B.5 Derivation of Theorem 1

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:

$$\begin{aligned}
\hat{b}_k &= \underset{b \in \mathcal{B}}{\arg\min} \left(u^{[k]} - X_b \hat{\beta}_b\right)^2\\
&= \underset{b \in \mathcal{B}}{\arg\min} \quad - {u^{[k]}}^\top X_b \left(X_b^\top X_b\right)^{-1} X_b^\top u^{[k]}\\
&= \underset{b \in \mathcal{B}}{\arg\max} \quad \sqrt{(\nabla_b \ell(\beta^{[k]}))^\top \left(H_b\right)^{-1} \nabla_b \ell(\beta^{[k]})}\\
&= \underset{b \in \mathcal{B}}{\arg\max} \quad \|\nabla_b \ell(\beta^{[k]})\|_{H_b{}^{-1}} \quad \text{(GSQ rule)}.
\end{aligned} \tag{34}$$

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 (24), we further notice that the update step for this L_2-Boosting variant is

$$\beta^{[k+1]} = \beta^{[k]} + \nu U_{\hat{b}_k} \hat{\beta}_{\hat{b}_k}, \tag{35}$$

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

$$\hat{\beta}_{\hat{b}_k} = \left(X_{\hat{b}_k}^\top X_{\hat{b}_k} \right)^{-1} X_{\hat{b}_k}^\top u^{[k]} = -(H_{\hat{b}_k})^{-1} \nabla_{\hat{b}_k} \ell \left(\beta^{[k]} \right). \tag{36}$$

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

$$\begin{aligned} \hat{\jmath}_k &= \underset{1 \le j \le d}{\arg\min} (u^{[k]} - X_j \hat{\beta}_j)^2 = \underset{1 \le j \le d}{\arg\min} \; - \frac{(X_j^\top u^{[k]})^2}{X_j^\top X_j} \\ &= \underset{1 \le j \le d}{\arg\max} \; \frac{(\nabla_j \ell(\beta^{[k]}))^2}{L_j} \quad \text{(GSL rule)}. \end{aligned}$$

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 (22). As before, we can examine the update step for this L_2-Boosting variant, which is

$$\beta^{[k+1]} = \beta^{[k]} + \nu e_{\hat{\jmath}_k} \hat{\beta}_{\hat{\jmath}_k}, \tag{37}$$

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

$$\hat{\beta}_{\hat{\jmath}_k} = \frac{X_{\hat{\jmath}_k}^\top u^{[k]}}{X_{\hat{\jmath}_k}^\top X_{\hat{\jmath}_k}} = - \frac{\nabla_{\hat{\jmath}_k} \ell(\beta^{[k]})}{L_{\hat{\jmath}_k}}. \tag{38}$$

This, again can be seen to be identical to the GSL update step as defined in (23). 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 3

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, 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 (7). 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

$$\begin{aligned} \hat{b}_k &= \underset{b \in \mathcal{B}}{\arg\min} \left(u^{[k]} - X_b \hat{\beta}_b \right)^2 + \lambda \hat{\beta}_b^\top P_b \hat{\beta}_b \\ &= \underset{b \in \mathcal{B}}{\arg\min} - {u^{[k]}}^\top X_b \left(X_b^\top X_b + \lambda P_b \right)^{-1} X_b^\top u^{[k]} \\ &= \underset{b \in \mathcal{B}}{\arg\max} \| \nabla_b \ell_{LS}(\beta^{[k]}) \|_{(H_b^{PLS})^{-1}} \quad \text{(GSQ rule)}. \end{aligned} \tag{39}$$

In the second line in (39), 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. The gradients of the penalized and unpenalized problem are only equivalent in the first iteration when no component is penalized yet. Thus, boosting still applies a GSQ rule, but uses the Hessian of the penalized instead of the unpenalized problem. The same can be observed for the update step which is

$$\beta^{[k+1]} = \beta^{[k]} + \nu U_{\hat{b}_k} \hat{\beta}_{\hat{b}_k} \tag{40}$$

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

$$\hat{\beta}_{\hat{b}_k} = -\left(H^{PLS}_{\hat{b}_k}\right)^{-1} \nabla_{\hat{b}_k} \ell^{LS}(\beta^k). \tag{41}$$

Note, that (39) and (41) scale the gradient with H^{PLS}_b, which is a block from the Hessian of the penalized problem. Hence, (39) is not equivalent to the update step that we have seen for L_2-Boosting (34). Further, in case GBCD with GSQ rule is applied to the same penalized problem in Eq. (7), the selection and updates steps would be identical to (39) and (41), with the important distinction that the gradient of the penalized problem is used. Hence, the two procedures are not equivalent in this case. □

B.6 Proof of Theorem 2

Proof. In the following, we adopt some of the techniques used by [35], 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 $\beta \in \mathbb{R}^p$. Using the matrix formulation of the L-smoothness condition (18), we can derive the following upper bound:

$$\begin{aligned} \ell(\beta^{[k+1]}) &\leq \ell(\beta^{[k]}) + \langle \nabla_{b_k} f(\beta^{[k]}), \beta^{[k+1]} - \beta^{[k]} \rangle + \frac{1}{2} \|\beta^{[k+1]} - \beta^{[k]}\|^2_{H_{b_k}} \\ &= \ell(\beta^{[k]}) - \nu(1 - \frac{\nu}{2}) \|\nabla_{b_k} \ell(\beta^{[k]})\|^2_{H_{b_k}{}^{-1}}, \end{aligned} \tag{42}$$

where the first inequality follows from the L-smoothness in (18) and for the second equality we used the GSQ related update (38) for $\beta^{[k+1]}$. We can rewrite the upper bound in (42) by using the norm

$$\|\vartheta\|_{\mathcal{B}} = \max_{b \in \mathcal{B}} \|\vartheta_b\|_{H_b{}^{-1}} \tag{43}$$

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 (42) as

$$\ell(\beta^{[k+1]}) \leq \ell(\beta^{[k]}) - \nu(1 - \frac{\nu}{2}) \|\nabla \ell(\beta^{[k]})\|_{\mathcal{B}}. \tag{44}$$

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 (43), 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

$$\|\vartheta\|_2^2 = \sum_b^{|\mathcal{B}|} \|\vartheta_b\|_2^2 \leq \sum_b^{|\mathcal{B}|} \lambda_{max}(H_b)\, \vartheta_b^\top H_b{}^{-1} \vartheta_b \leq L_{\mathcal{B}} \sum_b^{|\mathcal{B}|} \vartheta_b^\top H_b{}^{-1} \vartheta_b \leq L_{\mathcal{B}}\, |\mathcal{B}| \|\vartheta\|^2_{\mathcal{B}}. \tag{45}$$

Thus, we obtain the PL-inequality

$$\frac{1}{2} \|\nabla \ell(\beta)\|^2_{\mathcal{B}} \;\geq\; \frac{\mu}{L_{\mathcal{B}} |\mathcal{B}|} \left(\ell(\beta) - \ell^*\right), \tag{46}$$

where μ corresponds to the PL parameter of Definition 2 with L_2-norm. Lastly, by connecting the inequalities (42) and (46), and iterating over k iterations, we get our final convergence result

$$\ell(\beta^{[k]}) - \ell^* \leq \left(1 - \nu(2 - \nu) \frac{\mu}{L_{\mathcal{B}} |\mathcal{B}|}\right)^k \left(\ell(\beta^{[0]}) - \ell^*\right). \tag{47}$$

For quadratic functions in particular, $\mu = \lambda_{pmin}(Q)$ and we can use that $L_{\mathcal{B}} \leq \lambda_{max}(Q)$ to get

$$\ell(\beta^{[k]}) - \ell^* \leq \underbrace{\left(1 - \nu(2 - \nu) \frac{1}{|\mathcal{B}|} \frac{\lambda_{pmin}(Q)}{\lambda_{max}(Q)}\right)^k}_{=:\gamma} \left(\ell(\beta^{[0]}) - \ell^*\right). \tag{48}$$

□

B.7 Proof of Corollary 1

Proof. Corollary 1 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 1, 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 5

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 $0 \leqslant (\tilde{\lambda}_m)_i < 1$ [9]. 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

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

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

B.9 Derivation of Proposition 6 and further remarks

Derivation. As described in Algorithm 1, in each iteration BAMs fit a certain base procedure 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 that for GLMs we have $f = X\beta$, one can do so by

$$-\frac{\partial}{\partial \beta} \ell(\beta) = \frac{\partial}{\partial \beta} f(\beta) \; \widetilde{y} = X^\top \widetilde{y}. \tag{49}$$

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

$$\begin{aligned}
\hat{b}_k &= \arg\min_{b \in \mathcal{B}} \left(\widetilde{y}^{[k]} - X_b \hat{\beta}_b \right)^2 \\
&= \arg\min_{b \in \mathcal{B}} \quad - \widetilde{y}^{[k]^\top} X_b \left(X_b^\top X_b \right)^{-1} X_b^\top \widetilde{y}^{[k]} \\
&= \arg\max_{b \in \mathcal{B}} \quad \sqrt{(\nabla_b \ell(\beta^{[k]}))^\top \left(X_b^\top X_b \right)^{-1} \nabla_b \ell(\beta^{[k]})} \\
&= \arg\max_{b \in \mathcal{B}} \quad \| \nabla_b \ell(\beta^{[k]}) \|_{(X_b^\top X_b)^{-1}} \quad \text{(GSQ rule)},
\end{aligned} \tag{50}$$

which is analogous to the derivation in (34) and corresponds to the GSQ rule. Similarly, the update is

$$\hat{\beta}^{[k+1]} = \hat{\beta}^{[k]} + \nu U_{\hat{b}_k} \hat{\beta}_{\hat{b}_k} \tag{51}$$

with

$$\hat{\beta}_{\hat{b}_k} = \left(X_{\hat{b}_k}^\top X_{\hat{b}_k}\right)^{-1} X_{\hat{b}_k}^\top \tilde{y}^{[k]} = -\left(X_{\hat{b}_k}^\top X_{\hat{b}_k}\right)^{-1} \nabla_{\hat{b}_k} \ell(\beta^{[k]}). \tag{52}$$

Therefore, as long as $X_b^\top X_b$ provides an upper bound to the respective blocks of the Hessian in (27) 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. □

Remark 4. *Robust loss functions such as the L_1 and Huber loss are frequently used alternatives to the L_2 loss for gradient boosting methods [7]. 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.*

C Boosting flow for penalized problems

We first derive the boosting parameter flow.

C.1 Proof of Lemma 1

Derivation. Starting with the equality from previous proofs, we have

$$\begin{aligned} \beta^{[k]} &= \beta^{[k-1]} + \nu(X^\top X + \lambda P)^{-1} X^\top (y - X\beta^{[k-1]}) \\ \Leftrightarrow \frac{\beta^{[k]} - \beta^{[k-1]}}{\nu} &= (X^\top X + \lambda P)^{-1} X^\top (y - X\beta^{[k-1]}) \end{aligned} \tag{53}$$

For $\nu \to 0$ and $t \in \mathbb{R}_0^+$, we can convert this into a differential equation:

$$\frac{\beta(t+\nu) - \beta(t)}{\nu} \to \frac{d\beta(t)}{dt} = (X^\top X + \lambda P)^{-1} X^\top (y - X\beta(t)).$$

which gives us the differential equation that describes the gradient flow. The flow of the residuals is then given by

$$\frac{du(t)}{dt} = X(X^\top X + \lambda P)^{-1} X^\top u(t).$$

This ordinary differential equation describes an exponential decay of the residual, which has a simple solution and can be expressed in terms of the matrix exponential:

$$X\beta(t) - y = \exp\left(-X(X^\top X + \lambda P)^{-1} X^\top \cdot t\right) u(0),$$

where $\exp$ denotes the matrix exponential. Since $u(0) \equiv y$, we can rewrite this to the final boosting parameter flow:

$$\beta(t) = \beta^{OLS} - X^\dagger \exp\left(-X(X^\top X + \lambda P)^{-1} X^\top \cdot t\right) y \tag{54}$$

or alternatively

$$\beta(t) = X^\dagger (I - \exp\left(-X(X^\top X + \lambda P)^{-1} X^\top \cdot t\right)) y \tag{55}$$

□

C.2 Explicit optimization problem

Using the Demmler-Reinsch orthogonalization (DRO; [41]), we can write

$$X(X^\top X + \lambda P)^{-1} X^\top = A(I + \lambda \text{diag}(\sigma))^{-1} A^\top$$

with $A = XR^{-1}U$ with R from the Cholesky decomposition $X^\top X = R^\top R$ and vector σ from the singular value decomposition $R^{-T} P R^{-1} = U \text{diag}(\sigma) U^\top$. As $A^\top A = I$, we get

$$\exp\left(-X(X^\top X + \lambda P)^{-1} X^\top \cdot t\right) = A \exp\left(-(I + \lambda \text{diag}(\sigma))^{-1} \cdot t\right) A^\top.$$

due to the properties of the matrix exponential and

$$\beta(t) = R^{-1}U(I - \exp\left(-(I + \lambda \text{diag}(\sigma))^{-1} \cdot t\right) U^\top R^{-\top} X^\top y. \tag{56}$$

Note that using this formulation, we further have

$$X\beta(t) = A(I - \exp\left(-(I + \lambda \text{diag}(\sigma))^{-1} \cdot t\right) A^\top y. \tag{57}$$

As done in [2], we can further characterize the optimization problem solved by joint updates of BAMs with quadratic penalty:

Theorem 3. *Given* (56) *and* $\beta(0) = 0$*, the boosted parameter flow as defined in Lemma 1 solves the optimization problem*

$$\textit{minimize}_{\beta \in \mathbb{R}^p} \|y - X\beta\|^2 + \beta^\top \Xi_t \beta \tag{58}$$

where $\Xi_t = R^\top U(-\exp\left(-(I + \lambda \textit{diag}(\sigma))^{-1} \cdot t\right)^{-1} U^\top R.$

Proof. From (56) and (58), the following equality must hold:

$$R^{-1}U(I - \exp\left(-(I + \lambda \text{diag}(\sigma))^{-1} \cdot t\right) U^\top R^{-\top} X^\top y = (X^\top X + \Xi_t)^{-1} X^\top y. \tag{59}$$

Hence, we must have

$$R^{-1}U(I - \exp\left(-(I + \lambda \text{diag}(\sigma))^{-1} \cdot t\right) U^\top R^{-\top} = (X^\top X + \Xi_t)^{-1}.$$

Inverting both sides and solving for Ξ_t, we get

$$\Xi_t = R^\top U(I - \exp\left(-(I + \lambda \text{diag}(\sigma))^{-1} \cdot t\right)^{-1} U^\top R - X^\top X.$$

Using $X^\top X = R^\top U U^{-1} R$, we finally get

$$\Xi_t = R^\top U(-\exp\left(-(I + \lambda \text{diag}(\sigma))^{-1} \cdot t\right)^{-1} U^\top R.$$

□

C.3 Shrinkage maps

Let $a_i, i \in [p]$ be the columns of A. Following [2], we can derive the shrinkage map for BAMs and compare it to the one of Ridge regression with explicit regularization.

Corollary 2. *Both BAMs with implicit Ridge regularization and regression with explicit Ridge regularization define linear smoothers of the form*

$$\sum_{i=1}^{p} \varrho(s_i, \varphi) \cdot a_i a_i^\top y \tag{60}$$

with spectral shrinkage map $\varrho(\cdot, \varphi)$ *defined in [2] and parameters* φ*. This map is defined as* $\varrho^{Ridge}(\sigma, \lambda) = \sigma/(\sigma + \lambda)$ *[2]. The map for BAM is given by* $\varrho^{BAM}(\sigma, t) = 1 - \exp(-(t/(1 + \lambda\sigma)))$*.*

Proof. The fact that BAMs can be defined as linear smoother and the corresponding shrinkage map directly follow from the representation in (56). □

D Cubic smoothing splines and P-splines

Cubic Smoothing Splines. A *cubic smoothing spline* as considered in Section 3.2.2 is defined for a with a twice differentiable function f, by

$$\sum_{i=1}^{n} (y_i - f(x_i))^2 + \alpha \int \left(\frac{\partial^2 f}{\partial x^2}\right)^2 dx, \tag{61}$$

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 [9] 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, [13] proposed a specific type of penalized regression splines, which uses B-splines [12] 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 piecewise 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 [14], 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

$$\|y - X\beta\|^2 + \lambda\beta^\top \underbrace{D_2^\top D_2}_{:=P} \beta, \tag{62}$$

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 (62) is that the dimensionality of the penalization term is greatly reduced compared to the penalty in (61), as we consider a discretized version of it [43]. 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 (62) needs to be solved repeatedly.

E Additional experiments, experimental details and results

E.1 Convergence of penalized boosting to unpenalized model

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 5, this is in fact the case, despite the exact Ridge solution (without boosting) being far away from the LS solution.

As a second example, we fit a univariate spline problem using unpenalized and penalized B-splines (B-/P-splines, resp.) using boosting. Analogous to the result of Ridge boosting, we can observe (Figure 6) that boosting converges to the unpenalized B-spline 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.

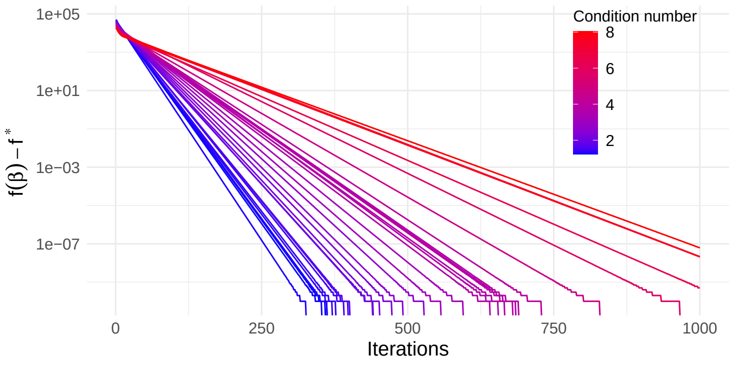

Figure 4: Linear convergence for different condition numbers (indicated by the color) for a linear model with two predictor variables. The condition numbers are induced by pairwise correlation of the predictor variables. Y axis on a logarithmic scale.

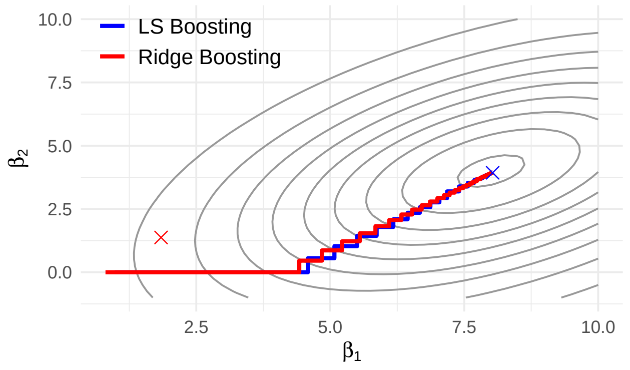

Figure 5: Parameter paths for LS (blue line) and Ridge (red line) boosting as well as their actual solutions (crosses in resp. colors).

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

E.2 Exponential family boosting

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 $h(f)$.

The plot for the binomial distribution for the experiment in Section 4.3 is given in Figure 7.

E.3 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

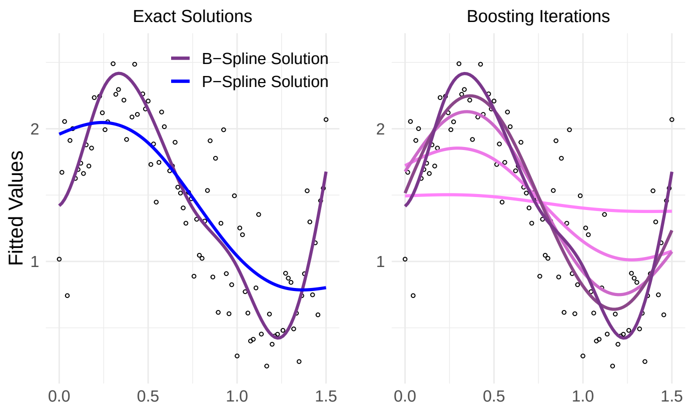

Figure 6: Left: Exact B-spline (purple) and P-spline solution (blue). Right: P-spline boosting iterates converging to the unpenalized (B-spline) solution (darkest color).

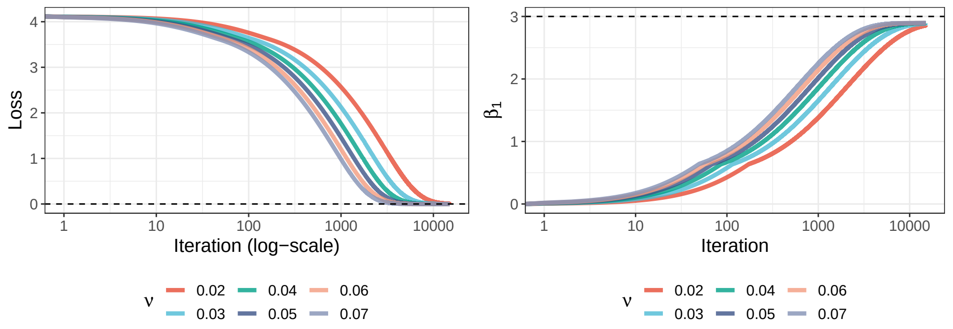

Figure 7: Left: Loss path for different learning rates (colors) showing convergence for BAMs for all rates. Right: corresponding β_1 parameters.

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 8 (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.

For each of the lines in Fig. 8, there is a theoretical boosting parameter path that is derived in Appendix C. We can check the equivalence by plotting the obtained boosting path against the path obtained by the corresponding flow (Fig. 9).

F 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.

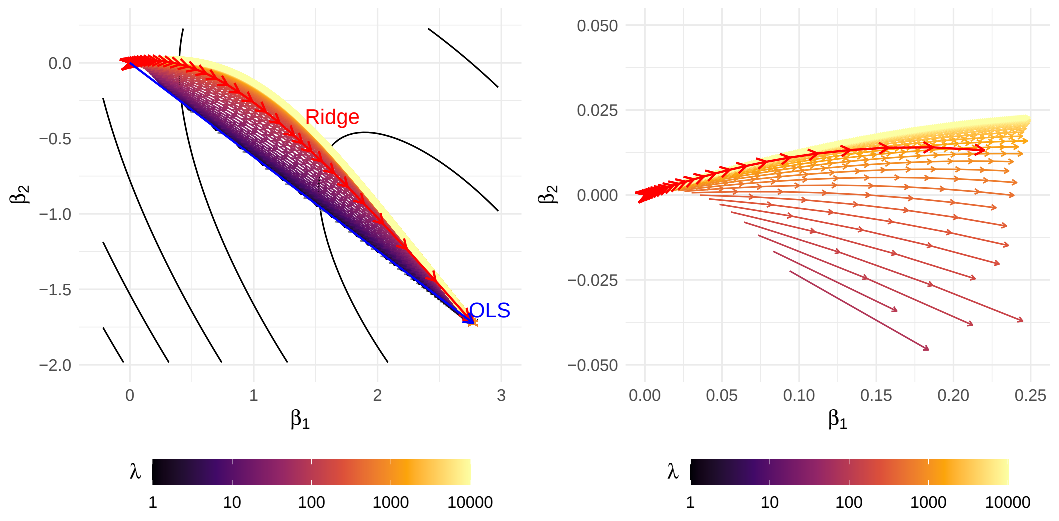

Figure 8: Left: Paths of BAM for different λ values (colored lines according to the legend) together with the Ridge regression path (red) and the OLS solution (blue). Contour lines in the background represent the loss surface. Right: Same plot as on the left but zoomed in.

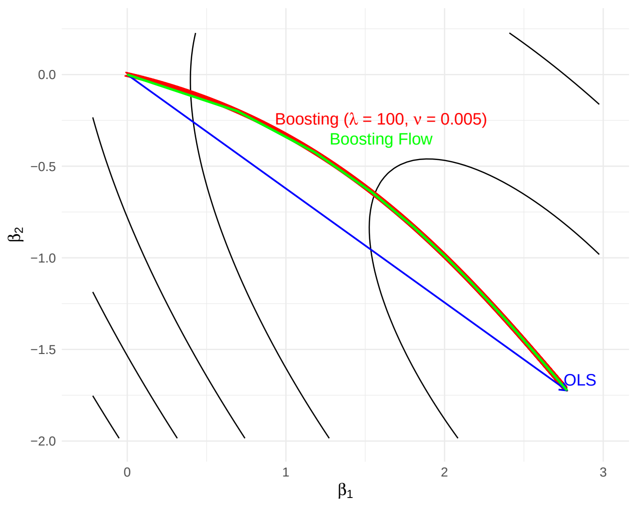

Figure 9: Example demonstrating the match between the theoretical boosting parameter flow and the discrete boosting path for a given step length ν and penalization λ