Estimating and evaluating mixed and semiparametric models with statistical and deep learning methods

Dissertation zur Erlangung des Doktorgrades
der Wirtschaftswissenschaftlichen Fakultät
der Georg-August-Universität Göttingen
im Promotionsprogramm
'Applied Statistics and Empirical Methods'

vorgelegt von
René-Marcel Kruse
geboren in Oldenburg

Göttingen, 2023

Prüfungskommission:

Erstgutachter:	Prof. Dr. Thomas Kneib Professur für Statistik Wirtschaftswissenschaftliche Fakultät Georg-August-Universität Göttingen
Zweitgutachter:	Prof. Dr. Benjamin Säfken Arbeitsgruppe Angewandte Statistik und Data Science Institut für Mathematik Technische Universität Clausthal
Drittgutachterin:	Prof. Dr. Elisabeth Bergherr Professur für Raumbezogene Datenanalyse und Statistische Lernverfahren Wirtschaftswissenschaftliche Fakultät Georg-August-Universität Göttingen

Acknowledgments

Working on this thesis was a rollercoaster ride of emotions. It could be stressful and frustrating at times, but also incredibly rewarding and enlightening. I am grateful for the support of the community at the chair, both old and new friends, as well as my family. I especially would like to express my thanks to:

... Thomas Kneib, whose approachability, openness, and calmness created a unique working environment. I sincerely appreciate the invaluable help, insightful discussions, and support for all my research ideas.

... Benjamin Säfken, for supporting my work, answering naive questions, and above all helping me to shape my chaotic plans into a coherent thesis.

... Elisabeth Bergherr, who was first my instructor in the introductory statistics course in my bachelor's and now, appropriately, taking the role as a third supervisor.

... Rouven, my friend with whom I shared successes and failures, but most importantly, a love for obscure music and black humour, and who could remind me that there is (sometimes) more to life than research.

... Aisouda, for being an incredible friend who has always been there for me. I can't express enough how much her constant support means to me. I am incredibly grateful.

... Anton, for initializing the third research project, a fruitful and easy collaboration, as well as his acceptance of my rather unorthodox working hours.

... Sina, Alexander and Chris for our games nights and our unfortunately unsuccessful attempt to defeat the pandemic in the pandemic.

... Markus, Lennart, Gianmarco, Katharina, Maximilian, all my colleagues, friends and everyone I forget, who made life in Göttingen so much easier and who, most importantly, stoically endured my constant bad jokes and puns.

And most of all, I want to thank my parents. Dagmar and Gerold, without whom I would not be the person that I am. Thank you for your unconditional support and trust in me.

Danke für alles.

Abstract

Semiparametric models are well-established, versatile and effective statistical models for analysing complex data by combining both parametric and non-parametric components and are used in a wide range of applications and fields. Meanwhile, deep learning models are a type of machine learning model designed to identify and represent intricate patterns and relationships in data, which are gaining popularity due to their impressive performance in various applications, including analysing structured tabular data and, in contrast to statistical models, unstructured data such as images, text, and sound recordings. However, the complexity of these models can make them challenging to understand and interpret, limiting their transparency and interpretability.

The purpose of this thesis is twofold. Firstly, develop more efficient model selection and evaluation methods that can handle model uncertainty for semiparametric and deep learning models. Secondly, to merge traditional statistical methods with machine and deep learning concepts, combining their strengths to mitigate their weaknesses.

Based on Stein's unbiased risk estimate, part one of this thesis introduces a criterion for determining squared loss optimal weights for model averaging of (conditional) linear mixed models. Furthermore, the complicated underlying optimisation of the presented criterion is discussed, and a possible solution via a specifically customised algorithm based on the augmented Lagrangian is introduced.

An essential part of model evaluation and selection in statistics is model complexity, often measured in degrees of freedom. In the second part of this thesis, three different methods for measuring model complexity based on the concept of covariance penalties associated with degrees of freedom are presented and ultimately compared and analysed using different simulations.

The third and final section of the thesis introduces a new type of neural additive models for location, scale, and shape by combining the GAMLSS distribution regression framework with neural network techniques and principles. This approach differs from previous deep learning methods as it can model the entire response distribution rather than just the mean response. The effectiveness of this method is evaluated using simulated and real data and compared against established statistical and deep learning methodologies.

Zusammenfassung

Semiparametrische Modelle sind bewährte, vielseitige und effiziente statistische Modelle für die Analyse komplexer Daten, die parametrische und nichtparametrische Komponenten kombinieren und in einer Vielzahl von Anwendungen und Bereichen eingesetzt werden. Deep Learning Modelle sind eine Art Machine Learning-Modelle, die darauf ausgelegt sind, komplexe Muster und Beziehungen in Daten zu erkennen und darzustellen. Sie erfreuen sich aufgrund ihrer beeindruckenden Leistungsfähigkeit in verschiedenen Anwendungen immer größerer Beliebtheit, z.B. bei der Analyse von strukturierten Tabellendaten oder, im Gegensatz zu statistischen Modellen, von unstrukturierten Daten wie Bildern, Text und Ton. Die Komplexität dieser Modelle führt jedoch dazu, dass sie schwer zu verstehen und nachzuvollziehen sind, was ihre Transparenz und Interpretierbarkeit einschränkt.

In dieser Arbeit werden zwei Ziele verfolgt. Erstens sollen effizientere Methoden zur Modellauswahl und -bewertung entwickelt werden, die mit Modellunsicherheiten sowohl bei statistischen als auch bei Deep Learning Modellen umgehen können. Zweitens sollen traditionelle statistische Methoden mit Konzepten des Machine Lernens und des Deep Learning zusammengeführt werden, um die jeweiligen Schwächen eines Ansatzes durch die Stärken des anderen zu kompensierten.

Basierend auf Stein's Unbiased Risk Estimate wird im ersten Teil dieser Arbeit ein Kriterium zur Bestimmung der optimalen Gewichte für das Model Averaging von (konditionalen) linearen gemischten Modellen eingeführt. Darüber hinaus wird die komplizierte zugrundeliegende Optimierung des vorgestellten Kriteriums diskutiert und eine mögliche Lösung durch einen speziell angepassten Algorithmus auf der Basis des augmented Lagrangian vorgestellt.

Ein wesentlicher Bestandteil der Modellauswahl- und bewertung in der Statistik ist die Modellkomplexität, die oft in Freiheitsgraden gemessen wird. Im zweiten Teil dieser Arbeit werden drei verschiedene Methoden zur Messung der Modellkomplexität auf der Grundlage des Konzepts der mit Freiheitsgraden verbundenen Covariance Penalties vorgestellt und schließlich anhand verschiedener Simulationen verglichen und analysiert.

Im dritten und letzten Teil der Arbeit wird die neue Klasse der Neural Additive Models for Location, Scale and Shape vorgestellt, dafür wird der GAMLSS-Modellansatz mit Techniken und Prinzipien neuronaler Netze kombiniert. Dieser Ansatz unterscheidet sich von anderen Deep Learning-Methoden, da er die gesamte Response-Verteilung und nicht nur die mean Prediction modellieren kann. Die Wirksamkeit dieser Methode wird anhand von simulierten und realen Daten bewertet und mit etablierten statistischen und Deep Learning-Methoden verglichen.

Contents

Chapter 1: Introduction

The following sections will introduce different statistical and mathematical concepts that have played an essential role in my thesis research. Section 1.1 introduces regression models, focusing on linear mixed models in Section 1.1.1 and distributional regression models in Section 1.1.3, as these play a central role in Part I and III. Section 1.2 introduces the fundamental ideas of deep learning, while Section 1.3 deals with the concept of model complexity, its meaning and how to measure it. Where the complexity of deep learning models constitutes the central point of focus of the research presented in Part II of this thesis, Part III, on the other hand, builds upon the combination of distributional regression approaches, namely generalised additive models for location, scale and shape, and artificial neural networks.

1.1 Regression models

Regression models aim to measure the influence of *covariates* $x_1, \ldots, x_k$, independent explanatory variables, on the *response* or *dependent* variable y with all its realisations $\boldsymbol{y} = (y_1, \ldots, y_n)' \in \mathbb{R}^n$. In the *classical linear regression* model the relationship between an individual y_i and the corresponding covariates is assumed to be linear plus some noise, the *error* term ε_i, such that the systematic component of the model can be represented as a linear combination of the underlying covariates as

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik} + \varepsilon_i = \boldsymbol{x}_i' \boldsymbol{\beta} + \varepsilon_i,$$

for each observation $i = 1, \ldots, n$, where $\boldsymbol{x}_i = (1, x_{i1}, \ldots, x_{ik})'$ and the unknown parameters $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_k)' \in \mathbb{R}^{k+1}$, the *regression coefficients*, determine the influence of the corresponding covariates on the response. Such that changing the value of x_1 by one (unit) leads to a corresponding increase of β_1 in the outcome variable y. This phenomenon is valid under the condition that all other covariates that affect y remain unchanged.

The underlying covariates are, however, usually summarized in the *desgin matrix* $\boldsymbol{X}$, and together with the vector of responses $\boldsymbol{y}$, coefficients $\boldsymbol{\beta}$ and errors $\boldsymbol{\varepsilon}$ resulting in the notation of the classical model as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}.$$

In order to estimate these unknown parameters we collect observation pairs $(y_1, \boldsymbol{x}_1), \ldots, (y_n, \boldsymbol{x}_n)$, and introduce the following assumptions for the error term $\boldsymbol{\varepsilon}$:

1. The errors have mean or expectation zero, i.e. $\mathbb{E}(\boldsymbol{\varepsilon}) = \mathbf{0}$.
2. The errors are homoscedastic, i.e. all follow the same constant finite variance $0 < \sigma^2 < \infty$.
3. The errors are uncorrelated, i.e. $Cov(\varepsilon_i, \varepsilon_j) = 0, \forall i \neq j$.

These assumptions also play a role in the *Gauss-Markov* theorem, that determines the estimator with the lowest sampling variance within the class of all possible linear unbiased estimators. In addition to the assumptions about the error terms, we introduce further that:

4. The assumed design matrix $\boldsymbol{X}$ has a full column $rank(\boldsymbol{X}) = k + 1 = p$, such that $\boldsymbol{X'X}$ is nonsingular and thus an invertible matrix.

And in order to obtain the *classical normal regression* model the follow has to hold as well:

5. Identically independent normally distributed errors such that $\boldsymbol{\varepsilon} \sim \mathcal{N}_n(\mathbf{0}, \sigma^2 \boldsymbol{I}_n)$.

However, the normality assumption here serves mainly to construct confidence intervals and hypothesis tests for the regression coefficients and it is not mandatory for the linear model to work. Following all assumptions for the classical normal regression model we obtain for the underlying response that $\mathbb{E}(\boldsymbol{y}) = \boldsymbol{X}\boldsymbol{\beta}$ and $Cov(\boldsymbol{y}) = \sigma^2 \boldsymbol{I}$ such that $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{X}\boldsymbol{\beta}, \sigma^2 \boldsymbol{I})$.

The unknown regression coefficients $\boldsymbol{\beta}$ can be estimated using the least squares method. This minimises the expected squared error with quadratic L_2-norm as follows

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{\boldsymbol{p}}}{\arg\min}\, \mathbb{E} \left\| \mathbf{y} - \mathbf{X}\boldsymbol{\beta} \right\|^2,$$

solving this, we get the least squares estimator $\hat{\boldsymbol{\beta}} = (\boldsymbol{X'X})^{-1}\boldsymbol{X'y}$. The least squares estimator also corresponds with the solution obtained by a *maximum-likelihood* estimation, a method that finds the maximiser of the underlying log-likelihood of the model under normality assumption. Many of the strict assumptions made here are relaxed in the following chapters to derive other, more flexible models. For more information, see Faraway (2006) and Fahrmeir (2013).

1.1.1 Linear mixed models

The linear model can be extended to include *random effects*, creating the class of *linear mixed models*. The new effect on the expectation equation assumes that the random variables follow a zero mean distribution. Interestingly, these effects have different roles and interpretations depending on the task or user. They allow, for example, the incorporation of dependent observations in the data by including of an implied correlation in the model structure (Fisher, 1919). Another way to understanding random effects is to consider them as a regularising effect, as they can collect the influences of comparable observations to improve model estimation (Bates et al., 2015).

The resulting linear mixed model equation is as follows

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

where $\boldsymbol{y}$ represents the vector of n observed responses $\boldsymbol{y} = (y_1, \ldots, y_n'), \boldsymbol{X} \in \mathbb{R}^{n \times p}$ and $\boldsymbol{Z} \in \mathbb{R}^{n \times q}$ represent the full column rank design matrices, with the vector of fixed effects $\boldsymbol{\beta} \in \mathbb{R}^p$ and the random effects $\boldsymbol{b} \in \mathbb{R}^q$. The vector $\boldsymbol{\varepsilon}$ represents the unobserved random errors. Both $\boldsymbol{b}$ and $\boldsymbol{\varepsilon}$ are assumed to be independent and follow a multivariate Gaussian distribution such that

$$\begin{pmatrix} \boldsymbol{b} \\ \boldsymbol{\varepsilon} \end{pmatrix} \sim \mathcal{N}\left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \boldsymbol{D} & 0 \\ 0 & \boldsymbol{\Sigma} \end{pmatrix} \right\},$$

where $\boldsymbol{D}$ is a block-diagonal, positive, semidefinite variance-covariance matrix depending on a covariance parameter vector, however, the normality assumption is not mandatory and is only introduced for convenience, allowing for likelihood-based procedures to estimate unknown parameters in $\boldsymbol{D}$ and of the residual variance (Laird and Ware, 1982).

For linear mixed models, there are two different interpretations depending on the understanding and application of the random effects. The *marginal* form treats the random effects as an additional part of the already random error term $\boldsymbol{\varepsilon}$, resulting in the marginal distribution $y \sim \mathcal{N}_n(\boldsymbol{X}\beta, \boldsymbol{V} := \boldsymbol{Z}\boldsymbol{D}\boldsymbol{Z}' + \boldsymbol{\Sigma})$(Fahrmeir, 2013). The *conditional* form, on the other hand, approaches the random effects differently by treating them as penalised coefficients resulting in the form of the conditional distribution $y \sim \mathcal{N}_n(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{b}, \boldsymbol{\Sigma})$ (Säfken et al., 2021).

Fixed as well as random effects can be estimated respectively predicted via

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^T\boldsymbol{V}^{-1}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{V}^{-1}\boldsymbol{y},$$
$$\hat{\boldsymbol{b}} = \boldsymbol{D}\boldsymbol{Z}^T\boldsymbol{V}^{-1}(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}),$$

where the resulting estimator of the fixed effects $\hat{\boldsymbol{\beta}}$ is the best linear unbiased estimator and is also the maximum-likelihood estimator of $\boldsymbol{\beta}$ and the predictor of the random effects $\hat{\boldsymbol{b}}$ is also the best linear unbiased predictor of $\boldsymbol{b}$ (Harville, 1976). In the case that $\boldsymbol{D}$ or $\boldsymbol{\Sigma}$ contain unknown parameters, denoted as $\boldsymbol{\theta}$, the estimator of the unknown parameters $\hat{\boldsymbol{\theta}}$ is given by the maximiser of the profile log-likelihood and is thus up to a constant equal to

$$\ell_P(\boldsymbol{\theta}) = -\frac{1}{2}\left[\log|\boldsymbol{V}_{\boldsymbol{\theta}}| + (\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^T\boldsymbol{V}_{\boldsymbol{\theta}}{}^{-1}(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})\right].$$

Instead of estimating $\boldsymbol{\theta}$ via profile log-likelihood, it is also possible to determine $\boldsymbol{\theta}$ via the marginal or restricted log-likelihood. However, this estimator underestimates the components (Fahrmeir, 2013). A possible alternative is a REML based estimator of $\hat{\boldsymbol{\theta}}$, which is less biased towards zero. For a more detailed overview of linear mixed models see Faraway (2006), Fahrmeir (2013) and Wood et al. (2017).

1.1.2 Additive models

An additional extension of the classical linear model is the inclusion of nonparametric terms in the linear predictor in the so-called class of *additive models*. With this extension, the equation of the underlying model is changed to the following form

$$y = \boldsymbol{X}\boldsymbol{\beta} + \sum_{j=1}^{J} f_j(\boldsymbol{z}_j) + \boldsymbol{\varepsilon}, \text{ with } \boldsymbol{\varepsilon} \sim \mathcal{N}_n(\boldsymbol{0}, \boldsymbol{\Sigma}),$$

where $f_j(\boldsymbol{z}_j)$ represents smooth functions of the underlying covariates in $\boldsymbol{z}_j \in \mathbb{R}^n$ with $j = 1, \ldots, J$ infinite dimensional smooth effects, which can be approximated and thus represented by a finite number of *basis functions*. The underlying concept of basis functions is to approximate an infinite-dimensional function via a linear combination of coefficients $\gamma_1, \ldots, \gamma_K$ and a finite number of basis functions $B_1, B_2, \ldots, B_K$, obtaining the function $f \approx \sum_{k=1}^{K} B_k\gamma_k$. The implied basis function for the covariate $\boldsymbol{z} = (z_1, \ldots, z_n)$ is evaluated at each of the observed points z_i,, as a result of this obtaining the matrix $\boldsymbol{B}$, where the k-th basis function evaluation at the i-th

observation z_i corresponds to the k-th column and row of $\boldsymbol{B}$. The B-spline basis is one of the most widely used basis representations (Schoenberg, 1946). For this approach, the domain of the covariates is divided by the so-called knots $\kappa_1, \ldots, \kappa_d$. While the assumed smooth function $f(\boldsymbol{z})$ is represented via $K = q + d - 1$ B-spline basis functions of degree q. These are $q + 1$ piecewise continuously differentiable connected polynomial functions of degree q. The underlying knots can be defined in different ways via an equidistant grid or, for example, via quantiles of the underlying covariates in $\boldsymbol{z}$. A B-spline of degree zero is represented by an indicator function using the knots as

$$B_k^0(z_i) = I(\kappa_k \leq z_i \leq \kappa_{k+1}).$$

Higher order B-splines can be defined recursively as

$$B_k^q(z_i) = \frac{z_i - \kappa_{k-1}}{\kappa_k - \kappa_{k-1}} B_{k-1}^{q-1}(z_i) + \frac{\kappa_{k+1} - z_i}{\kappa_{k+1}\kappa_{k+1-q}} B_k^{q-1}(z_i).$$

For sufficiently high degree q, the resulting splines are continuous and differentiable; therefore, they are efficient to evaluate and thus provide directly available (higher-order) derivatives, in addition to other mathematically and numerically desirable properties (Eilers and Marx, 1996). Using this definition of B-splines to determine basis functions requires $2l$ outer knots outside the original domain. Together with the inner knots, resulting in a total number of knots of $\kappa_{1-l}, \kappa_{1-l+1}, \ldots, \kappa_{m+l-1}, \kappa_{m+l}$.

Since the number and, in particular, the position of the knots affect the fit of the functions, their choice can directly contribute to the over- or underfitting of the function. In order to avoid the problem of selecting an appropriate number of knots, Eilers and Marx (1996) have introduced a penalty term to the B-Spline to enforce overall smoothness, creating the so-called *P-Splines*. P-Splines allow a flexible definition of f, while preventing overly rough estimates through a penalty term by estimating coefficients for a generous number of B-Spline basis functions with an appropriate penalty. Giving a new form of the criterion for estimating the underlying regression coefficients as

$$||f(\boldsymbol{z}) - \boldsymbol{B}\boldsymbol{\gamma}||^2 + \lambda \boldsymbol{\gamma}' \boldsymbol{P} \boldsymbol{\gamma} \rightarrow \min_{\boldsymbol{\gamma}},$$

where $\boldsymbol{P} \in \mathbb{R}^{K \times K}$ represents the penalty matrix, and the variable λ is the *smoothing parameter* that determines the smoothness of the underlying function estimator $\hat{f}$. This term is based on a relatively large number of knots, allowing the smoothness to be approximated by the difference of the

adjacent B-splines. For the determination of an optimal smoothing parameter λ, several approaches have been successful. Optimisation based on (generalised) cross-validation (GCV) or the Akaike information criterion (AIC) can be employed. An alternative approach uses the relationship between the restricted maximum likelihood (REML) estimator and the penalised least criterion. For more detailed illustrations and discussion of additive models, see Fahrmeir (2013) and Wood et al. (2017).

1.1.3 Distributional regression

Additive models are limited by their narrow focus on mean prediction and the implicit assumption of constant variance. An extension and further development of the additive models is the model class of *generalised additive models for location, scale and shape* or short GAMLSS by Rigby and Stasinopoulos (2005).

GAMLSS relaxes the assumptions about the distribution of the response variable and replaces them with a general distribution family. The underlying systematic part of the GAMLSS model is extended to not only addressing one arbitrary parameter, but all parameters of the conditional distribution of the response variable. The model assumes that the responses are conditionally independent given the respective covariates and the response to exhibit a parametric density $f(y_i|\boldsymbol{\theta}_i)$. The assumed conditional density can depend on up to K different distribution parameters $\boldsymbol{\theta}_i = (\theta_{i1}, \ldots, \theta_{iK})'$. The distributional parameters are each modelled via a separate additive predictor η_{θ_k}, depending additively on the underlying covariates. An essential component of the GAMLSS model is the monotonic link function $g_k(\cdot)$, which allows each parameter of the distribution vector to be conditional on different sets of covariates. Similarly to (generalized) additive models, the variable is modelled using additive nonparametric functions of the covariates, resulting in the GAMLSS to be defined by the set of equations as

$$g_k(\theta_k) = \beta_{0\theta_k} + \sum_{j=1}^{J_k} f_{j\theta_k}(x_{jk}) = \eta_{\theta_k},$$

where $k = 1, \ldots, K$ represents the kth parameter, $\beta_{0\theta_k}$ the intercept of the respective submodel, and $j = 1, \ldots, J_k$ denotes the corresponding covariate. Due to the parametric assumption, it is possible to employ likelihood-based approaches for estimation. Depending on the implementation, GAMLSS models are numerically estimated either using a back-fitting algorithm with Newton-Raphson steps (Rigby and Stasinopoulos, 2005), using trust-region algorithms (Marra and Racine, 2022), or using a functional gradient descent boosting approach (Mayr et al., 2010).

1.2 Deep learning

Deep learning is a type of machine learning that involves training *artificial neural networks* with many layers to learn increasingly abstract and hierarchical representations of data. Multiple layers allow deep neural networks to automatically discover more complex and nuanced features from data without the need for explicit feature engineering. One of the critical challenges in deep learning is avoiding overfitting, where the neural network becomes too specialized to the training data and performs poorly on new data. Various techniques have been developed to address this challenge, including dropout regularization, weight decay, and early stopping (Goodfellow et al., 2016). Deep learning has been successfully applied to various tasks, including image and speech recognition, natural language processing, and autonomous driving. For example, *convolutional neural networks* (CNNs) have achieved state-of-the-art performance on image classification and object detection tasks (Krizhevsky et al., 2012; He et al., 2016), while *recurrent neural networks* (RNNs) have been used for sequence modelling and language processing (Sutskever et al., 2014; Cho et al., 2014). Deep learning has also been used to generate novel content, such as images (Radford et al., 2015), music (Huang et al., 2018), and text (Brown et al., 2020). One of the most popular deep learning architectures for natural language processing is the transformer model, which has achieved state-of-the-art performance on various language tasks, including language translation and generation (Vaswani et al., 2017). Overall, deep learning has revolutionized the field of artificial intelligence and has led to significant advances in a wide range of domains. Ongoing research is focused on developing even more powerful and efficient deep learning models and understanding their underlying mechanisms and limitations.

1.2.1 The fundamentals of deep learning

Deep learning models are a very broad and complex class of models that can take different forms and functions depending on their architecture and assumptions. However, the mathematical and statistical theory on which all the models are based can be generalised.

To motivate and properly establish this, the following example is provided. Consider a problem where the data is of the form $\mathcal{Z} := \mathcal{X} \times Y$ with a joint probability. Moreover, observed data points are given as $\boldsymbol{d} = (z^{(i)})_{i=1}^m = ((x_i, y_i))_{i=1}^m \in \mathcal{Z}^m$. This data in turn comes from an assumed true, but

unknown, data-generating process. In this context, this process is given as

$$\boldsymbol{y} = \phi(\boldsymbol{x}) + \boldsymbol{\epsilon},$$

where $\phi(\cdot) : \mathbb{R} \to \mathbb{R}$, $\boldsymbol{y} \in \mathcal{Y} \subseteq \mathbb{R}$, $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^m$ and $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ for all $i = 1, \ldots, n$.

The task in deep learning is to find a model that performs measurably well on the given known *training data* $\boldsymbol{d}$ and performs well on previously unknown *test data*. This performance is measured by the respective assumed loss function $\mathcal{L}$ or given error measurement.

To find such a well performing model, assume that $\mathcal{Z}, \mathcal{X}$, and $\mathcal{Y}$ are known and measurable. Further, for the assumed loss function $\mathcal{L}$ overall measurable functions $\mathcal{Y}, \mathcal{X}$ holds $\mathcal{L} : \mathcal{M}(\mathcal{X}, \mathcal{Y}) \times \mathcal{Z} \to \mathbb{R}$. The goal is now to create a hypothesis set $\mathcal{F} \subset \mathcal{M}(\mathcal{X}, \mathcal{Y})$ to create a *learning* algorithm of the following form

$$\mathcal{A} : \bigcup_{m \in \mathbb{N}} \mathcal{Z}^m \to \mathcal{F},$$

which uses the given data to find a model $m_d = \mathcal{A}(\boldsymbol{d}) \in \mathcal{F}$, exhibiting strong performance on unknown data. The assumed hypothesis set here consists of all realisations of a given neural network given assumed architecture and parameter set.

For a more detailed mathematical description, see Berner et al. (2021) and Roberts et al. (2022).

1.2.2 Artificial neural networks

Deep learning models use neural networks as function approximators, where a series of operations are repeated one after the other in so-called layers to achieve better results through repeated learning (Roberts et al., 2022).

The most fundamental component of the neural network is the *neurons*, with each layer of a model consisting of several neurons in tandem (Goodfellow et al., 2016). Each of these neurons performs two different computational steps. In the first step, the so-called net input of the neurons z_i is calculated, which is essentially a linear aggregation of the incoming inputs $\boldsymbol{x}_j$, where all inputs are weighted and summed, and finally a bias b_i is added as given by

$$z_i(x) = b_i + \sum_{j=1}^{n} \boldsymbol{W}_{ij} \boldsymbol{x}_j, \quad \text{for} \quad i = 1, \ldots, m,$$

where m neurons each form a layer that takes an n-dimensional input vector, and $\boldsymbol{W}_{ij}$ represents the underlying matrix of weights. This stage also demonstrates the link between statistical models

and neural networks, as both approaches can be oversimplified as a summation over weighted inputs (covariates) and a bias (intercept). The resulting value is then passed to the activation function σ, which transforms the net input as $\sigma(z_i)$. Activation functions can take various forms, from simple linear to complex non-linear functions (Chollet, 2018). See Figure 1.1 for a graphical representation.

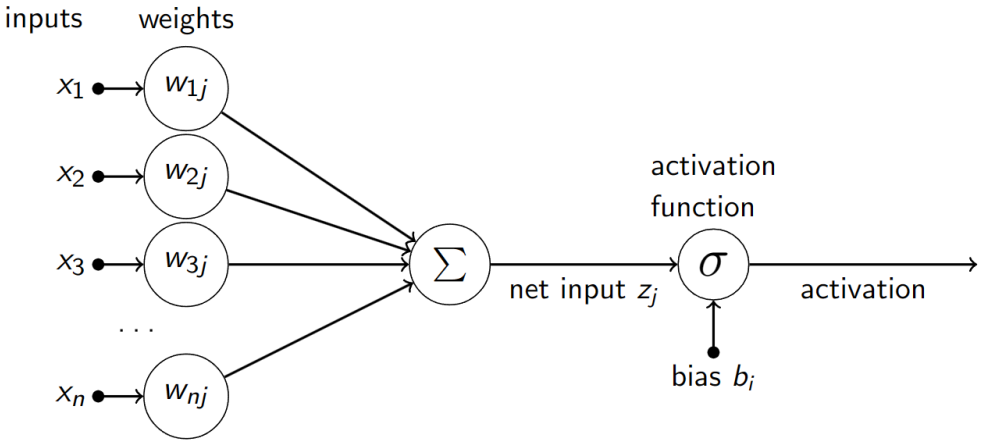

Figure 1.1: Stylised representation of a neuron in an artificial neural network.

In deep learning models, many of these neurons are combined into a layer, whereas many layers are, in turn, joined together in such a fashion that the output of one layer becomes the input of the next. This structure allows for increasingly flexible function sets to be created as the number of neurons, but also as the number of layers increases (Roberts et al., 2022). The number of neurons, layers and the chosen activation function create the so-called *network architecture*. Over time, various of configurations and unique forms of model architectures have emerged (Goodfellow et al., 2016; Berner et al., 2021). The concatenation of several layers, with all outputs of one layer as inputs of all neurons in the next layer, creates a so-called *multilayer perceptron* (MLP). MLPs are, in many ways, the simplest of the deep learning architectures and, therefore, a useful minimal model to illustrate deep learning (Roberts et al., 2022).

An MLP can be defined recursively in the first layer as

$$z_i^{(1)}(x_\alpha) \equiv b_i^{(1)} + \sum_{j=1}^{n_0} W_{ij}^{(1)} x_{j\alpha},$$

for $i = 1, \ldots, n_1$ and all subsequent layers as

$$z_i^{(\ell+1)}(x_\alpha) \equiv b_i^{(\ell+1)} + \sum_{j=1}^{n_\ell} W_{ij}^{(\ell+1)} \sigma\left(z_j^{\ell}(x_\alpha)\right),$$

for $i = 1, \ldots, n_{\ell+1}$ and $\ell = 1, \ldots, L-1$. Every layer ℓ of all layers L assumed here comprises of a total of n_ℓ neurons. The layers between the input and the output are also known as the *hidden layers*. An exemplary MLP is shown in Figure 1.2.

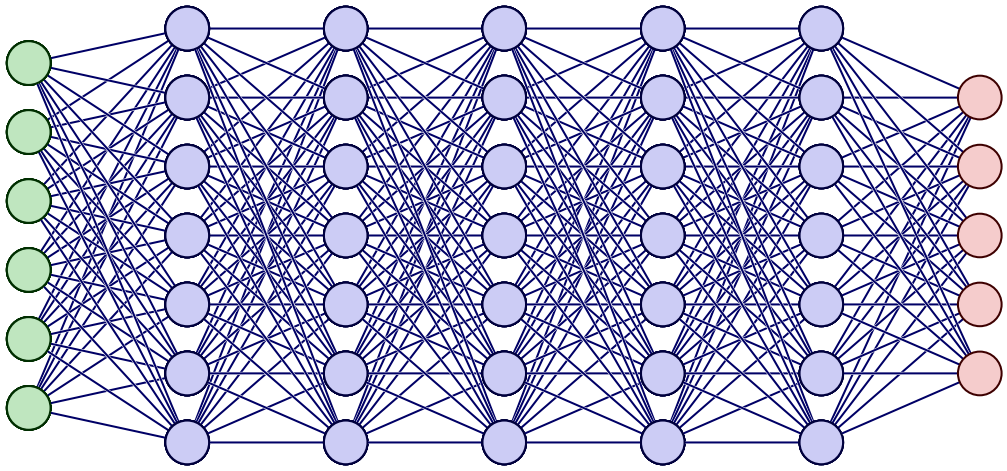

Figure 1.2: Representation of a deep (fully-connected) multilayer perceptron with five hidden layers, each with seven neurons. Green represents the input nodes, blue the hidden nodes and red the output nodes.

The number of neurons in layer ℓ is also known as the *width* of the layer, which is defined as $n_{\ell=1,\ldots,L-1}$, where the number of neurons in the first and last layer is fixed at n_0 and n_L to the dimension of the model input and the dimension of the model output, respectively. The total number of neurons and associated parameters of the MLP is given by $\sum_{\ell=1}^{L} (n_\ell + n_\ell n_{\ell-1})$ (Roberts et al., 2022).

A large number of model parameters of a deep learning model thus served as an inspiration for the subject of model complexity of deep learning models discussed in Part II.

1.2.3 Hybrid statistical deep learning models

Deep learning models are often criticised for their lack of transparency and the incomprehensibility of their predictions and calculations to humans (Buhrmester et al., 2019). Therefore, the decision-making process of a deep learning model is often criticised as a black box, where the user knows the inputs and the resulting outputs but cannot understand what happened in between (Savage, 2022). One area that is therefore becoming more and more of a focus is *explainable machine learning*. Researchers aim to develop tools and frameworks for machine learning models to make the underlying decision-making process more comprehensible for humans. One such approach is to

combine statistical models and their inherent explanatory strength with deep learning methods and their strong predictive power to create a new class of hybrid models with strong predictive properties and increased explanatory capacity. Approaches of this type include *deep regression* by Rügamer et al. (2021). This semi-structured deep distributional regression framework attempts to learn conditional distributions by combining additive regression models and deep networks. This approach relies on orthogonalizing the different model components to allow an interpretable combination of different subnetworks.

Another approach is *neural additive models* (NAMs), which combine some of the expressiveness of *deep neural networks* (DNNs) with the inherent intelligibility of generalised additive models from statistics. NAMs learn a linear combination of neural networks, each dedicated to a single input feature.

These networks are trained jointly and can learn arbitrarily complex relationships between their input and the output feature and can be applied to a wide range of tasks, including regression, classification, and time series forecasting. The basic structure of a NAM consists of a sum of several components, each of which represents a different aspect of the relationship between the response and predictor variables. These can be linear, non-linear or combined and can be learned using a variety of optimising algorithms.

The general form of a NAM can be written as

$$\mathbb{E}(y) = h\left(\beta + \sum_{j=1}^{J} f_j(x_j)\right),$$

with $h(\cdot)$ the activation function used in the output layer, $x \in \mathbb{R}^j$ the input features, β the global intercept term, and $f_j : \mathbb{R} \to \mathbb{R}$ representing the multi-layer perceptrons corresponding to the j-th feature. As the two frameworks differ mainly how which individual characteristics are modelled, the similarity with GAMs is obvious. The role of the $h(\cdot)$ activation function in the NAM is akin to that of the link function of GAMs (Wood et al., 2017). Part III will present an extension of the NAM approach by incorporating a more comprehensive distributional focused approach akin to the GAMLSS framework of Rigby and Stasinopoulos (2005).

1.3 Model evaluation, choice and the role of complexity

When applying statistical or machine learning models, a key question is selecting the model that provides the best insights into scientific questions or has the greatest predictive power among all considered candidate models. Furthermore, a scientifically sound and replicable model selection is necessary to draw reliable and reproducible conclusions and is a cornerstone of the scientific method. Therefore, it is unsurprising that this topic has long been the focus of research, and many different approaches have emerged from fields as diverse as statistics, signal processing, and information theory (Ding et al., 2018).

1.3.1 Model choice and selection

Model selection remains perhaps the most popular approach to dealing with the uncertainty of finding a fitting data analysis approach. Based on a given data set, model selection is the task of selecting a statistical, machine or deep learning model from a set of possible candidates. Several selection methods have been developed and proposed for this purpose.

Information criterion

One of the most widely used methods for model evaluation and selection are *information criteria*. Information criteria are generally based on likelihood functions and are mostly related to parametric model selection problems. One of the most famous and widely used is the Akaike information criterion (Akaike, 1973), which in itself is an estimator of the relative *Kullback-Leibler distance* (KLD) (Kullback and Leibler, 1951).

The Kullback-Leibler distance stems from the field of information theory, and its purpose is to measure the distance between two probability distributions, however, in the context of model selection, it is instead to be understood as a measure of the distance between a true but unknown data generating process given by the density $g(\boldsymbol{y})$ producing the observed values of $\boldsymbol{y}$ and a parametric model $f(\boldsymbol{y}|\boldsymbol{\vartheta})$; whose parameters are estimated in practice via maximum likelihood estimator $\hat{\boldsymbol{\vartheta}}(\boldsymbol{y})$ based on the observed $\boldsymbol{y}$ (Greven and Kneib, 2010). The overall aim is to minimise the expected relative Kullback-Leibler distance in order to obtain a good approximating model. It can be shown that the minimisation of the Kullback-Leibler distance is equivalent to the minimisation of the *Akaike information* (AI) of

$$AI = -2\mathbb{E}_{\boldsymbol{y}}\mathbb{E}_{\boldsymbol{z}} \log f(\boldsymbol{z}|\hat{\boldsymbol{\vartheta}}(\boldsymbol{y})),$$

where $\boldsymbol{z}$ represents a new, previously unknown realisation of g, however, due to its sole dependence on $\boldsymbol{y}$, the maximising log-likelihood in the estimation process for $\hat{\boldsymbol{\vartheta}}(\boldsymbol{y})$ assumed here is biased. Akaike was able to determine a, under certain regularity assumptions, bias correction which is asymptotically equal to twice the value of the dimension of $\boldsymbol{\vartheta}$, such that the bias corrected version of the Akaike information becomes the Akaike information criterion of the following form

$$AIC = -2 \log f(\boldsymbol{y}|\hat{\boldsymbol{\vartheta}}(\boldsymbol{y})) + 2k,$$

where k is equal to the dimension of $\boldsymbol{\vartheta}$, corresponding to the respective (effective) degrees of freedom and covariance penalties of the model under consideration (Efron, 2004). The more complex the model of interest, the larger the correction term. Consequently, when used as an approximation to a true underlying data-generating process, the resulting information-theoretic interpretation of AIC is as an estimator of the relative information lost by the model.

For the derivation of the AIC, some assumptions have to be made, which depending on the model approach, are no longer considered reasonable and the AIC has to be adjusted. As an example, for the linear mixed models considered in Part I, the AIC must be adjusted for the differing model assumptions, depending on whether they are marginal or conditional, as presented in Section 1.1.1; for a detailed presentation of possible adjustments, see Greven and Kneib (2010).

Cross-validation

Information criteria are not the only approach to evaluation and model selection. Other approaches have emerged as reliable alternatives, including *cross-validation* (CV) methods (Allen, 1974; Geisser, 1975). Unlike approaches such as AIC, CV-based methods relax the assumption that the models under consideration must be parametric, requiring only that the underlying data be permutable and that the predictive quality of the model is sufficiently measurable, however, intriguingly, it is possible to establish a link between AIC and cross-validation and other model selection techniques (Efron, 2004).

There are different approaches to cross-validation, but the general procedure can be stated as follows. First, the underlying data is divided into training and validation data. Each model under consideration is then fitted to the training data or, in the case of machine or deep learning models, trained and then validated based on the validation data. To minimise the underlying variability, this training and validation process is repeated several times, each time with a different split of the

underlying data. Subsequently, the average validation loss of the considered models is compared and the model with the lowest average loss is selected. The final step is to fit or respectively train the selected model on the whole data set once more (Burman et al., 1994).

A unique form of the CV method plays a role in Part II, the *k-fold cross-validation*, randomly dividing the data into k subsets of approximately equal size. Then, the models are trained on $k-1$ of the folds and validated on the omitted one until each has been used once for validation. See Hastie et al. (2009) for a more detailed procedure discussion.

1.3.2 Model averaging

In contrast to the classic model selection, *model averaging* deals with model uncertainty by averaging over the set of candidate models rather than having the user select a model according to a selection criterion. To motivate model averaging, assume K classical linear candidate models, as presented in Section 1.1.

The respective assumed models are given as

$$\boldsymbol{y}_k = \boldsymbol{X}_k\boldsymbol{\beta}_k + \boldsymbol{\varepsilon}.$$

Let in the following the estimated values for $\boldsymbol{y}$ be given by $\hat{\boldsymbol{y}}_k = \boldsymbol{X}_k\hat{\boldsymbol{\beta}}_k = \hat{\boldsymbol{H}}_k\boldsymbol{y}$, where $\hat{\boldsymbol{H}}$ is the corresponding hat matrix of the fitted model. Classical model selection would now select the one model that would be considered preferable by one or more model selection criteria like AIC or other suitable techniques. Model averaging, however, computes a weighted average over all candidate models.

For this, assume a weight vector of the form $\boldsymbol{w} = (w_1, \ldots, w_K)'$. This vector must satisfy some assumptions to be a weight vector, which are defined as follows

1. The sum of all weights must equal one, i.e. $\sum_{k=1}^{K} w_k = 1$.
2. Each weight must be equal or less than one, i.e. $w_k \geq 1$.
3. Any individual weight must be greater than or equal to zero, i.e. $0 \leq w_k$.

In summary, this implies that the weights must be non-negative and belong to the set $\mathcal{W} = \{w \in [0,1]^K : \sum_{k=1}^{K} w_k = 1\}$. Given such a vector, the model averaging estimator can be defined as follows as

$$\hat{\boldsymbol{y}} = \sum_{k=1}^{K} w_k\hat{\boldsymbol{y}}_k = \sum_{k=1}^{K} w_k\hat{\boldsymbol{H}}_k\boldsymbol{y}_k.$$

The presented model averaging approach depends significantly on the weights chosen for the estimation process (Hansen and Racine, 2012). Weights are commonly selected according to different information criteria, such as AIC. One of the most popular such approaches is based on a suggestion by Buckland et al. (1997) as

$$w_k = \frac{\exp\left(-IC_k/2\right)}{\sum_{i=1}^{K} \exp(-IC_i/2)},$$

where w_k denotes the corresponding weight of model k, with IC as the value of an arbitrarily assumed information criterion for the considered model k of all K candidate models. Similar to the different approaches to model selection, there are alternatives to weighting schemes based on information criteria. Akin to model selection, an application of cross-validation proposed by Hansen and Racine (2012) has emerged as one of the main alternatives for determining weight vectors. The authors' *jackknife model averaging* approach uses leave-one-out cross-validation to minimise a least squares weighting criterion. This approach is particularly suitable for general formulations of linear models, as it allows for the error to violate the homoscedasticity assumption and still leads to correct weights within the framework of the approach.

For a more in-depth overview of the topic of model averaging, see Claeskens and Hjort (2008), Wang et al. (2009) and Burnham and Anderson (2011).

1.3.3 Model complexity

The complexity of a statistical model is often understood as the ability of the approach to fit the data. For many users and in the literature, the term *degrees of freedom* is synonymous with model complexity and is used to measure or parameterise the inherent bias-variance trade-off in model selection (Janson et al., 2015).

The number of dimensions in which a random vector may vary is the original meaning of degrees of freedom in statistics and is essential for many different methods from model selection to prediction error estimation. For simple linear regression, where we have an n-dimensional response vector $\boldsymbol{y}$ and an $n \times p$ design matrix $\boldsymbol{X}$ with full column rank, the fitted values $\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}$ are an orthogonal projection of the vector $\boldsymbol{y}$ onto the p-dimensional column space of $\boldsymbol{X}$. The residuals of the model $\hat{\boldsymbol{\varepsilon}} = \boldsymbol{y} - \hat{\boldsymbol{y}}$ assumed here are thus the projection onto the orthogonal complement of the dimension $n - p$. Put simply, this means that for linear regression, the degrees of freedom are equal to the dimension p.

This definition of degrees of freedom is also called *model degrees of freedom* and is used to investigate model complexity; another form of degrees of freedom are the *residual degrees of freedom*, which are calculated as $n-p$.[1] Degrees of freedom also quantify the optimism of the residual sum of squares of an out-of-sample estimator. For example, in a simple linear regression, the residual sum of squares understates the mean squared prediction error by $2\sigma^2 p$ on average (Janson et al., 2015). Thus, it is possible to construct an unbiased estimator of the prediction error out of the residual sum of squares and a bias correction term based on the underlying model degrees of freedom as proposed by Mallows (1973). Therefore, the underlying degrees of freedom can also be regarded as a kind of penalty term for how closely the model fits the data (Mallows, 1973). However, this definition of degrees of freedom is less useful once beyond the scope of simple linear regression. For smoothing spline models or models with an inherent penalty, such as ridge models, the number of free parameters, is a poor measure of complexity. Consequently, several proposals for generalising the concept of degrees of freedom, extend the concept to other model classes and frameworks. One such proposal is that of Efron (1986), who formulated the degrees of freedom in the context of his optimism theorem.

Efron's approach can be summarised as follows. For $i = 1, \ldots, n$ let the mean μ_i be non-random and further let $y_i = \mu_i + \varepsilon_i$ hold. Assume that ε_i is zero in the mean and has a finite variance. In addition, let $\hat{y}_i$ be the estimator of μ_i and let y_i^* be new, unknown, independent observations from the same data generating process as y_i and identically distributed. Then the following will hold

$$2\sum_{i=1}^{n} Cov(y_i, \hat{y}_i) = \mathbb{E}\left[\sum_{i=1}^{n}(y_i^* - \hat{y}_i)^2\right] - \mathbb{E}\left[\sum_{i=1}^{n}(y_i - \hat{y}_i)^2\right].$$

According to Efron, for iid residuals with finite variance the degrees of freedom can thus be defined as follows

$$df = \frac{1}{\sigma^2}\text{trace}\left(Cov(\boldsymbol{y}, \hat{\boldsymbol{y}})\right) = \frac{1}{\sigma^2}\sum_{i=1}^{n} Cov\left(\boldsymbol{y}, \hat{\boldsymbol{y}}\right).$$

This definition of degrees of freedom[2] and their relationship to the so-called covariance penalties is essential in the complexity measures for neural networks considered in Part III.

[1] In the remainder of this thesis, the term 'degrees of freedom' will always refer to the concept of model degrees of freedom.

[2] For linear regression models with well-defined hat matrix this will reduce to $df = \text{trace}(\boldsymbol{H})$.

Chapter 2: Outline and contributions

This thesis deals with the theoretical and technical problems of combining statistical and deep learning models for better estimation and questions of expanding on existing model evaluation methods in statistics and deep learning. Each goal poses different research questions and challenges, which are addressed in three parts of this dissertation.

The **first** research question this thesis is concerned with is:

Can the application of conditional degrees of freedom, as initially presented by Greven and Kneib (2010) and utilized in the study by Zhang et al. (2014), be used to establish a model weighting mechanism that can effectively create an asymptotically optimal model averaging method for conditional linear mixed models?

A weight-finding criterion for asymptotically optimal weights for model averaging of conditional linear mixed models is derived in Part I, based on the work of Greven and Kneib (2010) and Zhang et al. (2014). Furthermore, the unique nonlinear optimisation problem under equality and inequality constraints of the underlying weight finding procedure is illustrated. In order to solve this inherently complex problem, an optimiser based on the augmented Lagrangian Nocedal and Wright (2006), specifically adapted to the problem set at hand, is developed and presented in the text. The corresponding Part I of the dissertation is based on the elaborations of the paper:

> **Kruse**, Silbersdorff, and Säfken (2022). "Model averaging for linear mixed models via augmented Lagrangian." *Computational Statistics & Data Analysis 167: 107351.*

Methods and algorithms developed in this paper are used in the published `R`-package:

> Säfken, Rügamer, Baumann, and **Kruse** (2021). "R-Package 'cAIC4'". *https://cran.r-project.org/package=cAIC4.*

The **second** research question that is discussed is:

Can concepts for the evaluation and measurement of statistical models be transferred to models from the field of deep learning? Moreover, how does it relate to methods that attempt to measure model complexity, such as degrees of freedom and covariance penalties?

In Part II, different methods for measuring the model complexity of deep learning models are presented. For artificial neural networks, model complexity cannot be measured directly, but a direct link can be shown to covariance penalties as presented by Efron (1983, 2004). Methods for estimating these penalties are derived in the forms of direct approximation of underlying derivates, in the form of a bootstrap based approach, and in the form of a k-fold cross-validation method. The introduced methods are presented and discussed in the context of model complexity of deep learning models. Furthermore, the properties and results of these methods are investigated and analysed in simulation studies. The corresponding Part II is based on the paper:

> **Kruse**, Säfken, and Kneib (2023). "Measuring Neural Complexity: A Covariance Penalty Approach". Under review at the *26th European Conference on Artificial Intelligence*

The **third** research question that is discussed is:

Can a hybrid modelling approach be used to combine distributional regression methods like GAMLSS with artificial neural networks?

Part III presents *neural additive models for location, scale, and shape* (NAMLSS) framework that combines the predictive power of classical deep learning models with the inherent advantages of distributional regression, while retaining the interpretability of additive models. Investigation of the method on various simulated and real data sets demonstrates the performance and advantages of the presented NAMLSS approach compared to other already established methods from statistics and deep learning alike. This proves to be an essential advantage, as an increasing focus of the deep learning community is on improving the interpretability and reliability of the approach.

> Thielmann, **Kruse**, Kneib, and Säfken (2023). "Neural Additive Models for Location Scale and Shape: A Framework for Interpretable Neural Regression Beyond the Mean.". Under review at the *Thirty-seventh Conference on Neural Information Processing Systems. arXiv preprint arXiv:2301.11862.*

Chapter 3: Summary and outlook

3.1 Conclusion

The combination of statistics and deep learning methods for estimating and evaluating semi-parametric models, or hybrid approaches as described in Part III, can take varying forms and impose different demands on the theoretical and practical aspects of the approaches, depending on the task and data at hand.

For statistical and deep learning models, performance in terms of prediction error plays an essential role in evaluating candidate models. In Part I, it can be seen how vital the prediction error, and in particular, the difference between the apparent error and the expected value of the prediction error, is for finding a scheme for optimal weights for model averaging. This difference, measured as a covariance penalty, is significant not only for statistical models, but also in the measurement of the complexity of deep learning models, as seen in Part II. It can be shown that while both approaches have different advantages and disadvantages, they also have many things in common and that different approaches in one area are also suitable for applications in the other.

In Part III, it has also been shown that it is not only possible to transfer model evaluation and assessment approaches from statistics to deep learning and vice versa, but that it is also possible to combine whole modelling frameworks such as generalised additive models for location, scale, and shape and artificial neural networks to create new hybrid modelling concepts. In this way, a simple mean prediction can be performed, and the ideas of the distributional regression of the GAMLSS approach can be combined with the neural network methods as the proposed NAMLSS framework. In particular, Part I presents the role of relative degrees of freedom as a special form of covariance penalty in determining the target criterion, which, when optimised, finds asymptotically optimal weights for model averaging of conditional linear mixed models. However, the new objective criterion represents a complex optimisation problem with equality and simultaneous inequality constraints, which must be solved via a multi-step optimisation process. For this purpose, a specially

adapted version of the augmented Lagrangian was derived, illustrated and demonstrated. With the new criterion and the self-developed optimisation strategy, it is possible to achieve results where the targeted weighting of different models based on the newly obtained weights achieves superior performance compared to other weighting schemes based on information criteria such as conditional AIC.

The need to identify and measure the underlying model complexity is obvious when considering statistical methods. However, moving away from statistical models and considering the class of deep learning models, it becomes apparent that model complexity is less explored and poses significant hurdles to model comparison and selection. Previous approaches attempted to introduce the concept of model complexity in terms of degrees of freedom from statistics into the world of deep learning through direct approximation based on finite difference methods. Part II, to further broaden our understanding of the complexity of deep learning models, focuses on generalising the topic of prediction error and the related topic of model complexity for a wide range of different model architectures and error measures. Three methods for estimating model complexity regarding covariance penalties a direct approximation based on an ensemble perturbation approach, k-fold cross-validation, and a (parametric) bootstrap method. The methods have demonstrated their advantages and possible shortcomings in various settings. The k-fold cross-validation exhibits a higher variance in its results. However, due to the relatively modest number of model training runs required, it has a relatively minor impact on computational cost. The direct approximation produces more consistent results, but at the cost of computational time, as it requires all underlying observations to be perturbed once. Bootstrapping proves to be a good compromise, and the approach is less resource-intensive than direct approximation, but more than cross-validation; however, it produces more consistent results than cross-validation. An interesting observation in the analysis of different deep learning architectures is that the complexity of deep learning models is less dependent on the number of model parameters or the depth of the assumed network, illustrating that the complexity of deep learning models is driven by more than just the number of parameters.

The popularity of deep learning models is largely based on their high effectiveness in various tasks regarding the high-level predictive power of the models used. One problem with these models is that their inner workings are not transparent, leading to poor interpretability of results. As a result of this black-box criticism, an increasing amount of research is focused on the explainability of deep learning models. One approach towards better explainability is the creation of hybrid models that try to combine the strengths of neural networks, i.e. the high predictive power, with

statistical models and their inherently good explanatory power. These hybrid models, like NAM (Agarwal et al., 2021), focus on models that only predict the mean and ignore other properties of the response distribution. Part III presents a combination of the ideas and concepts of the NAM approach with the ideas and inherent flexibility of GAMLSS-type distributional regression models, allowing the proposed NAMLSS framework to model the entire underlying response distribution, in addition to maintaining the same level of interpretability as the original NAM models. It can be seen that the proposed framework performs better than comparable deep learning or statistical models. Furthermore, it introduces a new level of (graphical) interpretability by modelling the different effects via their networks, which is a useful and important development compared to pure deep learning models. In contrast to other distribution-focused approaches such as GAMLSS, NAMLSS benefits from increased predictive power due to its underlying neural network structures and produces comparably better values regarding of log-likelihood results.

3.2 Outlook and direction of future research

The following section presents possible exciting avenues for future research based on the underlying paper findings in the order of the contributions.

Optimal weights for model averaging

The presented method so far focuses only on (conditionally) linear mixed models and neglects other forms of mixed models that deviate from the assumption of linearity. An extension or reimplementation of a criterion for determining squared loss-optimal weights for generalised linear models is another exciting step in weight optimisation research. The conditional model selection methods presented by Wood et al. (2016) are a particularly promising avenue.

Another possible extension is substituting the assumed error function in deriving the weighting criterion. In this way, a departure from the mean squared error could provide a generalisation of the approach. This goal could be achieved in conjunction with the research by Säfken and Kneib (2020) and the research on the covariance penalty approach presented in Part II. This may be particularly interesting for distributional regression models; see Kneib et al. (2021). Finally, extending the proposed method to boosting might also represent a compelling avenue for future research (Griesbach et al., 2021).

Deep learning model complexity

A phenomenon often encountered in the literature and practice is the so-called double descent, where the test error decreases even below the sweet spot of the u-shaped bias-variance curve as the model complexity is increased, which can occur not only for neural networks but also for other models such as regression trees or linear models (Belkin et al., 2020; Berner et al., 2021). An abstract definition of model complexity often measures this phenomenon. However, it needs to be clarified to what extent classical definitions, such as model complexity as the number of parameters of a model or, as shown in Part II, based on model inherent properties such as covariance penalties, should be used. Investigating of different possible definitions of model complexity and their relation to double descent presents an intriguing future research opportunity.

As a departure from the idea of the complexity of a model in terms of its parameters, there is also the concept of model complexity in terms of learnability. For example, the Rademacher complexity can be used to derive data-dependent upper bounds on the learnability of function classes, which helps us to make statements about how difficult it is to learn the underlying function class (Berner et al., 2021). This could also serve as a good measure of model complexity and for possible model selection, making exploring this approach an attractive possibility.

Neural additive models for location, scale and shape

In many situations, the data situation makes it necessary to model more than one response conditional on the covariates simultaneously.

One way of dealing with this is to incorporate copula-based methods into the NAMLSS framework. Copulas (Joe, 1997) allow the construction of multivariate continuous distributions over so-called copula functions and the corresponding marginal distributions. Copulas are particularly suitable for modelling correlation structures in the outcome of a regression. The use of copula methods is already possible within the GAMLSS framework. It would therefore be a natural extension of the NAMLSS approach presented and would only increase its utility and applicability. Similar to the extension of the GAMLSS framework based on Bayesian methods to *Bayesian additive models for location, scale and shape* by Umlauf et al. (2018), a similar approach lends itself to NAMLSS. Introducing a Bayesian-based training approach in conjecture with Bayesian modelling methods becoming increasingly popular makes this a logical next step in developing the NAMLSS framework.

Contributions

Part I

Model averaging for linear mixed models

Model averaging for linear mixed models via augmented Lagrangian

Contributing article:

Kruse, R. M., Silbersdorff, A., and Säfken, B. Model averaging for linear mixed models via augmented Lagrangian. *Computational Statistics & Data Analysis 167 (2022): 107351.*

Software:

Säfken, B., Rügamer, D., Baumann, P., and **Kruse, R. M**. R-Package 'cAIC4'. (2021).

Remark:

The original published version of the paper can be found in Appendix A.

Copyright:

Author contributions:

René-Marcel Kruse prepared the first draft, including the simulation studies, applications and implementations of the presented methods for the separately published cAIC4 R-Package. Alexander Silbersdorff and Benjamin Säfken added valuable input, suggested several notable modifications and proofread the manuscript.

Abstract:

Model selection for linear mixed models has been a focus of recent research in statistics. Yet, the method of model averaging has been sparsely explored in this context. A weight finding criterion for model averaging of linear mixed models is introduced, as well as its implementation for the programming language R. Since the optimization of the underlying criterion is non-trivial, a fast and robust implementation of the augmented Lagrangian optimization technique is employed. Furthermore, the influence of the weight finding criterion on the resulting model averaging estimator is illustrated through simulation studies and two applications based on real data.

Model averaging for linear mixed models via augmented Lagrangian

4.1 Introduction

The class of linear mixed models (Henderson, 1950) is a very powerful and flexible analytic tool, that enjoys popularity especially for the analysis of clustered and longitudinal data (Laird and Ware, 1982; Verbeke and Molenberghs, 2009), for spline smoothing (Ruppert et al., 2003; Wager et al., 2007) and for functional data analysis (Guo, 2002; Di et al., 2009).

Especially the development and advancement of software for fitting and evaluating linear mixed models is a very active field. It ranges from implementations for commercial statistics programs like `SAS`, to open-source versions like the de-facto standard in `R lme4` (Bates et al., 2015) or the `MixedModels` (Bates et al., 2020) Package for `Julia`. Due to the flexibility and thus, possible complexity of the models, the question of suitable model selection procedures becomes a focal point of research.

However, linear mixed model deviate from the imposed regularity conditions of classical linear models and thus introduce a problem with the use of information criteria for model choice, such as the widely adopted Akaike Information Criterion (Akaike, 1973; Wager et al., 2007). Furthermore, evaluating the suitability of the included random effects of models with nested or clustered structures suffer from limitations like boundary issues with likelihood-ratio tests (Crainiceanu and Ruppert, 2004; Wood, 2013). An overview of measures of explained variation and model selection in linear mixed-effects models can be found in Cantoni et al. (2021).

Vaida and Blanchard (2005) show, however, that it is possible to derive an AIC from the conditional form of the linear mixed effect model, which has proven to be particularly suitable accounting for possible shrinkage within the random effects (Säfken et al., 2021). Liang et al. (2008) suggest a version of the conditional AIC that corrects for the estimation uncertainty of the variance

parameters of the random effects. Still, this proposed version is computationally intensive as it relies on numerical approximation. Greven and Kneib (2010) prove that an analytical solution can be derived and thus, reduce the computational intensity of the corrected version of the conditional AIC.

Another approach addressing model uncertainty is model averaging. Instead of choosing a single model from a list of candidate models based on information criteria such as AIC or the Bayesian information criterion (Schwarz et al., 1978), a weighted average of the considered models is calculated and then used for analysis. An important key factor when applying model averaging is the selection of the underlying weights. Different proposals have been brought forward, the most prominent being the approach of information criteria based weights by Buckland et al. (1997). Yet, a majority of proposals aim at classical linear models and do encounter difficulties when applied to the model framework of linear mixed models. A proposal by Zhang et al. (2014) demonstrates that it is possible to construct an asymptotically optimal weight finding criterion for model averaging of linear mixed models based on the conditional AIC and a quadratic loss function. However, a computationally stable and fast optimization of such a weight determination criterion is not available up to date. The nonlinear nature of the criterion itself, as well as the nature of the underlying constraints in the form of simultaneous equality and inequality conditions, necessitates complex and advanced optimization methods that are not part of the basic version of the programming language `R` (R Core Team, 2019).

In this paper we present a weight finding criterion for the calculation of asymptotically optimal weights based on the work of Greven and Kneib (2010) and Zhang et al. (2014). In addition we present an implementation of the proposed weight finding criterion for the programming language `R`, which we have released as part of the R-Package `cAIC4` (Säfken et al., 2021). Furthermore, we describe the special nonlinear optimization under equality and inequality constraints of the underlying problem. We illustrate the approach of solving such a problem by applying the augmented Lagrangian method (Hestenes, 1969; Li et al., 2013) and present our implementation of the algorithm.

This paper is structured as follows: Section 4.2 introduces the theory and formulations of linear mixed models, as well as the estimation and the application of linear mixed models for spline smoothing. Section 4.3 presents the concept of the conditional AIC. This section also induces the concept of conditional model averaging and the proposed weight finding criterion. The following Section 4.4 provides an introduction to the underlying mathematical concepts of the augmented La-

grangian method, as well as its application to our weight finding optimization problem. Section 4.5 analyzes the properties of the implemented methods by applying them in three different simulation settings. Section 4.6 studies the proposed model averaging method applied to real-world examples, whereas the last Section 4.7, gives a summary of the findings of the previous sections and also gives an outlook of further work concerning model selection and model averaging of linear mixed models.

4.2 Linear mixed models

The general design of linear mixed models assumed in the following sections is

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

where $\boldsymbol{y}$ represents the vector of the n observed responses $\boldsymbol{y} = (y_1, \ldots, y_n)^T$, $\boldsymbol{X}$ and $\boldsymbol{Z}$ representing design matrices with full column ranks p and q, with the $p \times 1$ vector of fixed $\boldsymbol{\beta}$ and $\boldsymbol{b}$ as the $q \times 1$ vector of random effects. The $n \times 1$ vector $\boldsymbol{\varepsilon}$ represents the unobserved random errors. Both $\boldsymbol{b}$ and $\boldsymbol{\varepsilon}$ are assumed to be independent and follow a multivariate Gaussian distribution, such that

$$\begin{pmatrix} \boldsymbol{b} \\ \boldsymbol{\varepsilon} \end{pmatrix} \sim \mathcal{N}\left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \boldsymbol{D_\theta} & 0 \\ 0 & \boldsymbol{\Sigma} \end{pmatrix} \right\},$$

with $\boldsymbol{D_\theta}$ being a $q \times q$ block-diagonal, positive, semi-definite variance-covariance matrix that depends on a covariance parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_j)^T$ and $\boldsymbol{\Sigma}$ the overall model covariance matrix with dimension $n \times n$, which in the following illustrations is assumed to follow the standard case of $\sigma^2\boldsymbol{I}$. The normality assumption, however, is not mandatory and is only introduced for convenience, allowing for likelihood-based procedures to estimate unknown parameters in $\boldsymbol{D_\theta}$ and of the residual variance.

Furthermore let the marginal covariance matrix $\boldsymbol{V_\theta}$ of $\boldsymbol{y}$ be defined as follows

$$\boldsymbol{V_\theta} = \mathrm{cov}(\boldsymbol{y}) = \boldsymbol{\sigma}^2\boldsymbol{I} + \boldsymbol{Z}\boldsymbol{D}_\theta\boldsymbol{Z}^T.$$

The inherent randomness of the random effects makes it possible to formulate linear mixed models in two different forms, in a marginal or in a conditional ways. The marginal formulation treats the random effects as an additional part of the already random error term $\boldsymbol{\varepsilon}$ (Fahrmeir, 2013). The conditional formulation on the other hand approaches the random effects differently,

by treating them as penalized coefficients. In this form the conditional responses are distributed as follows

$$\boldsymbol{y}|\boldsymbol{b} \sim \mathcal{N}(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{b}, \sigma^2\boldsymbol{I}).$$

4.2.1 Estimation of linear mixed models

For given variance parameters $\boldsymbol{\theta}$, fixed as well as random effects can be estimated respectively predicted via

$$\begin{aligned} \hat{\boldsymbol{\beta}} &= \left(\boldsymbol{X}^T{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1}\boldsymbol{y}, \\ \hat{\boldsymbol{b}} &= \boldsymbol{D}_{\boldsymbol{\theta}}\boldsymbol{Z}^T{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1}(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}), \end{aligned} \tag{4.1}$$

where the resulting estimator of the fixed effects $\hat{\boldsymbol{\beta}}$ is the best linear unbiased estimator and it is also the maximum-likelihood estimator of $\boldsymbol{\beta}$ and the predictor of the random effects $\hat{\boldsymbol{b}}$ is also the best linear unbiased predictor of $\boldsymbol{b}$ (Harville, 1976). The corresponding profile log-likelihood for all underlying variance parameters $\boldsymbol{\theta}$ is thus up to a constant equal to

$$\ell_P(\boldsymbol{\theta}) = -\frac{1}{2}\left[\log|\boldsymbol{V}_{\boldsymbol{\theta}}| + (\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^T{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1}(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})\right]. \tag{4.2}$$

The maximization of the profile log-likelihood with respect to $\boldsymbol{\theta}$ delivers the ML-estimator $\hat{\boldsymbol{\theta}}_{ML}$. Instead of estimating $\boldsymbol{\theta}$ via profile log-likelihood, it is also possible to determine $\boldsymbol{\theta}$ via the marginal or restricted log-likelihood. Whereas the complementary restricted log-likelihood for $\boldsymbol{\theta}$ takes the following form (up to an additive constant) of

$$\ell_R(\boldsymbol{\theta}) = \ell_P(\boldsymbol{\theta}) - \frac{1}{2}\log\left|\boldsymbol{X}^T{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1}\boldsymbol{X}\right|, \tag{4.3}$$

maximizing this restricted log-likelihood results in the REML-estimator for $\boldsymbol{\theta}$, $\hat{\boldsymbol{\theta}}_{REML}$ (Harville, 1976). In a general setting the REML-estimator leads to a less biased estimation result than the ML-estimator (Fahrmeir, 2013).

4.2.2 Linear mixed models for spline smoothing

Apart from using the linear mixed models as a data analysis tool itself, this model class can also be used as a vehicle to fit semi-parametric models (Ruppert et al., 2003). This connection can most easily be explained for the case of truncated polynomials.

For the simple univariate smoothing case consider the following model

$$y_i = f(x_i) + \varepsilon_i, \quad i = 1, \ldots, n,$$

where $f(x_i)$ is represented by a sum of scaled basis functions and $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. In the case of truncated polynomials, the following basis representation is utilized

$$f(x) = \sum_{j=0}^{d} \beta_j x^j + \sum_{j=1}^{K} b_j \left(x - \varkappa_j\right)_+^d,$$

where the domain of x is partitioned by $K \in \mathbb{N}$ knots $\varkappa_1 < \cdots < \varkappa_K$ in such a way that for $d \in \mathbb{N}$

$$(z)_+^d = z^d \cdot I(z > 0) = \begin{cases} z^d & \text{if } z > 0, \\ 0 & \text{if } z \leq 0. \end{cases}$$

The penalised least-squares criterion is employed to prevent overfitting and to ensure smoothness of the estimated function, resulting in

$$\mathrm{ls}_{\mathrm{pen}}(\boldsymbol{\beta}, \boldsymbol{b}) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b}) + \boldsymbol{b}^T \boldsymbol{D}_\theta^{-1} \boldsymbol{b},$$

with $\boldsymbol{D}_\theta = \tau^2 \boldsymbol{I}_K$ where $\boldsymbol{\theta} = \tau^2$ and $\boldsymbol{\Sigma} = \sigma^2 \boldsymbol{I}$. Thus the relation between the variances is $\tau^2 = \xi\sigma^2$. In this case, given a fixed smoothing parameter ξ, the equation coincides with the best linear unbiased estimator for $\boldsymbol{\beta}$ and the best linear unbiased predictor for $\boldsymbol{b}$ from equation (4.1) in the linear mixed model case with fixed τ^2. The underlying parameter $\xi = \tau^2/\sigma^2$ can be understood as a trade-off between function fit and -smoothness. Interpreting this problem as a linear mixed effect model allows ξ to be understood as the variance ratio of random and fixed effects and therefore to be determined via the presented ML (4.2) or REML (4.3) approaches (Ruppert et al., 2003).

It is possible to represent penalized regression smoothers as part of mixed models, this allows the smoothing parameters to be estimated as part of the variance component parameters using the introduced likelihood procedures. As a consequence, linearly mixed models can be used to fit generalized additive mixed models (Wood, 2017).

4.3 Conditional model choice and model averaging

When it comes to the choice of linear mixed models and their random effect structures, the question arises as to which of the two likelihoods should be the basis for information criteria such as the AIC. The rationale for the choice depends on the intended application of the model (Vaida and Blanchard, 2005). The marginal approach allows statements about fixed population effects or of predictions about changed random effects structures. In contrast if the interest lies in statements based on the random effects of fitted models or in predictions based on existing random effect structures, the conditional form is particularly suitable. Due to these characteristic, the corresponding conditional AIC is, thus, better suited to select which random effects to include and which not to (Säfken et al., 2021). For a more mathematical investigation of the differences between the conditional and the marginal AIC, see Greven and Kneib (2010).

4.3.1 Conditional Akaike information criterion

One of the most widely used criteria for model selection is the Akaike Information Criterion (AIC) (Akaike, 1973). The AIC is an estimator of the relative Kullback-Leibler-Distance (Kullback and Leibler, 1951) and for simple linear regression models is up to a constant given by

$$\text{AIC} = -\frac{(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^T(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})}{\hat{\sigma}^2} + 2p,$$

with the number of parameters or degrees of freedom p. Considering a number of possible candidate models, the model that displays the lowest AIC value among all candidate models is most favourable. In more general this model selection criterion can also be derived as an estimator for the squared prediction error (Efron, 2004) as

$$\text{AIC} = (\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^T(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}) + 2\hat{\sigma}^2 \sum_{i=1}^{n} \left(\frac{\partial \hat{y}_i}{\partial y_i} \right),$$

with a substitution for the degrees of freedom first formalized by Stein et al. (1972)

$$\sum_{i=1}^{n} \frac{\partial \hat{y}_i}{\partial y_i} = \frac{\partial \hat{\boldsymbol{y}}}{\partial \boldsymbol{y}} = \text{tr}(\boldsymbol{H}) = \rho, \tag{4.4}$$

for simple linear regression models with hat matrix $\boldsymbol{H}$.

Two different AIC criteria can be employed when working with linear mixed models, the marginal AIC which is based on the marginal formulation of the log-likelihood, and the conditional AIC which is based on the conditional log-likelihood. Depending on the research question, the intention, as well as the interpretation, the respective approach varies (Vaida and Blanchard, 2005; Greven and Kneib, 2010).

The proposed estimator of the conditional AIC of Vaida and Blanchard (2005) takes the form of

$$\text{cAIC} = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b})^T \boldsymbol{V}_{\boldsymbol{\theta}} (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b}) + 2(\rho + 1).$$

The derivation of Vaida and Blanchard (2005) requires that the variance-covariance matrix of the random effects has to be known. Liang et al. (2008) propose a corrected version of the conditional AIC based on the numerical approximation of the degrees of freedom as in (4.4) and therefore mitigate the strictness of the assumptions in respect to the variance-covariance matrix.

This approach, however, introduces high computational costs. Greven and Kneib (2010) offer an analytical version of the bias correction term and allow the calculation of the corrected form of the cAIC without having to resort to complex numerical approximation. Theorem 3 of Greven and Kneib (2010) allows ρ to be formulated as

$$\rho = \text{tr}\left(\hat{\boldsymbol{H}}\right) + \sum_{j=1}^{J} \frac{\partial \hat{\boldsymbol{\theta}}_j}{\partial \boldsymbol{y}^T} \hat{\boldsymbol{A}} \boldsymbol{Q}_j \hat{\boldsymbol{A}} \boldsymbol{y},$$

where $\hat{\boldsymbol{H}} = \boldsymbol{I} - \sigma^2 \hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1} + \sigma^2 \hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1/2} (\hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1/2} \boldsymbol{X} (\boldsymbol{X}^T \hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1} \boldsymbol{X})^{-1} \boldsymbol{X}^T \hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1/2}) \hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1/2}$, $\boldsymbol{Q}_j = \partial \boldsymbol{V}_{\boldsymbol{\theta}} / \partial \boldsymbol{\theta}_j$, furthermore $\hat{\boldsymbol{\theta}}_j$ is the j-th element of $\hat{\boldsymbol{\theta}}$ and with $\hat{\boldsymbol{A}} = \sigma^2 \hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1/2} (\boldsymbol{I} - (\hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1/2} \boldsymbol{X} (\boldsymbol{X}^T \hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1} \boldsymbol{X})^{-1} \boldsymbol{X}^T \hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1/2})) \hat{\boldsymbol{V}_{\boldsymbol{\theta}}}^{-1/2}$.

To cover possible parameters on the boundary of the parameter space, we have to partition it similar to Self and Liang (1987). In this case, $\boldsymbol{\theta}$ consists of all elements of the matrix $\boldsymbol{D}_{\boldsymbol{\theta}}$ and it is sorted in such a way that the last s elements of the estimator $\hat{\boldsymbol{\theta}}$ are equal to zero and with $\hat{\boldsymbol{\eta}} = (\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_{J-s})^T$ it can be shown under Theorem 3 of Greven and Kneib (2010) that

$$\frac{\partial \hat{\boldsymbol{\theta}}_j}{\partial \boldsymbol{y}^T} = 0, \quad j = J - s + 1, \ldots, J, \quad \frac{\partial \hat{\boldsymbol{\eta}}}{\partial \boldsymbol{y}^T} = \boldsymbol{B}^{-1} \boldsymbol{G} = 0,$$

where $\boldsymbol{B}$ is a $(J-s)\times(J-s)$ matrix with (i,j)-th element such that

$$(b)_{ij} = -\operatorname{tr}(\hat{\boldsymbol{V}}^{-1}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{V}}^{-1}\hat{\boldsymbol{Q}}_j) - n\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{A}}\boldsymbol{y}\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_j\hat{\boldsymbol{A}}\boldsymbol{y}(\boldsymbol{y}^T\hat{\boldsymbol{A}}\boldsymbol{y})^{-2} + 2n\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_j\hat{\boldsymbol{A}}\boldsymbol{y}(\boldsymbol{y}^T\hat{\boldsymbol{A}}\boldsymbol{y})^{-1},$$

with $\boldsymbol{G}$ being a $(J-s)\times n$ matrix with the i-th row given by

$$\boldsymbol{g}_i = 2n\Big\{-(\boldsymbol{y}^T\hat{\boldsymbol{A}}\boldsymbol{y})^{-2}\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{A}}\boldsymbol{y}\boldsymbol{y}^T\hat{\boldsymbol{A}} + (\boldsymbol{y}^T\hat{\boldsymbol{A}}\boldsymbol{y})^{-1}\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{A}}\Big\}.$$

These expressions allow to calculate ρ directly so that no numerical procedures are necessary. The R-package `cAIC4` (Säfken et al., 2021) incorporates the corrected conditional AIC form and bias correction, and thus allows to perform model selection based on the conditional AIC and to calculate a analytical version of the degrees of freedom for models under consideration.

4.3.2 Conditional model averaging

Consider a given series of K possible linear mixed-effects candidate models according to (4.2), with the following form

$$\boldsymbol{y} = \boldsymbol{X}_k\boldsymbol{\beta}_k + \boldsymbol{Z}_k\boldsymbol{b}_k + \boldsymbol{\varepsilon}, \quad \boldsymbol{b}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{D}_{\boldsymbol{\theta_k}}), \quad k = 1,\dots,K.$$

The fixed and random effects, as well as the variance-covariance matrices, can be determined via the REML approach, presented in (4.1). Hence the conditional mean is given by $\hat{\boldsymbol{y}}_k = \boldsymbol{X}_k\hat{\boldsymbol{\beta}}_k + \boldsymbol{Z}_k\hat{\boldsymbol{b}}_k$, leading to the following representation of the predicted values in terms of the estimated hat matrix $\hat{\boldsymbol{H}}_k$ as $\hat{\boldsymbol{y}}_k = \hat{\boldsymbol{H}}_k\boldsymbol{y}$.

The purpose of model averaging is to compose a weighted average over the random, as well as the fixed effect, estimators. For this consider a corresponding weighting vector $\boldsymbol{w} = (w_1,\dots,w_K)^T$ belonging to the set $\mathcal{W} = \{\boldsymbol{w} \in [0,1]^K : \sum_{k=1}^K w_k = 1\}$. The model averaging estimator is thus described by

$$\hat{\boldsymbol{y}}(\boldsymbol{w}) = \sum_{k=1}^K w_k\hat{\boldsymbol{y}}_k = \sum_{k=1}^K w_k\hat{\boldsymbol{H}}_k\boldsymbol{y} = \hat{\boldsymbol{H}}(\boldsymbol{w})\boldsymbol{y},$$

with $\hat{\boldsymbol{H}}(\boldsymbol{w}) = \sum_{k=1}^K w_k\hat{\boldsymbol{H}}_k$.

One of the more straightforward and widely used methods to determine the weights for the model average is based on a proposal by Buckland et al. (1997). This approach can be sometimes found in the literature labeled as smoothed weights and the weight finding criterion takes on the following form

$$w_k = \frac{\exp\left(-\mathcal{I}_k/2\right)}{\sum_{i=1}^{K} \exp\left(-\mathcal{I}_i/2\right)}, \tag{4.5}$$

with $\mathcal{I}_k$ representing the information criteria value for the respective candidate model k. A second approach is to derive the weights in such a way that in theory the model averaging estimator is asymptotically optimal, as in Zhang et al. (2014). The resulting estimator is optimal in the sense that the squared error of the calculated model average estimator is asymptotically equal to that of the infeasible best possible model average estimator. The authors achieve this by utilizing the squared loss of the model averaging estimator to derive a suitable criterion for weight determination.

The underlying loss-function takes the following form of

$$L(\boldsymbol{w}) = (\hat{\boldsymbol{y}}(\boldsymbol{w}) - \boldsymbol{\mu})^T(\hat{\boldsymbol{y}}(\boldsymbol{w}) - \boldsymbol{\mu}),$$

where $\hat{\boldsymbol{y}}(\boldsymbol{w})$ represents the model average estimator and $\boldsymbol{\mu}$ the true but unknown mean.

By applying the theorem by Stein et al. (1972) the expected loss is given by

$$\begin{aligned} &E_{\boldsymbol{y}|\boldsymbol{b}}\left((\hat{\boldsymbol{y}}(\boldsymbol{w}) - \boldsymbol{\mu})^T(\hat{\boldsymbol{y}}(\boldsymbol{w}) - \boldsymbol{\mu})\right) = \\ &E_{\boldsymbol{y}|\boldsymbol{b}}\left((\boldsymbol{y} - \hat{\boldsymbol{y}}(\boldsymbol{w}))^T(\boldsymbol{y} - \hat{\boldsymbol{y}}(\boldsymbol{w})) + 2\sigma^2\boldsymbol{w}^T\boldsymbol{\rho} - n\sigma^2\right), \end{aligned} \tag{4.6}$$

the $K \times 1$ elements of $\boldsymbol{\rho}$ are being defined as $\rho_k = \operatorname{tr}\left(\partial\hat{\boldsymbol{y}}_k/\partial\boldsymbol{y}^T\right)$. The method presented here and especially formula (4.6) essentially rely on Stein's theorem.

In turn, this requires the normality assumption to hold for the conditional model $\boldsymbol{y}|\boldsymbol{b}$. There are approaches to generalize this method beyond normality, see for Ye (1998) and Efron (2004) or Säfken and Kneib (2020) in the context of mixed models. Such an extension is highly relevant as it would not only allow for other error distributions but also for other random effects distributions that as are used for robust linear mixed models which are based on skewed t distributions as proposed in Lin and Lee (2006) and Ho and Lin (2010).

Based on the design of the model averaging estimator, the weight finding criterion is, therefore, defined as follows

$$C(\boldsymbol{w}) = (\boldsymbol{y} - \hat{\boldsymbol{y}}(\boldsymbol{w}))^T (\boldsymbol{y} - \hat{\boldsymbol{y}}(\boldsymbol{w})) + 2\sigma^2 \boldsymbol{w}^T \boldsymbol{\rho}, \tag{4.7}$$

thus, the optimal vector of weights $\hat{\boldsymbol{w}}$ for the $\hat{\boldsymbol{y}}(\hat{\boldsymbol{w}})$ minimizes this criterion, such that $\hat{\boldsymbol{w}} = \text{argmin}_{\boldsymbol{w} \in \mathcal{W}} \hat{C}(\boldsymbol{w})$.

4.4 Practical model averaging with cAIC4

The weight selection criterion (4.7) can be seen as a nonlinear optimization problem, where the weights are subject to equality constraints and to bound constraints alike and can be formulated as

$$\min C(\boldsymbol{w}) \quad \text{subject to} \quad \sum_{k=1}^{K} w_k = 1 \text{ and } 0 \leq w_k \leq 1, \; k = 1, \ldots, K. \tag{4.8}$$

A generalized representation of the optimization problem of the target criterion is shown in Figure 4.1.

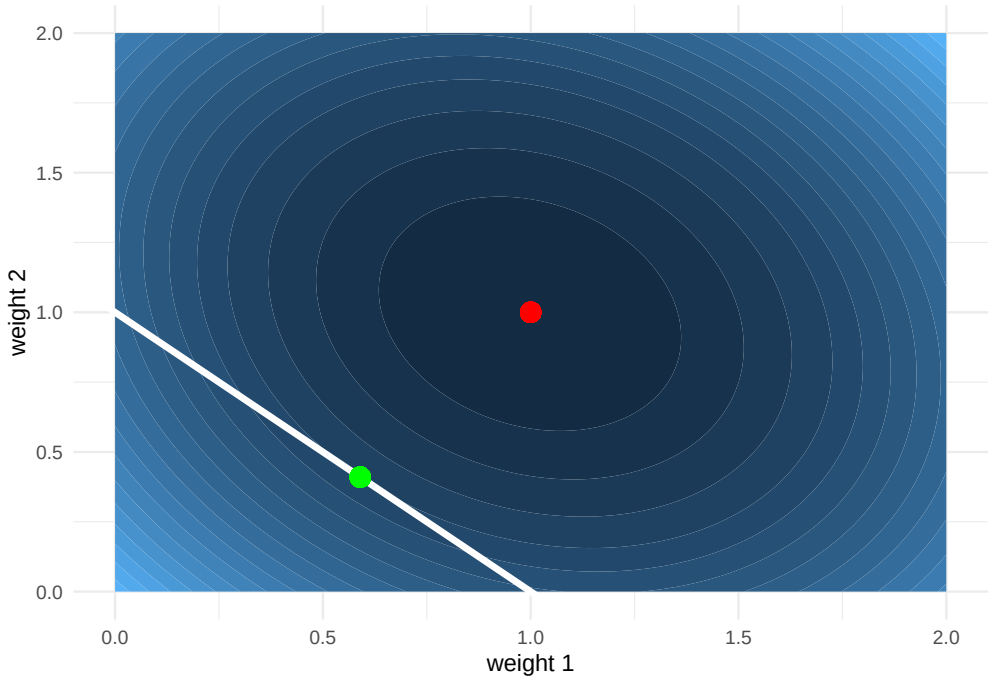

Figure 4.1: All possible values of the target weight finding criterion (4.7) for an example with two candidate models. The red dot marks the global minimum, the white line the assumed possible maximum values of the weights, and the green dot the optimum of the weights resulting from the restrictions.

To minimize a nonlinear problem such as (4.8), that is subject to equality and inequality constraints a more complex optimization approach has to be employed. The R-Packages Alabama (Varadhan, 2015) and Rsolnp (Ghalanos and Theussl, 2015) offer existing implementations of solvers for such a kind of optimization problem. The methods, however, implemented in the packages above require specific knowledge of the optimization methods and are further complicated by their broad general form. In response we propose a new implementation that is constructed in such a way that the setup, configuration and optimization of the underlying system can be performed easily.

4.4.1 The augmented Lagrange method with equality constraints

A common approach for solving such a problem is the augmented Lagrangian method (Hestenes, 1969; Powell, 1969). This approach adds a penalty term to the original target function, that represents a multiple of the constraint violations at each iteration and thus attempts to force the optimization result into the bound constraint's feasible solution space (Avriel, 2003). The augmented Lagrangian approach is based on the penalty method but aims to circumvent potential ill-conditioning inherent to these methods by directly integrating an estimator of the Lagrange multiplier into the target function (Nocedal and Wright, 2006). To introduce the augmented Lagrangian in its general form, let us assume a simple limited optimization problem with respect to a set of weight variables which, for the sake of simplicity, will all be assumed to be real-valued for the moment, i.e. $\boldsymbol{w} \in \mathbb{R}^K$ with following form of

$$\min C(\boldsymbol{w}) \quad \text{subject to} \quad h(\boldsymbol{w}) = 0, \tag{4.9}$$

where $C(\boldsymbol{w})$ is the cost function of the optimisation with $C : \mathbb{R}^K \to \mathbb{R}$ and the equality constraint functions $h = (h_1, \ldots, h_m)^T : \mathbb{R}^K \to \mathbb{R}^m$. The Lagrange function is described by

$$\mathcal{L}(\boldsymbol{w}, \boldsymbol{\lambda}) = C(\boldsymbol{w}) + \boldsymbol{\lambda}^T h(\boldsymbol{w}),$$

the Lagrange multiplier is given as $\boldsymbol{\lambda} \in \mathbb{R}^m$.

A possible problem with the general form of the objective function is that it does not necessarily have to be convex near the solution, which prevents duality methods like the Lagrangian from being effective. By adding the penalty term $\frac{\gamma}{2} h(\boldsymbol{w})^T h(\boldsymbol{w})$ with $\gamma > 0$, it is possible to impose a local convexity on the objective function, such that when the penalty γ term is sufficiently large, the Lagrangian will be locally convex (Luenberger et al., 2008).

The resulting augmented Lagrange function $\mathcal{L}_A$ is defined as follows

$$\mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) \stackrel{\text{def}}{=} C(\boldsymbol{w}) + \boldsymbol{\lambda}^T h(\boldsymbol{w}) + \frac{\gamma}{2} h(\boldsymbol{w})^T h(\boldsymbol{w}).$$

The resulting problem is equivalent to the original problem (4.9), since the penalty term does not change the objective function, the Lagrange multiplier and the optimal values and solution point. This in turn allows to solve the underlying problem by an interactive process in $\boldsymbol{\lambda}$. The approach for the optimization of the augmented Lagrangian can be seen in Algorithm 1.

Algorithm 1 Augmented Lagrangian method

Input: Initial weights $\boldsymbol{w}^0 \in \mathbb{R}^n$, tolerance η, multipliers $\boldsymbol{\lambda}^0$, penalty $\gamma_0 > 0$, increment ϵ

Output: Optimal values $\boldsymbol{w}^\star$, multipliers $\boldsymbol{\lambda}^\star$, penalty parameter $\gamma^\star$

while $\|\nabla \mathcal{L}_A(\boldsymbol{w}^l, \boldsymbol{\lambda}^l; \gamma_l)\| > \eta$ **do** solve for the target function with respect to $\boldsymbol{w}^{l+1}$ in a way that

$$\mathcal{L}_A(\boldsymbol{w}^{l+1}, \boldsymbol{\lambda}^l; \gamma_l) < \mathcal{L}_A(\boldsymbol{w}^l, \boldsymbol{\lambda}^l; \gamma_l)$$

update the Lagrange multipliers such that

$$\lambda^{l+1} = \lambda^l + h(\boldsymbol{w}^{l+1});$$

set the constraint γ such that

$$\gamma^{l+1} = \epsilon \gamma^l$$

set $l = l + 1$

4.4.2 Weight optimization via augmented Lagrangian

In its general form, the augmented Lagrangian only applies to equality constraint problems. To employ the method to our problem, the constraints have to be modified in such a fashion, that they include the bound constraints, i.e. the restrictions of the upper and lower limits of the possible weight values. The augmented Lagrange function subject to the problem at hand can be formulated as follows

$$\mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) = C(\boldsymbol{w}) + \boldsymbol{\lambda} h(\boldsymbol{w}) + \frac{\gamma}{2} h(\boldsymbol{w})^T h(\boldsymbol{w}).$$

The optimization under both constraints makes it necessary to divide the optimization into two different operations.

In the first, the augmented Lagrangian is applied only to the equality condition i.e. the sum of all weights must add up to one. After an approximate solution for the problem has been found, the next part of the optimization is to incorporate the bound-constraints. The lower and upper value bound-constraints of the weights are here left out and are considered explicitly in an additional step of the optimization. In this sub-problem, a sequential quadratic programming approach is used to solve the following nonlinear quadratic problem of (Nocedal and Wright, 2006)

$$\min \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) \quad \text{subject to} \quad 0 \leq w_k \leq 1,\ k = 1, \ldots, K.$$

At each iteration step j, a quadratic problem $q_j(\boldsymbol{w})$ with fixed γ and $\boldsymbol{\lambda}$ stemming from the results of the first step of the optimization, is solved for $\boldsymbol{w}$ according to

$$\min\ q_j(\boldsymbol{w}) = \nabla \mathcal{L}_A(\boldsymbol{w}_j, \boldsymbol{\lambda}_j; \gamma)^T (\boldsymbol{w} - \boldsymbol{w}_j) + \frac{1}{2} (\boldsymbol{w} - \boldsymbol{w}_j)^T \mathcal{H} (\boldsymbol{w} - \boldsymbol{w}_j).$$

The Lagrangian's gradient assumes the form of

$$\begin{aligned} \nabla \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) &= (\nabla_{\boldsymbol{w}} \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma), \nabla_{\boldsymbol{\lambda}} \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma)) \\ &= \left(\nabla_{\boldsymbol{w}} \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma), h(\boldsymbol{w})^T \right) \in \mathbb{R}^{1 \times (K+m)}, \end{aligned}$$

with $\nabla_{\boldsymbol{w}} \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) = \nabla C(\boldsymbol{w}) + (\boldsymbol{\lambda} + \gamma h(\boldsymbol{w}))^T \nabla h(\boldsymbol{w}) \in \mathbb{R}^{1 \times K}$. The corresponding Hessian of the augmented Lagrangian can be formulated as a block matrix

$$\mathcal{H} = \nabla^2 \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) = \begin{pmatrix} \nabla^2_{\boldsymbol{ww}} \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) & \nabla h(\boldsymbol{w})^T \\ \nabla h(\boldsymbol{w}) & 0 \end{pmatrix} \in \mathbb{R}^{(K+m) \times (K+m)},$$

where$\nabla_{\boldsymbol{ww}} \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) = \nabla^2 C(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) + (\boldsymbol{\lambda} + \gamma h(\boldsymbol{w}))^T \nabla^2 h(\boldsymbol{w}) + \gamma \nabla h(\boldsymbol{w})^T \nabla h(\boldsymbol{w}) \in \mathbb{R}^{K \times K}$.

The projection on the convex set $B = \{\boldsymbol{w} \mid a \leq w_k \leq b\}$ is described by $\mathcal{P}$, which is defined component-wise as

$$\mathcal{P}(w_k, a, b) = \begin{cases} a & \text{if } w_k \leq a, \\ w_k & \text{if } w_k \in (a, b), \\ b & \text{if } w_k \geq b, \end{cases} \qquad \text{for all } k = 1, 2, \ldots, K.$$

such that given a vector $\boldsymbol{w} \in B$ and the imposed bound constraints for each weight, we can show that the defining necessary properties of an $\boldsymbol{w}$ to be considered the solution of such problem, as well as the needed first-order-condition, are given by adjusting the general Karush-Kuhn-Tucker (KKT) conditions (Ruszczynski, 2011) such that

$$\boldsymbol{w} - \mathcal{P}(\boldsymbol{w} - \nabla_{\boldsymbol{w}} \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma), 0, 1) = 0.$$

If the calculated weights meet both the equality and the bound constraints, the optimal solution is found. If, however, only the equality constraints are fulfilled by the resulting weights, the Lagrange multiplier estimates $\boldsymbol{\lambda}$ are adjusted to allow a better estimation in the next iteration. If the equality constraints are not met, the value of the penalty parameter γ is increased with the aim of forcing the results into the feasible space to minimize the constraint violations.

4.5 Simulation studies

To illustrate the features and capabilities of the model averaging approach presented above, we conduct three different simulation studies in which we investigate its finite-sample properties.

4.5.1 Augmented Langrangian weights for smoothing splines

In the first simulation setting we use the connection between mixed models and smoothing splines as presented in Section 4.2.2 and investigate the behaviour of the proposed methods presented from Section 4.4 on these types of models. Comparing parametric and semiparametric models by means of information criteria is often difficult. In particular due to their inherent flexibility, spline models will generally offer superior in-sample predictive capacity in comparison to standard linear models – at the cost of consuming a much higher number of degrees of freedom. Thus the question arises how the presented model weighting criterion (4.8) incorporates different linear and nonlinear candidate models.

For this purpose, we simulate data where the underlying data-generating model incorporates a quadratic P-spline term. Notice that we use P-splines instead of the truncated polynomial splines presented in Section 4.2.2 due to their enhanced numerical and computational stability (Eilers and Marx, 1996). Subsequently a linear model and a linear mixed model are fitted to the data, where the mixed model includes a spline term. The variance of the spline term takes on different increasing values for each simulation $\tau_b^2 \in \{0, 0.5, 1, \ldots, 9, 9.5, 10\}$. The variance of the residuals is kept

constant, where each model combination is simulated for $\sigma_\varepsilon^2 \in \{1, 2, 4\}$. Every model combination is simulated 1000 times, with each simulation containing 100 observations.

Two models are fitted to the simulated data, a P-spline based semiparametric model (model 1 associated with w_1) of the form

$$\text{Model 1 } (w_1) : y_i = \beta_0 + f(x_i) + \varepsilon_i, \tag{4.10}$$

which is fitted using a linear mixed model as described in Section 4.2.2 and a classical linear model (model 2 associated with $w_2 = 1$ - w_1)

$$\text{Model 2 } (w_2) : y_i = \beta_0 + \beta_1 x_i + \varepsilon_i. \tag{4.11}$$

The proposed weight finding criterion (4.7) is applied to the two candidate models. Subsequently, the resulting weights are averaged over all simulations for each model constellation. The results of these calculations for the given combination of variances are shown in Figure 4.2.

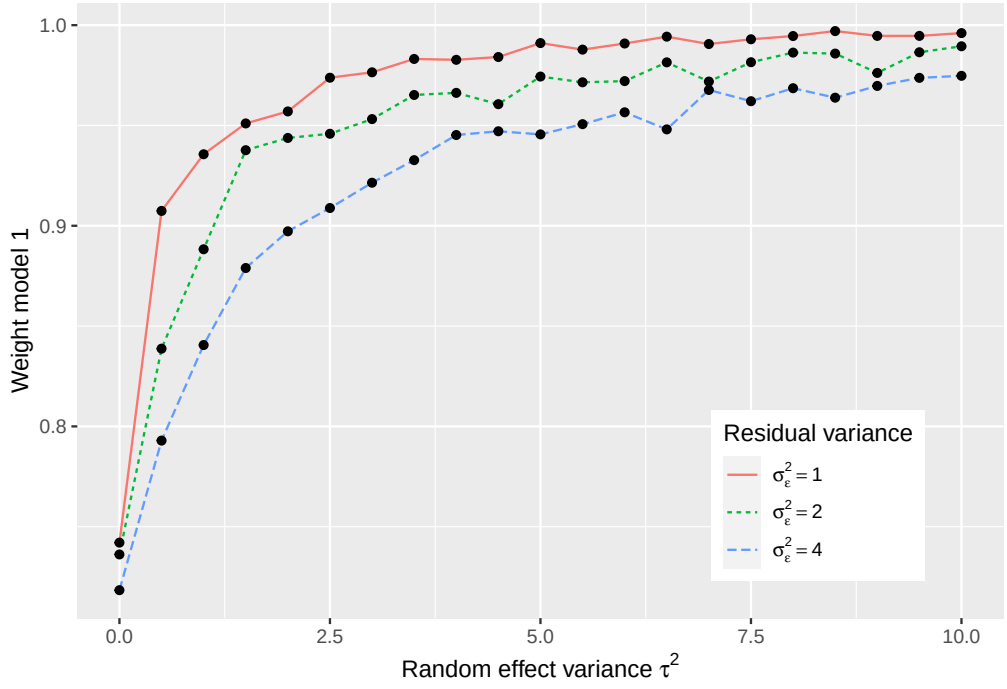

Figure 4.2: Weight of model (4.10) associated with weight w_1 in comparison to the weight w_2 of model (4.11) for different random effects variance values and different values of the error variance.

It can be seen that with the increasing variance of the random effect (i.e. a decreasing penalty parameter) used to generate the data, the weight for the model with the spline element increases

as well. This demonstrates that the improvement in the explanatory power of the spline model is detected by the proposed weight selection criterion and thus leads to a higher contribution of the semiparametric (or mixed) model to the resulting average model estimator. It also appears that given a higher residual variance the increase of the weights is slower, however, with a rising signal to noise ratio ($\tau^2/\sigma_\varepsilon^2$), we observe the anticipated shift towards a higher weight given to the semiparametric (or mixed) model.

4.5.2 Weights for multi-cluster hierarchical models

The second simulation study investigates the algorithm's behaviour in a multi-cluster mixed effects model framework as well as accounting for both non-normally distributed error terms entailing outliers and correlated error terms.

For the baseline data simulation, a true data generating linear mixed model is assumed that contains an intercept and two cluster levels with a random intercept each. The data generating model takes on the following form

$$y_{i,j,l} = \beta_0 + b_{1,j} + b_{2,l} + \varepsilon_{i,j,l}, \quad j,l = 1,\dots,10, \quad i = 1,\dots,100.$$

At first, the residuals and the random effects follow normal distributions, i.e. $\varepsilon_{i,j,l} \sim \mathcal{N}(0,\sigma^2)$, $b_{1,j} \sim \mathcal{N}(0,\tau_1^2)$ and $b_{2,l} \sim \mathcal{N}(0,\tau_2^2)$. For both levels the number of clusters in the true underlying model is 10 each. Each cluster consists out of 100 simulated individual observations. The random effects of the respective clusters are simulated such that the variances of the random effects fulfill $\tau_1^2 = 1 - \tau_2^2$. Each model setup is simulated 1000 times. Two linear mixed models are fitted to each dataset, whereby each model contains only one of the two random intercepts, i.e.

$$\text{Model 1 } (w_1): \quad y_{i,j} = \beta_0 + b_{1,j} + \varepsilon_{i,j},$$

and

$$\text{Model 2 } (w_2): \quad y_{i,l} = \beta_0 + b_{2,l} + \varepsilon_{i,l}.$$

The implemented weight choice criterion (4.7) is used to calculate an model average estimator based on the two candidate models. Table 4.1 and Figure 4.3 show the results of the simulations for changing variances of the random effect and a constant residual variance.

τ_1^2	0	0.25	0.50	0.75	1
Mean	0.002	0.260	0.499	0.499	0.989
Std.Error	0.002	0.136	0.174	0.137	0.004

Table 4.1: Calculated mean weights (and standard errors) for model 1 (w_1) for different given variance values.

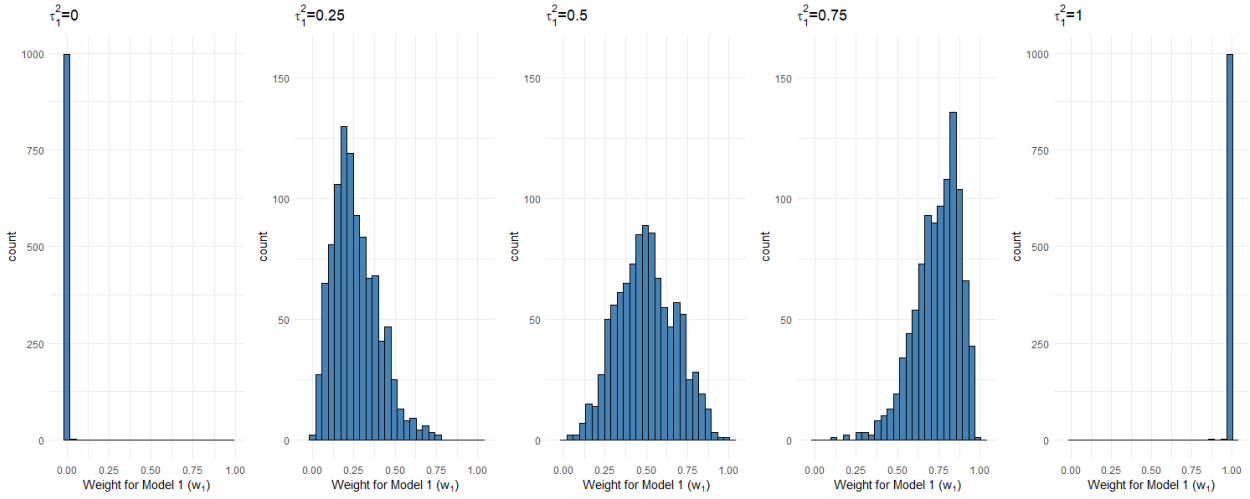

Figure 4.3: Histograms of calculated weights for the first model for five different given variances of the random effects

It becomes apparent that with an increasing variance of the first random effect, and therefore a decrease of the variance of the second random effect, the weight choice favours the first random effect. Furthermore, as can be seen in Table 4.1, the weights show an analogous behaviour in the case of a decrease of the first random effect variance. In the case that the variance of one of the two random effects is on the boundary, i.e. equal to zero, the corresponding weight is close to but not exactly zero. Furthermore the calculated weights show, that for $\tau_1 = \tau_2 = 0.5$ on average a model weight of 0.499 is chosen for model 1 and thus reflects the proportion of variation from the associated random effects. Ultimately, it can be observed that the computed weights reflect the respective simulated random effect variance. This indicates that the presented model averaging estimator can recognize the information from the multi-level mixed model structure and consequently calculate weights that lead to a model that closely represents the underlying true data generating model.

The framework of linear mixed models assumes normally distributed random effects and errors terms allowing for computational more convenient approaches. Violations of these assumptions, such as those caused by outliers or serially correlated within-subject errors, lead to less robust

models and unreliable inference results. To explore the extent to which the presented method is affected by these violations, two additional simulations are conducted. The first simulation adapts the first simulation design, however, outliers are introduced by selecting one to 50 values of the first random effect, where the number of potential outliers is drawn from a discrete uniform distribution, and scaled with random draws from a continuous uniform distribution ranging from 3 to 5 such that the adapted random effect $\tilde{b}_{1,j}$ is given by $\tilde{b}_{1,j} = \psi b_{1,j}$, where $\psi \sim \mathcal{U}(3, 5)$. In the second simulation the original simulation design is modified by including serial within-subject correlated errors. These are introduced by inducing a first order serial correlation with a correlation parameter of 0.5 such that $\tilde{\varepsilon}_{i,j,l} = 0.5\tilde{\varepsilon}_{i-1,j,l} + \varepsilon_{i,j,l}$. The results of both simulations are displayed in Figures 4.4 and 4.5 as well as Tables 4.2 and 4.3.

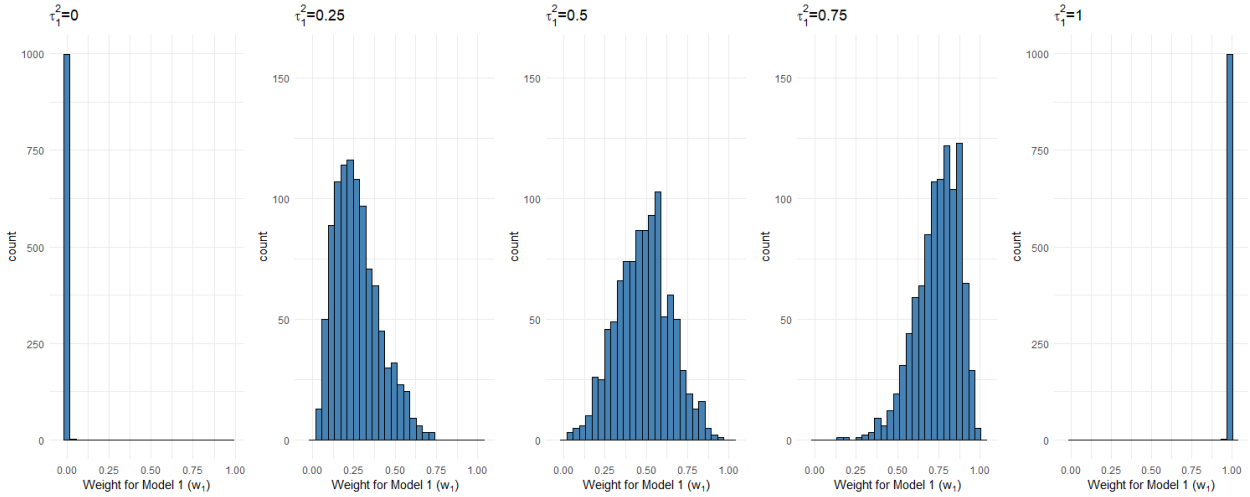

Figure 4.4: Histograms of calculated weights for the first model for five different given variances of the random effects with simulated correlated within-subject errors for the first random effect.

τ_1^2	0	0.25	0.50	0.75	1
Mean	0.001	0.271	0.482	0.742	0.989
Std.Error	0.001	0.135	0.163	0.131	0.003

Table 4.2: Calculated mean weights (and standard errors) for model 1 (w_1) for different given variance values with simulated correlated within-subject errors for the first random effect.

The results indicate that the presented method is relatively robust to the portrayed violations of the standard assumptions. This primarily stems from the fact that our method examines the relative fits of the models in comparison to each other. If a model is less able to explain the underlying data, then that model receives a smaller weight. It should be noted that linear mixed

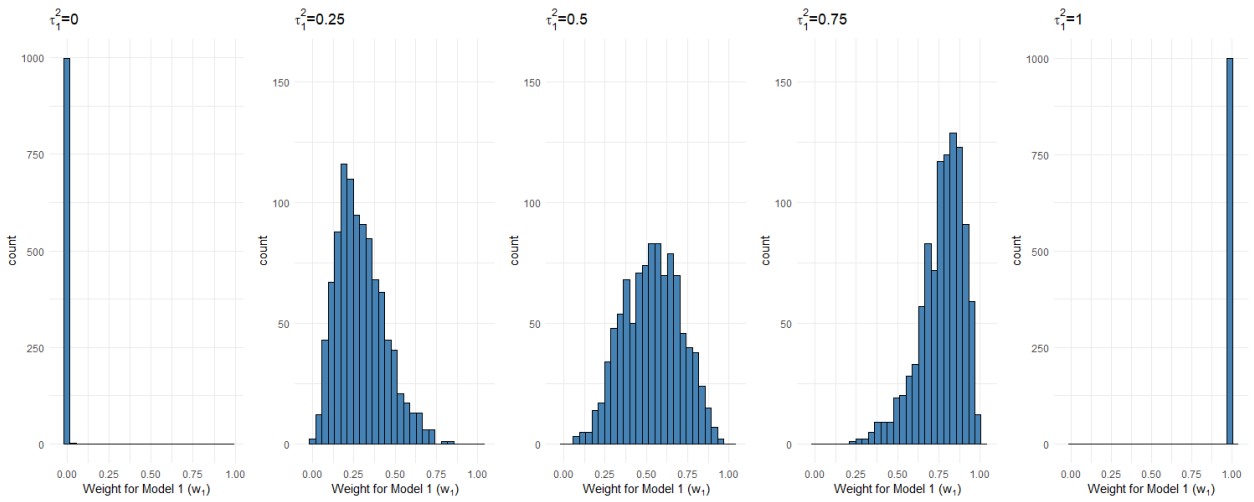

Figure 4.5: Histograms of calculated weights for the first model for five different given variances of the random effects with simulated outliers for the first random effect.

τ_1^2	0	0.25	0.50	0.75	1
Mean	0.001	0.292	0.534	0.765	0.989
Std.Error	0.001	0.145	0.172	0.134	0.003

Table 4.3: Calculated mean weights (and standard errors) for model 1 (w_1) for different given variance values with simulated outliers for the first random effect.

models, which attempt to make the model class more robust against these influences, such as Ho and Lin (2010) are a valid alternative to the classic linear mixed model in these settings. However the distributions of these robust models deviate from the normality assumption, thus an extension of the proposed model averaging scheme especially for formula (4.6) needs to be found in order to overcome the misspecification.

4.5.3 Relationship between weights and fixed effects

In this section, we compare our method with other model averaging methods, whereby to allow for comparisons between our implementation with the concept of Zhang et al. (2014) we employ a design based on Example 1 of their paper. In contrast to the two previous simulation settings the focus of this simulation study lies on the calculated model average estimator and the accuracy of the method in comparison to already implemented approaches. For this analysis, data is generated by a data-generating model which contains three fixed effects in the form $\boldsymbol{\beta} = (1, 0.2, 0.4)$ whereby the j-th row of the $\boldsymbol{X}_i$ matrix takes the following form $(1, x_{i,j_2}, x_{i,j_3})$. The true underlying model also

features three random effects, one random intercept and two random slopes. The elements of the j-th row of the $\boldsymbol{Z}_i$ matrix take the following form $(1, z_{i,j_2}, z_{i,j_3})$. The respective values of the design matrices $\boldsymbol{X}$ and $\boldsymbol{Z}$ originate independently from an $\mathcal{N}(0,1)$ distribution. The underlying data-generating model has 20 groups with 10 observations each. The data is simulated with a standard deviation of the residuals of $\sigma \in \{0.3; 0.9\}$. Furthermore, each model combination is simulated with four different random effect co-variance matrices. These four matrices are as follows

$$\boldsymbol{D}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \boldsymbol{D}_2 = \begin{pmatrix} 1.4 & 0 & 0 \\ 0 & 1.2 & 0 \\ 0 & 0 & 0.4 \end{pmatrix},$$

$$\boldsymbol{D}_3 = \begin{pmatrix} 1.4 & 0.4 & 0 \\ 0.4 & 1.2 & 0 \\ 0 & 0 & 0.4 \end{pmatrix}, \quad \boldsymbol{D}_4 = \begin{pmatrix} 1.4 & 0.4 & 0.6 \\ 0.4 & 1.2 & 0.2 \\ 0.6 & 0.2 & 0.4 \end{pmatrix}.$$

These covariance structures were chosen to incorporate various levels of complex random effect structures into this study, and to therefore determine to what extent these influence the ability of the different methods of finding weights for model averaging. For each of the configurations, 100 independent datasets are generated. The candidate models used for approximation include at least the fixed and the random intercept, the further candidate models include one of the two fixed coefficients and further random effects. Based on the fitted candidate models, weights for model averaging are now calculated as smooth weights as introduced in (4.5) based on the conditional AIC, as well as weights based on the presented asymptotic optimal approach (4.7). The ability of the different methods to calculate the model averaging approach is evaluated over the respective average squared loss.

	σ	D_1	D_2	D_3	D_4
Asymptotic	0.3	323.751	264.324	274.753	256.289
	0.9	360.498	285.706	290.286	290.674
Smoothed	0.3	337.241	275.546	285.849	266.612
	0.9	373.372	296.281	298.929	300.176

Table 4.4: Simulation results: averaged squared losses. Asymptotically optimal model averaging and cAIC smoothed weights

Table 4.4 presents the calculated squared losses for the respective methods for each underlying

covariance matrix and the residuals standard deviation. The model averaging approach presented and implemented in this work proves to be the superior method in terms of minimum average squared loss in all scenarios presented here.

4.6 Applications

In this section, we apply the proposed weight finding technique for model averaging on models fitted to two different real-world datasets. The first one is about the sensory assessment of TV characteristics. The second one is a common linear mixed model benchmarking dataset from an orthodontic study over time for several subjects.

4.6.1 Bang & Olufsen dataset

The first dataset was provided by the Danish electronics company Bang & Olufsen. Different characteristics of TV sets are measured using three response variables. The explanatory variables given are the TV set and image quality as measured and recorded by a panel of eight different assessors. See Kuznetsova et al. (2017) for the details of this study.

In the first application, we are interested in the influence of the explanatory variables on the response variable of the sharpness of motion and model it by means of random effects on the different assessors. To model the relationship we create three different linear mixed effects models, assuming that the response variable is influenced by the fixed effects of the TV set and the image quality, as well as an interaction of both. However, the three models differ in how the effects of the assessors are incorporated into the model. In the first model, it is assumed that there is a simple fixed random effect per assessor, in the second model a fixed random effect per assessor with an interaction effect between the TVs and the assessor is assumed and in the third, we assume an interaction between the assessor and the image quality. To compare the models, we calculate information criteria, the relative degrees of freedom and the mean squared error of the respective models. See Table 4.5 for the results.

If one decides to use the classical model selection approach of choosing the model with the smallest possible value for the information criterion under consideration, as well as a model averaging based on smooth weights, the second model with the assumed random effect relationship of an interaction between assessor and TV set would be chosen or respectively would receive a weight of one.

				Weights	
Model	rel. DF	cAIC	MSE	smoothed	asymptotic
1	30.47	864.88	3.806	0.000	0.327
2	36.48	844.10	3.188	1.000	0.673
3	19.61	864.18	4.213	0.000	0.000

Table 4.5: The values of various model choice criteria, smoothed weights based on the cAIC and weights resulting of the proposed model averaging estimator

Figure 4.6 shows the development of the weights and their trajectories during the optimization process. It can be observed that from the outset the weights quickly converge towards their final estimates. More specifically, from the tenth iteration onwards, the change of the weights becomes negligible. It should be noted that the starting values for all weights are set to be $w_i = 1/K$, K being the number of candidate models. By doing so, it is ensured that all weights are already within the feasible region of the optimization problem. Thereby, the equality constraints do not need to be explicitly enforced on the starting values and the algorithm can directly start minimizing the underlying weight choice function.

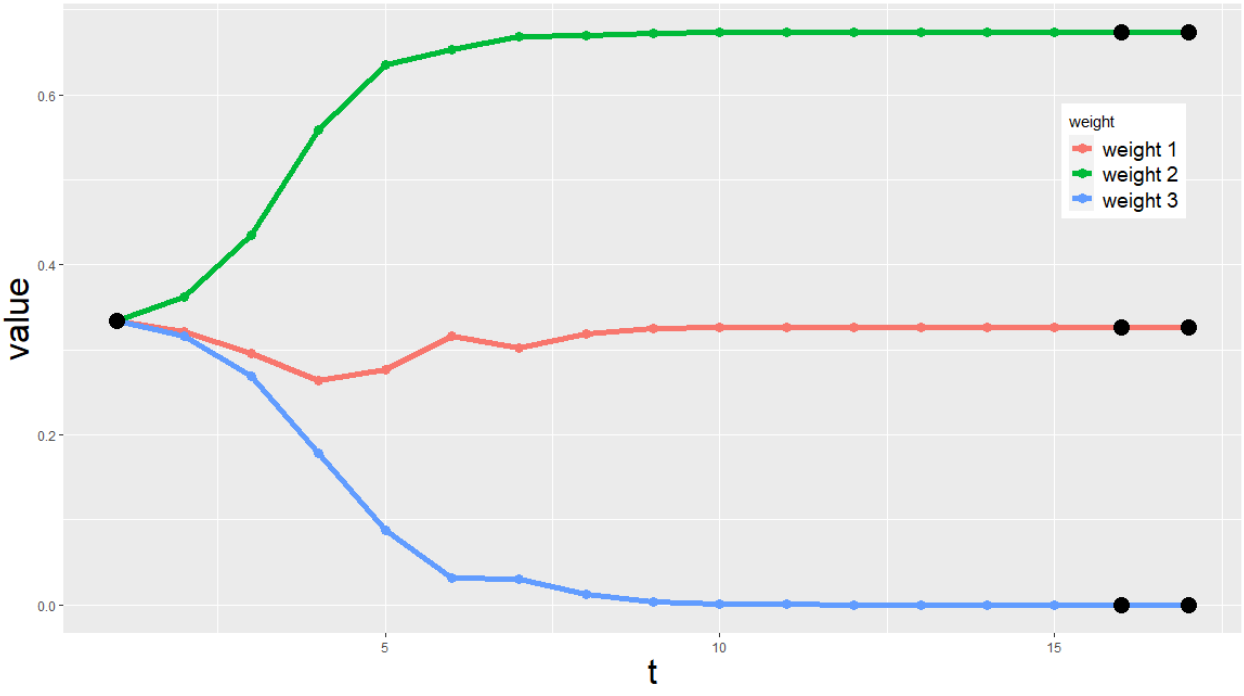

Figure 4.6: Representation of the trajectories of the weights during the optimisation. Large black dots indicate weight values resulting from major iteration, small coloured dots indicate the results of a minor iteration.

The augmented Lagrangian is constructed as a robust method with regard to the starting values, see chapter 17.4 in Nocedal and Wright (2006). If the starting values are misspecified and as such chosen that they are not within the feasible region the convergence often takes longer. The authors however did not find any convergence problems leading to wrong solutions in the simulations and applications due to misspecified starting values.

We compare the results of our proposed method based on the mean squared error, with the results of a model averaging estimator based smoothed weights, as well as a model averaging estimator build upon the assumption of equal weights. The mean squared error is calculated by

$$MSE = \frac{1}{n}(\boldsymbol{y} - \hat{\boldsymbol{y}}(\hat{\boldsymbol{w}}))^T(\boldsymbol{y} - \hat{\boldsymbol{y}}(\hat{\boldsymbol{w}})),$$

where $\boldsymbol{y}$ represents the responses, $\hat{\boldsymbol{y}}(\hat{\boldsymbol{w}})$ the model averaging estimator based on the calculated weights and n the number of observations. The model average estimator based on the proposed method achieves the smallest MSE value of all with 3.153, whereas the estimator based on equal weighting and on conditional AIC based smoothed weights offer MSE values of 3.491 and 3.188, respectively.

4.6.2 Orthodont dataset

The second application uses candidate models based on the well known Orthodont dataset. The dataset stems from a study at the University of North Carolina Dental School following the growth of 27 children from the age of 8 until age 14. Every second year, the distance between the pituitary and the pterygomaxillary fissure was measured via X-ray examination. For more details see Potthoff and Roy (1964). For this application, we fit three different models. The first models the measured distance with the help of an intercept and the fixed effect of age, as well as a random intercept per individual. The second model extends the first model by including the fixed effect of gender. The third model introduces an additional fixed effect by including an interaction between age and gender. Model defining quantities such as the conditional Akaike information criterion, the relative degrees of freedom and the mean squared error of the respective models can be observed in Table 4.6. In addition, the table includes the calculated weights of our proposed method in addition to the smoothed weights based on the conditional AIC values of each candidate model.

The resulting estimator based on the proposed method provides the lowest overall MSE with an value of 1.462 of all model averaging estimators considered. The equal weights estimator has an

				Weights	
Model	rel. DF	cAIC	MSE	smoothed	asymptotic
1	27.12	405.47	1.463	0.910	0.839
2	25.92	411.23	1.582	0.051	0.000
3	26.57	411.79	1.569	0.039	0.161

Table 4.6: The values of various model choice criteria, smoothed weights based on the cAIC and weights resulting of the proposed model averaging estimator

MSE value of 1.509 and the smoothed weights estimator has an MSE value of 1.463. This indicates a better ability of our methodology to evaluate, weigh and merge the underlying candidate models into a new model averaging estimator.

4.6.3 Computational aspects and suitability for applied users

To assess the computational requirements, all required calculations were executed a thousand times for both applications and the performance of the presented algorithm was contrasted with existent optimization routines in `R`. While the presented algorithm needs 10.764 milliseconds for the TVbo model set and 9.676 milliseconds on average for the Orthodont models, the general nonlinear optimiser `solnp` of the `Rsolnp` package (Ghalanos and Theussl, 2015) need 12.654 and 11.706 milliseconds. The nonlinear optimiser with constraints `constrOptim.nl` of the `alabama` package (Varadhan, 2015) requires an average of 35.18 milliseconds for the first application and 7.194 milliseconds for the second application. In both applications, the method presented has either the least required computing time or one that is close to the fast method. In contrast to `solnp` and `constrOptim.nl`, the proposed algorithm does not require the user to provide any complex input of starting weights or the underlying gradient. Furthermore, in comparison to the much more general implementations of nonlinear optimizers, all necessary quantities and objects are automatically created and calculated in the background. This in turn allows the implementation to be more straightforward and convenient to use for researchers willing to employ the proposed approach for determining asymptotic optimal weights for model averaging of linear mixed models.

4.7 Outlook

Model choice for the class of linear mixed models plays an important role due to their wide distribution and application in different fields. Especially the question of including random effects plays

a crucial part, which is complicated by the inherent problem of classical model choice methods concerning the underlying model assumptions. Thus, the use of classical information criteria such as AIC is discouraged due to the deviation from the classical model by the assumptions of the linear mixed model, while the usefulness of other methods such as likelihood-ratio test based approaches is impaired by the possibility of boundary issues.

Therefore, due to its nature of combining different candidate models, the technique of model averaging presents an interesting alternative to model selection of linear mixed models. On a technical level the choice of weights is critical for model averaging. As we have shown the proposed weight finding method by Zhang et al. (2014) of using the Steinian approximation of derivatives for an underlying weight criterion shows superior performance when compared to other approaches based on information criteria such as the conditional AIC. The proposed method is implemented as part of the R-Package `cAIC4` facilitating the use by applied researchers. Given that there is no universally applicable unbiased estimator of conditional AIC in analytical form without distributional assumptions (Saefken et al., 2014), the proposed method stops short of offering an all model-class encompassing solution for model averaging.

Such a generalisation would be especially valuable as further interesting models such as robust linear mixed models would fall under such an extended framework, see Lin and Lee (2006) and Ho and Lin (2010). Therefore misspecifications could be identified as in Bartolucci et al. (2017). This requires further research that we are planning to conduct in the future. An implementation of a criterion for finding a squared loss-optimal weights for generalised linear mixed models is another extension that is still required. A possible approach could be to use the methods proposed in Wood et al. (2016) for conditional model selection. A further possible extension could be to apply another error function than the squared error proposed in Zhang et al. (2014). Different possible error functions and the corresponding covariance penalties are presented in Säfken and Kneib (2020). This could be especially interesting for distributional regression models, see Kneib et al. (2021). Also an extention to boosting (Griesbach et al., 2021) would be interesting.

In terms of fields of applications, the proposed framework offers great potential for model averaging for applied researchers in order to offer more robust predictive capacity. One avenue which will be pursued by the authors of this paper is the use of model averaging in the context of epidemiological research along the lines of Silbersdorff et al. (2018).

Part II

Measuring complexity of deep learning models

On measuring complexity of deep learning models: A covariance penalty approach

Contributing article:

Kruse, R.M., Säfken, B., and Kneib, T. (2023). Measuring Neural Complexity: A Covariance Penalty Approach.
Under review at the *26th European Conference on Artificial Intelligence.*

Author contributions:

René-Marcel Kruse wrote the manuscript, including the simulation studies and implementations of the presented methods. Benjamin Säfken and Thomas Kneib added valuable input, suggested several notable modifications and proofread the manuscript.

Abstract:

We are witnessing the spectacular success of predictive deep learning algorithms in science and industry. However, the deep learning framework lacks a mature, rigorous, mathematically based approach to model evaluation and selection. For the development and application of possible decision regimes, the specifics of the variety of deep learning algorithms need to be considered. This paper addresses the question of how to quantify the underlying complexity of neural networks and how to utilise the knowledge gained to evaluate and select models. We introduce the concept of covariance penalties as a means of measuring the complexity of deep learning models and illustrate the theoretical and practical challenges of translating this concept to the domain of deep learning. Furthermore, we present the proposed methodologies with different simulation studies to demonstrate the validity of the given approaches and try to identify factors that drive model complexity for deep learning models.

On measuring complexity of deep learning models: A covariance penalty approach

5.1 Introduction

From classification to regression to language processing, deep learning methods have many potential applications and are shaping up to be one of the most important tools for data analysis and processing in the 21st century. In addition to the early widespread applications in image recognition and processing (Lecun et al., 1998; He et al., 2016), the later well-known examples such as playing games and eventually defeating world-class players in chess or Go (Silver et al., 2016, 2018), or the current advances in large-scale language models transforming the way we interact with information and the web (OpenAI, 2023), deep learning methods are becoming more widely used in scientific applications. From cancer tumour detection (Karabatak and Ince, 2009; Esteva et al., 2017) to drug development (Ma et al., 2015), deep learning methods are already widely used in medicine. However, these techniques are also very popular in the social sciences, with applications such as text analysis of conspiracy theories on Twitter (Kant et al., 2022) or the prediction of economic variables such as stock prices (Thormann et al., 2021). But only recently has attention focused on an integral part of scientific applications: model selection, interpretation and especially model evaluation.One of the focal points of the current research is the attempt to achieve a better explainability of the deep learning models and the associated evaluation of the models. One such approach towards more explainable deep learning is to merge statistical techniques with deep learning concepts to achieve greater explainability and transparency, like deep regression (Rügamer et al., 2021), NAM (Agarwal et al., 2021) or NAMLSS (Thielmann et al., 2023). An alternative to model selection and the handling of the associated uncertainty is the application of ensemble deep learning, in which the same model is trained several times and then an average is formed over all models (Kook et al., 2022). When applying methods as ensemble learning, however, the question arises as to which criteria

are used to select, or how to weight the underlying models and finally compare them. A selection according to pure loss performance leaves out many important characteristics of the models and neglects, especially in science, important aspects such as robustness and replicability. In classical statistical data analysis, model complexity plays an important role in model selection, but has only been superficially considered in the context of deep learning model evaluation. A common concept to quantify complexity are degrees of freedom, which are often defined as the number of parameters of the model, however, this is only valid for the simplest models and is more generally defined as the trace of the hat or projection matrix of the model (Zhang et al., 2014). More complex and non-linear models deviate from this simplified view and highlight the importance of a more mathematically stringent definition. A broader and more rigorous mathematical definition of the concept of degrees of freedom can be derived via Stein's lemma as the expected sensitivity of the estimated or predicted mean values with respect to the underlying response observations (Stein, 1981; Efron, 1986). Following Stein, Efron (2004) formulated degrees of freedom, and thus model complexity, in terms of covariance penalties. This definition is difficult to formulate analytically, especially for non-linear complex procedures, but it provides a broader understanding of model complexity and a more comprehensive approach. Ye (1998) proposes a numerical approximation of the degrees of freedom via a perturbation-based approach, which circumvents the need for closed form. Gao and Jojic (2016) were able to show that Ye's form of defining and computing degrees of freedom can be applied to deep learning models for classification and, under certain conditions, produce reliable results. Hauenstein et al. (2016) extend Ye's concept of degrees of freedom to approximate complexity of machines and deep learning models, and use the resulting values to compute model evaluation metrics such as AIC (Akaike, 1973). While the methods proposed have been shown to be effective for simple model architectures, they reach their limits when applied to deeper and more complex models and yield less reliable results due to the inherent inherently stochastic nature of neural networks.

In this paper, we define and illustrate algorithms to compute the covariance penalties of deep learning models and show how model architecture and regulation techniques influence the underlying model complexity. Our algorithms require additional runs of the modelling procedure, but allow a more targeted model choice and allow users to consider the issue of complexity and hence sparsity of a model. In our applications and experiments, we try to answer the following questions:

1. How do architecture and regularisation techniques affect model complexity? And, in particular, how do the various proposed methods differ in their results?

2. Are there any identifiable factors driving the complexity of models?

5.2 Measuring complexity of deep learning models

In this section we discuss the general concept of degrees of freedom, covariance penalties and their relation to prediction error estimation. We also introduce a generalisation of error measures, the Q-class of prediction errors formulated by Bregman (1967) and Efron (1986), as well as Efron's optimism theorem.

5.2.1 Definitions

Regardless of the multitude of possible architectures and applications, the underpinning mathematical-statistical theory of deep learning algorithms can be generalized. Assume a given problem with data of the form of $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$ with an given joint-probability $p(\boldsymbol{x}, \boldsymbol{y})$. Further assume that we have n observed data points as $\boldsymbol{d} = (z^{(i)})_{i=1}^m = ((x_i, y_i))_{i=1}^m \in \mathcal{Z}^m$ of the underlying true data-generated process

$$\boldsymbol{y} = \phi(\boldsymbol{x}) + \boldsymbol{\epsilon},$$

where $\phi(\cdot) : \mathbb{R} \to \mathbb{R}$, $\boldsymbol{y} \in \mathcal{Y} \subseteq \mathbb{R}$, $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^m$ and $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ for all $i = 1, \ldots, n$. The task is to identify a model that performs well on the training data $\boldsymbol{d}$ and that also performs well on unknown out-of-sample data. In order to find such a model, assume that $\mathcal{Z}, \mathcal{X}$ and $\mathcal{Y}$ are known and measurable. Moreover, let there be a loss function over all measurable functions of $\mathcal{X}, \mathcal{Y}$ as $\mathcal{L} : \mathcal{M}(\mathcal{X}, \mathcal{Y}) \times \mathcal{Z} \to \mathbb{R}$. Based on a subset of $\mathcal{M}(\mathcal{X}, \mathcal{Y})$ a hypothesis set is now selected to create a mapping

$$\mathcal{A} : \bigcup_{m \in \mathbb{N}} \mathcal{Z}^m \to \mathcal{F},$$

which uses the given data to find a model $m_d = \mathcal{A}(\boldsymbol{d}) \in \mathcal{F}$, that will also exhibit a strong performance on unknown data. Where the hypothesis set $\mathcal{F}$ represents all possible realisations of the neural network, given the respective assumed underlying architecture. For a more in-depth rigorous mathematical definition of neural networks see Berner et al. (2021).

5.2.2 Prediction error measurements and covariance penalties

Having selected and trained a model $m_{\boldsymbol{d}}$, we now want to measure the performance of the model on new unknown test data. However, the error of a given prediction $\hat{\boldsymbol{y}} = m_d(\boldsymbol{x})$ can be measured

in several ways depending on the underlying data and task, thus creating the need for a more generalised approach. A more general approach to working with different error functions ia a generalisation of a broad class of error measures introduced by Bregman (1967) and Efron (1986) respectively. Using a concave function $q(\cdot)$, a large class of error measures, called the Q-class of error measures or also known as the Q-class Bregman divergence, can be constructed via

$$Q(\boldsymbol{y}, \hat{\boldsymbol{y}}) = q(\hat{\boldsymbol{y}}) + q'(\hat{\boldsymbol{y}})(\boldsymbol{y} - \hat{\boldsymbol{y}}) - q(\boldsymbol{y}). \tag{5.12}$$

This class includes many different error measures, with the squared error and the AIC as the most prominent representatives (Säfken and Kneib, 2020).

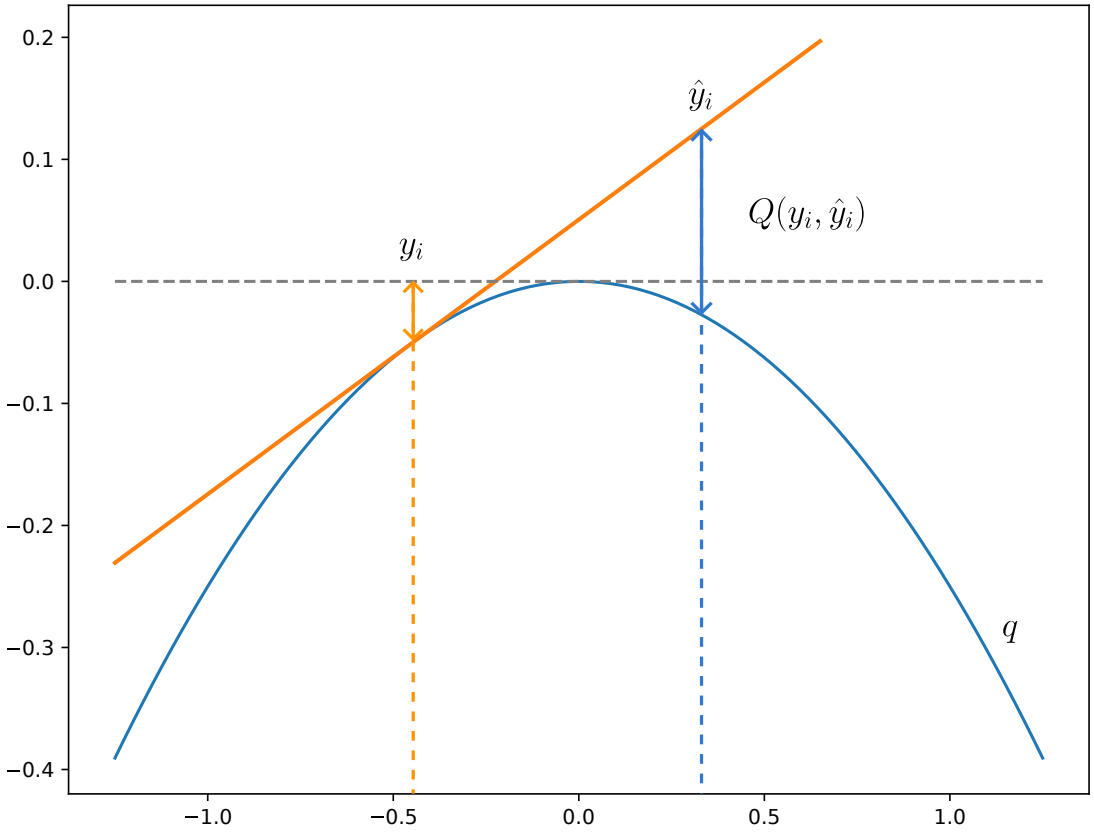

Figure 5.7: **Graphical depiction of the Q-class as defined in** (5.12)**:** The curve represents the concave q function. The dashed lines indicate the positions of y_i and $\hat{y}_i$. The difference between the tangent $q(\hat{y}_i) + q'(\hat{y}_i)(y_i - \hat{y}_i)$ and $q(\hat{y}_i)$ represents the error $Q(y_i, \hat{y}_i)$.

From the Q-class follows a notation of the total error equal to the sum of all error components such that

$$Q(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_{i=1}^{n} Q(y_i, \hat{y}_i).$$

However, the presented sum leads to a too-optimistic error estimator, such that the observed error does not equal the underlying true error. This is a problem for the evaluation of a model in terms of its theoretical performance on new, unknown data. Thus, rather than focusing on the overall error, we are focusing on the expectation of the overall predictive error, with respect to new, unknown data from the same data generation process as the original data.

The optimism theorem proposed by Efron (1983) establishes a connection between observed prediction error and the expected prediction error and allows for the calculation of the expectation of the total error. If one considers the observed error and the true underlying error of an arbitrary prediction rule, i.e. $\hat{\boldsymbol{y}} = m(\boldsymbol{x})$ the total error for the i-th component of an independent copy of $\boldsymbol{y}$, denoted by y_i^0, based on the aforementioned theorem can be formalted in quadratic loss example as follows

$$\mathbb{E}\left[Q(y_i^0, \hat{\boldsymbol{y}})\right] = \mathbb{E}\left[Q(y_i, \hat{y}_i) + 2 \cdot cov(y_i, \hat{y}_i)\right].$$

The covariance part of the equation can be seen as an adjustment correcting for the biases of the apparent error, hence the name of the covariance penalty. It should be noted, however, that $cov(\hat{y}_i, y_i)$ cannot be observed directly and must in most cases be estimated.

The first proposal to approximate the covariance term was made by Stein (1981) for the simplified case of Gaussian distributed data, where it was shown that

$$1/\sigma^2 \sum_{i=1}^{n} cov(\hat{y}_i, y_i) = \mathbb{E}\left(\sum_{i=1}^{n} \partial \hat{y}_i / \partial y_i\right), \tag{5.13}$$

illustrating the link between the covariance penalties and the expected optimism (Efron, 2004). Since the derivatives in (5.13) are observable, it is possible to construct an estimator for the total prediction error as follows

$$\widehat{Q}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = Q(y_i, \hat{y}_i) + 2\sigma^2 \sum_{i=1}^{n} \frac{\partial \hat{y}_i}{\partial y_i}, \tag{5.14}$$

which has since become known as Stein's Unbiased Risk Estimation (Efron, 2004). Moreover, it can be seen in Efron (1986, 2021) that the estimated covariance turns out to be further a good asymptotic approximator of the underlying model complexity.

5.3 Complexity of deep learning models

Covariance penalties, are in themselves not observable statistics and require estimation. Various parametric and non-parametric methodologies have been proposed, three of these, which are particularly suited to the application of more complex models and applications such as deep learning networks, will be discussed in the following section.

5.3.1 Direct Approximation of Complexity of Neural Networks

For deep learning models, as for many other models, the assumed derivatives $\sum_i \partial\hat{y}_i/\partial y_i$ of the predictions in (5.13) with respect to the underlying inputs are not available in closed form.

A direct approximation of model complexity as presented in Ye (1998) proves difficult for complex deep learning models. Thus, Ye proposed a simple linear regression to repeated perturbations of the underlying data of the form $(\hat{y_i}^* - \hat{y_i}) = \beta_0 + \beta_1(y_i^* - y_i)$ and summing the slope coefficients over all data points to arrive at the approximated derivatives and thus the estimated model complexity. This method, also known as the horizontal method, was proven in more detail by Elder (2003) as a more robust method compared to the direct calculation of differences, but was found to be less suitable for non-linear modelling methods (Hauenstein et al., 2016). On the other hand, Ramani et al. (2008) were able to show in a theoretical context that there exists a stochastic direct estimator of the model complexity of any nonlinear model of the form

$$\sum_i \frac{\partial f(\boldsymbol{y})}{\partial y_i} = \lim_{\epsilon \to 0} \mathbb{E}_b \left[b^T \left(\frac{f(\boldsymbol{y} + \epsilon \boldsymbol{b}) - f(\boldsymbol{y})}{\epsilon} \right) \right], \tag{5.15}$$

with $f(\boldsymbol{y})$ begin an approximate function used on response data $\boldsymbol{y}$, b a zero mean random vector with unit variance and ϵ is a small perturbation value. This demonstrates the possibility of approximating the derivatives of non-linear models by perturbing the input and thus, in the simplest form, allowing methods such as finite differences or even more sophisticated Monte Carlo methods, as proposed by Gao and Jojic (2016) and Ramani et al. (2008), to be used to approximate the underlying derivatives and hence the underlying covariance penalties. An important question that arises, however, is how to choose the number of perturbations that are introduced in each of the iterations.

While it is possible to change one point at a time, two problems arise. Firstly, the methods would require $(n+1)$ computations to obtain the result, which seems disproportionate for large data sets and complex models such as those in deep learning. Second, due to the stochastic nature of

the models and their training, the expected values are variable, which can lead to the change being much smaller than the perturbation itself, and thus much smaller or even negative values. This has been observed previously with various machine learning algorithms, such as random forests (Gao and Jojic, 2016; Hauenstein et al., 2016).

In fact, the direct approximation methodology proposed in this paper makes use of data-driven perturbations to estimate the underlying complexity. The introduced perturbations consist of a fixed small value h, equal to 10^{-6} in our applications, scaled in each iteration by $\epsilon(\sigma_y^2)$ randomly drawn from a normal distribution $\mathcal{N}(0, \epsilon \cdot \sigma_y^2)$. The factor ϵ in our simulations is $1/4$ and σ_y^2 represents the variance of the underlying response variable. Each computation is performed T times to account for the stochastic nature of the deep learning models, averaging out stochastic initialisation effects. The process continues until each point in the data set has been perturbed once. A summarising illustration of the method can be seen in Algorithm 2. To minimise the inherent stochasticity of the deep learning model, all models are initialised with the same random seeds, using the same batch number, stopping criteria and learning rates. Similar to ensemble deep learning, each calculation is repeated several times to account for the inherent randomness of the models.

Algorithm 2 Perturbation-based approximation

Input: Training data $\boldsymbol{X}, \boldsymbol{y}$, number of passes p, number of perturbation k

Output: Estimated covariance penalty $\widehat{cov}$ via approximated $\partial f(y_i)/\partial y_i$

for p times **do**
  while length(unperturbed $\boldsymbol{y}$) > 0 **do**
    Sample k values of the still unperturbed $\boldsymbol{y}$
    Perturb the corresponding response values and create $\boldsymbol{y}^*$
    Train model $\boldsymbol{X}$ and new $\boldsymbol{y}^*$
    Predict new $\hat{\boldsymbol{y}}^*$ based on $\boldsymbol{X}$
  end while
  Use horizontal method $(\hat{y_i}^* - \hat{y_i}) = \beta_0 + \beta_1(y_i^* - y_i)$
  Sum of slope coefficients $\widehat{cov}_p = \sum_i \beta_{1i}$
end for
Calculate mean over passes $\widehat{cov} = 1/p \sum_p \widehat{cov}_p$
return $\widehat{cov}$ =0

In the case of continuous responses, the form of the underlying perturbation is a matter of adding the perturbation term to the responses in $\boldsymbol{y}$. The difficulty with non continuous responses, such with a classification task whether binary or multi-class, is that we are dealing with fixed integer or boolean values, and thus the form of the perturbation must be adapted to the underlying data and the task of the model. In the simplest case of a binary problem, perturbation occurs by changing the target variable from True to False, 0 to 1, or the given encoding, and vice versa.

5.3.2 Bootstrap estimation of covariance penalties of deep learning models

While the direct approximation of the covariance penalties relies on the underlying derivatives, which can be sampled directly from the underlying observations in $\boldsymbol{y}$ by evaluating the associated Jacobian, this approach inherently suffers from drawbacks. One approach is to formulate a closed-form expression of the Jacobian, which is generally difficult and would need to be done for each new application. The numerical evaluation of the Jacobian in the case of large dimensions, such as in neural network applications, proves to be extraordinarily difficult, such that even if it were possible, it may lead to severe numerical instabilities. An alternative to estimating the covariance penalty directly is a bootstrap approach (Efron, 2004). This involves generating a bootstrap distribution based on the original model and its fitted or predicted values $\hat{\boldsymbol{y}}$ for the mean prediction and an estimator for the variance $\hat{\sigma}^2$. Assuming normally distributed data, this gives a distribution $\hat{f} = \mathcal{N}(\hat{\mu}, \hat{\sigma}^2)$ from which B new values for the response y_i^{*b} are calculated for the underlying values of the inputs in x_i . Using the new bootstrap response values and the original model $m()$, new predictions are obtained in the form of $\hat{y}_i^{*b} = m(y_i^{*b})$. Following these results, the covariance penalties are estimated from the observed bootstrap covariance via

$$\widehat{cov}_i = \sum_{b=1}^{B} \hat{\boldsymbol{y}}_i^{*b} \left(\boldsymbol{y}_i^{*b} - \bar{\boldsymbol{y}}_i^{*} \right) / (B-1) \text{ with } \bar{\boldsymbol{y}}_i^{*} = \sum_{b} \frac{\hat{\boldsymbol{y}}_i^{*b}}{B}.$$

For this type of parametric bootstrap, Borra and Di Ciaccio (2010) have shown that it may be necessary to adapt the procedure for non-parametric complex models such as deep learning models. The main problem here is that deep and machine learning models can fit noise and thus produce mean predictions equal to the response variable and a variance of $\hat{\sigma}^2 = 0$. In the light of such findings, the approach originally proposed by Efron can prove to be impractical in some applications. Thus, the authors propose to introduce an optional step of generating first preliminary estimates for $\boldsymbol{\mu}$ and σ^2 based on the underlying data to avoid potential drawbacks. An important

point of comparison with the presented approaches is that the bootstrap method represents a so-called global method, as in each simulation step all underlying data points are altered to compute the covariance penalty, contrasting direct perturbation and cross-validation based approaches. Like the direct approximation of the derivatives, the bootstrap is a model-based approach, i.e. it assumes that the model under consideration is true.

5.3.3 Cross-validation estimation of covariance penalties of deep learning models

Cross-validation is a widely used error prediction estimation method that automatically incorporates the underlying model complexity to measure the predictive performance of a model. The advantage of cross-validation over other approaches is model independence. Equally to the direct approximation approach based on the Steinian, only the i-th value is changed, such that this approach can be called a local method (Efron, 2004). Following the Q-class error notation presented earlier in (5.12), it is possible to formulate the covariance penalty as an estimate of the cross-validation. Given the data set $\boldsymbol{y}_{-i} = (y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n)$ reduced by the i-th data point and the corresponding mean estimator $\hat{\boldsymbol{y}}_{-i}$ utilising Efron's optimism theorem as in (5.13) allows to construct the connection between the expected prediction error of the cross-validation $Q(y_i, \hat{y}_{-i})$ and the apparent error $Q(y_i, \hat{y}_i)$ as

$$2\widehat{cov}_i = Q(y_i, \hat{y}_{-i}) - Q(y_i, \hat{y}_i).$$

This relationship allows the establishment of the underlying covariance penalty estimator, based on cross-validation, as follows

$$\sum_{i=1}^{n} \widehat{cov}_i = \frac{1}{2} \sum_{i=1}^{n} \Big[Q(y_i, \hat{y}_{-i}) - Q(y_i, \hat{y}_i) \Big].$$

Rather interestingly, it can be shown that the covariance penalties are a Rao-Blackwellization of the cross-validation approach (Efron, 2004). This suggests at least theoretically less accurate results of cross-validation compared to the direct approximation and bootstrap methods presented. An additional concern with estimating prediction error, and thus covariance penalty, via cross-validation is the high variance of the resulting estimates, the resulting unstable and thus less reliable estimates (Stone, 1977; Efron, 1983). A possible solution to this issue is to adopt K-fold cross-validation, which decreases the overall variance of the estimator but introduces a larger bias, such that K-fold cross-validation will never leads to an unbiased estimator (Hastie et al., 2009; Rosset

and Tibshirani, 2020). However, Burman (1989) demonstrated that there exists a version of the apparent error estimator, corrected for K-fold cross-validation, which depends on the number of folds K. Giving an indexing function $\kappa : \{1, \ldots, N\} \mapsto \{1, \ldots, K\}$ which indicates the partitioning into disjoint subsets after randomising the sample, while $\hat{y}_i^{-k}$ denotes the fitted or predicted values computed with the kth part of the data removed. The result of the K-fold error estimate can be described as follows

$$Q\left(y_i, \hat{y}_i^{-\kappa(i)}\right) = \frac{1}{K}\sum_{h=1}^{K}\frac{1}{m}\sum_{j\in\kappa(h)} Q\left(y_j, \hat{y}_j^{-\kappa(h)}\right) + \frac{1}{n}\sum_{i=1}^{n}\left[Q\left(y_i, \hat{y}_i\right) - \frac{1}{K}\sum_{h=1}^{K} Q\left(y_i, \hat{y}_i^{-\kappa(h)}\right)\right].$$

This allows the estimator of Efron's optimism for a K-fold CV approach to be rewritten as

$$\begin{aligned} 2\sum_{i=1}^{n}\widehat{cov}_i &= \frac{1}{K}\sum_{h=1}^{K}\left[\frac{n-m}{nm}\sum_{j\in\kappa(h)} Q\left(y_j, \hat{y}_j^{-\kappa(h)}\right) - \frac{m}{mn}\sum_{j\notin\kappa(h)} Q\left(y_i, \hat{y}_j^{-\kappa(h)}\right)\right] \\ &= \frac{n-m}{n}\left[\frac{1}{K}\sum_{h=1}^{K}\frac{1}{m}\sum_{j\in\kappa(h)} Q\left(y_i, \hat{y}_j^{-\kappa(h)}\right) - \frac{1}{K}\sum_{h}\frac{1}{n-m}\sum_{j\notin\kappa(h)} Q\left(y_i, \hat{y}_j^{-\kappa(h)}\right)\right]. \end{aligned}$$

This corrected version uses the mean function of the test sets, corrected for the K-fold training set fit, to estimate the corrected optimism (Borra and Di Ciaccio, 2010). Consequently, the methodology used relies on a K-fold cross-validation approach involving multiple repetitions to account for the inherent stochastic influences of initialisation, the deep learning model itself, and the K-fold sampling procedure. The method used here, like the direct approximation of derivatives and the bootstrap, uses multiple runs with re-initialisation of the underlying models to better address the inherent stochasticity of the deep learning approach.

5.4 Applications

We investigate the influence of different architectural characteristics of deep learning models, such as depth or number of neurons per layer, on model complexity, and the influence of regulatory effects, such as weight decay or dropout, on model complexity using the proposed methods. Another area of focus is the extent to which deep learning models are self-regulating through their inherent mechanics, and to what extent this determines or limits the overall complexity of the model.

5.4.1 Influence of model architecture on complexity

In order to assess the influence of the architecture on complexity, various deep learning models with different depths and widths are examined. In order to cover the different application scenarios, the simulations are carried out once for a continuous and once for a discrete response setting, representing a regression and a classification based task. In the case of continuous response variable, tabular data is generated from a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \sigma)$. The data generating model for μ are considered here with $x_i \sim \mathcal{U}(0, 1)$.

$$\boldsymbol{\mu} = \beta_0 + \beta_1 x_1 + \beta_2 x_2^2 + \beta_3 x_3^2 + \beta_4 x_1 \cdot x_4 + \beta_5 x_5^2$$

For the classification set-up, data for a binary response variable is simulated using ten feature variables, half of which are redundant. The two response categories are unequally weighted, with the first category with a prevalence of 20% and the second class at 80%. This inequality, together with the presence of redundant features, is intended to simulate the necessary more complex underlying characteristics of the data in the model training process, and thus the possible impact of the data on model complexity. The models considered vary in depth of hidden layers of $\in [0, 1, 2, 3]$ and the number of hidden units for each layer in the regression set up with $\in [5, 10, 15, 20]$. For the sake of comparability, all hidden layers have the same number of hidden units over the entire network. The deep learning models under consideration are trained via stochastic gradient descent with a maximal number of epochs of 1.000 and a patience of 100. This relatively long training period and patience is required to ensure convergence among the models, as even minor changes in the models can produce noticeable effects on the results for perturbation-based methods, therefore highlighting another drawback of this approach. We did not attempt to optimise the model settings for each data set, as we are not interested in developing the best overall model. Although the models used here are far removed from the parameter-rich architectures of modern applications, the lessons learned from the various models reviewed here can well illustrate the impact of architectural differences. Possible regularising effects of an optimiser will be looked at in the second simulation study.

Figure 5.8 illustrates the relationship between the total number of parameters of the deep learning models trained on the synthetic data sets presented and the calculated covariance penalty values. Note that the estimated covariance penalty values increase rapidly with the number of parameters. However, the calculated complexity values reach a plateau early and stay relatively constant for the continuous response application, whereas the estimated complexity for the classification setting

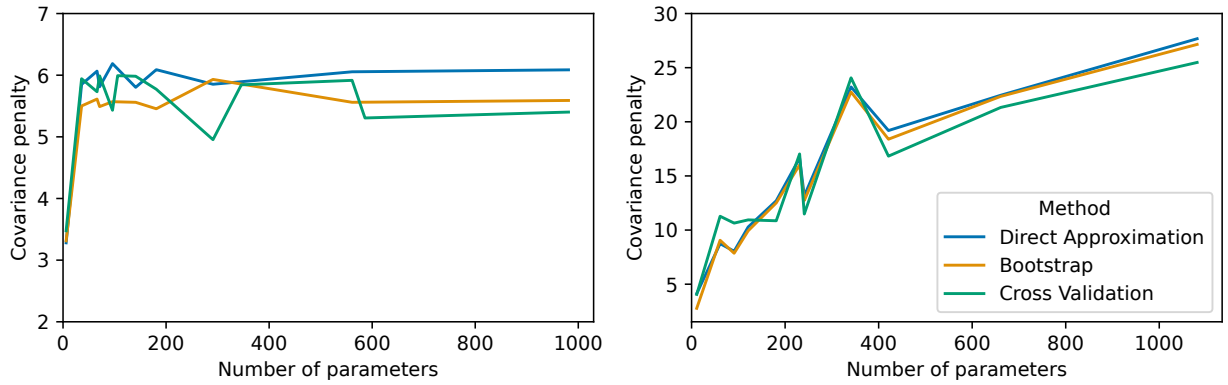

Figure 5.8: **Influence of number of parameters:** Graphs for the complexity of the models in terms of the number of parameters. We observe a rapid increase in complexity with few model parameters, but quickly reach a plateau for the continuous response. Although the calculated complexity increases with classification, there is more variation, implying that there are more factors driving complexity. A correlation as seen with simple linear models between model parameter and complexity are not present in the results.

seems to indicate that an increase in model parameter increases the complexity to some degree. The graph combines different models with different depths and widths, such that the variations, especially in the classification setting, suggest that there are several driving factors. A comparison of the three methods presented suggests, in line with the literature, that K-fold cross-validation produces a higher variance in its results whereas bootstrap and the direct approximator produce more constant results (Efron, 2004; Borra and Di Ciaccio, 2010).

In the regression set-up, we have relatively simple data structures, which in combination with the graphs suggests that the complexity of a neural network is governed by its self-regulating properties. As long as a deep learning model has sufficient parameters to map the data, such that increasing the number of parameters does not lead to an improvement in model performance, increasing the number of parameters does not increase model complexity. The influence of hidden layers and hidden units per layer can be seen in Figure 5.9.

Similar to the number of model parameters, there is a sharp increase in the estimated covariance penalty values and a plateau. Although the computed complexity increases with classification, there is more variation, suggesting that there are more factors driving complexity than the number of total parameters. A breakdown of the calculated complexities according to the respective depth of the models considered can be seen in Figure 5.10.

In the more complex classification setup, we can observe that the deeper models tend to have a higher level of complexity, indicating the role of depth in model complexity. Interestingly, again,

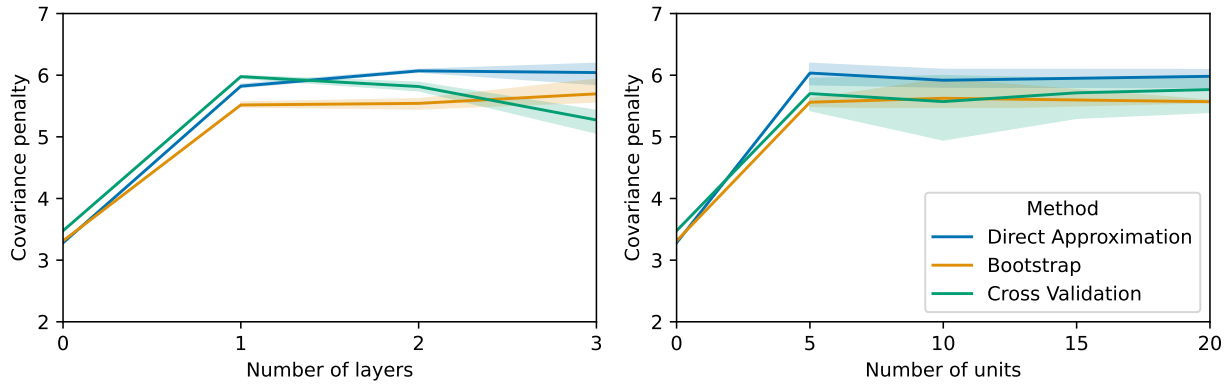

Figure 5.9: **Influence of architecture:** Graphs for the complexity of the models in terms of the number of hidden layers and units per hidden layer. We can see an increase in model complexity, but this plateaued relatively early. No single aspect seems to soley drive model complexity in this simulation study.

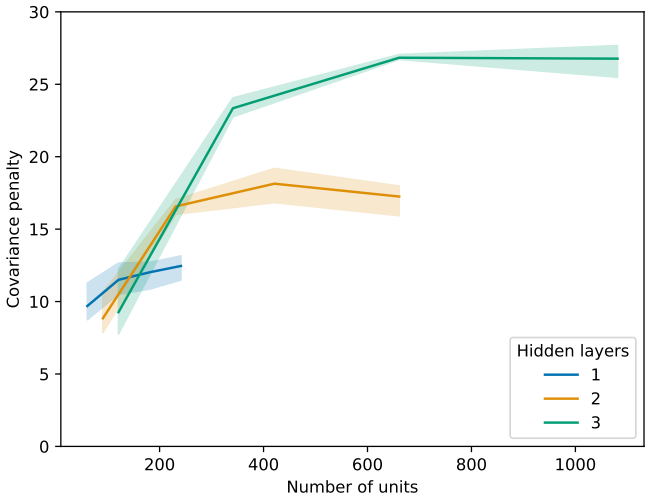

Figure 5.10: **Influence of number of parameters:** Graphs for the complexity of the models in terms of the number of parameters given the number of underlying hidden layers. Complexity increases rapidly with few model parameters, but soon reaches a plateau. A correlation as seen with simple linear models is not present in the result. However, a relationship between model depth and overall model complexity becomes apparent.

the increase in complexity is only partly due to an increase in model parameters. Instead, the depth of the model is an important factor influencing complexity. Also with this approach, as with the simple data set, we can observe that models eventually reach an equilibrium where increasing the number of model parameters does not lead to an increase in complexity. Based on the initial findings from the first data set and simulation, this suggests that the models eventually reach their

maximum complexity, supporting the hypothesis that the self-regulation of neural networks leads to a neural network that is as complex as it needs to be, and no more, in order to approximate the underlying data sufficiently well. The observation is consistent with the theoretical property that neural networks with at least two layers are universal approximators (Cybenko, 1989; Berner et al., 2021). Once this state has been reached, it follows that more hidden layers or more neurons per layer will not improve performance and that even if we increase the number of hidden layers or neurons per layer, the underlying model performance and complexity will not increase.

5.4.2 Regularisation impact on complexity

The ability of the neural network to learn from the given data is not only influenced by the architecture itself, i.e. the depth, width and activation functions, but regularisation techniques also play a major role. A demonstrable influence of the regularisation on underlying degrees of freedom and thus the accompanying complexity of the models can be shown for various statistical models (Tibshirani and Taylor, 2012). For the purpose of this analysis, we use the same setup as presented in the previous chapter. However, we train the respective models with different regularising influences. Here we focus on weight decay and dropout.

Weight decay itself represents a form of penalty, similar to approaches such as Ridge (Hoerl and Kennard, 1970) or Lasso regression (Santosa and Symes, 1986; Tibshirani, 1996). Here, the L^2-norm of all weights w of the model under consideration is calculated and added to the loss. The impact on the loss can be controlled via a decay parameter. Dropout is a regularisation technique used to reduce overfitting of artificial neural networks by randomly dropping hidden units from the neural network during training. Dropout has been shown to have a positive effect on the performance of neural networks (Srivastava et al., 2014).

To evaluate the influence of the regularisation methods, we perform repeated simulations to analyse the impact of dropout and weight decay on the model complexity, where for the former we add dropout layers to the model architecture and for the latter we apply weight decay with different decay rates and estimate the underlying covariance penalties for them. Here we take the data set for a continuous response variable from the first simulation study and train a simple two hidden layer MLP with ten neurons per layer on the data. To investigate the effect of dropout, we run the training for models with different values of dropout in the range $[0, 0.1, \ldots, 0.9]$. For the effect of weight decay, we use the same data and the same baseline model, but instead introduce weight decay in training and let the weight decay take different values with $[0, 10^{-7}, 10^{-6}, ..., 10^{-2}]$.

The effect of dropout and weight decay on the resulting model complexity is illustrated in Figure 5.11. Both dropout and weight decay affect the value of the estimated covariance penalty values.

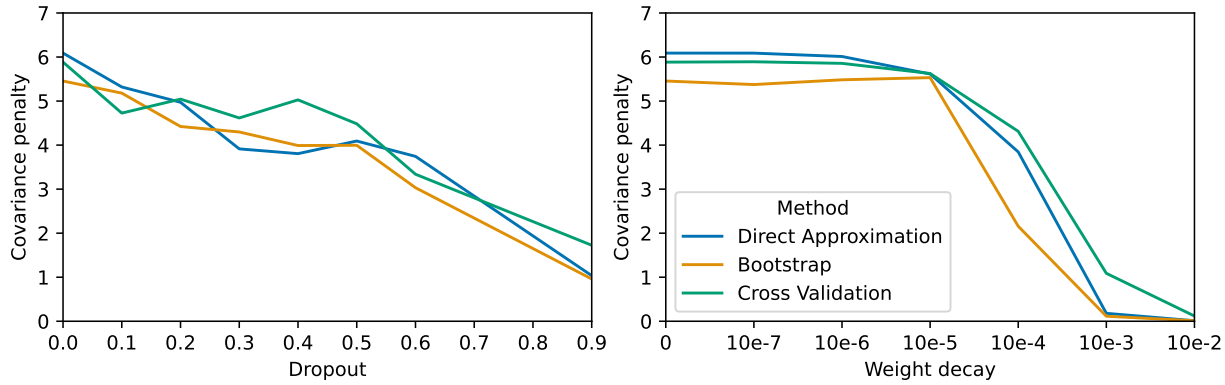

Figure 5.11: **Influence of regularisation:** The plots display the calculated values of covariance penalties on synthetic data given different dropout and weight decay rates as regularisation techniques. The results of the perturbation, cross-validation and bootstrap methods are depicted in the three different coloured graphs.

It is worth noting that the result for the influence of dropout for our synthetic data contradicts the results of Gao and Jojic (2016), who in their paper based on the concept of generalised degrees of freedom as complexity measures for classification deep learning models Ye (1998) show that dropout does not affect model complexity with synthetic data. The dropout leads to a constant decrease in the covariance penalty values as the dropout value increases. In terms of progression over the dropouts, the bootstrap and the perturbation-based results are very close to each other. The K-fold cross-validation results show a similar progression, but higher variance in the results, as expected from a cross-validation approach. The weight decay, on the other hand, shows a non-constant influence on the estimated model complexity. This influence of weight decay is consistent with experience with statistical models, where it can be shown in statistics that weight decay leads to a reduction in overall model complexity. The results of all three methods follow each other closely, with the bootstrap method tending to yield the lowest values and the perturbation-based method the largest. As in the first simulation study, the K-fold approach produces a higher variance in the results.

5.5 Discussion and conclusion

Attempts at complexity measures for deep learning models, such as the generalised degrees of freedom popular in statistics, have produced promising initial results, but have been limited in their approach to classification, as in the work of Gao and Jojic (2016), or have demonstrated that perturbation-based approaches reaches its limits due to the architecture's inherent processes of weighted averaging, as in the work of Hauenstein et al. (2016). We were able to establish that the concepts of covariance penalties as a measure of model complexity can be transferred from statistics to deep learning and can be implemented effectively. In different simulations, we have been able to show how the different methods relate to one another, but also how different features of deep learning models, such as architecture or regularisation techniques, influence the underlying model complexity. Specifically, the observation that models tend to reach a model complexity limit that, that once reached, cannot be further increased by additional model parameters. This finding suggests that deep learning models may only become as complex as they need to be due to their inherent ability to self-regulate, thus maintaining a level of self-controlled sparsity.

The comparison of the different methods presented provides an interesting insight into our covariance penalty approaches, showing that cross-validation tends to produce values for the upper bounds of complexity with higher complexity, while bootstrap produces lower bounds and the direct perturbation-based approach produces results in between. An inherent disadvantage of the methods presented here shows up in varying degrees in the additional computational effort that is introduced. In particular, the perturbation-based computation of the Steinian-like complexity estimator requires up to a total of (n + 1) additional runs, depending on the number of perturbations per iteration. In comparison, cross-validation and bootstrap based approaches are less computationally intensive, but still have a significant impact on the computation time. In all our applications, regardless of the method, we were able to show that there is no clear complexity-driving factor such as the number of model parameters, the number of neurons per layer or the depth. Overall, the results show that cross-validation and bootstrap are the more appropriate methods for estimating complexity, both methods require less computation than the direct perturbation-based approximation approach and yield more reliable results over all presented use cases.

Part III

Neural additive models for location, scale, and shape

Neural additive models for location, scale, and shape: A framework for interpretable neural regression beyond the mean

Contributing article:

Thielmann, A., Kruse, R. M., Kneib, T., Säfken, B. (2023). Neural Additive Models for Location Scale and Shape: A Framework for Interpretable Neural Regression Beyond the Mean. *arXiv preprint arXiv:2301.11862*.

Under review at the *Thirty-seventh Conference on Neural Information Processing Systems*.

Author contributions:

This manuscript was written by Anton Thielmann and René-Marcel Kruse. Both authors share the role of main contributors. Contributions for the paper and simulations were divided by sections. Benjamin Säfken and Thomas Kneib added valuable input, suggested notable modifications and proofread the manuscript.

Abstract:

Deep neural networks (DNNs) have proven to be highly effective in a variety of tasks, making them the go-to method for problems requiring high-level predictive power. Despite this success, the inner workings of DNNs are often not transparent, making them difficult to interpret or understand. This lack of interpretability has led to increased research on inherently interpretable neural networks in recent years. Models such as Neural Additive Models (NAMs) achieve visual interpretability through the combination of classical statistical methods with DNNs. However, these approaches only concentrate on mean response predictions, leaving out other properties of the response distribution of the underlying data. We propose Neural Additive Models for Location Scale and Shape (NAMLSS), a modelling framework that combines the predictive power of classical deep learning models with the inherent advantages of distributional regression while maintaining the interpretability of additive models.

Neural additive models for location, scale, and shape: A framework for interpretable neural regression beyond the mean

6.1 Introduction

Deep learning models have shown impressive performances on a variety of predictive tasks. They are state-of-the-art models for tasks involving unstructured data, such as image classification (Yu et al., 2022; Dosovitskiy et al., 2020), text classification (Huang et al., 2021; Lin et al., 2021), audio classification (Nagrani et al., 2021), time-series forecasting (Zhou et al., 2022; Zeng et al., 2022) and many more. However, the predictive performance comes not only at the price of computational demands. The black-box nature of deep neural networks poses hard challenges for interpretability. To achieve sample-level interpretability, existing methods resort to model-agnostic methods. Locally Interpretable Model Explanations (LIME) (Ribeiro et al., 2016) or Shapley values (Shapley, 1953) and their extensions (Sundararajan and Najmi, 2020) try to explain model predictions via local approximation and feature importance. Sensitivity-based approaches (Horel and Giesecke, 2020), exploiting significance statistics, can only be applied to single-layer feed-forward neural networks and can hence not be used to model difficult non-linear effects, requiring more complex model structures.

Subsequently, high-risk domains, such as e.g. medical applications often cannot exploit the advantages of complex neural networks due to their lack of innate interpretability. The creation of these innately interpretable models hence remains an important challenge. Achieving the interpretability from flexible statistical models as e.g. Generalized Linear Models (GLMs) (Nelder and Wedderburn, 1972) or Generalized Additive Models (GAMs) (Hastie, 2017), in deep neural networks, however, is inherently difficult. Recently, Agarwal et al. (2021) introduced Neural Ad-

ditive Models (NAMs), a framework that models all features individually and thus creates visual interpretability of the single features. While this is an important step towards interpretable deep neural networks, any insightfulness of aspects beyond the mean is lost in the model structure. To counter that, we propose the neural counterpart to Generalized Additive Models for Location, Scale and Shape (GAMLSS) (Rigby and Stasinopoulos, 2005), the Neural Additive Model for Location, Scale and Shape (NAMLSS). NAMLSS adopts and iterates on the model class of GAMLSS, in the same scope as NAMs (Agarwal et al., 2021) on GAMs.

The GAMLSS framework relaxes the exponential family assumption and replaces it with a general distribution family. The systematic part of the model is expanded to allow not only the mean (location) but all the parameters of the conditional distribution of the dependent variable to be modelled as additive nonparametric functions of the features, resulting in the following model notation:

$$\theta^{(k)} = {g^{(k)}}^{-1}\left(\beta^{(k)} + \sum_{j=1}^{J_k} f_j^{(k)}(x_j^{(k)})\right) = \eta_{\theta^{(k)}},$$

with the superscript $k = 1, \ldots, K$ denoting the k-th parameter and $j = 1, \ldots, J$ denoting the features.

The model assumes that the underlying response observations y_i for $i = 1, 2, \ldots, n$ are conditionally independent given the covariates. The assumed conditional density can depend on up to K different distributional parameters[3]. Each of these distribution parameters $\theta^{(k)}$ can be modelled using its additive predictor $\eta_{\theta^{(k)}}$ for $k = 1, \ldots, K$, allowing for complex relationships between the response and predictor variables, as well as the flexibility to choose different distributions for different parts of the response variable. An additional important component of the GAMLSS model is the link function $g^{(k)}(\cdot)$, which allows each parameter of the distribution vector to be conditional on different sets of covariates. In the case that the distribution under consideration features only one distribution parameter, the model simplifies to an ordinary GAM model. Therefore, GAMLSS is to be seen as a conceptual extension of the GAM idea and is suitable for the extension and generalisation of approaches such as NAMs which are themselves built upon the GAM idea. For an overview of the current state of regression models that focus on the full response distribution approaches see Kneib et al. (2021).

While NAMs learn linear combinations of different input features to learn arbitrary complex functions and at the same time provide improved interpretability, these models, like their statistical

[3]In practice most application focus on up to four $\boldsymbol{\theta}_i = \left(\theta_i^{(1)}, \theta_i^{(2)}, \theta_i^{(3)}, \theta_i^{(4)}\right)$.

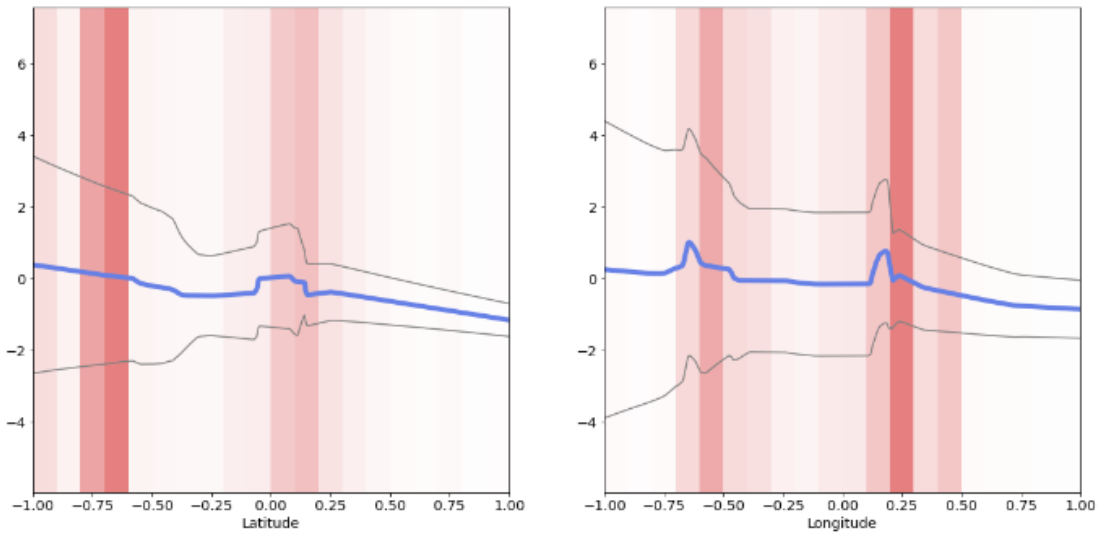

Figure 6.12: **California Housing:** Graphs for longitude and latitude respectively learned by the NAMLSS model. NAMLSS capture changes in mean as well as variance. Therefore the plotted standard deviations change in dependence of the longitude and latitude. The house price jumps around the location of Los Angeles are depictable. Additionally, we find a decrease in variance for areas further away from the large cities.

counterparts GAMs, focus exclusively on modelling mean and dispersion. This is in contrast to the GAMLSS and subsequently, the proposed NAMLSS, which substantially broadens the scope by allowing all underlying parameters of the response distribution to potentially depend on the information of the covariates.

Contributions The contributions of the paper hence can be summarized as follows:

- We present a novel architecture for Neural Additive Models for Location, Scale and Shape.
- Compared to state-of-the-art GAM, GAMLSS and DNNs our NAMLSS achieves similar results on benchmark datasets.
- We demonstrate that NAMLSS effectively captures the information underlying the data. Especially NAMLSS allows for prediction beyond point estimates, for instance prediction intervals.
- Lastly, we show that the NAMLSS approach allows to go beyond the mean prediction of the response and to model the entire response distribution.

6.2 Literature review

The idea of generating feature-level interpretability in deep neural networks by translating GAMs into a neural framework was already introduced by Potts (1999) and expanded by de Waal and du Toit (2007). While the framework was remarkably parameter-sparse, it did not use backpropagation and hence did not achieve as good predictive results as GAMs, while remaining less interpretable. More recently, Agarwal et al. (2021) introduced NAMs, a more flexible approach than the Generalized Additive Neural Networks (GANNs) introduced by de Waal and du Toit (2007) that leverages the recent advances in the field of Deep Learning.

NAMs are a class of flexible and powerful machine learning models that combine the strengths of neural networks and GAMs. These models can be used to model complex, non-linear relationships between response and predictor variables, and can be applied to a wide range of tasks including regression, classification, and time series forecasting. The basic structure of a NAM consists of a sum of multiple components, each representing a different aspect of the relationship between the response and predictor variables. These components can be linear, non-linear, or a combination of both, and can be learned using a variety of optimization algorithms. One of the key advantages of NAMs is their inherent ability to learn the interactions between different predictor variables and the response without the need for manual feature engineering. This allows NAMs to capture complex relationships in the data that may not be easily apparent to the human eye.

The general form of a NAM can be written as:

$$\mathbb{E}(y) = h\left(\beta + \sum_{j=1}^{J} f_j(x_j)\right), \tag{6.16}$$

where $h(\cdot)$ is the activation function used in the output layer, $x \in \mathbb{R}^j$ are the input features, β is the global intercept term, and $f_j : \mathbb{R} \to \mathbb{R}$ represents the Multi-Layer Perceptrons (MLPs) corresponding to the j-th feature. The similarity to GAMs is apparent, as the two frameworks mostly distinguish in the form the individual features are modelled. $h(\cdot)$ is comparable to the link function $g(\cdot)$.

Several extensions to the NAM framework have already been introduced. Pairwise or higher order interaction effects can be accounted for (Yang et al., 2021; Enouen and Liu, 2022; Wang et al., 2021; Dubey et al., 2022). Chang et al. (2021) introduced NODE-GAM, a differentiable model based on forgetful decision trees developed for high-risk domains. All these models follow the additive

framework from GAMs and learn the nonlinear additive features with separate networks, one for each feature or feature interaction, either leveraging MLPs (Potts, 1999; de Waal and du Toit, 2007; Agarwal et al., 2021; Yang et al., 2021; Radenovic et al., 2022), using decision trees (Chang et al., 2021) or using Splines (Rügamer et al., 2020; Seifert et al., 2022; Luber et al., 2023).

The applications of such models range from nowcasting (Jo and Kim, 2022), financial applications (Chen and Ye, 2022), to survival analysis (Peroni et al., 2022). While the linear combination of neural subnetworks provides a visual interpretation of the results, any interpretability beyond the feature-level representation of the model predictions is lost in their black-box subnetworks.

6.3 Beyond the mean

Obviously, the mean (or the arithmetic mean as its empirical counterpart) provides only a rather incomplete description of a probability distribution (the empirical distribution of corresponding observations, in case of the arithmetic mean). While this fact is widely acknowledged when it comes to exploratory data analysis, it is also widely ignored in the context of prediction models where the focus is typically on predicting expected outcomes. This narrow focus reflects an interest in common or average observations, but is misleading when phenomena such as risk, extremes, or uncertainty are central to an analysis. With the GAMLSS-based framework considered in this paper, we are able to quantify effects of covariates not only on the mean, but on any parameter of a potentially complex distribution assumed for the responses. As major advantage, the resulting models can determine changes in all aspects of the response distribution, such as variance, skewness or tail probabilities. This also contributes to properly disentangling aleatoric from epistemic uncertainty.

Changing the focus from regression models for the mean to regression for distributions also requires changes in the evaluation metric that is used to compare rivalling model specifications. More precisely, the evaluation metric should be proper Gneiting and Raftery (2007), i.e. enforce the analyst to report their true beliefs in terms of a predictive distribution. While the MSE that is commonly employed in mean-based modelling is proper for the mean, it is not for general distributions. We therefore will rely on the negative log-likelihood (also refereed to as the log-score) as a proper score for comparing distributional regression models.

While predicting all parameters from a distribution may not always improve predictive power, understanding the underlying data distribution is crucial in high-risk domains and can provide valuable insights about feature effects. As an example, Figure 6.13 illustrates the fit of our approach

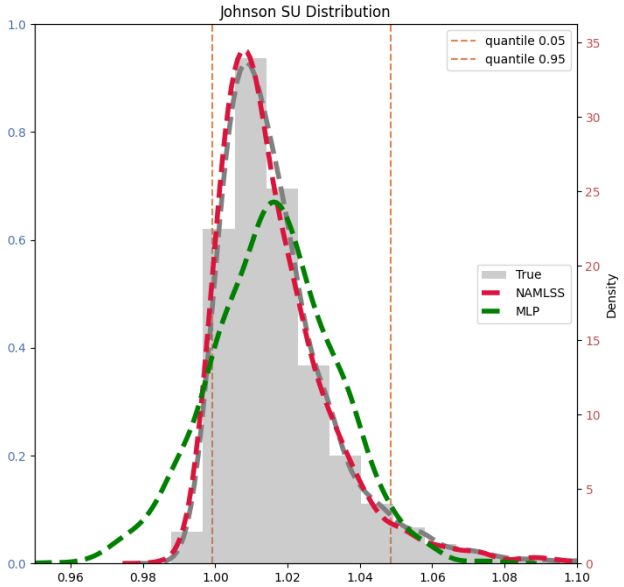

Figure 6.13: **Johnson's S_U distribution:** Simulated Johnson's S_U distribution and the fit of a simple NAMLSS (see Figure 6.14) and a MLP. While the MLP achieves an impressive fit concerning the quadratic loss, it clearly cannot capture the underlying distribution adequately.

on data following a Johnson's S_U distribution, including 3 features, compared to the fit of a MLP that minimizes the Mean Squared Error (MSE). The MLP has a better predictive performance with an MSE of 0.0002, however, NAMLSS is able to reflect the underlying data distribution much more accurately (as shown in Figure 6.13), even though it has an MSE of 0.0005. The idea of focusing on more than the underlying mean prediction is thus certainly relevant and has been an important part, especially of the statistical literature in recent years. There has been a strong focus on the GAMLSS (Rigby and Stasinopoulos, 2005) framework, conditional transformation models (Hothorn et al., 2014), density regression (Wang et al., 1996) or quantile and expectile regression frameworks. However, these methods are inferior to machine and deep learning techniques in terms of pure predictive power; the disadvantage of not being able to deal with unstructured data forms such as images, text or audio files; or the inherent problems of statistical models in dealing with extremely large and complex data sets. One resulting development to deal with these drawbacks is frameworks that utilize statistical modelling methods and combine them with machine learning techniques such as boosting to create new types of distributional regression models such as boosted generalized additive model for location, scale and shape as presented by Hofner et al. (2014). However, the models leveraging boosting techniques, while successfully modelling all distributional parameters, lack the inherent interpretability from GAMLSS or even the visual interpretability from NAMs.

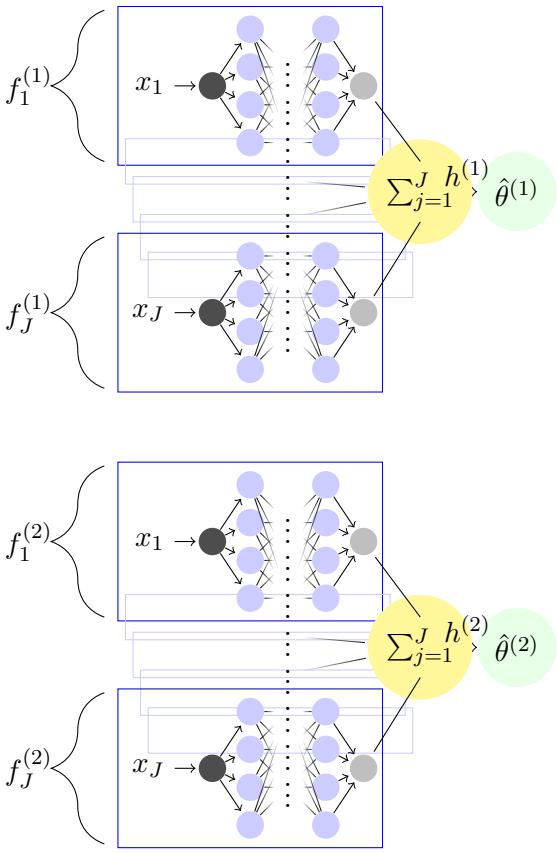

Figure 6.14: The network structure of a simple NAMLSS model. Each input variable as well as each distributional parameter is handled by a different neural network. $h^{(k)}$ are different activation functions depending on the distributional parameter that is modelled. E.g. a quadratic transformation for modelling the variance in a normally distributed variable to ensure the non-negativity constraint. The presented structure demonstrates a NAMLSS modelling a distribution with two parameters, e.g. a normal distribution.

6.4 Methodology

While NAMs incorporate some feature-level interpretability and hence entail easy interpretability of the estimated regression effects, they are unable to capture skewness, heteroskedasticity or kurtosis in the underlying data distribution due to their focus on mean prediction. Therefore, the presented method is the neural counterpart to GAMLSS, offering the flexibility and predictive performance of neural networks while maintaining feature-level interpretability and which allows estimation of the underlying total response distribution.

Let $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$ be the training dataset of size n. Each input $x = (x_1, x_2, \ldots, x_J)$ contains J features. y denotes the target variable and can be arbitrarily distributed. NAMLSS are trained by minimizing the negative log-likelihood as the loss function, $-\log(\mathcal{L}(\theta|y))$ by optimally approximating the distributional parameters, $\theta^{(k)}$.

Each parameter, $\theta^{(k)}$, is defined as:

$$\theta^{(k)} = h^{(k)}\left(\beta^{(k)} + \sum_{j=1}^{J} f_j^{(k)}(x_j)\right), \tag{6.17}$$

where $h^{(k)}(\cdot)$ denotes the output layer activation functions dependent on the underlying distributional parameter, $\beta^{(k)}$ denotes the parameter-specific intercept and $f_j^{(k)} : \mathbb{R} \to \mathbb{R}$ represents the feature network for parameter k for the j-th feature, subsequently called the *parameter-feature network*.

Just as in GAMLSS, $\theta^{(k)}$ can be derived from a subset of the J features, however, due to the inherent flexibility of the neural networks, defining each $\theta^{(k)}$ over all J is sufficient, as the individual feature importance for each parameter, $\theta^{(k)}$, is learned automatically. Each parameter-feature network, $f_j^{(k)}$, can be regularized employing regular dropout coefficients in conjunction with feature dropout coefficients, $\lambda_{1j}^{(k)}$ and $\lambda_{2j}^{(k)}$ respectively, as also implemented by Agarwal et al. (2021). For e.g. a normal distribution, NAMLSS would hence minimize

$$-\log\left(\mathcal{L}(\hat{\mu}, \hat{\sigma}^2|y)\right) = -\left(-\frac{n}{2}\log(2\pi\hat{\sigma}^2) - \frac{1}{2\hat{\sigma}^2}\sum_{i=1}^{n}(y_i - \hat{\mu})^2\right), \tag{6.18}$$

where

$$\hat{\mu} = \beta^{(1)} + \sum_{j=1}^{J} f_j^{(1)}(x_j) \quad \text{and} \quad \hat{\sigma}^2 = \log\left(1 + \exp\left[\beta^{(2)} + \sum_{j=1}^{J} f_j^{(2)}(x_j)\right]\right),$$

utilizing a Softplus activation function for the scale parameter and a linear activation for the location parameter.

Integrating possible feature interactions can easily be achieved in both architectures, either following Wang et al. (2021) by training a fully connected MLP on the residuals after the NAMLSS has converged, or by following Enouen and Liu (2022); Radenovic et al. (2022) and modelling pairwise (or higher order) feature interactions for all distributional parameters:

$$\theta^{(k)} = h^{(k)}\left(\beta^{(k)} + \sum_{j=1}^{J} f_j^{(k)}(x_j) + \sum_{j,t:j\neq t}^{J} f_{jt}^{(k)}(x_j, x_t)\right). \tag{6.19}$$

We propose two different network architectures that can both flexibly model all distributional parameters. The first is depicted in Figure 6.14 and creates J subnetworks for each of the K distributional parameters. Each distributional subnetwork is comprised of the sum of the parameter-feature networks $f_j^{(k)}$. Hence we create $K \times J$ parameter-feature networks. To account for distributional

restrictions, each distributional subnetwork is specified with possibly differing activation functions in the output layer. The second model architecture, possible due to the flexibility of neural networks, leverages the architecture of NAMs (see Formula (6.16)) and is depicted in Figure B.1 in the Appendix. Here, only J subnetworks are created, with each subnetwork having a K-dimensional output layer. This architecture thus creates the same number of subnetworks as a common NAM. Each distributional parameter, $\theta^{(k)}$, is subsequently obtained by summing over the k-th output of the J subnetworks. Every dimension in the output layer can be activated using different activation functions, according to parameter restrictions. This allows the capture of interaction effects between the given model parameters in each of the subnetworks[4]. Equation 6.17 would only slightly be adjusted, to account for the subnetwork f_j now mapping to $\mathbb{R}^k$, $f_j : \mathbb{R} \to \mathbb{R}^k$:

$$\theta^{(k)} = h^{(k)} \left(\beta^{(k)} + \sum_{j=1}^{J} f_j(x_j)\,[:, k] \right), \tag{6.20}$$

with $[:, k]$ denoting an index and representing the $k - th$ index of $f_j : \mathbb{R} \to \mathbb{R}^k$. Note, that the superscript $^{(k)}$ is missing from the subnetwork f_j, as only J subnetworks are trained.

6.5 Benchmarking

To demonstrate the competitiveness of the presented method, we perform several analyses. First, we compare NAMLSS with the most common statistical distributional regression approach GAMLSS (Stasinopoulos et al., 2000).

Synthetic data comparison study The synthetic data used for this task is generated from the same underlying processes. Five features are included in each application. The data-generating functions used to generate the true underlying distributional parameters can be found in Appendix B.1.5. Each of the five input vectors x_j is sampled from a uniform distribution $\mathcal{U}(0, 1)$, with a total of $n = 3000$ observations per data set. The remaining parameters are generated based on the input vectors and the chosen distribution. We selected distributions that are widely used, popular in science, or relatively complex to reflect a diverse range of scenarios. We compare models that specifically model all distributional parameters in this simulation study. The results can be found in Table 6.7.

[4]Note, that for distributions where only one parameter is modelled, the two proposed NAMLSS structures are identical.

Table 6.7: **Results for synthetic data:** We compare NAMLSS with the baseline of additive distributional models, GAMLSS.

Distribution	GAMLSS	NAMLSS	Gain (%)
	Neg Log-Likelihood ↓		
Binomial	397	274	**30%**
Poisson	800	802	**-0.25%**
Normal	600	589	**1.83%**
Inv. Gaussian	385	377	**2.1%**
Weibull	625	621	**0.64%**
Johnson's S_U	370	326	**11.9%**
Gamma	426	410	**3.8 %**
Logistic	731	682	**7.2 %**
Average			**7.18 %**

We find that the presented NAMLSS outperforms GAMLSS for all distributions except the Poisson distribution. This can be attributed to the fact that the Poisson distribution only involves a single distributional parameter.

Experiments with real world data We compare the performance of NAMLSS with several state-of-the-art models including neural as well as non-neural approaches and orientate on the benchmarks performed by Agarwal et al. (2021). Additionally, we compare related methods of distribution-focused data analysis approaches that overcome the focus on relating the conditional mean of the response to features and instead target the complete conditional response distribution. We choose the following baselines for the comparisons:

- **Multi-layer perceptron (MLP)**: Unrestricted fully connected deep neural network trained with either a mean squared error loss function (regression) or binary cross entropy (logistic regression).

- **Gradient boosted trees (XGBoost)**: Decision tree based gradient boosting. We use the implementation provided by Chen and Guestrin (2016).

- **Neural additive models (NAMs):** Linear combination of DNNs as described in equation (6.16) and presented by Agarwal et al. (2021).

- **Explainable boosting machines (EBMs)**: State-of-the-art generalized additive models leveraging shallow boosted trees (Nori et al., 2019).

- **Neural generalized additive model (NodeGAM)**: State-of-the-art generalized additive models leveraging neural oblivious decision trees (Chang et al., 2021).

- **Deep distributional neural network (DDNN)**: Similar to the multi-layer perceptron a fully connected neural network. However, not trained to minimize the previously mentioned loss functions but to minimize the negative log-likelihood of the specified distribution. All distributional parameters are predicted.

- **Generalized additive models for location scale and shape (GAMLSS)**: Standard GAMLSS models using the R implementation from Rigby and Stasinopoulos (2005).

- **gamboost for location scale and shape (gamboostLSS)**: Fitting GAMLSS by employing boosting techniques as proposed by Hofner et al. (2014).

We preprocess all used datasets exactly as done by Agarwal et al. (2021). We perform 5-fold cross-validation for all datasets and report the average performances over all folds as well as the standard deviations. For reproducability, we have only chosen publicly available datasets. The datasets, as well as the preprocessing and the seeds set for obtaining the folds, are described in detail in the Appendix, B.1.4.

We fit all models without an intercept and explicitly do not model feature interaction effects.

Table 6.8: Average Rank

Model	Avg. Rank
MLP	7.4
XGBoost	8.2
NAM	7.4
EBM	6.5
NodeGAM	7.8
DDNN	4.0
GAMLSS	4.5
gamboostLSS	4.2
NAMLSS1	1.9
NAMLSS2	**1.6**

For datasets following a Gaussian distribution we use the California Housing (CA Housing) dataset (Pace and Barry, 1997) from sklearn (Pedregosa et al., 2011), the Insurance dataset Lantz (2019), the Abalone dataset (Dua and Graff, 2017) and standard normalize the response variables. Thus, a normal distribution $\mathcal{N}(\mu, \sigma^2\mathcal{I})$ of the underlying response variables is assumed. As the (negative) log-likelihood of a normal distribution (see equation (B.1.1)) is dependent on two parameters, but models as an MLP or XGBoost only predict a single parameter, we adjust the computation accordingly and use the standard deviation calculated from the underlying data for XGBoost, EBM, NAM and MLP. For a (binary) classification benchmark we use the FICO dataset (FICO, 2018), the Shrutime dataset and the Telco dataset. A logistic distribution, $\mathcal{LO}(\mu, s)$, of

the underlying response variable was assumed (see equation (B.1.1) for the log-likelihood). Again, we use the true standard deviation of the underlying data for the models only resulting in a mean prediction. For the Melbourne and Munich datasets, also analyzed by Rügamer et al. (2020), we assume an Inverse Gamma distribution $\mathcal{IG}(\alpha, \beta)$ as the underlying data distribution (see equation (B.1.1) for the log-likelihood)[5].

Table 6.9: **Benchmark results for normal and inverse-gamma datasets:** For models not explicitly modelling a shape parameter, the shape is approximated with a constant as the true standard deviation of the dependent variable. Lower negative log-likelihoods (ℓ) are better. We report results on five commonly used datasets. The California housing dataset for predicting house prices (Pace and Barry, 1997), an Insurance dataset for predicting billed medical expenses (Lantz, 2019), the Abalone dataset for predicting number of rings in trees (Dua and Graff, 2017) and two AirBnb datasets.

	Negative log-likelihood ℓ (↓)				
		Normal		**Inv. Gamma**	
Model	CA Housing	Insurance	Abalone	Munich	Melbourne
MLP	4191 ±(42)	266.8 ± (11)	966.2 ±(27)	6827 ± (178)	22999 ± (232)
XGBoost	4219 ±(40)	266.8 ± (9)	982.0 ±(33)	5618 ± (152)	20471 ± (242)
NAM	4251 ±(43)	474.7 ± (73)	956.8 ±(22)	5892 ± (37)	25375 ± (844)
EBM	4202 ±(42)	263.8 ± (10)	965.1 ±(22)	5474 ± (56)	20361 ± (207)
NodeGAM	4206 ±(89)	279.1 ± (11)	958.3 ±(23)	5984 ± (135)	21896 ± (261)
DDNN	2681 ±(1279)	178.2 ± (30)	897.2 ±(159)	5555 ± (34)	20790 ± (29)
GAMLSS	3512 ±(67)	175.5 ± (28)	870.8 ±(16)	5419 ± (61)	26353 ± (45)
gamboostLSS	3812 ±(52)	173.0 ± (28)	815.1 ± (29)	5421 ± (33)	26436 ± (48)
NAMLSS[1]	2667 ± (91)	172.7 ± (23)	869.8 ±(118)	**5383** ± (24)	**19517** ± (68)
NAMLSS[2]	**2329** ± (176)	**172.6** ± (20)	**802.3** ±(41)	5422 ± (22)	19675 ± (67)

[1] With $J \times K$ subnetworks. See Table 6.14 for an exemplary network structure.

[2] With J subnetworks and each subnetwork returning a parameter for the location and shape respectively.
See Table B.1 for an exemplary network structure.

The NAMLSS approach achieves the lowest negative log-likelihood values for all of the datasets which speaks for its good approximation capabilities. One of the advantages of NAMLSS compared to DNNs is the feature level interpretability. Similar to NAMs, we can plot and visually analyze the results (see Figures 6.15 and 6.12). Additionally, we are capable of accurately representing sharp price jumps around the location of San Francisco, depicted by the jumps in the graphs for longitude and latitude (see Figure 6.12) as compared to GAMLSS, NAMLSS are additionally capable of representing jagged shape functions.

[5] See Appendix for further details on activation functions.

Table 6.10: **Benchmark results for Logistic and categorical datasets** For models not explicitly modelling a shape parameter, the shape is approximated with a constant as the true standard deviation of the dependent variable. Lower negative log-likelihoods (ℓ) are better. We report results on several commonly used datasets. FICO dataset (FICO, 2018)

	Negative log-likelihood ℓ (↓)		
		Logistic	
Model	FICO	Shrutime	Telco
MLP	1813 ±(6)	1240 ± (22)	1027 ± (19)
XGBoost	1976 ±(13)	1314 ± (18)	1123 ± (22)
NAM	1809 ± (8)	1247 ± (26)	1023 ± (28)
EBM	1944 ±(21)	1290 ± (26)	1094 ± (22)
NodeGAM	1942 ±(21)	1308 ± (29)	1097 ± (27)
DDNN	1230 ± (48)	-211 ± (364)	27 ± (314)
GAMLSS	1321 ± (30)	391 ± (126)	85 ± (173)
gamboostLSS	1191 ± (30)	-*	- *
NAMLSS [1]	1201 ± (41)	-220 ± (210)	**-22** ± (137)
NAMLSS [2]	**1160** ± (49)	**-237** ± (219)	-11 ± (114)

[1] With $J \times K$ subnetworks. See Table 6.14 for an exemplary network structure.
[2] With J subnetworks and each subnetwork returning a parameter for the location and shape respectively. See Table B.1 for an exemplary network structure.
* gamboostLSS was not able to execute.

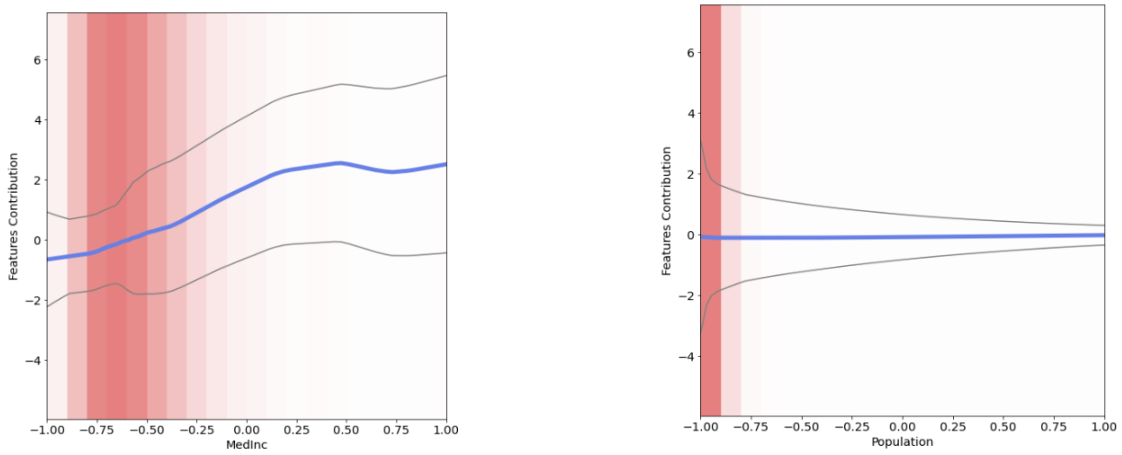

Figure 6.15: **California housing:** Graphs for median income and population respectively learned by the NAMLSS model. We see an increase in housing prices with a larger median income. Additionally, we find a larger variance in housing prices in less densely populated areas.

Additionally, we are able to accurately depict shifts in variance in the underlying data. It is, for example, clearly distinguishable, that with a larger median income, the house prices tend to vary much stronger than with a smaller median income (see Figure 6.15). A piece of information, that is lost in the models focusing solely on mean predictions

6.6 Conclusion and future work

We have presented neural additive models for location, scale and shape and their theoretical foundation as the neural counterpart to GAMLSS. NAMLSS can model an arbitrary number of parameters of the underlying data distribution while preserving the predictive quality of NAMs. The visual intelligibility achieved by NAMs is also maintained by NAMLSS, with the added benefit of gaining further insights from knowledge of additional distribution characteristics. Hence, NAMLSS are a further step in the direction of fully interpretable neural networks and already offer interpretability that may make them suitable for high-risk domains.

The extensibility of NAMLSS offers many different further applied and theoretical research directions. One important point is the extension of the modelling of the distribution of the response variable. Many empirical works focus on modelling not just one, but several responses conditionally on covariates. One way to do this is to use copula methods, which are a valuable extension of our approach, hence including a copula-based approach for NAMLSS models would greatly improve the overall general usefulness.

Another possible extension would be the adaptation to mixture density networks, as e.g. done by Seifert et al. (2022). Another possible focus is to switch our approach to a Bayesian-based training approach. Bayesian approaches are particularly well suited to deal with epistemic uncertainty and to incorporate it into the modelling. Another advantage is that Bayesian approaches are particularly suitable in cases where insufficiently small training datasets have to be dealt with and have been shown to have better prediction performance in these cases. Finally, there should be a focus on incorporating unstructured data to extend the previously purely tabular data with high-dimensional input structures.

6.7 Limitations

Although the presented method of NAMLSS takes advantage of the interpretation capabilities of the NAM framework and thus offers a better and easier interpretation of the results compared to pure deep learning approaches, it is still beholden to classical statistical models with their inherent interpretability and explainability. A critical point in the application of our proposed method, as well as comparable distributional statistical methods, is the choice of the correct distributional assumptions. The choice of the assumed distribution can strongly influence the results of the

model. Our approach requires some basic mathematical-statistical knowledge from the user. Also, the understanding that the presented approach focuses on (log)-likelihood and thus deviates from the classical approach of simply minimising an error measure may require some users to rethink their understanding of the model results. An obvious drawback of NAMLSS is that it requires more parameters to be trained. It is therefore computationally more expensive, as several additional subnetworks are required to model more parameters per feature.

Bibliography

Agarwal, R., Melnick, L., Frosst, N., Zhang, X., Lengerich, B., Caruana, R., and Hinton, G. E. (2021). Neural Additive Models: Interpretable Machine Learning with Neural Nets. *Agarwal, Rishabh, et al. "Neural additive models: Interpretable machine learning with neural nets." Advances in Neural Information Processing Systems*, 34.

Akaike, H. (1973). Information Theory and an Extension of the Maximum Likelihood Principle. *In B. N. Petrov, & F. Csaki (Eds.), Proceedings of the 2nd International Symposium on Information Theory (pp. 267-281).*

Allen, D. M. (1974). The Relationship Between Variable Selection and Data Agumentation and a Method for Prediction. *Technometrics : a journal of statistics for the physical, chemical, and engineering sciences*, 16(1):125–127.

Avriel, M. (2003). *Nonlinear programming: analysis and methods*. Courier Corporation.

Bartolucci, F., Bacci, S., and Pigini, C. (2017). Misspecification test for random effects in generalized linear finite-mixture models for clustered binary and ordered data. *Econometrics and Statistics*, 3:112–131.

Bates, D., Alday, P., Kleinschmidt, D., Calderón, J. B. S., Noack, A., Kelman, T., Bouchet-Valat, M., Gagnon, Y. L., Babayan, S., Mogensen, P. K., Piibeleht, M., Hatherly, M., Saba, E., and Baldassari, A. (2020). Juliastats/mixedmodels.jl: v2.3.0.

Bates, D., Mächler, M., Bolker, B., and Walker, S. (2015). Fitting Linear Mixed-Effects Models Using **lme4**. *Journal of Statistical Software*, 67(1).

Belkin, M., Hsu, D., and Xu, J. (2020). Two models of double descent for weak features. *SIAM Journal on Mathematics of Data Science*, 2(4):1167–1180.

Berner, J., Grohs, P., Kutyniok, G., and Petersen, P. (2021). The modern mathematics of deep learning.

Borra, S. and Di Ciaccio, A. (2010). Measuring the prediction error. A comparison of cross-validation, bootstrap and covariance penalty methods. *Computational Statistics & Data Analysis*, 54(12):2976–2989.

Bregman, L. M. (1967). The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. https://arxiv.org/abs/2005.14165v4.

Buckland, S. T., Burnham, K. P., and Augustin, N. H. (1997). Model Selection: An Integral Part of Inference. *Biometrics*, 53(2):603–618.

Buhrmester, V., Münch, D., and Arens, M. (2019). Analysis of Explainers of Black Box Deep Neural Networks for Computer Vision: A Survey.

Burman, P. (1989). A Comparative Study of Ordinary Cross-Validation, v-Fold Cross-Validation and the Repeated Learning-Testing Methods. *Biometrika*, 76(3):503–514.

Burman, P., Chow, E., and Nolan, D. (1994). A Cross-Validatory Method for Dependent Data. *Biometrika*, 81(2):351–358.

Burnham, K. P. and Anderson, D. R. (2011). *Model Selection and Multi-Model Inference: A Practical Information-Theoretic Approach*. Springer, New York London, new ed edition.

Cantoni, E., Jacot, N., and Ghisletta, P. (2021). Review and comparison of measures of explained variation and model selection in linear mixed-effects models. *Econometrics and Statistics*.

Chang, C.-H., Caruana, R., and Goldenberg, A. (2021). Node-gam: Neural generalized additive model for interpretable deep learning. *arXiv preprint arXiv:2106.01613*.

Chen, D. and Ye, W. (2022). Generalized gloves of neural additive models: Pursuing transparent and accurate machine learning models in finance. *arXiv preprint arXiv:2209.10082*.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. ACM.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. https://arxiv.org/abs/1406.1078v3.

Chollet, F. (2018). *Deep Learning with Python*. Manning Publications Co, Shelter Island, New York.

Claeskens, G. and Hjort, N. L. (2008). *Model Selection and Model Averaging*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge ; New York.

Cole, T. J. and Green, P. J. (1992). Smoothing reference centile curves: the lms method and penalized likelihood. *Statistics in medicine*, 11(10):1305–1319.

Crainiceanu, C. M. and Ruppert, D. (2004). Likelihood ratio tests in linear mixed models with one variance component. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(1):165–185.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, pages 303–314.

de Waal, D. A. and du Toit, J. V. (2007). Generalized additive models from a neural network perspective. In *Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)*, pages 265–270. IEEE.

Di, C.-Z., Crainiceanu, C. M., Caffo, B. S., and Punjabi, N. M. (2009). Multilevel functional principal component analysis. *The annals of applied statistics*, 3(1):458.

Ding, J., Tarokh, V., and Yang, Y. (2018). Model Selection Techniques: An Overview. *IEEE Signal Processing Magazine*, 35(6):16–34.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Dubey, A., Radenovic, F., and Mahajan, D. (2022). Scalable interpretability via polynomials. *arXiv preprint arXiv:2205.14108*.

Dürr, O., Sick, B., and Murina, E. (2020). *Probabilistic Deep Learning: With Python, Keras and TensorFlow Probability*. Manning Publications.

Efron, B. (1983). Estimating the error rate of a prediction rule: Improvement on cross-validation. *Journal of the American Statistical Association*, 78(382):316–331.

Efron, B. (1986). How biased is the apparent error rate of a prediction rule? *Journal of the American Statistical Association*, 81(394):461–470.

Efron, B. (2004). The estimation of prediction error: Covariance penalties and cross-validation. *Journal of the American Statistical Association*, 99(467):619–632.

Efron, B. (2021). Resampling plans and the estimation of prediction error. *Stats*, 4(4):1091–1115.

Eilers, P. H. C. and Marx, B. D. (1996). Flexible Smoothing with B-splines and Penalties. *Statistical Science*, 11(2):89–102.

Elder, J. F. (2003). The generalization paradox of ensembles. *Journal of Computational and Graphical Statistics*, 12(4):853–864.

Enouen, J. and Liu, Y. (2022). Sparse interaction additive networks via feature interaction detection and sparse selection. *arXiv preprint arXiv:2209.09326*.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115–118.

Fahrmeir, L. (2013). *Regression: Models, Methods and Applications*. Springer, Berlin New York.

Faraway, J. J. (2006). *Extending the Linear Model with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models*. Chapman & Hall/CRC, Boca Raton.

FICO (2018). Fico explainable machine learning challenge.

Fisher, R. A. (1919). XV.—The Correlation between Relatives on the Supposition of Mendelian Inheritance. *Transactions of the Royal Society of Edinburgh*, 52(2):399–433.

Gao, T. and Jojic, V. (2016). Degrees of freedom in deep neural networks. *arXiv:1603.09260 [cs, stat]*.

Geisser, S. (1975). The Predictive Sample Reuse Method with Applications. *Journal of the American Statistical Association*, 70(350):320–328.

Ghalanos, A. and Theussl, S. (2015). *Rsolnp: General Non-linear Optimization Using Augmented Lagrange Multiplier Method*. R package version 1.16.

Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102:359–378.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, Massachusetts.

Gorishniy, Y., Rubachev, I., Khrulkov, V., and Babenko, A. (2021). Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34.

Greven, S. and Kneib, T. (2010). On the behaviour of marginal and conditional AIC in linear mixed models. *Biometrika*, 97(4):773–789.

Griesbach, C., Säfken, B., and Waldmann, E. (2021). Gradient boosting for linear mixed models. *The International Journal of Biostatistics*, 17(2):317–329.

Guo, W. (2002). Functional mixed effects models. *Biometrics*, 58(1):121–128.

Hansen, B. E. and Racine, J. S. (2012). Jackknife model averaging. *Journal of Econometrics*, 167(1):38–46.

Harville, D. (1976). Extension of the Gauss-Markov Theorem to Include the Estimation of Random Effects. *The Annals of Statistics*, 4(2):384–395.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York, New York, NY.

Hastie, T. J. (2017). Generalized additive models. In *Statistical models in S*, pages 249–307. Routledge.

Hauenstein, S., Dormann, C. F., and Wood, S. N. (2016). Computing AIC for black-box models using generalised degrees of freedom: A comparison with cross-validation. *arXiv:1603.02743 [stat]*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, USA. IEEE.

Henderson, C. R. (1950). Estimation of genetic parameters. In *Biometrics*, volume 6, pages 186–187. International Biometric Soc 1441 I ST, NW, Suite 700, Washington, DC 20005-2210.

Hestenes, M. R. (1969). Multiplier and gradient methods. *Journal of optimization theory and applications*, 4(5):303–320.

Ho, H. J. and Lin, T.-I. (2010). Robust linear mixed models using the skew t distribution with application to schizophrenia data. *Biometrical Journal*, 52(4):449–469.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1):55–67.

Hofner, B., Mayr, A., and Schmid, M. (2014). gamboostlss: An r package for model building and variable selection in the gamlss framework. *arXiv preprint arXiv:1407.1774*.

Horel, E. and Giesecke, K. (2020). Significance tests for neural networks. *Journal of Machine Learning Research*, 21(227):1–29.

Hothorn, T., Kneib, T., and Bühlmann, P. (2014). Conditional transformation models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1):3–27.

Huang, C.-Z. A., Vaswani, A., Uszkoreit, J., Shazeer, N., Simon, I., Hawthorne, C., Dai, A. M., Hoffman, M. D., Dinculescu, M., and Eck, D. (2018). Music Transformer. https://arxiv.org/abs/1809.04281v3.

Huang, Y., Giledereli, B., Köksal, A., Özgür, A., and Ozkirimli, E. (2021). Balancing methods for multi-label text classification with long-tailed class distribution. *arXiv preprint arXiv:2109.04712*.

IBM (2019). Telco customer churn.

Janson, L., Fithian, W., and Hastie, T. J. (2015). Effective degrees of freedom: A flawed metaphor. *Biometrika*, 102(2):479–485.

Jo, W. and Kim, D. (2022). Neural additive models for nowcasting. *arXiv preprint arXiv:2205.10020*.

Joe, H. (1997). *Multivariate Models and Multivariate Dependence Concepts*. Chapman and Hall/CRC, New York.

Kaggle (2019). Churn modelling.

Kant, G., Wiebelt, L., Weisser, C., Kis-Katos, K., Luber, M., and Säfken, B. (2022). An iterative topic model filtering framework for short and noisy user-generated data: Analyzing conspiracy theories on twitter. *International Journal of Data Science and Analytics*.

Karabatak, M. and Ince, M. C. (2009). An expert system for detection of breast cancer based on association rules and neural network. *Expert Systems with Applications*, 36(2):3465–3469.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kneib, T., Silbersdorff, A., and Säfken, B. (2021). Rage against the mean–a review of distributional regression approaches. *Econometrics and Statistics*.

Kook, L., Götschi, A., Baumann, P. F., Hothorn, T., and Sick, B. (2022). Deep interpretable ensembles.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Kullback, S. and Leibler, R. A. (1951). On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86.

Kuznetsova, A., Brockhoff, P. B., and Christensen, R. H. B. (2017). lmerTest package: Tests in linear mixed effects models. *Journal of Statistical Software*, 82(13).

Laird, N. M. and Ware, J. H. (1982). Random-Effects Models for Longitudinal Data. *Biometrics*, 38(4):963.

Lantz, B. (2019). *Machine learning with R: expert techniques for predictive modeling*. Packt publishing ltd.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (Nov./1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Li, C., Yin, W., Jiang, H., and Zhang, Y. (2013). An efficient augmented lagrangian method with applications to total variation minimization. *Computational Optimization and Applications*, 56(3):507–530.

Liang, H., Wu, H., and Zou, G. (2008). A note on conditional aic for linear mixed-effects models. *Biometrika*, 95(3):773–778.

Lin, T. I. and Lee, J. C. (2006). A robust approach to t linear mixed models applied to multiple sclerosis data. *Statistics in Medicine*, 25(8):1397–1412.

Lin, Y., Meng, Y., Sun, X., Han, Q., Kuang, K., Li, J., and Wu, F. (2021). Bertgcn: Transductive text classification by combining gcn and bert. *arXiv preprint arXiv:2105.05727*.

Luber, M., Thielmann, A., and Säfken, B. (2023). Structural neural additive models: Enhanced interpretable machine learning. *arXiv preprint arXiv:2302.09275*.

Luenberger, D. G., Ye, Y., et al. (2008). *Linear and nonlinear programming*, volume 2. Springer.

Ma, J., Sheridan, R. P., Liaw, A., Dahl, G. E., and Svetnik, V. (2015). Deep Neural Nets as a Method for Quantitative Structure–Activity Relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274.

Mallows, C. L. (1973). Some Comments on C p. *Technometrics*, 15(4):661–675.

Marra, G. and Racine, R. (2022). Generalized Joint Regression Modelling.

Mayr, A., Fenske, N., Hofner, B., Kneib, T., and Schmid, M. (2010). GAMLSS for high-dimensional data – a flexible approach based on boosting. page 29.

Nagrani, A., Yang, S., Arnab, A., Jansen, A., Schmid, C., and Sun, C. (2021). Attention bottlenecks for multimodal fusion. *Advances in Neural Information Processing Systems*, 34:14200–14213.

Nash, W. J., Sellers, T. L., Talbot, S. R., Cawthorn, A. J., and Ford, W. B. (1994). The population biology of abalone (haliotis species) in tasmania. i. blacklip abalone (h. rubra) from the north coast and islands of bass strait. *Sea Fisheries Division, Technical Report*, 48:p411.

Nelder, J. A. and Wedderburn, R. W. (1972). Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384.

Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer Series in Operations Research. Springer, New York, 2nd ed edition.

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

OpenAI (2023). GPT-4 Technical Report.

Pace, R. K. and Barry, R. (1997). Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Peroni, M., Kurban, M., Yang, S. Y., Kim, Y. S., Kang, H. Y., and Song, J. H. (2022). Extending the neural additive model for survival analysis with ehr data. *arXiv preprint arXiv:2211.07814*.

Potthoff, R. F. and Roy, S. (1964). A generalized multivariate analysis of variance model useful especially for growth curve problems. *Biometrika*, 51(3-4):313–326.

Potts, W. J. (1999). Generalized additive neural networks. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 194–200.

Powell, M. J. (1969). A method for nonlinear constraints in minimization problems. *Optimization*, pages 283–298.

R Core Team (2019). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Radenovic, F., Dubey, A., and Mahajan, D. (2022). Neural basis models for interpretability. *arXiv preprint arXiv:2205.14120*.

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. https://arxiv.org/abs/1511.06434v2.

Ramani, S., Blu, T., and Unser, M. (2008). Monte-carlo sure: A black-box optimization of regularization parameters for general denoising algorithms. *IEEE Transactions on Image Processing*, 17(9):1540–1554.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Rigby, R. A. and Stasinopoulos, D. M. (2005). Generalized Additive Models for Location, Scale and Shape. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 54(3):507–554.

Roberts, D. A., Yaida, S., and Hanin, B. (2022). *The Principles of Deep Learning Theory*.

Rosset, S. and Tibshirani, R. J. (2020). From Fixed-X to Random-X Regression: Bias-Variance Decompositions, Covariance Penalties, and Prediction Error Estimation. *Journal of the American Statistical Association*, 115(529):138–151.

Rügamer, D., Kolb, C., Fritz, C., Pfisterer, F., Bischl, B., Shen, R., Bukas, C., e Sousa, L. B. d. A., Thalmeier, D., Baumann, P., Klein, N., and Müller, C. L. (2021). Deepregression: A flexible neural network framework for semi-structured deep distributional regression. *arXiv:2104.02705 [cs, stat]*.

Rügamer, D., Kolb, C., and Klein, N. (2020). Semi-structured deep distributional regression: Combining structured additive models and deep learning. *arXiv preprint arXiv:2002.05777*.

Ruppert, D., Wand, M. P., and Carroll, R. J. (2003). *Semiparametric regression*. Number 12. Cambridge university press.

Ruszczynski, A. (2011). *Nonlinear optimization*. Princeton university press.

Saefken, B., Kneib, T., van Waveren, C.-S., and Greven, S. (2014). A unifying approach to the estimation of the conditional akaike information in generalized linear mixed models. *Electronic Journal of Statistics*, 8(1):201–225.

Säfken, B. and Kneib, T. (2020). Conditional covariance penalties for mixed models. *Scandinavian Journal of Statistics*, 47(3):990–1010.

Säfken, B., Rügamer, D., Kneib, T., and Greven, S. (2021). Conditional Model Selection in Mixed-Effects Models with **cAIC4**. *Journal of Statistical Software*, 99(8).

Santosa, F. and Symes, W. W. (1986). Linear Inversion of Band-Limited Reflection Seismograms. *SIAM Journal on Scientific and Statistical Computing*, 7(4):1307–1330.

Savage, N. (2022). Breaking into the black box of artificial intelligence. *Nature*.

Schoenberg, I. J. (1946). Contributions to the problem of approximation of equidistant data by analytic functions. Part B. On the problem of osculatory interpolation. A second class of analytic approximation formulae. *Quarterly of Applied Mathematics*, 4(2):112–141.

Schwarz, G. et al. (1978). Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464.

Seifert, Q. E., Thielmann, A., Bergherr, E., Säfken, B., Zierk, J., Rauh, M., and Hepp, T. (2022). Penalized regression splines in mixture density networks.

Self, S. G. and Liang, K.-Y. (1987). Asymptotic properties of maximum likelihood estimators and likelihood ratio tests under nonstandard conditions. *Journal of the American Statistical Association*, 82(398):605–610.

Shapley, L. (1953). Quota solutions op n-person games1. *Edited by Emil Artin and Marston Morse*, page 343.

Silbersdorff, A., Lynch, J., Klasen, S., and Kneib, T. (2018). Reconsidering the income-health relationship using distributional regression. *Health economics*, 27(7):1074–1088.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Stasinopoulos, D., Rigby, R., and Fahrmeir, L. (2000). Modelling rental guide data using mean and dispersion additive models. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 49(4):479–493.

Stein, C. et al. (1972). A bound for the error in the normal approximation to the distribution of a sum of dependent random variables. In *Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Probability Theory*. The Regents of the University of California.

Stein, C. M. (1981). Estimation of the mean of a multivariate normal distribution. *The Annals of Statistics*, 9(6).

Stone, M. (1977). An Asymptotic Equivalence of Choice of Model by Cross-Validation and Akaike's Criterion. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):44–47.

Sundararajan, M. and Najmi, A. (2020). The many shapley values for model explanation. In *International conference on machine learning*, pages 9269–9278. PMLR.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. https://arxiv.org/abs/1409.3215v3.

Thielmann, A., Kruse, R.-M., Kneib, T., and Säfken, B. (2023). Neural Additive Models for Location Scale and Shape: A Framework for Interpretable Neural Regression Beyond the Mean.

Thormann, M.-L., Farchmin, J., Weisser, C., Kruse, R.-M., Säfken, B., and Silbersdorff, A. (2021). Stock price predictions with LSTM neural networks and twitter sentiment. *Statistics, Optimization & Information Computing*, 9(2):268–287.

Tibshirani, R. (1996). Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

Tibshirani, R. J. and Taylor, J. (2012). Degrees of freedom in lasso problems. *The Annals of Statistics*, 40(2).

Umlauf, N., Klein, N., and Zeileis, A. (2018). BAMLSS: Bayesian Additive Models for Location, Scale, and Shape (and Beyond). *Journal of Computational and Graphical Statistics*, 27(3):612–627.

Vaida, F. and Blanchard, S. (2005). Conditional akaike information for mixed-effects models. *Biometrika*, 92(2):351–370.

Varadhan, R. (2015). *alabama: Constrained Nonlinear Optimization*. R package version 2015.3-1.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention Is All You Need. https://arxiv.org/abs/1706.03762v5.

Verbeke, G. and Molenberghs, G. (2009). *Linear mixed models for longitudinal data*. Springer Science & Business Media.

Wager, C., Vaida, F., and Kauermann, G. (2007). Model selection for penalized spline smoothing using akaike information criteria. *Australian & New Zealand Journal of Statistics*, 49(2):173–190.

Wang, H., Zhang, X., and Zou, G. (2009). Frequentist model averaging estimation: A review. *Journal of Systems Science and Complexity*, 22(4):732–748.

Wang, P., Puterman, M. L., Cockburn, I., and Le, N. (1996). Mixed poisson regression models with covariate dependent rates. *Biometrics*, pages 381–400.

Wang, T., Yang, J., Li, Y., and Wang, B. (2021). Partially interpretable estimators (pie): black-box-refined interpretable machine learning. *arXiv preprint arXiv:2105.02410*.

Wood, S. N. (2013). A simple test for random effects in regression models. *Biometrika*, 100(4):1005–1010.

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

Wood, S. N., Li, Z., Shaddick, G., and Augustin, N. H. (2017). Generalized additive models for gigadata: Modeling the U.K. black smoke network daily data. *Journal of the American Statistical Association*, 112(519):1199–1210.

Wood, S. N., Pya, N., and Säfken, B. (2016). Smoothing Parameter and Model Selection for General Smooth Models. *Journal of the American Statistical Association*, 111(516):1548–1563.

Yang, Z., Zhang, A., and Sudjianto, A. (2021). Gami-net: An explainable neural network based on generalized additive models with structured interactions. *Pattern Recognition*, 120:108192.

Ye, J. (1998). On measuring and correcting the effects of data mining and model selection. *Journal of the American Statistical Association*, 93(441):120–131.

Yu, J., Wang, Z., Vasudevan, V., Yeung, L., Seyedhosseini, M., and Wu, Y. (2022). Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*.

Zeng, A., Chen, M., Zhang, L., and Xu, Q. (2022). Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504*.

Zhang, X., Zou, G., and Liang, H. (2014). Model averaging and weight choice in linear mixed-effects models. *Biometrika*, 101(1):205–218.

Zhou, T., Ma, Z., Wen, Q., Sun, L., Yao, T., Jin, R., et al. (2022). Film: Frequency improved legendre memory model for long-term time series forecasting. *arXiv preprint arXiv:2205.08897*.

Appendix

Appendix A: Supplemental material Part I

Computational Statistics and Data Analysis 167 (2022) 107351

Contents lists available at ScienceDirect

Computational Statistics and Data Analysis

www.elsevier.com/locate/csda

Model averaging for linear mixed models via augmented Lagrangian

René-Marcel Kruse [a,*], Alexander Silbersdorff [a,b], Benjamin Säfken [a,b]

[a] Chair of Statistics, University of Göttingen, Humboldtallee 3, 37073 Göttingen, Germany
[b] Campus-Institut Data Science, Goldschmidtstraße 1, 37077 Göttingen, Germany

ARTICLE INFO

Article history:
Received 22 December 2020
Received in revised form 16 September 2021
Accepted 16 September 2021
Available online 29 September 2021

Keywords:
Optimization
Augmented Lagrangian
Model averaging
Linear mixed models
Conditional AIC

ABSTRACT

Model selection for linear mixed models has been a focus of recent research in statistics. Yet, the method of model averaging has been sparsely explored in this context. A weight finding criterion for model averaging of linear mixed models is introduced, as well as its implementation for the programming language R. Since the optimization of the underlying criterion is non-trivial, a fast and robust implementation of the augmented Lagrangian optimization technique is employed. Furthermore, the influence of the weight finding criterion on the resulting model averaging estimator is illustrated through simulation studies and two applications based on real data.

1. Introduction

The class of linear mixed models (Henderson, 1950) is a very powerful and flexible analytic tool, that enjoys popularity especially for the analysis of clustered and longitudinal data (Laird and Ware, 1982; Verbeke and Molenberghs, 2009), for spline smoothing (Ruppert et al., 2003; Wager et al., 2007) and for functional data analysis (Guo, 2002; Di et al., 2009).

Especially the development and advancement of software for fitting and evaluating linear mixed models is a very active field. It ranges from implementations for commercial statistics programs like `SAS`, to open-source versions like the de-facto standard in `R lme4` (Bates et al., 2014) or the `MixedModels` (Bates et al., 2020) Package for `Julia`. Due to the flexibility and thus, possible complexity of the models, the question of suitable model selection procedures becomes a focal point of research.

However, linear mixed model deviate from the imposed regularity conditions of classical linear models and thus introduce a problem with the use of information criteria for model choice, such as the widely adopted Akaike Information Criterion (Akaike, 1973) (Wager et al., 2007). Furthermore, evaluating the suitability of the included random effects of models with nested or clustered structures suffer from limitations like boundary issues with likelihood-ratio tests (Crainiceanu and Ruppert, 2004; Wood, 2013). An overview of measures of explained variation and model selection in linear mixed-effects models can be found in Cantoni et al. (2021).

* Corresponding author.
E-mail addresses: rene-marcel.kruse@uni-goettingen.de (R.-M. Kruse), asilbersdorff@uni-goettingen.de (A. Silbersdorff), benjamin.saefken@uni-goettingen.de (B. Säfken).
URL: https://www.uni-goettingen.de/en/610058.html (R.-M. Kruse).

https://doi.org/10.1016/j.csda.2021.107351
0167-9473/

Vaida and Blanchard (2005) show, however, that it is possible to derive an AIC from the conditional form of the linear mixed effect model, which has proven to be particularly suitable accounting for possible shrinkage within the random effects (Säfken et al., 2018). Liang et al. (2008) suggest a version of the conditional AIC that corrects for the estimation uncertainty of the variance parameters of the random effects. Still, this proposed version is computationally intensive as it relies on numerical approximation. Greven and Kneib (2010) prove that an analytical solution can be derived and thus, reduce the computational intensity of the corrected version of the conditional AIC.

Another approach addressing model uncertainty is model averaging. Instead of choosing a single model from a list of candidate models based on information criteria such as AIC or the Bayesian information criterion (Schwarz et al., 1978), a weighted average of the considered models is calculated and then used for analysis. An important key factor when applying model averaging is the selection of the underlying weights. Different proposals have been brought forward, the most prominent being the approach of information criteria based weights by Buckland et al. (1997). Yet, a majority of proposals aim at classical linear models and do encounter difficulties when applied to the model framework of linear mixed models. A proposal by Zhang et al. (2014) demonstrates that it is possible to construct an asymptotically optimal weight finding criterion for model averaging of linear mixed models based on the conditional AIC and a quadratic loss function. However, a computationally stable and fast optimization of such a weight determination criterion is not available up to date. The nonlinear nature of the criterion itself, as well as the nature of the underlying constraints in the form of simultaneous equality and inequality conditions, necessitates complex and advanced optimization methods that are not part of the basic version of the programming language `R` (R Core Team, 2019).

In this paper we present a weight finding criterion for the calculation of asymptotically optimal weights based on the work of Greven and Kneib (2010) and Zhang et al. (2014). In addition we present an implementation of the proposed weight finding criterion for the programming language `R`, which we have released as part of the R-Package `cAIC4` (Säfken et al., 2018). Furthermore, we describe the special nonlinear optimization under equality and inequality constraints of the underlying problem. We illustrate the approach of solving such a problem by applying the augmented Lagrangian method (Hestenes, 1969; Li et al., 2013) and present our implementation of the algorithm.

This paper is structured as follows: Section 2 introduces the theory and formulations of linear mixed models, as well as the estimation and the application of linear mixed models for spline smoothing. Section 3 presents the concept of the conditional AIC. This section also induces the concept of conditional model averaging and the proposed weight finding criterion. The following Section 4 provides an introduction to the underlying mathematical concepts of the augmented Lagrangian method, as well as its application to our weight finding optimization problem. Section 5 analyzes the properties of the implemented methods by applying them in three different simulation settings. Section 6 studies the proposed model averaging method applied to real-world examples, whereas the last Section 7, gives a summary of the findings of the previous sections and also gives an outlook of further work concerning model selection and model averaging of linear mixed models.

2. Linear mixed models

The general design of linear mixed models assumed in the following sections is

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{b} + \boldsymbol{\varepsilon}, \tag{1}$$

where $\boldsymbol{y}$ represents the vector of the n observed responses $\boldsymbol{y} = (y_1, \ldots, y_n)^T$, $\boldsymbol{X}$ and $\boldsymbol{Z}$ representing design matrices with full column ranks p and q, with the $p \times 1$ vector of fixed $\boldsymbol{\beta}$ and $\boldsymbol{b}$ as the $q \times 1$ vector of random effects. The $n \times 1$ vector $\boldsymbol{\varepsilon}$ represents the unobserved random errors. Both $\boldsymbol{b}$ and $\boldsymbol{\varepsilon}$ are assumed to be independent and follow a multivariate Gaussian distribution, such that

$$\begin{pmatrix} \boldsymbol{b} \\ \boldsymbol{\varepsilon} \end{pmatrix} \sim \mathcal{N}\left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \boldsymbol{D}_{\boldsymbol{\theta}} & 0 \\ 0 & \boldsymbol{\Sigma} \end{pmatrix} \right\},$$

with $\boldsymbol{D}_{\boldsymbol{\theta}}$ being a $q \times q$ block-diagonal, positive, semi-definite variance-covariance matrix that depends on a covariance parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_j)^T$ and $\boldsymbol{\Sigma}$ the overall model covariance matrix with dimension $n \times n$, which in the following illustrations is assumed to follow the standard case of $\sigma^2 \boldsymbol{I}$. The normality assumption, however, is not mandatory and is only introduced for convenience, allowing for likelihood-based procedures to estimate unknown parameters in $\boldsymbol{D}_{\boldsymbol{\theta}}$ and of the residual variance.

Furthermore let the marginal covariance matrix $\boldsymbol{V}_{\boldsymbol{\theta}}$ of $\boldsymbol{y}$ be defined as follows

$$\boldsymbol{V}_{\boldsymbol{\theta}} = \operatorname{cov}(\boldsymbol{y}) = \boldsymbol{\sigma}^2 \boldsymbol{I} + \boldsymbol{Z}\boldsymbol{D}_{\theta}\boldsymbol{Z}^T.$$

The inherent randomness of the random effects makes it possible to formulate linear mixed models in two different forms, in a marginal or in a conditional ways. The marginal formulation treats the random effects as an additional part of the already random error term $\boldsymbol{\varepsilon}$ (Fahrmeir et al., 2013). The conditional formulation on the other hand approaches the random effects differently, by treating them as penalized coefficients. In this form the conditional responses are distributed as follows

$$\boldsymbol{y}|\boldsymbol{b} \sim \mathcal{N}(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{b}, \sigma^2\boldsymbol{I}).$$

2.1. Estimation of linear mixed models

For given variance parameters $\boldsymbol{\theta}$, fixed as well as random effects can be estimated respectively predicted via

$$\begin{aligned} \hat{\boldsymbol{\beta}} &= \left(\boldsymbol{X}^T\boldsymbol{V}_{\boldsymbol{\theta}}^{-1}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{V}_{\boldsymbol{\theta}}^{-1}\boldsymbol{y}, \\ \hat{\boldsymbol{b}} &= \boldsymbol{D}_{\boldsymbol{\theta}}\boldsymbol{Z}^T\boldsymbol{V}_{\boldsymbol{\theta}}^{-1}(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}), \end{aligned} \tag{2}$$

where the resulting estimator of the fixed effects $\hat{\boldsymbol{\beta}}$ is the best linear unbiased estimator and it is also the maximum-likelihood estimator of $\boldsymbol{\beta}$ and the predictor of the random effects $\hat{\boldsymbol{b}}$ is also the best linear unbiased predictor of $\boldsymbol{b}$ (Harville et al., 1976). The corresponding profile log-likelihood for all underlying variance parameters $\boldsymbol{\theta}$ is thus up to a constant equal to

$$\ell_P(\boldsymbol{\theta}) = -\frac{1}{2}\left[\log|\boldsymbol{V}_{\boldsymbol{\theta}}| + (\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^T\boldsymbol{V}_{\boldsymbol{\theta}}^{-1}(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})\right]. \tag{3}$$

The maximization of the profile log-likelihood with respect to $\boldsymbol{\theta}$ delivers the ML-estimator $\hat{\boldsymbol{\theta}}_{ML}$. Instead of estimating $\boldsymbol{\theta}$ via profile log-likelihood, it is also possible to determine $\boldsymbol{\theta}$ via the marginal or restricted log-likelihood. Whereas the complementary restricted log-likelihood for $\boldsymbol{\theta}$ takes the following form (up to an additive constant) of

$$\ell_R(\boldsymbol{\theta}) = \ell_P(\boldsymbol{\theta}) - \frac{1}{2}\log\left|\boldsymbol{X}^T\boldsymbol{V}_{\boldsymbol{\theta}}^{-1}\boldsymbol{X}\right|, \tag{4}$$

maximizing this restricted log-likelihood results in the REML-estimator for $\boldsymbol{\theta}$, $\hat{\boldsymbol{\theta}}_{REML}$ (Harville et al., 1976). In a general setting the REML-estimator leads to a less biased estimation result than the ML-estimator (Fahrmeir et al., 2013).

2.2. Linear mixed models for spline smoothing

Apart from using the linear mixed models as a data analysis tool itself, this model class can also be used as a vehicle to fit semi-parametric models (Ruppert et al., 2003). This connection can most easily be explained for the case of truncated polynomials. For the simple univariate smoothing case consider the following model

$$y_i = f(x_i) + \varepsilon_i, \quad i = 1, \ldots, n,$$

where $f(x_i)$ is represented by a sum of scaled basis functions and $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. In the case of truncated polynomials, the following basis representation is utilized

$$f(x) = \sum_{j=0}^{d}\beta_j x^j + \sum_{j=1}^{K} b_j\left(x - \varkappa_j\right)_+^d,$$

where the domain of x is partitioned by $K \in \mathbb{N}$ knots $\varkappa_1 < \cdots < \varkappa_K$ in such a way that for $d \in \mathbb{N}$

$$(z)_+^d = z^d \cdot I(z > 0) = \begin{cases} z^d & \text{if } z > 0, \\ 0 & \text{if } z \leq 0. \end{cases}$$

The penalised least-squares criterion is employed to prevent overfitting and to ensure smoothness of the estimated function, resulting in

$$\mathrm{ls}_{\mathrm{pen}}(\boldsymbol{\beta}, \boldsymbol{b}) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b}) + \boldsymbol{b}^T\boldsymbol{D}_{\theta}^{-1}\boldsymbol{b},$$

with $\boldsymbol{D}_{\theta} = \tau^2\boldsymbol{I}_K$ where $\boldsymbol{\theta} = \tau^2$ and $\boldsymbol{\Sigma} = \sigma^2\boldsymbol{I}$. Thus the relation between the variances is $\tau^2 = \xi\sigma^2$. In this case, given a fixed smoothing parameter ξ, the equation coincides with the best linear unbiased estimator for $\boldsymbol{\beta}$ and the best linear unbiased predictor for $\boldsymbol{b}$ from equation (2) in the linear mixed model case with fixed τ^2. The underlying parameter $\xi = \tau^2/\sigma^2$ can be understood as a trade-off between function fit and -smoothness. Interpreting this problem as a linear mixed effect model allows ξ to be understood as the variance ratio of random and fixed effects and therefore to be determined via the presented ML (3) or REML (4) approaches (Ruppert et al., 2003). It is possible to represent penalized regression smoothers as part of mixed models, this allows the smoothing parameters to be estimated as part of the variance component parameters using the introduced likelihood procedures. As a consequence, linearly mixed models can be used to fit generalized additive mixed models (Wood, 2017).

3. Conditional model choice and model averaging

When it comes to the choice of linear mixed models and their random effect structures, the question arises as to which of the two likelihoods should be the basis for information criteria such as the AIC. The rationale for the choice depends on the intended application of the model (Vaida and Blanchard, 2005). The marginal approach allows statements about fixed population effects or of predictions about changed random effects structures. In contrast if the interest lies in statements based on the random effects of fitted models or in predictions based on existing random effect structures, the conditional form is particularly suitable. Due to these characteristic, the corresponding conditional AIC is, thus, better suited to select which random effects to include and which not to (Säfken et al., 2018). For a more mathematical investigation of the differences between the conditional and the marginal AIC, see Greven and Kneib (2010).

3.1. Conditional Akaike information criterion

One of the most widely used criteria for model selection is the Akaike Information Criterion (AIC) (Akaike, 1973). The AIC is an estimator of the relative Kullback-Leibler-Distance (Kullback and Leibler, 1951) and for simple linear regression models is up to a constant given by

$$\text{AIC} = -\frac{(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^T(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})}{\hat{\sigma}^2} + 2p,$$

with the number of parameters or degrees of freedom p. Considering a number of possible candidate models, the model that displays the lowest AIC value among all candidate models is most favourable. In more general this model selection criterion can also be derived as an estimator for the squared prediction error (Efron, 2004) as

$$\text{AIC} = (\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^T(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}) + 2\hat{\sigma}^2 \sum_{i=1}^{n} \left(\frac{\partial \hat{y}_i}{\partial y_i} \right),$$

with a substitution for the degrees of freedom first formalized by Stein et al. (1972)

$$\sum_{i=1}^{n} \frac{\partial \hat{y}_i}{\partial y_i} = \frac{\partial \hat{\boldsymbol{y}}}{\partial \boldsymbol{y}} = \text{tr}(\boldsymbol{H}) = \rho, \tag{5}$$

for simple linear regression models with hat matrix $\boldsymbol{H}$.

Two different AIC criteria can be employed when working with linear mixed models, the marginal AIC which is based on the marginal formulation of the log-likelihood, and the conditional AIC which is based on the conditional log-likelihood. Depending on the research question, the intention, as well as the interpretation, the respective approach varies (Vaida and Blanchard, 2005; Greven and Kneib, 2010).

The proposed estimator of the conditional AIC of Vaida and Blanchard (2005) takes the form of

$$\text{cAIC} = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b})^T \boldsymbol{V}_{\boldsymbol{\theta}} (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{b}) + 2(\rho + 1).$$

The derivation of Vaida and Blanchard (2005) requires that the variance-covariance matrix of the random effects has to be known. Liang et al. (2008) propose a corrected version of the conditional AIC based on the numerical approximation of the degrees of freedom as in (5) and therefore mitigate the strictness of the assumptions in respect to the variance-covariance matrix.

This approach, however, introduces high computational costs. Greven and Kneib (2010) offer an analytical version of the bias correction term and allow the calculation of the corrected form of the cAIC without having to resort to complex numerical approximation. Theorem 3 of Greven and Kneib (2010) allows ρ to be formulated as

$$\rho = \text{tr}\left(\hat{\boldsymbol{H}}\right) + \sum_{j=1}^{J} \frac{\partial \hat{\boldsymbol{\theta}}_j}{\partial \boldsymbol{y}^T} \hat{\boldsymbol{A}} \boldsymbol{Q}_j \hat{\boldsymbol{A}} \boldsymbol{y},$$

where $\hat{\boldsymbol{H}} = \boldsymbol{I} - \sigma^2 \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1} + \sigma^2 \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1/2} (\hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1/2} \boldsymbol{X} (\boldsymbol{X}^T \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1} \boldsymbol{X})^{-1} \boldsymbol{X}^T \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1/2}) \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1/2}$, $\boldsymbol{Q}_j = \partial \boldsymbol{V}_{\boldsymbol{\theta}} / \partial \boldsymbol{\theta}_j$, furthermore $\hat{\boldsymbol{\theta}}_j$ is the j-th element of $\hat{\boldsymbol{\theta}}$ and with $\hat{\boldsymbol{A}} = \sigma^2 \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1/2} (\boldsymbol{I} - (\hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1/2} \boldsymbol{X} (\boldsymbol{X}^T \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1} \boldsymbol{X})^{-1} \boldsymbol{X}^T \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1/2})) \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}^{-1/2}$.

To cover possible parameters on the boundary of the parameter space, we have to partition it similar to Self and Liang (1987). In this case, $\boldsymbol{\theta}$ consists of all elements of the matrix $\boldsymbol{D}_{\boldsymbol{\theta}}$ and it is sorted in such a way that the last s elements of the estimator $\hat{\boldsymbol{\theta}}$ are equal to zero and with $\hat{\boldsymbol{\eta}} = (\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_{J-s})^T$ it can be shown under Theorem 3 of Greven and Kneib (2010) that

$$\frac{\partial \hat{\boldsymbol{\theta}}_j}{\partial \boldsymbol{y}^T} = 0, \quad j = J - s + 1, \ldots, J, \quad \frac{\partial \hat{\boldsymbol{\eta}}}{\partial \boldsymbol{y}^T} = \boldsymbol{B}^{-1} \boldsymbol{G} = 0,$$

where $\boldsymbol{B}$ is a $(J-s)\times(J-s)$ matrix with (i,j)-th element such that

$$(b)_{ij} = -\operatorname{tr}(\hat{\boldsymbol{V}}^{-1}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{V}}^{-1}\hat{\boldsymbol{Q}}_j) - n\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{A}}\boldsymbol{y}\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_j\hat{\boldsymbol{A}}\boldsymbol{y}(\boldsymbol{y}^T\hat{\boldsymbol{A}}\boldsymbol{y})^{-2} + 2n\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_j\hat{\boldsymbol{A}}\boldsymbol{y}(\boldsymbol{y}^T\hat{\boldsymbol{A}}\boldsymbol{y})^{-1},$$

with $\boldsymbol{G}$ being a $(J-s)\times n$ matrix with the i-th row given by

$$\boldsymbol{g}_i = 2n\left\{-(\boldsymbol{y}^T\hat{\boldsymbol{A}}\boldsymbol{y})^{-2}\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{A}}\boldsymbol{y}\boldsymbol{y}^T\hat{\boldsymbol{A}} + (\boldsymbol{y}^T\hat{\boldsymbol{A}}\boldsymbol{y})^{-1}\boldsymbol{y}^T\hat{\boldsymbol{A}}\hat{\boldsymbol{Q}}_i\hat{\boldsymbol{A}}\right\}.$$

These expressions allow to calculate ρ directly so that no numerical procedures are necessary. The R-package `cAIC4` (Säfken et al., 2018) incorporates the corrected conditional AIC form and bias correction, and thus allows to perform model selection based on the conditional AIC and to calculate a analytical version of the degrees of freedom for models under consideration.

3.2. Conditional model averaging

Consider a given series of K possible linear mixed-effects candidate models according to (1), with the following form

$$\boldsymbol{y} = \boldsymbol{X}_k\boldsymbol{\beta}_k + \boldsymbol{Z}_k\boldsymbol{b}_k + \boldsymbol{\varepsilon}, \quad \boldsymbol{b}_k \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{D}_{\boldsymbol{\theta}_k}\right), \quad k = 1,\dots,K.$$

The fixed and random effects, as well as the variance-covariance matrices, can be determined via the REML approach, presented in (2). Hence the conditional mean is given by $\hat{\boldsymbol{y}}_k = \boldsymbol{X}_k\hat{\boldsymbol{\beta}}_k + \boldsymbol{Z}_k\hat{\boldsymbol{b}}_k$, leading to the following representation of the predicted values in terms of the estimated hat matrix $\hat{\boldsymbol{H}}_k$ as $\hat{\boldsymbol{y}}_k = \hat{\boldsymbol{H}}_k\boldsymbol{y}$.

The purpose of model averaging is to compose a weighted average over the random, as well as the fixed effect, estimators. For this consider a corresponding weighting vector $\boldsymbol{w} = (w_1,\dots,w_K)^T$ belonging to the set $\mathcal{W} = \{\boldsymbol{w}\in[0,1]^K : \sum_{k=1}^K w_k = 1\}$. The model averaging estimator is thus described by

$$\hat{\boldsymbol{y}}(\boldsymbol{w}) = \sum_{k=1}^K w_k\hat{\boldsymbol{y}}_k = \sum_{k=1}^K w_k\hat{\boldsymbol{H}}_k\boldsymbol{y} = \hat{\boldsymbol{H}}(\boldsymbol{w})\boldsymbol{y},$$

with $\hat{\boldsymbol{H}}(\boldsymbol{w}) = \sum_{k=1}^K w_k\hat{\boldsymbol{H}}_k$.

One of the more straightforward and widely used methods to determine the weights for the model average is based on a proposal by Buckland et al. (1997). This approach can be sometimes found in the literature labeled as smoothed weights and the weight finding criterion takes on the following form

$$w_k = \frac{\exp(-\mathcal{I}_k/2)}{\sum_{i=1}^K \exp(-\mathcal{I}_i/2)}, \tag{6}$$

with $\mathcal{I}_k$ representing the information criteria value for the respective candidate model k. A second approach is to derive the weights in such a way that in theory the model averaging estimator is asymptotically optimal, as in Zhang et al. (2014). The resulting estimator is optimal in the sense that the squared error of the calculated model average estimator is asymptotically equal to that of the infeasible best possible model average estimator. The authors achieve this by utilizing the squared loss of the model averaging estimator to derive a suitable criterion for weight determination. The underlying loss-function takes the following form of

$$L(\boldsymbol{w}) = (\hat{\boldsymbol{y}}(\boldsymbol{w}) - \boldsymbol{\mu})^T(\hat{\boldsymbol{y}}(\boldsymbol{w}) - \boldsymbol{\mu}),$$

where $\hat{\boldsymbol{y}}(\boldsymbol{w})$ represents the model average estimator and $\boldsymbol{\mu}$ the true but unknown mean. By applying the theorem by Stein et al. (1972) the expected loss is given by

$$E_{\boldsymbol{y}|\boldsymbol{b}}\left((\hat{\boldsymbol{y}}(\boldsymbol{w}) - \boldsymbol{\mu})^T(\hat{\boldsymbol{y}}(\boldsymbol{w}) - \boldsymbol{\mu})\right) = E_{\boldsymbol{y}|\boldsymbol{b}}\left((\boldsymbol{y} - \hat{\boldsymbol{y}}(\boldsymbol{w}))^T(\boldsymbol{y} - \hat{\boldsymbol{y}}(\boldsymbol{w})) + 2\sigma^2\boldsymbol{w}^T\boldsymbol{\rho} - n\sigma^2\right), \tag{7}$$

the $K\times 1$ elements of $\boldsymbol{\rho}$ are being defined as $\rho_k = \operatorname{tr}\left(\partial\hat{\boldsymbol{y}}_k/\partial\boldsymbol{y}^T\right)$.

The method presented here and especially formula (7) essentially rely on Stein's theorem. In turn, this requires the normality assumption to hold for the conditional model $\boldsymbol{y}|\boldsymbol{b}$. There are approaches to generalize this method beyond normality, see for Ye (1998) and Efron (2004) or Säfken and Kneib (2020) in the context of mixed models. Such an extension is highly relevant as it would not only allow for other error distributions but also for other random effects distributions that as are used for robust linear mixed models which are based on skewed t distributions as proposed in Lin and Lee (2006) and Ho and Lin (2010).

Based on the design of the model averaging estimator, the weight finding criterion is, therefore, defined as follows

$$C(\boldsymbol{w}) = \left(\boldsymbol{y} - \hat{\boldsymbol{y}}(\boldsymbol{w})\right)^T \left(\boldsymbol{y} - \hat{\boldsymbol{y}}(\boldsymbol{w})\right) + 2\sigma^2 \boldsymbol{w}^T \boldsymbol{\rho}, \tag{8}$$

thus, the optimal vector of weights $\hat{\boldsymbol{w}}$ for the $\hat{\boldsymbol{y}}(\hat{\boldsymbol{w}})$ minimizes this criterion, such that $\hat{\boldsymbol{w}} = \text{argmin}_{\boldsymbol{w}\in\mathcal{W}} \hat{C}(\boldsymbol{w})$.

4. Practical model averaging with cAIC4

The weight selection criterion (8) can be seen as a nonlinear optimization problem, where the weights are subject to equality constraints and to bound constraints alike and can be formulated as

$$\min C(\boldsymbol{w}) \quad \text{subject to} \quad \sum_{k=1}^{K} w_k = 1 \text{ and } 0 \leq w_k \leq 1,\ k = 1, \ldots, K. \tag{9}$$

A generalized representation of the optimization problem of the target criterion is shown in Fig. 1.

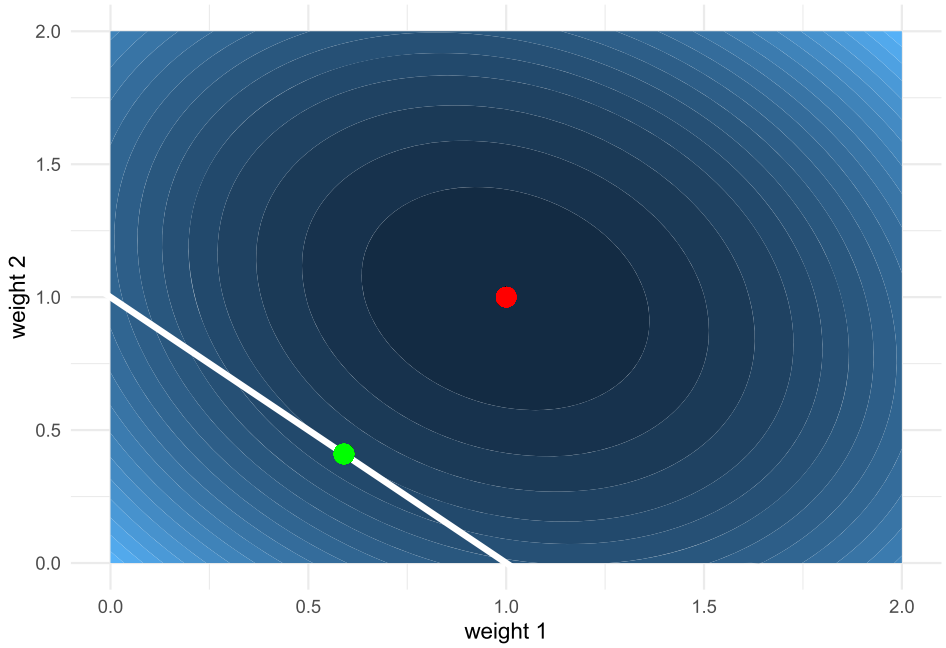

Fig. 1. All possible values of the target weight finding criterion (8) for an example with two candidate models. The red dot marks the global minimum, the white line the assumed possible maximum values of the weights, and the green dot the optimum of the weights resulting from the restrictions. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

To minimize a nonlinear problem such as (9), that is subject to equality and inequality constraints a more complex optimization approach has to be employed. The R-Packages Alabama (Varadhan, 2015) and Rsolnp (Ghalanos and Theussl, 2015) offer existing implementations of solvers for such a kind of optimization problem. The methods, however, implemented in the packages above require specific knowledge of the optimization methods and are further complicated by their broad general form. In response we propose a new implementation that is constructed in such a way that the setup, configuration and optimization of the underlying system can be performed easily.

4.1. The augmented Lagrange method with equality constraints

A common approach for solving such a problem is the augmented Lagrangian method (Hestenes, 1969; Powell, 1969). This approach adds a penalty term to the original target function, that represents a multiple of the constraint violations at each iteration and thus attempts to force the optimization result into the bound constraint's feasible solution space (Avriel, 2003). The augmented Lagrangian approach is based on the penalty method but aims to circumvent potential ill-conditioning inherent to these methods by directly integrating an estimator of the Lagrange multiplier into the target function (Nocedal and Wright, 2006). To introduce the augmented Lagrangian in its general form, let us assume a simple limited optimization problem with respect to a set of weight variables which, for the sake of simplicity, will all be assumed to be real-valued for the moment, i.e. $\boldsymbol{w} \in \mathbb{R}^K$ with following form of

$$\min C(\boldsymbol{w}) \quad \text{subject to} \quad h(\boldsymbol{w}) = 0, \tag{10}$$

where $C(\boldsymbol{w})$ is the cost function of the optimisation with $C : \mathbb{R}^K \to \mathbb{R}$ and the equality constraint functions $h = (h_1, \ldots, h_m)^T : \mathbb{R}^K \to \mathbb{R}^m$. The Lagrange function is described by

$$\mathcal{L}(\boldsymbol{w}, \boldsymbol{\lambda}) = C(\boldsymbol{w}) + \boldsymbol{\lambda}^T h(\boldsymbol{w}),$$

the Lagrange multiplier is given as $\boldsymbol{\lambda} \in \mathbb{R}^m$.

A possible problem with the general form of the objective function is that it does not necessarily have to be convex near the solution, which prevents duality methods like the Lagrangian from being effective. By adding the penalty term $\frac{\gamma}{2} h(\boldsymbol{w})^T h(\boldsymbol{w})$ with $\gamma > 0$, it is possible to impose a local convexity on the objective function, such that when the penalty γ term is sufficiently large, the Lagrangian will be locally convex (Luenberger et al., 2008). The resulting augmented Lagrange function $\mathcal{L}_A$ is defined as follows

$$\mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) \overset{\text{def}}{=} C(\boldsymbol{w}) + \boldsymbol{\lambda}^T h(\boldsymbol{w}) + \frac{\gamma}{2} h(\boldsymbol{w})^T h(\boldsymbol{w}).$$

The resulting problem is equivalent to the original problem (10), since the penalty term does not change the objective function, the Lagrange multiplier and the optimal values and solution point. This in turn allows to solve the underlying problem by an interactive process in $\boldsymbol{\lambda}$. The approach for the optimization of the augmented Lagrangian can be seen in Algorithm 1.

Algorithm 1: Augmented Lagrangian method.

Input : Initial weights $\boldsymbol{w}^0 \in \mathbb{R}^n$, tolerance η, multipliers $\boldsymbol{\lambda}^0$, penalty parameter $\gamma_0 > 0$, increment ϵ
Output: Optimal values $\boldsymbol{w}^\star$, multipliers $\boldsymbol{\lambda}^\star$, penalty parameter $\gamma^\star$

while $\|\nabla \mathcal{L}_A(\boldsymbol{w}^l, \boldsymbol{\lambda}^l; \gamma_l)\| > \eta$ **do**

solve for the target function with respect to $\boldsymbol{w}^{l+1}$ in a way that

$$\mathcal{L}_A(\boldsymbol{w}^{l+1}, \boldsymbol{\lambda}^l; \gamma_l) < \mathcal{L}_A(\boldsymbol{w}^l, \boldsymbol{\lambda}^l; \gamma_l)$$

update the Lagrange multipliers such that

$$\lambda^{l+1} = \lambda^l + h(\boldsymbol{w}^{l+1});$$

set the constraint γ such that

$$\gamma^{l+1} = \epsilon \gamma^l$$

set $l = l + 1$

end

4.2. Weight optimization via augmented Lagrangian

In its general form, the augmented Lagrangian only applies to equality constraint problems. To employ the method to our problem, the constraints have to be modified in such a fashion, that they include the bound constraints, i.e. the restrictions of the upper and lower limits of the possible weight values. The augmented Lagrange function subject to the problem at hand can be formulated as follows

$$\mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) = C(\boldsymbol{w}) + \boldsymbol{\lambda} h(\boldsymbol{w}) + \frac{\gamma}{2} h(\boldsymbol{w})^T h(\boldsymbol{w}).$$

The optimization under both constraints makes it necessary to divide the optimization into two different operations. In the first, the augmented Lagrangian is applied only to the equality condition i.e. the sum of all weights must add up to one. After an approximate solution for the problem has been found, the next part of the optimization is to incorporate the bound-constraints. The lower and upper value bound-constraints of the weights are here left out and are considered explicitly in an additional step of the optimization. In this sub-problem, a sequential quadratic programming approach is used to solve the following nonlinear quadratic problem of (Nocedal and Wright, 2006)

$$\min \mathcal{L}_A(\boldsymbol{w}, \boldsymbol{\lambda}; \gamma) \quad \text{subject to} \quad 0 \leq w_k \leq 1,\ k = 1, \ldots, K.$$

At each iteration step j, a quadratic problem $q_j(\boldsymbol{w})$ with fixed γ and $\boldsymbol{\lambda}$ stemming from the results of the first step of the optimization, is solved for $\boldsymbol{w}$ according to

$$\min\ q_j(\boldsymbol{w}) = \nabla \mathcal{L}_A(\boldsymbol{w}_j, \boldsymbol{\lambda}_j; \gamma)^T \left(\boldsymbol{w} - \boldsymbol{w}_j\right) + \frac{1}{2} \left(\boldsymbol{w} - \boldsymbol{w}_j\right)^T \mathcal{H} \left(\boldsymbol{w} - \boldsymbol{w}_j\right).$$

The Lagrangian's gradient assumes the form of

$$\begin{aligned}\nabla\mathcal{L}_A(\boldsymbol{w},\boldsymbol{\lambda};\gamma) &= (\nabla_{\boldsymbol{w}}\mathcal{L}_A(\boldsymbol{w},\boldsymbol{\lambda};\gamma), \nabla_{\boldsymbol{\lambda}}\mathcal{L}_A(\boldsymbol{w},\boldsymbol{\lambda};\gamma)) \\ &= \left(\nabla_{\boldsymbol{w}}\mathcal{L}_A(\boldsymbol{w},\boldsymbol{\lambda};\gamma), h(\boldsymbol{w})^T\right) \in \mathbb{R}^{1\times(K+m)},\end{aligned}$$

with $\nabla_{\boldsymbol{w}}\mathcal{L}_A(\boldsymbol{w},\boldsymbol{\lambda};\gamma) = \nabla C(\boldsymbol{w}) + (\boldsymbol{\lambda} + \gamma h(\boldsymbol{w}))^T \nabla h(\boldsymbol{w}) \in \mathbb{R}^{1\times K}$. The corresponding Hessian of the augmented Lagrangian can be formulated as a block matrix

$$\mathcal{H} = \nabla^2\mathcal{L}_A(\boldsymbol{w},\boldsymbol{\lambda};\gamma) = \begin{pmatrix} \nabla^2_{\boldsymbol{ww}}\mathcal{L}_A(\boldsymbol{w},\boldsymbol{\lambda};\gamma) & \nabla h(\boldsymbol{w})^T \\ \nabla h(\boldsymbol{w}) & 0 \end{pmatrix} \in \mathbb{R}^{(K+m)\times(K+m)},$$

where $\nabla_{\boldsymbol{ww}}\mathcal{L}_A(\boldsymbol{w},\boldsymbol{\lambda};\gamma) = \nabla^2 C(\boldsymbol{w},\boldsymbol{\lambda};\gamma) + (\boldsymbol{\lambda} + \gamma h(\boldsymbol{w}))^T \nabla^2 h(\boldsymbol{w}) + \gamma \nabla h(\boldsymbol{w})^T \nabla h(\boldsymbol{w}) \in \mathbb{R}^{K\times K}$.

The projection on the convex set $B = \{\boldsymbol{w} \mid a \le w_k \le b\}$ is described by $\mathcal{P}$, which is defined component-wise as

$$\mathcal{P}(w_k, a, b) = \begin{cases} a & \text{if } w_k \le a, \\ w_k & \text{if } w_k \in (a,b), \\ b & \text{if } w_k \ge b, \end{cases} \qquad \text{for all } k = 1, 2, \ldots, K.$$

such that given a vector $\boldsymbol{w} \in B$ and the imposed bound constraints for each weight, we can show that the defining necessary properties of an $\boldsymbol{w}$ to be considered the solution of such problem, as well as the needed first-order-condition, are given by adjusting the general Karush-Kuhn-Tucker conditions (Ruszczynski, 2011) such that

$$\boldsymbol{w} - \mathcal{P}(\boldsymbol{w} - \nabla_{\boldsymbol{w}}\mathcal{L}_A(\boldsymbol{w},\boldsymbol{\lambda};\gamma), 0, 1) = 0.$$

If the calculated weights meet both the equality and the bound constraints, the optimal solution is found. If, however, only the equality constraints are fulfilled by the resulting weights, the Lagrange multiplier estimates $\boldsymbol{\lambda}$ are adjusted to allow a better estimation in the next iteration. If the equality constraints are not met, the value of the penalty parameter γ is increased with the aim of forcing the results into the feasible space to minimize the constraint violations.

5. Simulation studies

To illustrate the features and capabilities of the model averaging approach presented above, we conduct three different simulation studies in which we investigate its finite-sample properties.

5.1. Augmented Langrangian weights for smoothing splines

In the first simulation setting we use the connection between mixed models and smoothing splines as presented in Section 2.2 and investigate the behaviour of the proposed methods presented from Section 4 on these types of models. Comparing parametric and semiparametric models by means of information criteria is often difficult. In particular due to their inherent flexibility, spline models will generally offer superior in-sample predictive capacity in comparison to standard linear models – at the cost of consuming a much higher number of degrees of freedom. Thus the question arises how the presented model weighting criterion (9) incorporates different linear and nonlinear candidate models.

For this purpose, we simulate data where the underlying data-generating model incorporates a quadratic P-spline term. Notice that we use P-splines instead of the truncated polynomial splines presented in Section 2.2 due to their enhanced numerical and computational stability (Eilers and Marx, 1996). Subsequently a linear model and a linear mixed model are fitted to the data, where the mixed model includes a spline term. The variance of the spline term takes on different increasing values for each simulation $\tau_b^2 \in \{0, 0.5, 1, \ldots, 9, 9.5, 10\}$. The variance of the residuals is kept constant, where each model combination is simulated for $\sigma_\varepsilon^2 \in \{1, 2, 4\}$. Every model combination is simulated 1000 times, with each simulation containing 100 observations.

Two models are fitted to the simulated data, a P-spline based semiparametric model (model 1 associated with w_1) of the form

$$\text{Model 1 } (w_1): y_i = \beta_0 + f(x_i) + \varepsilon_i, \tag{11}$$

which is fitted using a linear mixed model as described in Section 2.2 and a classical linear model (model 2 associated with w_2 = 1 - w_1)

$$\text{Model 2 } (w_2): y_i = \beta_0 + \beta_1 x_i + \varepsilon_i. \tag{12}$$

The proposed weight finding criterion (8) is applied to the two candidate models. Subsequently, the resulting weights are averaged over all simulations for each model constellation. The results of these calculations for the given combination of variances are shown in Fig. 2.

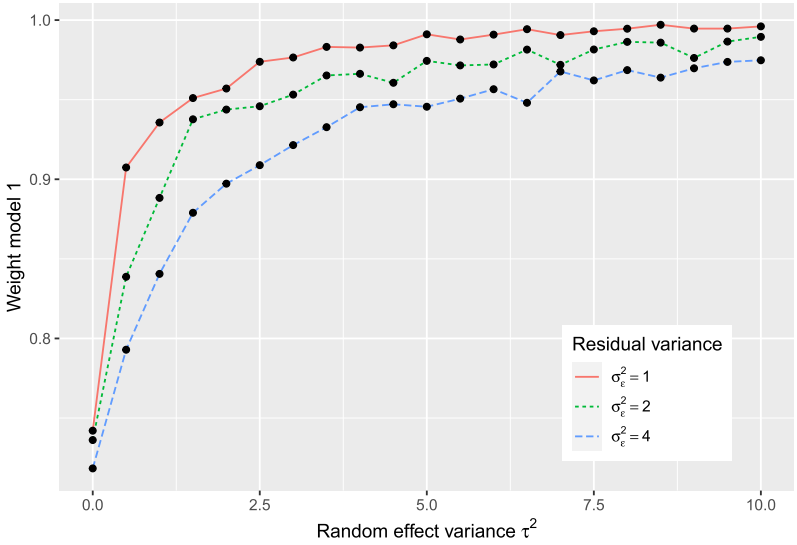

Fig. 2. Weight of model (11) associated with weight w_1 in comparison to the weight w_2 of model (12) for different random effects variance values and different values of the error variance.

It can be seen that with the increasing variance of the random effect (i.e. a decreasing penalty parameter) used to generate the data, the weight for the model with the spline element increases as well. This demonstrates that the improvement in the explanatory power of the spline model is detected by the proposed weight selection criterion and thus leads to a higher contribution of the semiparametric (or mixed) model to the resulting average model estimator. It also appears that given a higher residual variance the increase of the weights is slower, however, with a rising signal to noise ratio ($\tau^2/\sigma_\varepsilon^2$), we observe the anticipated shift towards a higher weight given to the semiparametric (or mixed) model.

5.2. Weights for multi-cluster hierarchical models

The second simulation study investigates the algorithm's behaviour in a multi-cluster mixed effects model framework as well as accounting for both non-normally distributed error terms entailing outliers and correlated error terms.

For the baseline data simulation, a true data generating linear mixed model is assumed that contains an intercept and two cluster levels with a random intercept each. The data generating model takes on the following form

$$y_{i,j,l} = \beta_0 + b_{1,j} + b_{2,l} + \varepsilon_{i,j,l}, \quad j,l = 1,\dots,10, \quad i = 1,\dots,100.$$

At first, the residuals and the random effects follow normal distributions, i.e. $\varepsilon_{i,j,l} \sim \mathcal{N}(0,\sigma^2)$, $b_{1,j} \sim \mathcal{N}(0,\tau_1^2)$ and $b_{2,l} \sim \mathcal{N}(0,\tau_2^2)$. For both levels the number of clusters in the true underlying model is 10 each. Each cluster consists out of 100 simulated individual observations. The random effects of the respective clusters are simulated such that the variances of the random effects fulfill $\tau_1^2 = 1 - \tau_2^2$. Each model setup is simulated 1000 times. Two linear mixed models are fitted to each dataset, whereby each model contains only one of the two random intercepts, i.e.

$$\text{Model 1 } (w_1): \quad y_{i,j} = \beta_0 + b_{1,j} + \varepsilon_{i,j},$$

and

$$\text{Model 2 } (w_2): \quad y_{i,l} = \beta_0 + b_{2,l} + \varepsilon_{i,l}.$$

The implemented weight choice criterion (8) is used to calculate an model average estimator based on the two candidate models. Table 1 and Fig. 3 show the results of the simulations for changing variances of the random effect and a constant residual variance.

Table 1
Calculated mean weights (and standard errors) for model 1 (w_1) for different given variance values.

τ_1^2	0	0.25	0.50	0.75	1
Mean	0.002	0.260	0.499	0.499	0.989
Std.Error	0.002	0.136	0.174	0.137	0.004

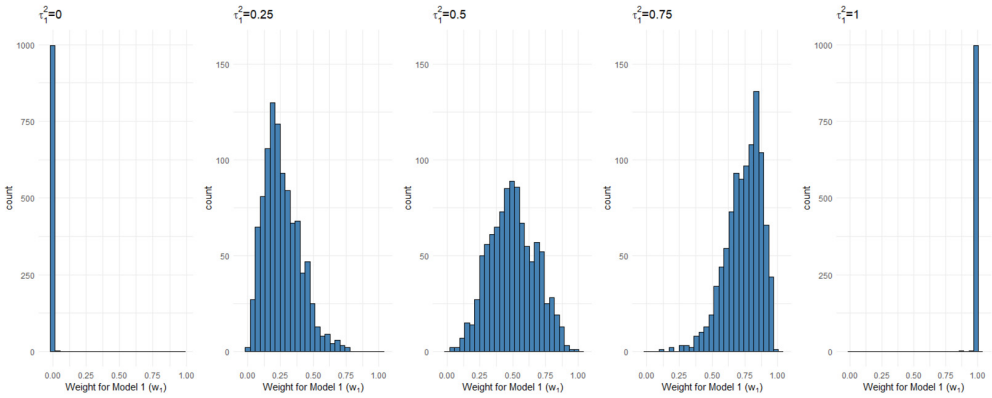

Fig. 3. Histograms of calculated weights for the first model for five different given variances of the random effects.

It becomes apparent that with an increasing variance of the first random effect, and therefore a decrease of the variance of the second random effect, the weight choice favours the first random effect. Furthermore, as can be seen in Table 1, the weights show an analogous behaviour in the case of a decrease of the first random effect variance. In the case that the variance of one of the two random effects is on the boundary, i.e. equal to zero, the corresponding weight is close to but not exactly zero. Furthermore the calculated weights show, that for $\tau_1 = \tau_2 = 0.5$ on average a model weight of 0.499 is chosen for model 1 and thus reflects the proportion of variation from the associated random effects.

Ultimately, it can be observed that the computed weights reflect the respective simulated random effect variance. This indicates that the presented model averaging estimator can recognize the information from the multi-level mixed model structure and consequently calculate weights that lead to a model that closely represents the underlying true data generating model.

The framework of linear mixed models assumes normally distributed random effects and errors terms allowing for computational more convenient approaches. Violations of these assumptions, such as those caused by outliers or serially correlated within-subject errors, lead to less robust models and unreliable inference results. To explore the extent to which the presented method is affected by these violations, two additional simulations are conducted.

The first simulation adapts the first simulation design, however, outliers are introduced by selecting one to 50 values of the first random effect, where the number of potential outliers is drawn from a discrete uniform distribution, and scaled with random draws from a continuous uniform distribution ranging from 3 to 5 such that the adapted random effect $\tilde{b}_{1,j}$ is given by $\tilde{b}_{1,j} = \psi b_{1,j}$, where $\psi \sim \mathcal{U}(3, 5)$.

In the second simulation the original simulation design is modified by including serial within-subject correlated errors. These are introduced by inducing a first order serial correlation with a correlation parameter of 0.5 such that $\tilde{\varepsilon}_{i,j,l} = 0.5\tilde{\varepsilon}_{i-1,j,l} + \varepsilon_{i,j,l}$. The results of both simulations are displayed in Figs. 4 and 5 as well as Tables 2 and 3.

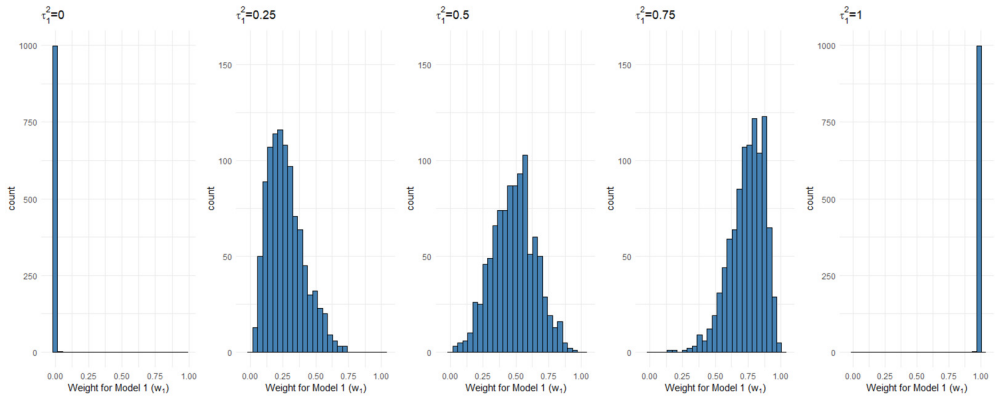

Fig. 4. Histograms of calculated weights for the first model for five different given variances of the random effects with simulated correlated within-subject errors for the first random effect.

The results indicate that the presented method is relatively robust to the portrayed violations of the standard assumptions. This primarily stems from the fact that our method examines the relative fits of the models in comparison to each other. If a model is less able to explain the underlying data, then that model receives a smaller weight. It should be noted

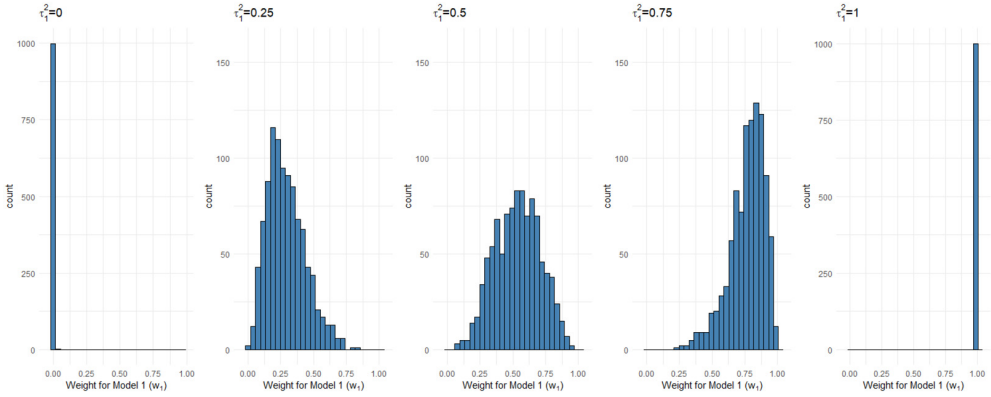

Fig. 5. Histograms of calculated weights for the first model for five different given variances of the random effects with simulated outliers for the first random effect.

Table 2
Calculated mean weights (and standard errors) for model 1 (w_1) for different given variance values with simulated correlated within-subject errors for the first random effect.

τ_1^2	0	0.25	0.50	0.75	1
Mean	0.001	0.271	0.482	0.742	0.989
Std.Error	0.001	0.135	0.163	0.131	0.003

Table 3
Calculated mean weights (and standard errors) for model 1 (w_1) for different given variance values with simulated outliers for the first random effect.

τ_1^2	0	0.25	0.50	0.75	1
Mean	0.001	0.292	0.534	0.765	0.989
Std.Error	0.001	0.145	0.172	0.134	0.003

that linear mixed models, which attempt to make the model class more robust against these influences, such as Ho and Lin (2010) are a valid alternative to the classic linear mixed model in these settings. However the distributions of these robust models deviate from the normality assumption, thus an extension of the proposed model averaging scheme especially for formula (7) needs to be found in order to overcome the misspecification.

5.3. Relationship between weights and fixed effects

In this section, we compare our method with other model averaging methods, whereby to allow for comparisons between our implementation with the concept of Zhang et al. (2014) we employ a design based on Example 1 of their paper. In contrast to the two previous simulation settings the focus of this simulation study lies on the calculated model average estimator and the accuracy of the method in comparison to already implemented approaches. For this analysis, data is generated by a data-generating model which contains three fixed effects in the form $\boldsymbol{\beta} = (1, 0.2, 0.4)$ whereby the j-th row of the $\boldsymbol{X}_i$ matrix takes the following form $(1, x_{i,j_2}, x_{i,j_3})$. The true underlying model also features three random effects, one random intercept and two random slopes. The elements of the j-th row of the $\boldsymbol{Z}_i$ matrix take the following form $(1, z_{i,j_2}, z_{i,j_3})$. The respective values of the design matrices $\boldsymbol{X}$ and $\boldsymbol{Z}$ originate independently from an $\mathcal{N}(0, 1)$ distribution. The underlying data-generating model has 20 groups with 10 observations each. The data is simulated with a standard deviation of the residuals of $\sigma \in \{0.3; 0.9\}$. Furthermore, each model combination is simulated with four different random effect co-variance matrices. These four matrices are as follows

$$\boldsymbol{D}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \boldsymbol{D}_2 = \begin{pmatrix} 1.4 & 0 & 0 \\ 0 & 1.2 & 0 \\ 0 & 0 & 0.4 \end{pmatrix},$$
$$\boldsymbol{D}_3 = \begin{pmatrix} 1.4 & 0.4 & 0 \\ 0.4 & 1.2 & 0 \\ 0 & 0 & 0.4 \end{pmatrix}, \quad \boldsymbol{D}_4 = \begin{pmatrix} 1.4 & 0.4 & 0.6 \\ 0.4 & 1.2 & 0.2 \\ 0.6 & 0.2 & 0.4 \end{pmatrix}.$$

These covariance structures were chosen to incorporate various levels of complex random effect structures into this study, and to therefore determine to what extent these influence the ability of the different methods of finding weights for

model averaging. For each of the configurations, 100 independent datasets are generated. The candidate models used for approximation include at least the fixed and the random intercept, the further candidate models include one of the two fixed coefficients and further random effects. Based on the fitted candidate models, weights for model averaging are now calculated as smooth weights as introduced in (6) based on the conditional AIC, as well as weights based on the presented asymptotic optimal approach (8). The ability of the different methods to calculate the model averaging approach is evaluated over the respective average squared loss.

Table 4
Simulation results: averaged squared losses. Asymptotically optimal model averaging and cAIC smoothed weights.

	σ	D_1	D_2	D_3	D_4
Asymptotic	0.3	323.751	264.324	274.753	256.289
	0.9	360.498	285.706	290.286	290.674
Smoothed	0.3	337.241	275.546	285.849	266.612
	0.9	373.372	296.281	298.929	300.176

Table 4 presents the calculated squared losses for the respective methods for each underlying covariance matrix and the residuals standard deviation. The model averaging approach presented and implemented in this work proves to be the superior method in terms of minimum average squared loss in all scenarios presented here.

6. Applications

In this section, we apply the proposed weight finding technique for model averaging on models fitted to two different real-world datasets. The first one is about the sensory assessment of TV characteristics. The second one is a common linear mixed model benchmarking dataset from an orthodontic study over time for several subjects.

6.1. Bang & Olufsen dataset

The first dataset was provided by the Danish electronics company Bang & Olufsen. Different characteristics of TV sets are measured using three response variables. The explanatory variables given are the TV set and image quality as measured and recorded by a panel of eight different assessors. See Kuznetsova et al. (2017) for the details of this study.

In the first application, we are interested in the influence of the explanatory variables on the response variable of the sharpness of motion and model it by means of random effects on the different assessors. To model the relationship we create three different linear mixed effects models, assuming that the response variable is influenced by the fixed effects of the TV set and the image quality, as well as an interaction of both. However, the three models differ in how the effects of the assessors are incorporated into the model. In the first model, it is assumed that there is a simple fixed random effect per assessor, in the second model a fixed random effect per assessor with an interaction effect between the TVs and the assessor is assumed and in the third, we assume an interaction between the assessor and the image quality. To compare the models, we calculate information criteria, the relative degrees of freedom and the mean squared error of the respective models. See Table 5 for the results.

Table 5
The values of various model choice criteria, smoothed weights based on the cAIC and weights resulting of the proposed model averaging estimator.

Model	rel. DF	cAIC	MSE	Weights	
				smoothed	asymptotic
1	30.47	864.88	3.806	0.000	0.327
2	36.48	844.10	3.188	1.000	0.673
3	19.61	864.18	4.213	0.000	0.000

If one decides to use the classical model selection approach of choosing the model with the smallest possible value for the information criterion under consideration, as well as a model averaging based on smooth weights, the second model with the assumed random effect relationship of an interaction between assessor and TV set would be chosen or respectively would receive a weight of one.

Fig. 6 shows the development of the weights and their trajectories during the optimization process. It can be observed that from the outset the weights quickly converge towards their final estimates. More specifically, from the tenth iteration onwards, the change of the weights becomes negligible. It should be noted that the starting values for all weights are set to be $w_i = 1/K$, K being the number of candidate models. By doing so, it is ensured that all weights are already within the feasible region of the optimization problem. Thereby, the equality constraints do not need to be explicitly enforced on the starting values and the algorithm can directly start minimizing the underlying weight choice function.

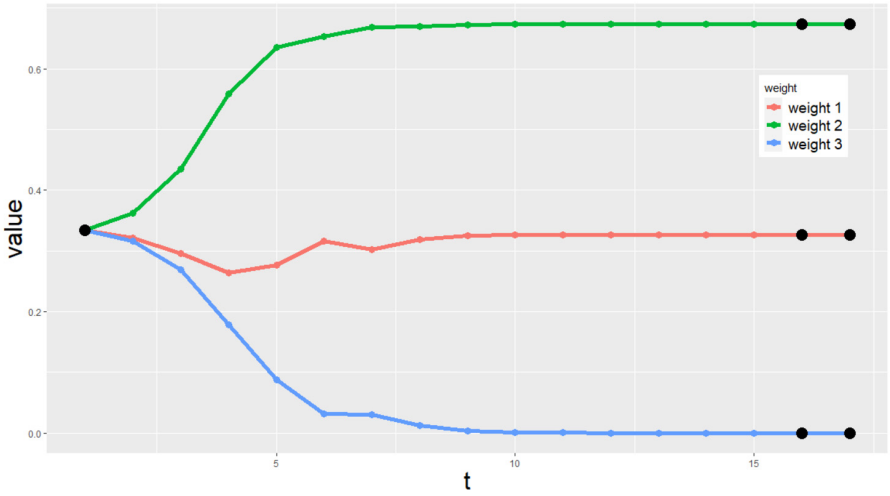

Fig. 6. Representation of the trajectories of the weights during the optimisation. Large black dots indicate weight values resulting from major iteration, small coloured dots indicate the results of a minor iteration.

The augmented Lagrangian is constructed as a robust method with regard to the starting values, see chapter 17.4 in Nocedal and Wright (2006). If the starting values are misspecified and as such chosen that they are not within the feasible region the convergence often takes longer. The authors however did not find any convergence problems leading to wrong solutions in the simulations and applications due to misspecified starting values.

We compare the results of our proposed method based on the mean squared error, with the results of a model averaging estimator based smoothed weights, as well as a model averaging estimator build upon the assumption of equal weights. The mean squared error is calculated by

$$MSE = \frac{1}{n}(\boldsymbol{y} - \hat{\boldsymbol{y}}(\hat{\boldsymbol{w}}))^T(\boldsymbol{y} - \hat{\boldsymbol{y}}(\hat{\boldsymbol{w}})),$$

where $\boldsymbol{y}$ represents the responses, $\hat{\boldsymbol{y}}(\hat{\boldsymbol{w}})$ the model averaging estimator based on the calculated weights and n the number of observations. The model average estimator based on the proposed method achieves the smallest MSE value of all with 3.153, whereas the estimator based on equal weighting and on conditional AIC based smoothed weights offer MSE values of 3.491 and 3.188, respectively.

6.2. Orthodont dataset

The second application uses candidate models based on the well known Orthodont dataset. The dataset stems from a study at the University of North Carolina Dental School following the growth of 27 children from the age of 8 until age 14. Every second year, the distance between the pituitary and the pterygomaxillary fissure was measured via X-ray examination. For more details see Potthoff and Roy (1964). For this application, we fit three different models. The first models the measured distance with the help of an intercept and the fixed effect of age, as well as a random intercept per individual. The second model extends the first model by including the fixed effect of gender. The third model introduces an additional fixed effect by including an interaction between age and gender. Model defining quantities such as the conditional Akaike information criterion, the relative degrees of freedom and the mean squared error of the respective models can be observed in Table 6. In addition, the table includes the calculated weights of our proposed method in addition to the smoothed weights based on the conditional AIC values of each candidate model.

Table 6
The values of various model choice criteria, smoothed weights based on the cAIC and weights resulting of the proposed model averaging estimator.

Model	rel. DF	cAIC	MSE	Weights	
				smoothed	asymptotic
1	27.12	405.47	1.463	0.910	0.839
2	25.92	411.23	1.582	0.051	0.000
3	26.57	411.79	1.569	0.039	0.161

The resulting estimator based on the proposed method provides the lowest overall MSE with an value of 1.462 of all model averaging estimators considered. The equal weights estimator has an MSE value of 1.509 and the smoothed weights

estimator has an MSE value of 1.463. This indicates a better ability of our methodology to evaluate, weigh and merge the underlying candidate models into a new model averaging estimator.

6.3. Computational aspects and suitability for applied users

To assess the computational requirements, all required calculations were executed a thousand times for both applications and the performance of the presented algorithm was contrasted with existent optimization routines in `R`. While the presented algorithm needs 10.764 milliseconds for the TVbo model set and 9.676 milliseconds on average for the Orthodont models, the general nonlinear optimiser `solnp` of the `Rsolnp` package (Ghalanos and Theussl, 2012) need 12.654 and 11.706 milliseconds. The nonlinear optimiser with constraints `constrOptim.nl` of the `alabama` package (Varadhan, 2015) requires an average of 35.18 milliseconds for the first application and 7.194 milliseconds for the second application.

In both applications, the method presented has either the least required computing time or one that is close to the fast method. In contrast to `solnp` and `constrOptim.nl`, the proposed algorithm does not require the user to provide any complex input of starting weights or the underlying gradient. Furthermore, in comparison to the much more general implementations of nonlinear optimizers, all necessary quantities and objects are automatically created and calculated in the background. This in turn allows the implementation to be more straightforward and convenient to use for researchers willing to employ the proposed approach for determining asymptotic optimal weights for model averaging of linear mixed models.

7. Outlook

Model choice for the class of linear mixed models plays an important role due to their wide distribution and application in different fields. Especially the question of including random effects plays a crucial part, which is complicated by the inherent problem of classical model choice methods concerning the underlying model assumptions. Thus, the use of classical information criteria such as AIC is discouraged due to the deviation from the classical model by the assumptions of the linear mixed model, while the usefulness of other methods such as likelihood-ratio test based approaches is impaired by the possibility of boundary issues. Therefore, due to its nature of combining different candidate models, the technique of model averaging presents an interesting alternative to model selection of linear mixed models.

On a technical level the choice of weights is critical for model averaging. As we have shown the proposed weight finding method by Zhang et al. (2014) of using the Steinian approximation of derivatives for an underlying weight criterion shows superior performance when compared to other approaches based on information criteria such as the conditional AIC. The proposed method is implemented as part of the R-Package `cAIC4` facilitating the use by applied researchers. Given that there is no universally applicable unbiased estimator of conditional AIC in analytical form without distributional assumptions (see Saefken et al., 2014), the proposed method stops short of offering an all model-class encompassing solution for model averaging. Such a generalisation would be especially valuable as further interesting models such as robust linear mixed models would fall under such an extended framework, see Lin and Lee (2006) and Ho and Lin (2010). Therefore misspecifications could be identified as in Bartolucci et al. (2017). This requires further research that we are planning to conduct in the future. An implementation of a criterion for finding a squared loss-optimal weights for generalised linear mixed models is another extension that is still required. A possible approach could be to use the methods proposed in Wood et al. (2016) for conditional model selection. A further possible extension could be to apply another error function than the squared error proposed in Zhang et al. (2014). Different possible error functions and the corresponding covariance penalties are presented in Säfken and Kneib (2020). This could be especially interesting for distributional regression models, see Kneib et al. (2021). Also an extension to boosting (Griesbach et al., 2021) would be interesting.

In terms of fields of applications, the proposed framework offers great potential for model averaging for applied researchers in order to offer more robust predictive capacity. One avenue which will be pursued by the authors of this paper is the use of model averaging in the context of epidemiological research along the lines of Silbersdorff et al. (2018).

Acknowledgements

We would like to sincerely thank the associate editor and the two anonymous reviewers for their comments and constructive inputs. Furthermore, we would like to thank Thomas Kneib for valuable feedback on the draft version of the paper.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

Akaike, H., 1973. Information Theory and an Extension of the Maximum Likelihood Principle, Proceedings of the 2nd International Symposium on Information. Czaki, Akademiai Kiado, Budapest.

Avriel, M., 2003. Nonlinear Programming: Analysis and Methods. Courier Corporation.

Bartolucci, F., Bacci, S., Pigini, C., 2017. Misspecification test for random effects in generalized linear finite-mixture models for clustered binary and ordered data. Econom. Stat. 3, 112–131. https://doi.org/10.1016/j.ecosta.2016.11.007. https://www.sciencedirect.com/science/article/pii/S2452306216300314.
Bates, D., Alday, P., Kleinschmidt, D., Calderón, J.B.S., Noack, A., Kelman, T., Bouchet-Valat, M., Gagnon, Y.L., Babayan, S., Mogensen, P.K., Piibeleht, M., Hatherly, M., Saba, E., Baldassari, A., 2020. Juliastats/mixedmodels.jl, v2.3.0. https://doi.org/10.5281/zenodo.3727845.
Bates, D., Mächler, M., Bolker, B., Walker, S., 2014. Fitting linear mixed-effects models using lme4. J. Stat. Softw. 67, 1–48. https://doi.org/10.18637/jss.v067.i01.
Buckland, S.T., Burnham, K.P., Augustin, N.H., 1997. Model selection: an integral part of inference. Biometrics, 603–618.
Cantoni, E., Jacot, N., Ghisletta, P., 2021. Review and comparison of measures of explained variation and model selection in linear mixed-effects models. Econom. Stat. https://doi.org/10.1016/j.ecosta.2021.05.005. https://www.sciencedirect.com/science/article/pii/S2452306221000630.
Crainiceanu, C.M., Ruppert, D., 2004. Likelihood ratio tests in linear mixed models with one variance component. J. R. Stat. Soc., Ser. B, Stat. Methodol. 66, 165–185.
Di, C.Z., Crainiceanu, C.M., Caffo, B.S., Punjabi, N.M., 2009. Multilevel functional principal component analysis. Ann. Appl. Stat. 3, 458.
Efron, B., 2004. The estimation of prediction error: covariance penalties and cross-validation. J. Am. Stat. Assoc. 99, 619–632.
Eilers, P.H.C., Marx, B.D., 1996. Flexible smoothing with b -splines and penalties. Stat. Sci. 11, 89–121. https://doi.org/10.1214/ss/1038425655.
Fahrmeir, L., Kneib, T., Lang, S., Marx, B., 2013. Regression. Springer Berlin Heidelberg, Berlin, Heidelberg.
Ghalanos, A., Theussl, S., 2012. Package 'rsolnp'.
Ghalanos, A., Theussl, S., 2015. Rsolnp: General Non-linear Optimization Using Augmented Lagrange Multiplier Method. R package version 1.16.
Greven, S., Kneib, T., 2010. On the behaviour of marginal and conditional aic in linear mixed models. Biometrika 97, 773–789. https://doi.org/10.1093/biomet/asq042.
Griesbach, C., Säfken, B., Waldmann, E., 2021. Gradient boosting for linear mixed models. Int. J. Biostat. 20200136. https://doi.org/10.1515/ijb-2020-0136.
Guo, W., 2002. Functional mixed effects models. Biometrics 58, 121–128.
Harville, D., et al., 1976. Extension of the gauss-markov theorem to include the estimation of random effects. Ann. Stat. 4, 384–395.
Henderson, C.R., 1950. Estimation of genetic parameters. In: Biometrics, International Biometric Soc 1441 I ST, NW, Suite 700, Washington, DC 20005-2210, pp. 186–187.
Hestenes, M.R., 1969. Multiplier and gradient methods. J. Optim. Theory Appl. 4, 303–320.
Ho, H.J., Lin, T.I., 2010. Robust linear mixed models using the skew t distribution with application to schizophrenia data. Biom. J. 52, 449–469.
Kneib, T., Silbersdorff, A., Säfken, B., 2021. Rage against the mean – a review of distributional regression approaches. Econom. Stat. https://doi.org/10.1016/j.ecosta.2021.07.006. https://www.sciencedirect.com/science/article/pii/S2452306221000824.
Kullback, S., Leibler, R.A., 1951. On information and sufficiency. Ann. Math. Stat. 22, 79–86.
Kuznetsova, A., Brockhoff, P.B., Christensen, R.H.B., 2017. lmerTest package: tests in linear mixed effects models. J. Stat. Softw. 82.
Laird, N.M., Ware, J.H., 1982. Random-effects models for longitudinal data. Biometrics, 963–974.
Li, C., Yin, W., Jiang, H., Zhang, Y., 2013. An efficient augmented lagrangian method with applications to total variation minimization. Comput. Optim. Appl. 56, 507–530.
Liang, H., Wu, H., Zou, G., 2008. A note on conditional aic for linear mixed-effects models. Biometrika 95, 773–778. https://doi.org/10.1093/biomet/asn023.
Lin, T.I., Lee, J.C., 2006. A robust approach to t linear mixed models applied to multiple sclerosis data. Stat. Med. 25, 1397–1412.
Luenberger, D.G., Ye, Y., et al., 2008. Linear and Nonlinear Programming, Vol. 2. Springer.
Nocedal, J., Wright, S., 2006. Numerical Optimization. Springer Science & Business Media.
Potthoff, R.F., Roy, S., 1964. A generalized multivariate analysis of variance model useful especially for growth curve problems. Biometrika 51, 313–326.
Powell, M.J., 1969. A method for nonlinear constraints in minimization problems. Optimization, 283–298.
R Core Team, 2019. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.
Ruppert, D., Wand, M.P., Carroll, R.J., 2003. Semiparametric Regression. 12. Cambridge University Press.
Ruszczynski, A., 2011. Nonlinear Optimization. Princeton University Press.
Saefken, B., Kneib, T., van Waveren, C.S., Greven, S., 2014. A unifying approach to the estimation of the conditional akaike information in generalized linear mixed models. Electron. J. Stat. 8, 201–225. https://doi.org/10.1214/14-EJS881.
Säfken, B., Rügamer, D., Kneib, T., Greven, S., 2018. Conditional model selection in mixed-effects models with caic4. http://arxiv.org/pdf/1803.05664v2.
Schwarz, G., et al., 1978. Estimating the dimension of a model. Ann. Stat. 6, 461–464.
Self, S.G., Liang, K.Y., 1987. Asymptotic properties of maximum likelihood estimators and likelihood ratio tests under nonstandard conditions. J. Am. Stat. Assoc. 82, 605–610.
Silbersdorff, A., Lynch, J., Klasen, S., Kneib, T., 2018. Reconsidering the income-health relationship using distributional regression. Health Econ. 27, 1074–1088.
Stein, C., et al., 1972. A bound for the error in the normal approximation to the distribution of a sum of dependent random variables. In: Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Probability Theory. The Regents of the University of California.
Säfken, B., Kneib, T., 2020. Conditional covariance penalties for mixed models. Scand. J. Stat. 47, 990–1010.
Vaida, F., Blanchard, S., 2005. Conditional akaike information for mixed-effects models. Biometrika 92, 351–370. https://doi.org/10.1093/biomet/92.2.351.
Varadhan, R., 2015. Alabama: constrained nonlinear optimization. https://CRAN.R-project.org/package=alabama. r package version 2015.3-1.
Verbeke, G., Molenberghs, G., 2009. Linear Mixed Models for Longitudinal Data. Springer Science & Business Media.
Wager, C., Vaida, F., Kauermann, G., 2007. Model selection for penalized spline smoothing using akaike information criteria. Aust. N. Z. J. Stat. 49, 173–190.
Wood, S.N., 2013. A simple test for random effects in regression models. Biometrika 100, 1005–1010.
Wood, S.N., 2017. Generalized additive models: an introduction with R. CRC press.
Wood, S.N., Pya, N., Säfken, B., 2016. Smoothing parameter and model selection for general smooth models. J. Am. Stat. Assoc. 111, 1548–1563.
Ye, J., 1998. On measuring and correcting the effects of data mining and model selection. J. Am. Stat. Assoc. 93, 120–131.
Zhang, X., Zou, G., Liang, H., 2014. Model averaging and weight choice in linear mixed-effects models. Biometrika 101, 205–218. https://doi.org/10.1093/biomet/ast052.

Appendix B: Supplemental material Part III

B.1 Supplemental material for NAMLSS

B.1.1 Log-likelihoods

As the presented method minimizes negative log-likelihoods, we created a comprehensive list of all the log-likelihoods of the distributions used in the paper.

(Bernoulli) Logistic distribution The log-likelihood for a logistic distribution is given by

$$\log\left(\mathcal{L}(\mu,\sigma|y)\right) = \sum_{i=1}^{n}\left[y_i \log(\frac{1}{1+e^{-(\frac{y_i-\mu}{\sigma})}}) + (1-y_i)\log(1-\frac{1}{1+e^{-(\frac{y_i-\mu}{\sigma})}})\right],$$

with n number of observations and the parameters location $\mu \in \mathbb{R}$, scale $\sigma \in \mathbb{R}^+$ and $x \in \mathbb{R}$.

Binomial distribution The log-likelihood function for a binomial distribution is given by

$$\log\left(\mathcal{L}(k|n,p)\right) = k\log(p) + (n-k)\log(1-p) + \log\binom{n}{k},$$

where n is the number of trials, the parameters success probability is given by $p \in [0,1]$ and the number of successes is denoted as $k \in \mathbb{N}_0$.

Inverse Gamma distribution The log-likelihood function of the invers gamma distribution is

$$\log\left(\mathcal{L}(\alpha,\beta|y)\right) = -n\left(\alpha+1\right)\overline{\log \boldsymbol{y}} - n\log\Gamma(\alpha) + n\alpha\log\beta - \sum_{i=1}^{n}\beta y_i^{-1}.$$

with $\alpha > 0$ and $\beta > 0$ and where the upper bar operand indicates the arithmetic mean

Normal distribution The log-likelihood function for a normal distribution is given by

$$log\left(\mathcal{L}(\mu,\sigma^2|y)\right) = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-\mu)^2,$$

where n is the underlying number of observations and parameters $y \in \mathbb{R}$, location $\mu \in \mathbb{R}$ and scale $\sigma \in \mathbb{R}^+$.

Inverse Gaussian distribution The log-likelihood function of the inverse Gaussian distribution is given by

$$\log\left(\mathcal{L}(\mu,\sigma|x)\right) = \frac{n}{2}\ln(\sigma) - \sum_{i=1}^{n}\frac{\sigma(x_i-\mu)^2}{2\mu^2 x_i},$$

with n is as the number of observations and the parameters location $\mu \in \mathbb{R}^+$, scale $\sigma \in \mathbb{R}^+$ and $x \in \mathbb{R}^+$.

Poisson distribution The log-likelihood function for a Poisson distribution with parameter λ is given by

$$\log\left(\mathcal{L}(\lambda|x)\right) = \sum_{i=1}^{n}[x_i\log(\lambda) - \lambda - \log(x_i!)]$$

where $x = (x_1, x_2, ..., x_n)$ is the sample, n is the number of observations and x_i are non-negative integers.

Johnson's S_U The log-likelihood function of the Johnson's S_U distribution is defined as

$$\log\left(\mathcal{L}(\beta,\omega,\mu,\sigma|y)\right) = n\log\left[\frac{\beta}{\omega\sqrt{2\pi}}\right] - \frac{\beta^2}{2\omega^2}\sum_{i=1}^{n}\left[\frac{(y_i-\mu)^2}{\sigma^2} + \ln\left(1+\frac{(y_i-\mu)^2}{\omega^2\sigma^2}\right)\right],$$

with n is as the number of observations and the parameters location $\mu \in \mathbb{R}$, scale $\sigma \in \mathbb{R}^+$, shape $\omega \in \mathbb{R}^+$, skewness $\beta \in \mathbb{R}$ and $y \in \mathbb{R}$.

Weibull distribution The log-likelihood function of the Weibull distribution is defined as

$$\log\left(\mathcal{L}(\lambda,\beta,|y)\right) = n\ln\beta - n\beta\ln\lambda - \sum_{i=1}^{n}\left(\frac{y_i}{\lambda}\right)^{\beta} + (\beta-1)\sum_{i=1}^{n}\ln y_i,$$

with n is the number of observations and with the location $\lambda \in \mathbb{R}^+$, the shape $\beta \in \mathbb{R}^+$ and $y \in \mathbb{R}^+$.

B.1.2 Activation functions

For DDNN and NAMLSS, independent of the implementation, we use a Softplus activation for the scale parameter σ^2 to ensure non-negativity and a linear activation for the mean μ.

For the AirBnB datasets, also analyzed by Rügamer et al. (2020), we assume an inverse Gamma distribution $\mathcal{IG}(\alpha, \beta)$ as the underlying data distribution (see equation (B.1.1) for the log-likelihood). For NAMLSS as well as DDNN we have to adjust the activation functions, as both models minimize the log-likelihood via the parameters α and β. However, the mean prediction resulting from these parameters is defined via

$$\mu = \frac{\beta}{\alpha - 1}$$

and is hence only defined for $\alpha > 1$. The activation functions thus need to ensure an α prediction that is larger than 1 and a β prediction that is larger than 0. Hence we again use a Softplus activation for the β output layer [6]. For the α prediction, we use the following activation function element-wise

$$h(x) = \begin{cases} \log(1 + \exp(x)), & \text{if } \log(1 + \exp(x)) > 1, \\ \frac{1}{\log(1+\exp(x))}, & \text{else.} \end{cases} \tag{B.1}$$

To compute the log-likelihood for the models resulting in a mean prediction we compute the parameters α and β as follows

$$\alpha = \frac{\mu^2}{\sigma^2 + 2},$$

$$\beta = \mu \frac{\mu^2}{\sigma^2 + 1},$$

with σ^2 denoting the variance of the mean predictions. For XGBoost and EBM we use a simple transformation of the target variable to ensure that $\mu > 0$. Hence we fit the model on $\log(y)$ and re-transform the predictions accordingly with $\exp(\hat{y})$. For a (binary) classification benchmark we use the FICO dataset (FICO, 2018), the Shrutime dataset and the Telco dataset. A logistic distribution, $\mathcal{LO}(\mu, s)$, of the underlying response variable was assumed (see equation (B.1.1) for the log-likelihood). Again, we use the true standard deviation of the underlying data for the models only resulting in a mean prediction. The models resulting in a mean prediction use binary cross-entropy as the loss function and hence a sigmoid activation function on the output layer.

[6] Interestingly, the NAM did not converge using the Softplus activation function as the MLP did. Using the Softplus activation resulted in tremendously large mean gamma deviances and log-likelihoods, as the model kept predicting values that were nearly zero. Hence, we were only able to achieve good results for the NAM using the activation function given by formula (B.1).

B.1.3 Network architecture

We propose two different network architectures that can both flexibly model all distributional parameters. The first one is depicted in Figure 6.14 and creates J subnetworks for each distributional parameter. Each distributional subnetwork is comprised of the sum of $f_j^{(k)}$. Hence we create $K \times J$ subnetworks. To account for distributional restrictions, each distributional subnetwork is specified with possibly differing activation functions in the output layer.

The second model architecture is depicted in Figure B.1. Here we only create J subnetworks and hence have the same amount of subnetworks as a common NAM. Each subnetwork then has a k-dimensional output layer. Each distributional Parameter, $\theta^{(k)}$, is subsequently obtained by summing over the k-th output of the J subnetworks. Each dimension in the output layer can be activated using different activation functions, adjusting to parameter restrictions.

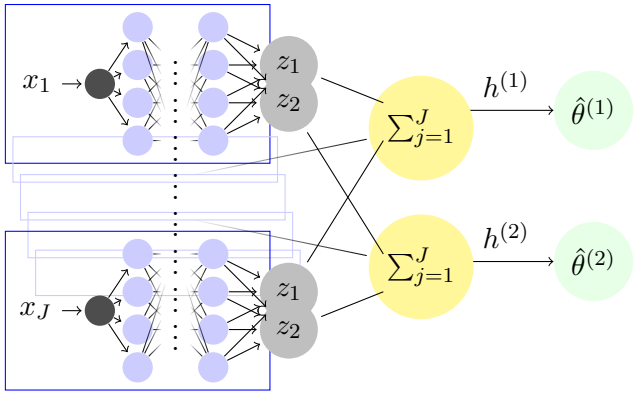

Figure B.1: The network structure of a simple NAMLSS model. Each input variable as well as each distributional parameter is handled by a different neural network. h_k are different activation functions depending on the distributional parameter that is modelled. E.g. a quadratic transformation for modelling the variance in a normally distributed variable to ensure the non-negativity constraint.

B.1.4 Benchmarking

The benchmark study for used real-world datasets was performed under similar conditions. All datasets are publicly available and we describe every pre-processing step as well as all model specifications in detail in the following.

B.1.5 Synthetic data generation

For the simulation of the data, respectively their underlying distribution parameters $\theta = \left(\theta^{(1)}, \theta^{(2)}, \theta^{(3)}, \theta^{(4)}\right)$, the following assumptions are made

$$
\begin{aligned}
\theta^{(1)} &= \frac{30}{13} x_1 \left((3x_2 + 1.5) - 2 \sin\left(\frac{x_3}{2}\right) \right)^{-1} + \frac{113}{115} x_4 + 0.1 x_5, \\
\theta^{(2)} &= \exp\left(-0.0035 x_1 + (x_2 - 0.23)^2 - 1.42 x_3\right) + 0.0001 x_4, \\
\theta^{(3)} &= \frac{1}{42}(4x_1 - 90x_2), \\
\theta^{(4)} &= exp\left(0.0323 * x_2 + 0.0123 - 0.0234 * x_4\right),
\end{aligned}
$$

where each of the five input vectors x_j is sampled from a uniform distribution $\mathcal{U}(0, 1)$, with a total of $n = 3000$ observations per data set.

Preprocessing

We implement the same preprocessing for all used datasets and only slightly adapt the preprocessing of the target variable for the two regression problems, California housing and Insurance. We closely follow Gorishniy et al. (2021) in their preprocessing steps and use the preprocessing also implemented by Agarwal et al. (2021). Hence all numerical variables are scaled between -1 and 1, all categorical features are one-hot encoded. In contrast to Gorishniy et al. (2021) we do not implement quantile smoothing, as one of the biggest advantages of neural models is the capability to model jagged shape functions. We use 5-fold cross-validation and report mean results as well as the standard deviations over all datasets. For reproducibility, we use the sklearn (Pedregosa et al., 2011) Kfold function with a random state of 101 and shuffle equal to true for all datasets. For the two regression datasets, we implement a standard normal transformation of the target variable. This results in better performances in terms of log-likelihood for all models only predicting a mean and is hence even disadvantageous for the presented NAMLSS framework.

Datasets

California housing The California housing (CA Housing) dataset Pace and Barry (1997) is a popular publicly available dataset and was obtained from sklearn Pedregosa et al. (2011). It is also used as a benchmark in Agarwal et al. (2021) and Gorishniy et al. (2021) and we achieve

Table B.1: Statistics of the benchmarking datasets.

Dataset	No. Samples	No. Features	Distribution	Task
California housing	20640	8	Normal $\mathcal{N}(\mu, \sigma)$	Regression
Insurance	1338	6	Normal $\mathcal{N}(\mu, \sigma)$	Regression
Abalone	4177	10	Normal $\mathcal{N}(\mu, \sigma)$	Regression
Munich	4568	9	Inverse Gamma $\mathcal{IG}(\alpha, \beta)$	Regression
Melbourne	16868	11	Inverse Gamma $\mathcal{IG}(\alpha, \beta)$	Regression
Fico	10459	23	Logistic $\mathcal{LO}(\mu, s)$	Classification
Shrutime	10000	10	Logistic $\mathcal{LO}(\mu, s)$	Classification
Telco	7032	19	Logistic $\mathcal{LO}(\mu, s)$	Classification

similar results concerning the MSE for the models which were used in both publications. The dataset contains the house prices for California homes from the U.S. census in 1990. The dataset is comprised of 20640 observations and besides the logarithmic median house price of the blockwise areas as the target variable contains eight predictors. As described above, we additionally standard normalize the target variable. All other variables are preprocessed as described above.

Insurance The Insurance dataset is another regression type dataset for predicting billed medical expenses (Lantz, 2019). The dataset is publicly available in the book *Machine Learning with R* by Lantz (2019). Additionally, the data is freely available on Github and Kaggle. It is a small dataset with only 1338 observations. The target variable is *charges*, which represents the *Individual medical costs billed by health insurance*. Similar to the California housing regression we standard normalize the response. Additionally, the dataset includes 6 feature variables. They are preprocessed as described above, which, due to one-hot encoding leads to a feature matrix with 9 columns.

Abalone The Abalone dataset contains information for the prediction of the age of abalone, a type of sea snail, based on their physical measurements. The data set is taken from the original publication (Nash et al., 1994) and today is a part of UCI Machine Learning Repository. A dataset of 4177 observations, 10 features and one response variable is obtained after processing the data.

Munich For the AirBnB data, we orientate on Rügamer et al. (2020) and used the data for the city of Munich. The dataset is also publicly available and was taken from Inside AirBnB on January 15, 2023. After excluding the variables *ID*, *Name*, *Host ID*, *Host Name*, *Last Review* and after removing rows with missing values the dataset contains 4568 observations. Additionally, we drop the *Neighbourhood* variable as firstly the predictive power of that variable is limited at best and secondly not to create too large feature matrices for GAMLSS. Hence, in addition to the target

variable, the dataset contains 9 variables. All preprocessing steps are subsequently performed as described above and the target variable, *Price*, is not preprocessed at all.

Melbourne The dataset is also publicly available and was taken from Inside AirBnB. The second Airbnb dataset (Melbourne) originates from the same source as the Munich Airbnb dataset. The data processing follows the same procedure as described in the Munich section. All preprocessing steps are then performed as described above and the target variable *Price* is not preprocessed at all.

FICO Similar to Agarwal et al. (2021) we also use the FICO dataset for our benchmarking study. However, we use it as described on the website and hence use the *Risk Performance* as the target variable. A detailed description of the features and their meaning is available at the Explainable Machine Learning Challenge. The dataset is comprised of 10459 observations. We did not implement any preprocessing steps for the target variable.

Shrutime This dataset contains information on the customers of a bank and the target variable is a binary variable reflecting whether the customer has left the bank (closed his account) or remains a customer. The corresponding data set can be found at Kaggle and is introduced by Kaggle (2019). After the processing described above, the set consists of 10000 observations, each with 10 features.

Telco The Telco customer churn data contains information about a fictitious telco company that provided home phone and internet services to 7043 customers. It details which customers left, stayed or signed up for their service. Several key demographics are included for each customer, as well as a satisfaction score, a churn score and a customer lifetime value (CLTV) index and was introduced by IBM (2019). After the processing described above, the set consists of 7043 observations, each with 19 features.

Model architectures and hyperparameters

As we do not implement extensive hyperparameter tuning for the presented NAMLSS framework, we do not perform hyperparameter tuning for the comparison models. We fit all models without an intercept. However, we try to achieve the highest comparability by choosing similar modelling frameworks, network architectures and hyperparameters where possible. All neural models are hence fit with identical learning rates, batch sizes, hidden layer sizes, activation functions and

regularization techniques. Through all neural models and all datasets, we use the ADAM optimizer (Kingma and Ba, 2014) with a starting learning rate of 1e-04. For the larger datasets, *California housing*, *Abalone*, *FICO*, *Telco* and *Shrutime* we orient on Agarwal et al. (2021) and use larger batch sizes of 1024. For the smaller dataset, *Insurance*, we use a smaller batch size of 256 and for the *Munich* and *Melbourne* dataset we use a batch size of 512. For every dataset and for every neural model, the maximum number of epochs is set to 2000. However, we implement early stopping with a patience of 150 epochs and no model over no fold and no dataset ever trained for the full 2000 epochs. Additionally, we reduce the learning rate with a factor of 0.95 with patience of 10 epochs for all models for all datasets. We use the rectified linear unit (ReLU) activation function for all hidden layers for all models

$$h(x) = \begin{cases} 0, & x < 0 \\ x, & \text{else.} \end{cases}$$

We also experimented with the Exponential centred hidden Unit (ExU) activation function presented by Agarwal et al. (2021) but found no improvement in model performance and even a slight deterioration for most models.

For the statistical models used from the GAMLSS and gamboostLSS frameworks, we do not optimize the model hyperparameters, as with neural networks. We use the respective default settings unless otherwise stated in the modelling descriptions included in the Appendix. We try to keep the model settings equal between all models, if applicable. All GAMLSS models use the same RS solver proposed by Rigby and Stasinopoulos (2005), in cases where this approach does not lead to convergence, the alternative CG solver presented by Cole and Green (1992) is employed. To exclude possible numerical differences, the same distributions from the GAMLSS R package are used for modelling the response distribution and calculating the log-likelihoods. gamboostLSS allows the use of different boosting approaches. Here we use the implemented boosting methods based on GAMs and GLMs and choose the model that performs better in terms of log-likelihood and the assumed loss.

California housing and Abalone We orient again on Agarwal et al. (2021) and use the following hidden layer sizes for all networks: $[1000, 500, 100, 50, 25]$. The second hidden layer is followed by a 0.25 dropout layer. While subsequently the NAM and NAMLSS have much more trainable parameters than the MLP and the DNN, we find that the MLP and DNN outperform the NAM and NAMLSS in terms of mean prediction. Additionally, we encountered severe overfitting when

Table B.2: Hyperparameters for the neural models for the California housing and the Abalone dataset

Hyperparameter	NAMLSS[1]	NAMLSS[2]	DNN	MLP	NAM
Learning rate	1e-04	1e-04	1e-04	1e-04	1e-04
Dropout	0.25	0.25	0.25	0.25	0.25
Hidden layers	[1000, 500, 100, 50, 25]	[1000, 500, 100, 50, 25]	[1000, 500, 100, 50, 25]	[1000, 500, 100, 50, 25]	[1000, 500, 100, 50, 25]
LR decay, patience	0.95 - 10	0.95 - 10	0.95 - 10	0.95 - 10	0.95 - 10
Activation	ReLU	ReLU	ReLU	ReLU	ReLU
Output activation	Linear, Softplus	Linear, Softplus	Linear, Softplus	Linear	Linear

[1] With 2 $\times$8 subnetworks. See Table 6.14 for an exemplary network structure.

[2] With 8 subnetworks and each subnetwork returning a parameter for the location and shape respectively. See Table B.1 for an exemplary network structure.

using the same number of parameters in an MLP as in the NAM and NAMLSS implementation. For the mean predicting models, we use a one-dimensional output layer with a linear activation. For the DNN and both NAMLSS implementations, we use a linear activation over the mean prediction and a Softplus activation for the variance prediction with

$$h(x) = \log(1 + \exp(x)).$$

For the NAMLSS implementation depicted in Figure 6.14 we use a smaller network structure for predicting the variance with two hidden layers of sizes 50 and 25 without any form of regularization as Dürr et al. (2020) found that using smaller networks for predicting the scale parameters is sufficient. For XGBoost we use the default parameters from the Python implementation. For the explainable boosting machines, we increased the number of maximum epochs to the default value of 5000 but set the early stopping patience considerably lower to 10, as otherwise, the model reached far worse results compared to the other models. We additionally increased the learning rate to 0.005 compared to the learning rate used in the neural approaches as a too small learning rate resulted in bad results. Otherwise, we kept all other hyperparameters as the default values. The GAMLSS and gamboostLSS models assume a normal distribution, with a location estimator μ employing an identity link and a scale estimator σ with a log-link function. Due to numerical instabilities, we choose to use the GLM-based boosting method instead of the default GAM-based version.

Insurance As the insurance dataset is considerably smaller than all other datasets we use slightly different model structures, as the model structure used for the California housing and Abalone

Table B.3: Hyperparameters for the neural models for the Insurance dataset

Hyperparameter	NAMLSS[1]	NAMLSS[2]	DNN	MLP	NAM
Learning rate	1e-04	1e-04	1e-04	1e-04	1e-04
Dropout	0.5	0.5	0.5	0.5	0.5
Hidden layers	$[250, 50, 25]$	$[250, 50, 25]$	$[250, 50, 25]$	$[250, 50, 25]$	$[250, 50, 25]$
LR decay, patience	0.95 - 10	0.95 - 10	0.95 - 10	0.95 - 10	0.95 - 10
Activation	ReLU	ReLU	ReLU	ReLU	ReLU
Output activation	Linear, Softplus	Linear, Softplus	Linear, Softplus	Linear	Linear

[1] With 2 $\times$9 subnetworks. See Table 6.14 for an exemplary network structure.

[2] With 9 subnetworks and each subnetwork returning a parameter for the location and shape respectively. See Table B.1 for an exemplary network structure.

datasets led to worse results. Hence, for all neural models, we use hidden layers of sizes $[250, 50, 25]$. The first layer is followed by a 0.5 dropout layer. Again, we use a simple linear activation for the models only predicting the mean and a linear and a Softplus activation for the models predicting the mean and the variance respectively. For the first NAMLSS implementation (see Figure 6.14) we again use a smaller network for predicting the variance with just one hidden layer with 50 neurons.

For XGBoost and EBM we use the same hyperparameter specifications as for the California housing and Abalone datasets.

The GAMLSS and gamboostLSS models assume a normal distribution, with a location estimator μ employing an identity link and a scale estimator σ with a log-link function. The boosting for location, scale and shape method employed uses the GLM based, instead of the GAM, based version.

FICO, Telco and Shrutime For the logistic datasets, we use the exact same model structure as for the Insurance dataset, as the model structures implemented for the California housing dataset resulted in worse results. However, as it is a binary classification problem we use a Sigmoid activation for the MLP as well as the NAM. For the DNN and both NAMLSS implementations, we use a Sigmoid activation for the location and a Softplus activation for the scale. To generate the log-likelihoods for the models only predicting a mean, we again use the true standard deviation of the underlying data.

For XGBoost and EBM we had to adjust the hyperparameters in order to get results comparable to the MLP, NAM or NAMLSS. Hence, for EBM we use 10 as the maximum number of leaves, 100 early stopping rounds and again the same learning rate of 0.005. For XGboost we use 500 estimators with a maximum depth of 15. η is set to 0.05. For the GAMLSS and gamboost models we use a logistic distribution to model the response distribution, where μ estimator uses identity and the σ estimator uses a log-link function.

Table B.4: Hyperparameters for the neural models for the FICO, Telco and Shrutime datasets

Hyperparameter	NAMLSS[1]	NAMLSS[2]	DNN	MLP	NAM
Learning rate	1e-04	1e-04	1e-04	1e-04	1e-04
Dropout	0.5	0.5	0.5	0.5	0.5
Hidden layers	$[250, 50, 25]$	$[250, 50, 25]$	$[250, 50, 25]$	$[250, 50, 25]$	$[250, 50, 25]$
LR decay, patience	0.95 - 10	0.95 - 10	0.95 - 10	0.95 - 10	0.95 - 10
Activation	ReLU	ReLU	ReLU	ReLU	ReLU
Output activation	Sigmoid, Softplus	Sigmoid, Softplus	Sigmoid, Softplus	Sigmoid	Sigmoid

[1] With 2 $\times$23 subnetworks. See Table 6.14 for an exemplary network structure.

[2] With 23 subnetworks and each subnetwork returning a parameter for the location and shape respectively. See Table B.1 for an exemplary network structure.

Munich and Melbourne We fit the AirBnB datasets, with an Inverse Gamma distribution where applicable. However, we train the models that only predict the mean with the squared error loss function. While one might suspect worse performances due to that, we find that using the squared error actually leads to much smaller gamma deviances compared to the models leveraging the Inverse Gamma distribution. Additionally, we use slightly smaller model structures than for the California housing dataset.

For all neural models, we use hidden layers of sizes $[512, 256, 50]$. The first hidden layer is followed by a 0.5 dropout layer. Throughout the hidden layers, we use ReLU activation functions. However, we deviate from that for the output layer activation functions. For the MLP we use a Softplus activation function for the output layer, ensuring that strictly positive values are predicted. For NAMLSS as well as the DNN we have to adjust the activation functions, as both models minimize the log-likelihood via the parameters α and β.

However, the mean prediction resulting from these parameters is defined via

$$\mu = \frac{\beta}{\alpha - 1}$$

and is hence only defined for $\alpha > 1$. The activation functions thus need to ensure a α prediction that is larger than 1 and a β prediction that is larger than 0. Hence we again use a Softplus activation for the β output layer.

For the α prediction, we use the following activation function element-wise

$$h(x) = \begin{cases} \log(1 + \exp(x)), & \text{if } \log(1 + \exp(x)) > 1 \\ \frac{1}{\log(1+\exp(x))}, & \text{else.} \end{cases}$$

To compute the log-likelihood for the models resulting in a mean prediction we compute the parameters α and β as follows

$$\alpha = \frac{\mu^2}{\sigma^2 + 2},$$

$$\beta = \mu \frac{\mu^2}{\sigma^2 + 1},$$

with σ^2 denoting the variance of the mean predictions.

For XGBoost and EBM we use a simple transformation of the target variable in order to ensure that $\mu > 0$. Hence we fit the model on $\log(y)$ and re-transform the predictions accordingly with $\exp(\hat{y})$. Otherwise, we use the same hyperparameters as for the California housing dataset. Interestingly, the NAM did not converge using the Softplus activation function as the MLP did. Using the Softplus activation resulted in tremendously large mean gamma deviances and log-likelihoods, as the model kept predicting values that were nearly zero. Hence, we were only able to achieve good results for the NAM using the activation function given by formula (B.1). The presented GAMLSS and gamboostLSS models assume an inverse Gamma distribution with both μ and σ utilizing the log-link function. It should be noted that the RS algorithm does not converge with GAMLSS, which is why CG is used.

Table B.5: Hyperparameters for the neural models for the Munich and Melbourne datasets

Hyperparameter	NAMLSS[1]	NAMLSS[2]	DNN	MLP	NAM
Learning rate	1e-04	1e-04	1e-04	1e-04	1e-04
Dropout	0.5	0.5	0.5	0.5	0.5
Hidden layers	$[512, 256, 50]$	$[512, 256, 50]$	$[512, 256, 50]$	$[512, 256, 50]$	$[512, 256, 50]$
LR decay, patience	0.95 - 10	0.95 - 10	0.95 - 10	0.95 - 10	0.95 - 10
Activation	ReLU	ReLU	ReLU	ReLU	ReLU
Output activation	Gamma*, Softplus	Gamma*, Softplus	Gamma*, Softplus	Linear	Linear

[1] With 2 $\times$23 subnetworks. See Table 6.14 for an exemplary network structure.

[2] With 23 subnetworks and each subnetwork returning a parameter for the location and shape respectively.

* See formula (B.1) for the detailed element-wise activation function.

See Table B.1 for an exemplary network structure.

Appendix C: Declaration of authorship

Table C.1: Declaration of authorship

Paper 1	*Model averaging for linear mixed models via augmented Lagrangian*
Role	First authorship
Idea	contributed substantially
Concept	contributed substantially
Literature work	leading
Data work	leading
Empirical work	leading
Writing	leading
Technical	leading
Paper 2	*On measuring complexity of deep learning models: A covariance penalty approach*
Role	First authorship
Idea	contributed substantially
Concept	leading
Literature work	leading
Data work	leading
Empirical work	leading
Writing	leading
Technical	leading
Paper 3	*Neural additive models for location, scale, and shape: A framework for interpretable neural regression beyond the mean*
Role	Shared first authorship with Anton Thielmann
Idea	contributed substantially
Concept	co-leading
Literature work	contributed substantially
Data work	co-leading
Empirical work	co-leading
Writing	contributed substantially
Technical	co-leading

Place and date

René-Marcel Kruse

Appendix D: Versicherung gem. §12 PStO

Versicherung bei Zulassung zur Promotionsprüfung

Ich versichere,

1. Dass ich die eingereichte Dissertation
   „Estimating and Evaluating Mixed and Semiparametric Models with Statistical and Deep Learning Methods“ selbstständig angefertigt habe und nicht die Hilfe Dritter in einer de Prüfungsrecht und wissenschaftlicher Redlichkeit widersprechenden Weise in Anspruch genommen habe.
2. Dass ich das Prüfungsrecht einschließlich der wissenschaftlichen Redlichkeit – hierzu gehört die strikte Beachtung des Zitiergebots, so dass die Übernahme fremden Gedankenguts in der Dissertation deutlich gekennzeichnet ist – beachtet habe.
3. Dass beim vorliegenden Promotionsverfahren kein Vermittler gegen Entgelt eingeschaltet worden ist sowie im Zusammenhang mit dem Promotionsverfahren und seiner Vorbereitung
   - Kein Entgelt gezahlt oder entgeltliche Leistungen erbracht worden sind
   - Keine Dienste unentgeltlich in Anspruch genommen wurden, die dem Sinn und Zweck eines Prüfungsverfahrens widersprechen
4. dass ich eine entsprechende Promotion nicht anderweitig beantragt und hierbei die eingereichte Dissertation oder Teile daraus vorgelegt habe.

Mir ist bekannt, dass Unwahrheiten hinsichtlich der vorstehenden Versicherung die Zulassung zur Promotionsprüfung ausschließen und im Falle eines späteren Bekanntwerdens die Promotionsprüfung für ungültig erklärt werden oder der Doktorgrad aberkannt werden kann.

_____

(Datum, Unterschrift)