LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

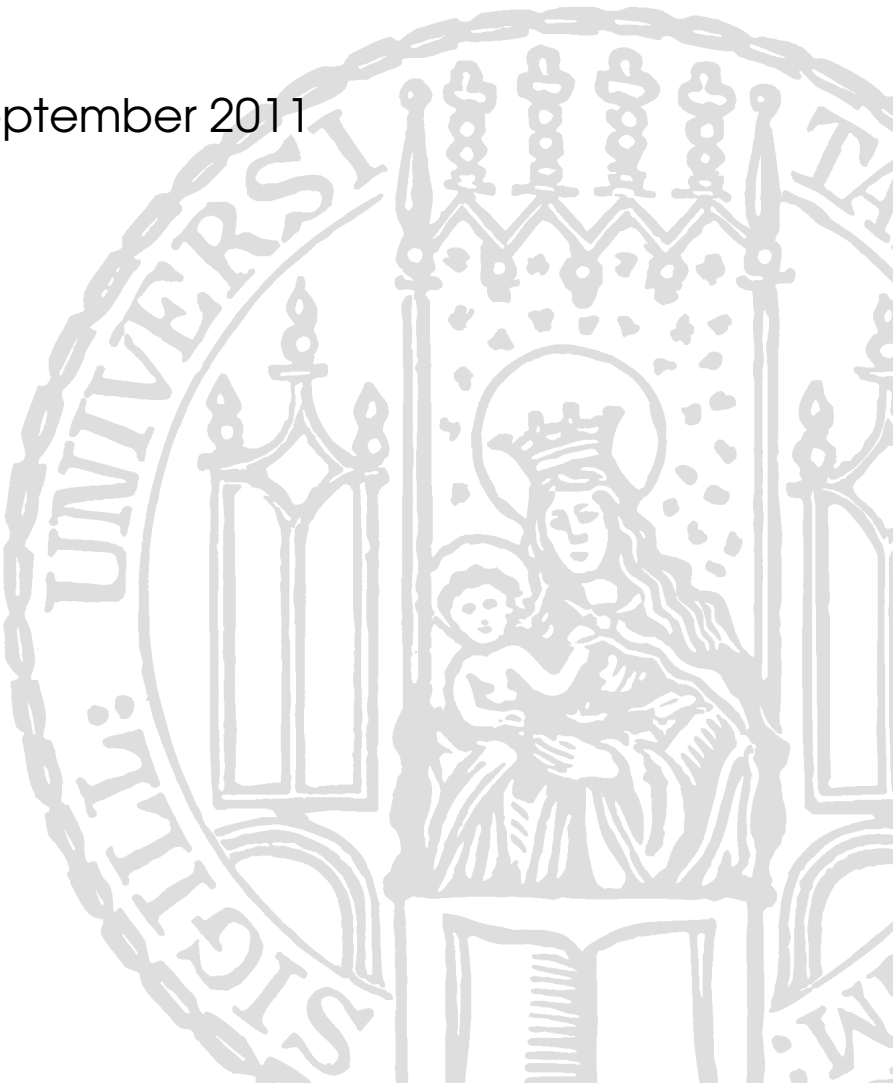
Boosting in Structured Additive Models

DISSERTATION

an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig-Maximilians-Universität München

vorgelegt von Diplom-Statistiker
BENJAMIN HOFNER

München, den 29. September 2011



Datum der Einreichung: 29. September 2011

Erstberichterstatter: Prof. Dr. Torsten Hothorn
Zweitberichterstatter: Prof. Dr. Olaf Gefeller
Auswärtiger Gutachter: Prof. Dr. Thomas Kneib

Tag der mündlichen Prüfung: 05. Dezember 2011

Danksagung

> A humane science must be adapted to the requirements of a balanced and rewarding life.
>
> *(Paul Karl Feyerabend[1])*

- Torsten Hothorn möchte ich an dieser Stelle danken für die sehr gute Betreuung dieser Arbeit und viele anregende und stimulierende Diskussionen, sowie für die außerordentlich gute Zusammenarbeit. Es war und ist mir eine große Freude gemeinsam an **mboost** zu arbeiten.

- Olaf Gefeller möchte ich danken für die stimulierende Atmosphäre am Institut und die Freiheit die ich in meiner Forschung hatte. Durch ihn hatte ich die Möglichkeit mich auf meine Dissertation zu konzentrieren, jedoch auch in anderen Bereichen Erfahrungen zu sammeln und dadurch viele Dinge in einem neuen Licht zu sehen. Des weiterem danke ich Ihm herzlich dafür, dass er sich bereit erklärt hat als Gutachter für diese Arbeit zu fungieren.

- Thomas Kneib sei gedankt für anregende Diskussionen und die daraus resultierenden Denkanstöße, für seine Unterstützung und nicht zuletzt für die Bereitschaft diese Arbeit zu begutachten.

- Matthias Schmid gebührt besonderer Dank für die hervorragende Betreuung vor Ort. Er war mir in vielen Punkten dieser Arbeit ein kritischer Diskussionspartner und hat mit seinen Anregungen maßgeblich zum Gelingen dieser Arbeit beigetragen.

- Jörg Müller möchte ich für die gute Zusammenarbeit bei den Species Distribution Models für den Rotmilan danken. H.-J. Fünfstück vom Bayer. Landesamt für Umwelt, Staatliche Vogelschutzwarte sei gedankt für das Überlassen des Rotmilan-Bildes (Fig. 4.10).

[1] Feyerabend PK (1999). *Conquest of Abundance: A Tale of Abstraction Versus the Richness of Being*. The University of Chicago Press. p. 217.

- Allen Kolleginnen und Kollegen am IMBE möchte ich für so manche fachliche Diskussion, aber auch für die nötigen Ablenkungen bei einer Tasse Kaffee, bei einer Partie Kicker, im Theater oder bei sportlichen Aktivitäten danken. Sie haben nicht zuletzt zu einer angenehmen Arbeitsatmosphäre und somit zum Gelingen dieser Arbeit beigetragen. Insbesondere möchte ich Andreas Mayr für überaus konstruktive Verbesserungsvorschläge zu dieser Arbeit und der guten Zusammenarbeit in der Arbeitsgruppe Computationale Biostatistik danken.
- Mein besonderer Dank gilt Amelie Elsäßer, für das kritische Lesen der Arbeit und ihre immerwährende Unterstützung während der letzten Jahre.

Abstract

Variable selection and model choice are of major concern in many statistical applications, especially in regression models for high-dimensional data. Boosting is a convenient statistical method that combines model fitting with intrinsic model selection. We investigate the impact of base-learner specification on the performance of boosting as a model selection procedure. We show that variable selection may be biased if the base-learners have different degrees of flexibility, both for categorical covariates and for smooth effects of continuous covariates. We investigate these problems from a theoretical perspective and suggest a framework for unbiased model selection based on a general class of penalized least squares base-learners. Making all base-learners comparable in terms of their degrees of freedom strongly reduces the selection bias observed with naive boosting specifications. Furthermore, the definition of degrees of freedom that is used in the smoothing literature is questionable in the context of boosting, and an alternative definition is theoretically derived. The importance of unbiased model selection is demonstrated in simulations and in an application to forest health models.

A second aspect of this thesis is the expansion of the boosting algorithm to new estimation problems: by using constraint base-learners, monotonicity constrained effect estimates can be seamlessly incorporated in the existing boosting framework. This holds for both, smooth effects and ordinal variables. Furthermore, cyclic restrictions can be integrated in the model for smooth effects of continuous covariates. In particular in time-series models, cyclic constraints play an important role. Monotonic and cyclic constraints of smooth effects can, in addition, be extended to smooth, bivariate function estimates. If the true effects are monotonic or cyclic, simulation studies show that constrained estimates are superior to unconstrained estimates. In three case studies (the modeling the presence of Red Kite in Bavaria, the modeling of activity profiles for Roe Deer, and the modeling of deaths caused by air pollution in São Paulo) it is shown that both constraints can be integrated in the boosting framework and that they are easy to use.

All described results were included in the R add-on package **mboost**.

Zusammenfassung

Insbesondere in Regressionsmodellen für hochdimensionale Daten kommt der Variablenselektion und der Modellwahl eine herausragende Bedeutung zu. Boosting-Verfahren bieten die Möglichkeit die Modellanpassung mit intrinsischer Modellwahl zu kombinieren. In dieser Arbeit wird der Einfluss der Spezifikation der Base-learner auf die Modellwahl untersucht. Es zeigt sich, dass sowohl für kategoriale Einflussvariablen als auch für glatte Effekte stetiger Einflussgrößen Baselearner mit höheren Freiheitsgraden bevorzugt werden. Um diese Verzerrung zu reduzieren oder gar zu vermeiden müssen die Freiheitsgrade gleich gewählt werden. Darüber hinaus wird der in der Smoothing-Literatur vorherrschende Freiheitsgradbegriff im Kontext von Boosting in Frage gestellt und eine alternative Definition theoretisch begründet. Die hergeleiteten Resultate werden in Simulationsstudien untersucht und beispielhaft für die Modellierung von Waldschadensdaten herangezogen.

Ein weiterer Aspekt dieser Arbeit besteht in der Erweiterung des Boosting-Algorithmus auf neue Fragestellungen: Durch die Einbeziehung von Nebenbedingungen in die Schätzung der Base-learner können monotonie-restringierte Effekte nahtlos in den bestehende Rahmen integriert werden. Dies ist sowohl für glatte Effekte als auch für ordinale Variablen möglich. Darüber hinaus lassen sich zyklische Restriktionen für glatte Funktionen einer stetigen Variable in die Modellschätzung einbeziehen. Zyklische Restriktionen spielen insbesondere in der Modellierung von Zeitreihen eine wichtige Rolle. Monotonie und zyklische Effekte lassen sich darüber hinaus ebenso auf glatte, bivariate Funktionen erweitern. Beide Arten von Restriktionen stellen sich in Simulationsstudien gegenüber unrestringierten Modellen als überlegen heraus, falls in Wahrheit ein monotoner bzw. ein zyklischer Effekt vorliegt. In drei Anwendungen (der Modellierung des Vorkommens von Rotmilanen in Bayern, der Modellierung von Aktivitätsmustern beim Reh und der Modellierung der Todesfälle aufgrund von Luftverschmutzung in São Paulo) zeigt sich, dass sich die beschriebenen Restriktionen in Boosting-Modelle integrieren und einfach verwenden.

Alle beschriebenen Ergebnisse fanden Eingang in das R Paket **mboost**.

Contents

1. Introduction

> Science may be described as the art of systematic over-simplification — the art of discerning what we may with advantage omit.
>
> *(Sir Karl Raimund Popper[1])*

The methodological advances in statistical modeling during the last two decades make it possible to determine relationships in complex, high-dimensional data sets that are hard to handle by using classical methods such as linear models with stepwise variable selection. Especially ideas from computer science and machine learning had — and still have — a big impact on developments in this area. As Breiman (2001b) argues, these two fields have developed very important ideas and algorithms that should greatly influence the way statisticians think. Perhaps the three most influential approaches that emerged in the field of machine learning are support vector machines (Vapnik 1995), boosting (Freund and Schapire 1996) and random forests (Breiman 2001a). While random forests are a rather simple, yet very powerful non-parametric approach to regression modeling, support vector machines and boosting might rather be seen as 'meta-algorithms' that provide rich frameworks and can be used to derive specialized solutions. Typically, these machine learning approaches are complex methods. However, statisticians and related scientists should see the great benefits arising from models derived with these approaches. As Breiman argues, these new approaches build a rich basis for the derivation of more accurate models in small data sets, and they can be used to derive models for high-dimensional data sets as well.

[1] Popper K (1988). *The open universe: An argument for indeterminism*. Routledge, London. Edited by W.W. Bartley, III. p. 44.

1.1. Statistical Modeling and Machine Learning – Two Worlds?

In his thought-provoking paper, Breiman (2001b) advocates to use tools from the field of machine learning, which he calls "algorithmic models", and to focus on the prediction accuracy of models, rather than using what he calls "data models". In his eyes, statisticians tended (or still tend) to focus too much on statistical models. Breiman (2001b, p. 199) states that the research focus on data models has "led to irrelevant theory and questionable scientific conclusions". He proposes to focus on prediction accuracy and algorithmic properties such as convergence or the reasons for the good prediction properties of algorithms. No doubt, this focus is relevant in many machine learning problems, where huge data sets are used to make superb predictions on new data. However, statisticians and practitioners typically are also interested in making inference about the unknown regression relationship. In many fields, such as medicine, researchers do not purely ask for prediction machines but want to draw conclusions on the quality of a therapy in subgroups of patients, or to understand the mechanisms of diseases. Algorithmic models derived from random forests, support vector machines, or boosting, however, do not necessarily provide measures that statisticians are used to interpret and that allow to draw conclusions (i.e., make statistical inference). For random forests and tree-based gradient boosting, variable importance measures were introduced (Breiman 2001a, Friedman 2001, Strobl, Boulesteix, Zeileis, and Hothorn 2007) that allow to pick 'important' variables from a potentially large number of covariates. For boosting, the link between algorithmic models and data models was created by the groundbreaking papers of Friedman, Hastie, and Tibshirani (2000), and Bühlmann and Yu (2003), who interpreted functional gradient descent boosting as an optimization algorithm that fits forward stagewise additive models. The resulting model can be interpreted as a generalized additive model. Consequently, statisticians can interpret these models based on regression coefficients (linear model), or by looking at the partial contributions of each model component (additive model). Hence, with boosting, we have an algorithmic model with good algorithmic properties (as we will discuss in the next chapter), which is at the same time an interpretable statistical data model. Thus, the dichotomy that Breiman exaggeratedly devised can be bridged to a certain extend. One should mention that the idea to use statistical models and to consider their algorithmic properties is also acknowledged by

Breiman (2001b). Two approaches, which are also considered in this work, can be seen as a way to combine algorithmic and statistical models. According to Breiman (2001b), Wahba's (1990) research on smoothing splines, embedded in the theory of reproducing kernel Hilbert spaces, is one good example of a new research focus in the statistics community. In Section 2.3.4, thin plate splines (a generalization of cubic smoothing splines) and general radial basis functions are introduced; both can be theoretically derived from reproducing kernel Hilbert spaces (see, e.g., Anjyo and Lewis 2011). Breiman (2001b) also mentions the (at this time new) boosting algorithm as a possible method to bridge the gap between statistics and machine learning. Lots of research has been conducted since then to understand the statistical properties of this machine learning procedure (see Chapter 2). Nowadays, boosting algorithms provide a versatile framework to derive models from, which are well interpretable in many cases.

1.2. Scope of this Work

In this thesis we investigate ways to model complex data situations using boosting approaches. We aim to fit models to complex data sets that comprise spatial components, and which might require spatio-temporal components or spatially varying coefficients. In these complex settings we try to find a relatively sparse and interpretable model. For this purpose, variable and model selection are required. With different degrees of complexity and flexibility of the competing model terms, unbiased or bias reduced selection of the modeling alternatives becomes an absolute necessity. In Chapter 3, we aim to predict the health status of beeches in a northern Bavarian forest district. The model should be interpretable and useful, thus, unbiased model selection is demanded. The data set consists of continuous, ordinal and binary covariates that describe the environmental conditions at the location of the trees. Furthermore, the locations should be used to model additional spatial variation.

Prior demands on the effects, such as monotonicity or cyclicity are another major topic of this thesis. Constrained effect estimation allows to include these requirements in the estimation process. In Chapter 4, we face these challenges. In a first case study, we model the species distribution of Red Kite in Bavaria. In this model we have spatial information of the Red Kite sightings as well as many environmental variables of different nature. The aim is to find an interpretable model for the

conditional breeding probability of Red Kite. As some effects become biologically implausible, monotonic constraints are incorporated in the modeling process.

The second study investigates the activity profiles of Roe Deer. Temporal information is given along with environmental variables and the activity measurement of the observed Roe Deer. The activity of the animals changes during the day and throughout the year. Thus, we try to find a smooth interaction surface that describes the activity during the day and within the year. In order to allow smooth transitions at the boundaries (end of the day and end of the year) periodic constraints should be considered.

The third case study investigating the impact of SO_2 on the number of respiratory deaths, combines periodic effects with monotonicity constrained effects. We want to model long term trends and periodic trends of mortality due to respiratory causes within the year. At the same time, increasing concentrations of the pollutant of interest, i.e., SO_2, should result in an increased expected number of deaths.

In summary, in this thesis some important challenges in the context of complex data sets are investigated. First, spatial models are considered. Second, we investigate a framework for unbiased selection of competing variables and competing modeling alternatives. Third, possibilities for the inclusion of subject matter knowledge via monotonicity constraints or cyclic effects are demonstrated. All these topics have in common that the resulting model should be easily accessible and interpretable while allowing enough flexibility to be reliable and generalizable.

As spatial and complex data, which requires further restrictions, is very common in the field of ecology, all data sets considered in this thesis arose — more or less — from this scientific area. However, medical applications are also conceivable. Spatial data might for example result from large clinical trials where the spread of diseases is investigated. An example are cancer studies with data from population-based cancer registries. In this context it is also of interest to model temporal developments and periodic patterns. Hence, periodic constraints are also of interest. The link between ecological and medical applications is given in the São Paulo air pollution study, where the influence of ecological parameters on human health is considered.

1.3. Structured Additive Models

All models discussed in this thesis can be regarded as structured additive models. Let us consider a set of observations $(y_i, \boldsymbol{x}_i^\top), i = 1, \ldots, n$, where y_i is the response variable and $\boldsymbol{x}_i = (x_i^{(1)}, \ldots, x_i^{(p)})$ consists of p possible predictors of different nature, such as categorical or continuous covariates. To model the dependency of the response on the predictor variables, we consider a structured regression model, where $\mathbb{E}(y|\boldsymbol{x}) = h(\eta(\boldsymbol{x}))$ with (known) response function h, and an *structured additive predictor* $\eta(\boldsymbol{x})$ of the form

$$\eta(\boldsymbol{x}) = \beta_0 + \sum_{j=1}^{p} f_j(x^{(j)}) \tag{1.1}$$

is used. The functions $f_j(\cdot)$ are generic representations for modeling alternatives such as linear effects ($f_j(x^{(j)}) = x^{(j)}\beta$, where $x^{(j)}$ is one of the predictors), categorical effects ($f_j(x^{(j)}) = \mathbf{z}^\top \boldsymbol{\beta}$, where $\mathbf{z}$ results from dummy or effect coding of a categorical covariate $x^{(j)}$) and smooth effects ($f_j(x^{(j)}) = f_{j,\text{smooth}}(x^{(j)})$, where $x^{(j)}$ is one of the continuous predictors). Here, we assume that each covariate has exactly one modeling alternative. However, more complex models with multiple modeling alternatives or functions that depend on multiple variables are possible as well. Hence, other modeling alternatives such as spatial and random effects can also be expressed in this framework (see Fahrmeir, Kneib, and Lang 2004, for details). Generalized additive models (GAMs) as introduced by Hastie and Tibshirani (1986, 1990) appear as an important special case of (1.1) if all generic functions represent smooth effects. Having the model formulation at hand, two challenges arise: First, a method for model fitting in this flexible framework is needed. Second, the question which covariates should enter the model, and how these covariates should be modeled, needs to be answered. Both issues can be addressed in one framework by applying a component-wise boosting approach.

1.4. Transferring Machine Learning Tools to the "Real World"

Thinking of functional gradient descent boosting (see Chapter 2) the resulting models are in many cases illustrative and relatively easy to interpret. The machinery in the background, i.e., the boosting algorithm, might be irrelevant for the inter-

pretation of the final model. On the other hand, some of the results need to be seen in the light of the algorithm. For example, the algorithm internally performs variable selection. This can also be used for model selection, i.e., boosting is able to determine which modeling alternative is more appropriate for a covariate. As we will show in Chapter 3, some care needs to be taken when model terms with very different degrees of flexibility are specified or if the predictors are of different nature such as continuous and categorical variables. Another point is that one should keep in mind that the results arise from a regularized model, hence the effect sizes are shrunken toward zero. Furthermore, it is helpful to know how these models can be optimized or tuned, for example, by the use of cross-validation methods. Finally, one should be aware that in regularized models naturally no tests and p-values are available. However, with some further effort, resampling-based confidence bands or selection of variables with a built in error control (i.e., stability selection; cf. Meinshausen and Bühlmann 2010, and Section 2.5) can be derived. Thus, some notion of the model uncertainty is available.

In this light, statisticians need to find ways to 'sell' models derived with machine learning tools to scientists in fields where these models are of interest. The resulting models, for example models derived with boosting, are relatively easy to interpret and can be tuned to have a good prediction accuracy. Thinking of the data sets resulting from 'omics' (e.g., genomics, proteomics, or metabolomics) or other high dimensional data sets, which can be easily collected due to the general availability of sensors, GPS-trackers and computers, regularized, computer-intensive fitting methods are without alternatives. Statisticians and practitioners need to get used to these computational methods. For the interpretation of the model one does not need to understand every fiddly detail of the algorithms. However, the derivation of the model should be the responsibility of researchers with a thorough understanding of the statistics and algorithms involved. This task can be simplified but *not automated* by readily available software tools, which guide the researcher by well defined defaults (which build a good starting point but need to be carefully reconsidered in a second step). These tools may assist researchers to (easily) gain knowledge from their data. We try to meet this demands by providing a well developed and versatile add-on package to the statistical computing environment R (R Development Core Team 2011) called **mboost** (Hothorn, Bühlmann, Kneib, Schmid, and Hofner 2011a). Details of the implementation and examples are given in Chapter 2 and in Appendix B.

Contributions

The work presented in this thesis results in parts from different collaborations of mine and were influenced by various discussions with my colleagues. Most notably, Chapter 3 is mainly based on the paper of Hofner, Hothorn, Kneib, and Schmid (2011a). Parts of Chapter 4 presenting some general ideas for monotonicity constrained regression and the application of monotonic constraints to species distribution models were published in Hofner, Müller, and Hothorn (2011c). Furthermore, an extended version of the simulations presented there is part of this thesis. In Section 2.6, ideas and work from a manuscript on boosting GAMLSS models (Mayr, Fenske, Hofner, Kneib, and Schmid 2011b) are presented in a condensed fashion. Another basis of this thesis was the R add-on package **mboost** (Hothorn *et al.* 2011a, Hothorn, Bühlmann, Kneib, Schmid, and Hofner 2011b) and the associated paper by Hothorn, Bühlmann, Kneib, Schmid, and Hofner (2010).

2. An Introduction to Boosting

> Boosting is the best off-the-shelf classifier in the world.
>
> *(Attributed to Leo Breiman [1])*

Leo Breiman's often cited statement that "boosting is the best off-the-shelf classifier in the world" (Hastie, Tibshirani, and Friedman 2001, p. 302) refers to the very beginnings of boosting: the development of AdaBoost (Freund and Schapire 1996, 1997). Boosting was originally designed for classification. It uses many weak classifiers such as trees or stumps (trees with one split and two terminal nodes) and *boosts* their performance by combining them to a strong 'ensemble'. Improvements of the classifiers are most prominent in situations with many features and few observations, such as data mining (Hastie *et al.* 2001, p. 302).

In this thesis the focus will be on the "statistical view of boosting" (Friedman *et al.* 2000). This allows to formulate and estimate (regression) models in the context of boosting. The concept of boosting is expanded from classification to regression models in general, and new classes of weak learners (or base-learners as they will be called in this thesis) are introduced . Thus, strictly speaking, Breiman's statement does not apply anymore. Still, boosting has proven to be a strong competitor and an adaptable method suited for many estimation problems.

Before going into detail, a short overview of the development of boosting from a classification tool in the machine learning community to a widely applicable method in the statistical context is given.

2.1. A (Short) History of Boosting

The foundations of boosting were first laid down by a paper on the strength of weak learnability (Schapire 1990). Later, Freund and Schapire developed the now famous AdaBoost algorithm for classification (Freund and Schapire 1996, 1997).

[1] see Hastie T, Tibshirani R, Friedman J (2001). *The elements of statistical learning: Data mining, inference, and prediction*. Springer, New York. p. 302.

AdaBoost gains its strength from using ensembles of weak learners (i.e., boosting the performance of the weak learner) and from adaptively choosing observation weights based on the classification of the previous iteration. One adaptively boosts the weak learners — hence the name of the algorithm. In each step, the weights of correctly classified observations are decreased, while weights are increased for misclassified observations. This results in an algorithm that focuses on the "difficult" observations rather than the "easy" ones (see,e.g., Ridgeway 1999). The fact that the algorithm tries to fit all observations and not only the easy ones gives some idea on why AdaBoost performs so well.

For statisticians the problem of this notion was that it is not linked to the way statisticians are used to learn from data. Statisticians tend to fit models by least squares minimization, by likelihood maximization or by similar methods. Breiman (1998, 1999) established the link between AdaBoost and statistical learning by showing that AdaBoost can be seen as an optimization algorithm in function space. A further step to the break through of boosting in the statistical community was achieved by Friedman *et al.* (2000) who showed that AdaBoost is equivalent to a "forward stagewise additive model" (additive does not mean that the model is additive in the covariates but is formed by an additive combination of weak learners). From this point onward, boosting rapidly evolved from a classification algorithm to a versatile modeling algorithm based on the view of boosting as a functional gradient descent (FGD) optimization algorithm. This 'view' is sketched in the next section in more detail.

An overview of the development of boosting and first steps to regression modeling by boosting methods can be found in Ridgeway's (1999) paper on the "state of boosting". Later, Bühlmann and Yu (2003) introduced *component-wise* functional gradient descent boosting: each base-learner depends only on a subset of the possible predictors and in each boosting iteration only the best-fitting base-learner is updated. This leads to an intrinsic selection of base-learners and thus variable selection. An overview of recent directions and developments of boosting in the statistical community is given in Bühlmann and Hothorn (2007). The following section gives a detailed overview.

2.2. Functional Gradient Descent Boosting

As Friedman *et al.* (2000) and Friedman (2001) state, many learning techniques (or modeling approaches as statisticians call them) aim at minimizing the *expected loss* function

$$\eta^*(x) = \arg\min_{\eta} \mathbb{E}_{Y,X}\Big(\rho(y, \eta(x))\Big),$$

where $\eta(x)$ a function that is to be optimized, i.e., a function that is fitted to the outcome y, as statisticians would call it. The loss function ρ is typically assumed to be differentiable and convex with respect to the second argument. Well known examples for the loss function are the squared error loss $\rho(y, \eta(x)) = (y - \eta(x))^2$, which is minimized by least squares estimation or loss functions based on the (negative) log-likelihood.

Friedman *et al.* (2000) and Friedman (2001) developed a general framework for functional gradient descent boosting. Let the observations be $(y_i, \mathbf{x}_i^\top)$, $i = 1, \ldots, n$, where y_i is a scalar response and the possible predictors are denoted by $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})^\top$. Estimates of the minimizer $\eta^*(x)$ can be found by minimizing the *empirical risk*

$$\mathcal{R} = \sum_{i=1}^{n} \rho(y_i, \eta(\mathbf{x}_i^\top)) \tag{2.1}$$

instead of the expected loss. Let $\eta_i := \eta(\mathbf{x}_i^\top)$ and consider $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_n)^\top$ as an n-dimensional parameter vector. Then, one aims at finding the optimal parameter vector

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \sum_{i=1}^{n} \rho(y_i, \eta_i).$$

One way to solve this minimization problem is to use a steepest descent algorithm, where $\hat{\boldsymbol{\eta}}$ is found by an iterative procedure: In step $m = 0$ we start by initializing $\hat{\boldsymbol{\eta}} = \hat{\boldsymbol{\eta}}^{[0]}$. In step m, updates are computed based on the negative gradient of $\rho(y, \eta)$ evaluated at the current estimate $\hat{\boldsymbol{\eta}}^{[m-1]}$:

$$u_i^{[m]} = -\left.\frac{\partial \rho(y_i, \eta_i)}{\partial \eta_i}\right|_{\eta_i = \hat{\eta}_i^{[m-1]}}, \qquad i = 1, \ldots, n.$$

With the negative gradient vector $\mathbf{u}^{[m]} = (u_1^{[m]}, \ldots, u_n^{[m]})^\top$, the updated solution becomes

$$\hat{\boldsymbol{\eta}}^{[m]} = \hat{\boldsymbol{\eta}}^{[m-1]} + \nu^{[m]} \mathbf{u}^{[m]}, \tag{2.2}$$

with step-length

$$\nu^{[m]} = \arg\min_{\nu} \rho(\mathbf{y}, \hat{\boldsymbol{\eta}}^{[m-1]} - \nu \mathbf{u}^{[m]}). \tag{2.3}$$

So far, no covariate information or special functional form is considered for $\boldsymbol{\eta}$. Constraining the negative gradient $\mathbf{u}^{[m]}$ to a class of parameterized functions of the covariates — the so-called base-learner — leads to functional gradient descent boosting as noted and elaborated by Breiman (1999), Friedman *et al.* (2000), and Friedman (2001) amongst others. The constraint is not directly applied but it is introduced by relating the negative gradient to the base-learners by (penalized) least squares estimation. The negative gradient is replaced by a parameterized estimate in Equation (2.2). The resulting generic functional gradient descent algorithm (Friedman 2001) is given in Algorithm 1. The negative gradient in step (a) is the best steepest descent direction in the n-dimensional data space at point $\hat{\eta}^{[m-1]}(\mathbf{x})$ (Friedman 2001). The estimate in step (b) is a constrained approximation of this steepest descent direction. As one can see from the update step (c), the final estimate $\hat{\eta}^{[m]}$ is additive in the base-learners, as claimed earlier. Note that in Friedman's (2001) original algorithm the step-length factor ν is optimized using a line search (Eq. 2.3). In Section 2.4.1, we discuss why the additional line search is usually not necessary.

2.2.1. Component-wise FGD Boosting

Based on the functional gradient descent (FGD) boosting algorithm, Bühlmann and Yu (2003) developed a *component-wise* boosting algorithm (see Algorithm 2). The main difference is that they use n_{BL} base-learners instead of one single base-learner as in Algorithm 1. Each of the base-learners typically depends only on a subset of the covariates. A simple and common example is that for each of the covariates a separate *linear* base-learner is specified — i.e., a simple linear model regressing $\mathbf{u}^{[m]}$ only on the jth covariate $\mathbf{x}^{(j)}$. In this case the number of base-learners n_{BL} is equal to the number of covariates p.

One could also consider to specify *smooth* base-learners or decompose smooth effects into parametric parts for the null space and smooth deviations from the parametric parts (Kneib, Hothorn, and Tutz 2009). With the decomposition, we get two base-learners per covariate[2] (a linear base-learner and a smooth base-learner that captures the deviation from linearity). Usually, the base-learners correspond to the modeling alternatives as expressed by the generic functions f_j in the struc-

[2] in the case of P-splines with a second order difference penalty; see Equation (3.9)

Algorithm 1 Generic functional gradient descent algorithm

Initialize Set the iteration index $m := 0$, initialize the estimate with an offset value, typically $\hat{\eta}(\cdot) \equiv 0$ or $\hat{\eta}(\cdot) \equiv \arg\min_c n^{-1} \sum_{i=1}^{n} \rho(y_i, c)$.

Iterate

(a) **Negative gradient** Increase m by 1. Compute the negative gradient of the loss function $\rho(\mathbf{y}_i, \eta)$ evaluated at the function values of the previous iteration $\hat{\eta}^{[m-1]}(\mathbf{x}_i^\top)$:

$$u_i^{[m]} = - \left. \frac{\partial \rho(y_i, \eta)}{\partial \eta} \right|_{\eta = \hat{\eta}^{[m-1]}(\mathbf{x}_i^\top)}, \qquad i = 1, \ldots, n.$$

(b) **Estimation** Relate the negative gradient vector $\mathbf{u}^{[m]} = (u_1^{[m]}, \ldots, u_n^{[m]})^\top$ to the observed values $\mathbf{x}_1^\top, \ldots, \mathbf{x}_n^\top$ of covariates by fitting $\mathbf{u}^{[m]}$ by a real-values base-learner $g(\cdot)$ of the covariates. Hence, $\hat{\mathbf{g}}^{[m]} = (\hat{g}^{[m]}(\mathbf{x}_1^\top), \ldots, \hat{g}^{[m]}(\mathbf{x}_n^\top))^\top$ is a constrained estimate of $\mathbf{u}^{[m]}$.

(c) **Update** Compute the update

$$\hat{\eta}^{[m]}(\cdot) = \hat{\eta}^{[m-1]}(\cdot) + \nu \hat{g}^{[m]}(\cdot)$$

with step-length factor $0 < \nu \leq 1$.

Stop Stop if $m = m_{\text{stop}}$ for a given stopping iteration m_{stop}.

tured additive predictor (Eq. 1.1). Details on possible base-learners are given in Section 2.3.

The transition from FGD boosting to component-wise FGD boosting allows model fitting with intrinsic variable selection. As we select only *one* base-learner (= modeling alternative) in each boosting iteration (see Alg. 2, step (b2)), variable selection is achieved by stopping the boosting procedure after an appropriate number of iterations $m_{\text{stop,opt}}$. In settings with many covariates, some base-learners and thus some covariates are never selected. At maximum $m_{\text{stop,opt}}$ covariates are chosen in the course of the boosting algorithm, but typically a lot less variables enter the model. Obviously, the base-learner selection (Eq. 2.5) is the crucial part for variable and model selection. The intrinsic base-learner selection can be further employed for model selection. While for variable selection, one specifies one base-learner for each covariate, model selection is incorporated by additionally specifying base-learners for different, competing modeling alternatives. An example is

Algorithm 2 *Component-wise* generic functional gradient descent algorithm

Initialize Set the iteration index $m := 0$, and initialize the function estimates and the additive predictor with offset values, typically $\hat{f}_j^{[0]}(\cdot) \equiv 0,\ j = 1, \ldots, n_{\text{BL}}$, and $\hat{\eta}^{[0]}(\cdot) \equiv \arg\min_c \frac{1}{n} \sum_{i=1}^n \rho(y_i, c)$.

Iterate

(a) **Negative gradient** Increase m by 1. Compute the negative gradient of the loss function $\rho(\mathbf{y}_i, f)$ evaluated at the function values of the previous iteration $\hat{\eta}^{[m-1]}(\mathbf{x}_i^\top)$:

$$u_i^{[m]} = -\left.\frac{\partial \rho(y_i, \eta)}{\partial \eta}\right|_{\eta = \eta^{[m-1]}(\mathbf{x}_i^\top)}, \qquad i = 1, \ldots, n. \tag{2.4}$$

(b) **Estimation**

(b1) Relate the negative gradient vector $\mathbf{u}^{[m]} = (u_1^{[m]}, \ldots, u_n^{[m]})^\top$ to the observed values $\mathbf{x}_1^\top, \ldots, \mathbf{x}_n^\top$ of covariates by separately fitting $\mathbf{u}^{[m]}$ by real-values base-learner $g_j(\mathbf{x}), \forall j \in \{1, \ldots, n_{\text{BL}}\}$. Each base-learner $g_j(\mathbf{x})$ typically depends only on a subset of the covariates in $\mathbf{x}$.

(b2) Select the best-fitting base-learner g_{j^*}, i.e., the base-learner that minimizes the residual sum of squares (RSS),

$$j^* = \arg\min_j \sum_{i=1}^n \left(u_i^{[m]} - \hat{g}_j^{[m]}(\mathbf{x}_i^\top)\right)^2. \tag{2.5}$$

(c) **Update** Compute the update for the additive predictor

$$\hat{\eta}^{[m]}(\cdot) = \hat{\eta}^{[m-1]}(\cdot) + \nu \hat{g}_{j^*}^{[m]}(\cdot)$$

and the function estimate

$$\hat{f}_{j^*}^{[m]}(\cdot) = \hat{f}_{j^*}^{[m-1]}(\cdot) + \nu \hat{g}_{j^*}^{[m]}(\cdot), \tag{2.6}$$

while leaving all other function estimates $\hat{f}_j$, $j \neq j^*$ unchanged. In each update step, only a fraction $0 < \nu \leq 1$ of the fitted values is added, where ν is a step-length factor in the gradient descent approach.

Stop Stop if $m = m_{\text{stop}}$ for a given stopping iteration m_{stop}.

given by the decomposition of smooth effects into a linear effect and the smooth deviation from linearity as sketched above (cf. Kneib *et al.* 2009).

L_2Boosting A special case of the generic FGD algorithm is L_2Boosting as investigated by Bühlmann and Yu (2003), where the rescaled L_2-loss

$$\rho(y_i, \eta(\mathbf{x}_i)) = \frac{1}{2}(y_i - \eta_i)^2$$

is used. The factor $1/2$ is used so that the negative gradient in step m becomes

$$u_i^{[m]} = y_i - \eta_i^{[m-1]},$$

thus, the negative gradient vector $\mathbf{u}^{[m]}$ is just a vector of residuals. Hence, L_2Boosting can be described as refitting of residuals. The idea of refitting residual has gained early attention with two refits in the twicing algorithm (Tukey 1977). From this point of view the strength of boosting in *regression* settings becomes a bit clearer: similar to the AdaBoost algorithm, L_2Boosting especially concentrates on the 'difficult' observations, which are not yet fitted good enough.

Other Loss Functions In addition to AdaBoost and L_2Boosting, one can use a huge variety of other loss functions that reflect the estimation problem at hand. A popular choice is the negative log-likelihood, which is arising from the regression problem: for example, the negative Binomial log-likelihood for binary classification, or the negative Poisson log-likelihood for count data. Other problems require special loss functions that are not derived from likelihoods. Robust estimation can be achieved, for example, by using the L_1-loss or the Huber-loss (see Bühlmann and Hothorn 2007, for details on these loss functions). Loss functions tailored to the estimation of survival models include the negative Cox partial log-likelihood (Ridgeway 1999) or negative log-likelihoods arising from accelerated failure time models (Schmid and Hothorn 2008b). Survival models can also be fitted in the boosting framework using weighted regression with weights according to the inverse probability of censoring (see Hothorn, Bühlmann, Dudoit, Molinaro, and van der Laan 2006a, Bühlmann and Hothorn 2007). Proportional odds models are introduced in Schmid, Hothorn, Maloney, Weller, and Potapov (2010a). Further modeling alternatives include quantile (Fenske, Kneib, and Hothorn 2011) or expectile regression (Sobotka and Kneib 2010). Furthermore, with some modifications, the boosting

framework can be used to estimate generalized additive model for location, scale and shape (GAMLSS; Mayr *et al.* 2011b, Mayr, Fenske, Hofner, Kneib, and Schmid 2011a). A short introduction to GAMLSS models and the corresponding boosting approach is given in Section 2.6.

2.3. Base-learners

While the loss function defines the stochastic component of the model (distributional assumption), base-learners reflect the structural assumption (cf. structured additive predictor; Sec. 1.3). By choosing the base-learners, the available set of modeling alternatives is fixed. All base-learners $\hat{g}_j(\mathbf{x})$ that are considered in this thesis are estimated by (penalized) least squares. Hence, they can be estimated by solving the penalized least squares criterion

$$\mathcal{Q}(\boldsymbol{\beta}) = (\mathbf{u} - \mathbf{X}\boldsymbol{\beta})^\top (\mathbf{u} - \mathbf{X}\boldsymbol{\beta}) + \lambda \mathcal{J}(\boldsymbol{\beta}), \tag{2.7}$$

where $\mathcal{J}(\boldsymbol{\beta})$ is a quadratic penalty of the form

$$\mathcal{J}(\boldsymbol{\beta}) = \boldsymbol{\beta}^\top \mathbf{K} \boldsymbol{\beta}.$$

The response $\mathbf{u}$ is defined by the negative gradient (Eq. 2.4). The design matrix for $\mathbf{x}$ is denoted as $\mathbf{X}$, λ is the smoothing parameter and $\mathbf{K}$ is a suitable penalty matrix. The parameter vector is given as $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^\top$.

The solution to Equation (2.7) can be expressed as a *penalized linear model* of the form

$$\begin{aligned} \hat{g}_j(\mathbf{x}) &= \mathbf{X}\hat{\boldsymbol{\beta}} \\ &= \mathbf{X}(\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{K})^{-1} \mathbf{X}^\top \mathbf{u}, \end{aligned} \tag{2.8}$$

where $\hat{\boldsymbol{\beta}}$ is a (penalized) estimate of the coefficient vector $\boldsymbol{\beta}$. It is important to note that the term 'linear model', does not refer to models that contain only linear effects of the variable $\mathbf{x}$. In contrary, the resulting model might be highly non-linear in $\mathbf{x}$, that is, $\hat{g}_j(\mathbf{x})$ might comprise smooth functions of $\mathbf{x}$. The model (2.8) is only linear in the design matrix $\mathbf{X}$ as one can see from the first line of Equation (2.8). Examples for non-linear base-learners (of $\mathbf{x}$) comprise P-splines or radial basis functions, which are both introduced in the subsequent sections.

The smoothing parameter λ in Equations (2.7) and (2.8) governs the amount of

penalization. Higher values of λ result in a stronger impact of the penalty, while unpenalized least squares base-learners emerge as a special case when $\lambda = 0$. In Chapter 3 we show the usefulness of penalized linear base-learners in the setting of unbiased or at least bias reduced variable selection.

From Equation (2.8) one can deduce that the final boosting estimate for the modeling alternative f_j can be derived as

$$\hat{f}_j^{[m_{\text{stop}}]} = \sum_{m=1}^{m_{\text{stop}}} \nu \cdot \hat{g}_j^{[m]},$$

where $\hat{g}_j^{[m]} \equiv 0$ if the jth base-learner was not selected in iteration m. This means, f_j is estimated as the weighted sum over $\hat{g}_j^{[m]}$ for all iterations m where the jth base-learner was selected. Due to this fact and due to the linear nature of each single base-learner, we can also define the parameter estimates of the jth base-learner in the m_{stop}th iteration as the sum

$$\hat{\boldsymbol{\beta}}_j^{[m_{\text{stop}}]} = \sum_{m=1}^{m_{\text{stop}}} \nu \cdot \hat{\boldsymbol{\beta}}_{j,u}^{[m]},$$

where $\hat{\boldsymbol{\beta}}_{j,u}^{[m]} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{K})^{-1} \mathbf{X}^\top \mathbf{u}^{[m]}$ is the parameter estimate of the jth base-learner if the base-learner was selected in step m, and $\hat{\boldsymbol{\beta}}_{j,u}^{[m]} = \mathbf{0}$ otherwise.

In the following sections, an introduction to some of the possible base-learners is given. A special focus is set to base-learners used or developed in later sections. In the latter case details on these base-learners are derived in later sections.

2.3.1. Linear Base-learners

Linear base-learners can be used to express a linear effect of a continuous covariate x. In this case, the design matrix $\mathbf{X}$ in Equation (2.8) is just the covariate vector and possibly comprises an additional vector of ones for the intercept. The ith row of the design matrix is then defined as $\mathbf{X}_i = (1, x_i)$. At the same time, a group of covariates can be used within one linear base-learner, say $\mathbf{x}_i = (x_i^{(1)}, \ldots, x_i^{(\tilde{p})})^\top$, where $\tilde{p}$ is smaller or equal than the number of covariates p. In this case, the design matrix has one column per covariate. Again one can specify an additional intercept. Now the ith row reads $\mathbf{X}_i = (1, \mathbf{x}_i^\top)$. The covariates are then handled as one group, which means that all covariates enter the model if the base-learner is chosen. If the base-learner is never selected, none of its covariates enters the model.

In the case of categorical variables, the design matrix reflects the dummy-coding with given contrasts. For a categorical covariate $\mathbf{x}$ with n_{cat} categories, let the ith row of the dummy-coded design matrix be $\mathbf{X}_i = (1, x_i^{(2)}, \ldots, x_i^{(n_{\text{cat}})}),\ i = 1, \ldots, n$, where $x_i^{(k)} = 1$ if $x_i = k$, and $x_i^{(k)} = 0$ otherwise. Other codings are possible but are not considered in detail in this thesis.

Ridge Penalized Base-Learners Often, an unpenalized linear base-learner is used. However, it is possible to extend linear base-learners to penalized linear base-learners. In this case a ridge penalty can be introduced (Hoerl and Kennard 1970). Then the penalty matrix becomes $\mathbf{K} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix of appropriate dimensions. With the ridge penalty all coefficients are evenly shrunken toward zero. This is especially attractive in the case of dummy-coding for an unordered categorical covariate or a group of variables. Shrinkage then allows to make base-learners comparable with respect to the flexibility they offer (cf. Chapter 3). If we have variables with many categories or group many variables in one base-learner, the base-learner gets stronger, i.e., it can carry more information and is more flexible. Regularization, e.g., via ridge regression, introduces a method to obtain weak base-learners in this situation. This leads to a base-learner with a low variance and a potentially large bias. Many authors noted that boosting seems especially strong in reducing the bias through subsequent boosting steps (Friedman *et al.* 2000, Bühlmann and Yu 2003, Park, Lee, and Ha 2009). Hence, the base-learners should be weak in the sense of having a low variance but (potentially) a high bias ('low-variance principle'; see e.g., Bühlmann and Hothorn 2007).

Penalized Ordinal Base-Learners Other penalties, for example for ordered categorical variables, might also be useful. Hofner *et al.* (2011a) introduced a penalized base-learner for ordered factors that exploits the neighborhood information and shrinks the differences of effects for adjacent categories. This seems reasonable as it is often the case that an ordering of the covariate's categories converts to a similar ordering of the corresponding effects and this additional information can be incorporated to enforce stable estimation. Therefore, we consider a ridge-type penalty for the *differences* of adjacent parameters that favors smooth coefficient sequences similar to P-splines (see below). A slight difference arises from the fact that the effect of the reference category is restricted to zero. We will use (without loss of generality) the restriction $\beta_1 = 0$ and use a base-learner without intercept. Hence,

the penalty is given by

$$\mathcal{J}(\boldsymbol{\beta}; d) = \sum_{j=d+1}^{n_{\text{cat}}} (\Delta^d \beta_j)^2, \tag{2.9}$$

where $\boldsymbol{\beta} = (\beta_2, \ldots, \beta_{n_{\text{cat}}})^\top$ is the vector of dummy-coded effects, n_{cat} refers to the number of categories of the covariate, and $\beta_1 = 0$. The difference operator Δ^d can be defined recursively as

$$\begin{aligned} \Delta^1 \beta_j &= \Delta \beta_j = \beta_j - \beta_{j-1}, \\ \Delta^2 \beta_j &= \Delta(\Delta \beta_j) = \Delta(\beta_j - \beta_{j-1}) = \Delta \beta_j - \Delta \beta_{j-1} = \beta_j - 2\beta_{j-1} + \beta_{j-2}, \end{aligned} \tag{2.10}$$

and so on. Usually, for penalized ordinal base-learners, differences of order one are used. In matrix notation the penalty can be written as $\mathcal{J}(\boldsymbol{\beta}; d) = \boldsymbol{\beta}^\top \mathbf{K} \boldsymbol{\beta}$, and penalty matrix $\mathbf{K} = \mathbf{D}_{(d)}^\top \mathbf{D}_{(d)}$, with a first order difference matrix of the form

$$\mathbf{D}_{(1)} = \begin{pmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{pmatrix}, \tag{2.11}$$

where empty cells are equal to zero. For more details see also Gertheiss and Tutz (2009) and Hofner *et al.* (2011a).

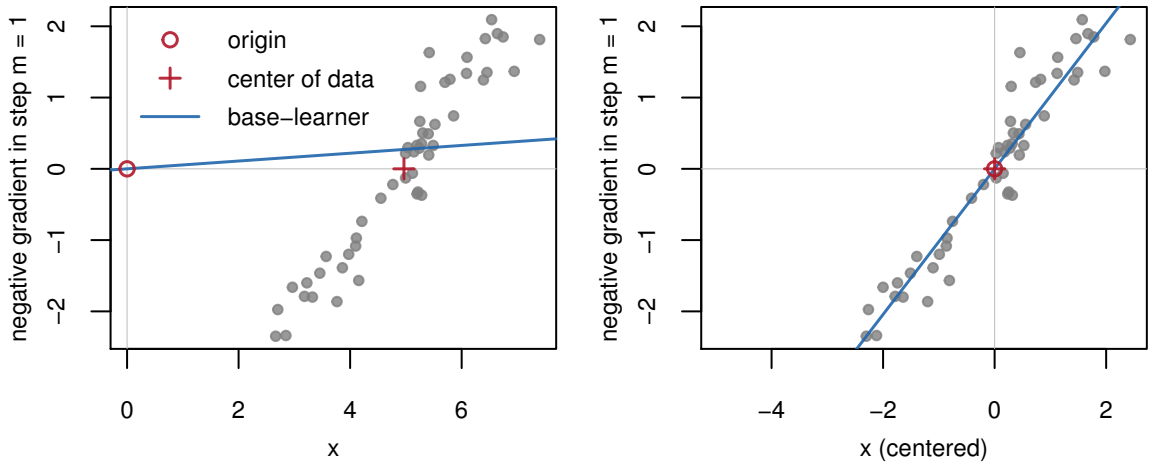
Centering of Linear Base-learners Without Intercept For linear base-learners that are specified without intercept[3], it is of great importance to center the covariates before fitting the model. Without centering of the covariates, linear effects that result from base-learners without intercept are forced through the origin (with no data lying there). Hence, the convergence will be very slow or the algorithm will not converge to the "correct" solution even in very simple cases. As an example, consider one normally distributed predictor $\mathbf{x} = (x_1, \ldots, x_n)^\top$, and a model

$$\mathbf{y} = \beta \mathbf{x} + \boldsymbol{\varepsilon},$$

with $\beta = 1$ and $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, 0.3^2)$. Usually, a model without intercept could be fitted to estimate β. However, if we apply boosting, L_2Boosting in this case, the

[3]Linear base-learners without intercept are obtained in **mboost** by specifying a base-learner `bols(x, intercept = FALSE)`. Furthermore, if the fitting function `glmboost()` is used – which fits generalized linear models by boosting – the single base-learners never contain an intercept.

negative gradient in the first boosting step is, per default, the centered response, i.e., $\mathbf{u}^{[1]} = \mathbf{y} - 1/n \sum_{i=1}^{n} y_i$. For other loss functions the negative gradient in the first boosting iteration is not exactly the mean-centered response. Yet, the negative gradient in the first step is always "centered" around zero. In this situation, the application of a base-learner without intercept (i.e., a simple linear model without intercept) is not sufficient anymore to recover the effect β (see Figure 2.1(a)). The true effect is completely missed. To solve this problem, it is sufficient to use a centered predictor $\mathbf{x}$. Then, the center of the data is shifted to the origin (see Figure 2.1(b)) and the model without intercept goes through the origin as well. The model and the data match up.



(a) Without Centering (b) With Centering

Figure 2.1.: L_2Boosting in the first boosting step, i.e., with centered response variable as outcome. A base-learner without intercept misses the true effect completely if $\mathbf{x}$ is not centered (left) and is able to capture the true structure if $\mathbf{x}$ is centered (right).

Centering the predictors does not change the estimated effects of the predictors. Yet, the intercept needs to be corrected as can be seen from the following example. Consider two predictors and estimate a model with centered predictors, i.e,

$$\begin{aligned} \hat{\mathbf{y}} &= \hat{\beta}_0 + \hat{\beta}_1(\mathbf{x}_1 - \bar{x}_1) + \hat{\beta}_2(\mathbf{x}_2 - \bar{x}_2) \qquad \Leftrightarrow \\ \hat{\mathbf{y}} &= \underbrace{(\hat{\beta}_0 - \hat{\beta}_1\bar{x}_1 - \hat{\beta}_2\bar{x}_2)}_{=\hat{\beta}_0^*} + \hat{\beta}_1\mathbf{x}_1 + \hat{\beta}_2\mathbf{x}_2. \end{aligned}$$

Hence, the intercept from a model without centering of the covariates equals $\hat{\beta}_0^* = \hat{\beta}_0 - \sum \hat{\beta}_j \bar{x}_j$, where $\hat{\beta}_0$ is the intercept estimated from a model with centered predictors.

2.3.2. P-spline Base-Learners

To model smooth effects of continuous variables, we utilize penalized B-splines (i.e., P-splines). They were introduced by Eilers and Marx (1996) for nonparametric regression and were later transferred to the boosting framework by Schmid and Hothorn (2008a). Consider observations $\mathbf{x} = (x_1, \ldots, x_n)^\top$ of a single variable x, then a non-linear function $f(x)$ can be approximated as

$$f(x) = \sum_{j=1}^{J} \beta_j B_j(x; \delta) = \mathbf{B}(x)^\top \boldsymbol{\beta}, \tag{2.12}$$

where $B_j(\cdot; \delta)$ is the jth B-spline basis function of degree δ, and the design matrix is $\mathbf{B}(x) = (B_1(x), ..., B_J(x))^\top$ (where for simplicity, δ was dropped). The basis functions are defined on a grid of $J - (\delta - 1)$ inner knots $\xi_1, \ldots, \xi_{J-(\delta-1)}$ with additional boundary knots (and usually a knot expansion in the boundary knots). For more details on the construction of B-splines we refer to Eilers and Marx (1996). The function estimates can be written in matrix notation as $\hat{f}(\mathbf{x}) = \mathbf{B}\hat{\boldsymbol{\beta}}$, where the design matrix $\mathbf{B} = (\mathbf{B}(x_1), \ldots, \mathbf{B}(x_n))^\top$ comprises the B-spline basis vectors $\mathbf{B}(x)$ evaluated for each observation x_i, $i = 1, \ldots, n$.

The function estimate $\hat{f}(\mathbf{x})$ might adapt the data too closely and might become too erratic. To enforce smoothness of the function estimate, an additional penalty is used that penalizes large differences of the coefficients of adjacent knots. Hence, for a continuous response $\mathbf{u}$ (in our case the negative gradient vector (Eq. 2.4), i.e., the 'working residuals'), we can estimate the function via a penalized least squares criterion

$$\mathcal{Q}(\boldsymbol{\beta}) = (\mathbf{u} - \mathbf{B}\boldsymbol{\beta})^\top (\mathbf{u} - \mathbf{B}\boldsymbol{\beta}) + \lambda \mathcal{J}(\boldsymbol{\beta}; d) \tag{2.13}$$

with a quadratic difference penalty on the coefficients

$$\mathcal{J}(\boldsymbol{\beta}; d) = \sum_{j=d+1}^{J} (\Delta^d \beta_j)^2 = \boldsymbol{\beta}^\top \mathbf{D}_{(d)}^\top \mathbf{D}_{(d)} \boldsymbol{\beta}, \tag{2.14}$$

where d is the order of the difference penalty for the P-spline and λ is the smoothing parameter that governs the trade-off between the smoothness and the closeness to the data. The difference operator is defined as in Equation (2.10). Higher order difference penalties can be easily derived. Difference penalties of order one penalize the deviation from a constant. Second order differences penalize the deviation from a straight line. In general, differences of order d penalize deviations

from polynomials of order $d-1$. The unpenalized effects, i.e., the constant or the straight line for $d=1$ or $d=2$, respectively, are called the null space of the penalty. For a detailed discussion see Section 3.1 (especially Equation (3.9)).

The difference matrices $\mathbf{D}_{(d)}$ are constructed such that they lead to the appropriate differences: first order differences result from a matrix of the form

$$\mathbf{D}_{(1)} = \begin{pmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{pmatrix}, \tag{2.15}$$

and second order differences from a difference matrix of the form

$$\mathbf{D}_{(2)} = \begin{pmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \\ & & & 1 & -2 & 1 \end{pmatrix}, \tag{2.16}$$

where again empty cells are equal to zero. This estimation problem can then be written as a penalized linear model (2.8) with design matrix $\mathbf{X} = \mathbf{B}$ and penalty matrix $\mathbf{K} = \mathbf{D}_{(d)}^{\top}\mathbf{D}_{(d)}$.

2.3.3. Bivariate P-spline Base-learners

Bivariate P-splines are an extension of univariate P-splines (see above) that allow to model smooth effects of two variables. Spatial effects are the most common effects that are modeled using bivariate P-splines. However, they can be used to model smooth interaction surfaces for other than spatial variables as well. This is for example illustrated in Section 4.6, where the activity profile of Roe Deer is modeled as bivariate smooth effect dependent on the calendar day and the time of the day. A bivariate B-spline of degree δ for two variables x_1 and x_2 can be constructed as the product of two univariate B-spline bases

$$B_{jk}(x_1, x_2; \delta) = B_j^{(1)}(x_1; \delta) \cdot B_k^{(2)}(x_2; \delta).$$

The bivariate B-spline basis is formed by all possible products B_{jk}, $j = 1, \ldots, J$, $k = 1, \ldots, K$. For the sake of simplicity we again drop the degree of the basis functions δ. In the direction of x_1, the basis functions are defined on a grid of

$J - (\delta - 1)$ inner knots, and in the direction of x_2, $K - (\delta - 1)$ inner knots are used. Hence, different knot meshes for x_1 and x_2 are possible. Furthermore, one could also use B-spline basis functions with different degrees δ_1 and δ_2 for x_1 and x_2, respectively. However, we will not further elaborate this in this thesis. A bivariate function $f(x_1, x_2)$ can be approximated as

$$f(x_1, x_2) = \sum_{j=1}^{J} \sum_{k=1}^{K} \beta_{jk} B_{jk}(x_1, x_2) = \mathbf{B}(x_1, x_2)^\top \boldsymbol{\beta},$$

where the vector of B-spline bases for variables (x_1, x_2) equals

$$\mathbf{B}(x_1, x_2) = \Big(B_{11}(x_1, x_2), \ldots, B_{1K}(x_1, x_2), B_{21}(x_1, x_2), \ldots, B_{JK}(x_1, x_2)\Big)^\top,$$

and the coefficient vector

$$\boldsymbol{\beta} = (\beta_{11}, \ldots, \beta_{1K}, \beta_{21}, \ldots, \beta_{JK})^\top. \tag{2.17}$$

The $(n \times JK)$ design matrix then combines the vectors $\mathbf{B}(\mathbf{x}_i)$ for observations $\mathbf{x}_i = (x_{i1}, x_{i2})$, $i = 1, \ldots, n$, such that the ith row contains $\mathbf{B}(\mathbf{x}_i)$:

$$\mathbf{B} = \Big(\mathbf{B}(\mathbf{x}_1), \ldots, \mathbf{B}(\mathbf{x}_i), \ldots, \mathbf{B}(\mathbf{x}_n)\Big)^\top. \tag{2.18}$$

The design matrix $\mathbf{B}$ can be conveniently obtained by first evaluating the univariate B-spline bases $\mathbf{B}_1 = \Big(B_j^{(1)}(x_{i1})\Big)_{\substack{i=1,\ldots,n \\ j=1,\ldots,J}}$ and $\mathbf{B}_2 = \Big(B_k^{(2)}(x_{i2})\Big)_{\substack{i=1,\ldots,n \\ k=1,\ldots,K}}$ of the variables x_1 and x_2. The univariate bases can subsequently be used to construct the design matrix as

$$\mathbf{B} = (\mathbf{B}_1 \otimes \boldsymbol{e}_K^\top) \odot (\boldsymbol{e}_J^\top \otimes \mathbf{B}_2), \tag{2.19}$$

where $\boldsymbol{e}_K = (1, \ldots, 1)^\top$ is a vector of length K and $\boldsymbol{e}_J = (1, \ldots, 1)^\top$ a vector of length J. The symbol $\otimes$ denotes the Kronecker product and $\odot$ denotes the element-wise product. Definitions and examples for both products are given in Appendix A.1. A graphical display of a subset of the tensor product B-spline basis can be found in Figure 2.2.

As for univariate P-splines, a suitable penalty matrix is required to enforce smoothness. The bivariate penalty matrix can be constructed from separate, univariate difference penalties for x_1 and x_2, respectively. Consider the $(J \times J)$ penalty matrix $\mathbf{K}_1 = \mathbf{D}_1^\top \mathbf{D}_1$ for x_1, and the $(K \times K)$ penalty matrix $\mathbf{K}_2 = \mathbf{D}_2^\top \mathbf{D}_2$ for x_2.

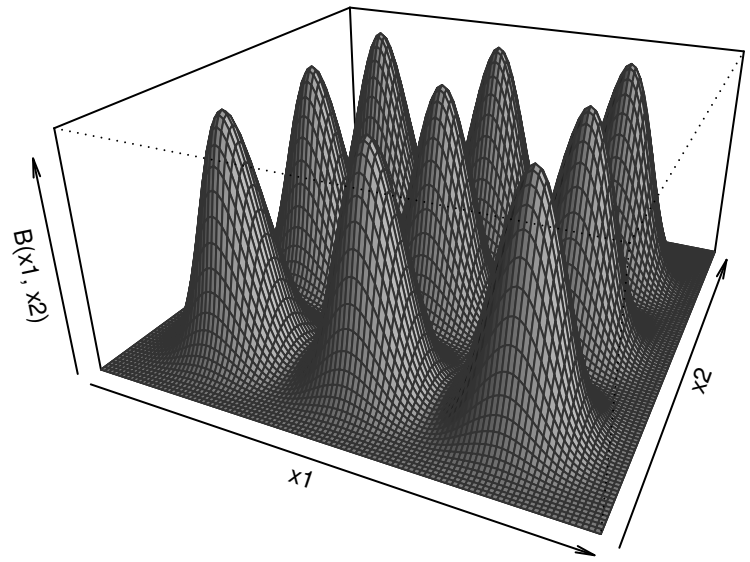


Figure 2.2.: Subset of a bivariate tensor product B-spline basis of degree three.

The penalties are constructed using difference matrices $\mathbf{D}_1$ and $\mathbf{D}_2$ of (the same) order d. However, different orders of differences d_1 and d_2 could be used if this is required by the data at hand. The combined difference penalty can then be written as the sum of Kronecker products

$$\mathcal{J}_{\text{spatial}}(\boldsymbol{\beta}; d) = \boldsymbol{\beta}^\top (\mathbf{K}_1 \otimes \mathbf{I}_K + \mathbf{I}_J \otimes \mathbf{K}_2)\boldsymbol{\beta}, \tag{2.20}$$

with identity matrices $\mathbf{I}_J$ and $\mathbf{I}_K$ of rank J and K, respectively. The penalty is constructed such that differences of the coefficients in the directions of x_1 and x_2 are considered. This corresponds to row-wise and column-wise differences for the matrix of coefficients

$$\mathcal{B} = \left(\beta_{jk}\right)_{\substack{j=1,\ldots,J \\ k=1,\ldots,K}}. \tag{2.21}$$

Hence, in a matrix-like notation to resemble the neighborhood, the first order differences for β_{jk} are

$$\begin{array}{ccc} & \beta_{jk} - \beta_{j(k-1)} & \\ \beta_{jk} - \beta_{(j-1)k} & & \beta_{(j+1)k} - \beta_{jk} \\ & \beta_{j(k+1)} - \beta_{jk} & \end{array}.$$

Higher order differences can be visualized analogously.

With the response vector $\mathbf{u}$, models can then be estimated by optimizing the

penalized least squares criterion in analogy to univariate P-splines:

$$\mathcal{Q}(\boldsymbol{\beta}) = (\mathbf{u} - \mathbf{B}\boldsymbol{\beta})^\top (\mathbf{u} - \mathbf{B}\boldsymbol{\beta}) + \lambda \mathcal{J}_{\text{spatial}}(\boldsymbol{\beta}; d), \tag{2.22}$$

with design matrix (2.18) and penalty (2.20).

For more details on tensor product splines we refer to Wood (2006b). Furthermore, Kneib *et al.* (2009) give an introduction to tensor product P-splines in the context of boosting.

2.3.4. Radial Basis Function Base-learners

Radial basis-functions (RBFs) are another way to estimate interaction surfaces or spatial effects. Let $(u_i, \mathbf{x}_i^\top)$, $i = 1, \ldots, n$ denote the vectors of observations, where the response $\mathbf{u}$ is the negative gradient vector and the covariates $\mathbf{x}_i \in \mathbb{R}^q$. In spatial models $\mathbf{x}_i$ represents the two dimensional location of observation i. However, $\mathbf{x}_i$ could also be a scalar ($q = 1$; e.g. in time series) or have higher dimensions ($q > 2$). In the remainder of this section we consider (without loss of generality) spatial models, i.e., $\mathbf{x}_i \in \mathbb{R}^2$. Before we consider special types of radial basis functions we develop the general concept of radial basis functions. As they solely depend on the *distance* between an observation $\mathbf{x}_i$ and a knot $\boldsymbol{\xi}_j \in \mathbb{R}^q$, and on the distances between different knots ξ_j and $\xi_{j'}$, they can be used in any case where a suitable distance measure exists. In two dimensional space one usually applies the Euclidean norm, however in principle an arbitrary norm could be used. With $r_{ij} = \|\mathbf{x}_i - \boldsymbol{\xi}_j\|$ we get a radial basis function

$$B_{\boldsymbol{\xi}_j}(\mathbf{x}_i) = B(\|\mathbf{x}_i - \boldsymbol{\xi}_j\|) = B(r_{ij}).$$

with a basis function B (examples for B see below). Hence, for each knot $\boldsymbol{\xi}_j$ one gets a radial basis function $B_{\boldsymbol{\xi}_j}$ centered around this knot. In the following paragraphs we will briefly introduce two special cases of radial basis functions, namely kriging and thin plate splines.

Kriging as Special RBF Approach To obtain smooth estimates of a function, a basis function expansion for a given grid of knots can be applied. The aim is to estimate the model

$$u_i = f(\mathbf{x}_i) + \varepsilon_i, \qquad i = 1, \ldots, n,$$

where $\varepsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$. The idea of kriging is to consider a zero mean, stationary Gaussian process for $f(\mathbf{x}_i) \equiv \gamma_i$, i.e., $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_n)^\top \sim \mathcal{N}(\mathbf{0}, \tau^2 \mathbf{R})$ with correlation matrix $\mathbf{R}$. To obtain parsimonious models we use an isotropic parametric correlation function $\mathbf{R}_{i,j} = \rho(\gamma_i, \gamma_j) = \rho(\|\mathbf{x}_i - \mathbf{x}_j\|)$. The model can be expressed in matrix notation as

$$f(\mathbf{x}_i) = \mathbf{I}\boldsymbol{\gamma},$$

where $\mathbf{I}$ is the $(n \times n)$ identity matrix. To incorporate the Gaussian process assumption to the estimation problem, we can exploit the fact that Gaussian priori assumptions on the parameters can be expressed as a special quadratic penalty in a penalized least squares problem (see Appendix A.2). Here, the penalty becomes

$$\mathcal{J}(\boldsymbol{\gamma}) = \boldsymbol{\gamma}^\top \mathbf{R}^{-1} \boldsymbol{\gamma} \,,$$

and $\mathbf{R}^{-1}$ is the penalty matrix. Substituting the identity matrix $\mathbf{I}$ by $\mathbf{R}\mathbf{R}^{-1}$ and defining $\mathbf{X} := \mathbf{R}$ and $\boldsymbol{\beta} := \mathbf{R}^{-1}\boldsymbol{\gamma}$, the function estimate can be equivalently written as

$$f(\mathbf{x}_i) = \mathbf{I}\boldsymbol{\gamma} = \underbrace{\mathbf{R}\mathbf{R}^{-1}}_{=\mathbf{I}} \boldsymbol{\gamma} = \mathbf{X}\boldsymbol{\beta}.$$

Expanding the penalty with $\mathbf{R}^{-\top}\mathbf{R} = \mathbf{I}$ (note that $\mathbf{R}$ is symmetric), it can be written dependent on $\boldsymbol{\beta}$ as

$$\mathcal{J}(\boldsymbol{\beta}) = \underbrace{\boldsymbol{\gamma}^\top (\mathbf{R}^{-\top}}_{=\boldsymbol{\beta}^\top} \mathbf{R}) \underbrace{\mathbf{R}^{-1}\boldsymbol{\gamma}}_{=\boldsymbol{\beta}} = \boldsymbol{\beta}^\top \mathbf{R} \boldsymbol{\beta} \,.$$

Altogether, the estimation problem can be written as a penalized least squares problem

$$\mathcal{Q}(\boldsymbol{\beta}) = (\mathbf{u} - \mathbf{X}\boldsymbol{\beta})^\top (\mathbf{u} - \mathbf{X}\boldsymbol{\beta}) + \lambda \boldsymbol{\beta}^\top \mathbf{K} \boldsymbol{\beta}, \tag{2.23}$$

where the quotient $\lambda = \sigma^2/\tau^2$ plays the role of the smoothing parameter, $\mathbf{X} = \mathbf{R}$ and $\mathbf{K} = \mathbf{R}$ (cf. Fahrmeir, Kneib, and Lang 2007, p. 384). With the design matrix $\mathbf{R} = \Big(\rho(\|\mathbf{x}_i - \mathbf{x}_j\|)\Big)_{i,j}$ it follows that $\rho(\cdot, \mathbf{x}_j)$ plays the role of a (radial) basis function with knot $\mathbf{x}_j$ and $\mathbf{R}$ is the corresponding penalty matrix. The solution to this penalized least squares problem is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{K})^{-1} \mathbf{X}^\top \mathbf{u} = (\mathbf{R}^\top \mathbf{R} + \lambda \mathbf{R})^{-1} \mathbf{R}^\top \mathbf{u}, \tag{2.24}$$

i.e., we have again a base-learner of form (2.8).

As in other smoothing approaches, such as P-splines, one can use a reduced set

of knots $\{\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_J\}$ with $J \leq n$. This leads to reduced knot or low-rank kriging as described in Nychka and Saltzman (1998) and Kammann and Wand (2003). The design matrix for low-rank kriging is then defined as

$$\mathbf{X} = \Big(\rho(\|\mathbf{x}_i - \boldsymbol{\xi}_j\|)\Big)_{i=1,\ldots,n,\, j=1,\ldots,J}$$

and the corresponding (low-rank) penalty matrix is defined as

$$\mathbf{K} = \Big(\rho(\|\boldsymbol{\xi}_j - \boldsymbol{\xi}_{j'}\|)\Big)_{j=1,\ldots,J,\, j'=1,\ldots,J}\,.$$

With these matrices, $f(\mathbf{x}_i)$ can again be estimated by penalized least squares as in Equation (2.23). One problem that occurs with low-rank kriging is the appropriate choice of knots. To specify knots in higher dimensions that cover the data region one can use a space filling algorithm (Johnson, Moore, and Ylvisaker 1990, Nychka and Saltzman 1998). The resulting knots are a subset of the original observations. Usually, the number of knots for low-rank kriging can be smaller than the number of knots for tensor product P-splines. Knots in low-rank kriging can be chosen data adaptive. Thus, in regions with many observations, the space filling algorithm places more knots than in regions with few observations. In contrast, bivariate P-splines span a *rectangular* lattice and knots are placed irrespective of the number of observations in this region. Hence, in regions with little or no data we get the same knot mesh as in regions with many observations. As a consequence, more knots are required to cover the data region appropriately. As the space filling algorithm tends to be slow for larger numbers of observations and knots, a random subset of the observations could be used to determine the knot locations in larger data sets.

To simplify the involved computations, one can reformulate the estimation problem (2.23), both for kriging and low-rank kriging, by reparameterizing the design matrix as $\tilde{\mathbf{X}} = \mathbf{X}\mathbf{K}^{-1/2}$. In this case, the penalty simplifies to the ridge penalty, i.e., $\tilde{\mathbf{K}} = \mathbf{I}$ (Kammann and Wand 2003).

A crucial problem for kriging that was not mentioned so far, is the choice of the radial basis function, i.e., the choice of the correlation function $\rho(\cdot)$. Stein (1999, Sec. 1.7) advocates to use the Matérn family to specify the correlation function, due to its flexibility w.r.t. modeling different degrees of differentiability and the fact that it contains, e.g., the exponential model, and (as a limit) the Gaussian model (Stein 1999, Sec. 1.6). The Matérn family is specified as a function of κ and θ, i.e., $\rho(\cdot; \kappa, \theta)$. The parameter $\kappa > 0$ defines the smoothness, and the parameter

$\theta > 0$ defines the effective range. The typical choice $\kappa = 1.5$ leads to $\rho(r; \kappa = 1.5, \theta) = \exp(-|r/\theta|)(1 + |r/\theta|)$. More generally, modified Bessel functions (Stein 1999) are needed to express Matérn correlation functions. A simple rule to specify the effective range is to use $\hat{\theta} = \max_{i,j}(\mathbf{x}_i - \mathbf{x}_j)/c$, where $c > 0$ is chosen such that $\rho(\max_{i,j}(\mathbf{x}_i - \mathbf{x}_j); \kappa, \theta = \hat{\theta}) = \varepsilon$ or equivalently $\rho(c; \kappa, \theta = 1) = \varepsilon$ with a small number ε, for example $\varepsilon = 0.001$ (Kneib and Fahrmeir 2006). This choice of $\hat{\theta}$ ensures the scale invariance of the basis function $\rho(\cdot)$.

Thin Plate Splines Instead of using correlation functions as basis functions, we can specify the basis functions along the lines of thin plate splines (TPS) (e.g., Nychka 2000, Wood 2003). Thin plate splines are a generalization of cubic smoothing splines and are the minimizer of the penalized least squares criterion

$$\mathcal{Q}(f) = \frac{1}{n}\sum_{i=1}^{n}(u_i - f(\mathbf{x}_i))^2 + \lambda \mathcal{J}_s(f)$$

with $\lambda > 0$ and a penalty

$$\mathcal{J}_s(f) = \int_{\mathbb{R}^q} \sum_{\mathcal{A}} \frac{s!}{\alpha_1! \ldots \alpha_q!} \left(\frac{\partial^s f}{\partial x_1^{\alpha_1} \ldots \partial x_q^{\alpha_q}} \right)^2 d\mathbf{x}, \quad \mathcal{A} = \left\{ \boldsymbol{\alpha} : \alpha_j \in \mathbb{N}_0, \sum_{j=1}^{q} \alpha_j = s \right\},$$

where s is the order of differentiation in the wiggliness penalty, q is the dimension of the covariate space, and $2s > q$. The solution to the penalized least squares criterion can then be written as

$$f(\mathbf{x}) = \mathbf{Z}\boldsymbol{\delta} + \mathbf{X}\boldsymbol{\beta},$$

where $\mathbf{Z}$ represents the null space and $\boldsymbol{\delta}$ the corresponding parameter vector. The null space of thin plate splines, i.e., the part which is not penalized by $\mathcal{J}_s$, consists of all polynomials with degree smaller or equal to $s - 1$. The coefficient vector of the penalized part is denoted by $\boldsymbol{\beta}$. The corresponding design matrix can be written as $\mathbf{X} = \Big(c(s,q) \cdot B(\|\mathbf{x}_i - \mathbf{x}_j\|)\Big)_{i,j}$ with constant $c(s,q)$ and radial basis function

$$B(\|\mathbf{x} - \boldsymbol{\xi}\|) = B_{\boldsymbol{\xi}}(\mathbf{x}) = \begin{cases} \|\mathbf{x} - \boldsymbol{\xi}\|^{2s-q} & \text{if } q \text{ odd} \\ \|\mathbf{x} - \boldsymbol{\xi}\|^{2s-q} \log(\|\mathbf{x} - \boldsymbol{\xi}\|) & \text{if } q \text{ even.} \end{cases}$$

Estimation of the smooth part can be expressed in the same manner as for the kriging estimates above (French, Kammann, and Wand 2001). Note that the constant can be neglected. In this case the coefficients and the smoothing parameter λ change by the factor $c(s,q)$ and $c(s,q)^{-1}$, respectively. For computational reasons we again prefer low rank smoothing where the knots are specified using the space filling algorithm (Nychka and Saltzman 1998) leading to approximate thin plate splines. For more details in a mixed model estimation framework we refer to Ruppert, Wand, and Carroll (2003, Ch. 13). In the special case of $s = 2$ and $q = 1$ we get the well known cubic smoothing splines (see Reinsch 1967) or approximate cubic smoothing splines if we use reduced rank smoothing.

2.3.5. Constrained Base-learners

Constrained effect estimates allow us to model monotonicity constrained effects as well as cyclic effects. In the first case, researchers can specify their knowledge on the nature of the effect and constrain the estimate to be monotonically increasing or decreasing. This stabilizes the estimate and helps to avoid erratic estimates, which might be due to sparse data in certain regions, or noisy data. Cyclic effect estimates are another way to facilitate the inclusion of subject matter knowledge into estimation. Especially seasonal trends can be modeled using cyclic estimates. The constraint stabilizes the estimation especially at the boundary regions. In both cases, the further constraint arising from a priori knowledge, might even improve the prediction accuracy, as the erratic behavior of the estimate is inhibited (see Sec. 4.3). Both constraints can be used in conjunction with univariate P-splines as well as with bivariate P-splines. Monotonicity constraints can be furthermore applied to ordered categorical variables. A detailed introduction and derivation of constrained base-learners is given in Chapter 4.

2.4. Tuning the Boosting Algorithm

There are several potential tuning parameters in the boosting algorithm: the step-length factor ν, the degrees of freedom df of the base-learners and the stopping iteration m_{stop}.

2.4.1. Step-Length

The step-length factor ν ($0 < \nu \leq 1$) is of minor importance for the performance of the boosting algorithm. The step-length factor resembles the learning rate of boosting. Smaller values of ν correspond to more shrinkage. In the original algorithm, Friedman (2001) used an additional line search (Eq. 2.3) in step (c) of the generic FGD algorithm (Algorithm 1). However, Bühlmann and Hothorn (2007) argue that this is not necessary for a good prediction performance but it increases the computational demand. It is sufficient to specify the step-length ν 'small enough'. As discussed by Friedman (2001), small values of ν such as $\nu = 0.1$ lead to a good performance[4] and dramatically improve the algorithm compared to no shrinkage (i.e., $\nu = 1$). A further reduction of the step-length usually results in an increased number of boosting iterations $m_{\text{stop,opt}}$. However, these additional iterations have no impact on the predictive performance of the *final* model (Schmid and Hothorn 2008a).

The step-length factor can be seen as an incremental shrinkage (Friedman 2001): that means, in each boosting step the estimate is shrunken by the rate ν. Decreasing the learning rate to a certain extent leads to a strong increase in performance. On the other hand, global shrinkage of the entire model results in less pronounced performance improvements as incremental shrinkage (Friedman 2001). The effect of incremental shrinkage is very complex. Therefore, the reason for the good performance is not completely understood by now. One possibility for the superiority of incremental shrinkage could be caused by groups of (highly) correlated predictors. If we choose one of these predictors and add only a fraction of its fit, other predictors of the same group still stand a chance of entering the model in subsequent iterations. Consequently the overall fit might be improved. If the first predictor enters the model without additional shrinkage, the other covariates will have little chance to enter the model as well, as in this situation substantial additional improvements are hard to gain. Global shrinkage of the model with only one (or few) of the correlated covariates would not result in the same effect as incremental shrinkage (Friedman 2001, p. 1206).

Hastie *et al.* (2001, pp. 328f.) further elaborate on the shrinkage effect of boosting. They show that component-wise L_2Boosting can be seen as a solution to the lasso problem if the step-size $\nu \to 0$. This connection only holds if the design matrix needs to satisfy the positive cone condition (Efron, Hastie, Johnstone, and

[4] In the package **mboost**, $\nu = 0.1$ is set as the default value.

Tibshirani 2004). If this condition does not hold, the boosting solutions still resemble the solutions of the lasso at least qualitatively (Hastie *et al.* 2001, p. 329). Thus, boosting can be seen as one way to solve L_1 penalized models in high-dimensional settings. As for the lasso, the resulting models tend to be sparse. Despite the fact that lasso and boosting are not equivalent in general it seems useful for the understanding of boosting to think of it as a method that is similar to L_1-penalized methods (Bühlmann and Hothorn 2007, p. 492).

2.4.2. Degrees of Freedom

Another tuning parameter in the context of boosting are the degrees of freedom of the base-learners. These are directly related to the smoothing parameter λ of the base-learner (2.8), as we will show in Section 3.1. In short, one could state that the degrees of freedom are not at all crucial for the performance of the boosting algorithm as long as they are small enough[5]. Small degrees of freedom result in *weak* learners as desired. However, this is not the complete truth. As we will show in Section 3, it is desirable that the degrees of freedom additionally are equal for all base-learners. This helps to avoid biased base-learner selection. Nevertheless, for boosting the degrees of freedom play only a minor role. In other penalized modeling approaches, the penalty parameter λ — or correspondingly the degrees of freedom (*df*) — are the *major* tuning parameters, which control the shape and flexibility of the final estimate; see, for example, Buja, Hastie, and Tibshirani (1989) for penalization in the context of additive models and, Hoerl and Kennard (1970) for penalization in ridge regression models. In boosting, the flexibility of the smooth terms is not directly governed by the initial degrees of freedom for the base-learner. By repeatedly selecting the base-learner, the final effect can adapt higher order smoothness. Hence, the number of boosting iterations controls much of the flexibility of the estimates (see next section). At the same time, overfitting is much slower in boosting than in other approaches where small increases of the degrees of freedom might result in heavy overfitting (see Bühlmann and Yu 2003, in the context of boosting with smoothing splines).

[5] In package **mboost**, the default for `df` for smooth base-learners is set to 4, except for bivariate P-splines where the default is set to 6 due to a null space that already has 4 degrees of freedom.

2.4.3. Stopping Iteration

The most important tuning parameter for boosting is the number of boosting iterations m_{stop}. Boosting, especially AdaBoost, is generally considered to be resistant to overfitting (e.g., Breiman 1998, Friedman *et al.* 2000). However, as argued in Bühlmann and Yu (2000) and Friedman *et al.* (2000, rejoinder), this resistance to overfitting needs to be largely attributed to the way it is evaluated. In the classification context, the model is often assessed via the misclassification rate, which itself is very stable against overfitting. Evaluating the performance with the out-of-bag exponential loss (for AdaBoost) or the Binomial log-likelihood (for LogitBoost) shows that overfitting can and will occur eventually (Friedman *et al.* 2000, rejoinder). Nevertheless, overfitting is relatively slow if appropriate weak learners are used and additional shrinkage via the step-length ν is introduced (Bühlmann and Hothorn 2007). Overfitting can be avoided if one stops the boosting algorithm at an appropriate stage. This is called 'early stopping'. There are two ways to determine a suitable stopping iteration $m_{\text{stop,opt}}$: First, one could estimate $m_{\text{stop,opt}}$ based on information criteria such as the AIC (Akaike 1974), the corrected AIC (Hurvich, Simonoff, and Tsai 1998), or the gMDL criterion (Hansen and Yu 2001). Second, resampling based alternatives such as cross-validation (Stone 1974) or the bootstrap (Efron 1979) can be used. The idea of all approaches is to mimic the performance of the model on new data, while taking possible artifacts due to overfitting of the learning data into account.

Definitions of AIC and gMDL in the context of boosting are given in Bühlmann and Hothorn (2007). Essentially, they use the trace of the hat matrix of componentwise boosting to estimate the degrees of freedom of the boosted model. For L_2Boosting, they derived the hat matrix in iteration m as

$$\boldsymbol{\mathcal{B}}_m = \mathbf{I} - \prod_{r=1}^{m} (\mathbf{I} - \nu \boldsymbol{\mathcal{S}}^{(j_r^*)}), \tag{2.25}$$

where $\boldsymbol{\mathcal{S}}^{(j_r^*)}$ is the smoother matrix of the rth boosting step for the best-fitting base-learner j_r^* and $\mathbf{I}$ is the identity matrix. Bühlmann and Hothorn (2007) also derive approximate hat matrices for boosting based on the (negative) binomial log-likelihood and the (negative) Poisson log-likelihood (see their formulae (7.3) and (7.4), respectively). The degrees of freedom of the model in iteration m can then be

estimated as the trace of the approximate hat matrix:

$$\text{df}(m) = \text{tr}(\boldsymbol{\mathcal{B}}_m).$$

The AIC in iteration m then becomes $\text{AIC}(m) = -2\,\ell(\hat{f}^{[m]}) + 2\,\text{df}(m)$, where $\ell(\hat{f}^{[m]})$ is the log-likelihood in iteration m.

Chang, Huang, and Huang (2010) propose an early stopping rule based on information criteria. They record the AIC while running the boosting algorithm and use a change point detection rule to find the minimum AIC 'on-the-fly'.

Hastie (2007) criticizes the use of information criteria, which are based on some notion of degrees of freedom for the boosting model, to determine the optimal stopping iteration $m_{\text{stop,opt}}$ of the boosting algorithm. He states, that the estimators for the model degrees of freedom given in the boosting literature (e.g., Bühlmann and Yu 2003, Bühlmann and Hothorn 2007, Tutz and Binder 2006) neglect the flexibility of choosing the base-learners in each iteration and solemnly consider the flexibility of the model as if the selection steps were predefined. Hence, the degrees of freedom underestimate the true degrees of freedom. The AIC tends to stop to late, i.e., the optimal stopping iteration $m_{\text{stop,opt}}$ is over-estimated.

To achieve an unbiased estimator for the optimal stopping iteration of the boosting algorithm, it is preferable to use cross-validation techniques (including k-fold cross-validation and the bootstrap), which do not require the estimation of the degrees of freedom for the model. Cross-validation avoids the biased estimates of the degrees of freedom and is available for any kind of regression problem, i.e., not only in special cases where an (approximate) hat matrix can be derived. In both cases, the data set is split into two parts, a learning sample and a validation sample. The model is fitted on the learning sample and the empirical out-of-bag risk, i.e., the empirical risk on the validation sample, is recorded for each iteration up to the initial m_{stop}. This is done repeatedly. Finally, one chooses the iteration that results in the lowest *aggregated* out-of-bag estimate of the empirical risk.

If we are not sure how many iterations are required for the optimal stopping iteration, a large initial value of m_{stop} might be required. Thus, each of the cross-validation or bootstrap runs need to run up to this iteration. Here, the tendency of AIC-based stopping to overshoot slightly, could be used to find an initial 'guess' for $m_{\text{stop,opt}}$, which we call $\widehat{m}_{\text{stop,AIC}}$. Based on this value one can now run the search for an optimal stopping iteration on a smaller set as one can stop at $\widehat{m}_{\text{stop,AIC}}$. This idea could also be combined with the change point detection method of Chang

et al. (2010). However, we feel that this is not that necessary. It seems sufficient to run the algorithm and check from time to time if the minimum is reached and if the optimal m_{stop} is not to close to the current m_{stop}. In situations with few observations and many candidate predictor variables ($n \ll p$), the use of $m_{\text{stop,AIC}}$ could be especially beneficial. In this situation, computing the AIC 'on the fly' is relatively quick and easy. Cross-validation — with a high number of initial boosting steps m_{stop} — on the other hand might become computationally very demanding as in each boosting step all the base-learners need to be evaluated. One could then use the estimate $\widehat{m}_{\text{stop,AIC}}$ as an upper bound for the number of boosting iterations in cross-validation. For data sets with many observations (i.e., when n becomes large), this strategy is less useful. The computation of the approximate hat matrix becomes more demanding as it is an $(n \times n)$ matrix.

2.5. Boosting as a Method for Sparse Modeling

A feature that directly relates to the component-wise fitting and the subsequent selection of base-learners is that boosting works especially well in settings where the number of covariates is large and where it is desirable to select a relatively small subset of predictors. In these situations boosting usually outperforms standard regression models with subset selection methods (Schmid and Hothorn 2008a). As each base-learner depends only on a small subset of the predictors and as in each iteration only one of the base-learners is fitted to the negative gradient vector, component-wise boosting can be even applied in settings where the number of predictors is much larger than the number of observations ($n \ll p$). Many classical variable selection techniques fail in this case. Others can only add up to (at absolute maximum) n variables to the model. In contrast, due to regularization, the final boosting model could — in theory — depend on all p predictors. However, usually this does not happen and it is not desirable as sparse models are easier to handle and interpret. The optimal model depends on few predictors only while having a very good prediction capability. As discussed in the introduction (Chapter 1), there are two competing goals when we try to learn (i.e., extract information) from data: the first goal is prediction, the second is interpretation. Prediction is the main target in machine learning, while interpretation is rather targeted in the statistical community. Even if these goals are competing they are not always mutually exclusive. Sparse, regularized models often lead to better prediction accuracy and

at the same time are easier to interpret and understand. This is both facilitated by sparse models, as long as the level of sparsity is chosen with care. First, sparse models are easier to interpret as there are fewer variables (and corresponding effects), and less complicated (interaction) structures will be derived in the model. Second, sparse models tend to be superior with regard to prediction: models that are too rich tend to fit the learning data very well — but they perform poorly on new data. A very rich model is usually not generalizable. Hence, prediction fails and at the same time the estimated effects should be doubted. On the other hand, models that are too sparse are contra-productive as well. They will miss important parts of structure in the data and hence are easy to interpret but highly misleading in their results. The prediction performance usually decreases notably. Therefore, it is desirable with respect to both goals (interpretation of the results and prediction modeling) to find a fair amount of sparsity, which relates to a fair amount of informative variables.

To further enhance sparsity of the model several approaches exist in the boosting literature. We will briefly sketch three: The first approach, sparse L_2Boosting (Bühlmann and Yu 2006), changes the criterion that is used to select the best fitting base-learner (Alg. 2, step (b2)). The residual sum of squares (RSS) is replaced by a penalized criterion such as the AIC, the BIC or the generalized minimum description length (gMDL; see above for details) where the degrees of freedom arise from the approximation of the model degrees of freedom. Bühlmann and Yu show that this leads in many situations to sparser models, while the prediction performance at least does not suffer substantially. The second approach, twin boosting (Bühlmann and Hothorn 2010), uses two successive runs of the boosting algorithm. In the first run a standard boosting estimate is obtained. In the second run, the selection of the base-learners is not based on the RSS but on the RSS rescaled by the importance of the base-learner in the first boosting run. Hence, base-learners that were not selected in the first pass are not subject to selection in the second pass. Base-learners that had only a small contribution (e.g., had an effect close to zero) are less likely to be selected in the second round. The third approach is called stability selection (Meinshausen and Bühlmann 2010). Stability selection can be applied to a wide range of methods including boosting. It allows to extract influential variables (or base-learners in the boosting context) with an error control. Stability selection is especially useful in settings with many potential predictors. To achieve variable selection, the empirical probability of a base-learner to enter the model is investigated via subsampling (i.e., the model is trained on random subsets of

the original data of size $\lfloor n/2 \rfloor$; see, for example, Bühlmann and Yu 2002). Only base-learners that enter the model with a probability higher than a specific cut-off value are considered to be influential. At the same time, stability selection controls the family-wise error rate. Hence, non-influential variables are only selected by stability selection with a controllable, small probability. The final model is usually sparser and hence, easier to interpret than the model without stability selection.

Altogether, the third approach seems to be most promising as no changes in the boosting algorithm are required, and as, furthermore, no estimates for the degrees of freedom of the model are required. Stability selection can be simply applied to the final model (almost) no matter how the underlying estimation and selection algorithm works. Moreover, stability selection controls the probability of falsely selected base-learners.

2.6. Fitting GAMLSS with Boosting

As a short outlook that goes beyond the actual scope of this thesis we want to briefly discuss generalized additive models for location, scale and shape (GAMLSS) in the context of boosting. Boosted GAMLSS can be seen as an extension of the boosting algorithm itself, which is used as a basis to fit models for multiple components. Generalized additive models for location, scale and shape were first introduced by Rigby and Stasinopoulos (2001, 2005). This class of models represents a flexible extension of GAMs (Hastie and Tibshirani 1986, 1990). GAMs are typically used to model the mean of the conditional distribution of an outcome. This conditional distribution follows an exponential family. Hence, the scale and shape are implicitly defined by the mean — in combination with the distribution. However, if heteroscedastic or skewed distributions shall be modeled, this approach is not suitable. In contrast, GAMLSS models regress multiple parameters of the conditional distribution — say $\theta_1, \ldots, \theta_K$, where usually $K \leq 4$ — on a set of covariates. These parameters might include the location (e.g., the mean), the scale (e.g., the standard deviation or the variance) and additional shape parameters (e.g., the skewness or the degrees of freedom). By using GAMLSS models, one gets rid of some of the shortcomings and limitations of classical GAMs. The price to pay is the additional complexity of the model and possibly a higher degree of instability compared to simpler generalized linear or generalized additive models.

Other problems arise on the computational, algorithmic side. One problem of

the standard algorithms, which are used to estimate GAMLSS models (Rigby and Stasinopoulos 2005, App. B), is that they cannot be used to fit models to high-dimensional data with less observations than variables ($n \ll p$). If many possible predictors are available variable selection becomes a crucial part of modeling the data. This is especially important for GAMLSSs as we have multiple components, which are all regressed on (subsets of) these predictors. Rigby and Stasinopoulos (2005) propose to use the generalized AIC (GAIC), with AIC and BIC as special cases, for model selection. The problem with this approach is that it tends to be highly instable. Furthermore, in data sets with many predictors the only feasible approaches that make use of the (G)AIC are stepwise approaches. This increases the instability even further and an 'optimal' or 'near-optimal' model might be completely missed in this case. Other problems associated with the AIC are that it tends to overfit the data, i.e., in tendency the AIC favors models that are too large (e.g. Ripley 2004). In penalized fitting approaches, such as generalized additive models and their extensions, further conceptual problems arise, as — strictly speaking — the AIC is only valid for maximum likelihood estimation (Ripley 2004).

Our idea to overcome the problem of model selection, in this complex and (potentially) high-dimensional context, makes use of boosting with its intrinsic variable selection feature. The approach is based on a recently published boosting algorithm for multi-dimensional prediction functions (Schmid, Potapov, Pfahlberg, and Hothorn 2010b), which was extended to the fitting of GAMLSS models (Mayr *et al.* 2011b). In essence, an additional inner loop is processed within each boosting step. In each boosting iteration, we cycle consecutively through all distribution parameters $\theta_1, \ldots, \theta_K$ of the GAMLSS model. For each parameter θ_k, the negative gradient is computed with respect to this parameter (i.e., the partial derivative of the loss function with respect to θ_k) and the current values of the distribution parameters are plugged in as offset values. Subsequently, the negative gradient vector is fitted by (component-wise) base-learners and the current estimate of θ_k is updated with the best-fitting base-learner. Further details about the algorithm are provided in Mayr *et al.* (2011b).

In the simplest version, one uses one common m_{stop} for all distribution parameters. In many cases this might be sufficient. However, in other cases the model for the mean of the distribution θ_1 might, for example, be more complex than the model for the standard deviation θ_2. In this case, using one single stopping iteration might either result in overfitting of the standard deviation or in 'underfitting' of the mean, or possibly a mixture of both. Hence, it might be more sensible to

allow different values of $m_{\text{stop},k}$, $k = 1, \ldots, K$ for the different components. In the algorithm this is achieved by skipping the update of parameters θ_k after $m_{\text{stop},k}$ steps. Only parameters that did not reach their maximum number of iterations are further updated. An issue that arises in the case of 'multi-dimensional stopping' is that each of the stopping parameters has to be optimized simultaneously, i.e., by taking the stopping iterations of the other parameters into account. In line with the classical boosting approach cross-validation techniques are well advised, however, multi-dimensional cross-validation might become problematic if many distribution parameters are estimated. Multi-dimensional stopping is of special interest in high-dimensional settings where variable selection is desired. More research on feasible strategies to find the optimal (multi-dimensional) stopping iterations is required. One solution to this problem might be given by the ideas on AIC-based pre-stopping as discussed in Section 2.4.3. However, in this case, approximate degrees of freedom would be required. On the other hand, if prediction is of primary interest and variable selection is of minor importance, multi-dimensional stopping seems less crucial. Using one single m_{stop} might often be sufficient (as suggested by the simulation studies in Mayr *et al.* 2011b) due to the slow overfitting behavior of boosting.

2.7. Model-based Boosting in R

The generic functional gradient descent algorithm and all base-learners that were discussed in this chapter are implemented in the R (R Development Core Team 2011) add-on package **mboost** (Hothorn *et al.* 2011a). Base-learners that were newly developed as part of this thesis are implemented in the current development version only[6] (**mboost** 2.1-0; Hothorn *et al.* 2011b). Linear base-learners, both for categorical and continuous covariates are implemented in the function `bols()`. A change in this function compared to the standard versions of **mboost** was required to reflect the research presented in this thesis. The change only affects the way categorical covariates are handled if base-learners without intercept (`bols(..., intercept = false)`) are requested. We elaborate on this in Appendix B.2 together with the required code.

In addition to the implementation of various estimation problems as specified by the loss function (via `family` in **mboost**), and many different base-learners,

[6] at the time of writing; the code will be moved to the official package on CRAN in the near future

mboost offers a convenient interface to compute the cross-validated risk (`cvrisk`) and allows the use of parallel computation methods for this task. Parallelization can, for example, be achieved by using the packages **multicore** (Urbanek 2011) or **snow** (Tierney, Rossini, Li, and Sevcikova 2011). Alternatively users can extract information criteria (AIC, corrected AIC and gMDL) by the function `AIC()` for models fitted with appropriate families. Further convenience functions to extract the coefficient estimates, to plot the effects, to make predictions, or to manipulate the model are available in the package **mboost**.

More details, a list of the contributions to the package **mboost**, and a short introduction — showcasing some of the code that was used to produce the analyses in the thesis — is given in Appendix B.

An implementation of boosted GAMLSS models is available in the R add-on package **gamboostLSS** (Hofner, Mayr, Fenske, and Schmid 2011b). By depending on the R package **mboost**, **gamboostLSS** incorporates a wide range of base-learners (cf. Sec. 2.3) and can rely on a well-tested and mature software back end. Furthermore, convenience functions like those implemented in **mboost** exist.

3. A Framework for Unbiased Model Selection

> Statistical thinking will one day be as necessary for efficient citizenship as the ability to read or write.
>
> *(Attributed to H. G. Wells [1])*

The finding that is going to be presented in this chapter is the fact that generalized or structured additive models fitted using a component-wise functional gradient boosting algorithm suffer from variable selection bias if base-learners with different flexibility are considered as competitors. Basically, a variable might be considered influential not only because of its correlation with the response but also because of its measurement scale. The problem was first observed in the regression tree community in the 1980s (Breiman, Friedman, Olshen, and Stone 1984, Loh and Vanichsetakul 1988) and still receives a lot of attention (in so-called 'unbiased trees', see for example Loh 2002, Kim and Loh 2003, Hothorn, Hornik, and Zeileis 2006b). More recently, Strobl *et al.* (2007) observed a tendency of variable importance measures obtained from random forests to favor categorical covariates with a large number of levels compared to a covariate that has the same effect size but is measured at fewer levels. This variable selection bias is hard to deal with analytically. Therefore, this selection bias is usually defined and assessed in the situation where none of the covariates is actually associated with the response ("null case", see for example Loh 2002, Kim and Loh 2003, Hothorn *et al.* 2006b, Strobl *et al.* 2007). Variable selection is said to be unbiased if each covariate has the same probability of being selected in the null case, regardless of its scale. Similarly, model selection is said to be unbiased if each modeling alternative (linear, non-linear, etc.) for one covariate is selected with the same probability in the null case.

The sources of variable and model selection bias in component-wise boosting

[1] see Huff D (1954). *How to lie with statistics*. W.W. Norton & Company, New York, 1993 Paperback edition. p. 3.

under both the null case and when informative covariates are present in the model are theoretically investigated. The results give insights into how to modify the algorithm to reduce the effect of variable selection bias.

3.1. (Un-) Biased Selection of Base-Learners

Model estimation is carried out using the component-wise, model-based boosting approach described in Chapter 2. More precisely we apply the generic functional gradient descent algorithm derived there (Algorithm 2). In this chapter we focus on three special base-learners, namely, ridge-penalized categorical base-learners, penalized ordinal base-learners and P-spline base-learners: In the case of a continuous covariate x, we consider penalized least squares base-learners based on P-splines (cf. Sec. 2.3.2). While an unpenalized least squares base-learner might be the first choice for (dummy coded) categorical covariates, we consider the more general approach of univariate ridge regression with treatment contrasts to serve as base-learner (cf. Sec. 2.3.1). In the case of ordinal categorical covariates, one could also use a ridge penalty for the coefficients of the dummy coded design matrix if penalized estimation is desired. However, penalized ordinal base-learners (cf. Sec. 2.3.1) might be favorable as one often might expect the transition from on category to a neighboring one to result in relatively small changes. As stated earlier, all three base-learners under consideration can be expressed as penalized linear models of the form $g_j(\mathbf{x}) = \mathbf{X}(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{K})^{-1}\mathbf{X}^\top\mathbf{u}$, with a suitable design matrix $\mathbf{X}$, smoothing parameter λ, and an appropriate penalty matrix $\mathbf{K}$.

Using the component-wise boosting approach naturally leads to variable and model selection if we choose an appropriate stopping iteration $\widehat{m}_{\text{stop,opt}}$. However, the selection of base-learners in each iteration can be seriously biased if the competing base-learners have different degrees of flexibility. This bias is intuitively plausible if one tries to distinguish whether a covariate x has a linear or a smooth effect on y. In this case, the usual strategy would be to specify a linear base-learner $g_1(x) = \beta x$ and a smooth base-learner $g_2(x) = f_{\text{smooth}}(x)$, and to choose between the two based on the RSS (Eq. 2.5) in the boosting algorithm. However, the smooth base-learner offers much more flexibility and typically incorporates a linear effect for x as a special case. Hence, we can expect that boosting (almost) always prefers the smooth base-learner over the linear base-learner, regardless of the nature of the true effect. A similar selection bias can be expected when performing variable

selection between competing categorical covariates with different numbers of categories. The covariate with more categories offers greater flexibility and thus is preferred in general when using unpenalized least squares base-learners.

In the following, we will theoretically investigate the presence of a selection bias in the selection of base-learners for the special case of L_2-boosting in the null model, i.e., when the response y is independent of the covariates. Here, a reasonable selection procedure should randomly select the base-learners without preference of more flexible base-learners.

Theorem 3.1 *Let* $\mathbf{x}_1$ *and* $\mathbf{x}_2$ *be categorical covariates with* M_1 *and* M_2 *categories and design matrices* $\mathbf{X}_1$ *and* $\mathbf{X}_2$*. Let* $\mathbf{u}$ *be the* $(n \times 1)$ *negative gradient vector arising in the first step of the boosting algorithm for a response variable* $\mathbf{y}$ *of i.i.d. normally distributed random variables with variance* σ^2 *that is independent of* $\mathbf{x}_1$ *and* $\mathbf{x}_2$*; i.e.,* $\mathbf{u}$ *is simply the centered response variable. Let* $\hat{\boldsymbol{\beta}}_1$ *and* $\hat{\boldsymbol{\beta}}_2$ *denote the effect estimates resulting from unpenalized least squares base-learners and define the difference of the residual sums of squares as* $\boldsymbol{\Delta} = (\mathbf{u} - \mathbf{X}_1\hat{\boldsymbol{\beta}}_1)^\top(\mathbf{u} - \mathbf{X}_1\hat{\boldsymbol{\beta}}_1) - (\mathbf{u} - \mathbf{X}_2\hat{\boldsymbol{\beta}}_2)^\top(\mathbf{u} - \mathbf{X}_2\hat{\boldsymbol{\beta}}_2)$*. Then we have*

$$\mathbb{E}(\boldsymbol{\Delta}) = \sigma^2(M_2 - M_1), \tag{3.1}$$

i.e., $\mathbb{E}(\boldsymbol{\Delta}) = 0$ *if and only if* $M_1 = M_2$*.*

Proof for Theorem 3.1 The difference of the RSS is given by

$$\begin{aligned}
\boldsymbol{\Delta} = \mathrm{RSS}_{\mathbf{X}_1} - \mathrm{RSS}_{\mathbf{X}_2} &= (\mathbf{u} - \mathbf{X}_1\hat{\boldsymbol{\beta}}_1)^\top(\mathbf{u} - \mathbf{X}_1\hat{\boldsymbol{\beta}}_1) - (\mathbf{u} - \mathbf{X}_2\hat{\boldsymbol{\beta}}_2)^\top(\mathbf{u} - \mathbf{X}_2\hat{\boldsymbol{\beta}}_2) \\
&= \mathbf{u}^\top(\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1^\top\mathbf{X}_1)^{-1}\mathbf{X}_1^\top)^2\mathbf{u} - \mathbf{u}^\top(\mathbf{I} - \mathbf{X}_2(\mathbf{X}_2^\top\mathbf{X}_2)^{-1}\mathbf{X}_2^\top)^2\mathbf{u} \\
&= \mathbf{u}^\top(\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1^\top\mathbf{X}_1)^{-1}\mathbf{X}_1^\top)\mathbf{u} - \mathbf{u}^\top(\mathbf{I} - \mathbf{X}_2(\mathbf{X}_2^\top\mathbf{X}_2)^{-1}\mathbf{X}_2^\top)\mathbf{u} \\
&= \mathbf{u}^\top\mathbf{Q}\mathbf{u}
\end{aligned}$$

with $\mathbf{Q} = [\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1^\top\mathbf{X}_1)^{-1}\mathbf{X}_1^\top] - [\mathbf{I} - \mathbf{X}_2(\mathbf{X}_2^\top\mathbf{X}_2)^{-1}\mathbf{X}_2^\top]$, and

$$\mathrm{tr}(\mathbf{Q}) = [n - (M_1 - 1)] - [n - (M_2 - 1)] = M_2 - M_1\ . \tag{3.2}$$

Using the theorem for the expected value of quadratic forms (Ruppert *et al.* 2003, App. A.4.5) it holds:

$$\mathbb{E}(\boldsymbol{\Delta}) = \mathbb{E}(\mathbf{u}^\top\mathbf{Q}\mathbf{u}) = \mathrm{tr}[\mathbf{Q}\,\mathrm{cov}(\mathbf{u})] + \mathbb{E}(\mathbf{u})^\top\mathbf{Q}\mathbb{E}(\mathbf{u})\ . \tag{3.3}$$

As we assumed $\mathbb{E}(\mathbf{u}) = \mathbf{0}$ and $\text{cov}(\mathbf{u}) = \sigma^2\mathbf{I}$, we obtain

$$\mathbb{E}(\boldsymbol{\Delta}) = \sigma^2 \,\text{tr}(\mathbf{Q}) = \sigma^2(M_2 - M_1) \,. \tag{3.4}$$

□

Theorem 3.1 can be interpreted such that the expected difference of the RSS is greater than zero if the number of additional categories of $\mathbf{x}_2$, i.e., $M_2 - M_1$, is greater than zero, which reflects that a selection bias in favor of $\mathbf{x}_2$ is present. To overcome this problem, the base-learners should be made comparable with respect to their flexibility even if the number of categories is different. A specific possibility to achieve this is presented in the following theorem.

Theorem 3.2 *Assume that the assumptions from Theorem 3.1 hold. Furthermore, replace the categorical base-learners with ridge penalized base-learners, where the penalty matrices* $\mathbf{K}_1$ *and* $\mathbf{K}_2$ *are identity matrices (of appropriate dimensions) and* λ_1 *and* λ_2 *are the corresponding smoothing parameters. Let* $\boldsymbol{\mathcal{S}}_1 = \mathbf{X}_1(\mathbf{X}_1^\top\mathbf{X}_1 + \lambda_1\mathbf{K}_1)^{-1}\mathbf{X}_1^\top$ *be the smoother matrix of* $\mathbf{x}_1$ *and* $\boldsymbol{\mathcal{S}}_2$ *be defined accordingly for* $\mathbf{x}_2$*. Then*

$$\mathbb{E}(\boldsymbol{\Delta}) = 0 \qquad \Leftrightarrow \qquad \text{tr}\left(2\boldsymbol{\mathcal{S}}_1 - \boldsymbol{\mathcal{S}}_1^\top\boldsymbol{\mathcal{S}}_1\right) = \text{tr}\left(2\boldsymbol{\mathcal{S}}_2 - \boldsymbol{\mathcal{S}}_2^\top\boldsymbol{\mathcal{S}}_2\right), \tag{3.5}$$

where $\boldsymbol{\Delta}$ *is the difference in RSS resulting from the penalized least squares fits.*

Proof for Theorem 3.2 The difference of the RSS in the ridge penalized model is given by

$$\begin{aligned}
\boldsymbol{\Delta} &= (\mathbf{u} - \mathbf{X}_1\hat{\boldsymbol{\beta}}_{\text{pen},1})^\top(\mathbf{u} - \mathbf{X}_1\hat{\boldsymbol{\beta}}_{\text{pen},1}) - (\mathbf{u} - \mathbf{X}_2\hat{\boldsymbol{\beta}}_{\text{pen},2})^\top(\mathbf{u} - \mathbf{X}_2\hat{\boldsymbol{\beta}}_{\text{pen},2}) \\
&= (\mathbf{u} - \boldsymbol{\mathcal{S}}_1\mathbf{u})^\top(\mathbf{u} - \boldsymbol{\mathcal{S}}_1\mathbf{u}) - (\mathbf{u} - \boldsymbol{\mathcal{S}}_2\mathbf{u})^\top(\mathbf{u} - \boldsymbol{\mathcal{S}}_2\mathbf{u}) \\
&= \mathbf{u}^\top(\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1^\top\mathbf{X}_1 + \lambda_1\mathbf{K}_1)^{-1}\mathbf{X}_1^\top)^2\mathbf{u} - \mathbf{u}^\top(\mathbf{I} - \mathbf{X}_2(\mathbf{X}_2^\top\mathbf{X}_2 + \lambda_2\mathbf{K}_2)^{-1}\mathbf{X}_2^\top)^2\mathbf{u} \\
&= \mathbf{u}^\top\mathbf{Q}_{\text{pen}}\mathbf{u}
\end{aligned}$$

with

$$\begin{aligned}
\mathbf{Q}_{\text{pen}} &= (\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1^\top\mathbf{X}_1 + \lambda_1\mathbf{K}_1)^{-1}\mathbf{X}_1^\top)^2 - (\mathbf{I} - \mathbf{X}_2(\mathbf{X}_2^\top\mathbf{X}_2 + \lambda_2\mathbf{K}_2)^{-1}\mathbf{X}_2^\top)^2 \\
&= -2\mathbf{X}_1(\mathbf{X}_1^\top\mathbf{X}_1 + \lambda_1\mathbf{K}_1)^{-1}\mathbf{X}_1^\top + (\mathbf{X}_1(\mathbf{X}_1^\top\mathbf{X}_1 + \lambda_1\mathbf{K}_1)^{-1}\mathbf{X}_1^\top)^2 \\
&\quad +2\mathbf{X}_2(\mathbf{X}_2^\top\mathbf{X}_2 + \lambda_2\mathbf{K}_2)^{-1}\mathbf{X}_2^\top - (\mathbf{X}_2(\mathbf{X}_2^\top\mathbf{X}_2 + \lambda_2\mathbf{K}_2)^{-1}\mathbf{X}_2^\top)^2 \\
&= \left[-2\boldsymbol{\mathcal{S}}_1 + \boldsymbol{\mathcal{S}}_1^\top\boldsymbol{\mathcal{S}}_1\right] - \left[-2\boldsymbol{\mathcal{S}}_2 + \boldsymbol{\mathcal{S}}_2^\top\boldsymbol{\mathcal{S}}_2\right] \,.
\end{aligned} \tag{3.6}$$

With $\mathbb{E}(\mathbf{u}) = \mathbf{0}$ and $\text{cov}(\mathbf{u}) = \sigma^2\mathbf{I}$, $\sigma^2 > 0$ it follows from (3.4) that

$$\begin{aligned} \mathbb{E}(\boldsymbol{\Delta}) &= 0 \quad \Leftrightarrow \\ \text{tr}(\mathbf{Q}_{\text{pen}}) &= 0 \quad \Leftrightarrow \\ \text{tr}\left\{\left[-2\boldsymbol{\mathcal{S}}_1 + \boldsymbol{\mathcal{S}}_1^\top\boldsymbol{\mathcal{S}}_1\right] - \left[-2\boldsymbol{\mathcal{S}}_2 + \boldsymbol{\mathcal{S}}_2^\top\boldsymbol{\mathcal{S}}_2\right]\right\} &= 0 \quad \Leftrightarrow \\ \text{tr}\left[2\boldsymbol{\mathcal{S}}_1 - \boldsymbol{\mathcal{S}}_1^\top\boldsymbol{\mathcal{S}}_1\right] &= \text{tr}\left[2\boldsymbol{\mathcal{S}}_2 - \boldsymbol{\mathcal{S}}_2^\top\boldsymbol{\mathcal{S}}_2\right] . \end{aligned}$$

□

From Theorem 3.2 one can deduce that the effective degrees of freedom, which are defined as

$$\text{df} := \text{tr}\left(2\boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{S}}^\top\boldsymbol{\mathcal{S}}\right), \tag{3.7}$$

should be comparable for the two competing base-learners in order to overcome the selection bias. Note that the degrees of freedom resulting from Theorem 3.2 differ from the standard definition in the smoothing literature given by $\widetilde{\text{df}} := \text{tr}\,(\boldsymbol{\mathcal{S}})$. However, (3.7) is an alternative definition for the degrees of freedom in penalized models. As stated by Buja *et al.* (1989) and confirmed by Theorem 3.2, it is the preferred choice if one compares two models with respect to their RSS. Both definitions of degrees of freedom are implemented in the dedicated R package **mboost**. One can switch to the degrees of freedom (3.7) with the command

```
R> options(mboost_dftraceS = FALSE)
```

Setting the option to `TRUE`, which is the default at the moment, uses the definition $\widetilde{\text{df}}$ for the degrees of freedom.

The following theorem shows that a selection bias occurs also if competing linear and smooth base-learners for the same covariate are specified. As for categorical covariates we consider the null case to derive the results.

Theorem 3.3 *Let* $\mathbf{x}$ *be a continuous covariate with design matrix* $\mathbf{X} = (\mathbf{1}, \mathbf{x})$. *A smooth effect for* $\mathbf{x}$ *is modeled using P-splines with the design matrix* $\mathbf{B} = (B_1(\mathbf{x}), \ldots, B_J(\mathbf{x}))$, *which consists of B-spline basis functions evaluated at the values of* $\mathbf{x}$. *Let* $\mathbf{K} = \mathbf{D}^\top\mathbf{D}$ *be the penalty matrix, where* $\mathbf{D}$ *is a difference matrix of order 2. The corresponding smoothing parameter is denoted by* λ. *Let* $\mathbf{u}$ *be the* $(n \times 1)$ *negative gradient vector arising in the*

first step of the boosting algorithm for a response variable $\mathbf{y}$ *of i.i.d. normally distributed random variables with variance* σ^2 *that is independent of* $\mathbf{x}$*, i.e.,* $\mathbf{u}$ *is simply the centered response variable. Let* $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{u}$ *and* $\hat{\boldsymbol{\beta}}_{\text{pen}} = (\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{K})^{-1}\mathbf{B}^\top\mathbf{u}$ *denote the effect estimates resulting from unpenalized linear base-learner, and from the P-spline base-learner, respectively. Define the difference of the residual sum of squares as* $\boldsymbol{\Delta} = (\mathbf{u} - \mathbf{X}\hat{\boldsymbol{\beta}})^\top(\mathbf{u} - \mathbf{X}\hat{\boldsymbol{\beta}}) - (\mathbf{u} - \mathbf{B}\hat{\boldsymbol{\beta}}_{\text{pen}})^\top(\mathbf{u} - \mathbf{B}\hat{\boldsymbol{\beta}}_{\text{pen}})$*. Then it holds that*

$$\mathbb{E}(\Delta) > 0 \quad (\textit{if } \lambda < \infty).$$

Proof for Theorem 3.3 The difference of the RSS is given by

$$\begin{aligned}
\boldsymbol{\Delta} = \text{RSS}_{\text{lin}} - \text{RSS}_{\text{pen}} &= \mathbf{u}^\top(\mathbf{I} - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top)^2\mathbf{u} - \mathbf{u}^\top(\mathbf{I} - \mathbf{B}(\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{K})^{-1}\mathbf{B}^\top)^2\mathbf{u} \\
&= \mathbf{u}^\top(\mathbf{I} - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{u} - \mathbf{u}^\top(\mathbf{I} - \mathbf{B}(\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{K})^{-1}\mathbf{B}^\top)^2\mathbf{u} \\
&= \mathbf{u}^\top\mathbf{Q}\mathbf{u}
\end{aligned}$$

with

$$\begin{aligned}
\mathbf{Q} &= [\mathbf{I} - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top] - [\mathbf{I} - \mathbf{B}(\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{K})^{-1}\mathbf{B}^\top]^2 \\
&= -\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top + 2\mathbf{B}(\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{K})^{-1}\mathbf{B}^\top - (\mathbf{B}(\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{K})^{-1}\mathbf{B}^\top)^2. \quad (3.8)
\end{aligned}$$

It holds that $\text{tr}(\mathbf{B}(\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{K})^{-1}\mathbf{B}^\top) = \text{tr}((\mathbf{I} + \lambda\widetilde{\mathbf{K}})^{-1}) = \sum_{j=1}^{J}(1 + \lambda d_j)^{-1}$, where $\widetilde{\mathbf{K}} = (\mathbf{B}^\top\mathbf{B})^{-1/2}\mathbf{K}(\mathbf{B}^\top\mathbf{B})^{-1/2}$ and d_j are the eigenvalues of $\widetilde{\mathbf{K}}$ (cf. Eilers and Marx 1996). For second order difference penalty matrices $\mathbf{D}$, two eigenvalues are equal to zero, all others eigenvalues d_j are positive. Thus

$$\begin{aligned}
\text{tr}(\mathbf{Q}) &= -2 + 2\sum_{j=1}^{J}(1 + \lambda d_j)^{-1} - \sum_{j=1}^{J}(1 + \lambda d_j)^{-2} \\
&\geq -2 + 2\sum_{j=1}^{J}(1 + \lambda d_j)^{-1} - \sum_{j=1}^{J}(1 + \lambda d_j)^{-1} \\
&= -2 + \sum_{j=1}^{J}(1 + \lambda d_j)^{-1} \geq 0,
\end{aligned}$$

where $\text{tr}(\mathbf{Q}) = 0$ if and only if $\lambda \to \infty$. As we assumed $\mathbb{E}(\mathbf{u}) = \mathbf{0}$ and $\text{cov}(\mathbf{u}) = \sigma^2\mathbf{I}$, by using (3.3) we obtain for a finite smoothing parameter λ

$$\mathbb{E}(\boldsymbol{\Delta}) = \sigma^2\,\text{tr}(\mathbf{Q}) > 0.$$

□

Hence, a selection bias in favor of smooth base-learners exists for finite smoothing parameter λ.

Lemma 3.4 *Assume that the assumptions from Theorem 3.3 hold. Let* $\boldsymbol{\mathcal{S}} = \mathbf{B}(\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{K})^{-1}\mathbf{B}^\top$ *be the smoother matrix of the P-spline. To make the linear and smooth term comparable*

$$\text{df} = \text{tr}(2\boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{S}}^\top\boldsymbol{\mathcal{S}})$$

needs to be controlled.

Proof for Lemma 3.4 To avoid bias, it must hold that $\mathbb{E}(\Delta) = 0$. Let $\mathbf{H} = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$ be the hat matrix of $\mathbf{X}$. With Equation (3.8) we get

$$\begin{aligned}
\mathbb{E}(\Delta) &= 0 \Leftrightarrow \\
\text{tr}(\mathbf{Q}) &= 0 \Leftrightarrow \\
\text{tr}(-\mathbf{H} + 2\boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{S}}^\top\boldsymbol{\mathcal{S}}) &= 0 \Leftrightarrow \\
\text{tr}(2\boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{S}}^\top\boldsymbol{\mathcal{S}}) &= \text{tr}(\mathbf{H})
\end{aligned}$$

Hence, $\text{df} = \text{tr}(2\boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{S}}^\top\boldsymbol{\mathcal{S}})$ must be made comparable to $\text{tr}(\mathbf{H}) = \text{rank}(\mathbf{X})$, which is the number of independent parameters in the linear model, i.e., the degrees of freedom for the linear base-learner.

□

Thus, the appropriate degrees of freedom have the same form as for categorical effects. Controlling $\text{df} = \text{tr}(2\boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{S}}^\top\boldsymbol{\mathcal{S}})$ can be seen as an improved version of the model selection scheme that was proposed by Kneib *et al.* (2009) who used $\widetilde{\text{df}}$ instead of df. Following these lines, one should specify equal degrees of freedom for *all* base-learners if unbiased model and variable selection are the goal. As we do not include an intercept in the base-learners but specify a separate base-learner for the intercept, the natural choice for these common degrees of freedom is one single free parameter as it appears for a linear least squares base-learner of one single continuous covariate. This can easily be achieved for categorical covariates by setting the smoothing parameter to an appropriate value (see below).

In contrast to categorical covariates, we cannot make the degrees of freedom arbitrarily small for P-splines, even with λ approaching infinity, since a polynomial of order $d - 1$ remains unpenalized by a dth order difference penalty (Eilers and Marx 1996). This is the so called null space. As we usually apply second order

differences, a linear effect (with intercept) remains unpenalized and thus $df \geq 2$ for all λ. This can also be seen from the eigenvalue decomposition in the proof of Theorem 3.3. To be able to specify a base-learner with one degree of freedom, a reparameterization as described by Kneib *et al.* (2009) is needed. The smooth (jth) base-learner is decomposed into parametric parts for the unpenalized polynomial and a smooth deviation from this polynomial

$$g_j(x) = \beta_{0,j} + \beta_{1,j}x + \ldots + \beta_{d-1,j}x^{d-1} + g_{\text{centered}}(x), \tag{3.9}$$

where only $g_{\text{centered}}(x)$ is modeled using the reparameterized, centered P-spline base-learner. Now, we can specify separate base-learners for each parametric effect and a base-learner with one degree of freedom for the smooth deviation from the polynomial. For more details on the technical realization and further implications of the decomposition we refer to the article by Kneib *et al.* (2009).

As mentioned above, we propose to specify the smoothness of all base-learners via the degrees of freedom. We use an initial value df_{init} for each penalized base-learner and solve $\text{tr}\left(2\boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{S}}^\top\boldsymbol{\mathcal{S}}\right) = df_{\text{init}}$ for λ. The following lemma provides a convenient, numerically efficient way to compute the degrees of freedom and therefore to determine the corresponding λ.

Lemma 3.5 *Let $\boldsymbol{\mathcal{S}} = \mathbf{X}(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{K})^{-1}\mathbf{X}^\top$ be the smoother matrix of $\mathbf{x}$ with non-negative definite, symmetric penalty matrix $\mathbf{K}$. Let $\mathbf{R}^\top\mathbf{R} = \mathbf{X}^\top\mathbf{X}$ be the Cholesky decomposition of the cross-product of the design matrix. Then, the degrees of freedom $\text{df}(\lambda) = \text{tr}\left(2\boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{S}}^\top\boldsymbol{\mathcal{S}}\right)$ are equal to*

$$\text{df}(\lambda) = 2\sum_{j=1}^{M}\frac{1}{1+\lambda d_j} - \sum_{j=1}^{M}\frac{1}{(1+\lambda d_j)^2} \tag{3.10}$$

where $d_j \geq 0$ are the M non-negative singular values of $\mathbf{R}^{-\top}\mathbf{K}\mathbf{R}^{-1}$.

The proof of Lemma 3.5 can be derived using the Demmler–Reinsch orthogonalization (cf. Ruppert 2002). As Lemma 3.5 only requires the penalty matrix $\mathbf{K}$ to be non-negative definite and symmetric, we can use (3.10) to compute the degrees of freedom for all base-learners considered in this thesis.

3.2. Variable Selection Bias under Test — A Simulation Study

3.2.1. Biased Selection of Categorical Covariates

To empirically evaluate the bias introduced by categorical covariates with potentially many categories, we examine two situations: the null case, where *none* of the covariates has an influence on the response and a set of power cases, where a subset of the covariates influences the response. The null case is used to examine the selection bias and the proposed solution formally, whereas the power case is used to verify that the model also benefits from the proposed correction in case of influential covariates.

In the null case, the response is simply i.i.d. normally distributed, $y_i \overset{i.i.d.}{\sim} \mathcal{N}(0,1)$, $i = 1, \ldots, n$ but we fit a model with structure

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}_1\gamma_1 + \boldsymbol{\varepsilon}, \tag{3.11}$$

where the $(n \times p)$ matrix $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p)$ is formed of continuous covariates, $\boldsymbol{Z}_1$ is the dummy coded design matrix of a categorical covariate z_1, and $\boldsymbol{\beta}$ and γ_1 are the corresponding parameter vectors. The $p = 25$ continuous covariates were sampled as realizations $X_1, \ldots, X_p \overset{i.i.d.}{\sim} U[0,1]$ and the categorical covariate z_1, with varying numbers of categories $n_{\text{cat}} \in \{2, \ldots, 10\}$, was sampled from a discrete uniform distribution on $\{1, \ldots, n_{\text{cat}}\}$.

In the first power case, the response depends on five continuous covariates; the remaining 20 continuous covariates and the categorical covariate have no influence on y. Again, we fit a model with structure (3.11).

In the second power case setting, we add a second, *informative* categorical covariate z_2 with the same numbers of categories as z_1, which is sampled i.i.d. from a discrete uniform distribution as used for z_1. The response now depends on five continuous covariates and on the categorical covariate z_2. The model is fitted according to the structure

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}_1\gamma_1 + \boldsymbol{Z}_2\gamma_2 + \boldsymbol{\varepsilon} \tag{3.12}$$

where $\boldsymbol{Z}_2$ and γ_2 are the design matrix and the coefficient vector for z_2, respectively. The effects of the categories for z_2 do not exceed two and hence are comparable to

the other effects in (3.11). Table 3.1 summarizes all simulation settings.

For both the null case and the power cases, the sample size was set to $n = 150$ and the error terms are i.i.d. samples from $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ with σ^2 chosen such that the fraction of explained variance is $R^2 \approx 0.3$. In the power cases we simulated $B = 100$ data sets while $B = 1000$ simulation replicates were considered in the null case. Changes of the sample size ($n = 50, 500, 1000$), the number of covariates ($p = 10, 100$), the sampling distribution (standard normal and binomial distribution), or the fraction of explained variance ($R^2 = 0.5$) led to qualitatively same results in the tested scenarios. The stopping iteration $\widehat{m}_{\text{stop,opt}}$ was determined based on an independent test sample of size 750.

Table 3.1.: Overview of different simulation schemes for categorical covariates z_1 and z_2 (upper part) and continuous covariate z_1 (lower part).

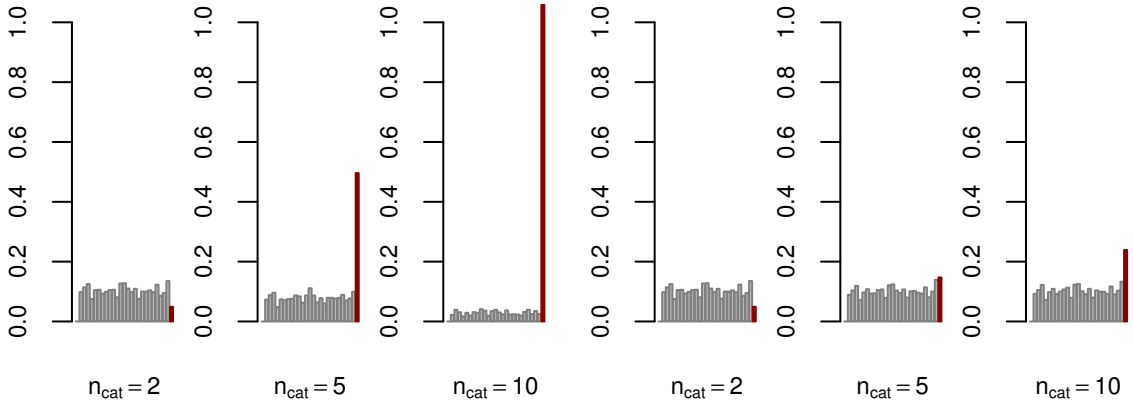
		Effects for $x_1, \ldots, x_{25}$	Effects for z_1 (and z_2)
Null model		$\boldsymbol{\beta} = (0, \ldots, 0)^\top$	$\gamma_1 = (0, \ldots, 0)^\top$
Power case 1	z_1 non-influential	$\boldsymbol{\beta} = (-2, -1, 1, 2, 3, 0, \ldots, 0)^\top$	$\gamma_1 = (0, \ldots, 0)^\top$
Power case 2	z_1 non-influential	$\boldsymbol{\beta} = (-2, -1, 1, 2, 3, 0, \ldots, 0)^\top$	$\gamma_1 = (0, \ldots, 0)^\top$
	z_2 influential		$\gamma_2 = (\frac{2}{n_{\text{cat}}/2}, \frac{3}{n_{\text{cat}}/2}, \ldots, \frac{n_{\text{cat}}}{n_{\text{cat}}/2})^\top$
Null model		$\boldsymbol{\beta} = (0, \ldots, 0)^\top$	$f_z(z_1) \equiv 0$
Power case 1	z_1 non-influential	$\boldsymbol{\beta} = (-2, -1, 1, 2, 3, 0, \ldots, 0)^\top$	$f_z(z_1) \equiv 0$
Power case 2	z_1 linear effect	$\boldsymbol{\beta} = (-2, -1, 1, 2, 3, 0, \ldots, 0)^\top$	$f_z(z_1) = 1.5 z_1$
Power case 3	z_1 smooth effect	$\boldsymbol{\beta} = (-2, -1, 1, 2, 3, 0, \ldots, 0)^\top$	$f_z(z_1) = \sin(-(2z_1)^2 - 0.6(2z_1)^3)$

All models were fitted using the `gamboost` function from the R package **mboost** and one base-learner was specified per model component. All continuous covariates were standardized since we use base-learners without intercept (and specify an additional base-learner for the intercept; see Section 2.3.1). For the base-learner of categorical covariates, we considered either unpenalized or penalized least squares base-learners. In the remainder of this chapter we use the terms "unpenalized model" and "penalized model" to refer to models where the base-learner for the categorical effect is unpenalized and ridge penalized, respectively.

Null Case Here, the criterion for the variable selection bias is the selection frequency of the base-learners averaged over all simulation runs. In the null model case, i.e., in the case where no covariate has an influence on the response, a sensible selection procedure should not prefer one base-learner over another but should randomly select any of the non-informative covariates.

The selection rates in models with and without penalized base-learners can be found in Figure 3.1. Obviously, the selection frequency of the categorical covariate

increases with increasing n_{cat} if no ridge penalty is applied. When applying the ridge penalty, the selection frequency of the categorical covariate becomes comparable to the selection frequency of the continuous covariates. Hence, we can conclude that using penalized categorical base-learners improves the boosting algorithm in the null case with respect to the selection rates.



(a) Unpenalized Model (b) Ridge Penalized Model

Figure 3.1.: *Null Model:* Average selection frequencies of base-learners for $n_{\text{cat}} = \{2, 5, 10\}$ in the "optimal step" $\widehat{m}_{\text{stop,opt}}$ without and with ridge penalty. The last bar in each graph represents the selection frequency of the categorical covariate.

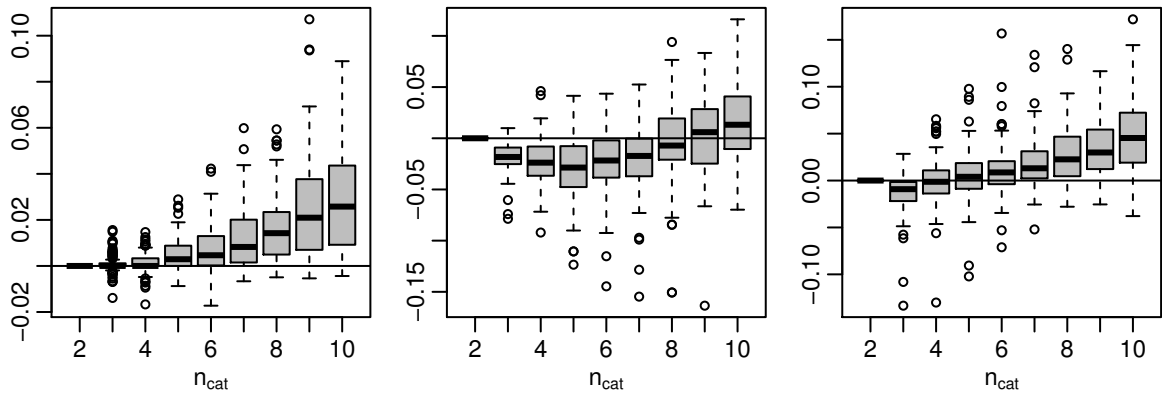
Power Cases Applying the ridge penalty to correct for the selection bias shrinks the parameter estimates such that the resulting dfs are all equal to 1 (independently of n_{cat}).

Unlike in the null case, one cannot assess the selection bias in power cases directly via the selection frequency as we have a complicated mixture of effect size, type of effect, etc. Therefore, to assess the quality of the resulting models and the effect of the ridge penalty in power cases, we use the mean squared prediction error (MSE) of the coefficients

$$\text{MSE} = \frac{1}{\widetilde{p}} \sum_{j=1}^{\widetilde{p}} (\hat{\beta}_j - \beta_j)^2, \tag{3.13}$$

where $\widetilde{p}$ is the number of coefficients including those for the categorical effect(s).

Figure 3.2(a) shows that the MSE is decreased in the first power case when using the penalized base-learner for z_1. In the the second power case, an additional informative categorical covariate z_2 is included in the model. Figure 3.2(b) shows



(a) Power Case 1 (b) Power Case 2: with z_2 (c) Power Case 2: without z_2

Figure 3.2.: *Power Cases:* Boxplots represent $\text{MSE}_{\text{unpenalized}} - \text{MSE}_{\text{penalized}}$ in the first power case (left) and in the second power case, where the MSE is computed with (middle) and without (right) the influential, categorical covariate z_2.

the differences of the MSEs for the models with unpenalized and penalized base-learners for the categorical covariates. As the boxes cover zero or sometimes are even located below, the "unpenalized model" seems to be better. However, the figures represent a mixture of two different, competing effects: The effect of the non-influential categorical covariate is shrunken toward zero with the penalized base-learner (decrease of MSE). At the same time, the effect of the influential covariate is also shrunken toward zero introducing an additional bias (increase of MSE) leading to the apparent U-shape in Figure 3.2(b). However, this is true for *any* penalization approach. Looking at Figure 3.2(c), one can see the differences of MSEs, where the influential, categorical covariate z_2 is excluded (for the calculation of the MSE but *not* for the estimation of the model). Here one clearly sees the superiority of the penalized approach. One can conclude that penalization has the advantage to reduce the selection bias for non-informative covariates and additionally shrinks the effects of influential covariates. If one deals with high-dimensional settings, and variable selection and shrinkage are desired beforehand, the ridge penalty is exactly what one would like to apply.

Comparison to Stepwise Linear Model To benchmark the bias-corrected approach, we compared the resulting MSEs to the mean squared errors of a linear model with forward stepwise selection based on AIC. In our settings, the boosting models are better on average than the stepwise models. In the first power case and in the second power case (MSE computed without z_2) the boosting model was better in more than 75% of the cases with respect to the mean squared prediction

error. If we compute the MSE with the informative categorical covariate z_2, the shrinkage effect decreases the superiority of boosting a bit. Nevertheless, boosting is still superior to stepwise regression in the majority of the cases.

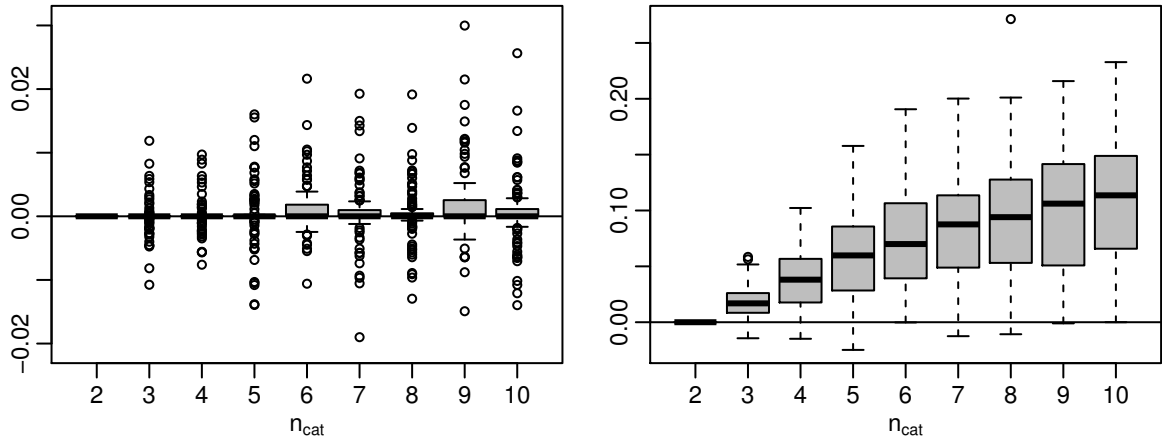
Penalized Ordinal Base-learners If categorical covariates are ordinal, one can use ordinal penalized base-learners (see Sec. 2.3.1) instead of ridge penalized base-learners. To assess the properties of this penalty, we used the same simulation setting as for unordered covariates (see Table 3.1).

To summarize the results (see Appendix C.1), we can conclude that ordinal penalized base-learners show basically the same behavior as ridge penalized base-learners: In the null case, the penalized ordinal base-learners reduce the selection bias such that the selection frequencies of all base-learners are approximately equal (Fig. C.1).

Looking at the performance of the penalty approach for ordinal covariates, we can state an overall improvement compared to the unpenalized model. In the first power case, the MSE is improved in comparison to the unpenalized model, and ordinal penalized base-learner and ridge penalized base-learner are overall comparable (see Figure 3.3(a)). In the second power case with an additional, informative covariate z_2, the ridge penalized model shows an improvement compared to the unpenalized model, but the ordinal penalty offers a further improvement over the ridge penalty, which does not exploit the ordinal structure of z_2. This can be seen in Figure 3.3(b), where we see another increase in the differences of the MSE. This is possibly due to a weaker penalization of the higher categories, which have a bigger effect (cf. Table 3.1): The ridge penalty shrinks all coefficients equally toward zero, whereas the ordinal penalty just shrinks the increase with respect to the preceding category toward zero. Hence, we can conclude that it is preferable to exploit the ordinal structure of the covariates if possible and only use ridge penalized base-learners if no ordinal structure can be assumed.

3.2.2. Biased Model Selection

To evaluate the preferred selection of smooth effects compared to linear effects, we again examined the null case and a set of power cases. The data were generated similarly to the categorical case (Sec. 3.2.1), where the categorical covariate was



(a) Power Case 1 (b) Power Case 2

Figure 3.3.: *Comparison of model with ridge penalized and ordinal penalized base-learner:* Boxplots represent $\text{MSE}_{\text{ridge penalized}} - \text{MSE}_{\text{ordinal penalized}}$ with non-influential categorical covariate (left) and with additional influential categorical covariate (right).

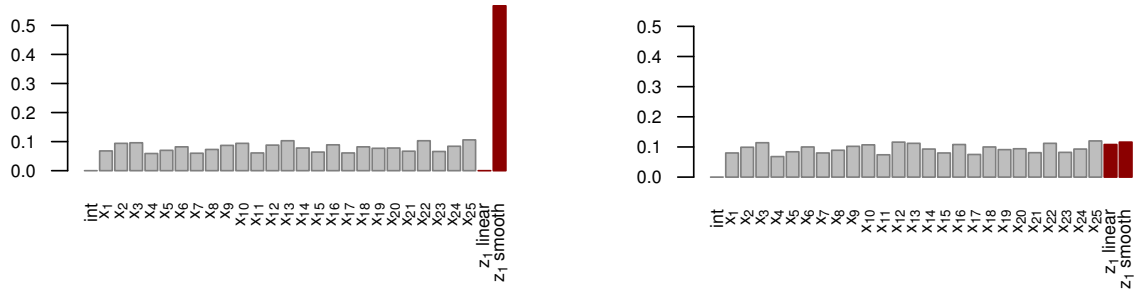
replaced by a continuous covariate. This leads to the model

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + f_z(\boldsymbol{z}_1) + \boldsymbol{\varepsilon} \tag{3.14}$$

with the $(n \times 1)$ dimensional response vector $\boldsymbol{y}$, the $(n \times p)$ matrix $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p)$, and the corresponding parameter vector $\boldsymbol{\beta}$ of length p (see Table 3.1). The function $f_z(\cdot)$ can be of different nature, e.g., it can be a linear function, a smooth function, or it can be a function that is equal to zero for all realizations $\boldsymbol{z}_1$ (cf. Table 3.1). The $p = 25$ continuous covariates were all sampled i.i.d. from $U[0, 1]$, as well as the continuous covariate of interest z_1. The error term was sampled again from $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ with σ^2 such that the fraction of explained variance is $R^2 \approx 0.3$. To empirically evaluate the selection bias introduced by smooth terms compared to linear terms we simulated $B = 1000$ (null case) and $B = 100$ (power cases) data sets with $n = 150$ observations from model (3.14). The optimal stopping iteration $\widehat{m}_{\text{stop,opt}}$ was determined on an independent test sample of size 750. Changes of the sampling distribution (standard normal distribution), explained variance ($R^2 = 0.5$), sample size ($n = 50, 500, 1000$) or number of covariates ($p = 10, 100$) were examined and led to essentially the same results.

Null Case We use the null case (i.e., the case where no variable influences the response) to show that the model selection is biased if no correction is applied. We specify a linear base-learner for z_1, and additionally a smooth base-learner with

$df = 4$. A reasonable model selection procedure should not prefer either of the two modeling alternatives for z_1. However, the smooth base-learner for z_1 with four degrees of freedom is highly preferred due to the increased flexibility. At the same time the linear base-learner for z_1 is never selected (see Figure 3.4(a)). If we apply the decomposition (3.9) and specify one degree of freedom for each base-learner, the selection bias in favor of smooth effects vanishes completely (see Figure 3.4(b)). In conclusion, model selection in the naive specification is highly biased whereas the bias is strongly reduced in the case of equally flexible base-learners.



(a) Model without Decomposition

(b) Model with Decomposition

Figure 3.4.: *Null Case:* Mean selection frequency of base-learners in the "optimal step" $\widehat{m}_{\text{stop,opt}}$ with $df = 4$ (left) and decomposition (with $df = 1$; right).

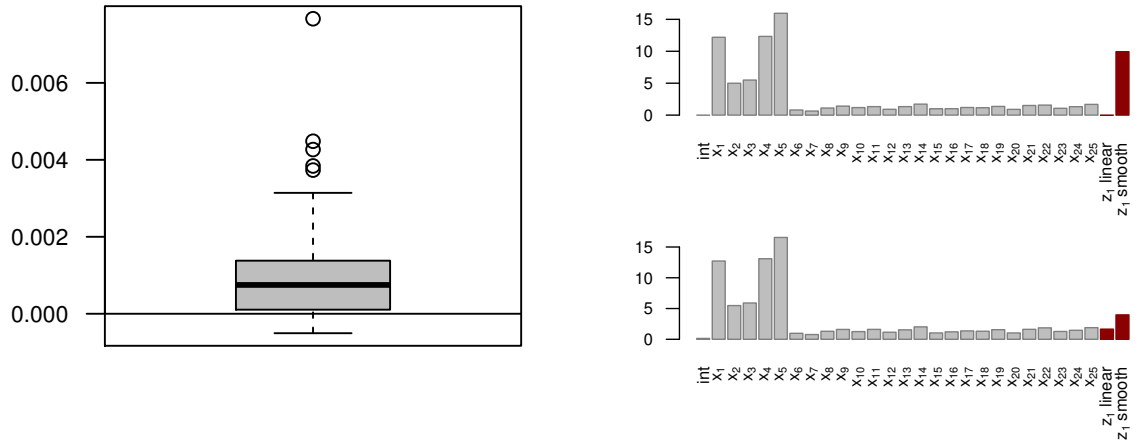
Power Cases To measure the performance of the penalized approach and gain some insight into the nature of the bias that is introduced by competing linear and smooth base-learners we use the following L_2-norm

$$\Delta^{L_2}_{\text{partial},i} = \int_{\min(\boldsymbol{x}_i)}^{\max(\boldsymbol{x}_i)} [\hat{f}_i(\tilde{x}) - f_i(\tilde{x})]^2 d\tilde{x}.$$

The norm measures the deviation of the estimated partial function from the true function. For numerical evaluation, the model was predicted on a fine, equidistant grid, and the trapezoidal rule was applied to evaluate the integral. As a summary measure we use the mean L_2-deviation $\Delta^{L_2} = \tilde{p}^{-1} \sum_{i=1}^{\tilde{p}} \Delta^{L_2}_{\text{partial},i}$, where $\tilde{p}$ is the number of covariates. Thus, Δ^{L_2} is an extension of the MSE (3.13) (see Sec. 3.2.1) to smooth effects. Additionally, we examined the selection frequencies of the corresponding base-learners.

Power Case 1 From Figure 3.5(a) we can conclude that the decomposition improves the model when a smooth base-learner is used but the respective covariate

is not influential. Figure 3.5(b) shows that the decomposition leads to a decreased selection of the smooth base-learner and an increase of the selection of the competing linear base-learner, which implies a reduction of the model selection bias. However, the bias does not seem to be completely removed.

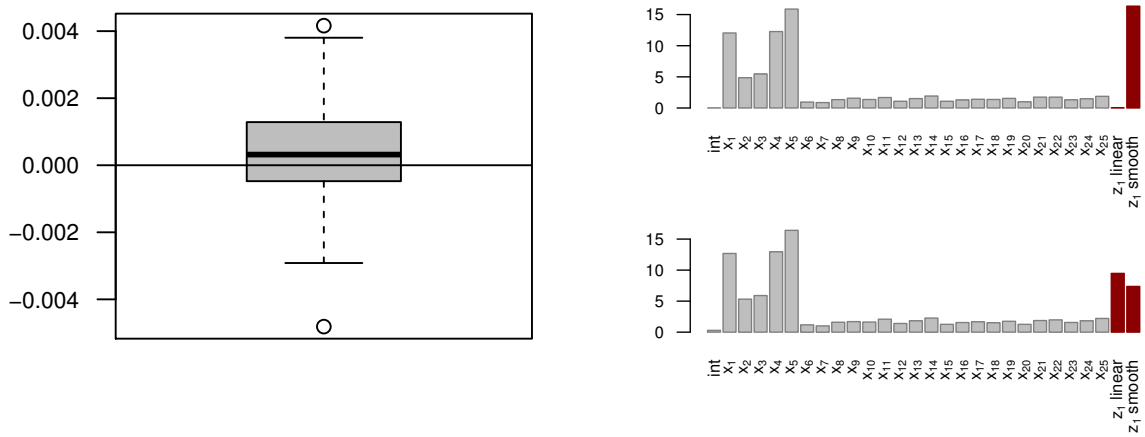


(a) Partial Deviation (b) Selection Frequency

Figure 3.5.: *Power Case 1:* Partial deviation $\Delta^{L_2}_{\text{model}} - \Delta^{L_2}_{\text{model with decomposition}}$ (left) and mean selection frequency of base-learners in the "optimal step" $\widehat{m}_{\text{stop,opt}}$ with $df = 4$ (upper) and decomposition (with $df = 1$; lower). The bars 2 to 6 represent the influential covariates $x_1, \ldots, x_5$.

Power Case 2 In the following paragraph, $f_Z(z_1) = \gamma z_1$, with $\gamma = 1.5$, i.e., the covariate of interest has a linear effect. Figure 3.6(a) shows that the model with decomposition is almost always better (or comparable) to the model without decomposition with respect to the deviations of the partial fits, i.e., the fitted functions seem to be better in many cases. At the same time, we can observe a considerable increase of the selection frequency of the linear base-learner for z_1 if the decomposition is applied. We can still observe an reasonable amount of selections for the smooth base-learner. However, the linear base-learner is selected more often than the smooth base-learner. Note that, without the decomposition, the linear base-learner is never selected. Hence, the true nature of the underlying effect is completely missed in this case.

Power Case 3 To investigate the behavior in the case where a smooth effect for z_1 is present, we chose $f_Z(z_1) = \sin(-(2z_1)^2 - 0.6(2z_1)^3)$. We thus have a combination of a sine form with a polynomial of z_1. From Figure 3.7(a) we can conclude that the mean L_2-deviation Δ^{L_2} is slightly larger with model decomposition but an

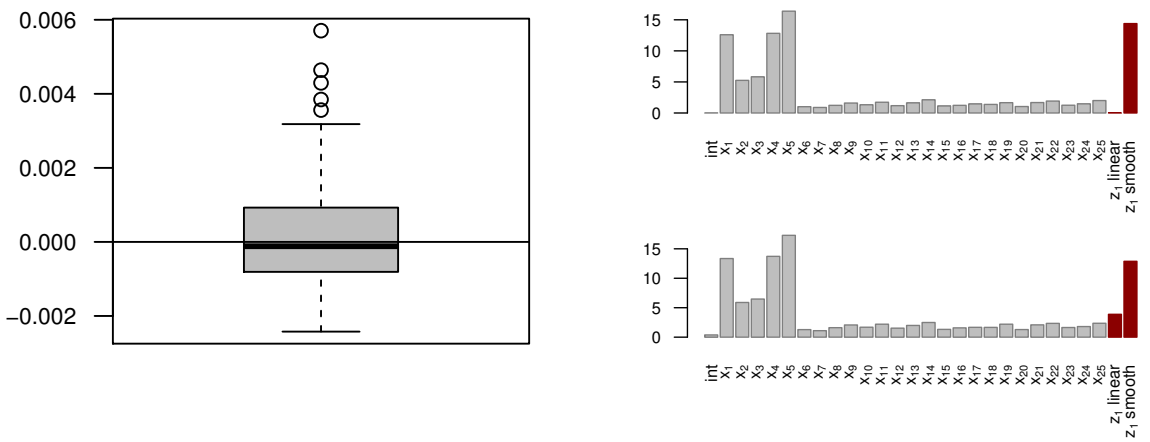


(a) Partial Deviation (b) Selection Frequency

Figure 3.6.: *Power Case 2:* Partial deviation $\Delta^{L_2}_{\text{model}} - \Delta^{L_2}_{\text{model with decomposition}}$ (left) and mean selection frequency of base-learners in the "optimal step" $\widehat{m}_{\text{stop,opt}}$ with $df = 4$ (upper) and decomposition (with $df = 1$; lower).

considerable overlap with zero exists. Other non-linear functions were investigated and showed similar results. Depending on the effect size and the functional form the box lies more or less around zero. This shows that the model with decomposition and without decomposition are almost equally good in the case of smooth effects. The non-inferiority of the "decomposition model" is expected, as the model without decomposition has a flexible base-learner directly customized to smooth effects. Again we see an increased selection frequency of the linear effect for z_1 if we apply the decomposition, which can be partially ascribed to the slight linear effect that is present for z_1 (see Figure 3.7(b)). Despite the fact that the model with P-spline decomposition reduces the selection of smooth effects, the model without decomposition (and thus greater flexibility to model the smooth effect) is not better with respect to the partial deviation. This might be partially attributed to the weaker base-learner in the model with decomposition which leads to a more stable estimate as can be seen in Figure 3.8.

Summing up, we could observe that models with the P-spline decomposition (3.9), where all base-learners are specified with one degree of freedom, performed better than models where a linear and a competing smooth base-learner for z_1 with four degrees of freedom are specified.



(a) Partial Deviation

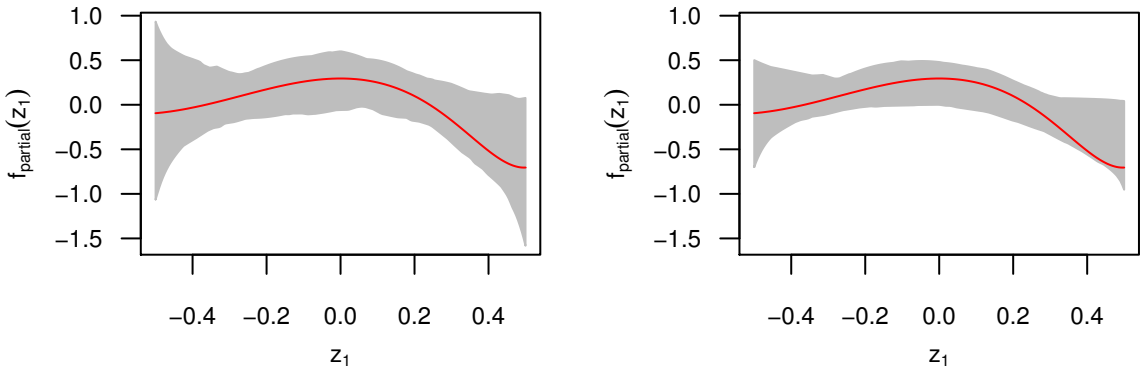
(b) Selection Frequency

Figure 3.7.: *Power Case 3:* Partial deviation $\Delta^{L_2}_{\text{model}} - \Delta^{L_2}_{\text{model with decomposition}}$ (left) and mean selection frequency of base-learners in the "optimal step" $\widehat{m}_{\text{stop,opt}}$ with $df = 4$ (upper) and decomposition (with $df = 1$; lower).

3.3. Forest Health Prediction

In our application, we consider models describing the forest health status. The aim is to identify predictors of the health status of beeches, which is measured in terms of the degree of defoliation. The data originates from yearly visual forest health inventories carried out from 1983 to 2004 in a northern Bavarian forest district. The data consists of 83 plots of beeches within a 15 km $\times$ 10 km area with a total of $n = 1793$ observations. The response is a dichotomized version of the defoliation index indicating defoliation above 25%. Obviously, the data set combines a longitudinal and a spatial structure. An overview of the covariates is given in Table 3.2.

Previous analysis (Kneib and Fahrmeir 2006, 2010, Kneib *et al.* 2009) resulted in models that contained categorical covariates, as well as linear and smooth effects of continuous covariates. Additionally, a spatial effect and a random effect for the plots could be identified. When considering a structured additive regression model of comparable complexity in a naive boosting implementation, biased model selection of smooth model components as well as categorical covariates with several categories is likely to occur. In the following, we will apply the methodology developed in this chapter to achieve bias corrected variable and model selection for the forest health data. Since the outcome is binary, we minimize the negative binomial log-likelihood, i.e., we fit a structured logit model to the data. We consider



(a) Without Decomposition (b) With Decomposition

Figure 3.8.: *Power Case 3*: True (partial) effect of z_1 together with point-wise 95% variability intervals based on simulation replicates.

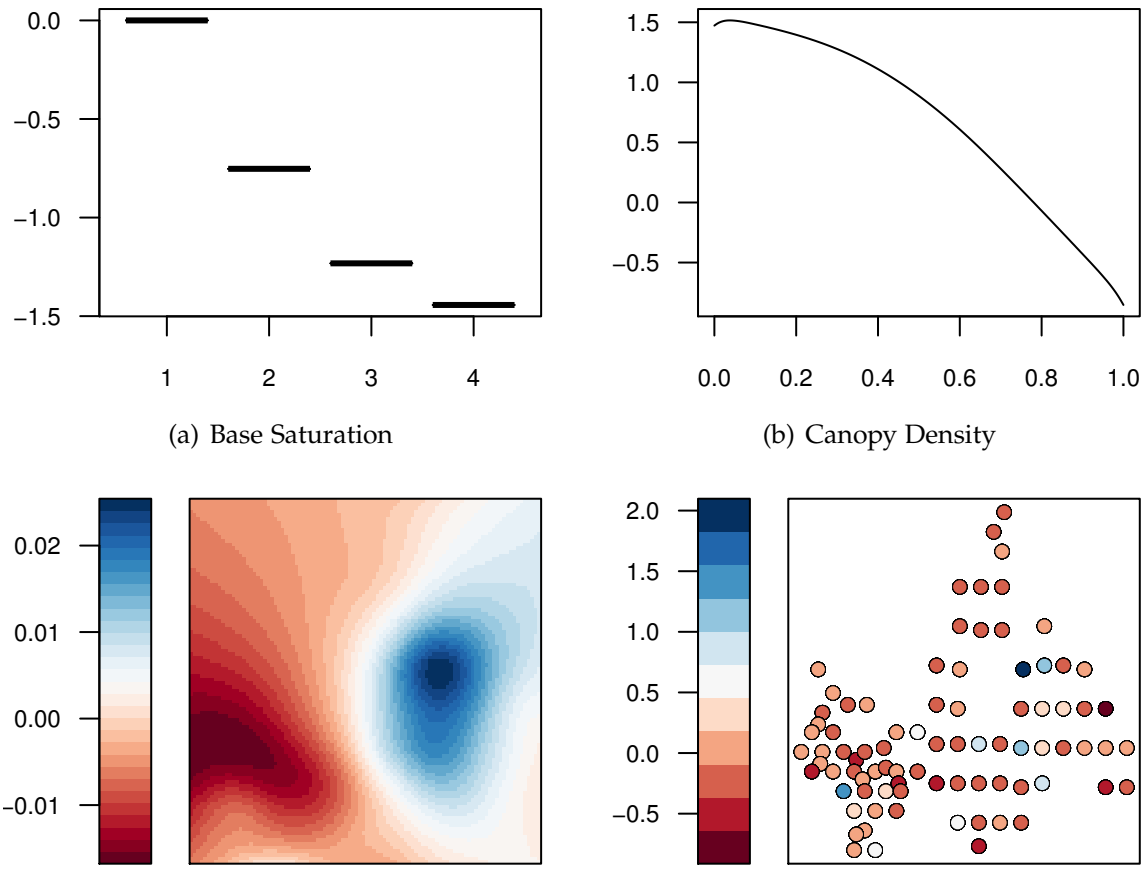
a candidate model with the additive predictor

$$\begin{aligned}\eta &= \boldsymbol{x}^\top \boldsymbol{\beta} + f_1(\text{ph}) + f_2(\text{canopy}) + f_3(\text{soil}) + f_4(\text{inclination}) \\ &\quad + f_5(\text{elevation}) + f_6(\text{time}) + f_7(\text{age}) + f_8(s_1, s_2) + b_{\text{plot}},\end{aligned} \tag{3.15}$$

where $\boldsymbol{x}$ contains the parametric effects of the categorical covariates fertilization, stand, humus and saturation. The ordinal covariates humus and saturation are modeled using penalized ordinal base-learners, whereas the other categorical covariates are modeled using ridge penalized base-learners. The smooth effects $f_1, \ldots, f_7$ are specified as a combination of linear base-learners and univariate cubic P-splines with 20 inner knots and a second order difference penalty. For the spatial effect f_8 we assumed bivariate cubic P-splines with first order difference penalties and 12 inner knots for each of the directions. Finally, the plot-specific effects b_{plot} are represented by a ridge-type "random effects" base-learner with fixed degrees of freedom (see Kneib *et al.* 2009, for details). All continuous covariates were centered as suggested in Section 2.3.1. To correct for the selection bias, one degree of freedom was assigned to each single base-learner including the spatial and random effect base-learners.

The optimal stopping iteration was estimated via stratified bootstrap, i.e., we randomly selected plots (with replacement) and not single observations, as the plots can be considered as the observational units. The resulting model included five covariates, and additionally the spatial information and the random intercept for the plots. Fertilization (represented as a binary indicator for the application of fertilization) was included in the model with a negative effect on defoliation

($\beta_{\text{fert}} = -0.77$). Age and calendar time (both included as linear base-learners) had positive effects on defoliation ($\beta_{\text{age}} = 0.016$ and $\beta_{\text{year}} = 0.070$). This means that the severity of defoliation increases each year if the other covariates (including the age of the trees in the plot) are kept fixed. The estimated effects of base saturation, which was modeled using a penalized ordinal base-learner, and canopy density, which was included as a combination of a linear and a smooth base-learner, can be found in the upper part of Figure 3.9. The spatial effect was included in the model but is clearly dominated by the spatially unstructured, plot-specific effect (Figure 3.9). The remaining six covariates were not included in the final model. In summary, our boosting framework allows to fit a complex model, comprising many different kinds of effects, while interpretable and biologically meaningful results are obtained. Code to fit the forest health prediction model can be found in Appendix B.3.



(a) Base Saturation

(b) Canopy Density

(c) Spatial Effect

(d) Random Effect

Figure 3.9.: Partial effects of base saturation, canopy density, spatial effect and random effect (without spatial variation) on forest health.

3.3.1. A Comparison with the Uncorrected Model

To benchmark our approach, we compared the bias-corrected modeling approach with the uncorrected approach. We used the same candidate model (3.15) but specified each smooth base-learner and the random effect base-learner with four degrees of freedom and added the categorical base-learners unpenalized. We used stratified 10-fold cross-validation where the corrected as well as the uncorrected model were fitted on the learning sample. The optimal stopping iteration within each *learning sample* was estimated by stratified bootstrap separately for each model (i.e., within each learning sample). Each of the 10 *test samples* was used to determine the out-of-bag risk, i.e., the negative log-likelihood. We observed that the corrected model was superior to the uncorrected model in 80% of the cases with respect to the prediction error measured by the negative log-likelihood (see Figure 3.10). The improvement in the prediction error is relatively small but visible. Thus, using equal degrees of freedom resulted in an improved prediction accuracy, and it corrected for biased selection of base-learners.

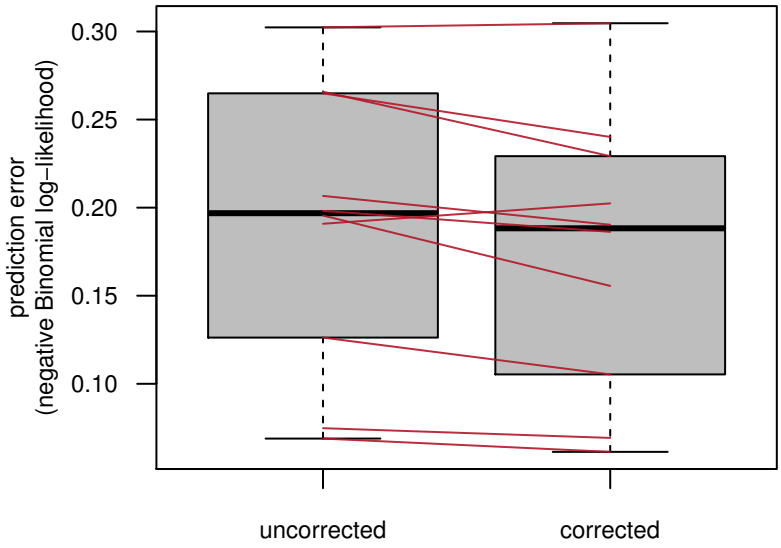


Figure 3.10.: Comparison of the risk for each of the 10 test samples, where the model was fitted without the correction (left) and with the correction (right). For each learning sample, each model was tuned separately by 25-fold cross-validation on the learning sample. The red lines connect the paired bootstrap samples.

Table 3.2.: Description of the covariates for the forest health data. All continuous covariates were centered before they were included in the model, categorical covariates were dummy coded with the first category as reference.

Covariate	Description
age	average age of trees at the observation plot in years (continuous, $7 \leq$ age ≤ 234)
time	calendar time (continuous, $1983 \leq$ time ≤ 2004)
elevation	elevation above sea level in meters (continuous, $250 \leq$ elevation ≤ 480)
inclination	inclination of slope in percent (continuous, $0 \leq$ inclination ≤ 46)
soil	depth of soil layer in centimeters (continuous, $9 \leq$ soil ≤ 51)
ph	ph-value at 0-2cm depth (continuous, $3.28 \leq$ ph ≤ 6.05)
canopy	density of forest canopy in percent (continuous, $0 \leq$ canopy ≤ 1)
humus	thickness of humus layer in 5 categories (ordinal, higher categories represent higher proportions)
saturation	base saturation in 4 categories (ordinal, higher categories indicate higher base saturation)
stand	type of stand (categorical, $-1 =$ mixed forest, $1 =$ deciduous forest)
fertilization	fertilization (categorical, $-1 =$ no, $1 =$ yes)

3.4. Concluding Remarks

Component-wise boosting techniques offer the possibility to fit a wide range of models with intrinsic variable and model selection. To reduce selection bias of the base-learners, equal degrees of freedom need to be assigned to all base-learners. This can be achieved by using penalized least squares base-learners. We considered ridge penalized base-learners for categorical covariates, penalized base-learners with a ridge penalty applied to the differences of adjacent coefficients for ordinal covariates, and P-spline base-learners for smooth model terms. For the latter, an additional reparameterization step has to be applied to differentiate between an unpenalized polynomial and the penalized deviation (Kneib *et al.* 2009).

For all base-learners, degrees of freedom can be specified using the definition $\text{df} := \text{tr}\left(2\boldsymbol{\mathcal{S}} - \boldsymbol{\mathcal{S}}^\top \boldsymbol{\mathcal{S}}\right)$. This definition is tailored for the comparison of residual sums of squares (RSS) and also appeared naturally from our theoretical considerations about the selection bias. Specifying equal degrees of freedom for all modeling components to achieve bias-corrected model selection can be easily incorporated in all component-wise functional gradient descent boosting approaches. These are not restricted to additive models with L_2-loss, but cover a wide class of models includ-

ing generalized additive models and survival models. In Section 3.3 we presented a case-study on forest health where a complex spatial regression model was used to classify defoliation of trees. The model was built making use of the approach proposed in this chapter. This illustrates that the penalization approach can be extended to more sophisticated models, which include spatial effects or plot-specific random effects (Kneib *et al.* 2009).

In contrast to our approach, most of the literature dedicated to likelihood-based boosting (based on Fisher scoring) currently advocates to use one single smoothing parameter λ for the penalty (e.g., Tutz and Binder 2006), or even to decrease the penalty for some covariates to prefer them in the selection step without a thorough theoretic reasoning, yielding a faster convergence toward the maximum (partial) likelihood estimates (Binder and Schumacher 2008). Obviously, one could also define the smoothing parameters in this framework such that the resulting degrees of freedom are equal and thus obtain bias-corrected variable and model selection procedures based on likelihood-based boosting. However, one problem arises in this context: The degrees of freedom for a fixed smoothing parameter change over the subsequent boosting iterations (Hofner, Hothorn, and Kneib 2008), for example, due to changes in the working weights for GLMs.

Alternatively, one could think of altering the goodness-of-fit criterion that is used to determine the best fitting base-learner in each step instead of making the competing base-learners comparable by specifying equal degrees of freedom to achieve unbiased base-learner selection. Examples include penalized alternatives to the RSS such as AIC and BIC. In the context of likelihood-based boosting, Binder and Schumacher (2008) proposed to use the *penalized* partial likelihood or the AIC as selection criteria. Another idea could be to use F-tests, which also account for the different degrees of freedom. However, our experience from simulation studies shows that such approaches do not work well when comparing base-learners with very different degrees of freedom. Criteria such as AIC and BIC are composed of two parts: one that measures the fit via the likelihood and one term for the penalty. In the linear regression model, which applies here, the AIC is defined as $\text{AIC} = n/\log(\text{RSS}/n) + 2\,df$. The degrees of freedom df are the initially defined degrees of freedom of each base-learner. In the course of the boosting procedure, the information still left in the data decreases sequentially and consequently the RSS is forced to decrease. Thus, the criterion will be dominated by the penalty term. The selection of base-learners in later iterations is based solely on the penalty while neglecting the fit to the data. Hence, base-learners with fewer degrees of freedom

are preferred, more flexible terms are even completely ignored in later iterations. The same reasoning applies for BIC as well as F-tests. The problem that arises here is that the penalty is not chosen adaptively to the maximum variance that *could* be explained.

A possible solution for this problem is given in Bühlmann and Yu (2006). The authors propose to use AIC, BIC or gMDL criteria to select the best-fitting base-learner in each boosting step. In contrast to our discussion above Bühlmann and Yu do not use the degrees of freedom of the single base-learner but use the degrees of freedom of the boosted model. In this case, the degrees of freedom of the base-learners in step m do not govern the selection in step m that strong, as they are only one part of the overall degrees of freedom. In some exemplary simulations we could show that the problem of penalized criteria (e.g., AIC and BIC), that more flexible base-learners are never selected in later iterations, vanishes. Nevertheless, this approach does not solve the problem of biased base-learner selection as in this case the penalty induced by the model degrees of freedom even seems to be too weak.

In conclusion, unbiased model selection seems not possible if different degrees of freedom are specified. As we directly *specify* the degrees of freedom of the base-learners, no high-dimensional optimization via cross-validation is needed to choose appropriate smoothing parameters. Choosing the appropriate complexity of each smooth term is reduced to one-dimensional cross-validation: one only needs to choose an appropriate stopping iteration m_{stop}. Yet, due to the iterative nature of boosting, the final degrees of freedom for one covariate can vary greatly, even if equal degrees of freedom are specified for the base-learners.

4. Constrained Regression in Multivariate STAR Models

> We have *no* idea about the 'real' nature of things . . . . The function of modeling is to arrive at descriptions which are *useful*.
>
> *(Richard Bandler and John Grinder[1])*

In many situations, researchers have prior knowledge of or assumptions on the shape of effects. A very strong assumption is the common linearity assumption. The effect estimate is constrained to follow a straight line. The linearity assumption implicitly imposes a monotonicity assumption. Yet, the direction of monotonicity is not restricted, i.e., monotonically increasing or decreasing functions are possible. In many models it seems more appropriate to relax the linearity assumption but to maintain the monotonicity assumption. The linearity assumption can be relaxed if one switches to (generalized) additive models (GAM) or structured additive regression (STAR) models (see Sec. 1.3).

An example of a structured additive model with monotonicity constraints is given in the first case study of this chapter in the context of modeling the breeding distribution of the Red Kite *(Milvus milvus)* in the German federal state of Bavaria (see Section 4.5). Hothorn, Müller, Schröder, Kneib, and Brandl (2011c) used a very flexible STAR model for describing the breeding distribution. In certain aspects, the resulting model leads to biologically implausible effects, such as non-monotonic estimates for certain environmental variables, which are difficult to justify and might be due to artifacts in the data. Particularly at the extremes of environmental gradients, observational data, e.g., in atlases, often represents low numbers of observations (e.g., very high or very low elevation). This can lead to artificial bumps in the estimated regression curves that lack any biological background. Also a low amount of information in the middle of an environmental

[1] Richard Bandler and John Grinder. (1979) Frogs into Princes: Neuro Linguistic Programming. Real People Press, Utah, USA. p.7.

gradient may induce such effects. Here, we extend the modeling framework suggested in an earlier, related work (Hothorn *et al.* 2011c) and allow for monotonicity constraints to be incorporated on the smooth, additive regression curves in the global model component. In doing so, we add a Bayesian flavor to species distribution models while technically adhering to a frequentist's approach. In this sense, allowing for monotonicity constraints builds a bridge between the two worlds of statistics.

Another case of a priori information resulting in constrained estimation is given if one tries to model periodic, seasonal effects. The effect should be continuous over time. Thus, for example the first and the last day of the year should be continuously joined if a seasonal effect is estimated. Huge jumps for effects only one day apart seem unrealistic. In this case a strong assumption could be a sine (or a similar periodic function) where the frequency is chosen such that the effects match up for the first and last day and only the amplitude is estimated. This functional assumption can be relaxed by using smooth functions that match up, i.e. smooth functions with a cyclic constraint. The second case study considered in this chapter investigates the activity of European Roe Deer *(Capreolus capreolus)* over the day and over the seasons while controlling for further covariates such as the age or the sex of the animal and climatic data (see Section 4.6). For the time of day, as well as for the day of the year, cyclic constraints are well advised. This has two effects: First of all it allows to fit a plausible model as we avoid jumps at the boundaries. Second, we stabilize the estimation at the boundaries as we "borrow information from the other side", i.e. we exploit the cyclic nature of the data.

A third case study, which is on mortality due to air pollution in São Paulo, is presented in Section 4.7. Saldiva, Pope, Schwartz, Dockery, Lichtenfels, Salge, Barone, and Bohm (1995) investigate the impact of SO_2 on the mortality with respiratory causes in elderly people (over 65 years of age). Smooth estimates of the effect of SO_2 on respiratory deaths showed a very erratic behavior. This seems unreasonable as an increase in the air pollutant should not result in a decreased risk of death. Hence, a monotonic increasing effect should lead to a more stable and interpretable model. Leitenstorfer and Tutz (2007) used an approach based on likelihood-based boosting (Tutz and Binder 2006) to estimate a model with monotonic effect. They also included time as a covariate to control for seasonality. The study ranged over four successive years from January 1994 to December 1997. This allows to estimate cyclicity constrained effects for the seasonal component. Hence, we combine both, monotonicity constraints and cyclic effects in one model. We will use the air pol-

lution study to compare our approach with the approach of Leitenstorfer and Tutz (2007). Differences of the approaches will be discussed in depth.

In the following section, a short overview on possible approaches to fit monotonicity constrained effects, and periodic effects is given. In Section 4.2, a novel method to estimate monotonicity-constrained, smooth effects in the boosting framework is introduced. Subsequently, we demonstrate how this idea can also be applied to estimate monotonic effects for ordinal variables. Finally, cyclic P-splines are introduced, which can be used to model periodic effects. The constrained estimation approaches are empirically evaluated in Section 4.3. An extension of monotonicity and cyclicity constraints to bivariate P-splines is given in Section 4.4. We end the chapter with the three case studies that were already introduced in the preceding paragraphs.

4.1. Constrained Regression — History and Present

4.1.1. Monotonic Effects

There are many recent approaches and lots of work has been done in the field of monotonic regression in the last years with the advent of additive models. However, the history of monotonic regression approaches dates back until the 1950-ies when Brunk (1955), Ayer, Brunk, Ewing, Reid, and Silverman (1955), and van Eeden (1956) — amongst others — independently developed approaches to isotonic regression. A more extensive treatment of the early history of monotonic regression is given in Barlow, Bartholomew, Bremner, and Brunk (1972) and de Leeuw, Hornik, and Mair (2009b). The works of Ayer *et al.* (1955) and van Eeden (1956) established the bases for the pool-adjacent-violators algorithm (PAVA). Despite its age, the PAV algorithm is still subject to active research. De Leeuw *et al.* (2009b) give an extensive treatment of the PAV algorithm and derive an active set method to solve the associated convex programming problem with linear constraints. Their algorithm is implemented in the R package **isotone** (de Leeuw, Hornik, and Mair 2009a). Many other packages that implement PAVA exist (for a comprehensive list see de Leeuw *et al.* 2009b). PAVA aims at finding a step function that optimizes a convex loss function under the constraint that the function is monotonic. With n (unique) observations, in practice this results in a function with at maximum n steps. Bacchetti (1989) generalized the idea to additive models, allowing multiple monotonic predictors. Fitting is then achieved by a backfitting-like algorithm built

around PAVA: the cyclic pool adjacent violators (CPAV) algorithm. This idea is further extended to high dimensional models where variable selection in a lasso-like style is used (Fang and Meinshausen 2011). The Lasso isotonic regression approach provides the possibility of specifying models with monotonic effects where the direction of monotonicity (increasing or decreasing) is unknown. This is especially interesting in cases with lots of predictors. However, as for PAVA, this algorithm leads to monotonic step functions. While this is desirable in some cases, discrete step functions are not in line with linear or smooth models, which are considered in many situations. Furthermore, for continuous predictors it may be often be inappropriate to assume discontinuous effects. Many processes in nature have a smooth transition, i.e. small changes in the predictor variable result in small changes in the response.

Approaches resulting in continuous function estimates include a recent approach of Dette, Neumeyer, and Pilz (2006) and Dette and Scheder (2006) implemented in the R package **monoProc** (Scheder 2009). The idea is to estimate, in a first step, an unconstrained, smooth function $\hat{f}$ based, for example, on local polynomials or kernel smoothers. In a second step a "monotonized" version $\hat{f}_{\text{mono}}$ of the initial smoother is computed. This approach leads to strictly monotonic functions but it may result in functions that are not necessarily differentiable. Furthermore, application of the approach in multivariate settings would require an iterative procedure such as (an adapted version of) backfitting: in each backfitting step an unconstrained estimate needs to be computed, which is monotonized successively in each step. Another recent approach, called COBS (constrained B-spline smoothing; He and Ng 1999), aims at fitting smooth quantile regression models with constraints. Models are estimated by minimizing the L_1 loss with an additional L_1 or L_∞ roughness penalty to ensure smoothness. Monotonicity is incorporated via linear programming algorithms such as the Frisch-Newton algorithm (Koenker and Ng 2005a). The resulting estimates are smooth and monotonic. For further details on this approach we refer to He and Shi (1998), Koenker and Ng (2005b), and Ng and Maechler (2007). Ramsay (1988) uses *integrated* M-splines (i.e., I-splines) to achieve monotonic splines. As the I-spline basis is a monotonic basis, restricting the coefficients to be non-negative is sufficient to achieve a monotonic increasing function estimate. Tutz and Leitenstorfer (2007) use this approach in a likelihood-based boosting algorithm. A boosting algorithm for monotonic models based on B-splines is considered in Leitenstorfer and Tutz (2007). This approach is similar to the approach considered here and will be discussed in more detail in Section 4.7. A

general approach to monotonic regression based on P-splines was introduced earlier by Eilers (2005) and Bollaerts, Eilers, and van Mechelen (2006). The authors use P-splines with an additional asymmetric penalty to enforce monotonicity. In this work, we will focus on this approach as it perfectly fits in our boosting framework: All other base-learners are either unpenalized or apply L_2 penalties, which can be estimated using a penalized least squares criterion. Monotonic P-splines can be formulated and estimated alike in a penalized least squares framework. Details of the approach and its integration in model-based boosting are given in Section 4.2.1.

4.1.2. Periodic Effects

The work on periodic function estimation is largely driven by longitudinal studies. Jones and Brelsford (1967), for example, consider auto-regressive processes and use a Fourier series expansion for the regression coefficients. Lewis and Stevens (1991) propose a method to determine cyclic behavior of the underlying data using linear, truncated regression splines on lagged time-series values in the MARS framework (Friedman 1991). This results in piecewise linear but flexible functions with the ability to detect interactions within the auto-regressive process. Fok and Ramsay (2006) propose an approach to model both, a cyclic trend and a non-periodic long-term trend. They allow that the cyclic trend may evolve over time, i.e., a varying-coefficient model is used. Both, the varying coefficients and the long-term trend are estimated using B-splines (without penalty), while the cyclic trend makes use of Fourier basis functions. A problem of this approach is that the number of basis functions, and the placement of knots needs to be very carefully chosen as no penalty is used to control the smoothness of the fitted model.

Quite recently, periodic function estimation was tackled by using general spline methodology with additional constraints. Examples to incorporate cyclic constraints into spline smoothing include the works of Harvey and Koopman (1993) and Harvey, Koopman, and Riani (1997). To model periodic effects, they use cubic regression splines where the estimated effects (as well as the first and second derivatives) at the boundaries are required to be equal. Zhang, Lin, and Sowers (2000) use cubic smoothing splines with a periodic constraint to model longitudinal hormone data. Formulation and estimation of periodic smoothing splines are handled in a mixed model framework. Zhang, Lin, and Sowers (2007) expand this approach in order to model population based periodic functions, subject specific periodic functions, and take to the measurement error of the data into account.

This is done using a two-stage regression calibration method. Another approach extends smoothing splines by altering the penalty to a more general form, which is based on linear differential operators. The resulting splines are termed L-splines (see e.g., Gu 2002, Sec. 4.3). Smoothing splines build a special case of L-splines with the linear differential operator being the second derivative operator. In this case, a linear function remains unpenalized. In the context of periodic splines, L-splines can be constructed such that a cyclic function remains unpenalized (see e.g., Ramsay and Dalzell 1991, Heckman and Ramsay 2000). More recently, Welham, Cullis, Kenward, and Thompson (2006) used L-splines in a mixed model framework to model periodic functions.

For P-splines, it is relatively easy to include cyclic penalties. It only requires to slightly alter the difference penalty. Early versions of periodic function estimation using P-splines were even considered in the rejoinder to the original P-spline paper (Eilers and Marx 1996). The usual difference penalty is altered by inclusion of a sine term. This forces the estimated function (for large values of the smoothing parameter λ) to follow a sine function. Eilers, Gampe, Marx, and Rau (2008) proposed another approach to model seasonal data. They model a smooth trend based on P-splines and add a seasonal component, which is modeled as a sum of two varying coefficient terms. The two varying coefficient terms are modeled as the interaction of a P-spline and a sine function, and a cosine function, respectively. Eilers and Marx (2010) proposed to alter both the design matrix and the penalty matrix of P-splines by appropriately 'wrapping' the ends of both matrices. This approach will be further considered here. An in-depth derivation and discussion of the approach is given in Section 4.2.3.

4.2. Constrained Univariate Regression

4.2.1. Estimating Monotonic, Smooth Effects

The approaches to flexible, yet smooth modeling considered in this chapter are based on B-splines with difference penalties on adjacent knots (i.e., P-splines, Eilers and Marx 1996). An introduction to P-spline base-learners is found in Section 2.3. Consider observations $(y_i, x_i)_{i=1}^n$ and define the negative gradient vector $\mathbf{u} = (u_1, \ldots, u_n)^\top$ as in Equation (2.4) (for simplicity we dropped the iteration index $[m]$). Let $\mathbf{B} = (\mathbf{B}(x_1), \ldots, \mathbf{B}(x_n))^\top$ be the $(n \times J)$ matrix of B-spline basis functions. To achieve a smooth, yet monotonic function estimate, Eilers (2005) introduced P-

splines with an *additional asymmetric difference penalty*. The penalized least squares criterion (2.13) becomes

$$\mathcal{Q}(\boldsymbol{\beta}) = (\mathbf{u} - \mathbf{B}\boldsymbol{\beta})^\top (\mathbf{u} - \mathbf{B}\boldsymbol{\beta}) + \lambda_1 \mathcal{J}(\boldsymbol{\beta}; d) + \lambda_2 \mathcal{J}_{\text{asym}}(\boldsymbol{\beta}; c), \tag{4.1}$$

with the quadratic difference penalty of order d as in usual P-splines,

$$\mathcal{J}(\boldsymbol{\beta}; d) = \sum_{j=d+1}^{J} (\Delta^d \beta_j)^2 = \boldsymbol{\beta}^\top \mathbf{D}_{(d)}^\top \mathbf{D}_{(d)} \boldsymbol{\beta},$$

and the additional asymmetric penalty

$$\mathcal{J}_{\text{asym}}(\boldsymbol{\beta}; c) = \sum_{j=c+1}^{J} v_j (\Delta^c \beta_j)^2 = \boldsymbol{\beta}^\top \mathbf{D}_{(c)}^\top \mathbf{V} \mathbf{D}_{(c)} \boldsymbol{\beta}, \tag{4.2}$$

where c is the order of the differences. The difference matrices $\mathbf{D}_{(c)}$ and $\mathbf{D}_{(d)}$ are constructed as in Equations (2.15) and (2.16). The choice of c implies the type of additional constraints: monotonicity for $c = 1$ or convexity / concavity for $c = 2$. In the remainder of the chapter, we restrict our attention to monotonicity constraints; however, one can always think of convex or concave constraints as well. The asymmetric penalty looks very much like the P-spline penalty (2.14) with the important distinction of weights v_j, which are specified as

$$v_j = \begin{cases} 0 & \text{if } \Delta^c \beta_j > 0 \\ 1 & \text{if } \Delta^c \beta_j \leq 0. \end{cases} \tag{4.3}$$

The weights are collected in the diagonal matrix $\mathbf{V} = \text{diag}(\mathbf{v})$. With $c = 1$, this enforces monotonically *increasing* functions. Changing the direction of the inequalities in the distinction of cases leads to monotonically *decreasing* functions. As the weights (4.3) depend on the coefficients $\boldsymbol{\beta}$, a solution to (4.1) can only be found by iteratively minimizing $\mathcal{Q}(\boldsymbol{\beta})$ with respect to $\boldsymbol{\beta}$, where the weights $\mathbf{v}$ are updated in each iteration. This is equivalent to repeatedly solving

$$\hat{\boldsymbol{\beta}} = (\mathbf{B}^\top \mathbf{B} + \lambda_1 \mathbf{D}_{(d)}^\top \mathbf{D}_{(d)} + \lambda_2 \mathbf{D}_{(c)}^\top \mathbf{V} \mathbf{D}_{(c)})^{-1} \mathbf{B}^\top \mathbf{u} \tag{4.4}$$

with updated weights. The estimation converges if no further changes in the weight matrix $\mathbf{V}$ occur. In our experience, the algorithm converges very quickly within a few steps. The penalty parameter λ_2, which is associated with the additional con-

straint (4.2), could be chosen, for example, by cross-validation techniques. However, this would increase the computational costs dramatically, and, more importantly, λ_2 resembles the researcher's *a priori* belief in monotonicity. Hence, the proposed procedure is located in between pure frequentist and Bayesian approaches. Eilers (2005) proposes that the penalty parameter chosen should be quite large (e.g., 10^6), where larger values are associated with a stronger impact of the monotonic constraint on the estimation. In the limit of $\lambda_2 = 0$, no monotonicity constraint is applied to the estimation procedure. For small values of λ_2, the monotonicity constraint is a suggestion rather than a constraint. However, if the penalty parameter chosen is large enough (as suggested), the actual size is negligible. The effect of the monotonicity constraint can be observed in Figure 4.1, where the blue curve is fitted via P-splines (corresponding to $\lambda_2 = 0$) and the red curve with monotonic P-splines ($\lambda_2 = 10^6$). As the corresponding derivatives (dashed lines) show, the constrained fit is monotonically increasing (derivative always greater zero), while the unconstrained fit is not monotonic.

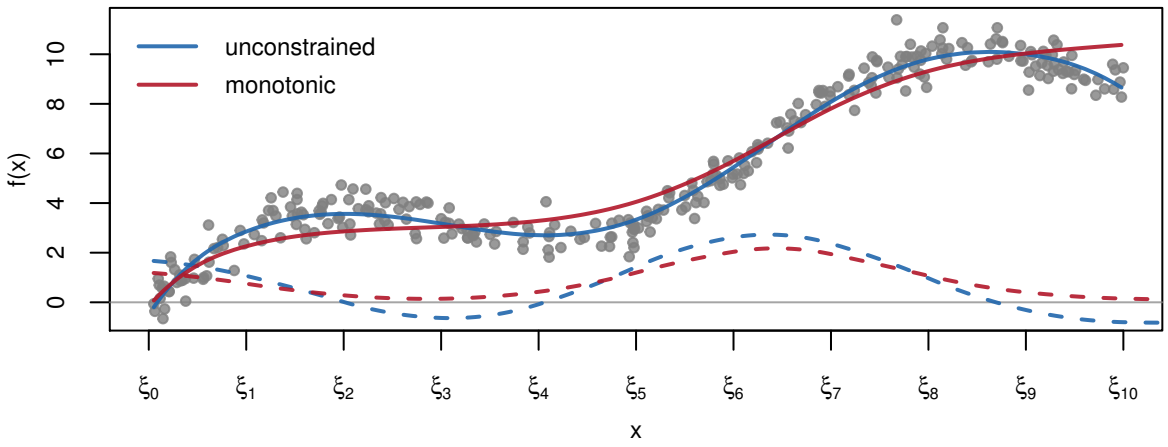


Figure 4.1.: Unconstrained P-splines (blue) and monotonic P-splines (red) fitted to simulated data (gray, filled circles) together with corresponding derivatives (dashed lines), where $\xi_0, \ldots, \xi_{10}$ denote the knots of the P-splines

A simulation study evaluating the properties of monotonic, smooth effects can be found in Section 4.3. In short, we simulate artificial data where the true effect of the covariates on the response is monotonic. We show that effect estimates of unconstrained models regularly violate monotonicity and often result in models with a higher mean squared error compared to monotonicity-constrained models. Hence, in this setting, monotonicity-constrained models are clearly superior.

4.2.2. Estimating Monotonic Categorical Effects

As Eilers' (2005) asymmetric difference penalty relies only on the neighborhood information, i.e., implies an ordering of β_j (for P-splines induced by the knots), we can extend the estimation scheme to *ordered factors* x. The ordering of categories is reflected in an ordering of the effects for the respective categories. Let x be an ordered, categorical variable with categories $1, \ldots, n_{\text{cat}}$. We use dummy-coded variables $x^{(2)}, \ldots, x^{(n_{\text{cat}})}$ (the first category is the reference category), where $x^{(k)} = 1$ if $x = k$ and $x^{(k)} = 0$ otherwise. As for the penalized ordinal base-learner (see Section 2.3.1) we consider, without loss of generality, a base-learner without intercept. With $\mathbf{x}^{(k)} = (x_1^{(k)}, \ldots, x_n^{(k)})^\top$, $k = 2, \ldots, n_{\text{cat}}$, we obtain the dummy-coded $(n \times n_{\text{cat}} - 1)$ design matrix $\mathbf{X} = (\mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n_{\text{cat}})})$ of the ordinal variable. The coefficient vector is $\boldsymbol{\beta} = (\beta_2, \ldots, \beta_{n_{\text{cat}}})^\top$. The estimation problem can then be written as a penalized least squares problem

$$\mathcal{Q}(\boldsymbol{\beta}) = (\mathbf{u} - \mathbf{X}\boldsymbol{\beta})^\top (\mathbf{u} - \mathbf{X}\boldsymbol{\beta}) + \lambda_2 \mathcal{J}_{\text{asym}}(\boldsymbol{\beta}; c), \tag{4.5}$$

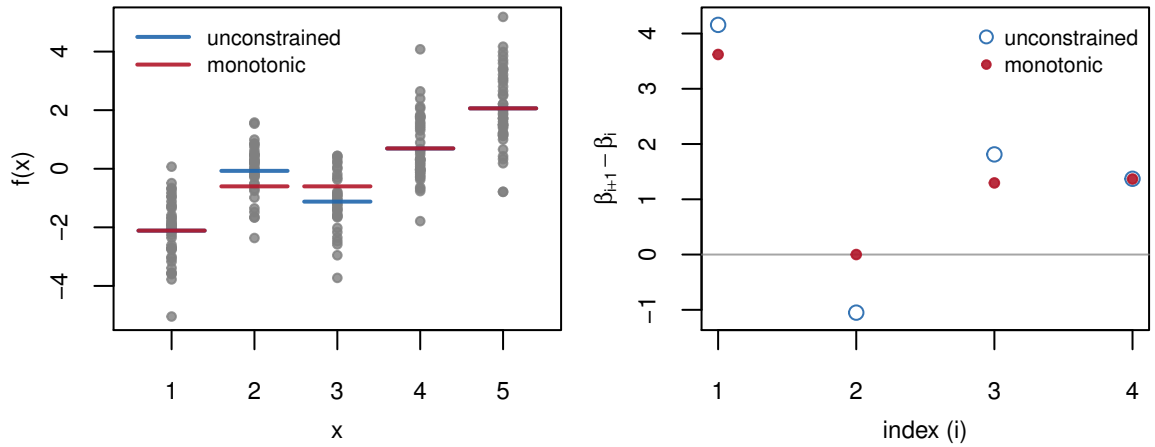
with an asymmetric penalty defined similar to Equation (4.2). The only difference is that the difference matrix $\mathbf{D}_{(c)}$ is replaced by a difference matrix where the first column is removed as $\beta_2 - 0 = \beta_2$ must fulfill the constraint (as the effect of the reference category is set to 0). Hence, for $c = 1$ we get the difference matrix (2.11). This enforces monotonically increasing or decreasing effect estimates — depending on the inequality sign that is used in the definition of the weights $\mathbf{V}$ (Eq. 4.3).

Figure 4.2(a) shows the effect of the monotonicity constraint. The effects of the second and third category clearly violate monotonicity, which is eliminated with the monotonicity constraint. For monotonically increasing effects, the differences of adjacent effect estimates should always be positive (cf. derivative for smooth functions). This is enforced by the penalty as Figure 4.2(b) shows.

As for monotonic P-splines, the asymmetric monotonicity penalty can be combined with a smoothness penalty. Here we use the penalty from penalized ordinal base-learners (see Equation (2.9), Section 2.3.1) This further penalty ensures that the jumps between adjacent categories are small, i.e., neighboring categories are assumed to be similar and thus estimates should not be too erratic. In the combination

$$\mathcal{Q}(\boldsymbol{\beta}) = (\mathbf{u} - \mathbf{X}\boldsymbol{\beta})^\top (\mathbf{u} - \mathbf{X}\boldsymbol{\beta}) + \lambda_1 \mathcal{J}(\boldsymbol{\beta}; d) + \lambda_2 \mathcal{J}_{\text{asym}}(\boldsymbol{\beta}; c), \tag{4.6}$$

we get a monotonicity-constrained, "smooth" estimate for the categorical variable.



(a) Categorical Effects (b) Differences of Effects of Adjacent Categories

Figure 4.2.: Unconstrained (blue) and constrained (red) estimates for categorical effects (a) fitted to simulated data (gray, filled circles), and differences of effects of adjacent categories, where blue circles represent unconstrained and filled, red dots represent constrained estimates (b).

Both estimation criteria (4.5) and (4.6) can be minimized by repeatedly solving Equation (4.4), where the B-spline design matrix $\mathbf{B}$ is replaced by the dummy-coded design matrix $\mathbf{X}$.

Again, we set up a simulation study to evaluate the properties of ordinal, monotonic effects (Section 4.3). As in the previous case, we simulate artificial data where the true effect of the ordinal covariates on the response is monotonic. We show that effect estimates of unconstrained models regularly violate monotonicity and in the mean result in models with a higher mean squared error compared to monotonicity-constrained models. Hence, in this setting, monotonicity-constrained models for ordinal variables are clearly superior to unconstrained models for ordinal factors.

4.2.3. Estimating Cyclic, Smooth Effects

P-splines with a cyclic constraint (Eilers and Marx 2010) can be used to model periodic, seasonal data. The cyclic B-spline basis functions are constructed without knot expansion. The B-splines are "wrapped" at the boundary knots (see Fig. 4.3). The boundary knots ξ_0 and ξ_J (equal to ξ_{12} in Fig. 4.3) play a vital role in this setting, as they specify the points were the function estimate should be smoothly joined. If x is, for example, the time during the day ξ_0 is 0:00, whereas ξ_J is 24:00.

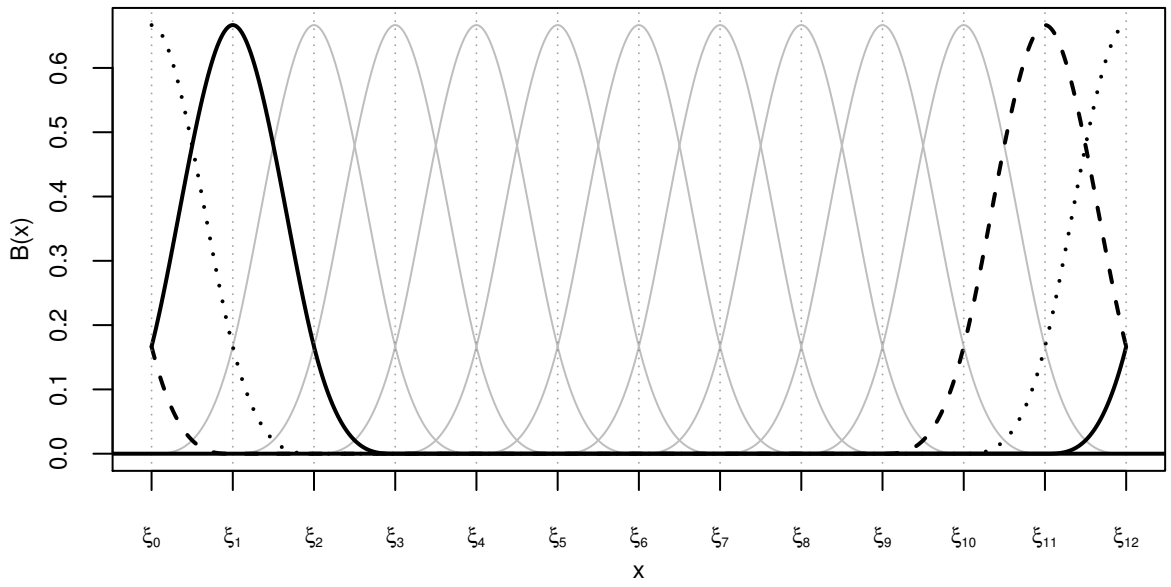


Figure 4.3.: Illustration of cyclic P-splines of degree three, with 11 inner knots and boundary knots ξ_0 and ξ_{12}; The black curves correspond to B-splines that are "wrapped" at the boundary knots leading to a cyclic representation of the function.

As one can see in Figure 4.3, the dashed B-spline basis depends on observations in $[\xi_9, \xi_{12}] \cap [\xi_0, \xi_1]$. Thus, it depends on observations from both ends of the range of the covariate. The same holds for the other bold, black B-splines in Figure 4.3. Defining the B-spline basis in this fashion leads to a cyclic B-spline basis with the $(n \times (J+1))$ design matrix $\mathbf{B}_{\text{cyclic}}$. The corresponding coefficients are collected in the $((J+1) \times 1)$ vector $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_J)$.

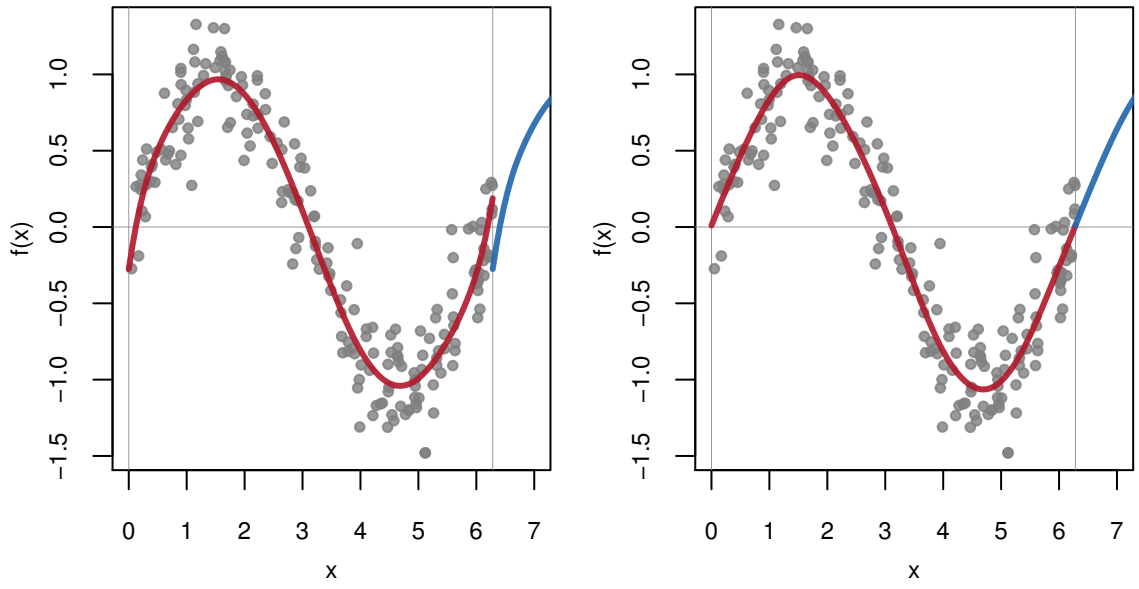
Specifying a cyclic basis guarantees that the resulting function estimate is continuous in the boundary knots. However, no smoothness constraint is imposed so far. This can be achieved by a cyclic difference penalty, for example,

$$\mathcal{J}_{\text{cyclic}}(\boldsymbol{\beta}) = \sum_{j=0}^{J} (\beta_j - \beta_{j-1})^2 \quad (\text{with } d = 1)$$

or

$$\mathcal{J}_{\text{cyclic}}(\boldsymbol{\beta}) = \sum_{j=0}^{J} (\beta_j - 2\beta_{j-1} + \beta_{j-2})^2 \quad (\text{with } d = 2),$$

where the index j is "wrapped", i.e., $j := p + j$ if $j < 0$. Thus, the differences between β_0 and β_J or even β_{J-1} are taken into account for the penalty. Hence, the boundaries of the support are stabilized and smoothness in and around the boundary knots is enforced. This can also be seen in Figure 4.4. The non-cyclic estimate (Fig. 4.4(a)) is strongly influenced by the 'extreme' values on the left and



(a) Non-cyclic Effect (b) Cyclic Effect

Figure 4.4.: The red curve is the function estimated with non-cyclic (a) and cyclic (b) P-splines. The blue curve is the same function shifted one period to the right. The data was simulated from a cyclic sine function with period 2π (gray dots). The cyclic estimate is more stable in the boundary regions and the ends are meeting (see right plot at $x = 2\pi$).

on the right side. As a consequence, the function 'overshoots' at the boundaries, the ends are not meeting. The cyclic estimate (Fig. 4.4(b)), in contrast, is stabilized at the boundaries and the ends are smoothly joined.

Coefficients can then be estimated using the penalized least squares criterion (2.13) where the design matrix and the penalty matrix are replaced with the corresponding cyclic counterparts. In matrix notation the penalty can be written as

$$\mathcal{J}_{\text{cyclic}}(\boldsymbol{\beta}; d) = \boldsymbol{\beta}^\top \widetilde{\mathbf{D}}_{(d)}^\top \widetilde{\mathbf{D}}_{(d)} \boldsymbol{\beta}, \tag{4.7}$$

with difference matrices

$$\widetilde{\mathbf{D}}_{(1)} = \begin{pmatrix} 1 & & & & -1 \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{pmatrix} \tag{4.8}$$

and

$$\widetilde{\mathbf{D}}_{(2)} = \begin{pmatrix} 1 & & & & 1 & -2 \\ -2 & 1 & & & & 1 \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \end{pmatrix}, \tag{4.9}$$

where empty cells are equal to zero. All together, this leads to the penalized least squares problem

$$\mathcal{Q}(\boldsymbol{\beta}) = (\mathbf{u} - \mathbf{B}_{\text{cyclic}}\,\boldsymbol{\beta})^\top (\mathbf{u} - \mathbf{B}_{\text{cyclic}}\,\boldsymbol{\beta}) + \lambda \mathcal{J}_{\text{cyclic}}(\boldsymbol{\beta}; d), \tag{4.10}$$

which is essentially identical to the usual P-spline criterion. Only the basis and the penalty matrices are altered as discussed above.

As discussed in Section 2.3.2, P-splines have a null space, i.e., an unpenalized effect, which depends on the order of the differences in the penalty. Differences of order two lead to a linear effect that remains unpenalized, even in the limit of $\lambda \to \infty$. However, cyclic P-splines have a null space that only includes a constant, irrespective of the order of the difference penalty. Globally seen, i.e., for the complete function estimate, the order of the penalty plays no role (even in the limit $\lambda \to \infty$). Locally, however, the order of the difference penalty has an effect. For example, with $d = 2$ the estimated function is penalized for deviations from linearity and hence, locally approaches a straight line (with increasing λ).

As for monotonic effects, we empirically study the properties of cyclic effects in the following section. We use truly periodic effects and evaluate the estimates of models with cyclic constraints and unconstrained models using the MSE, as before. We show that cyclic effects are superior, both with respect to the MSE and regarding the number of violations of cyclicity.

4.3. Empirical Evaluation of Constrained Effect Estimates

We evaluated the estimation procedure for monotonic smooth and monotonic ordinal effects, and cyclic effects and compared the resulting effects to unconstrained estimates. In all settings, we used observations $\mathbf{x}_i = (x_{1i}, x_{2i})^\top$, $i = 1, \ldots, n$, and

true effects $f_1(x_i)$ and $f_2(x_2)$ (see below for the specification of the predictor $\mathbf{x}_i$ and the effects). The response $\mathbf{y}$ was simulated according to

$$y_i = \alpha + f_1(x_{1i}) + f_2(x_{2i}) + \varepsilon_i, \quad i = 1, \ldots, n,$$

with intercept α, and random error $\varepsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$ with σ^2 such that fractions of the explained variance (R^2) of approximately 0.3, 0.5, and 0.8 were realized. The number of observations n was chosen to be 100, 200, and 500, respectively. In combination with the different variances, this results in nine scenarios altogether. For each scenario $B = 100$ data sets were generated. The model fit was evaluated by the mean squared error for $N = 10,000$ new observations:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} \Big(\big[\hat{\alpha} + \hat{f}_1(x_{1i}) + \hat{f}_2(x_{2i})\big] - \big[\alpha + f_1(x_{1i}) + f_2(x_{2i})\big] \Big)^2.$$

4.3.1. Smooth, Monotonic Effects

The observations $\mathbf{x}_i = (x_{1i}, x_{2i})^\top$, $i = 1, \ldots, n$, were drawn independently from a uniform distribution on [0, 1]. The effect of x_1 follows a polynomial of degree three

$$f_1(x_1) = 10(x_1 - 0.5)^3,$$

and the effect of x_2 follows a logistic function,

$$f_2(x_2) = \frac{1}{1 + \exp\{-20(x_2 - 0.5)\}}.$$

The intercept was set to $\alpha = 2$. The functions are depicted in Fig. 4.5.

For each of the data sets two models were estimated: one model with monotonic P-splines and one with unconstrained P-splines. The results are given in Table 4.1. One can conclude that the monotonic model outperforms the unconstrained model. Only in cases with many observations ($n = 200$ and $n = 500$) combined with very little noise in the data ($\sigma^2 = 0.1$), the unconstrained model is marginally better than the constrained model.

The accuracy of the model fit as measured by the MSE captures only one part of constrained modeling: the overall goodness of fit. However, the task is to estimate *monotonic* effects. This affects the quality of the model and, in situations with real data, the interpretability of the model. Thus, the resulting effect estimate should be monotonic (or show only minor deviations from monotonicity). Here we

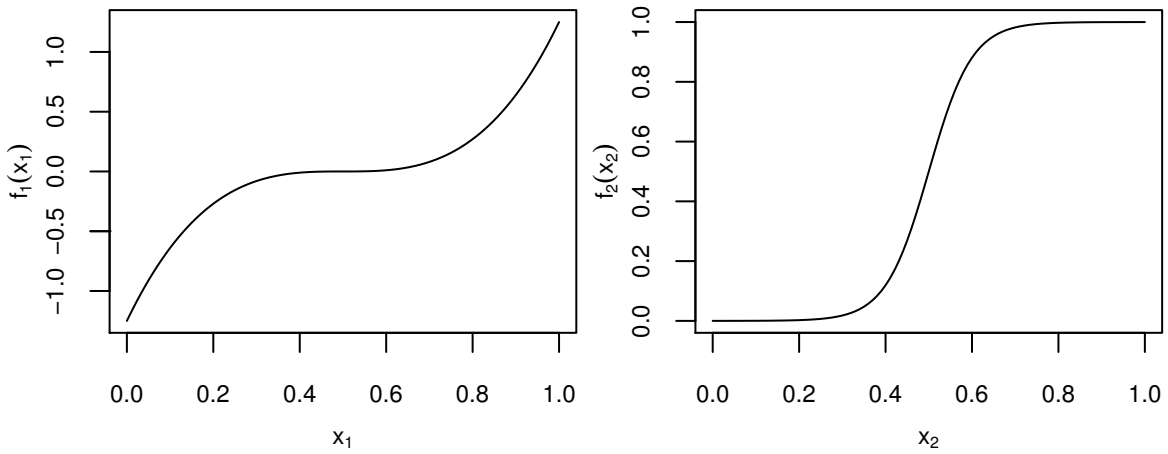


Figure 4.5.: Empirical evaluation of smooth, monotonic effects: True functions $f_1(x_1)$ and $f_2(x_2)$. Note that both functions are monotonically increasing.

consider models to be monotonic if the differences of coefficients of adjacent knots are positive or only marginally negative. Here, an effect is said to be monotonic if

$$\Delta\beta_j > -10^{-4}, \quad \text{for all } j. \tag{4.11}$$

Using this boundary to define monotonicity we find that for the monotonically constrained effects, monotonicity is never violated (see Table C.1 in the appendix). Function estimates from models fitted using unconstrained P-splines violate monotonicity in approximately 75% of the cases. Violations of monotonicity even increase with decreasing noise σ^2. Hence, the situations where the unconstrained

Table 4.1.: *Smooth, monotonic effects:* MSE of monotonicity-constrained and unconstrained models; Mean values and corresponding standard errors (se) estimated from 100 simulation runs.

n	σ^2	monotonic	(se)	unconstrained	(se)
100	1.0	0.0759	(0.0035)	0.0981	(0.0046)
	0.4	0.0349	(0.0015)	0.0431	(0.0019)
	0.1	0.0122	(0.0004)	0.0129	(0.0005)
200	1.0	0.0405	(0.0020)	0.0449	(0.0022)
	0.4	0.0189	(0.0008)	0.0198	(0.0008)
	0.1	0.0076	(0.0002)	0.0063	(0.0002)
500	1.0	0.0155	(0.0008)	0.0171	(0.0007)
	0.4	0.0083	(0.0003)	0.0081	(0.0003)
	0.1	0.0046	(0.0001)	0.0032	(0.0001)

model outperforms the monotonic model are the worst situations with respect to violations of monotonicity. It seems that the reduced MSE comes at the price of non-monotonicity.

4.3.2. Ordinal, Monotonic Effects

In the second scenario, with ordinal, monotonic effects, the observations $\mathbf{x}_i = (x_{1i}, x_{2i})^\top$, $i = 1, \ldots, n$, were drawn independently from a discrete uniform distribution on $\{1, 2, \ldots, 5\}$. With dummy coded design matrices $\mathbf{X}_1$ and $\mathbf{X}_2$ (the first category is used as reference category), the true effect functions $f_1(x_1)$ and $f_2(x_2)$ can be written in matrix notation as

$$f_l(x_l) = \mathbf{X}_l \boldsymbol{\beta}_l, \qquad l \in \{1, 2\},$$

with $\boldsymbol{\beta}_1 = (1, 2, 3, 4)^\top$ and $\boldsymbol{\beta}_2 = \left(\frac{\exp(1)}{40}, \frac{\exp(2)}{40}, \frac{\exp(3)}{40}, \frac{\exp(4)}{40}\right)^\top$. The intercept was set to $\alpha = 4$. Both effects are monotonic increasing in with increasing categories.

For each of the data sets two models were estimated: one model with monotonicity-constraint and one with unconstrained effect estimates. The results are given in Table 4.2 . One can conclude that the monotonic model outperforms the unconstrained model in all given scenarios.

Table 4.2.: *Ordinal, monotonic effects:* MSE of monotonicity-constrained and unconstrained models; Mean values and corresponding standard errors (se) estimated from 100 simulation runs.

n	σ^2	monotonic	(se)	unconstrained	(se)
100	1.0	0.3982	(0.0204)	0.5548	(0.0268)
	0.4	0.1961	(0.0102)	0.2566	(0.0127)
	0.1	0.0533	(0.0028)	0.0657	(0.0033)
200	1.0	0.2098	(0.0104)	0.2569	(0.0115)
	0.4	0.0939	(0.0048)	0.1126	(0.0055)
	0.1	0.0248	(0.0012)	0.0284	(0.0013)
500	1.0	0.0746	(0.0036)	0.0921	(0.0039)
	0.4	0.0326	(0.0016)	0.0388	(0.0017)
	0.1	0.0090	(0.0004)	0.0100	(0.0005)

We also assess the frequency of violations of the monotonicity for all estimated models. As before, we consider models to be monotonic if Equation (4.11) holds. The results are given in the appendix (Tab. C.2). Using constrained estimates,

monotonicity is virtually never violated. In the unconstrained model, much more violations of monotonicity are present. In contrast to the setting with P-splines, the number of violations decreases with increasing sample size and decreasing noise. Furthermore, the overall number of violations is smaller (compare Table C.1 in the appendix). It seems that the signal of categorical covariates is less affected by noise and that the true, monotonic effects are recovered more easily.

4.3.3. Cyclic Effects

For the evaluation of cyclic effects, observations $\mathbf{x}_i = (x_{1i}, x_{2i})^\top, i = 1, \ldots, n$, were drawn independently from a uniform distribution on $[0, 2\pi]$. The true cyclic function $f_1(x_1) = sin(x_1)$; the second cyclic function f_2 is a single B-spline basis of order three, with equidistant inner knots, and boundary knots in 0 and 2π. The intercept $\alpha = 1$ was used. Both functions are depicted in Figure 4.6

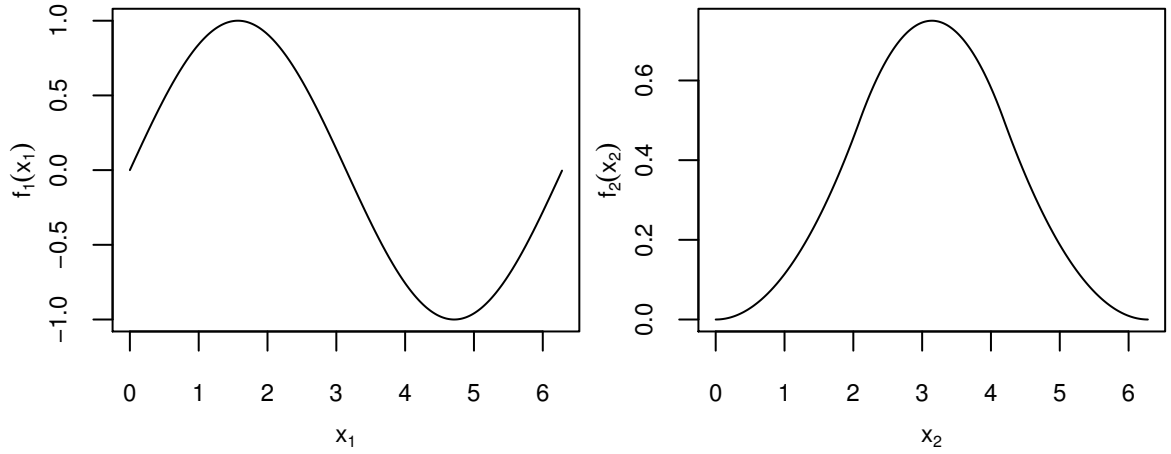


Figure 4.6.: Empirical evaluation of cyclic effects: True functions $f_1(x_1)$ and $f_2(x_2)$. Note that both functions are truly periodic with period 2π.

We fitted the data using a constrained model with cyclic constraints both for x_1 and x_2 and boundary knots in 0 and 2π. As a comparison, an unconstrained model was fitted. The MSE of the cyclic model is smaller throughout all nine scenarios (see Table 4.3).

To evaluate the ability to recover cyclic effects, we use the *absolute difference* of the boundaries of the predicted functions, i.e.,

$$|\Delta \hat{f}_l| := |\hat{f}_l(0) - \hat{f}_l(2\pi)|, \qquad l \in \{1, 2\}.$$

If a function is cyclic, it follows that there is no difference between the function

Table 4.3.: *Cyclic effects:* MSE of models with cyclic constraint and unconstrained models; Mean values and corresponding standard errors (se) estimated from 100 simulation runs.

n	σ^2	cyclic	(se)	unconstrained	(se)
100	1.0	0.1126	(0.0052)	0.1354	(0.0054)
	0.4	0.0539	(0.0025)	0.0679	(0.0027)
	0.1	0.0150	(0.0007)	0.0200	(0.0008)
200	1.0	0.0613	(0.0032)	0.0688	(0.0030)
	0.4	0.0278	(0.0013)	0.0316	(0.0013)
	0.1	0.0074	(0.0003)	0.0090	(0.0003)
500	1.0	0.0232	(0.0013)	0.0263	(0.0013)
	0.4	0.0104	(0.0006)	0.0122	(0.0005)
	0.1	0.0028	(0.0001)	0.0040	(0.0002)

values at the boundaries, i.e., $|\Delta\hat{f}| = 0$. We considered a function estimate $\hat{f}$ to be cyclic even if minor deviations occurred, i.e., $\hat{f}$ is said to be cyclic if $|\Delta\hat{f}| \leq 0.1$. In all scenarios, the effect estimates from the cyclic model are found to be (truly) cyclic. The effect estimates of the unconstrained model are rarely found to be cyclic. Even more, the unconstrained model shows on average an absolute difference $|\Delta\hat{f}|$ of 0.29 (for details see Table C.3 in the appendix). With ranges of $[-1, 1]$ for f_1, and $[0, 0.75]$ for f_2 this absolute difference is remarkable.

4.4. Constrained Effects for Bivariate P-Splines

In Section 2.3.3, bivariate P-splines were introduced. As bivariate P-splines are directly derived from univariate P-splines, they can be extended — basically in the same manner as univariate P-splines — to include monotonicity constraints or cyclicity constraints. In the following sections we will shortly derive both concepts for bivariate P-splines.

4.4.1. Bivariate, Monotonic P-Splines

If it is desired to have a monotonic interaction it is not sufficient to specify an interaction of monotonic effects. Monotonicity of marginal effects does not transfer to monotonicity of an interaction surface: This can be easily verified by considering

the interaction of two increasing, linear functions

$$f(x_1, x_2) = x_1 \cdot x_2, \quad x_1, x_2 \in [-1, 1].$$

For any (fixed) value $x_2 < 0$, the function $f(x_1, x_2)$ is linear decreasing in x_1 and for any $x_2 > 0$, the function $f(x_1, x_2)$ is linear increasing in x_1. The function f is monotonic in both, x_1 and x_2 given the other variable. However, the direction of monotonicity changes depending on the sign of the other variable. The resulting surface is highly non-monotonic (cf. Figure 4.7). Thus, instead of the marginal functions, the complete surface needs to be constrained in order to achieve a monotonic surface.

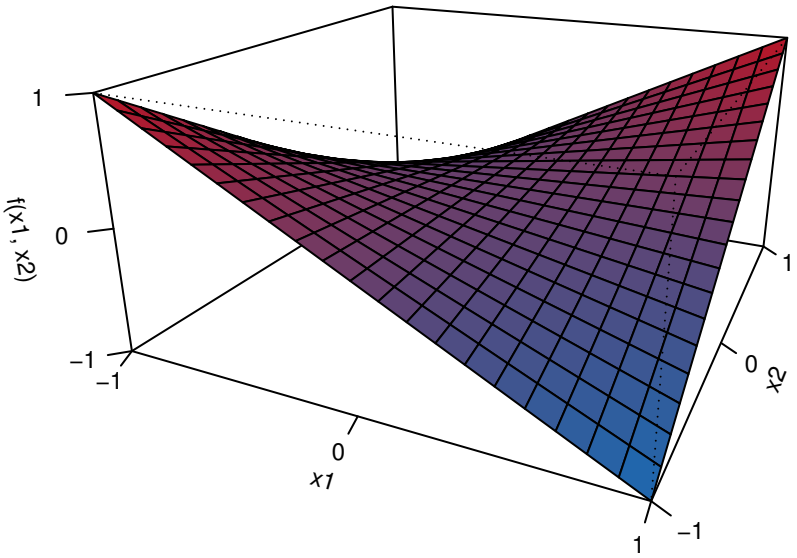


Figure 4.7.: Non-monotonic interaction surface of $f(x_1, x_2) = x_1 \cdot x_2, \quad x_1, x_2 \in [-1, 1]$.

To obtain a monotonic surface estimate, we utilize bivariate P-splines and add monotonic constraints for the row- and column-wise differences of the matrix of coefficients (Eq. 2.21). As proposed by Bollaerts *et al.* (2006), one can use two independent asymmetric penalties to allow for different directions of monotonicity — i.e., increasing in one variable, say x_1, decreasing in the other variable, say x_2 — or different prior beliefs in monotonicity resembled by different penalty parameters λ. Let $\mathbf{u}$ denote the $(n \times 1)$ negative gradient vector (2.4), or an arbitrary continuous response in settings other than boosting. Let $\mathbf{x}_i = (x_{i1}, x_{i2})$, $i = 1, \ldots, n$, denote the observations of variables x_1 and x_2, and let $\mathbf{B} = \Big(\mathbf{B}(\mathbf{x}_1), \ldots, \mathbf{B}(\mathbf{x}_n)\Big)^\top$ denote the $(n \times JK)$ design matrix comprising the bivariate B-spline bases of $\mathbf{x}_i$ (Eq. 2.18) with J knots in the direction of x_1 and K knots in the direction of x_2. The corresponding

coefficient vector (2.17) of dimension $(JK \times 1)$ is denoted by $\boldsymbol{\beta}$. Monotonicity is enforced by the asymmetric difference penalties

$$\mathcal{J}_{\text{asym},1}(\boldsymbol{\beta}; c) = \sum_{j=c+1}^{J} \sum_{k=1}^{K} v_{jk}^{(1)} (\Delta_1^c \beta_{jk})^2 = \boldsymbol{\beta}^\top (\mathbf{D}_1 \otimes \mathbf{I}_K)^\top \mathbf{V}_1 (\mathbf{D}_1 \otimes \mathbf{I}_K) \boldsymbol{\beta} \tag{4.12}$$

$$\mathcal{J}_{\text{asym},2}(\boldsymbol{\beta}; c) = \sum_{j=1}^{J} \sum_{k=c+1}^{K} v_{jk}^{(2)} (\Delta_2^c \beta_{jk})^2 = \boldsymbol{\beta}^\top (\mathbf{I}_J \otimes \mathbf{D}_2)^\top \mathbf{V}_2 (\mathbf{I}_J \otimes \mathbf{D}_2) \boldsymbol{\beta} \tag{4.13}$$

where Δ_1^c are the column-wise differences and Δ_2^c the row-wise differences, i.e. $\Delta_1^1 \beta_{jk} = \beta_{jk} - \beta_{(j-1)k}$ and $\Delta_2^1 \beta_{jk} = \beta_{jk} - \beta_{j(k-1)}$. The corresponding difference matrices are denoted by $\mathbf{D}_1$ and $\mathbf{D}_2$, and $\mathbf{I}_J$ and $\mathbf{I}_K$ denote identity matrices of rank J and K, respectively. The weights $v_{jk}^{(l)}$, $l = 1, 2$ are specified in analogy to (4.3) for monotonic increasing estimates ($c = 1$) as

$$v_{jk}^{(l)} = \begin{cases} 0 & \text{if } \Delta_l^c \beta_{jk} > 0 \\ 1 & \text{if } \Delta_l^c \beta_{jk} \leq 0. \end{cases} \tag{4.14}$$

For the matrix notation, the weights are collected in the diagonal matrices $\mathbf{V}_l = \text{diag}(\mathbf{v}^{(l)})$. Changing the inequality sign in (4.14) leads to monotonic decreasing function estimates, while differences of order $c = 2$ lead to convex or concave constraints.

The constraint estimation problem for monotonic surface estimates in matrix notation becomes

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\beta}) = (\mathbf{u} - \mathbf{B}\boldsymbol{\beta})^\top (\mathbf{u} - \mathbf{B}\boldsymbol{\beta}) &+ \lambda_1 \mathcal{J}_{\text{spatial}}(\boldsymbol{\beta}; d) \\ &+ \lambda_{21} \mathcal{J}_{\text{asym},1}(\boldsymbol{\beta}; c) \\ &+ \lambda_{22} \mathcal{J}_{\text{asym},2}(\boldsymbol{\beta}; c), \end{aligned} \tag{4.15}$$

where $\mathcal{J}_{\text{spatial}}(\boldsymbol{\beta}; d)$ is the standard bivariate P-spline penalty of order d and λ_1 is the corresponding penalty parameter.

For univariate monotonic P-splines, the usual P-spline penalty and the asymmetric penalty have the same form except for the weights $\mathbf{V}$. This can be also shown for *bivariate* P-splines:

Theorem 4.1 *Let* $\mathbf{V}$ *be an identity matrix of appropriate dimension, i.e.,* $\mathbf{V}$ *is dropped from the equation. Then Equation* (4.12) *can be simplified to* $\boldsymbol{\beta}^\top (\mathbf{K}_1 \otimes \mathbf{I}_K) \boldsymbol{\beta}$ *and Equation* (4.13) *can be simplified to* $\boldsymbol{\beta}^\top (\mathbf{I}_J \otimes \mathbf{K}_2) \boldsymbol{\beta}$. *With* $\lambda_{21} = \lambda_{22}$, *the sum of the penalties* (4.12) *and*

(4.13) *is equal to the bivariate P-spline penalty* (2.20).

Proof As the transpose is distributive over the Kronecker product (Steeb 1991, p. 12) and as we assume $\mathbf{V} = \mathbf{I}$

$$(\mathbf{D}_1 \otimes \mathbf{I}_K)^\top \mathbf{V} (\mathbf{D}_1 \otimes \mathbf{I}_K) = (\mathbf{D}_1^\top \otimes \mathbf{I}_K^\top)(\mathbf{D}_1 \otimes \mathbf{I}_K).$$

As the matrices $\mathbf{D}_1^\top \mathbf{D}_1$ and $\mathbf{I}_K^\top \mathbf{I}_K$ exist, it holds (Steeb 1991, p. 16)

$$\begin{aligned}(\mathbf{D}_1^\top \otimes \mathbf{I}_K^\top)(\mathbf{D}_1 \otimes \mathbf{I}_K) &= (\mathbf{D}_1^\top \mathbf{D}_1) \otimes (\mathbf{I}_K^\top \mathbf{I}_K) \\ &= (\mathbf{K}_1 \otimes \mathbf{I}_K).\end{aligned}$$

The proof that Equation (4.13) can be simplified to $\boldsymbol{\beta}^\top (\mathbf{I}_J \otimes \mathbf{K}_2)\boldsymbol{\beta}$ is straightforward along the lines of this proof.

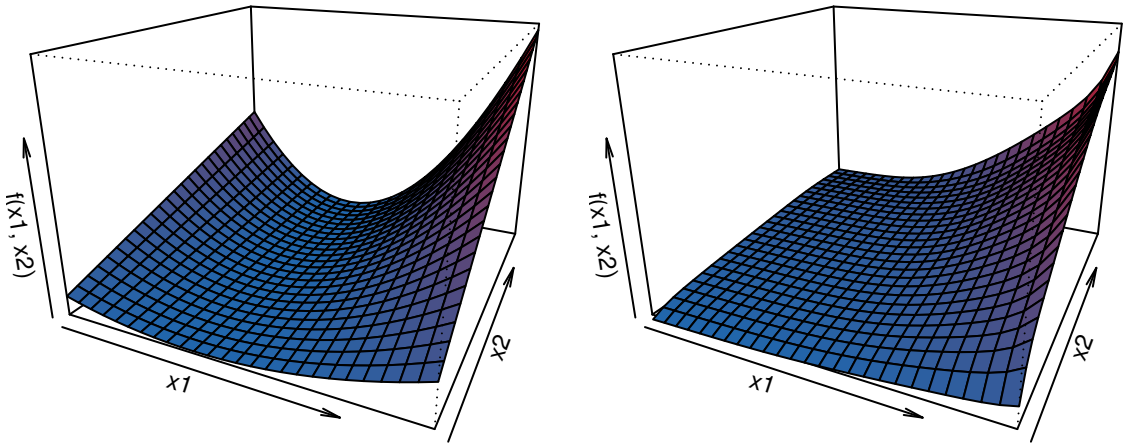
□

From Theorem 4.1, one can conclude that the asymmetric penalties only differ from the usual bivariate P-spline penalty in the weight matrix $\mathbf{V}$.

Example for monotonic surface estimation As an example for bivariate, monotonic smoothing, let us consider uniformly distributed observations $x_{i1}, x_{i2} \overset{i.i.d.}{\sim} U[-2, 3]$, $i = 1, \ldots, n$, and the outcome $y_i = 0.5 + x_{i1}^2 \cdot (x_{i2} + 3) + \varepsilon_i$, where $\varepsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, 0.5^2)$. The function (without the error term ε_i) is depicted in Figure 4.8(a). There is a clear non-monotonic effect, which becomes especially apparent for large values of x_2. Figure 4.8(b) shows the estimated function where the non-monotonic function is estimated by monotonic, bivariate P-splines. The resulting estimated function appears to be monotonic. As a comparison we estimated the same model without monotonicity constraint. To check if the apparent monotonicity truly holds, we plotted the row- and column-wise differences of the coefficients (see Figure 4.9). The differences of the unconstrained model vary from negative to positive values while differences from the constrained model are always greater or equal to zero. Hence, one can conclude that the monotonicity constrained estimate is monotonic while the unconstrained estimate is clearly non-monotonic.

4.4.2. Bivariate, Cyclic P-Splines

Based on bivariate P-splines, cyclic constraints in both directions of x_1 and x_2 are straightforward to implement. With J knots in the direction of x_1 and K knots



(a) True Function (b) Estimated, Monotonic Function

Figure 4.8.: Bivariate, true effect and bivariate, monotonic estimate thereof. Both figures are plotted using the same range for comparability.

in the direction of x_2, one builds the univariate, *cyclic* design matrices $\mathbf{B}^{(1)}_{\text{cyclic}}$ and $\mathbf{B}^{(2)}_{\text{cyclic}}$ for x_1 and x_2, respectively. The bivariate design matrix then is

$$\mathbf{B}_{\text{cyclic}} = (\mathbf{B}^{(1)}_{\text{cyclic}} \otimes \boldsymbol{e}_K^\top) \odot (\boldsymbol{e}_J^\top \otimes \mathbf{B}^{(2)}_{\text{cyclic}}),$$

as in Equation (2.19).

With the univariate, *cyclic* difference matrices $\widetilde{\mathbf{D}}^{(1)}$ and $\widetilde{\mathbf{D}}^{(2)}$ for x_1 and x_2, respectively (see (Eq. 4.8) and (Eq. 4.9) for cyclic first and second order differences), we obtain cyclic penalty matrices $\mathbf{K}_1 = (\widetilde{\mathbf{D}}^{(1)})^\top \widetilde{\mathbf{D}}^{(1)}$ and $\mathbf{K}_2 = (\widetilde{\mathbf{D}}^{(2)})^\top \widetilde{\mathbf{D}}^{(2)}$. Thus, in analogy to the usual bivariate P-spline penalty (Eq. 2.20) the bivariate cyclic penalty can be written as

$$\mathcal{J}_{\text{cyclic, spatial}}(\boldsymbol{\beta}) = \boldsymbol{\beta}^\top (\mathbf{K}_1 \otimes \mathbf{I}_K + \mathbf{I}_J \otimes \mathbf{K}_2)\boldsymbol{\beta},$$

where $\boldsymbol{\beta}$ collects the coefficients of the bivariate P-spline as in Equation (2.17), and $\mathbf{I}_J$ and $\mathbf{I}_K$ denote identity matrices of rank J and K, respectively. Estimation is then a straight forward application of the penalized least squares criterion as in (Eq. 2.22). An example of bivariate, cyclic splines is given in Section 4.6, where a cyclic surface is used to estimate the effect of time (during the day) and calender day on Roe Deer activity.

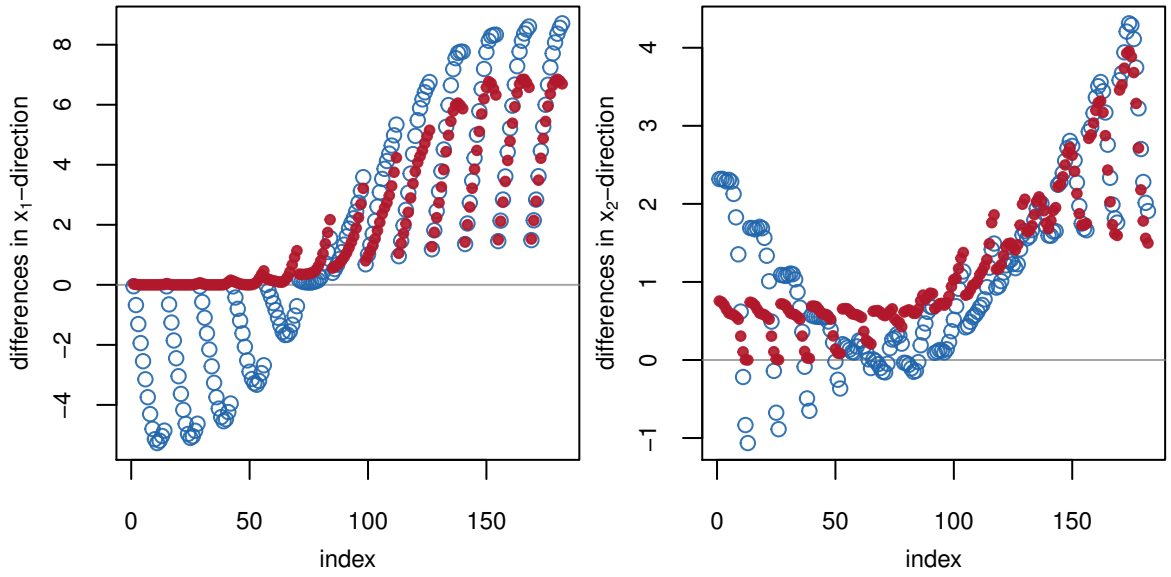


Figure 4.9.: Differences of adjacent coefficients in the direction of x_1 (left) and in the direction if x_2 (right). The blue circles resemble the unconstrained model and the red dots resemble the monotonicity constrained model. Non-negative differences indicate a monotonically increasing effect. A change of signs indicates non-monotonic behavior.

4.5. Monotonicity-Constrained Species Distribution Models for the Red Kite *(Milvus milvus)*

Species distribution models (SDM) estimate the regression relationship between the environment and the distribution of a species. The environment is characterized by a potentially large number of variables that can be used to model occurrence or abundance of species and typically include climatic variables or variables characterizing the habitat. While simple models, e.g., an ordinary linear model, appeal to researchers and practitioners, more complex models, e.g., regression trees, may uncover structures that simpler models will miss (Elith and Leathwick 2009). Recently, Hothorn *et al.* (2011c) developed a framework for species distribution models based on component-wise boosting. The influence of the environment on species distributions can be described in a very flexible manner (i.e., as a combination of linear effects, non-linear effects, spatial effects, etc.) while maintaining interpretability of the regression effects. Within this framework, ecologists can specify local components (such as spatial autocorrelation, spatio-temporal effects, and non-stationary effects) along with global model components. The global model component describes the effects of the environmental variables as a sum of smooth functions.

Here, we focus on SDM for the Red Kite (*Milvus milvus*; see Figure 4.10). We assume that the probability of confirmed Red Kite breeding at some point s (here given as longitude and latitude in Bavaria) and time t under environmental conditions as characterized by environmental variables $\mathbf{x} = (x^{(1)}, \ldots, x^{(p)})$ is given by the inverse logistic transformation of the regression function η to be estimated:

$$\mathbb{P}(\text{ presence of Red Kite breeding } | \, \mathbf{x}, s, t) = \text{logit}^{-1}(\eta(\mathbf{x}, s, t)). \tag{4.16}$$

Hothorn *et al.* (2011c) decompose the contributions of environmental variables, space, and time via a structured additive regression model (see Fahrmeir *et al.* 2004, and (Eq. 1.1)) in an additive fashion:

$$\eta(\mathbf{x}, s, t) = \underbrace{f_{\text{env}}(\mathbf{x})}_{\text{global}} + \underbrace{f_{\text{ns}}(\mathbf{x}, s) + f_{\text{s}}(s) + f_{\text{st}}(s, t)}_{\text{local}}. \tag{4.17}$$

The local model terms f_{ns}, f_{s}, and f_{st}, which capture non-stationary effects, and the spatial- and spatio-temporal autocorrelation are discussed in detail by Hothorn *et al.* (2011c). In essence, these effects can be fitted using bivariate P-splines. Here, we primarily focus on the global model term f_{env}. The global model term describes the influence of environmental variables on Red Kite breeding as the sum of functions in the manner of STAR models, i.e., $f_{\text{env}}(\mathbf{x}) = \sum_{j=1}^{p} f_j(x^{(j)})$. At the same

Figure 4.10.: Red Kite *(Milvus milvus)* observed near Murnau, 05.06.2010. *By courtesy of H.-J. Fünfstück.*

time, however, the local components are taken into account in the model. The term $f_j(x^{(j)})$ represents the contribution of the jth environmental variable to the global model component. The primary aim of this section is to derive species distribution models that allow to incorporate monotonicity constraints on certain global model terms. This can be achieved by applying monotonic P-spline base-learners and monotonic ordinal effects as introduced in Section 4.2.1 and 4.2.2.

4.5.1. Red Kite Breeding Distribution

We extracted the Red Kite breeding data on grid cells of the topographic map (scale 1:25000 for Bavaria) with an average cell area of 33.9 km^2 from the Bavarian breeding atlas in the periods 1979–1983 (Nitsche and Plachter 1987) and 1996–1999 (Bezzel, Geiersberger, Lossow, and Pfeifer 2005). A minimum of one record in a quadrant was sufficient to define the presence of breeding; all other squares were taken as absence of Red Kite breeding. An overview of the environmental variables used for the Red Kite habitat selection model is given in Tables 4.4 and 4.5, together with summary statistics. The environmental variables were extracted from the WorldClim database (`www.worldclim.org`), and land cover was extracted from the CORINE 2000 map (`http://www.corine.dfd.dlr.de`). CORINE data, with their resolution of 100 m $\times$ 100 m, are highly useful in modeling the species distribution of the Red Kite. They provide detailed information on the habitat's patchiness, and structure and on the amount of specific habitats for a prey of bird as the Red Kite with a breeding territory of around 10 km^2. CORINE data have been successfully used for birds and bats in other studies (Pfeifer, Müller, Stadler, and Brandl 2010, Mehr, Brandl, Hothorn, Dziock, Förster, and Müller 2011). The accuracy of the WorldClim data decreases in mountains. However, in our study area, mountains (the Alps) comprise only a small portion of the area. Furthermore, these alpine areas are not inhabited by the Red Kite, and therefore this does not weaken our analysis. For a more detailed description of the data, we refer the reader to Hothorn *et al.* (2011c).

Hothorn *et al.* (2011c) compared species distribution models of different complexities for Red Kite breeding. Their results indicate that a model with smooth, additive effects of the environmental variables along with a non-stationary effect of altitude and spatio-temporal autocorrelation fits the data best (model '`add/vary`' in their notation). We use this model as a starting point, i.e., we specify smooth effects for all continuous variables, linear effects for categorical variables, an addi-

Table 4.4.: Overview and summary statistics of the numeric environmental variables (first, second and third quartile) for Red Kite breeding.

Description	25%	50%	75%
Forest Coverage (in %)	18.00	30.00	47.00
Field (in %)	4.25	26.00	50.00
Complex (in %)	1.00	7.00	17.00
Mixed Forests (in %)	0.00	3.00	11.00
Coniferous Forests (in %)	8.00	19.00	33.00
Meadows (in %)	4.00	11.00	22.00
Total Number of Patches (in %)	19.25	26.00	32.00
Total Patch Density (number per 100 ha)	0.57	0.75	0.95
Total Largest Patch Index (in %)	26.78	40.27	57.72
Length of Total Edge (in m)	85225.00	104000.00	121900.00
Total Edge Density (in m)	24.97	30.57	35.74
Total Landscape Shape Index	3.91	4.69	5.38
Area Forest (in ha)	614.25	1018.50	1568.75
Percentage of Landscape Forest (in %)	18.00	29.89	46.27
Number of Forest Patches (number)	4.00	6.00	9.00
Forest Patch Density (number per 100 ha)	0.12	0.18	0.26
Forest Largest Patch Index (in %)	7.87	16.66	34.62
Forest Edge Density (in m)	11.02	15.37	19.70
Forest Landscape Shape Index	3.23	4.18	5.15
Annual Mean Temp. (in °C)	7.53	7.98	8.29
Mean Diurnal Range (in °C)	8.49	8.70	8.99
Isothermality (in %)	31.00	31.72	32.05
Temp. Seasonality (in standard deviation)	6624.12	6715.44	6935.49
Max. Temp. of Warmest Month (in °C)	22.11	22.69	23.20
Min. Temp. of Coldest Month (in °C)	-5.64	-4.74	-4.03
Temp. Annual Range (in °C)	26.68	27.13	28.06
Mean Temp. of Wettest Quarter (in °C)	15.96	16.47	16.86
Mean Temp. of Driest Quarter (in °C)	0.45	1.00	2.50
Mean Temp. of Warmest Quarter (in °C)	15.96	16.47	16.86
Mean Temp. of Coldest Quarter (in °C)	-1.53	-1.01	-0.54
Annual Precipitation (in mm)	689.86	764.58	946.46
Precipitation of Wettest Month (in mm)	80.97	95.49	118.90
Precipitation of Driest Month (in mm)	41.42	44.87	51.29
Precipitation Seasonality (coefficient of variation)	20.91	25.76	30.30
Precipitation of Wettest Quarter (in mm)	230.63	271.73	339.16
Altitude (in m)	387.20	453.67	550.00

Table 4.5.: Overview of the categorical environmental variables for Red Kite breeding. "0%" means that the characteristic is not present in the observation cell; "$>$ 0%" means that the characteristic is present. Values in the table represent the absolute frequency of observing the corresponding category.

Description	0%	$>$ 0%
Water Coverage	1501	417
Wasting Asset	1734	184
Dumping Ground	1911	7
Rocks	1866	52
Heather and Moors	1875	43
Industry	1572	346
Broad-leaf Forests	1245	673
Moors	1827	91
Orchards	1833	85
Bogs	1830	88
Traffic	1883	35
Shrubs	1570	348
Rivers	1776	142
Lakes	1650	268
Vineyard	1874	44
Cities and Villages	284	1634

tional smooth, spatial component to account for spatial correlation (possibly varying for study period), and a spatially varying effect for altitude. Furthermore, we apply monotonic restrictions (all decreasing) on the effects of 'Coniferous Forest', 'Cities and Villages', 'Precipitation of Wettest Month' and 'Precipitation of Wettest Quarter'. The effects of these environmental variables as estimated by Hothorn *et al.* (2011c) are difficult to interpret because of local extremes (minima and maxima) and erratic fluctuations. For example, small coniferous forests are sufficient and often used for breeding. The surrounding non-forest land is the major foraging area, and thus its composition is the main limitation factor of a grid cell (Mebs and Schmidt 2006). Thus, it is appropriate to assume a monotonic decreasing influence of coniferous forests on Red Kite breeding. Similarly for climatic variables, such as the wettest quarter, one has to expect a monotonic decrease of the effect on Red Kite breeding as this variable determines the density of ground vegetation. With an increasing density caused by increasing precipitation, the availability of the major prey will decrease. We therefore expect a positive effect only with lower levels of precipitation and an increasing negative effect with higher levels. Again, here the observational data underlying the model may lead to artificial bumps in

the estimated functions. One should note that it is of particular importance to specify monotonic constraints for both variables of precipitation as these variables are highly correlated and hence share a lot of information. The reasons for our belief in monotonicity are the same for both variables. Furthermore, if one effect remains unconstrained, it may capture some of the erratic behavior left over from the constrained estimate. In this case, the model and its interpretation could be misleading. For the proportion of cities and villages, we also have to expect a monotonically decreasing effect because the species may forage even in the vicinity of small villages, but will clearly avoid larger cities, possibly because of human disturbance and a decrease in foraging habitats, such as meadows, and nesting habitats, such as forests (Mebs and Schmidt 2006).

Here, we estimated two different models: (1) an additive model with monotonicity constraints on the effects of four variables (`'mono'`), and (2) an unconstrained model (`'add/vary'`; cf. Hothorn *et al.* 2011c). The latter model is used as a competitor to assess how monotonicity constraints of some variables affect the estimates of these and other effects. For the monotonicity constrained model `'mono'`, we applied P-splines with 4.5 degrees of freedom and 20 inner knots for all continuous variables. The spatial effect, the change of the spatial effect between the two time periods and the spatially varying effect for the altitude were specified as bivariate P-splines with 4.5 degrees of freedom and 6 inner knots per coordinate. The variables 'Coniferous Forest', 'Precipitation of Wettest Month' and 'Precipitation of Wettest Quarter' were estimated using monotonic P-splines with 4.5 degrees of freedom and 20 inner knots while the effect of 'Cities and Villages' was estimated with a monotonicity constrained, ordinal factor. The penalty parameter that enforces monotonicity was chosen, as recommended by Eilers (2005), as $\lambda_2 = 10^6$. The unconstrained model `'add/vary'` only differs in the four constrained effect estimates, where standard P-splines and categorical effects were estimated.

The optimal stopping iteration was estimated separately for the monotonicity-constrained and the unconstrained model via stratified 25-fold bootstrap, i.e., we randomly selected grid cells, which represent our observation units, and not single observations. Hothorn *et al.* (2011c) used stability selection (Meinshausen and Bühlmann 2010), which is briefly discussed in Section 2.5 in this thesis, to extract the relevant covariates for interpretation. We used the same covariates to be able to assess the differences or similarities of the two models. Identification of influential variables with an error control for the inclusion of non-informative terms was not of primary interest in this study but could be applied directly to the monotonic

model as presented in Hothorn *et al.* (2011c). Code to fit the Red Kite breeding distribution can be found in Appendix B.4.

4.5.2. Results

The resulting effect estimates of the two models are depicted in Figure 4.11. The upper row depicts the effects that were estimated with monotonicity constraints in model 'mono'. We can conclude from this model that Red Kites prefer dry areas with broad-leaf forests or mixed forests and avoid especially coniferous forests for breeding. Areas with a mixture of meadows and trees are more likely to be used as Red Kite breeding habitats as indicated by our model.

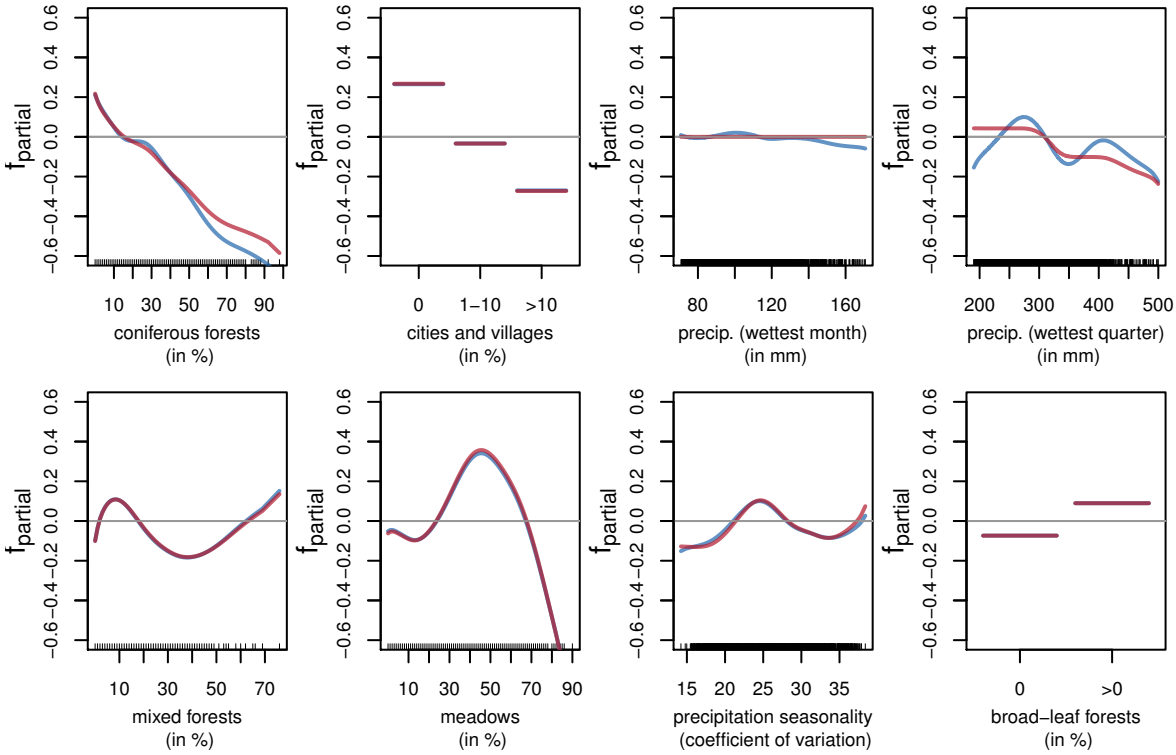
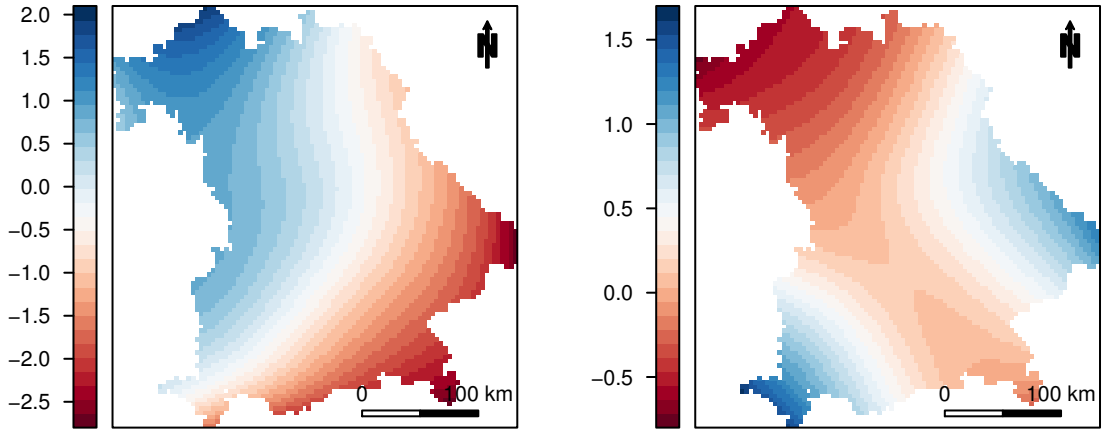


Figure 4.11.: Estimated partial effects (i.e., the other effects are set to zero in each graph) of the environmental variables on Red Kite breeding. Estimates from the unconstrained model 'add/vary' are given in blue. The effect estimates from the model with monotonicity constraints 'mono' are given in red, where the effects depicted in the upper row of graphs were subjected to a (decreasing) monotonicity constraint.

In the model, we did not only consider the global effects of the measured environmental variables, but also local effects, i.e., effects that depend on the observation point. The spatial component in the model allows us to capture the spatial autocorrelation, i.e., observations that are close in space are more likely to be simi-

lar than observations that are further apart. The spatial autocorrelation, depicted in Figure 4.12, shows a large effect that cannot be attributed to other measured environmental variables. Moreover, we found a change in the spatial pattern over time. The spatial autocorrelation of the first period (1979–1983) is given Figure 4.12(a); the difference of the second period (1996–1999) to the first observation period is given in Figure 4.12(b). This difference can be interpreted as a difference in log-odds ratio between the two study periods. The northwestern part of Bavaria underwent a change toward lower breeding probabilities in the second observation period. The central part of Bavaria showed no changes in the breeding probabilities. The southwestern and eastern regions of Bavaria showed increased probabilities.



(a) Spatial Autocorrelation (1979 – 1983)

(b) Change of Spatial Autocorrelation

Figure 4.12.: Left: Spatial autocorrelation for Red Kite breeding in the monotonic model `'mono'`. This model component captures variability in the data that cannot be attributed to other environmental variables. In northwestern Bavaria, the breeding probability is higher while in southeastern Bavaria, the breeding probability is lower. Right: Change of the spatial autocorrelation between the two time periods 1979–1983 and 1996–1999. The figure represents the difference to the first period.

The spatially varying effect of altitude is depicted in Figure 4.13. Altitude showed a positive effect in the northwestern part of Bavaria. These areas are relatively low. Compared to areas at similar altitude in the rest of Bavaria this region showed increased breeding probabilities at this low altitude. The mountainous eastern and the southern parts of Bavaria showed a negative effect of altitude, i.e., higher altitudes induce a decrease in breeding probabilities.

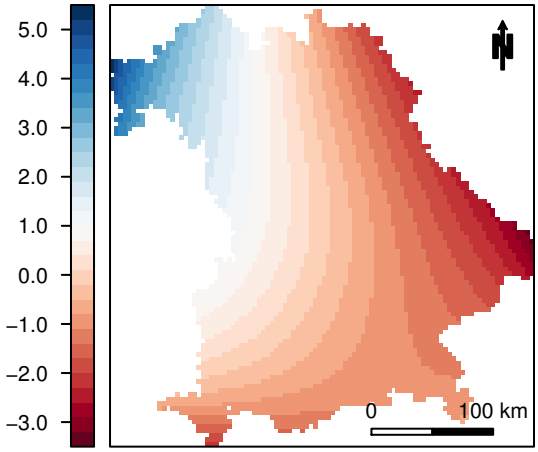


Figure 4.13.: Spatially varying effect of altitude in the monotonic model `'mono'`. The surface can be interpreted as the coefficient estimate of altitude at a given point in Bavaria. Altitude has a negative effect on the probability of Red Kite breeding in the east, i.e., with increasing altitude there, the breeding probability decreases. In the northwestern part of Bavaria, altitude has a positive effect on the breeding probability, i.e., with increasing altitude there, the probability of breeding increases.

Comparison with the Unconstrained Model Comparing the global effects of the two models (see Figure 4.11), we can conclude that the effect estimate of 'Coniferous Forests' shows a decreasing trend for model `'add/vary'`; however, model `'mono'` shows an even smoother, monotonic fit and keeps the main functional form. The ordinal variable 'Cities and Villages', which represents the population density, had a monotonically decreasing effect in the original model. To keep this effect, we specified a monotonic constraint. In this case, the effect was still monotonically decreasing, and thus the additional penalization hardly affected the estimation. In the case of 'Precipitation of Wettest Month', we observed a small, rather erratic effect in model `'add/vary'`, which completely vanished when we restricted the effect to be monotonically decreasing. At the same time, the effect of 'Precipitation of Wettest Quarter' was estimated as a small but clearly monotonically decreasing effect. With respect to the remaining, "unconstrained" effects, the two models were very similar. The unconstrained smooth and categorical effects were practically identical (Figure 4.11). To further assess the differences between the additive model `'add/vary'` and the monotonic model `'mono'`, we plotted the differences of the estimated local components, i.e., the estimated spatial, spatial-temporal, and

spatially varying effects, in Figure C.3 (see Appendix C.3). We conclude that, compared to the range of the effect estimates, the differences between the monotonic model and the non-restricted model are negligible in all three cases. This shows that the model estimation in boosting is very stable with respect to the unconstrained effects, i.e., constraining some of the effects where it is ecologically reasonable hardly influences the estimates and the interpretation of other variables. Nevertheless, monotonic effects are useful as they allow models that are easier to interpret and reflect the subject-matter knowledge of ecologists better than unconstrained effect estimates.

4.6. Activity of Roe Deer *(Capreolus capreolus)*

In the Bavarian Forest National Park (Germany) a management scheme for wild animals is applied and investigated. Within this scheme, researchers try to understand the activity profiles of different species such as Lynx, Wild Boars and Roe Deer. Here, we focus on the activity of European Roe Deer *(Capreolus capreolus)*.

According to Stache, Hothorn, Heurich, and Heller (2010), animal activity is influenced by exogenous factors such as the azimuth of the sun (i.e., day and night rhythm and seasons), temperature, precipitation and the depth of snow. Another important role play endogenous factors such as the species (e.g., reflected in their diet; Roe Deer are browsers), age and sex. Additionally, as Roe Deer tend to be solitary animals, a high level of individual specific variation in activity is to be expected.

Collection of Activity Data The data was collected in the Bavarian Forest National Park. The park comprises 244 km^2 and is situated in southeast Germany on the border to the Czech Republic. The wooded, low mountain region is situated at an altitude of 650m to 1450m above sea level (Stache *et al.* 2010).

The activity data on Roe Deer was recorded by telemetric necklaces (Vectronic Aerospace, Berlin). The necklace contains a GPS-GSM module (records and sends position), a temperature sensor (records a mixture of ambient temperature and the body temperature of the animal), and an acceleration sensor unit with two sensors for two different acceleration directions, which records the activity of the animal. The sensors measure the acceleration continuously and save it every five minutes. All animals that were tagged were living within an area where Roe Deer management (e.g., supplementary winter feeding and culling) ceased in 2007. Precipitation

Table 4.6.: Overview of the characteristics for the 29 Roe Deer given in absolute and relative frequencies. Necklace (*) has a relative frequency $> 100\%$ in total as three Roe Deer were tagged with two different necklaces within the observation period. Hence, 29 Roe Deer were tagged with 32 sensors.

	Frequency	Rel. frequency (in %)
Sex		
Female	9	31.0
Male	20	69.0
Age		
0 - 1 year	10	34.5
$\geq$ 2 years	19	65.5
Necklace (*)		
Type 0	23	79.3
Type 1	8	27.6

and depth of snow were gathered from a local weather station (Waldhäuser, altitude 940m). For more details on Roe Deer activity, the study area as well as on further technical background of the collars and a first analysis of the data set we refer to Heller (2009), and Stache *et al.* (2010).

Data Analysis The data set contains records for 29 European Roe Deer in the years 2006 to 2008. Table 4.6 shows the characteristics of the animals that were observed. Acceleration sensor measurements, temperature and precipitation were aggregated over 15 minutes. This led to a total of 832102 observations (see Table 4.7). We used the sum of both acceleration sensors to represent the Roe Deer activity in this analysis. The activity is represented by a number ranging from 0 to 510, where higher values represent higher activity. An overview is given in Table 4.8.

Activity profiles for the day and for the year should be provided. As earlier

Table 4.7.: Absolute and relative frequencies of records per year.

	Frequency	Rel. frequency (in %)
Year of Observation		
2006	315931	38.0
2007	312080	37.5
2008	204091	24.5

Table 4.8.: Overview and summary statistics of the numeric variables (minimum, first, second and third quartile, and maximum) for Roe Deer activity.

Description	min	25%	50%	75%	max
Activity	0.00	2.00	22.00	76.00	510.00
Temperature (in °C)	-15.77	0.88	7.30	12.56	29.20
Precipitation (in mm)	0.00	0.00	0.00	4.56	76.00
Depth of Snow (in cm)	0.00	0.00	0.00	13.37	191.00

analysis showed, the activity of male and female animals is very different. Hence, sex should be considered as an effect modifier in the analysis by defining sex-specific activity profiles. We considered a Gaussian model with additive predictor

$$\begin{aligned}\mathbb{E}(\text{activity}|\cdot) = \mathbf{x}^\top \boldsymbol{\beta} &+ f_1(\text{calendar day}) \cdot \text{temperature} \\ &+ f_2(\text{calendar day}) \cdot \text{depth of snow} \\ &+ f_3(\text{calendar day}) \cdot \text{precipitation} \\ &+ f_4(\text{time}, \text{calendar day}) \\ &+ f_5(\text{time}, \text{calendar day}) I_{(\text{sex = male})} \\ &+ b_{\text{roe}},\end{aligned} \tag{4.18}$$

where $\mathbf{x}$ contains the categorical covariates sex, type of necklace, year and age. Temperature, depth of snow and precipitation entered the model rescaled to $|x| \leq 1$ by dividing the variables by the respective absolute maximum values. The effects of temperature (f_1), depth of snow (f_2) and precipitation (f_3) depend on the calender day. An interaction surface (f_4) for time of the day and calender day is specified to flexibly model the daily activity profiles throughout the year. An additional effect for male Roe Deer is specified with f_5. Finally, a random intercept b_{roe} for each Roe Deer is included. All smooth base-learners (corresponding to f_1 to f_5) and the random intercept were specified with three degrees of freedom (remember: cyclic splines have a null space consisting of a constant only). The time-varying effects f_1 to f_3 were specified as P-splines with cyclic constraint. The interaction surfaces f_4 and f_5 were specified as cyclic tensor product P-splines with 12 interior knots per variable (i.e., 144 interior knots in total). For all other values, the default was taken. Overfitting did not occur in this data set with approximately 800,000 observations and only nine base-learners. Hence, early stopping by using cross-validation was not possible. To allow for enough flexibility in the model while keeping the running

time relatively low, we fitted 1000 boosting iterations with a step length of $\nu = 0.1$.

4.6.1. Results

Figure 4.14 shows the selected base-learners and their partial contribution to the model. The partial contributions of the components corresponding to f_4 and f_5 show the highest variation and hence explain the most. The individual activity of the Roe Deer b_{roe} gives also a substantial contribution to the total predicted variation. The time-varying effects of temperature and depth of snow (f_1 and f_2) show minor contributions. Finally, the type of necklace explains additionally some of the variability. The overall explained variability $R^2 \approx 10.1\%$. Despite the fact that this seems quite low, the resulting model can be considered very reasonable: The estimated effects are biological very meaningful and represent current researchers' knowledge on Roe Deer (see Stache *et al.* 2010, and the references therein).

The most prominent components are the interaction surfaces referring to the time-dependent activity profiles for male and female Roe Deer (Figure 4.15), given climatic variables and characteristics of the Roe Deer. The profiles show that Roe Deer are most active in the twilight phases in the mornings and evenings. This holds for the whole year and for both, male and female animals. In general, the activity profile for female and male Roe Deer is very similar, but male activity is on a much higher level and has more variability. The activity of Roe Deer is strongly influenced by the season: During summer, the activity is on a much higher level

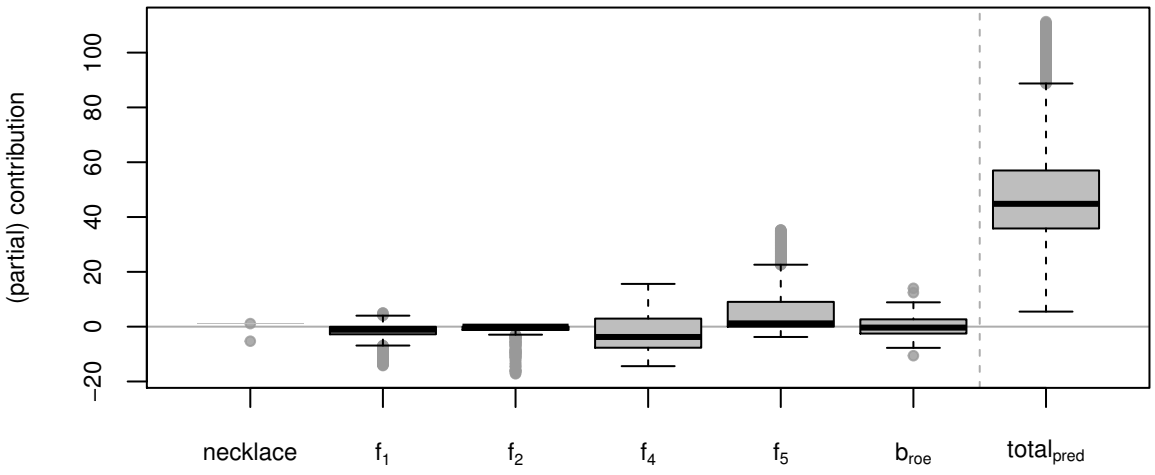


Figure 4.14.: Partial contributions of the selected modeling components and total predicted values. The notation f_1 to f_5 refers not only to the time-varying effect but to the corresponding, complete base-learner. As necklace is a factor, only the effect sizes are given and no boxplot is shown.

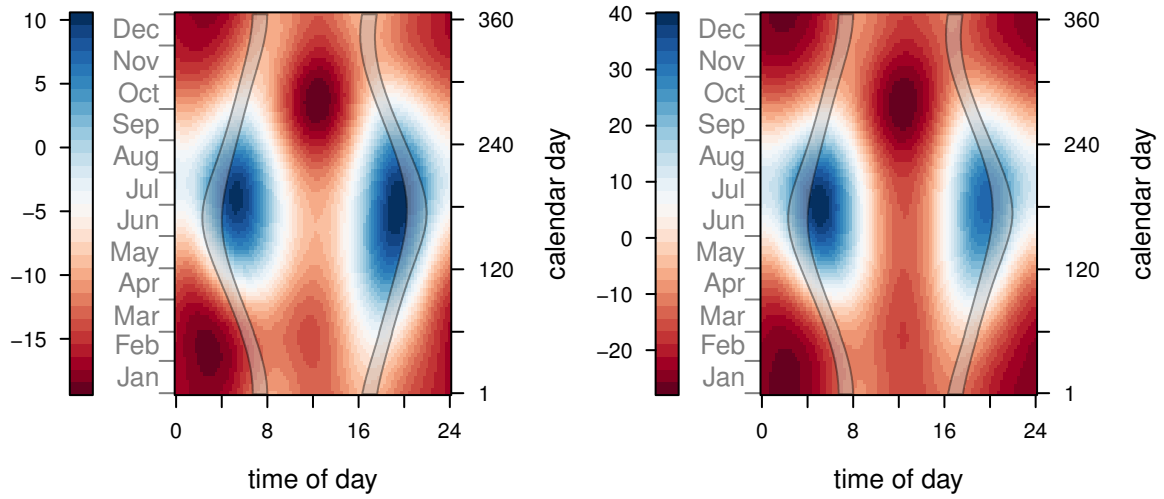


Figure 4.15.: Influence of time on roe deer activity: Combined effect of calender day and time of day for female roe (= f_4; left) and the effect of male roe (= $f_4 + f_5$; right) together with twilight phases (gray).

throughout the complete day. The phase of least activity is around noon. This is even enhanced in autumn. In spring, activity is more evenly distributed over the daytime and the hours after midnight are the least active.

To control for the individual activity of each Roe Deer, a random intercept base-learner was included in the model. Figure 4.16 depicts the estimated random effect for each Roe Deer. Both sexes have a relatively high variation within individuals with a range of approximately 20 units. Some animals, such as the 3rd Roe Deer, have a much higher individual activity while others, such as the 4th female Roe Deer, have a much lower personal activity.

The effects of the climatic variables are depicted in Figure 4.17. One can see that a higher temperature leads to lower activity (negative effect of temperature) with the exception of January until March, where higher temperatures lead to an increase in Roe Deer activity (positive effect of temperature). The depth of snow has a negative effect on Roe Deer activity throughout the year: Deeper snow leads to a decreased activity. The effect of the depth of snow is stronger in the summer month (where there is hardly any snow), and it is less strong in January and February. However, in the latter period the depth of snow is the deepest.

Discussion With this type of collars and the resulting activity measurements, one can access the overall activity of an animal. The source and type of activity cannot

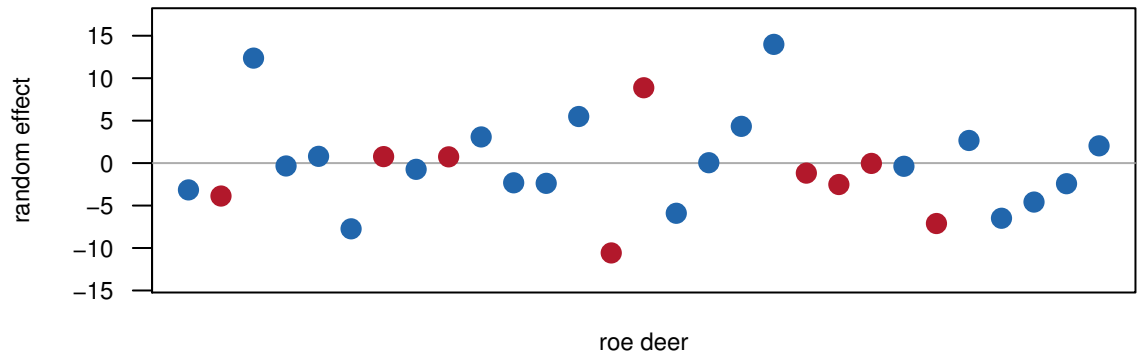
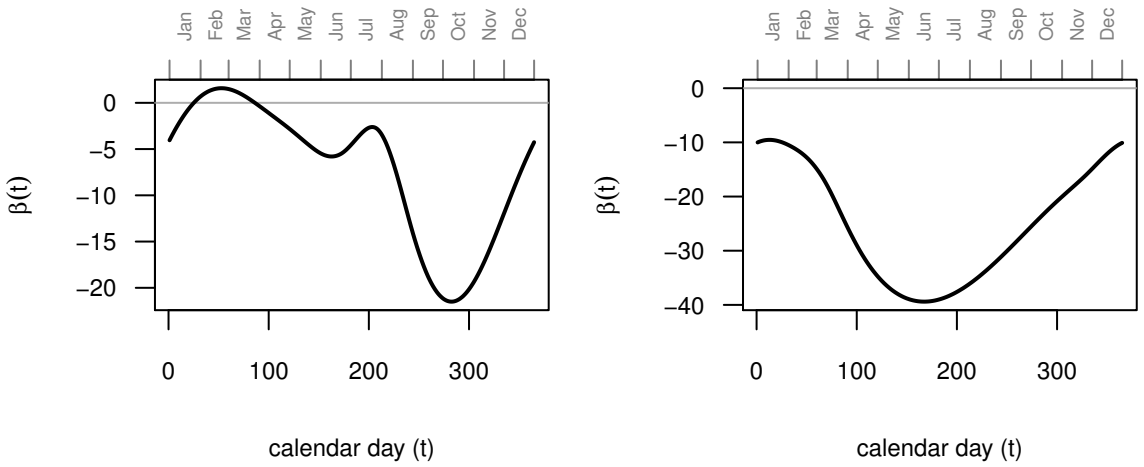


Figure 4.16.: The random intercept of activity for each Roe Deer. The colors code the sex of the animal, where blue means 'male' and red means 'female'. There is a huge difference in the activity for the animals (30 units). No difference can be observed for sex.



(a) Varying Effect of Temperature

(b) Varying Effect of Depth of Snow

Figure 4.17.: The time-varying effect (i.e. '$\beta(t)$') of temperature is depicted in (a) and the time-varying effect of depth of snow in (b). Note that both variables are rescaled, i.e., $\beta(t)$ is the maximal effect. At a given day, the effects of temperature and depth of snow are linear. The higher the amplitude for a given day, the stronger the effect. Snow always reduces the activity while snow in late winter has the smallest negative effect. While high temperatures have a negative effect on activity for most of the year, in early spring it increases the activity. However, only in the late summer (August to October) the effect is "strong".

be monitored by aggregated data. Furthermore, mounting the collars at different areas of the animal's neck can lead to varying measurements (Stache *et al.* 2010). Hence, part of the intra-individual variation might be due to the collar itself while the rest of it is due to individual differences of the animals. A further source of variability of the measurements is induced by the two different series of the

necklaces.

The endogenous covariate age did not contribute to the model. This indicates that there is no difference in adult and sub-adult animals, at least in our study period. The year of observation did not play a role either. Hence, the changes in Roe Deer management in the National Park in 2007 do not seem to have influenced the Roe Deer activity.

The most relevant factors for activity were the light intensity as represented by the seasonal components, and the individual activity levels of the animals. This coincides with the idea of Roe Deer being solitary, crepuscular animals and with the results from other studies (see Stache *et al.* 2010, and the references therein). Overall, the explained variance is relatively low. However, the modeling strategy that we applied resulted in interpretable and biological meaningful models that agree with existing knowledge.

4.7. São Paulo Air Pollution

Air pollution studies try to investigate the impact of various air pollutants on people's health or mortality (see e.g., Spix *et al.* 1993, Saldiva *et al.* 1995). Many modern studies concentrate on particulate matter such as PM_{10} or $PM_{2.5}$ (particulate matter with aerodynamic diameter $\leq 10\mu$m and $\leq 2.5\mu$m, respectively). Another area of current research interest focuses on other pollutants that are due to car traffic or industry such as sulfur dioxide (SO_2), carbon monoxide (CO) or ozone (O_3). In this section we will analyze data from São Paulo, Brazil, which was collected from January 1994 to December 1997. In this study all above mentioned pollutants were measured together with climatic variables such as temperature and humidity. Furthermore, data on deaths in various age groups caused by respiratory and non-respiratory causes was collected.

We focus on elderly people (65+ years; cf. Saldiva *et al.* 1995) and consider only the pollutant SO_2. The impact of the temperature and the concentration of the pollutant were considered to have a delayed influence with a lag of 2 days while humidity was used without lag, as Conceição *et al.* (2001) proposed. Surely, the model could be refined by using an auto-regressive process or nonlinear time-series models (i.e., smooth effect estimates for lagged outcomes) as proposed in the boosting context by Robinzonov, Tutz, and Hothorn (2011). However, we want to compare our results to the results of Leitenstorfer and Tutz (2007) who modeled

Table 4.9.: Overview and summary statistics of the numeric variables (minimum, first, second and third quartile, and maximum); Covariates that are used with a lag of 2 days in the analysis are indicated by *.

Description	min	25%	50%	75%	max
Number of respiratory deaths (p. day)	2.00	9.00	12.00	15.00	35.00
Time (in days)	3.00	346.50	708.00	1123.50	1461.00
SO_2 (in $\mu g/m^3$)*	1.57	12.39	18.67	27.74	76.13
Min. temperature (in °C)*	0.60	12.80	15.10	18.00	21.90
Relative humidity (in %)	42.85	75.81	81.85	86.91	96.53
Number of deaths (p. day, other causes)	34.00	56.00	63.00	71.00	118.00

the number of deaths with lagged influence of SO_2 and temperature but without an *additional* correction for the time-series nature of the data. An overview of the numerical variables with summary statistics can be found in Table 4.9. In addition to these covariates we included the day of the week in the analysis to account for variations throughout the week. To account for changes throughout the year we added the day of the year to the analysis to account for seasonal patterns. To account for large scale variation over the years, we additionally modeled a (long-term) trend. The raw data of the daily concentration of SO_2 and the daily mortality (caused by respiratory diseases) is depicted in Figure 4.18. An unadjusted correlation between the number of deaths and the concentration of the pollutant is revealed. However, we can clearly see a seasonal variation within the number of deaths and the concentration. So, the apparent effect of the pollutant might just be due to the seasonal variation. Thus, a model taking seasonal effects into account is required to assess the effect of the pollutant.

The response, number of respiratory deaths in elderly people, consists of counts for rare events. Hence, the effect of the pollutant SO_2 corrected for climatic variables and effects of time (on a large scale, i.e., the yearly pattern and the trend, and on a small scale, i.e., the weekly pattern) was modeled using an additive Poisson model with log-link. The effect of the pollutant SO_2 was modeled with monotonically increasing constraint, which reflects the expected dose-response relationship. If SO_2 has an impact on health in elderly people, one expects higher concentrations of the pollutant to result in higher mortality. We model the number of respiratory deaths as

$$\mathbb{E}(\text{respiratory death}|\cdot) = \exp\Big\{\mathbf{x}^\top \boldsymbol{\beta} + f_1(\text{time}) + f_2(\text{SO}_2)\Big\},$$

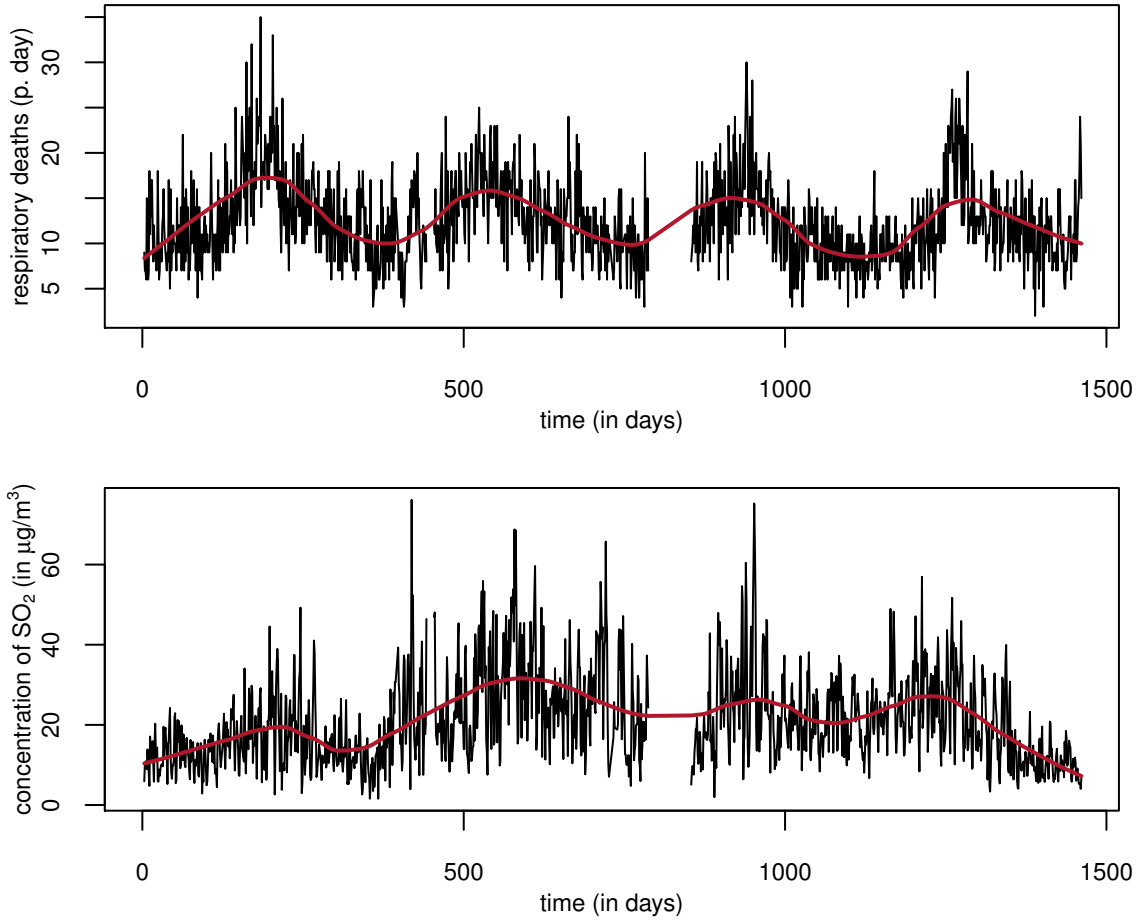


Figure 4.18.: Daily measurements of the number of respiratory deaths and SO_2 in São Paulo together with loess smoother.

where **x** represents the parametrically modeled covariates temperature, humidity, number of non-respiratory deaths and the days of the week with 0-1 dummy-coding where Sunday is the reference category. The smooth function of time, f_1, is decomposed into two parts: a seasonal component f_{season} and a component for the trend f_{trend}, yielding $f_1(\cdot) = f_{\text{season}}(\cdot) + f_{\text{trend}}(\cdot)$. The seasonal component is modeled as a cyclic effect for the day of the year to assess the structure of respiratory deaths throughout the year. The trend is modeled as a smooth effect over time. The smooth effect of SO_2, f_2, is modeled with monotonic constraint. Technically, we modeled f_{season} as a P-spline of order 3 with 20 inner knots, a second order difference penalty and fixed boundary knots at day 0 and 365. The trend was modeled using the P-spline decomposition (Kneib *et al.* 2009) with a linear effect of time and a reparameterized P-spline (see Equation (3.9)), with one degree of freedom, 20 inner knots and differences of order one. The monotonic effect was modeled as

a P-spline of order three with increasing constraint and again a difference penalty of order one. The other covariates were included as linear base-learners.

Leitenstorfer and Tutz (2007) used almost the same model with some minor differences. They did not split the large scale seasonal component f_1 into two parts, one for changes over the years and one for the periodic effect within one year. They used only one smooth function to model the influence of time for all four years together. Thus, in their model, the years can have a different *level* of the effect but as well a different *shape* of the seasonal effect. While this approach might offer slightly more flexibility, it is harder to assess the common periodic effect within a calendar year. Furthermore, Leitenstorfer and Tutz (2007) specified separate base-learners for each day of the week while we use one base-learner for the complete covariate 'day of the week'. The most notable change, however, is the fitting approach. Leitenstorfer and Tutz (2007) also used a component-wise boosting approach to model the monotonic effect of SO_2. They use the same idea as proposed in Section 4.2.1: They exploit the fact that it is sufficient for monotonicity that the differences of adjacent coefficients have the same sign. However, they use a different approach to achieve monotonic coefficient sequences. They group the B-splines and assign one coefficient to each of the two groups. Hence, the P-spline function estimate (2.12) is simplified to

$$\hat{f}(x) = \hat{\beta}_{(1)} \left(\sum_{j=1}^{r} B_j(x) \right) + \hat{\beta}_{(2)} \left(\sum_{j=r+1}^{J} B_j(x) \right).$$

Monotonicity holds if the 'grouped' coefficients fulfill $\hat{\beta}_{(1)} \leq \hat{\beta}_{(2)}$. Estimation of the grouped coefficients is done for all possible splits $r = 1, \ldots, J-1$. Each of the groupings represent one base-learner and only the best-fitting base-learner is updated in the component-wise boosting approach. However, only splits for which the monotonicity condition holds can be selected. The final estimate is then the sum over all estimates of $f(x)$, which is the sum of monotonic functions and thus monotonic as well. Hence, the fitting procedure of $f(x)$ itself is not monotonicity constrained but the final model fit resulting from the boosting algorithm is monotone in x. Note that Leitenstorfer and Tutz (2007) do not use unpenalized B-splines. They additionally apply a ridge penalty on the B-spline coefficients to avoid too large jumps, i.e., they use P-splines with differences of order zero.

In addition to our monotonic approach (`'mboost:mono'`), which is implemented in the package **mboost**, and the monotonic approach of Leitenstorfer and Tutz

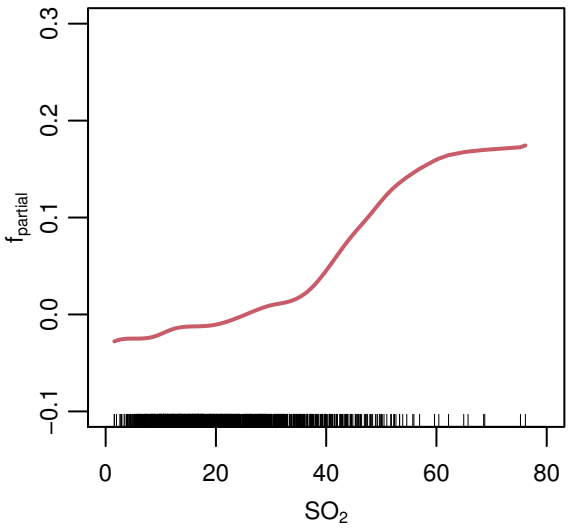


Figure 4.19.: Estimated effect of SO_2 for the monotonic model '`mboost:mono`' centered around zero.

('`LT2007`'), we fitted a model without monotonicity constraint ('`mboost:uncons`') via **mboost**, and an unconstrained additive model with the function `gam` in the R package **mgcv** (Wood 2006a, 2010) ('`mgcv`'). The latter model was also used for comparison in Leitenstorfer and Tutz (2007).

4.7.1. Results

The estimated smooth effect for the pollutant SO_2 resulting from the monotonic model '`mboost:mono`' (Figure 4.19) indicates that an increase of the pollutant's concentration does not result in a (substantially) higher mortality up to a concentration of $40\,\mu g/m^3$. From this point onwards a steep increase in the expected mortality can be observed which flattens again for concentrations above $60\,\mu g/m^3$. Hence, a dose-response relationship can be observed where higher pollutant concentrations results in a higher expected mortality. At the same time the model indicates that increasing pollutant concentrations are almost harmless until a threshold is exceeded, and the harm of SO_2 is not further increased after an upper threshold. Investigating the effect of PM_{10} Saldiva *et al.* (1995) found no "safe" threshold in their study on elderly people in São Paulo. They also investigated the effect of SO_2 but did not report on details, such as possible threshold values, in this case. The more recent study on air pollution in São Paulo for children (Conceição *et al.* 2001) used only linear effects for pollutant concentrations. Hence, no threshold values

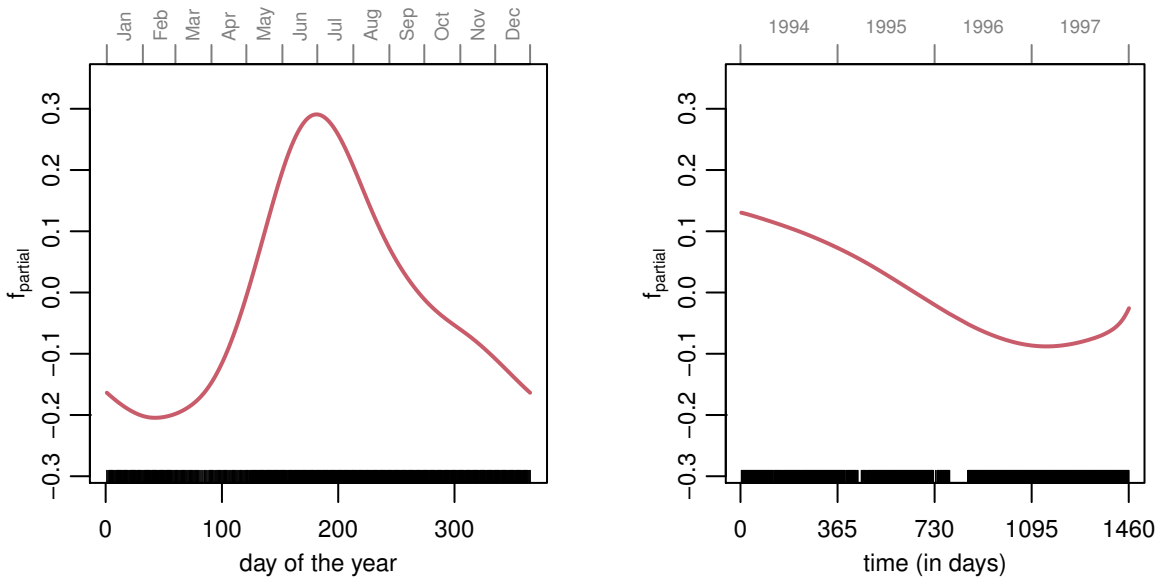


Figure 4.20.: Seasonal effect (left) and long-term trend (right) for the monotonic model `'mboost:mono'`.

can be estimated.

As we used a cyclic constraint for the seasonal effect, the ends of the function estimate meet, i.e., the days 365 and 1 are smoothly joined (see Figure 4.20, left). The effect shows a clear peak in the cool and dry winter months (May to August, as we are on the southern hemisphere) while a decreased risk of mortality in the warm summer months can be observed. This is in line with other studies (e.g., Saldiva *et al.* 1995). Looking at the trend over the years (Figure 4.20, right) we see a decrease in mortality from 1994 to early 1997 and an increase thereafter. However, one should keep in mind that this trend is combined with the periodical effect to form the complete temporal pattern.

The estimated linear effects for all four models are given in Table 4.10. At first we focus on the monotonic model `'mboost:mono'`. It shows a negative effect of humidity indicating that higher humidity reduces the expected number of deaths. The minimum temperature shows no effect, at least no additional effect to the seasonal component, which might capture a temperature effect as well. Regarding the days of the week, an increase in mortality on Monday can be observed. This might be due to different behavior and thus personal exposure to the pollutant on weekends or, more likely, due to a lag in recording on weekends.

Comparison of Modeling Strategies In contrast to the monotonic estimate, the effect estimate of `'mgcv'` for the pollutant SO_2 is very wiggly for small values up

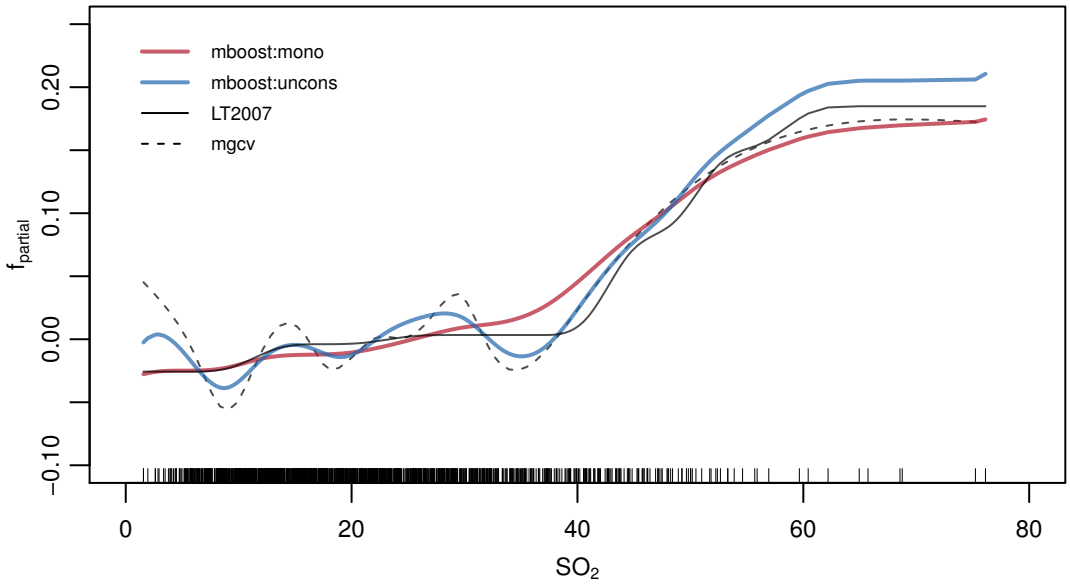


Figure 4.21.: Comparison of the four modeling approaches. The estimated effects of SO_2 are centered around zero. Blue lines represent the effect of the `'mboost:uncons'` model, red lines represent the effect of the `'mboost:mono'` model. The black lines correspond to the `'LT2007'` model (solid) and the `'mgcv'` model (dashed).

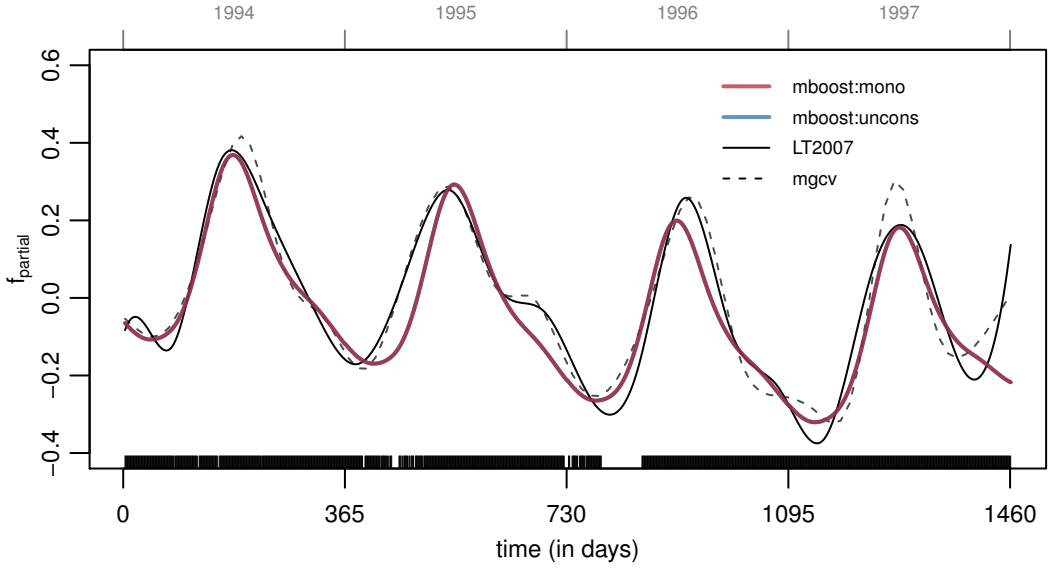


Figure 4.22.: Effect of time where the long-term trend is combined with the seasonal pattern. The blue line (not visible as this model coincides with `'mboost:mono'`, red line) represents the effect of the `'mboost:uncons'` model, the red line represents the effect of the `'mboost:mono'` model. The black lines correspond to the `'LT2007'` model (solid) and the `'mgcv'` model (dashed).

Table 4.10.: Estimated coefficients of all four models. The model of main interest is highlighted in gray.

	'mboost:mono'	'LT2007'	'mboost:uncons'	'mgcv'
(Intercept)	2.5215	2.5383	2.5215	2.6246
Min. temperature	0.0000	−0.0014	−0.0021	−0.0015
Humidity	−0.0026	−0.0028	−0.0028	−0.0033
Monday	0.0385	0.0250	0.0387	0.0316
Tuesday	−0.0256	−0.0449	−0.0264	−0.0397
Wednesday	−0.0107	−0.0290	−0.0113	−0.0250
Thursday	−0.0353	−0.0503	−0.0345	−0.0481
Friday	−0.0448	−0.0632	−0.0454	−0.0599
Saturday	−0.0199	−0.0359	−0.0176	−0.0331
Non-respiratory deaths	0.0046	0.0038	0.0046	0.0029

to a concentration of $40\,\mu g/m^3$ (see Figure 4.21). Non-monotonic effects might perhaps be reasonable for large concentrations of the pollutant as people might adjust their behavior (less sport, more indoor activity, ...) as pointed out by a referee in Leitenstorfer and Tutz (2007). However, for large values the violations of monotonicity are minor despite the sparse data base there. For small values, an impact of the pollutant on the behavior of people and thus on their exposure to SO_2 is not expected and the erratic behavior of the estimator is biologically not explicable. The estimated effect of 'mboost:uncons' is less erratic for low concentrations of the pollutant. Using differences of higher order (e.g., of order 2) leads to an even smoother fit for small values, but still a decrease in the estimated function for increasing concentrations of the pollutant can be observed. Hence, to stabilize the smooth estimator for concentrations $< 40\,\mu g/m^3$ and at the same time for larger values with a sparse data base, we fit a monotonically increasing function. Both monotonic models lead to very similar effect estimates but the 'LT2007' model shows more "steps". This can be attributed to the choice of the penalty: 'LT2007' used a ridge penalty while 'mboost:mono' used first order differences. Using higher order penalties leads to a smoother behavior of the estimated function.

The seasonal effect coincides for the models 'mboost:mono' and 'mboost:uncons' (Figure 4.22). The shape of the estimated functions resulting from 'LT2007' and 'mgcv' differs between the years. Still, the estimates are very similar compared to the models resulting from **mboost**. Yet, the estimation of the complete time pattern without decomposition into the trend and the periodical, seasonal effect is less

stable. Especially at the boundaries, the models 'LT2007' and 'mgcv' are highly variable, if not to say, expose an erratic behavior. Modeling the trend and the periodic effect separately may have the disadvantage that some of the small scale changes (e.g. around day 730) are missed. However, without this decomposition, models do not allow a direct inspection of the seasonal effect throughout the year. Hence, decomposing the influence time into seasonal effects and smooth long-term effects seems highly preferable as it offers a stable, yet flexible method to model the data and allows an easier and more profound interpretation.

The four considered models have almost the same linear effects for the covariates (see Table 4.10) or show at least the same direction and magnitude of the effect. Hence, one can conclude that the linear effects in this model are very stable and are hardly influenced by the fitting method — boosting vs. penalized iteratively weighted least squares (P-IWLS; see e.g., Wood 2008) — or constraints — monotonic and cyclic constraints — that are applied to the smooth effects. Furthermore, we see that both boosting methods for monotonic effects, 'mboost:mono' and 'LT2007' result in almost the same model here. However, the **mboost** implementation seems favorable to the current implementation of 'LT2007': First, the **mboost** implementation is more flexible as one can freely specify different orders of penalty and B-spline basis functions. Furthermore, one can use a wide range of base-learner for other variables that are to be included in the model. Second, the current implementation in **mboost** is much quicker than the implementation of 'LT2007'. While the speed of 'LT2007' could be improved to some extend by using a more efficient implementation it seems likely that it will stay less efficient than the penalization based implementation of **mboost** for two reasons: Leitenstorfer and Tutz (2007) compute per monotonic effect $J - 1$ competing base-learners in each boosting iteration. Hence, more base-learners need to be estimated. The second drawback of their approach is that in each boosting iteration at most one grouping of coefficients is updated. Thus, to obtain a smooth function with $k \leq J$ different coefficients β_j, at least $k - 1$ boosting iterations are required. In contrast, the **mboost** implementation uses one base-learner only and commonly updates all coefficients of this base-learner. Third, **mboost** offers a very rich set of tools to further manipulate models or extract information such as coefficients from them. In practice, this is very important.

4.8. Concluding Remarks

In this chapter, we extended the flexible modeling framework based on boosting to allow the inclusion of monotonic or cyclic constraints for certain variables. The monotonicity constraint on continuous variables leads to monotonic, yet smooth effects. Many other approaches to monotonic modeling result in non-smooth function estimates (e.g., Dette *et al.* 2006, de Leeuw *et al.* 2009b, Fang and Meinshausen 2011). Most of these approaches have appealing theoretical properties. In many application contexts such as life sciences in the broadest sense, however, we feel that smooth effect estimates are more plausible and hence preferable. In the limit, i.e., if the 'true effect' is monotonic and if there is little noise in the data, our approach leads to P-spline estimates. Hence, in this case, it does not matter whether monotonic constraints are used or not (see for example the effects of 'Coniferous Forests' and 'Cities and Villages' in the study on Red Kite breeding; Sec. 4.5). Monotonic effects can be furthermore applied to bivariate P-splines. In this case, one can specify different the monotonicity constraint for each variable separately. However, it is ensured, that the resulting interaction surface is monotonic (as specified).

Monotonicity constraints might be especially useful in (but are not necessarily restricted to) data sets with relatively few observations or noisy data as supported by our simulation study (see Sec. 4.3). Another example where monotonic effects were recently applied can be found in Leathwick, Elith, Francis, Hastie, and Taylor (2006). There, the survey method was correlated with the outcome in a way that led to counter-intuitive results. The introduction of monotonicity constraints helped to estimate more appropriate models.

Cyclic estimates can be easily used to model, for example, for seasonal effects. The resulting estimate is a smooth effect estimate where the boundaries are smoothly matched. Cyclic effects can be applied straight forward to model surfaces where the boundaries in each direction should match by using cyclic tensor product P-splines (Section 4.4.2). Furthermore varying coefficient models can be fitted where the smooth, time-varying effect has a cyclic constraint as shown in the analysis of Roe Deer activity (Section 4.6). The idea of cyclic effects could also be extended to ordinal covariates with a temporal, periodic effect — such as days of the week. As for smooth, cyclic effects, the estimate should not depend on the 'gap'. It should not matter if Sunday is chosen as the first day of the week (as common in the USA) or if Monday is chosen (as common in Europe). If no penalty is applied, this naturally holds. However, we would often like to avoid

large jumps from day to day as these are not reasonable. Using the ordinal penalty as introduced in Section 2.3.1 resolves this problem. At the same time, however, the first and last category play a special role as they do not have two neighbors in this coding. Introducing a cyclic penalty like for the smooth effects would help to solve this problem and stabilize estimation for cyclic, ordinal variables.

Finally, both restrictions — monotonic and cyclic constraints — can be mixed in one model: Some of the covariates are monotonicity restricted, others have cyclic constraints and the reminder is modeled, for example, as smooth effects without further restrictions or as linear effects. Such a model is used to model the São Paulo air pollution data in Section 4.7.

Despite the ease of use, monotonicity restrictions should be utilized with care: knowledge of the subject matter should govern the choice of variables that should be modeled monotonically. Monotonicity constraints should be used with great care as, for example, other predominant influences such as strong competitors might govern the species distribution (Austin 2002). Furthermore, a priori assumptions might not always reflect the truth and should be carefully reconsidered if a monotonic effect seems inappropriate. In this context, one should keep in mind that also other models have assumptions and constraints, such as linearity, but these are usually ignored or forgotten. Compared to monotonicity constraints, cyclic effects are easier to use. Cyclic constraints are primarily applied to model seasonal effects, i.e., special effects of time. These are much easier to identify from the context of the data.

Both, monotonic P-splines and cyclic P-splines integrate seamlessly in the functional gradient descent boosting approach as implemented in **mboost**. This allows one to have a single framework to fit possible complex models. Additionally, the idea of asymmetric penalties for adjacent coefficients was transferred from P-splines to ordinal factors. The resulting novel approach to monotonic effect estimation for ordinal variables can be integrated in the boosting framework as well. Because of the built-in selection step in each boosting iteration, the algorithm allows one to decide whether the monotonic effect is present or not (see 'Wettest Month' for Red Kites). The boosting algorithm informs us whether the monotonic effect is present in the data. A truly non-monotonic effect, which is estimated using monotonicity constrained base-learners, will not be selected by the algorithm and hence be set to zero. However, one should be aware that monotonicity constrained and unconstrained effects for one variable cannot be specified in the same model. This would lead to a preference of the unconstrained effect as it can better adapt

to the data in those cases where small violations of the constraint occur. To compare monotonic and non-monotonic effects, it is necessary to estimate two separate models, as we did for the Red Kite breeding distribution. These models can then be easily compared with respect to the stability and interpretability of the results. The goodness of fit could also be assessed, but one should keep in mind that the unconstrained model might fit the data better, yet might be less easy to interpret. The possibility to constrain certain effects in highly flexible statistical methods allows researchers to start with the most flexible and complex model that, in a step-wise refinement process, is then simplified by restricting certain parts of the model to monotonic or even linear functions without losing too much model accuracy. The resulting simpler model will be easier to interpret and to understand, and, finally, to accept.

5. Summary and Future Directions

"And if you do not think about the future, you cannot have one."

(Mr. Montross in John Galsworthy (1928)[1])

High-dimensional or complex data situations are very common nowadays. Tools like computers and smartphones are present in almost every situation of our everyday life and produce massive amounts of data. In scientific environments the development of tiny sensors, microarrays, semi-automated analysis systems for biological or medical samples and other high-tech solutions produce a plethora of measurements.

In this thesis we discussed boosting algorithms as one way to tackle scientific questions that consequently arise from these data sets. Various aspects of the boosting methodology were analyzed. Algorithmic properties were investigated as well as aspects of statistical modeling.

On the algorithmic side, the selection bias in favor of more flexible base-learners was investigated in Chapter 3. We could theoretically proof that a selection bias in favor of categorical variables with more categories exists (Theorem 3.1) and that smooth base-learners are preferred to linear base-learners (Theorem 3.3). The bias is associated with the flexibility of the base-learners, which can be measured by the degrees of freedom. We could show that the usual definition of degrees of freedom in the smoothing literature is not suitable for the specification of competing base-learners with equal flexibility. The definition that naturally followed from our considerations in the boosting context coincides with the definition of the degrees of freedom that is preferred if models are compared with respect to their residual sums of squares (Buja *et al.* 1989). We could show — theoretically and in simulation studies — that the use of penalized base-learners with equal degrees of freedom is able to reduce the source of biased selection. An in-depth evaluation of ridge penalized base-learners, P-spline base-learners (with decomposition and reparameterization) and of the newly derived penalized ordinal base-learners (see

[1] Galsworthy J (1928), *Swan Song*, Heinemann, London. Pt. II, Ch. 6

Sec. 2.3.1) was conducted. Modeling forest health by using the new approach for unbiased selection resulted in a sparse and well interpretable model. The selected variables and modeling alternatives were chosen with an appropriate complexity.

Another goal of this thesis was to expand the modeling possibilities that are available for boosting. Generalized additive models for location, scale and shape as briefly discussed in Section 2.6 expand the class of available models itself. With GAMLSS, models for multiple components of a conditional distribution can be specified and estimated. The loss function that is used, depends on multiple components (i.e., parameters) and, hence, partial derivatives with respect to each component are used in the boosting algorithm. A problem that still needs to be solved in the context of boosting models with multiple prediction functions is early stopping. Multi-dimensional cross-validation quickly becomes computationally demanding to the point of infeasibility. One-dimensional cross-validation can be done on a fine grid containing all possible values up to the initial m_{stop}. Multi-dimensional cross-validation, on the other hand, is only viable on a relatively coarse grid. Consequently, the true optimal combination of stopping iterations might be missed. A finer grid reduces the impact of missing the truly optimal multi-dimensional m_{stop} but increases the complexity and hence the computing time. Using AIC-based pre-stopping (see Sec. 2.4.3) could help to narrow down the area of interest. However, in this case approximate degrees of freedom for all components (e.g., location, scale and shape) of the boosting model are required.

Within a model class (as specified by the loss function in boosting), the base-learners govern the way that partial effects are modeled. Base-learners and model classes can be freely combined. Thus, an enormous range of (new) modeling possibilities emerges. Possibilities to model the influence of predictors include the newly implemented base-learner for radial basis functions (see Sec. 2.3.4). Radial basis functions allow to model interaction surfaces and spatial effects. Moreover, one could think of extending this approach to other (non-spatial) problems: The only thing that is required for the computation of radial basis functions is some notion of distance (or of similarity) between pairs of observations, and between knots in the observation space and the observations themselves. An approach that was recently published uses RKHS regression to model single nucleotide polymorphism (SNP) data (Gianola and van Kaam 2008, González-Recio *et al.* 2008, de los Campos *et al.* 2009). Combining their approach with boosting seems highly promising as variable selection is a notorious problem in high-dimensional SNP data.

In Chapter 4 ways to include constraints into boosting models were discussed.

Two distinct base-learner were derived, namely, a base-learner for monotonicity-constrained effects and a base-learner for cyclic effect estimates. Both constraints can be applied to smooth effects of single covariates as well as to smooth, bivariate interaction surfaces. Monotonic effects were furthermore discussed for ordinal regressors. We evaluated their performance in simulation studies and could show that (if the true effects are monotonic or cyclic, respectively) constrained regression estimates outperform unconstrained estimates (see Sec. 4.3). In three applications we showed the applicability of the proposed methods. In Section 4.5 we used monotonic effects to improve the interpretability of species distribution models for the Red Kite. Section 4.6 was dedicated to cyclic interaction surfaces, which were used to predict the activity of Roe Deer over the day and throughout the year. Finally, we used monotonic *and* cyclic effects together in a time-series model to model the occurrence of respiratory deaths in São Paulo.

In the context of constrained regression, further extensions are self-evident. In many regression contexts, it might, for example, be required that the effect is strictly positive, i.e., $f(x) > 0$ for all values of x. Or one might want the slope to be bounded, i.e., $f'(x) \leq m$ or $f'(x) \geq m$ (cf. Hazelton and Turlach 2011). While the first constraint could be realized in *simple settings* by a suitable link function (e.g., the log-link), both constraints can be also formulated in terms of constraints on the coefficients. For P-splines it is sufficient to require all coefficients to be positive in order to get a positive function estimate. Likewise, a constraint on the (first) differences of the coefficients results in a bounded slope. Another idea is a partial monotonicity constraint. In this case, monotonicity is only claimed over some interval $[a, b] \subseteq [\min(x), \max(x)]$. For partial monotonicity constraints in the context of penalized splines with truncated power series basis see Hazelton and Turlach (2011).

As a tool to measure variable importance and to select important base-learners, we briefly discussed the newly developed stability selection (Meinshausen and Bühlmann 2010) approach in Section 2.5. The generic method is formulated for arbitrary modeling tools. However, a thorough investigation of the properties in the context of boosting seems well advised. Special guidance on the choice and the impact of the possible tuning parameter(s) (the threshold value for the inclusion probability or the average number of selected variables) is needed. Furthermore, the impact of the number of observations n on the detectability of influential variables requires further investigation. In particular with very few observations n stability selection might suffer as it is based on subsampling with sample size $n/2$.

Another issue is the further implication of stability selection: Stability selection results in a set of stable variables. This set does not (necessarily) correspond to a model with a specific tuning parameter m_{stop} but is a "fundamentally new solution" (Meinshausen and Bühlmann 2010, p. 434). To identify influential or — in other words — relevant variables, the set of stable variables is sufficient. However, if one wants to make predictions or interpret the coefficients of this set of variables, things get complicated. One possibility is to use a standard model where m_{stop} is tuned using cross-validation approaches but restrict the focus on the stable variables. This means that one only looks at effects of stable variables but potentially keeps non-stable variables in the final model. This approach was pursued in the context of the species distribution models for the Red Kite (cf. Hothorn *et al.* 2011c, and Section 4.5), where we primarily wanted to enable the interpretation of the ecological mechanisms. If the aim is to make predictions or to evaluate the predictive accuracy of the stable variables *only*, this approach is not sensible, though. In this case it seems preferable to refit the model using only stable variables as possible predictors. Hothorn (2010) raises the question how this affects the out-of-bag prediction error. He argues that the effect depends on the strictness of the error control and, hence, on the sparseness of the 'stable model'. If it is likely that important variables are missed, the prediction accuracy will suffer. Model fitting with stable variables only is not well advised in this case. Further research in the boosting context is required in this light.

Appendices

A. Algebraic Details

A.1. Matrix Algebra for Bivariate P-splines

This section gives a short introduction on matrix algebra with a focus on the computations that are needed for bivariate P-splines. For further details on matrix algebra in statistical applications we refer to Gentle (2007), and Harville (2008). A short introduction to matrix algebra with a focus on spline smoothing is given in Ruppert *et al.* (2003, Appendix A).

Definition A.1 Kronecker product *Let* $\mathbf{X}$ *and* $\mathbf{Y}$ *be matrices of dimensions* $(m \times n)$ *and* $(p \times q)$*, respectively. The Kronecker product* $\mathbf{X} \otimes \mathbf{Y}$ *is then defined as a matrix of dimension* $(mp \times nq)$

$$\mathbf{X} \otimes \mathbf{Y} := \begin{pmatrix} x_{11}\mathbf{Y} & x_{12}\mathbf{Y} & \dots & x_{1n}\mathbf{Y} \\ \vdots & & & \vdots \\ x_{m1}\mathbf{Y} & x_{m2}\mathbf{Y} & \dots & x_{mn}\mathbf{Y} \end{pmatrix}.$$

Definition A.2 Element-wise product *Let* $\mathbf{X}$ *and* $\mathbf{Y}$ *be matrices with equal dimensions* $(m \times n)$*. The element-wise matrix product* $\mathbf{X} \odot \mathbf{Y}$ *is then defined as the* $(m \times n)$ *matrix*

$$\mathbf{X} \odot \mathbf{Y} := \begin{pmatrix} x_{11}y_{11} & x_{12}y_{12} & \dots & x_{1n}y_{1n} \\ \vdots & & & \vdots \\ x_{m1}y_{m1} & x_{m2}y_{m2} & \dots & x_{mn}y_{mn} \end{pmatrix}.$$

A.1.1. Simple example in the context of P-splines

With Definitions A.1 and A.2 we can compute a bivariate P-spline as given in Section 2.3.3 (see Equations (2.19) and (2.20)). In the following, we consider a highly simplified example. Let $\mathbf{B}^{(1)}$ be a (2×2) B-spline design matrix (for variable x_1) and $\mathbf{B}^{(2)}$ be a (2×4) B-spline design matrix (for variable x_2). To compute the combined basis starting from the univariate B-spline bases we need to 'inflate' the

matrices first:

$$\begin{aligned}\mathbf{B}^{(1)} \otimes \boldsymbol{e}_K^\top &= \begin{pmatrix} b_{11}^{(1)} & b_{12}^{(1)} \\ b_{21}^{(1)} & b_{22}^{(1)} \end{pmatrix} \otimes \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix} \\ &= \begin{pmatrix} b_{11}^{(1)} & b_{11}^{(1)} & b_{11}^{(1)} & b_{11}^{(1)} & b_{12}^{(1)} & b_{12}^{(1)} & b_{12}^{(1)} & b_{12}^{(1)} \\ b_{21}^{(1)} & b_{21}^{(1)} & b_{21}^{(1)} & b_{21}^{(1)} & b_{22}^{(1)} & b_{22}^{(1)} & b_{22}^{(1)} & b_{22}^{(1)} \end{pmatrix}. \end{aligned} \tag{A.1}$$

Analogously, we compute the 'inflated' version for the second B-spline:

$$\begin{aligned}\boldsymbol{e}_J^\top \otimes \mathbf{B}^{(2)} &= \begin{pmatrix} 1 & 1 \end{pmatrix} \otimes \begin{pmatrix} b_{11}^{(2)} & b_{12}^{(2)} & b_{13}^{(2)} & b_{14}^{(2)} \\ b_{21}^{(2)} & b_{22}^{(2)} & b_{23}^{(2)} & b_{24}^{(2)} \end{pmatrix} \\ &= \begin{pmatrix} b_{11}^{(2)} & b_{12}^{(2)} & b_{13}^{(2)} & b_{14}^{(2)} & b_{11}^{(2)} & b_{12}^{(2)} & b_{13}^{(2)} & b_{14}^{(2)} \\ b_{21}^{(2)} & b_{22}^{(2)} & b_{23}^{(2)} & b_{24}^{(2)} & b_{21}^{(2)} & b_{22}^{(2)} & b_{23}^{(2)} & b_{24}^{(2)} \end{pmatrix}. \end{aligned} \tag{A.2}$$

Finally, the element-wise product of (A.1) and (A.2) than is

$$\begin{aligned}&\left(\mathbf{B}^{(1)} \otimes \boldsymbol{e}_K^\top\right) \odot \left(\boldsymbol{e}_J^\top \otimes \mathbf{B}^{(2)}\right) = \\ &\quad \begin{pmatrix} b_{11}^{(1)}b_{11}^{(2)} & b_{11}^{(1)}b_{12}^{(2)} & b_{11}^{(1)}b_{13}^{(2)} & b_{11}^{(1)}b_{14}^{(2)} & b_{12}^{(1)}b_{11}^{(2)} & b_{12}^{(1)}b_{12}^{(2)} & b_{12}^{(1)}b_{13}^{(2)} & b_{12}^{(1)}b_{14}^{(2)} \\ b_{21}^{(1)}b_{21}^{(2)} & b_{21}^{(1)}b_{22}^{(2)} & b_{21}^{(1)}b_{23}^{(2)} & b_{21}^{(1)}b_{24}^{(2)} & b_{22}^{(1)}b_{21}^{(2)} & b_{22}^{(1)}b_{22}^{(2)} & b_{22}^{(1)}b_{23}^{(2)} & b_{22}^{(1)}b_{24}^{(2)} \end{pmatrix}, \end{aligned}$$

which is the bivariate B-spline basis.

For the penalty matrices we get

$$\begin{aligned}\mathbf{K}^{(1)} \otimes \boldsymbol{I}_K &= \begin{pmatrix} k_{11}^{(1)} & k_{12}^{(1)} \\ k_{21}^{(1)} & k_{22}^{(1)} \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} k_{11}^{(1)} & 0 & 0 & 0 & k_{12}^{(1)} & 0 & 0 & 0 \\ 0 & k_{11}^{(1)} & 0 & 0 & 0 & k_{12}^{(1)} & 0 & 0 \\ 0 & 0 & k_{11}^{(1)} & 0 & 0 & 0 & k_{12}^{(1)} & 0 \\ 0 & 0 & 0 & k_{11}^{(1)} & 0 & 0 & 0 & k_{12}^{(1)} \\ k_{21}^{(1)} & 0 & 0 & 0 & k_{22}^{(1)} & 0 & 0 & 0 \\ 0 & k_{21}^{(1)} & 0 & 0 & 0 & k_{22}^{(1)} & 0 & 0 \\ 0 & 0 & k_{21}^{(1)} & 0 & 0 & 0 & k_{22}^{(1)} & 0 \\ 0 & 0 & 0 & k_{21}^{(1)} & 0 & 0 & 0 & k_{22}^{(1)} \end{pmatrix}\end{aligned} \tag{A.3}$$

and

$$\begin{aligned}\boldsymbol{I}_J \otimes \mathbf{K}^{(2)} &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} k_{11}^{(2)} & k_{12}^{(2)} & k_{13}^{(2)} & k_{14}^{(2)} \\ k_{21}^{(2)} & k_{22}^{(2)} & k_{23}^{(2)} & k_{24}^{(2)} \\ k_{31}^{(2)} & k_{32}^{(2)} & k_{33}^{(2)} & k_{34}^{(2)} \\ k_{41}^{(2)} & k_{42}^{(2)} & k_{43}^{(2)} & k_{44}^{(2)} \end{pmatrix} \\ &= \begin{pmatrix} k_{11}^{(2)} & k_{12}^{(2)} & k_{13}^{(2)} & k_{14}^{(2)} & 0 & 0 & 0 & 0 \\ k_{21}^{(2)} & k_{22}^{(2)} & k_{23}^{(2)} & k_{24}^{(2)} & 0 & 0 & 0 & 0 \\ k_{31}^{(2)} & k_{32}^{(2)} & k_{33}^{(2)} & k_{34}^{(2)} & 0 & 0 & 0 & 0 \\ k_{41}^{(2)} & k_{42}^{(2)} & k_{43}^{(2)} & k_{44}^{(2)} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & k_{11}^{(2)} & k_{12}^{(2)} & k_{13}^{(2)} & k_{14}^{(2)} \\ 0 & 0 & 0 & 0 & k_{21}^{(2)} & k_{22}^{(2)} & k_{23}^{(2)} & k_{24}^{(2)} \\ 0 & 0 & 0 & 0 & k_{31}^{(2)} & k_{32}^{(2)} & k_{33}^{(2)} & k_{34}^{(2)} \\ 0 & 0 & 0 & 0 & k_{41}^{(2)} & k_{42}^{(2)} & k_{43}^{(2)} & k_{44}^{(2)} \end{pmatrix}\end{aligned} \tag{A.4}$$

The combined penalty matrix follows as the element-wise sum of (A.3) and (A.4):

$$\mathbf{K}^{(1)} \otimes \boldsymbol{I}_K + \boldsymbol{I}_J \otimes \mathbf{K}^{(2)} =$$

$$\begin{pmatrix}
k_{11}^{(1)} + k_{11}^{(2)} & k_{12}^{(2)} & k_{13}^{(2)} & k_{14}^{(2)} & k_{12}^{(1)} & 0 & 0 & 0 \\
k_{21}^{(2)} & k_{11}^{(1)} + k_{22}^{(2)} & k_{23}^{(2)} & k_{24}^{(2)} & 0 & k_{12}^{(1)} & 0 & 0 \\
k_{31}^{(2)} & k_{32}^{(2)} & k_{11}^{(1)} + k_{33}^{(2)} & k_{34}^{(2)} & 0 & 0 & k_{12}^{(1)} & 0 \\
k_{41}^{(2)} & k_{42}^{(2)} & k_{43}^{(2)} & k_{11}^{(1)} + k_{44}^{(2)} & 0 & 0 & 0 & k_{12}^{(1)} \\
k_{21}^{(1)} & 0 & 0 & 0 & k_{22}^{(1)} + k_{11}^{(2)} & k_{12}^{(2)} & k_{13}^{(2)} & k_{14}^{(2)} \\
0 & k_{21}^{(1)} & 0 & 0 & k_{21}^{(2)} & k_{22}^{(1)} + k_{22}^{(2)} & k_{23}^{(2)} & k_{24}^{(2)} \\
0 & 0 & k_{21}^{(1)} & 0 & k_{31}^{(2)} & k_{32}^{(2)} & k_{22}^{(1)} + k_{33}^{(2)} & k_{34}^{(2)} \\
0 & 0 & 0 & k_{21}^{(1)} & k_{41}^{(2)} & k_{42}^{(2)} & k_{43}^{(2)} & k_{22}^{(1)} + k_{44}^{(2)}
\end{pmatrix}$$

A.2. Linear Models with Gaussian Prior as Special Penalized Least Squares Models

Theorem A.1 *Let* $\mathbf{y}$ *be the* $(n \times 1)$ *response vector, and let* $\mathbf{X}$ *be a suitable design matrix. Let the linear model for* $\mathbf{y}$ *be defined as*

$$\mathbf{y} = \mathbf{X}\beta + \boldsymbol{\varepsilon},$$

with independent errors $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$*. Then, a Bayesian model with an arbitrary Gaussian prior on the parameters is equivalent to a penalized linear model with a special, quadratic penalty.*

Proof Let the assumptions of Theorem A.1 hold, and assume an arbitrary Gaussian prior on the parameters $\boldsymbol{\beta}$, i.e.,

$$\begin{aligned}
\mathbf{y}|\mathbf{X}, \boldsymbol{\beta} &\sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}) \\
\boldsymbol{\beta} &\sim \mathcal{N}(\mathbf{0}, \tau^2\boldsymbol{\Sigma}).
\end{aligned}$$

Thus it follows

$$\begin{aligned}
P(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) &\propto \exp\left(-\frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\right) \\
P(\boldsymbol{\beta}) &\propto \exp\left(-\frac{1}{2\tau^2}\boldsymbol{\beta}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}\right)
\end{aligned}$$

Using Bayes' theorem we get

$$\begin{aligned} P(\mathbf{y}|\mathbf{X}) &= P(\mathbf{y}|\mathbf{X},\boldsymbol{\beta}) \cdot P(\boldsymbol{\beta}) \\ &\propto \exp\left(-\frac{1}{2\sigma^2}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^\top(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}) - \frac{1}{2\tau^2}\boldsymbol{\beta}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}\right) \end{aligned}$$

Maximizing $P(\mathbf{y}|\mathbf{X})$ is equivalent to minimizing

$$-2\sigma^2\log(P(\mathbf{y}|\mathbf{X})) = (\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^\top(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}) + \frac{\sigma^2}{\tau^2}\boldsymbol{\beta}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}, \tag{A.5}$$

which is the penalized least squares estimate with quadratic penalty, penalty parameter σ^2/τ^2 and penalty matrix $\boldsymbol{\Sigma}^{-1}$.

□

B. Implementation Details

> And that's really the essence of programming. By the time you've sorted out a complicated idea into little steps that even a stupid machine can deal with, you've learned something about it yourself.
>
> *(Douglas Adams[1])*

With the work on this thesis lots of code for the R packages **mboost** (Hothorn *et al.* 2011a,b) and **gamboostLSS** (Hofner *et al.* 2011b) emerged. In **mboost**, most notably changes to the definition and computation of the degrees of freedom were implemented. The base-learners for radial basis functions, monotonic effects, and cyclic effects were also products of the research presented here. The package **gamboostLSS** was entirely created in this period. As it seems not sensible to show the manuals of evolving software in a static environment like this thesis, we try to highlight some of the features described above and show some code that was used in the analyses presented in this thesis. Further documentation can be found in the R help system of the packages, in the accompanying vignettes (package **mboost** only) or the corresponding papers (see, for example, Bühlmann and Hothorn 2007, Hothorn *et al.* 2010, Mayr *et al.* 2011b).

B.1. Overview of News and Changes in `mboost`

During the preparation of this thesis **mboost** was migrated from version 1.x to version 2.x, with many changes in the user-interface and even more changes in the back end. Lots of work was dedicated to these changes. A non-exhaustive overview of newly implemented code that was developed as part of this thesis and significant changes, which are visible to the user and resulted from the work on this thesis, is given in the following list:

[1] Adams D (1991), *Dirk Gently's Holistic Detective Agency*, Pocket Books, New York. Paperback edition. p. 25.

Base-learners

`bmono()` Newly added base-learner for monotonic, convex or concave functions of one or two variables. For theoretical details on the base-learner see Chapter 4.

`brad()` Newly added base-learner that implements radial basis functions. For details on the base-learner see Section 2.3.4.

`buser()` Newly added function that specifies a base-learner with user-specified design and penalty matrices, which then are used to minimize a (penalized) least squares criterion with quadratic penalty. This can be used to easily specify base-learners that are not (yet) implemented.

`bbs(x, cyclic = TRUE)` Newly added base-learner that computes uni- and bivariate cyclic P-splines. For theoretical details on the base-learner see Chapter 4.

`bols(x)` Newly added penalty for *ordinal* factors `x`. For details see Section 2.3.1.

`bols(x, intercept = FALSE)` If the intercept is removed in `bols`, it is checked if `x` is centered or a constant and a warning is issued if not.

`by = z` The argument `by` in `bols()` and `bbs()` is now able to handle factors `z` with more than two levels.

Null space All P-spline based functions now check that the specified degrees of freedom are greater than the range of the (unpenalized) null space. A warning is issued if the degrees of freedom are equal to the range and an error occurs if they are smaller. Furthermore, the default value for `df` is increased to 6 in `bspatial()`.

Changes to the Fitting Methods and Interface

`options(mboost_dftraceS = FALSE)` The newly derived degrees of freedom (see Section 3.1) were implemented. They can be used if `mboost_dftraceS` is set to `FALSE` using the global options function.

`df2lambda()` The function is used internally to compute degrees of freedom `df` for a given smoothing parameter `lambda` or vice versa. The new efficient computation of degrees of freedom (see Lemma 3.5) is now implemented in this function and is used automatically for all penalized base-learners. If `df2lambda()` is likely to become numerically instable (i.e., in the case of large entries in the design matrix) a warning is issued.

`glmboost(..., center = TRUE)` As a consequence of the finding that the centering of base-learners without intercept is of great importance (see Section 2.3.1), we changed the default to `center = TRUE`. Furthermore, columns of design matrices that correspond to contrasts are now also centered (if an intercept is specified in the model) leading to (much) faster risk minimization.

`cvrisk()` For parallel computations of the cross-validated risk, the interface of `cvrisk()` now also allows to use, for example, the packages **multicore** (Urbanek 2011) or **snow** (Tierney *et al.* 2011).

Various improvements in storage, speed, and stability were included into **mboost**. They make, for example, use of the package **Matrix** (Bates and Maechler 2011).

Convenience Functions

`extract()` The newly added generic function allows users to extract various characteristics of a single base-learner or a fitted model.

`coef.glmboost()` The intercept term is now adjusted for internally centered covariates (i.e., `center = TRUE`; see Section 2.3.1). Additionally, the new argument `off2int = TRUE` adds the offset to the intercept.

`plot.mboost` The functionality of the (experimental) `plot` function was improved. For example, the handling of the `which` argument is now better. In case of bivariate effects **lattice** plots are used and the resulting lattice object is returned. Furthermore, the handling of varying coefficients was improved.

`lines.mboost` The newly added (experimental) `lines` function allows to easily plot multiple partial effects into one single devise.

B.2. Different Implementation of `bols`

In this thesis an implementation of the linear base-learner (`bols`) different to the linear base-learner in the current version of **mboost** was used. They differ in the implementation of the option '`intercept = FALSE`' for categorical variables. In the 2.x series of **mboost**, base-learners are used where the design matrix of the categorical variable z with n_{cat} categories is build using

```
R> model.matrix(~ z - 1)
```

This leads to models where the explicit intercept (a column of ones) is removed but each category of z is specified by a separate mean value. The result is a design matrix $\mathbf{X}$ with n_{cat} columns again, where the element $X_{ij} = 1 \iff z_i = j$ and $X_{ij} = 0$ otherwise. This coding results from the above code irrespective of the specified contrasts. We call this 'mean coding'. Actually, we would prefer to have a base-learner where the coding is specified by the contrasts and the column of ones is completely dropped, i.e., the intercept is truly removed. This is achieved by using something similar to

```
R> X <- model.matrix(z)
R> X <- X[ , -1, drop = FALSE]
```

The alternative definition used in this thesis is given in the following code. The main differences reflect the change of code discussed above. They can be found in the function that creates the design matrix for linear base-learners `X_ols`. Changes in base-learner `bols` itself and other changes in `X_ols` are only required to call the correct functions[2] if the modified code is sourced after package **mboost** is loaded. The changes are highlighted in *italics*.

```
bols <-  function(..., by = NULL, index = NULL, intercept = TRUE, df = NULL,
                 lambda = 0, contrasts.arg = "contr.treatment") {

    if (!is.null(df)) lambda <- NULL

    cll <- match.call()
    cll[[1]] <- as.name("bols")

    mf <- list(...)
    if (length(mf) == 1 && (mboost:::isMATRIX(mf[[1]]) || is.data.frame(mf[[1]]))) {
        mf <- mf[[1]]
        ### spline bases should be matrices
        if (mboost:::isMATRIX(mf) && !is(mf, "Matrix"))
            class(mf) <- "matrix"
```

[2]We need to explicitly call non-exported functions from **mboost** by using the prefix '`mboost:::`'.

```
} else {
    mf <- as.data.frame(mf)
    cl <- as.list(match.call(expand.dots = FALSE))[2][[1]]
    colnames(mf) <- sapply(cl, function(x) as.character(x))
}
if(!intercept && !any(sapply(mf, is.factor)) &&
   !any(sapply(mf, function(x){uni <- unique(x);
                               length(uni[!is.na(uni)])}) == 1)){
    ## if no intercept is used and no covariate is a factor
    ## and if no intercept is specified (i.e. mf[[i]] is constant)
    if (any(sapply(mf, function(x)
                        abs(mean(x, na.rm=TRUE) / sd(x,na.rm=TRUE))) > 0.1))
        ## if covariate mean is not near zero
        warning("covariates should be (mean-) centered if ",
                sQuote("intercept = FALSE"))
}
vary <- ""
if (!is.null(by)){
    stopifnot(is.data.frame(mf))
    mf <- cbind(mf, by)
    colnames(mf)[ncol(mf)] <- vary <- deparse(substitute(by))
}

CC <- all(mboost:::Complete.cases(mf))
### option
DOINDEX <- is.data.frame(mf) &&
    (nrow(mf) > options("mboost_indexmin")[[1]] || is.factor(mf[[1]]))
if (is.null(index)) {
    ### try to remove duplicated observations or
    ### observations with missings
    if (!CC || DOINDEX) {
        index <- mboost:::get_index(mf)
        mf <- mf[index[[1]],,drop = FALSE]
        index <- index[[2]]
    }
}

ret <- list(model.frame = function()
            if (is.null(index)) return(mf) else return(mf[index,,drop = FALSE]),
            get_call = function(){
                cll <- deparse(cll, width.cutoff=500L)
                if (length(cll) > 1)
                    cll <- paste(cll, collapse="")
                cll
            },
            get_data = function() mf,
            get_index = function() index,
            get_names = function() colnames(mf),
            get_vary = function() vary,
            set_names = function(value) {
                if(length(value) != length(colnames(mf)))
                    stop(sQuote("value"), " must have same length as ",
                         sQuote("colnames(mf)"))
                for (i in 1:length(value)){
```

```
                        cll[[i+1]] <<- as.name(value[i])
                    }
                    attr(mf, "names") <<- value
                })
    class(ret) <- "blg"

    ret$dpp <- mboost:::bl_lin(ret, Xfun = X_ols, args = mboost:::hyper_ols(
                          df = df, lambda = lambda,
                          intercept = intercept, contrasts.arg = contrasts.arg))
    return(ret)
}


X_ols <-  function(mf, vary, args) {
    if (mboost:::isMATRIX(mf)) {
        X <- mf
        contr <- NULL
    } else {
        ### set up model matrix
        fm <- paste("~ ", paste(colnames(mf)[colnames(mf) != vary],
                    collapse = "+"), sep = "")
        ## removed: if (!args$intercept) fm <- paste(fm, "-1")
        fac <- sapply(mf[colnames(mf) != vary], is.factor)
        if (any(fac)){
            if (!is.list(args$contrasts.arg)){
                txt <- paste("list(", paste(colnames(mf)[colnames(mf) != vary][fac],
                                            "= args$contrasts.arg",
                                            collapse = ", "),")")
                args$contrasts.arg <- eval(parse(text=txt))
            }
        } else {
            args$contrasts.arg <- NULL
        }
        X <- model.matrix(as.formula(fm), data = mf,
                          contrasts.arg = args$contrasts.arg)
        contr <- attr(X, "contrasts")
        ## newly added:
        if (!args$intercept)
            X <- X[ , -1, drop = FALSE]
        if (vary != "") {
            by <- model.matrix(as.formula(paste("~", vary, collapse = "")),
                               data = mf)[ , -1, drop = FALSE] # drop intercept
            DM <- lapply(1:ncol(by), function(i) {
                ret <- X * by[, i]
                colnames(ret) <- paste(colnames(ret), colnames(by)[i], sep = ":")
                ret
            })
            if (is(X, "Matrix")) {
                X <- do.call("cBind", DM)
            } else {
                X <- do.call("cbind", DM)
            }
        }
    }
```

```
    ### set up penalty matrix
    ANOVA <- (!is.null(contr) && (length(contr) == 1)) && (ncol(mf) == 1)
    K <- diag(ncol(X))
    ### for ordered factors use difference penalty
    if (ANOVA && any(sapply(mf[, names(contr), drop = FALSE], is.ordered))) {
        K <- diff(diag(ncol(X) + 1), differences = 1)[, -1, drop = FALSE]
        if (vary != "" && ncol(by) > 1){       # build block diagonal penalty
            K <- kronecker(diag(ncol(by)), K)
        }
        K <- crossprod(K)
    }
    list(X = X, K = K)
}
```

B.3. Implementation Details for Chapter 3

In this section we present the code that was used to derive the model for forest health prediction (Section 3.3). To load the data set and pre-process the data, the following code was used

```
R> ## load data from website:
R> beeches <- read.delim(paste("http://www.stat.uni-muenchen.de/~kneib/",
+                              "regressionsbuch/download/buche.raw", sep =""),
+                        sep = " ", na.strings = ".")
R> ## remove covariate "frische"
R> beeches["frische"] <- NULL
R> ## rename covariates GER --> ENG
R> nms <- c("year", "x", "y", "inclination",
+           "elevation","soil", "fertilization", "age", "canopy",
+           "stand", "humus", "beeches", "id", "saturation", "ph")
R> names(beeches) <- nms
R> ## use complete cases only
R> beeches <- beeches[complete.cases(beeches), ]
R> ## add column for intercept
R> beeches$intercept <- rep(1, nrow(beeches))
R> ## create defoliation indicator
R> beeches$defoliation <- factor(as.numeric(beeches$beeches > 25))
R> ## reduce number of categories for humus
R> beeches$humus[beeches$humus > 4] <- 4
R> ## factors
R> factors <- c("fertilization", "stand", "id")
```

```
R> for (f in factors)
+      beeches[[f]] <- as.factor(beeches[[f]])
R> ## ordered factors
R> ordered_factors <- c("humus", "saturation")
R> for (f in ordered_factors)
+      beeches[[f]] <- as.ordered(beeches[[f]])
R> ## continuous covariates
R> continuous <- c("year", "x", "y", "inclination", "elevation", "soil",
+                  "age", "canopy", "ph")
R> ## center continuous covariates
R> indx <- names(beeches) %in% continuous
R> original_mean <- lapply(beeches[, indx], mean)
R> beeches[, indx] <- sapply(beeches[, indx], scale, center = TRUE, scale = FALSE)
```

To sample from the observation plots and not from the singe observations, the following code was used:

```
R> fold <- function(id, folds = 10, replace = FALSE){
+      n <- length(unique(id))
+      k <- folds
+      ntest <- floor(n / k)
+      if (replace == FALSE){
+          a <- sample(unique(id), ntest * (k-1), replace = FALSE)
+          res <- matrix(NA, ncol = k, nrow = length(id))
+          for (i in 1:(k-1)){
+              res[,i] <- as.numeric(!(id %in% a[(1:ntest) + ntest * (i-1)]))
+          }
+          res[,k] <- as.numeric((id %in% a))
+          stopifnot(all(rowSums(res) == (k-1)))
+      } else {
+          if (!all(sort(unique(id)) == 1:n))
+              stop(sQuote("id"), " is not a sequence of type 1:upper")
+          a <- rmultinom(folds, n, rep(1, n)/n)
+          res <- a[id,]
+      }
+      return(res)
+  }
```

Finally, the model can be fitted with the code below:

```
R> library("mboost")
R> ## make sure to use correct df2lambda:
R> options(mboost_dftraceS = FALSE)
R> ## set seed
R> set.seed(1907)
R> ## model formula
R> fm <- defoliation ~ bols(intercept, intercept = FALSE) +
+      bols(fertilization, intercept = FALSE, df = 1) +
+      bols(stand, intercept = FALSE, df = 1) +
+      bols(humus, intercept = FALSE, df = 1) +
+      bols(saturation, intercept = FALSE, df = 1) +
+      bols(ph, intercept = FALSE) +
+      bbs(ph, center = TRUE, df = 1, knots = 20) +
+      bols(canopy, intercept = FALSE) +
+      bbs(canopy, center = TRUE, df = 1, knots = 20) +
+      bols(soil, intercept = FALSE) +
+      bbs(soil, center = TRUE, df = 1, knots = 20) +
+      bols(inclination, intercept = FALSE) +
+      bbs(inclination, center = TRUE, df = 1, knots = 20) +
+      bols(elevation, intercept = FALSE) +
+      bbs(elevation, center = TRUE, df = 1, knots = 20) +
+      bols(year, intercept = FALSE) +
+      bbs(year, center = TRUE, df = 1, knots = 20) +
+      bols(age, intercept = FALSE) +
+      bbs(age, center = TRUE, df = 1, knots = 20) +
+      bspatial(x, y, center = TRUE, df = 1, differences = 1, knots = 12) +
+      brandom(id, df = 1)
R> ## fit model
R> forest <- gamboost(fm, data = beeches,
+                     family = Binomial(),
+            control = boost_control(trace = TRUE))
R> ## stratified bootstrap
R> cv_folds <- fold(beeches$id, replace = TRUE)
R> mstop_10 <- cvrisk(forest, folds = cv_folds, grid = 1:5000)
R> ## subset model to optimal number of boosting iterations
R> ## (i.e., continue until mstop is reached in this case)
R> forest[mstop(mstop_10)]
```

B.4. Implementation Details for Chapter 4

In this section we present the code that was used to derive the unconstrained and the monotonicity-constrained species distribution model for Red Kite (Section 4.5).

```
R> library("mboost")
R> set.seed(290875)
R> ## download dataset (in a zip file combined with other data sets)
R> if (!file.exists("bva.Rda")){
+     download.file("http://esapubs.org/archive/ecol/E092/161/redkite/bva.Rda",
+                   destfile = "bva.Rda", quiet = FALSE)
+ }
R> ## load data (Bavaria only)
R> load("bva.Rda")
R> ## make "Stadt" ordered
R> bva_year$Stadt <- ordered(bva_year$Stadt)
R> ## parameters
R> ctrl <- boost_control(mstop = 20, trace = TRUE)
R> nboot <- 25
R> nobs <- nrow(bva_year)/2
R> bs <- rmultinom(nboot, nobs, rep(1, nobs) / nobs)
R> bs <- apply(bs, 2, function(x) rep(x, rep(2, length(x))))
R> ## response and spatial component
R> spatial <- paste("bspatial(X, Y, knots = 6, df = 4.5)",
+                   "bspatial(X, Y, by = NNvar, knots = 6, df = 4.5)",
+                   sep = " + ")
R> vary <- "bspatial(X, Y, by = year, knots = 6, df = 4.5)"
R> responses <- response <- "Rotmilan"
R> ################################################################
R> # monotonic model
R> ################################################################
R> input_mono <- c("Nadelwald", "Stadt", "bio13", "bio16")
R> constr_mono <- c("decreasing", "decreasing", "decreasing", "decreasing")
R> idx <- 10:(length(bva_year))
R> input <- names(bva_year)[idx][ ! names(bva_year)[idx] %in% input_mono ]
R> fct <- sapply(input, function(i) is.factor(bva_year[[i]]))
R> ## set up formula for linear and smooth terms + spatial
R> rhs <- paste(c(paste("bbs(", input[!fct], ", df = 4.5)",
+                       sep = "", collapse = " + "),
+                 paste("bmono(", input_mono, ", constraint = \"",
+                       constr_mono,"\", df = 4.5)", sep = ""),
```

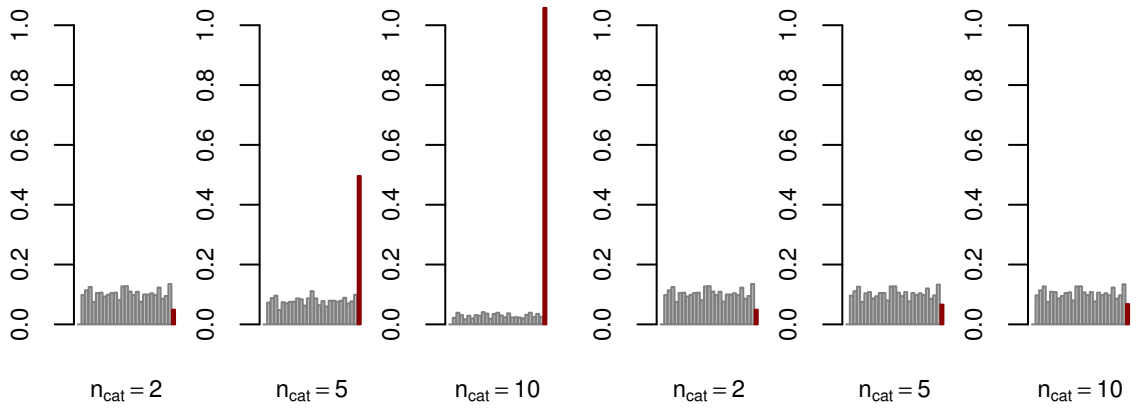
```
+                 paste("bols(", input[fct], ")", sep = "", collapse = " + ")),
+               collapse = " + ")
R> fm_mono <- as.formula(paste(response, " ~ ", rhs, "+", spatial, "+", vary,
+                              collapse = ""))
R> print(fm_mono)
R> ## Attention: this may take some time and requires larger amount of RAM
R> ## (especially cross-validation via cvrisk). Make sure package 'multicore' is
R> ## installed on multicore Unix systems. There might be problems on Windows with
R> ## parallelization: add 'papply = lapply' as argument to cvrisk() in this case.
R>
R> ## model fitting
R> m_mono <- mboost(fm_mono, data = bva_year, control = ctrl, family = Binomial())
R> ## determine optimal number of boosting iterations via bootstrap
R> gr <- seq(from = 10, to = 5000, by = 10)
R> grs <- seq(from = 10, to = 1000, by = 10)
R> cv_mono <- cvrisk(m_mono, fold = bs, grid = gr)
R> m_mono[mstop(cv_mono)]
R> save("cv_mono", "m_mono", "input_mono", "constr_mono",
+       file = "redkite_mono.Rda")
R> ############################################################
R> # unconstrained model
R> ############################################################
R>
R> ## For details see Hothorn et al., 2011
R>
R> fct2 <- sapply(input_mono, function(i) is.factor(bva_year[[i]]))
R> ## set up formula for linear and smooth terms + spatial
R> rhs <- paste(c(paste("bbs(", input[!fct], ", df = 4.5)", sep = ""),
+                 paste("bbs(", input_mono[!fct2], ", df = 4.5)", sep = ""),
+                 paste("bols(", input_mono[fct2], ")", sep = ""),
+                 paste("bols(", input[fct], ")", sep = "")),
+               collapse = " + ")
R> fm_vary <- as.formula(paste(response, " ~ ", rhs, "+", spatial, "+", vary,
+                              collapse = ""))
R> print(fm_vary)
R> ## model fitting
R> m_vary <- mboost(fm_vary, data = bva_year, control = ctrl, family = Binomial())
R> ## to save computation time we use mstop = 2920 (as determined in Hothorn
R> ## et. al, 2011); cross-validation with the same seed as in the previous
R> ## analyses leads to the same value of mstop.
R> m_vary[2920]
```

C. Additional Results

C.1. Additional Simulation Results for Ordinal Penalized Base-learners

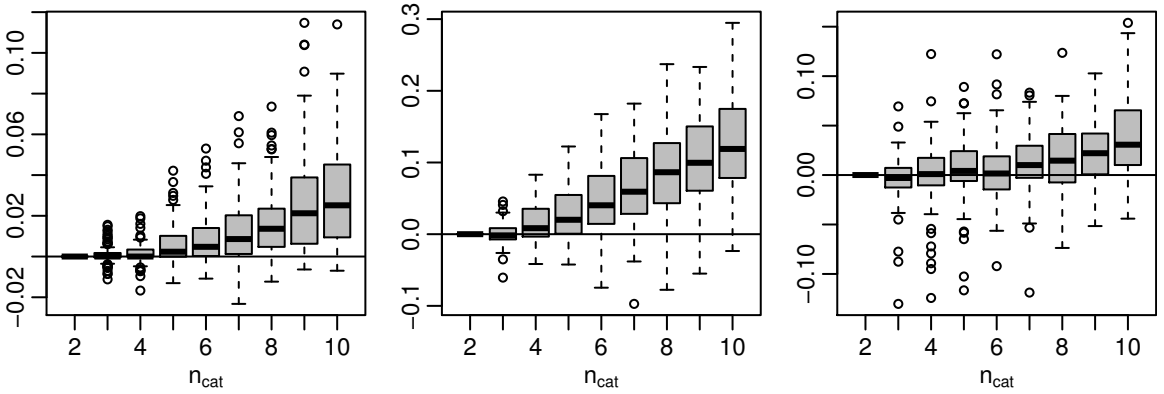
In this section, additional simulation results for ordinal penalized base-learners are presented. The main discussion can be found in Section 3.2.1. Figure C.1 shows the selection frequencies of the unpenalized base-learner together with the selection frequencies of ordinal penalized base-learners. A clear bias reduction can be observed in the latter case.

In the power cases (see Figure C.2), a clear reduction of the MSE can be found for ordinal penalized models (compared to unpenalized models). In contrast to ridge penalized ordinal variables, the improvement is not reduced by a strong overlapping shrinkage effect. It seems that shrinkage occurs but is less pronounced in the case of the ordinal penalty than in the ridge case.



(a) Unpenalized Model (b) Penalized Ordinal Model

Figure C.1.: *Null Model:* Average selection frequencies of base-learners for $n_{cat} = \{2, 5, 10\}$ in the "optimal step" $\widehat{m}_{stop,opt}$ without and with ordinal penalty. The last bar in each graph represents the selection frequency of the categorical covariate.



(a) Power Case 1 (b) Power Case 2: with z_2 (c) Power Case 2: without z_2

Figure C.2.: *Power Cases:* Boxplots represent $\text{MSE}_{\text{unpenalized}} - \text{MSE}_{\text{penalized}}$ in the first power case (left) and in the second power case, where the MSE is computed with (middle) and without (right) the influential, categorical covariate z_2.

C.2. Additional Results for Empirical Evaluation of Constrained Effect Estimates

In this section, additional results for the empirical evaluation of constrained effect estimates are given. In Tables C.1, and C.2, the number frequency of violations of monotonicity are given for smooth, monotonic functions and ordinal, monotonic functions. Table C.3 shows the average absolute difference at the boundaries for cyclic functions. For more details we refer to Section 4.3.

Table C.1.: *Smooth, monotonic effects:* Number of violations of monotonicity in monotonicity-constrained and unconstrained models; Mean values and corresponding standard errors (se) estimated from 100 simulation runs are given. A maximum of two violations is possible as two effects are estimated per model.

n	σ^2	monotonic	(se)	unconstrained	(se)
100	1.0	0	(0)	1.58	(0.0554)
	0.4	0	(0)	1.49	(0.0595)
	0.1	0	(0)	1.61	(0.0530)
200	1.0	0	(0)	1.43	(0.0655)
	0.4	0	(0)	1.47	(0.0611)
	0.1	0	(0)	1.63	(0.0525)
500	1.0	0	(0)	1.59	(0.0534)
	0.4	0	(0)	1.65	(0.0520)
	0.1	0	(0)	1.72	(0.0451)

Table C.2.: *Ordinal, monotonic effects:* Number of violations of monotonicity in monotonicity-constrained and unconstrained models; Mean values and corresponding standard errors (se) estimated from 100 simulation runs are given. A maximum of two violations is possible as two effects are estimated per model.

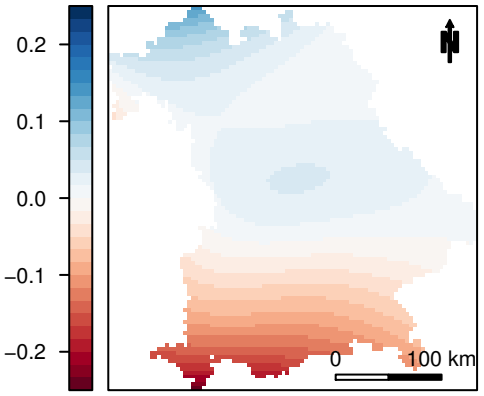
n	σ^2	monotonic	(se)	unconstrained	(se)
100	1.0	0.00	(0)	1.02	(0.0651)
	0.4	0.00	(0)	0.78	(0.0543)
	0.1	0.00	(0)	0.49	(0.0502)
200	1.0	0.00	(0)	0.83	(0.0493)
	0.4	0.00	(0)	0.70	(0.0461)
	0.1	0.00	(0)	0.37	(0.0485)
500	1.0	0.01	(0.01)	0.50	(0.0503)
	0.4	0.00	(0)	0.30	(0.0461)
	0.1	0.00	(0)	0.05	(0.0219)

Table C.3.: *Cyclic effects:* Absolute difference of the functions' boundaries $|\Delta\hat{f}|$ in models with cyclic constraint and in unconstrained models; Mean values and corresponding standard errors (se) estimated from 100 simulation runs are given.

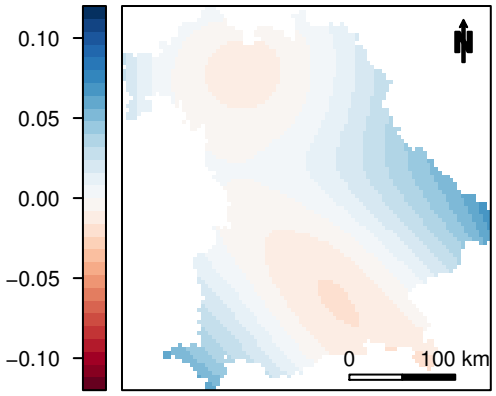
n	σ^2	cyclic	(se)	unconstrained	(se)
100	1.0	0	(0)	0.4935	(0.0332)
	0.4	0	(0)	0.4163	(0.0250)
	0.1	0	(0)	0.2724	(0.0161)
200	1.0	0	(0)	0.3661	(0.0233)
	0.4	0	(0)	0.2885	(0.0171)
	0.1	0	(0)	0.1966	(0.0112)
500	1.0	0	(0)	0.2487	(0.0142)
	0.4	0	(0)	0.2008	(0.0109)
	0.1	0	(0)	0.1483	(0.0075)

C.3. Additional Results for SDM for Red Kite

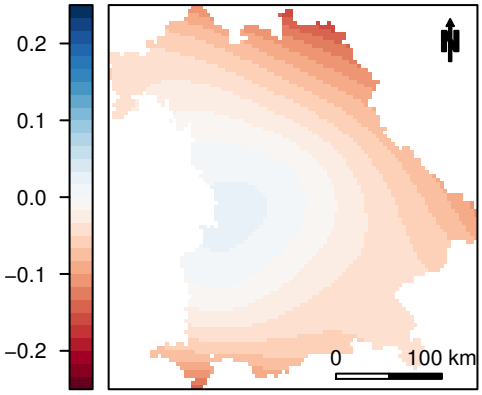
In this section, additional results for the Red Kite breeding distribution are presented. The main results can be found in Section 4.5.2. Figure C.3 shows the differences between the additive model '`add/vary`' and the monotonic model '`mono`'for the estimated spatial, spatial-temporal, and spatially varying effects. We can conclude that, compared to the range of the effect estimates, the differences between the monotonic model and the non-restricted model are negligible in all three cases.



(a) Spatial Effect



(b) Time-varying Effect



(c) Varying Effect for Altitude

Figure C.3.: Difference of estimates (`'mono' - 'add/vary'`) for the spatial effect (upper left), change of spatial effect over time-periods (upper right) and varying effect for altitude (lower). All differences are rather low compared to the effects, which range from -2.72 to 1.97 (spatial autocorrelation), -0.76 to 1.62 (change of spatial autocorrelation over time-periods) and -3.22 to 5.45 (spatially varying effect for altitude).

Bibliography

Akaike H (1974). "A new look at the statistical model identification." *IEEE Transactions on Automatic Control*, **19**, 716–723.

Anjyo K, Lewis J (2011). "RBF interpolation and Gaussian process regression through an RKHS formulation." *Journal of Mathematics for Industry*, **3**, 63–71.

Austin M (2002). "Spatial prediction of species distribution: An interface between ecological theory and statistical modelling." *Ecological Modelling*, **157**, 101–118.

Ayer M, Brunk HD, Ewing GM, Reid WT, Silverman E (1955). "An empirical distribution function for sampling with incomplete information." *Annals of Mathematical Statistics*, **26**, 641–647.

Bacchetti P (1989). "Additive isotonic models." *Journal of the American Statistical Association*, **84**, 289–294.

Barlow RE, Bartholomew DJ, Bremner JM, Brunk HD (1972). *Statistical inference under order restrictions: The theory and application of isotonic regression*. John Wiley & Sons, London.

Bates D, Maechler M (2011). *Matrix: Sparse and dense matrix classes and methods*. R package version 0.9996875-3, URL `http://CRAN.R-project.org/package=Matrix`.

Bezzel E, Geiersberger I, Lossow GV, Pfeifer R (2005). *Brutvögel in Bayern, Verbreitung 1996 bis 1999*. Eugen Ulmer Verlag, Stuttgart, Germany.

Binder H, Schumacher M (2008). "Allowing for mandatory covariates in boosting estimation of sparse high-dimensional survival models." *BMC Bioinformatics*, **9**, 14.

Bollaerts K, Eilers PHC, van Mechelen I (2006). "Simple and multiple P-splines regression with shape constraints." *British Journal of Mathematical and Statistical Psychology*, **59**, 451–469.

Breiman L (1998). "Arcing classifiers (with discussion)." *The Annals of Statistics*, **26**, 801–849.

Breiman L (1999). "Prediction games & arcing algorithms." *Neural Computation*, **11**, 1493–1517.

Breiman L (2001a). "Random forests." *Machine Learning*, **45**, 5–32.

Breiman L (2001b). "Statistical modeling: The two cultures (with discussion)." *Statistical Science*, **16**, 199–231.

Breiman L, Friedman JH, Olshen RA, Stone CJ (1984). *Classification and regression trees*. Wadsworth, California.

Brunk HD (1955). "Maximum likelihood estimates of monotone parameters." *Annals of Mathematical Statistics*, **26**, 607–616.

Bühlmann P, Hothorn T (2007). "Boosting algorithms: Regularization, prediction and model fitting." *Statistical Science*, **22**, 477–505.

Bühlmann P, Hothorn T (2010). "Twin boosting: Improved feature selection and prediction." *Statistics and Computing*, **20**, 119–138.

Bühlmann P, Yu B (2000). "Discussion: Additive logistic regression: A statistical view of boosting." *The Annals of Statistics*, **28**, 377–386.

Bühlmann P, Yu B (2002). "Analyzing bagging." *The Annals of Statistics*, **30**, 927–961.

Bühlmann P, Yu B (2003). "Boosting with the L_2 loss: Regression and classification." *Journal of the American Statistical Association*, **98**, 324–339.

Bühlmann P, Yu B (2006). "Sparse boosting." *Journal of Machine Learning Research*, **7**, 1001–1024.

Buja A, Hastie T, Tibshirani R (1989). "Linear smoothers and additive models (with discussion)." *The Annals of Statistics*, **17**, 453–555.

Chang YCI, Huang Y, Huang YP (2010). "Early stopping in L_2Boosting." *Computational Statistics and Data Analysis*, **54**, 2203–2213.

Conceição GMS, Miraglia SGEK, Kishi HS, Saldiva PHN, Singer JM (2001). "Air pollution and child mortality: A time-series study in São Paulo, Brazil." *Environmental Health Perspectives*, **109**, 347–350.

de Leeuw J, Hornik K, Mair P (2009a). *isotone: Active set and generalized PAVA for isotone optimization*. R package version 1.0-0, URL `http://CRAN.R-project.org/package=isotone`.

de Leeuw J, Hornik K, Mair P (2009b). "Isotone optimization in R: Pool-adjacent-violators algorithm (PAVA) and active set methods." *Journal of Statistical Software*, **32**, 5.

de los Campos G, Gianola D, Rosa GJM (2009). "Reproducing kernel Hilbert spaces regression: A general framework for genetic evaluation." *Journal of Animal Science*, **87**, 1883–1887.

Dette H, Neumeyer N, Pilz KF (2006). "A simple nonparametric estimator of a strictly monotone regression function." *Bernoulli*, **12**, 469–490.

Dette H, Scheder R (2006). "Strictly monotone and smooth nonparametric regression for two or more variables." *Canadian Journal of Statistics*, **34**, 535–562.

Efron B (1979). "Bootstrap methods: Another look at the jackknife." *The Annals of Statistics*, **7**, 1–26.

Efron B, Hastie T, Johnstone I, Tibshirani R (2004). "Least angle regression (with discussion)." *The Annals of Statistics*, **32**, 407–451.

Eilers PHC (2005). "Unimodal smoothing." *Journal of Chemometrics*, **19**, 317–328.

Eilers PHC, Gampe J, Marx BD, Rau R (2008). "Modulation models for seasonal time series and incidence tables." *Statistics in Medicine*, **27**, 3430–3441.

Eilers PHC, Marx BD (1996). "Flexible smoothing with B-splines and penalties (with discussion)." *Statistical Science*, **11**, 89–121.

Eilers PHC, Marx BD (2010). "Splines, knots, and penalties." *Wiley Interdisciplinary Reviews: Computational Statistics*, **2**, 637–653.

Elith J, Leathwick J (2009). "Species distribution models: Ecological explanation and prediction across space and time." *Annual Review of Ecology, Evolution, and Systematics*, **40**, 677–697.

Fahrmeir L, Kneib T, Lang S (2004). "Penalized structured additive regression: A Bayesian perspective." *Statistica Sinica*, **14**, 731–761.

Fahrmeir L, Kneib T, Lang S (2007). *Regression – Modelle, Methoden und Anwendungen*. Springer, Berlin.

Fang Z, Meinshausen N (2011). "LASSO ISOtone for high dimensional additive isotonic regression." *Journal of Computational and Graphical Statistics*. To appear, URL `http://arxiv.org/abs/1006.2940v1`.

Fenske N, Kneib T, Hothorn T (2011). "Identifying risk factors for severe childhood malnutrition by boosting additive quantile regression." *Journal of the American Statistical Association*. To appear, URL `http://epub.ub.uni-muenchen.de/10510/`.

Fok CCT, Ramsay JO (2006). "Fitting curves with periodic and nonperiodic trends and their interactions with intensive longitudinal data." In "Models for Intensive Longitudinal Data," pp. 109–123. Oxford University Press.

French JL, Kammann EE, Wand MP (2001). "Comment: Semiparametric nonlinear mixed-effects models and their applications." *Journal of the American Statistical Association*, **96**, 1285–1288.

Freund Y, Schapire R (1996). "Experiments with a new boosting algorithm." In "Proceedings of the Thirteenth International Conference on Machine Learning," pp. 148–156. Morgan Kaufmann Publishers Inc., San Francisco, CA.

Freund Y, Schapire RE (1997). "A decision-theoretic generalization of on-line learning and an application to boosting." *Journal of Computer and System Sciences*, **55**, 119–139.

Friedman J, Hastie T, Tibshirani R (2000). "Additive logistic regression: A statistical view of boosting (with discussion)." *The Annals of Statistics*, **28**, 337–407.

Friedman JH (1991). "Multivariate adaptive regression splines." *The Annals of Statistics*, **19**, 1–50.

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

Gentle JE (2007). *Matrix algebra: Theory, computations, and applications in statistics*. Springer, New York.

Gertheiss J, Tutz G (2009). "Penalized regression with ordinal predictors." *International Statistical Review*, **77**, 345–365.

Gianola D, van Kaam JBCHM (2008). "Reproducing kernel Hilbert spaces regression methods for genomic assisted prediction of quantitative traits." *Genetics*, **178**, 2289–2303.

González-Recio O, Gianola D, Long N, Weigel KA, Rosa GJM, Avendaño S (2008). "Nonparametric methods for incorporating genomic information into genetic evaluations: An application to mortality in broilers." *Genetics*, **178**, 2305–2313.

Gu C (2002). *Smoothing spline ANOVA models*. Springer, New York.

Hansen M, Yu B (2001). "Model selection and minimum description length principle." *Journal of the American Statistical Association*, **96**, 746–774.

Harvey A, Koopman SJ (1993). "Forecasting hourly electricity demand using time-varying splines." *Journal of the American Statistical Association*, **88**, 1228–1236.

Harvey A, Koopman SJ, Riani M (1997). "The modeling and seasonal adjustment of weekly observations." *Journal of Business & Economic Statistics*, **15**, 354–368.

Harville DA (2008). *Matrix algebra from a statistician's perspective*. Springer, New York.

Hastie T (2007). "Comment: Boosting algorithms: Regularization, prediction and model fitting." *Statistical Science*, **22**, 513–515.

Hastie T, Tibshirani R (1986). "Generalized additive models." *Statistical Science*, **1**, 297–310.

Hastie T, Tibshirani R (1990). *Generalized additive models*. Chapman & Hall / CRC, London.

Hastie T, Tibshirani R, Friedman J (2001). *The elements of statistical learning: Data mining, inference, and prediction*. Springer, New York.

Hazelton M, Turlach B (2011). "Semiparametric regression with shape constrained penalized splines." *Computational Statistics & Data Analysis*, **55**, 2871–2879.

He X, Ng P (1999). "COBS: qualitatively constrained smoothing via linear programming." *Computational Statistics*, **14**, 315–338.

He X, Shi P (1998). "Monotone B-spline smoothing." *Journal of the American Statistical Association*, **93**.

Heckman NE, Ramsay JO (2000). "Penalized regression with model-based penalties." *The Canadian Journal of Statistics*, **28**, 241–258.

Heller E (2009). *Bewegungsaktivität von Rehwild im Tages- und Jahresverlauf*. Bachelor's Thesis, Ludwig-Maximilians-Universität München. URL `http://epub.ub.uni-muenchen.de/11022/`.

Hoerl AE, Kennard RW (1970). "Ridge regression: Biased estimation for nonorthogonal problems." *Technometrics*, **12**, 55–67.

Hofner B, Hothorn T, Kneib T (2008). "Variable selection and model choice in structured survival models." *Technical report*, Department of Statistics, Ludwig-Maximilians-Universität München. URL `http://epub.ub.uni-muenchen.de/7901/`.

Hofner B, Hothorn T, Kneib T, Schmid M (2011a). "A framework for unbiased model selection based on boosting." *Journal of Computational and Graphical Statistics*, **20**, 956–971.

Hofner B, Mayr A, Fenske N, Schmid M (2011b). *gamboostLSS: Boosting methods for GAMLSS models*. R package version 1.0-3, URL `http://CRAN.R-project.org/package=gamboostLSS`.

Hofner B, Müller J, Hothorn T (2011c). "Monotonicity-constrained species distribution models." *Ecology*, **92**, 1895–1901.

Hothorn T (2010). "Discussion: Stability selection." *Journal of the Royal Statistical Society. Series B*, **72**, 463–464.

Hothorn T, Bühlmann P, Dudoit S, Molinaro A, van der Laan MJ (2006a). "Survival ensembles." *Biostatistics*, **7**, 355–373.

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

Hothorn T, Bühlmann P, Kneib T, Schmid M, Hofner B (2011a). *mboost: Model-based boosting*. R package version 2.0-12, URL `http://CRAN.R-project.org/package=mboost`.

Hothorn T, Bühlmann P, Kneib T, Schmid M, Hofner B (2011b). *mboost: Model-based boosting*. R package version 2.1-0, URL `http://R-forge.R-project.org/projects/mboost/`.

Hothorn T, Hornik K, Zeileis A (2006b). "Unbiased recursive partitioning: A conditional inference framework." *Journal of Computational and Graphical Statistics*, **15**, 651–674.

Hothorn T, Müller J, Schröder B, Kneib T, Brandl R (2011c). "Decomposing environmental, spatial, and spatiotemporal components of species distributions." *Ecological Monographs*, **81**, 329–347.

Hurvich C, Simonoff J, Tsai C (1998). "Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion." *Journal of the Royal Statistical Society. Series B*, **60**, 271–293.

Johnson ME, Moore LM, Ylvisaker D (1990). "Minimax and maximin distance designs." *Journal of Statistical Planning and Inference*, **26**, 131–148.

Jones RH, Brelsford WM (1967). "Time series with periodic structure." *Biometrika*, **54**, 403–408.

Kammann EE, Wand MP (2003). "Geoadditive models." *Applied Statistics*, **52**, 1–18.

Kim H, Loh WY (2003). "Classification trees with bivariate linear discriminant node models." *Journal of Computational and Graphical Statistics*, **12**, 512–530.

Kneib T, Fahrmeir L (2006). "Structured additive regression for categorical space-time data: A mixed model approach." *Biometrics*, **62**, 109–118.

Kneib T, Fahrmeir L (2010). "A space-time study on forest health." In RE Chandler, M Scott (eds.), "Statistical Methods for Trend Detection and Analysis in the Environmental Sciences," Wiley. To appear.

Kneib T, Hothorn T, Tutz G (2009). "Variable selection and model choice in geoadditive regression models." *Biometrics*, **65**, 626–634.

Koenker R, Ng P (2005a). "Frisch-Newton algorithm for sparse quantile regression." *Acta Mathematicae Applicatae Sinica, English Series*, **21**, 225–236.

Koenker R, Ng P (2005b). "Inequality constrained quantile regression." *Sankhya: The Indian Journal of Statistics (2003-)*, **67**, 418–440.

Leathwick JR, Elith J, Francis MP, Hastie T, Taylor P (2006). "Variation in demersal fish species richness in the oceans surrounding New Zealand: An analysis using boosted regression trees." *Marine Ecology Progress Series*, **321**, 267–281.

Leitenstorfer F, Tutz G (2007). "Generalized monotonic regression based on B-splines with an application to air pollution data." *Biostatistics*, **8**, 654–673.

Lewis PAW, Stevens JG (1991). "Nonlinear modeling of time series using multivariate adaptive regression splines (MARS)." *Journal of the American Statistical Association*, **86**, 864–877.

Loh WY (2002). "Regression trees with unbiased variable selection and interaction detection." *Statistica Sinica*, **12**, 361–386.

Loh WY, Vanichsetakul N (1988). "Tree-structured classification via generalized discriminant analysis (with discussion)." *Journal of the American Statistical Association*, **83**, 715–725.

Mayr A, Fenske N, Hofner B, Kneib T, Schmid M (2011a). "Boosting generalized additive models for location, scale and shape." In D Conesa, A Forte, A Lopez-Quilez, F Munoz (eds.), "Proceedings of the 26th International Workshop on Statistical Modelling," pp. 384–389. Valencia, Spain.

Mayr A, Fenske N, Hofner B, Kneib T, Schmid M (2011b). "GAMLSS for high-dimensional data - a flexible approach based on boosting." *Applied Statistics*. Accepted.

Mebs T, Schmidt D (2006). *Greifvögel Europas, Nordafrikas und Vorderasiens: Biologie. Bestandsverhältnisse. Bestandsgefährdung*. Franckh-Kosmos Verlags GmbH & Co., Stuttgart, Germany.

Mehr M, Brandl R, Hothorn T, Dziock F, Förster B, Müller J (2011). "Land use is more important than climate for species richness and composition of bat assemblages on a regional scale." *Mammalian Biology*, **76**, 451–460.

Meinshausen N, Bühlmann P (2010). "Stability selection (with discussion)." *Journal of the Royal Statistical Society. Series B*, **72**, 417–473.

Ng P, Maechler M (2007). "A fast and efficient implementation of qualitatively constrained quantile smoothing splines." *Statistical Modeling*, **7**, 315–328.

Nitsche G, Plachter H (1987). *Atlas der Brutvögel Bayerns 1979–1983*. Bayerisches Landesamt für Umweltschutz, München.

Nychka D (2000). "Spatial process estimates as smoothers." In MG Schimek (ed.), "Smoothing and Regression. Approaches, Computation and Application," pp. 393–424. Wiley, New York.

Nychka D, Saltzman N (1998). "Design of air-quality monitoring networks." In D Nychka, WW Piegorsch, LH Cox (eds.), "Case studies in environmental statistics," pp. 51–76. Springer, New York.

Park B, Lee Y, Ha S (2009). "L_2 boosting in kernel regression." *Bernoulli*, **15**, 599–613.

Pfeifer R, Müller J, Stadler J, Brandl R (2010). "Welchen Einfluß haben urbane Lebensräume auf die Artenvielfalt? Eine quantitative Analyse am Beispiel der Vogelwelt Bayerns." *Ornithologischer Anzeiger*, **48**, 126–142.

R Development Core Team (2011). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL `http://www.R-project.org`.

Ramsay JO (1988). "Monotone regression splines in action." *Statistical Science*, **3**, 425–441.

Ramsay JO, Dalzell CJ (1991). "Some tools for functional data analysis." *Journal of the Royal Statistical Society. Series B*, **53**, 539–572.

Reinsch CH (1967). "Smoothing by spline functions." *Numerische Mathematik*, **10**, 177–183.

Ridgeway G (1999). "The state of boosting." *Computing Science and Statistics*, **31**, 172–181.

Rigby RA, Stasinopoulos DM (2001). "The GAMLSS project: A flexible approach to statistical modelling." In B Klein, L Korsholm (eds.), "New Trends in Statistical Modelling: Proceedings of the 16th International Workshop on Statistical Modelling," pp. 249–256. Odense, Denmark.

Rigby RA, Stasinopoulos DM (2005). "Generalized additive models for location, scale and shape (with discussion)." *Applied Statistics*, **54**, 507–554.

Ripley B (2004). "Selecting amongst large classes of models." In N Adams, M Crowder, D Hand, D Stephens (eds.), "Methods and Models in Statistics," pp. 155–170. Imperial College Press, London.

Robinzonov N, Tutz G, Hothorn T (2011). "Boosting techniques for nonlinear time series models." *Advances in Statistical Analysis*. doi:10.1007/s10182-011-0163-4. To appear.

Ruppert D (2002). "Selecting the number of knots for penalized splines." *Journal of Computational and Graphical Statistics*, **11**, 735–757.

Ruppert D, Wand M, Carroll R (2003). *Semiparametric regression*. Cambridge University Press, Cambridge.

Saldiva P, Pope CI, Schwartz J, Dockery D, Lichtenfels A, Salge J, Barone I, Bohm G (1995). "Air pollution and mortality in elderly people: A time-series study in São Paulo, Brazil." *Archives of Environmental Health*, **50**, 159–164.

Schapire RE (1990). "The strength of weak learnability." *Machine Learning*, **5**, 197–227.

Scheder R (2009). *monoProc: Strictly monotone smoothing procedure*. R package version 1.0-6, URL `http://CRAN.R-project.org/package=monoProc`.

Schmid M, Hothorn T (2008a). "Boosting additive models using component-wise P-splines." *Computational Statistics & Data Analysis*, **53**, 298–311.

Schmid M, Hothorn T (2008b). "Flexible boosting of accelerated failure time models." *BMC Bioinformatics*, **9**, 269.

Schmid M, Hothorn T, Maloney K, Weller D, Potapov S (2010a). "Geoadditive regression modeling of stream biological condition." *Environmental and Ecological Statistics*, pp. 1–25. doi:10.1007/s10651-010-0158-4. To appear.

Schmid M, Potapov S, Pfahlberg A, Hothorn T (2010b). "Estimation and regularization techniques for regression models with multidimensional prediction functions." *Statistics and Computing*, **20**, 139–150.

Sobotka F, Kneib T (2010). "Geoadditive expectile regression." *Computational Statistics and Data Analysis*. doi:10.1016/j.csda.2010.11.015. To appear.

Spix C, Heinrich J, Dockery D, Schwartz J, Völksch G, Schwinkowski K, Cöllen C, Wichmann HE (1993). "Air pollution and daily mortality in Erfurt, East Germany, 1980-1989." *Environmental Health Perspectives*, **101**, 518–526.

Stache A, Hothorn T, Heurich M, Heller E (2010). "Activity patterns of European Roe Deer (Capreolus capreolus) in the Bavarian Forest National Park." Unpublished manuscript.

Steeb WH (1991). *Kronecker product of matrices and applications*. Bibliographisches Institut, Mannheim.

Stein ML (1999). *Interpolation of spatial data: Some theory for kriging*. Springer, New York.

Stone M (1974). "Cross-validatory choice and assessment of statistical predictions." *Journal of the Royal Statistical Society. Series B*, **36**, 111–147.

Strobl C, Boulesteix AL, Zeileis A, Hothorn T (2007). "Bias in random forest variable importance measures: Illustrations, sources and a solution." *BMC Bioinformatics*, **8**, 25.

Tierney L, Rossini AJ, Li N, Sevcikova H (2011). *snow: Simple network of workstations*. R package version 0.3-7, URL `http://CRAN.R-project.org/package=snow`.

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

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

Tutz G, Leitenstorfer F (2007). "Generalized smooth monotonic regression in additive modeling." *Journal of Computational and Graphical Statistics*, **16**, 165–188.

Urbanek S (2011). *multicore: Parallel processing of R code on machines with multiple cores or CPUs*. R package version 0.1-5, URL `http://CRAN.R-project.org/package=multicore`.

van Eeden C (1956). "Maximum likelihood estimation of ordered probabilities." *Koninkl. Nederl. Akademie van Wetenschappen, Proc. Ser. A.*, **59**, 444–455.

Vapnik V (1995). *The nature of statistical learning theory*. Springer, New York.

Wahba G (1990). *Spline models for observational data*. CBMS-NSF Regional Conference Series in Applied Mathematics 59. SIAM, Philadelphia.

Welham SJ, Cullis BR, Kenward MG, Thompson R (2006). "The analysis of longitudinal data using mixed model l-splines." *Biometrics*, **62**, 392–401.

Wood SN (2003). "Thin plate regression splines." *Journal of the Royal Statistical Society. Series B*, **65**, 95–114.

Wood SN (2006a). *Generalized additive models: An introduction with R*. Chapman & Hall / CRC, London.

Wood SN (2006b). "Low-rank scale-invariant tensor product smooths for generalized additive mixed models." *Biometrics*, **62**, 1025–1036.

Wood SN (2008). "Fast stable direct fitting and smoothness selection for generalized additive models." *Journal of the Royal Statistical Society. Series B*, **70**, 495–518.

Wood SN (2010). *mgcv: GAMs with GCV/AIC/REML smoothness estimation and GAMMs by PQL*. R package version 1.7-2, URL `http://CRAN.R-project.org/package=mgcv`.

Zhang D, Lin X, Sowers M (2000). "Semiparametric regression for periodic longitudinal hormone data from multiple menstrual cycles." *Biometrics*, **56**, 31–39.

Zhang D, Lin X, Sowers M (2007). "Two-stage functional mixed models for evaluating the effect of longitudinal covariate profiles on a scalar outcome." *Biometrics*, **63**, 351–362.