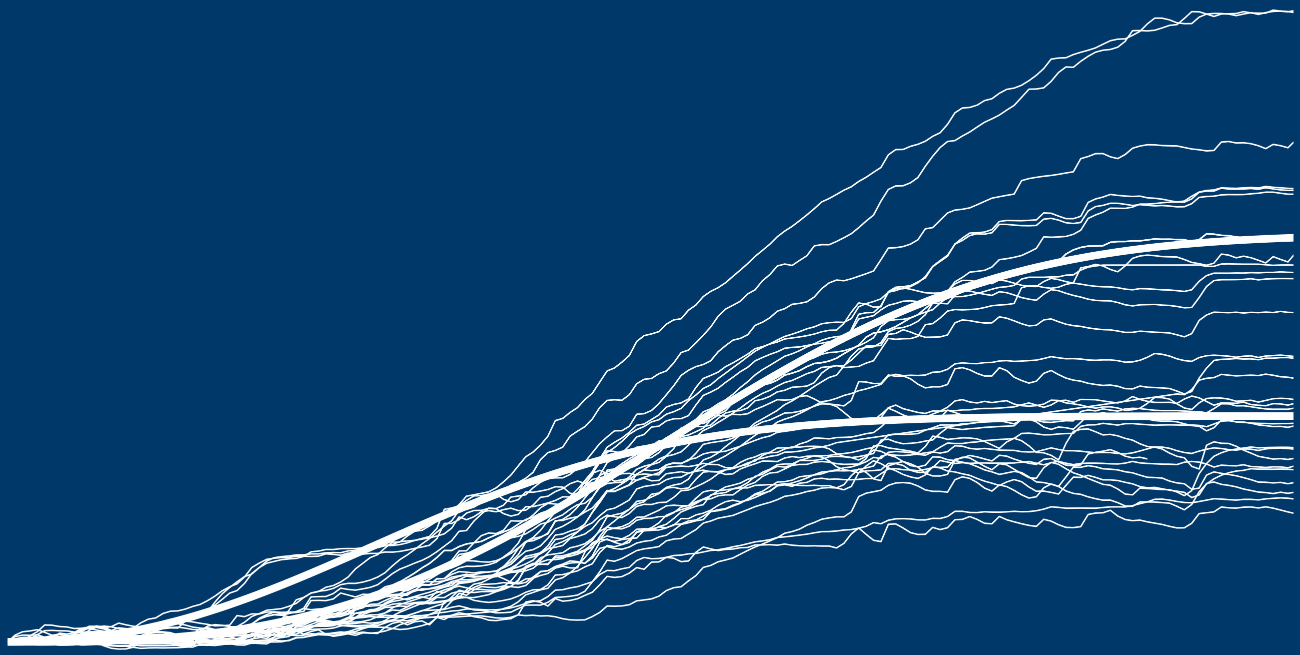
Hannes Riebl

Semi-Parametric Distributional Regression in Forestry and Ecology

Software, Models and Applications



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN IN PUBLICA COMMODA
SEIT 1737

Semi-Parametric Distributional Regression in Forestry and Ecology: Software, Models and Applications

Dissertation zur Erlangung
des Doktorgrades der Wirtschaftswissenschaftlichen Fakultät
der Georg-August-Universität Göttingen

vorgelegt von
Hannes Riebl
aus Laupheim

Göttingen, 2023

DOI: 10.53846/goediss-10051

Prüfungsausschuss:

1. Prof. Dr. Thomas Kneib,
   Professur für Statistik,
   Wirtschaftswissenschaftliche Fakultät,
   Georg-August-Universität Göttingen

2. Prof. Dr. Elisabeth Bergherr,
   Professur für Raumbezogene Datenanalyse und Statistische Lernverfahren,
   Wirtschaftswissenschaftliche Fakultät,
   Georg-August-Universität Göttingen

3. Prof. Dr. Niko Balkenhol,
   Professur für Wildtierwissenschaften,
   Fakultät für Forstwissenschaften und Waldökologie,
   Georg-August-Universität Göttingen

Tag der Disputation: 13. Juni 2023

Thanks to the tomato and the elephant…

Generated with the DALL·E image generator by OpenAI using the prompt “a van gogh painting of an elephant hugging a gigantic tomato with its trunk under a starry sky”.

Zusammenfassung

Die neueren Entwicklungen im Bereich der Machine-Learning-Software, etwa das automatische Differenzieren und die JIT-Kompilierung (JIT = Just in Time), haben die Forschung im maschinellen Lernen erheblich verändert. Sie haben die Modellentwicklung beschleunigt und zum Entstehen von KI-Werkzeugen wie dem Chatbot ChatGPT und dem Bildgenerator DALL·E beigetragen. Im Kontext der probabilistischen Programmierung werden ähnliche Methoden eingesetzt, um effiziente gradienten-basierte Inferenzalgorithmen zu implementieren, die auf eine Vielzahl von Bayesianischen Modellen anwendbar sind, z. B. Hamiltonian Monte Carlo (HMC) und der No-U-Turn Sampler (NUTS). Diese kumulative Dissertation umfasst drei Forschungsartikel, die Methoden des maschinellen Lernens und der probabilistischen Programmierung mit semi-parametrischen Regressionsmodellen aus der angewandten Statistik kombinieren. So wird die Entwicklung neuer Modelle mit semi-parametrischen Prädiktoren und den entsprechenden Inferenzalgorithmen möglich. Außerdem werden verschiedene Anwendungen in der Forstwissenschaft und der Ökologie vorgestellt.

Im ersten Artikel präsentieren wir das probabilistische Programmier-Framework Liesel, mit dem wir eine Software-Basis für effiziente und zuverlässige Forschung in der angewandten Statistik schaffen wollen, die geeignet ist für die Implementierung komplexer Modelle und Inferenzalgorithmen. Der Schwerpunkt der Software liegt auf semi-parametrischen Prädiktoren mit linearen, nicht-linearen, zufälligen und räumlichen Effekten von Kovariablen. Ein typischer Workflow mit Liesel wäre: (1) Konfiguration eines Modellgraphen, z. B. mithilfe des R-Interface von Liesel, (2) Anpassung des Modellgraphen zur Umsetzung neuer Forschungsideen, und (3) vollständige Bayes-Inferenz mit der mitgelieferten MCMC-Bibliothek (MCMC = Markov Chain Monte Carlo), entweder mit einem Standardalgorithmus oder einer benutzerdefinierten Variante. Sampler wie HMC und NUTS werden unterstützt und können mit herkömmlichen Methoden kombiniert werden, z. B. mit IWLS-Proposals (IWLS = Iterative Weighted Least Squares) und Gibbs-Updates. Liesel ist in Python geschrieben und nutzt die Machine-Learning-Bibliothek JAX als Backend.

Im zweiten und dritten Artikel werden Erweiterungen und Anwendungen der semi-parametrischen Verteilungsregression in der Forstwissenschaft und der Ökologie diskutiert. Die neuen Modelle ergeben sich aus der Einführung bestimmter Response-Strukturen in einen Regressionskontext, z. B. in Form von Gauß-Prozessen (GPs) mit parametrischen Mittelwert- und Kovarianzfunktionen. Das GP-Modell wenden wir auf Messungen von hochauflösenden Dendrometern an. Diese Geräte erfassen neben dem irreversiblen Wachstum von Baumstämmen auch die reversiblen Schwankungen aufgrund des Wassergehalts. Mit unserem Modell können die Daten in eine permanente und eine temporäre Komponente zerlegt werden, wobei sich Unterschiede zwischen Bäumen und Jahren durch Kovariablen erklären lassen. Im letzten Artikel schlagen wir das Multi-Species-Count-Modell (MSCM) vor, mit dem Zusammenhänge zwischen Umweltbedingungen und verschiedenen Indizes für Artenvielfalt geschätzt werden können. Wir nutzen das Modell mit semi-parametrischen Prädiktoren, um die Effekte von Rotbuche, Fichte und Douglasie auf die Artenvielfalt verschiedener Taxa zu bestimmen, basierend auf Daten, die im Graduiertenkolleg (GRK) 2300 erhoben wurden, und unter Berücksichtigung der räumlichen Korrelation.

Abstract

Recent advances in machine learning software, such as automatic differentiation and just-in-time (JIT) compilation, have significantly changed machine learning research. They have accelerated model development and contributed to the emergence of AI tools such as the chatbot ChatGPT and the image generator DALL·E. In the context of probabilistic programming, similar methods are used to implement efficient gradient-based inference algorithms applicable to a broad range of Bayesian models, e.g. Hamiltonian Monte Carlo (HMC) and the No-U-Turn Sampler (NUTS). This cumulative dissertation includes three research papers that combine methods from machine learning and probabilistic programming with semi-parametric regression models from applied statistics. This combination enables the development of novel models with semi-parametric regression predictors and the corresponding inference algorithms. Moreover, various applications in forestry and ecology are presented.

In the first paper, we present the probabilistic programming framework Liesel, which aims to provide a software basis for efficient and reliable research in applied statistics, suitable for the implementation of complex models and inference algorithms. The software focuses on semi-parametric regression predictors with linear, non-linear, random and spatial covariate effects. A typical workflow with Liesel would be: (1) configuration of a model graph as a baseline, e.g. using Liesel's R interface, (2) adaptation of the model graph to implement new research ideas, and (3) fully Bayesian inference using the included Markov chain Monte Carlo (MCMC) library, either with a standard algorithm or a user-defined variant. Samplers such as HMC and NUTS are supported and can be combined with conventional methods, e.g. iterative weighted least squares (IWLS) proposals and Gibbs updates. Liesel is written in Python and uses the machine learning library JAX as a backend.

The second and third paper discuss extensions and applications of semi-parametric distributional regression in forestry and ecology. The new models arise from the introduction of certain response structures into a regression context, e.g. in the form of Gaussian processes (GPs) with parametric mean and covariance functions. We apply the GP model to measurements from high-resolution circumference dendrometers. These instruments record both the irreversible growth of tree stems as well as the reversible shrinking and swelling due to the water content. With our model, the data can be decomposed into a permanent and a temporary component, and differences between trees and years can be explained by covariates. In the last paper, we propose the multi-species count model (MSCM) to estimate relationships between environmental conditions and different indices of species diversity. We use the model with semi-parametric regression predictors to assess the effects of European beech, Norway spruce and Douglas fir on the species diversity of various taxa, based on data collected in the Research Training Group (RTG) 2300 and taking into account spatial correlation.

Contents

1 Introduction

Regression is among the oldest and most popular methods in statistics, and aims to describe the relationship between a dependent variable (also called response or outcome variable) and one or more independent variables (also called predictor or explanatory variables, or covariates). Regression can be used for prediction and inference, with interpretability being one of its particular strengths, especially of its simplest form, which assumes a linear relationship between the response and explanatory variables, usually combined with a normally distributed error term (and hence a normally distributed response variable, see Fahrmeir et al., 2013, Section 3).

As these assumptions are too restrictive for many applications, semi-parametric regression predictors allowing for additive combinations of linear, non-linear, random and spatial covariate effects have been proposed (Wood, 2017, Section 4 and 5). Research in statistics has also focused on the development of non-Gaussian regression, e.g. logistic regression for binary outcomes, Poisson regression for count data, and Cox regression for survival data. Most of these models fall into the category of generalized linear models (GLMs, Nelder and Wedderburn, 1972), or even more broadly into generalized additive models for location, scale and shape (GAMLSS, Rigby and Stasinopoulos, 2005).

The introduction to this dissertation skims the evolution of various regression models from the simplest to more recent, flexible models (Section 1.1), before providing a brief overview of Bayesian inference and Markov chain Monte Carlo (MCMC) methods for the estimation of semi-parametric distributional regression models (Section 1.2). In Section 1.3, Bayesian inference is considered in the context of recent trends in Bayesian computing, especially probabilistic programming. Several concepts from probabilistic programming are implemented in the Liesel software package, which was developed as part of this dissertation. Finally, in Section 1.4, short application cases of semi-parametric distributional regression in the fields of forestry and ecology are presented, based not only on the manuscripts that are part of this dissertation, but also on two further articles with my co-authorship.

1.1 From linear to semi-parametric distributional regression

In the classical linear regression model, the response variable y_i is explained by the covariates x_i, for $i = 1, \dots n$, which are assumed to have a linear effect on the expected value of the response variable. Stochasticity is introduced in the model through an unobserved additive error term ε_i, i.e.

$$y_i = \beta_0 + x_{i1}\beta_1 + \cdots + x_{ik}\beta_k + \varepsilon_i = \eta_i + \varepsilon_i, \tag{1}$$

where β_0 is the intercept, $\beta_1, \dots, \beta_k$ are the slope coefficients, and $\eta_i = \beta_0 + x_{i1}\beta_1 + \cdots + x_{ik}\beta_k$ is the regression predictor. The model can be expressed more concisely in matrix notation as

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

where

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \quad \mathbf{X} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1k} \\ \vdots & \vdots & \cdots & \vdots \\ 1 & x_{n1} & \cdots & x_{nk} \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_k \end{pmatrix} \quad \text{and} \quad \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix}.$$

The standard assumptions on the error term $\boldsymbol{\varepsilon}$ are the so-called Gauss-Markov assumptions, i.e. $\mathrm{E}(\boldsymbol{\varepsilon}) = \mathbf{0}$, and $\mathrm{Cov}(\boldsymbol{\varepsilon}) = \mathrm{E}(\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}') = \sigma^2\mathbf{I}$. If the design matrix $\mathbf{X}$ is of full column rank, the regression coefficients $\boldsymbol{\beta}$ can be estimated with ordinary least squares (OLS), i.e. $\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{y}$. The OLS estimator minimizes the residual sum of squares $\hat{\boldsymbol{\varepsilon}}'\hat{\boldsymbol{\varepsilon}}$ over the residuals $\hat{\boldsymbol{\varepsilon}} = \boldsymbol{y} - \hat{\boldsymbol{y}} = \boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{\beta}}$. Under the Gauss-Markov assumptions, the OLS estimator is the best linear unbiased estimator (BLUE), i.e. it has minimum variance in the class of linear unbiased estimators (Seber and Lee, 2003, Section 3.2).

Note that the Gauss-Markov assumptions do not determine a specific probability distribution for the error term ε_i. For maximum likelihood inference (which is equivalent to OLS for the regression coefficients $\boldsymbol{\beta}$ in the linear regression model) and to construct confidence intervals and hypothesis tests, it is common to assume a Gaussian distribution for the errors, however. In combination with the Gauss-Markov assumptions, this implies that the errors are independent and identically distributed as $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ (Seber and Lee, 2003, Section 3.4).

It might seem counterintuitive at first, but it is possible to use linear regression to estimate non-linear relationships between a response variable y_i and some observed variables $\boldsymbol{v}_i$. Suppose, for example, we want to model the salary of the employees of a company as a function of their age and gender, i.e. the response variable is $y_i = \mathit{salary}_i$, and the observed variables are $\boldsymbol{v}_i = (\mathit{age}_i, \mathit{gender}_i)$. It might be plausible to assume that the salary increases with age, but more gradually for senior employees, and that the increase also depends on gender. To take these assumptions into account, we could define the covariate vector $\boldsymbol{x}_i$ as a function of the observed variables $\boldsymbol{v}_i$, i.e. $\boldsymbol{x}_i = \boldsymbol{x}_i(\boldsymbol{v}_i)$. An appropriate covariate vector could be $\boldsymbol{x}_i = (1, \mathit{age}_i, \mathit{age}_i^2, \mathit{gender}_i, \mathit{gender}_i \times \mathit{age}_i, \mathit{gender}_i \times \mathit{age}_i^2)$. Note that the regression predictor η_i is still linear in $\boldsymbol{\beta}$ and $\boldsymbol{x}_i$ in this case, but not in the observed variables $\boldsymbol{v}_i$.

Semi-parametric regression

Two drawbacks of using polynomials for modeling non-linear effects as in the previous example are their numerical instability and the difficulty to control the "wiggliness" of the estimated effects in a systematic way. For these reasons, semi-parametric regression predictors, also known as (structured) additive regression predictors, have been proposed as an alternative. They are a powerful and flexible modeling approach that can combine parametric and non-parametric covariate effects on a response variable y_i. The normal semi-parametric regression model is defined as

$$y_i = \boldsymbol{x}_i'\boldsymbol{\beta} + \sum_{l=1}^{L} f_l(\boldsymbol{z}_{il}) + \varepsilon_i,$$

where the parametric covariate effects $\boldsymbol{x}_i'\boldsymbol{\beta}$ include the intercept, and the non-parametric covariate effects $f_l(\boldsymbol{z}_{il})$, for $l = 1, \ldots, L$, are centered around zero to avoid identification issues (Fahrmeir et al., 2004; Wood, 2017, Chapter 4). The functions $f_l(\boldsymbol{z}_{il})$ are modeled as linear basis expansions of the covariates $\boldsymbol{z}_{il}$ (Hastie et al., 2009, Chapter 5). Depending on the choice of the basis and the penalty or prior on the regression coefficients $\boldsymbol{\gamma}_l$, the functions can represent non-linear, spatial and random effects, among others. Finally, the errors are independent and identically distributed as $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$.

The non-parametric covariate effect $f_l(\boldsymbol{z}_{il})$ is defined as the linear basis expansion $f_l(\boldsymbol{z}_{il}) = \mathbf{B}_l \boldsymbol{\gamma}_l$, where the entries of the design matrix $\mathbf{B}_l$ are given by $\mathbf{B}_{l,ij} = b_{lj}(\boldsymbol{z}_{il})$, and b_{lj} is the j-th basis function of the l-th non-parametric covariate effect, e.g. a B-spline basis function for some given knots. Therefore, the semi-parametric regression model can also be expressed in matrix notation as

$$\boldsymbol{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{B}_1\boldsymbol{\gamma}_1 + \cdots + \mathbf{B}_L\boldsymbol{\gamma}_L + \boldsymbol{\varepsilon},$$

and estimated with OLS. To avoid overfitting, however, certain smoothness properties are usually enforced on the non-parametric covariate effects through regularization, e.g. using P-splines that penalize (second) differences between coefficients of neighboring B-splines (Eilers and Marx, 1996; Lang and Brezger, 2004). This kind of regularization requires alternative estimation procedures, e.g. penalized maximum likelihood in a frequentist or MCMC methods in a Bayesian setting.

Bayesian regularization is usually implemented via informative priors, e.g. the multivariate normal prior

$$p(\boldsymbol{\gamma} \mid \tau^2) \propto \tau^{-\mathrm{rk}(\mathbf{K})} \exp(-0.5\tau^{-2}\boldsymbol{\gamma}'\mathbf{K}\boldsymbol{\gamma}),$$

where τ^2 is the variance (or smoothing) parameter, and $\mathbf{K}$ is a (potentially rank-deficient) penalty matrix. Note that we are omitting the index l here for the sake of simplicity. Apart from P-splines, other common non-parametric covariate effects include random effects, where the penalty matrix reduces to $\mathbf{K} = \mathbf{I}$, (intrinsic) Gaussian Markov random fields, where $\mathbf{K}$ is determined by the neighborhood structure of the spatial units (Rue and Held, 2005), and Gaussian processes, for which Vecchia approximations can be used to construct the penalty matrix (Katzfuss and Guinness, 2021).

Distributional regression

The other important direction in which linear and semi-parametric regression models can be extended is the assumption on their response distribution. For the previous models, the Gaussian distribution of the error term $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ implies the Gaussian distribution of the response variable $y_i \sim \mathcal{N}(\eta_i, \sigma^2)$. The first steps towards more flexible response distributions are generalized linear models (GLMs, Nelder and Wedderburn, 1972) and generalized additive models (GAMs, Hastie and Tibshirani, 1986; Wood, 2017), which can be used to model binary outcomes, count data and continuous non-negative responses. In GLMs and GAMs, the response distribution needs to belong to the exponential family, which is the case for the binomial, Poisson and gamma distribution, among others. Both model classes comprise one regression predictor η_i that is mapped to the conditional expectation of the response variable, i.e. $\mathrm{E}(y_i) = \mu_i = h(\eta_i)$, where h is an appropriate one-to-one response function from the real line to the domain of the mean of the response distribution. For the Gaussian distribution, the response function h is usually the identity function, for the binomial distribution, it is the logistic function, and for the Poisson distribution, the exponential function.

In comparison to GLMs and GAMs, generalized additive models for location, scale and shape (GAMLSS, Rigby and Stasinopoulos, 2005) relax the assumptions on the response distribution and the number of regression predictors even further. In GAMLSS, the response distribution can belong to any parametric family, and the number of regression predictors corresponds to the number of parameters of the response distribution. The model class, which is sometimes also referred to as distributional regression or regression beyond the mean, is defined as

$$y_i \sim \mathcal{D}(\theta_{i1} = h_1(\eta_{i1}), \theta_{i2} = h_2(\eta_{i2}), \dots, \theta_{ip} = h_p(\eta_{ip})), \tag{2}$$

where $\mathcal{D}$ is the response distribution, $\theta_{i1}, \dots, \theta_{ip}$ are the (response) parameters, $\eta_{i1}, \dots, \eta_{ip}$ are the corresponding (semi-parametric) regression predictors, and $h_1, \dots, h_p$ are the response functions mapping the regression predictors to the domain of the response parameters. In models for *location*, *scale* and shape, the first response parameter usually corresponds to the mean of the response distribution, and the second response parameter to the variance or standard deviation. This has the benefit of rendering the response parameters and covariate effects interpretable, but not all response distributions can be parameterized in this way.

The range of response distributions that fit into the GAMLSS framework is broad and includes discrete, continuous and mixed distributions (Rigby et al., 2019). Klein et al. (2015b) discuss various count data distributions in a regression context. For fractional responses, e.g. single or multiple percentages, the beta or Dirichlet distributions may be appropriate choices (Klein et al., 2015a). GAMLSS can also be used to study multivariate responses, e.g. using conventional multivariate distributions (Michaelis et al., 2018) or copulas to describe complex dependence structures with arbitrary marginal distributions (Klein and Kneib, 2016).

1.2 MCMC inference in semi-parametric distributional regression

Simple regression models can be estimated using least-squares methods. The estimation of semi-parametric and distributional regression models is more involved: For frequentist inference, approaches like (penalized) maximum likelihood estimation and gradient boosting have been proposed. In contrast, this dissertation deals exclusively with Bayesian inference using Markov chain Monte Carlo (MCMC) methods, which have proven to be a flexible and efficient alternative to frequentist approaches (Klein, 2014). In this section, we first provide a brief overview of the basic concepts of Bayesian inference, and then discuss its application in semi-parametric distributional regression.

Bayesian inference allows us to update our beliefs about the parameters $\boldsymbol{\theta}$ of a model based on new evidence or data $\boldsymbol{y}$. For this purpose, a prior distribution $p(\boldsymbol{\theta})$ is assumed for the parameters $\boldsymbol{\theta}$, representing our beliefs about the parameters *before* observing any data. After observing some data, we update our beliefs to obtain the posterior distribution $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ of the parameters $\boldsymbol{\theta}$ *given* the data $\boldsymbol{y}$. The update is performed using Bayes' rule, i.e.

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}) = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{y})},$$

where $p(\boldsymbol{y} \mid \boldsymbol{\theta})$ is the likelihood, and $p(\boldsymbol{y})$ is the marginal likelihood (or evidence) of the model. The caveat is that the evidence $p(\boldsymbol{y}) = \sum_{\boldsymbol{\theta}} p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})$ (in the case of discrete parameters) or $p(\boldsymbol{y}) = \int p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta}$ (in the case of continuous parameters) is usually intractable. For this reason, we need to find ways to assess the posterior distribution based on the proportionality $p(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})$. See Gelman et al. (2015), Section 1.3, or Fahrmeir et al. (2013), Appendix B.5, among many others, for a gentle introduction to Bayesian inference.

The posterior distribution $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ obtained from Bayesian inference can also be used for prediction. Assume we are about to observe a new data point $\tilde{\boldsymbol{y}}$. Rather than being limited to point prediction, the Bayesian approach allows us to express the uncertainty about $\tilde{\boldsymbol{y}}$ through the so-called *posterior predictive distribution*

$$p(\tilde{\boldsymbol{y}} \mid \boldsymbol{y}) = \int p(\tilde{\boldsymbol{y}} \mid \boldsymbol{\theta})p(\boldsymbol{\theta} \mid \boldsymbol{y})\, d\boldsymbol{\theta}.$$

If we are interested in a point prediction, we can compute the expected value (or some other measure of central tendency) of $\tilde{\boldsymbol{y}}$ from the posterior predictive distribution.

Since the 1990s, it is possible to perform Bayesian inference on a broad range of models using MCMC methods like Gibbs or Metropolis-Hastings (Robert and Casella, 2011). A modern and often highly efficient variant of Metropolis-Hastings is Hamiltonian Monte Carlo (HMC, Neal, 2011; Betancourt, 2018), which uses the gradient of the model log-probability function with respect to the parameters to sample from the posterior distribution. Other state-of-the-art approaches to Bayesian inference include sequential Monte Carlo (Chopin and Papaspiliopoulos, 2020) and variational inference (Blei et al., 2017). Variational inference is related to the expectation-maximization (EM) algorithm and can provide fast but sometimes inaccurate approximations to the posterior distribution. For computationally intensive models such as Bayesian deep neural networks, variational inference has become the standard in the field (Gelman et al., 2020, Section 3).

This dissertation relies on MCMC for Bayesian inference. MCMC allows us to sample from the posterior distribution $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ by constructing a Markov chain with the posterior distribution as its stationary distribution. Given an initial probability distribution $p^{(0)}(\boldsymbol{\theta} \mid \boldsymbol{y})$, the probability distribution at the $(l+1)$-th iteration of the chain is

$$p^{(l+1)}(\tilde{\boldsymbol{\theta}} \mid \boldsymbol{y}) = \int t(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta} \mid \boldsymbol{y})p^{(l)}(\boldsymbol{\theta} \mid \boldsymbol{y})\, d\boldsymbol{\theta},$$

where the art of developing an MCMC method is in constructing an appropriate transition probability $t(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta} \mid \boldsymbol{y})$. As $l \to \infty$, the probability distribution $p^{(l)}(\boldsymbol{\theta} \mid \boldsymbol{y})$ will eventually converge to the posterior distribution $p(\boldsymbol{\theta} \mid \boldsymbol{y})$, if (1) the posterior distribution is an *invariant distribution* of the chain, and (2) the chain is *ergodic*. The invariance of the posterior distribution under the chain is often shown through the *detailed balance* property. For more details, see MacKay (2003, Section 29.6).

One key advantage of MCMC is the characterization of the posterior distribution through a sample (van de Meent et al., 2021, Section 1.1.3). Thanks to the strong law of large numbers, this allows us to approximate the posterior mean of an arbitrary generated quantity $f(\boldsymbol{\theta})$ as the arithmetic mean of the function evaluations $f(\boldsymbol{\theta}^{(l)})$ at the states of the chain $\boldsymbol{\theta}^{(l)}$, for $l = 1, \dots, L$, i.e.

$$\lim_{L \to \infty} \frac{1}{L} \sum_{l=1}^{L} f(\boldsymbol{\theta}^{(l)}) \to \int f(\boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \boldsymbol{y}) \, d\boldsymbol{\theta},$$

where $\boldsymbol{\theta}^{(l)}$ is a draw from the posterior distribution $p(\boldsymbol{\theta} \mid \boldsymbol{y})$. In the simplest case, we can define $f(\boldsymbol{\theta}) = \boldsymbol{\theta}$ and compute the posterior mean of the parameters themselves. Examples of more complex generated quantities are given in Appendix C, where various species diversity indices are computed from the model parameters.

For semi-parametric distributional regression, Klein et al. (2015b) propose an MCMC algorithm based on a Metropolis-within-Gibbs scheme. They construct transition probabilities for the regression coefficients $\boldsymbol{\beta}$ from iterative weighted least squares (IWLS, Gamerman, 1997) proposals. These proposals make use of the expected or observed Fisher information to approximate the curvature of the posterior distribution. For the smoothing parameters τ^2 of the non-parametric covariate effects, Klein et al. use conjugate inverse gamma priors and the corresponding Gibbs updates. Assuming an inverse gamma prior with the hyperparameters a and b, the full conditional distribution has the parameters a^* and b^*, where $a^* = 0.5 \times \mathrm{rk}(\mathbf{K}) + a$, $b^* = 0.5 \times \boldsymbol{\beta}'\mathbf{K}\boldsymbol{\beta} + b$, $\mathbf{K}$ is the penalty matrix, and $\boldsymbol{\beta}$ are the regression coefficients of the non-parametric covariate effect.

Gamerman's IWLS proposals tend to become expensive and unstable if too many parameters are sampled in one block, because they require (an approximation of) the second derivative of the log-posterior. Therefore, this dissertation explores the use of HMC and the No-U-Turn Sampler (NUTS, Hoffman and Gelman, 2014) as an alternative for Bayesian inference in semi-parametric distributional regression. HMC is designed to simulate the evolution of a Hamiltonian system, defined by a potential and a kinetic energy function. It makes use of numerical integration to generate posterior samples, and requires only the first derivative of the log-posterior. NUTS was developed as a variant of HMC that frees the user from the duty of hyperparameter tuning (to a great extent). It uses a recursive algorithm to build a binary tree of possible states, responding to the curvature of the posterior distribution, and is usually easier to use and more efficient than HMC and other MCMC methods. Based on the experiments conducted for this dissertation, we can confirm that NUTS works efficiently for most models in the context of semi-parametric distributional regression.

1.3 Probabilistic programming and Bayesian statistics

In recent years, the fields of applied and Bayesian statistics have undergone a fundamental transformation, due to the emergence of new machine learning techniques, such as automatic differentiation and just-in-time (JIT) compilation. Probabilistic programming is a novel programming paradigm that applies machine learning techniques to formulate and estimate models based on probability theory, with the aim of automating Bayesian inference (van de Meent et al., 2021). Traditionally, the development of specific inference schemes for new statistical models has been an important component of research in Bayesian statistics. This approach is challenged by probabilistic programming, which attempts to represent an alternative to the "manual" development of model-specific inference schemes.

Probabilistic programming languages (PPLs) are the tools that implement the concepts of probabilistic programming, allowing users to represent and reason about uncertainty in real-world models. The models are defined by assigning probability distributions to a number of random variables and then, given some observed data, estimated by approximating the posterior distribution of the variables. For this purpose, PPLs provide programming constructs such as random variables and probability distributions for defining (log-)probability functions of complex models. Most PPLs also provide at least one inference algorithm, typically a variant of MCMC or variational inference depending on the specific PPL, for assessing the posterior distribution.

To be more concrete, consider the following simple example of an i.i.d. Gaussian data vector $\boldsymbol{y}$ with a conjugate normal-inverse-gamma prior for the model parameters μ and σ^2. In Stan (Stan Development Team, 2023), arguably the most popular probabilistic programming language at the time of writing, the model can be expressed as follows:

```
data {
  int<lower=0> N;
  vector[N] y;
}

parameters {
  real mu;
  real<lower=0> sig2;
}

model {
  sig2 ~ inv_gamma(0.1, 0.1);
  mu ~ normal(0.0, sqrt(sig2));
  y ~ normal(mu, sqrt(sig2));
}
```

Assume we observe the data vector $\boldsymbol{y} = (-0.084, 0.922, -0.369, -0.334, -2.333)$. As the prior is conjugate, we can compute the posterior means $\hat{\mu} = -0.366$ and $\hat{\sigma}^2 = 1.857$ analytically. Stan, however, takes a different approach to estimate the model using the gradient-based NUTS algorithm for MCMC sampling. It accounts for the constraint on the parameter space, transforms the variance parameter σ^2 to the real line, computes the derivatives of the model log-probability with respect to the transformed parameters using automatic differentiation, compiles the whole program and runs it on the observed data. The output from Stan is the following, confirming the analytic results:

```
Running MCMC with 4 sequential chains...

Chain 1 finished in 0.0 seconds.
Chain 2 finished in 0.0 seconds.
Chain 3 finished in 0.0 seconds.
Chain 4 finished in 0.0 seconds.

All 4 chains finished successfully.
Mean chain execution time: 0.0 seconds.
Total execution time: 0.7 seconds.

    var  mean median   sd  mad    q5   q95 rhat ess_bulk ess_tail
logprob -4.14  -3.78 1.13 0.84 -6.51 -3.03 1.00     1361     1631
     mu -0.36  -0.36 0.53 0.46 -1.24  0.49 1.00     1537     1373
   sig2  1.83   1.32 1.71 0.80  0.53  4.89 1.00     1810     1569
```

One advantage of probabilistic programming is that different versions of a model can be estimated with little effort. If, for example, we wanted to assume independent priors for the model parameters μ and σ^2, we could just replace the statement `mu ~ normal(0.0, sqrt(sig2));` on line 13 of the Stan code with `mu ~ normal(0.0, 10.0);`. The independent priors are non-conjugate, but Stan would work nonetheless, updating the derivatives automatically and running the same inference algorithm without requiring further user intervention.

It may not be too surprising that Stan is capable of estimating this simple model. If, however, a hypothetical PPL existed that worked efficiently for models of arbitrary complexity, its impact on applied Bayesian statistics would be tremendous, as it would render the development of model-specific inference schemes completely unnecessary. This would free resources for research on model development, diagnosis, interpretation and comparison, hence accelerating research in Bayesian statistics and its application in different fields. In a comprehensive overview paper, Gelman et al. (2020) introduce a framework for a so-called "Bayesian workflow" for real-world data analysis. The development of model-specific Bayesian inference schemes is not discussed as part of the workflow, highlighting the relevance of probabilistic programming already at the present day.

In more abstract terms, probabilistic programming languages allow us to shift our attention from the computation of the posterior $p(\boldsymbol{\theta} \mid \boldsymbol{y}, \boldsymbol{x})$ to the question: How can we formulate useful generative models, i.e. probability distributions $p(\boldsymbol{y}, \boldsymbol{\theta} \mid \boldsymbol{x})$ or even $p(\boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{x})$ over data $\boldsymbol{y}$ and $\boldsymbol{x}$ and parameters $\boldsymbol{\theta}$? How can we interpret and apply these models to real-world problems in meaningful ways? Gelman et al. (2020, Section 2.5) describe partially generative models, i.e. probability distributions $p(\boldsymbol{y}, \boldsymbol{\theta} \mid \boldsymbol{x})$, as the current standard in Bayesian statistics. These models are generative on the responses $\boldsymbol{y}$ and the parameters $\boldsymbol{\theta}$, but include unmodeled data $\boldsymbol{x}$, e.g. covariates and hyperparameters. On the one hand, they

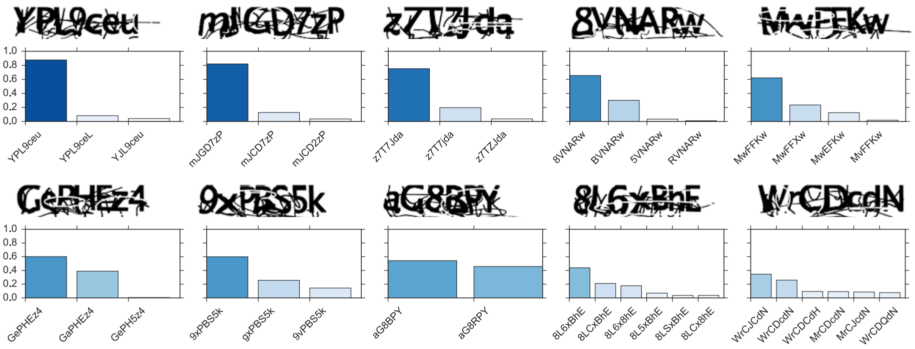


Figure 1: Posterior distributions over character sequences encoded in Facebook captchas from 2017, reproduced from van de Meent et al. (2021) and Le et al. (2017). The generative probabilistic model in this case is a probability distribution over the character sequences and the captchas that are generated from them. The posterior distributions are obtained by conditioning the model on the observed captchas.

are a step forward from frequentist models $p(\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{x})$, which do not define a probability distribution for the parameters $\boldsymbol{\theta}$, but on the other hand, they can be extended to fully generative models $p(\boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{x})$, where a probability distribution is assumed for all data and parameters that are part of the model.

The range of problems that can be solved with generative probabilistic models becomes clearer when considering an example (taken from van de Meent et al. (2021), Section 1.1) outside the domain of classical statistics. Assume we want to decode the captchas shown in Figure 1. To approach this problem with probabilistic programming, we need to understand “vision as inverse graphics” (Kulkarni et al., 2015) and describe the “graphics-generating process” in a probabilistic program, i.e. we need to write a program that produces captchas in a similar style as the original ones. Figure 1 shows how conditioning such a model on an observed captcha yields a posterior distribution over the character sequences that could have been used to produce the captcha. In contrast, the standard machine learning approach would require collecting a large number of captchas, hand-labeling them, and finally designing and training a neural network to map the captchas back to the original character sequences (Bursztein et al., 2014).

1.3.1 The Liesel probabilistic programming framework

One important component of this dissertation was the development of the software package Liesel (see Appendix A). Liesel takes inspiration from probabilistic programming, and aims to bring the convenience of probabilistic programming and machine learning (graph-based model representations, automatic differentiation, state-of-the-art gradient-based MCMC samplers) to research in applied statistics, especially on semi-parametric distributional regression and the corresponding inference algorithms. On top of that, the graph-based model representations used in Liesel are general enough to accommodate almost any Bayesian model beyond semi-parametric distributional regression. Liesel is, however, *not* a probabilistic programming *language*, as it does not define its own syntax (but rather works as a software *framework* for Python), and maybe more importantly, it does not aim to provide a one-size-fits-all inference algorithm, or otherwise try to automate Bayesian inference.

In contrast to probabilistic programming languages, Liesel can be described as a platform that assists the applied statistician with the development of semi-parametric distributional regression and other complex models, as well as the corresponding inference algorithms. Liesel comprises a graph-based model building library, an MCMC library with support for modular inference algorithms combining multiple kernels (both implemented in Python), and an R interface (RLiesel) for the configuration of semi-parametric regression predictors. Each component can be used independently of the others, e.g. the MCMC library also works with third-party model implementations. Using JAX (Bradbury et al., 2023) as a backend, we can take advantage of automatic differentiation, JIT compilation, and high-performance computing devices, e.g. tensor processing units (TPUs).

Although most probabilistic programming languages provide general-purpose inference algorithms such as NUTS, in practice, they often require non-trivial model reparameterizations to sample from the posterior efficiently (Stan Development Team, 2023, Section 25.7). The case study in Appendix A describes a situation where NUTS performs quite poorly despite attempts to reparameterize the model. At the same time, the Metropolis-within-Gibbs algorithm with IWLS and Gibbs updates proposed by Klein et al. (2015b) specifically for semi-parametric distribution regression suffers from similar issues: It is efficient for most members of the model class, but in some cases, splitting the parameter vector into separate blocks deteriorates the mixing of the MCMC chains. Appendix B on Gaussian process (GP) responses in semi-parametric distributional regression presents an example of this phenomenon. For these responses, a new blocking scheme had to be developed to improve the performance of the sampler.

Liesel combines the advantages of both approaches: From the graph-based model representation, a JIT-compilable log-probability function can be obtained for the use with Liesel's MCMC library Goose. In the simplest case, Goose can be used to run NUTS on all model parameters simultaneously, resembling the inference algorithm of Stan and similar PPLs. If required, however, the user can also configure a model-specific sampling scheme combining multiple parameter blocks and MCMC kernels. Goose assists the user by providing many popular MCMC kernels and warmup schemes out-of-the-box. This way, the user can decide on a case-by-case basis whether a reparameterization of the model or the development of a specific inference scheme is the most promising approach to achieve efficient Bayesian inference.

Similar to the flexibility in the development of inference algorithms, Liesel also provides a modeling library that supports the applied statistician with the configuration of semi-parametric distributional regression and other complex models. The library is designed around graph-based representations of these models and builds on well-tested software for semi-parametric regression, especially the `mgcv` package for R (Wood, 2023). One particular strength of Liesel's modeling library is its ability to customize models at runtime with the so-called graph builder.

The graph builder can be used to combine existing models, transform parameters, and add variables and nodes to the graph, e.g to extend the prior hierarchy. The implementation of the multi-species count model presented in Appendix C makes extensive use of this feature, combining the self-implemented graph-based response structure with semi-parametric regression predictors from Liesel's R interface.

The features of Liesel's modeling library correspond closely with Gelman et al.'s notion of modularization of Bayesian model building, which they describe as part of the Bayesian workflow. They provide the following examples of modules or placeholders in a prototypical model building process: "[...] we model data with a normal distribution and then replace this with a longer-tailed or mixture distribution; we model a latent regression function as linear and replace it with nonlinear splines or Gaussian processes; we can treat a set of observations as exact and then add a measurement-error model [...]" (Gelman et al., 2020, Section 2.2). Longer-tailed and mixture distributions fit naturally into the distributional regression framework, splines and Gaussian processes into semi-parametric regression predictors, both of which are well supported by Liesel. Finally, the graph builder would make it straightforward to integrate a measurement error model into an existing model, which previously assumed the covariates as fixed. Gelman et al. note that, traditionally in the statistical literature, whole models were given specific names, which goes against the concept of modularization and makes it harder to adapt them to individual use cases.

In summary, our aims with the Liesel probabilistic programming framework are the following:

- Bring the convenience of probabilistic programming techniques (automatic differentiation, HMC and NUTS, sophisticated MCMC warmup schemes) to the development of semi-parametric distributional regression models and the corresponding inference algorithms in the field of applied statistics.
- Allow the user to work with flexible graph-based representations of complex statistical models, which can be combined and modified at runtime. For semi-parametric distributional regression models, all standard components are provided out-of-the-box.
- Facilitate model development by providing tools for model visualization, prior and posterior predictive simulation, as well as MCMC summaries and diagnostics, partly based on the ArviZ software package (Kumar et al., 2019).

Liesel runs on Linux, macOS and with some limitations on Windows, and can be used on laptops, desktop computers and servers. The latest release, currently version 0.2.3, can be installed from the Python Package Index (PyPI). The source code is available under the MIT license on GitHub, where bugs can be reported and new features can be requested (https://github.com/liesel-devs/liesel). On the project homepage (https://liesel-project.org), we provide the API documentation and a collection of user tutorials.

1.4 Applications in Forestry and Ecology

The second goal of this dissertation is to demonstrate that semi-parametric and distributional regression have relevant applications in the fields of forestry and ecology, especially when combined with graph-based model representations and probabilistic programming techniques. Semi-parametric regression methods have been in use in forestry and ecology for several years, see e.g. Villarini et al. (2009) and Hawkins et al. (2013). In the remainder of this section, we discuss applications that have been developed during my time as a PhD student at the Chair of Statistics and in the Research Training Group (RTG) 2300. The applications include intra-annual tree growth (Appendix B) and species diversity in mixed forest stands (Appendix C), where the manuscripts are both part of this cumulative dissertation. In addition, two applications on growth allocation of tree seedlings (Bebre et al., 2021b) and small mammal habitat selection (Bebre et al., 2021a) are presented. The corresponding articles were written with colleagues from the RTG 2300 but are not part of this dissertation.

1.4.1 Decomposing dendrometer measurements into irreversible growth and reversible shrinking and swelling

In Appendix B, we propose a statistical method to separate the permanent and temporary components of measurements of tree stems obtained from high-resolution circumference dendrometers. The measurements capture the irreversible growth of the stems (due to the formation of new cells in the cambium), as well as the reversible shrinking and swelling (due to changes in the water content). By embedding Gaussian processes (GPs) with parametric mean and covariance functions as response structures in a distributional regression framework with structured additive predictors, we are able to decompose the measurements and explain differences between trees and years by covariates. While classical distributional regression focuses on univariate responses, a number of bivariate and trivariate distributions are available in the vector generalized additive model framework (Yee, 2015). Klein and Kneib (2016) discuss Bayesian inference in multivariate distributional regression with copula-based response distributions. Extending this line of thought, we demonstrate that distributional regression is also possible with GPs as an example of more general, continuous response structures.

Our model is related to regression for functional data as described by Shi and Choi (2011), Greven and Scheipl (2017) and Scheipl et al. (2015). However, our approach uses parametric mean functions, where the parameters of the mean functions represent the distributional parameters and are linked to covariates, while Greven and Scheipl (2017) focus on non-parametric mean functions using suitable basis expansions. If prior knowledge on the shape of the mean functions is available, e.g. that intra-annual tree growth curves follow a sigmoid shape, our model can be considered as more realistic and stable. In other cases, the assumption of parametric mean functions might be too restrictive. Finally, while most literature on functional data is concerned with time-indexed data, our model can accommodate GPs on different (potentially non-Euclidean) metric spaces.

The bamlss package for R (Umlauf et al., 2018) was used to implement the model, as Liesel was not available in 2018, when the project was started. In hindsight, the implementation would have been substantially easier with Liesel for at least two reasons: First, to evaluate the model log-probability, a large number of correlation matrices needs to be factorized. Unless the range parameters of the correlation matrices are updated, the Cholesky factors can be cached to improve the performance of the model. The caching was cumbersome to implement in bamlss, but would have been straightforward with Liesel, where the Cholesky factors could have been stored in an auxiliary variable in the model graph. Second, the standard deviation and range of GPs are often strongly correlated, resulting in poor mixing of the MCMC chains if they are sampled in separate blocks. To combine the parameter blocks, a new sampler had to be written from scratch in bamlss, while with Liesel's MCMC library Goose, the same could have been achieved with a small configuration change.

1.4.2 Estimating the effect of tree species and geographic location on various species diversity indices

Loss of biodiversity is one of the most pressing environmental issues of our time, as it affects the integrity of ecosystems and hence human well-being, as emphasized by the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES) in its global assessment report from 2019. At the University of Göttingen, the RTG 2300 studies the ecosystem of pure and mixed forest stands of European beech, Norway spruce and Douglas fir. The cultivation of non-native Douglas fir in managed forests in central Europe is often considered to mitigate the effects of climate change. To gain a better understanding of the consequences of introducing Douglas fir into the native ecosystem, Glatthorn et al. (2023) study the abundance and diversity of multiple taxa—e.g. fungi, plants, arthropods and small mammals—at eight field sites and 40 forest plots in Lower Saxony in northwest Germany.

As a statistically sounder, more integrated alternative to the two-step analysis of Glatthorn et al. (2023), we propose the multi-species count model (MSCM, Appendix C) to assess the relationship between the geographic location, the composition of tree species and the species diversity at the forest plots of the RTG. We apply the model to derive different species diversity indices for three taxa—collembola, small mammals and vegetation. As the model belongs to the class of Bayesian hierarchical models, we can incorporate structured additive predictors combining linear, non-linear, random and spatial covariate effects. We find that, for all three taxa, species richness and Shannon diversity are consistently higher in southern Lower Saxony than in the north. In contrast, the effect of tree species on species diversity remains ambiguous. For the vegetation and small mammals, we observe trends towards greater species diversity if the proportions of Norway spruce and Douglas fir are increased, while the effect is reversed for collembola.

The model and the inference algorithm were implemented with Liesel and Goose. We first developed a graph-based representation of the response structure in Liesel, which we then extended with structured additive predictors from R (R Core Team, 2023) and mgcv (Wood, 2023) using RLiesel. As some of the model parameters are discrete, they could not be sampled with NUTS, and even for the continuous parameters, one combined NUTS block proved to be inefficient. For this reason, we developed a custom Metropolis-within-Gibbs scheme combining a number of NUTS and Gibbs kernels. The sampling scheme iterates over the parameters in the model graph from the bottom to the top, so that the highest level of the prior hierarchy is sampled last. Using Goose, orchestrating the warmup and posterior phase of multiple kernels as in this application case is straightforward.

1.4.3 Linking seedling growth to light availability and competition type in a controlled pot experiment

In the study by Bebre et al. (2021b), we explore the impact of light availability and competition type on the growth allocation of tree seedlings. While light availability is critical for seedling establishment, growth and survival, their response to limited resources and competitive pressure is determined by species-specific traits, e.g. shade tolerance and rooting depth. To confirm this, we carried out a controlled pot experiment with three species (European beech, Norway spruce and Douglas fir). Three light availability levels (10%, 20% and 50%) were applied to pots containing four seedlings, which were grown either in monocultures or mixtures of two species. Our analysis reveals that with decreasing light availability, all species allocate more growth to height than diameter. Significant differences can be observed between conifers and broadleaf species, however. Taller seedlings tend to allocate less growth to height, as they already possess a competitive advantage.

For the analysis, we assume a cylindrical model for the seedling stems and compute the *relative growth allocation to height* as $R = \Delta V_H / \Delta V$, where ΔV_H is the volume growth due to height growth, and ΔV is the total volume growth over one growing season. The variable R is constrained to the unit interval, but as no values on the boundaries are observed, we use a log-normal location-scale regression model to relate the growth allocation to explanatory variables. The model is defined as

$$\log(R_{ijt}) = \eta_{ijt} + \varepsilon_{ijt}, \text{ where } \varepsilon_{ijt} \sim \mathcal{N}(0, (0.01 + \exp(\zeta_{ijt}))^2),$$

and the response variable R_{ijt} is the growth allocation of seedling j in pot i and year t. The predictor η_{ijt} describes the mean of the response, and the predictor ζ_{ijt} the standard deviation of the error term ε_{ijt}. Random intercepts for the pots are included in both structured additive predictors, in combination with fixed effects of species and competitor identity, height and diameter of the seedling, light availability, year and some interaction terms.

Figure 2 shows how the predicted probability distributions of the relative growth allocation to height shift towards the right as the light availability decreases. This indicates that seedlings tend to allocate more resources to height rather than diameter growth when exposed to less light. In addition to the mean shift, the growth allocation also shows more variability with decreasing light. Employing a distributional regression approach in this application is also important for another reason: The stronger the competition in one pot, the poorer the growth of the non-dominant seedlings, resulting in an increased standard deviation in that pot. If, for example, inter-specific competition exceeds intra-specific competition, pots containing species mixtures are going to have a greater standard deviation,

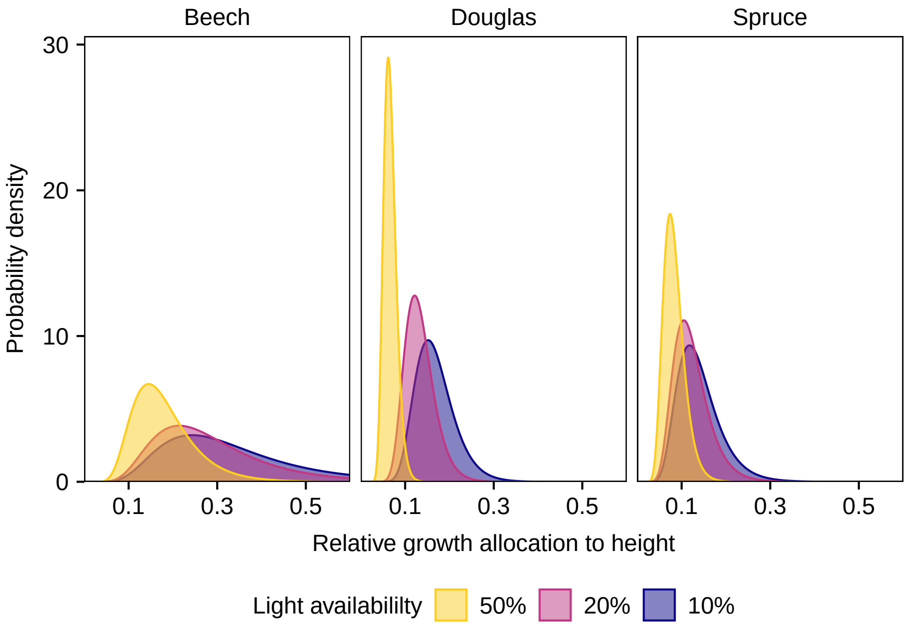


Figure 2: Probability distributions of the relative growth allocation to height depending on light availability and tree species, obtained from the log-normal location-scale regression model described in Section 1.4.3. All other explanatory variables are fixed to a representative value.

which needs to be reflected in the model specification. Moreover, in some pots, the competition is stronger due to idiosyncrasies in planting or soil, warranting the use of random intercepts in the predictor for the standard deviation.

1.4.4 Predicting small mammal habitat selection from understory complexity and mobile laser scans

In the study by Bebre et al. (2021a), we propose an understory complexity index derived from mobile laser scans (MLS) of forest plots to assess small-scale habitat preferences of mice and voles. Vegetation cover, downed branches and lying deadwood provide small mammals with food, shelter and nesting sites, and thus play an important role in habitat selection. The MLS-derived understory complexity index effectively captures relevant characteristics of the forest understory by measuring the variation in height of objects within a 5-meter radius around the trapping points. We demonstrate that understory complexity is a strong predictor of small mammal capture probability, suggesting that MLS could represent a promising technology for investigating habitat preferences of small mammals, with potential applicability to other species groups.

In order to relate the probabilities of capturing a mouse or vole in a specific trap to explanatory variables, we use a multinomial logistic regression model (Fahrmeir et al., 2013, Chapter 6.2) with structured additive predictors (Wood, 2017). The probabilities for mice and voles are compared to the reference category of observing an empty trap. The predictors for both mice and voles incorporate fixed effects of understory complexity and time of day, as well as random effects of trap check and plot, and a spatial effect to account for correlation within the plots. The application shows (1) that two predictors for mice and voles can be combined in one model, a key feature of distributional regression, (2) that structured additive predictors can integrate different types of covariate effects, in this case fixed, random and spatial effects, and (3) that the estimation of complex semi-parametric regression models with a high number of parameters is feasible, as the within-plot correlation is modeled with Gaussian Markov random fields for 14 forest plots $\times$ 64 trapping points $\approx$ 900 regression coefficients.

The predicted capture probabilities, as derived from the multinomial logistic regression model, are presented in Figure 3. The strong positive relationship between understory complexity and capture probabilities is evident. Moreover, the capture probabilities vary strongly between mice and voles, sites, and time of day.

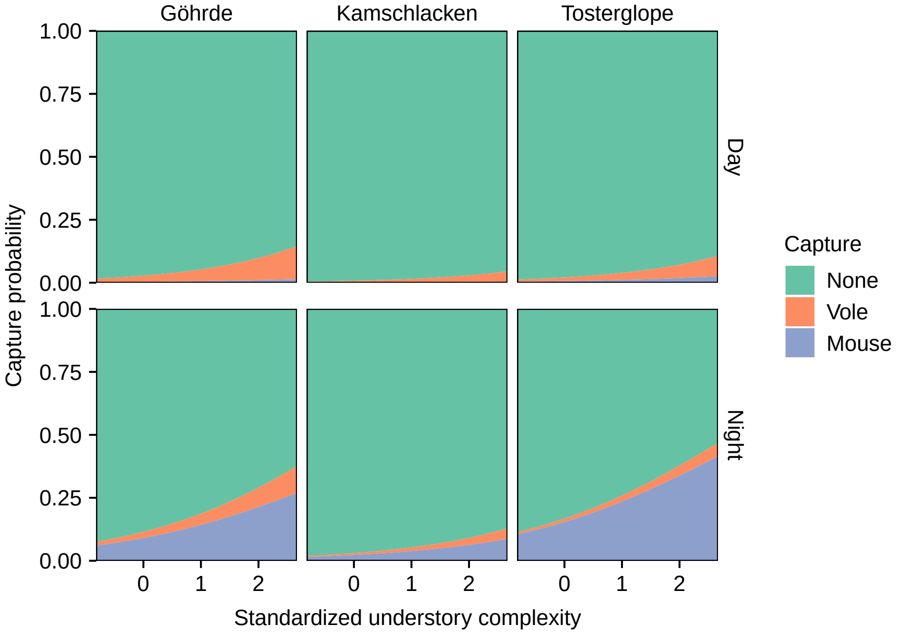


Figure 3: Stacked probabilities of capturing a mouse or vole depending on the understory complexity, site and time of day, obtained from the multinomial logistic regression model described in Section 1.4.4. For all other explanatory variables, the average over the full dataset is used.

2 Summaries of the manuscripts

The manuscripts that are part of this dissertation are in various ways related to the semi-parametric distributional regression framework described in the introduction, spanning topics from the development of modular, reusable and reliable research software in applied statistics and semi-parametric regression to extensions of the model class. The model extensions are implemented via novel response structures, most of which are motivated by applications in the fields of forestry and ecology.

This section contains short summaries of each of the manuscripts that may be found in the appendix of the dissertation. Some additional information on the context of the projects is also provided. Finally, in fulfillment of the requirements for a cumulative dissertation at the Faculty of Business and Economics of the University of Göttingen, my contributions and the contributions of all co-authors are declared in full detail.

Liesel: A Probabilistic Programming Framework for Developing Semi-Parametric Regression Models and Custom Bayesian Inference Algorithms

Joint work with Paul F.V. Wiemann and Thomas Kneib.
Published on *arXiv*: https://doi.org/10.48550/arXiv.2209.10975.
See Appendix A.

Liesel is a probabilistic programming framework focusing on but not limited to semi-parametric regression. It comprises a graph-based model building library, a Markov chain Monte Carlo (MCMC) library with support for modular inference algorithms combining multiple kernels (both implemented in Python), and an R interface (RLiesel) for the configuration of semi-parametric regression models. The development of the Liesel software was initiated by PW and me during our PhD studies in 2018. In November 2019, the grant proposal "LIESEL – A Software Framework for Bayesian Semi-Parametric Distributional Regression", co-authored by PW and me, was submitted to the German Research Foundation (DFG) by TK. The proposal was accepted as grant KN 922/11-1 in May 2021, primarily funding one postdoc for three years. Since April 2022, PW and me are developing the Liesel software collaboratively with two PhD students at the Chair of Statistics of TK. The latest version, v0.2.3 at the time of writing, was released in March 2023.

The article "Liesel: A Probabilistic Programming Framework for Developing Semi-Parametric Regression Models and Custom Bayesian Inference Algorithms" was published on arXiv in September 2022 and is currently being updated for a submission to the *Journal of Statistical Software*. It describes all components of the Liesel software framework in depth, emphasizing that they can be used independently of each other, e.g. the MCMC library also works with third-party model implementations. With Liesel, we aim to provide a platform for efficient and reliable statistical research on complex models and estimation algorithms. The typical workflow with Liesel is the following: (1) development of a model graph as a baseline, e.g. using RLiesel, (2) manipulation of the model graph to incorporate new research ideas using Liesel's modeling library, and (3) fully

Bayesian inference with the MCMC library, using either a default or user-defined algorithm. Several prominent MCMC kernels such as the No U-Turn Sampler (NUTS) are provided out-of-the-box and can be combined with self-written kernels to new powerful sampling schemes. Liesel also comes with various tools for summarizing, visualizing and diagnosing MCMC chains. Using JAX as a backend, we can take advantage of modern machine-learning technology such as automatic differentiation, just-in-time (JIT) compilation and high-performance computing devices, e.g. tensor processing units (TPUs).

My contributions to the software and the article are the following:

- I designed and developed the Liesel software framework together with PW. I am the main developer of Liesel's modeling library and the R interface, and contributed parts of the MCMC library.
- I co-supervised two PhD students and two student assistants, who were involved in the development of the Liesel software between 2019 and 2023. Together with PW, I established and coordinated the agile software development process of the Liesel team.
- I contributed to the grant proposal "LIESEL – A Software Framework for Bayesian Semi-Parametric Distributional Regression", which was accepted by the DFG, providing a significant amount of funding for the development of the Liesel software for three years.
- I conceptualized and wrote major parts of the manuscript "Liesel: A Probabilistic Programming Framework for Developing Semi-Parametric Regression Models and Custom Bayesian Inference Algorithms", in particular the section on Liesel's modeling library, the R interface and the case study.
- I designed, implemented and evaluated the results of the case study comparing the performance of different MCMC sampling schemes on a semi-parametric distributional regression model.
- I created the entity-relationship and sequence diagrams using the Mermaid software, as well as the model graphs using Graphviz and the figures for the case study using R and `ggplot2`.

PW and me contributed equally to the development of the Liesel software and the manuscript, both of which evolved through numerous productive discussions between the two of us. PW is the main developer of Liesel's MCMC library, contributed to the modeling library, and co-supervised the PhD students and student assistants on the Liesel team. He wrote parts of the grant proposal and the manuscript, in particular the introduction and the section on the MCMC library. TK wrote major parts of the grant proposal, found collaboration partners and application projects for the Liesel team, and assisted us with discussions throughout the project. All authors revised and edited the final manuscript.

Modelling Intra-Annual Tree Stem Growth with a Distributional Regression Approach for Gaussian Process Responses

Joint work with Nadja Klein and Thomas Kneib.
Published in the *Journal of the Royal Statistical Society: Series C (Applied Statistics)*: https://doi.org/10.1093/jrsssc/qlad015.
See Appendix B.

In this article, we develop a novel model class within the semi-parametric distributional regression framework, using Gaussian processes (GPs) with parametric mean and covariance functions as response structures. The approach is motivated by an application to measurements from high-resolution circumference dendrometers, capturing both the irreversible growth and the reversible shrinking and swelling due to the water content of a tree stem. Our method can be used to decompose the dendrometer measurements into a permanent and a temporary component, and to explain differences between the trees and years by covariates. The covariate effects can be modeled in a flexible fashion with structured additive predictors comprising linear, non-linear, random and spatial effects. Different choices for the mean and covariance functions of the GPs, connections with other statistical model classes, and Markov chain Monte Carlo (MCMC) inference are discussed in the article. Finally, the efficiency of the proposed sampling scheme is evaluated and two interesting model extensions are illustrated in an extensive simulation study.

My contributions to the article are the following:

- I developed and formalized the idea for the model class within the semi-parametric distributional regression framework.
- I developed the MCMC sampling scheme building on previous work of NK and TK, and derived the required mathematical quantities.
- I implemented the model class and the MCMC sampling scheme, including the documentation and the unit tests, in the new `bamlssGP` package for R, which depends on the `bamlss` package.
- I designed, implemented and evaluated the results of the simulation study with three scenarios, each focusing on different aspects of the MCMC sampling scheme and the model class.
- I developed the idea for the application on intra-annual tree stem growth, implemented it for a dataset of 85 trees and two growing seasons, and interpreted the estimation results in their ecological context.
- I illustrated the results of the simulation study and the application graphically using the `ggplot2` package for R and the software `gnuplot`.
- I published the computer code for the simulation study, the application and the graphics on GitHub to follow the best practices of open science and to make the study reproducible.
- I wrote most of the manuscript and all of the supplementary material.

NK contributed to the structure and framing of the article, and assisted me with the mathematical formulation of the model class and the inference algorithm. TK wrote the section on mixed models as a related model class and parts of the discussion section. He also assisted me with the design of the simulation scenarios and with discussions throughout the project. All authors revised and edited the final manuscript.

A Structured Additive Multi-Species Count Model for Assessing the Relation Between Site Conditions and Species Diversity

Joint work with Jonas Glatthorn and Thomas Kneib.
Unpublished manuscript.
See Appendix C.

In the final manuscript, we propose the multi-species count model (MSCM), a model to assess the relationship between the environmental conditions at a number of field sites and different species diversity indices derived from species counts at the sites. The model is motivated by a meta-analysis of data from the Research Training Group (RTG) 2300, aiming to partially replicate the work of Glatthorn et al. (2023). The RTG 2300 is a research project conducted in Lower Saxony, northwest Germany, studying the impact of admixed Norway spruce and Douglas fir on the ecosystem of European beech forests. As the MSCM belongs to the class of Bayesian hierarchical models, structured additive predictors can be incorporated to identify the effect of the coniferous tree species on biodiversity, while accounting for the spatial correlation of the field sites.

We also describe the MSCM from a theoretical perspective, draw connections with several related model classes such as zero-inflated Poisson regression and multi-species occupancy models (MSOMs), and discuss a few interesting model extensions and generalizations. The model and the proposed MCMC algorithm for fully Bayesian inference are implemented in Python using the probabilistic programming framework Liesel and its MCMC library Goose. Therefore, the project also serves as a proof a concept and one of the first complex use cases of Liesel and Goose. The performance of the Goose-based inference algorithm is evaluated in a simulation study.

My contributions to the article are the following:

- I developed and formalized the idea for the model class within the semi-parametric distributional regression framework.
- I implemented the model class and the MCMC sampling scheme in Python using the probabilistic programming framework Liesel.
- I designed, implemented and evaluated the results of the simulation study with three scenarios.
- I investigated the relationship of the model class with other statistical and ecological models, e.g. zero-inflated Poisson regression and multi-species occupancy models.
- I applied the model class to three different taxa, which were surveyed on the 40 experimental plots of the RTG 2300, and interpreted the estimation results in their ecological context.
- I illustrated the results of the simulation study and the application graphically using the `ggplot2` package for R.
- I wrote the complete manuscript.

JG contributed the pre-processed data from the RTG 2300 and discussed the ecological context and interpretation with me. TK contributed to the structure and framing of the article, assisted me with the design of the simulation scenarios and the application, and with discussions throughout the project. All authors revised and edited the final manuscript.

3 Conclusion and outlook

This cumulative dissertation presents three manuscripts on semi-parametric distributional regression, all in a Bayesian context, but approaching the topic from different angles, focusing on software, models and applications. In Appendix A, we present the probabilistic programming framework Liesel, which allows the applied statistician to express Bayesian models using a graph-based, "hackable" model representation, and to develop custom MCMC algorithms from building blocks such as HMC, NUTS and Gibbs kernels, as well as sophisticated warmup schemes for hyperparameter tuning. RLiesel, the R interface of the framework, can be used to configure (and later extend) semi-parametric distributional regression models. Using JAX as a backend, Liesel can take advantage of state-of-the-art machine learning technology, e.g. automatic differentiation, just-in-time compilation and tensor processing units.

Furthermore, we introduce two novel response structures within the distributional regression framework. In Appendix B, we use Gaussian processes (GPs) with parametric mean and covariance functions to model intra-annual tree stem growth, decomposing high-resolution dendrometer measurements into permanent growth and temporary shrinking and swelling. Various properties of both measurement components are explained by semi-parametric regression predictors based on covariates such as tree species and diameter at breast height (DBH). In Appendix C, we propose the multi-species count model (MSCM) for assessing the relationship between site conditions and species diversity. The model is applied to data from the Research Training Group (RTG) 2300, studying the effect of European beech, Norway spruce and Douglas fir on different species diversity indices and taxa, while accounting for spatial correlation with semi-parametric regression predictors.

One of the key achievements of this dissertation is re-imagining the *distributional* regression framework by extending the spectrum of possible response distributions and *response structures*. GAMLSS were originally designed for univariate responses, although they have been extended to bivariate and trivariate distributions. This dissertation considers further generalizations of response structures in GAMLSS, preferably expressed as subgraphs of semi-parametric distributional regression models. This perspective is in line with the concept of modularization of model building, as described by Gelman et al. (2020, Section 2.2) as part of the Bayesian workflow. In fact, probability distributions are just simple statistical models, and vice versa, many statistical models can be interpreted as probability distributions. The linear regression model (1), for example, defines a probability distribution $\mathcal{D}$ with the parameters $\boldsymbol{\theta} = (\beta_0, \beta_1, \dots, \beta_k, \sigma^2)$ for the data $\boldsymbol{y}$. In this sense, linear regression is not a closed model, but can itself be considered as a response distribution, i.e. a module, in the distributional regression framework (2). The concept of "regression on regression coefficients" has been explored before, e.g. in the multi-level structured additive regression model by Lang et al. (2014), but the argument here is more general: To extend the distributional regression framework, we can think beyond what we usually imagine a probability distribution to be. This allows us to incorporate response structures such as GPs or the MSCM into the distributional regression framework, opening up a variety of possibilities for interesting applications in different fields.

In the remainder of this section, we outline avenues for the continuation of the projects presented in this dissertation. We start with a discussion of the probabilistic programming framework Liesel, for which we plan to extend and improve the software itself, and to implement application cases based on the existing software. The application cases include the migration of the GP responses to Liesel, which would facilitate the development of further model extensions, e.g. to non-Gaussian stochastic processes. Finally, generalizations of the MSCM, which is already implemented in Liesel, are discussed, especially to allow for meta-studies across different taxa.

Further development and applications of Liesel and Goose

The development of the probabilistic programming framework Liesel will be continued in the coming years. Our software depends on a number of libraries that are under active development. API updates, new features and other changes in these libraries need to be reflected in Liesel. Currently, a team of four developers at the University of Göttingen and Texas A&M University is actively working on and with Liesel. The DFG grant KN 922/11-1 is funding parts of the development of Liesel until March 2025. With two PhD students working on Liesel-related topics, there also is a strong research interest in extensions and improvements of the software.

With regard to the software itself, we are planning to extend Liesel with more model components and new MCMC algorithms. As Liesel's technology stack makes it straightforward to implement gradient-based methods, we are considering to add stochastic gradient MCMC (SG-MCMC) methods in the near future. SG-MCMC is a novel class of Monte Carlo algorithms that scales well to models with a large number of observations. Compared to traditional MCMC methods, SG-MCMC reduces the computational cost of Bayesian inference by using subsamples of the original dataset. Another priority will be the development of an enhanced interface between Liesel and Goose, i.e. the modeling and MCMC library, which will reduce the effort required to configure a Goose inference scheme for a Liesel model.

Several applied projects have already been realized with Liesel to date, e.g. a multi-level model for variance partitioning on different spatial scales by Marques et al. (2023), who used an early version of Goose for the project. The model is designed for data from forest inventories that is collected on a fine spatial scale within forest plots, and on a coarse scale between plots, similar to the experimental design of the Research Training Group (RTG) 2300 described in Appendix C. For this kind of data, it is plausible to assume a spatial correlation both within and between plots, which the model by Marques et al. can account for. Another example of a use case of Liesel and Goose is the Python package `liesel-bctm` (https://github.com/liesel-devs/liesel-bctm) by Johannes Brachem, implementing Bayesian conditional transformation models (CTMs) and the corresponding MCMC samplers. The `liesel-bctm` package is based on previous work of Carlan and Kneib (2022) and Carlan et al. (2023).

More ideas for Liesel and Goose are described in the project proposal of the DFG grant KN 922/11-1, e.g. in the work packages on random scaling effects, distributional structural equation models and joint modeling. The proposal also outlines how Liesel's graph builder could be used to combine existing models to correct for dependent and non-Gaussian measurement errors of covariates, a requirement of increasing importance as many recent models combine data from different sources. Finally, Liesel could serve as a basis for the implementation of deep distributional regression models, embedding neural networks into graph-based representations of distributional regression models. For this purpose, a broad range of deep learning libraries for JAX could be wrapped and combined with the existing model components of Liesel. The deep distributional regression approach allows us to combine the predictive strength of neural networks with regression techniques beyond the mean, taking into account parameters such as the scale or shape of the response distribution.

Non-Gaussian stochastic processes in distributional regression

The GP responses were originally implemented in R based on the `bamlss` package. As discussed in Section 1.4.1, the implementation of the software would have been substantially easier with Liesel and Goose. Several missing features would also be more straightforward to add to the software using Liesel, e.g. support for arbitrary mean and covariance functions beyond the ones discussed in the manuscript, which would require some form of automatic differentiation. Thanks to JAX' just-in-time compilation, we would also expect a significant speed-up of Liesel over `bamlss`. The improved performance and the detailed configuration options of Goose would allow us to explore new, potentially more efficient sampling schemes with less effort than before.

From a statistical perspective, the motivation of the model was the analysis of intra-annual tree stem growth, but the flexibility and versatility of the approach is emphasized throughout the manuscript and the simulation study. To explore the full potential of the model class in various fields of application, other choices of index sets, as well as mean and covariance functions of the GPs need to be studied. Finally, further generalizations of the response structures are possible using non-Gaussian stochastic processes. These stochastic processes could either be defined by wrapping GPs with pointwise non-Gaussian distributions, or by considering different types of stochastic processes, e.g. point processes such as Poisson or Cox processes.

Optimizing the multi-species count model for meta-studies

Due to its low data requirements, the MSCM is a useful model for meta-studies across various taxa that are sampled according to different protocols. In these cases, repeated surveys at each site as required by multi-species occupancy models (MSOMs) are not always available. Based on the MSCM as described in the manuscript, this sort of meta-study involves the estimation of one model per taxon, before the results can be summarized over the taxa. To further strengthen the model's aptitude for meta-studies, we are planning to extend the model graph so that different taxa can be combined in one MSCM. This way, summary statistics over the taxa could be obtained from a single, integrated model, and assumptions about how the effects of the environmental factors are correlated across the taxa could be expressed via appropriate priors.

Another option for extending the MSCM would be adding a second structured additive predictor for the expected abundances, potentially including a functional relationship between the second and the first predictor for the occupancy probabilities. Assuming a functional relationship between the predictors seems plausible, as the same environmental factors that drive the occupancy probabilities are also likely to affect the expected abundances. Finally, the response structure of the MSCM combining a count distribution for the total number of observations at a site and a multinomial distribution over the species deserves further attention, and could be compared to independent zero-inflated count distributions for each site and species. As shown in the manuscript, if the total number of observations follows a Poisson distribution, the MSCM is equivalent to zero-inflated Poisson regression. For other count distributions, when an exact mathematical equivalence does not hold, a comparison based on simulations and real-world data would be worthwhile to assess which of the models has the better explanatory and predictive performance.

References

Ieva Bebre, Scott Appleby, Hannes Riebl, and Dominik Seidel. Linking small mammal capture probability with understory structural complexity using mobile laser scanning-derived metrics: A case study. Unpublished Manuscript, 2021a.

Ieva Bebre, Hannes Riebl, and Peter Annighöfer. Seedling Growth and Biomass Production under Different Light Availability Levels and Competition Types. *Forests*, 12(10):1376, October 2021b. ISSN 1999-4907. DOI: 10.3390/f12101376.

Michael Betancourt. A Conceptual Introduction to Hamiltonian Monte Carlo. arXiv, July 2018. DOI: 10.48550/arXiv.1701.02434.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, April 2017. ISSN 0162-1459. DOI: 10.1080/01621459.2017.1285773.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: Composable transformations of Python+NumPy programs, 2023. URL: https://github.com/google/jax.

Elie Bursztein, Jonathan Aigrain, Angelika Moscicki, and John C. Mitchell. The End Is Nigh: Generic Solving of Text-Based CAPTCHAs. In *8th USENIX Workshop on Offensive Technologies (WOOT '14)*, 2014. URL: https://www.usenix.org/conference/woot14/workshop-program/presentation/bursztein.

Manuel Carlan and Thomas Kneib. Bayesian discrete conditional transformation models. *Statistical Modelling*, September 2022. ISSN 1471-082X. DOI: 10.1177/1471082X221114177.

Manuel Carlan, Thomas Kneib, and Nadja Klein. Bayesian Conditional Transformation Models. *Journal of the American Statistical Association*, March 2023. ISSN 0162-1459. DOI: 10.1080/01621459.2023.2191820.

Nicolas Chopin and Omiros Papaspiliopoulos. *An Introduction to Sequential Monte Carlo*. Springer Series in Statistics. Springer, Cham, 2020. ISBN 978-3-030-47845-2. DOI: 10.1007/978-3-030-47845-2.

Paul H. C. Eilers and Brian D. Marx. Flexible smoothing with B-splines and penalties. *Statistical Science*, 11(2):89–121, May 1996. ISSN 2168-8745. DOI: 10.1214/ss/1038425655.

Ludwig Fahrmeir, Thomas Kneib, and Stefan Lang. Penalized Structured Additive Regression for Space-Time Data: A Bayesian Perspective. *Statistica Sinica*, 14(3):731–761, 2004. ISSN 1017-0405. URL: https://www.jstor.org/stable/24307414.

Ludwig Fahrmeir, Thomas Kneib, Stefan Lang, and Brian Marx. *Regression: Models, Methods and Applications*. Springer, Berlin, Heidelberg, 2013. ISBN 978-3-642-34333-9. DOI: 10.1007/978-3-642-34333-9.

Dani Gamerman. Sampling from the posterior distribution in generalized linear mixed models. *Statistics and Computing*, 7(1):57–68, March 1997. ISSN 1573-1375. DOI: 10.1023/A:1018509429360.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, New York, third edition, July 2015. ISBN 978-0-429-11307-9. DOI: 10.1201/b16018.

Andrew Gelman, Aki Vehtari, Daniel Simpson, Charles C. Margossian, Bob Carpenter, Yuling Yao, Lauren Kennedy, Jonah Gabry, Paul-Christian Bürkner, and Martin Modrák. Bayesian Workflow. arXiv, November 2020. DOI: 10.48550/arXiv.2011.01808.

Jonas Glatthorn, Scott Appleby, Niko Balkenhol, Peter Kriegel, Likulunga Emmanuel Likulunga, Jing-Zhong Lu, Dragan Matevski, Andrea Polle, Hannes Riebl, Carmen Alicia Rivera Pérez, Stefan Scheu, Alexander Seinsche, Peter Schall, Andreas Schuldt, Severin Wingender, and Christian Ammer. Species diversity of forest floor biota in non-native Douglas-fir stands is similar to that of native stands. *Ecosphere*, 14(7):e4609, 2023. ISSN 2150-8925. DOI: 10.1002/ecs2.4609.

Sonja Greven and Fabian Scheipl. A general framework for functional regression modelling. *Statistical Modelling*, 17(1-2):1–35, February 2017. ISSN 1471-082X. DOI: 10.1177/1471082X16681317.

Trevor Hastie and Robert Tibshirani. Generalized Additive Models. *Statistical Science*, 1(3):297–310, August 1986. ISSN 2168-8745. DOI: 10.1214/ss/1177013604.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, New York, 2009. ISBN 978-0-387-84858-7. DOI: 10.1007/978-0-387-84858-7.

Ed Hawkins, Thomas E. Fricker, Andrew J. Challinor, Christopher A. T. Ferro, Chun Kit Ho, and Tom M. Osborne. Increasing influence of heat stress on French maize yields from the 1960s to the 2030s. *Global Change Biology*, 19 (3):937–947, 2013. ISSN 1365-2486. DOI: 10.1111/gcb.12069.

Matthew D. Hoffman and Andrew Gelman. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(47):1593–1623, 2014. ISSN 1533-7928. URL: https://jmlr.org/papers/v15/hoffman14a.html.

Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services. Global assessment report on biodiversity and ecosystem services of the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services. Technical report, Zenodo, May 2019.

Matthias Katzfuss and Joseph Guinness. A General Framework for Vecchia Approximations of Gaussian Processes. *Statistical Science*, 36(1):124–141, February 2021. ISSN 2168-8745. DOI: 10.1214/19-STS755.

Nadja Klein. *Bayesian Structured Additive Distributional Regression*. PhD thesis, University of Göttingen, 2014.

Nadja Klein and Thomas Kneib. Simultaneous inference in structured additive conditional copula regression models: A unifying Bayesian approach. *Statistics and Computing*, 26(4):841–860, July 2016. ISSN 1573-1375. DOI: 10.1007/s11222-015-9573-6.

Nadja Klein, Thomas Kneib, Stephan Klasen, and Stefan Lang. Bayesian Structured Additive Distributional Regression for Multivariate Responses. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 64(4): 569–591, August 2015a. ISSN 0035-9254. DOI: 10.1111/rssc.12090.

Nadja Klein, Thomas Kneib, and Stefan Lang. Bayesian Generalized Additive Models for Location, Scale, and Shape for Zero-Inflated and Overdispersed Count Data. *Journal of the American Statistical Association*, 110(509):405–419, January 2015b. ISSN 0162-1459. DOI: 10.1080/01621459.2014.912955.

Tejas D. Kulkarni, William F. Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep Convolutional Inverse Graphics Network. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, 2015. URL: https://papers.nips.cc/paper_files/paper/2015/hash/ced556cd9f9c0c8315c fbe0744a3baf0-Abstract.html.

Ravin Kumar, Colin Carroll, Ari Hartikainen, and Osvaldo Martin. ArviZ a unified library for exploratory analysis of Bayesian models in Python. *Journal of Open Source Software*, 4(33):1143, 2019. DOI: 10.21105/joss.01143.

Stefan Lang and Andreas Brezger. Bayesian P-Splines. *Journal of Computational and Graphical Statistics*, 13(1):183–212, March 2004. ISSN 1061-8600. DOI: 10.1198/1061860043010.

Stefan Lang, Nikolaus Umlauf, Peter Wechselberger, Kenneth Harttgen, and Thomas Kneib. Multilevel structured additive regression. *Statistics and Computing*, 24(2):223–238, March 2014. ISSN 1573-1375. DOI: 10.1007/s11222-012-9366-0.

Tuan Anh Le, Atilim Giineş Baydin, Robert Zinkov, and Frank Wood. Using synthetic data to train neural networks is model-based reasoning. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 3514–3521, May 2017. DOI: 10.1109/IJCNN.2017.7966298.

David J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, version 7.2 (fourth printing) edition, 2003. ISBN 978-0-521-67051-7.

Isa Marques, Paul F. V. Wiemann, and Thomas Kneib. A Variance Partitioning Multi-level Model for Forest Inventory Data with a Fixed Plot Design. *Journal of Agricultural, Biological and Environmental Statistics*, May 2023. ISSN 1537-2693. DOI: 10.1007/s13253-023-00548-z.

Patrick Michaelis, Nadja Klein, and Thomas Kneib. Bayesian Multivariate Distributional Regression With Skewed Responses and Skewed Random Effects. *Journal of Computational and Graphical Statistics*, 27(3):602–611, July 2018. ISSN 1061-8600. DOI: 10.1080/10618600.2017.1395343.

Radford M. Neal. MCMC Using Hamiltonian Dynamics. In *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC, New York, 2011. ISBN 978-0-429-13850-8.

John A. Nelder and Robert W. M. Wedderburn. Generalized Linear Models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384, 1972. ISSN 0035-9238. DOI: 10.2307/2344614.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2023. URL: https://www.r-project.org.

Robert A. Rigby and Mikis D. Stasinopoulos. Generalized additive models for location, scale and shape. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 54(3):507–554, 2005. ISSN 1467-9876. DOI: 10.1111/j.1467-9876.2005.00510.x.

Robert A. Rigby, Mikis D. Stasinopoulos, Gillian Z. Heller, and Fernanda De Bastiani. *Distributions for Modeling Location, Scale, and Shape: Using GAMLSS in R*. Chapman and Hall/CRC, New York, October 2019. ISBN 978-0-429-29854-7. DOI: 10.1201/9780429298547.

Christian Robert and George Casella. A Short History of Markov Chain Monte Carlo: Subjective Recollections from Incomplete Data. *Statistical Science*, 26 (1):102–115, February 2011. ISSN 2168-8745. DOI: 10.1214/10-STS351.

Havard Rue and Leonhard Held. *Gaussian Markov Random Fields: Theory and Applications*. Chapman and Hall/CRC, New York, February 2005. ISBN 978-0-429-20882-9. DOI: 10.1201/9780203492024.

Fabian Scheipl, Ana-Maria Staicu, and Sonja Greven. Functional Additive Mixed Models. *Journal of Computational and Graphical Statistics*, 24(2):477–501, April 2015. ISSN 1061-8600. DOI: 10.1080/10618600.2014.901914.

George A. F. Seber and Alan J. Lee. *Linear Regression Analysis*. Wiley Series in Probability and Statistics. John Wiley & Sons, New York, second edition, 2003. ISBN 978-0-471-41540-4. DOI: 10.1002/9780471722199.

Jian Qing Shi and Taeryon Choi. *Gaussian Process Regression Analysis for Functional Data*. Chapman and Hall/CRC, New York, August 2011. ISBN 978-0-429-15106-4. DOI: 10.1201/b11038.

Stan Development Team. Stan Modeling Language Users Guide and Reference Manual, Version 2.32, 2023. URL: https://mc-stan.org.

Nikolaus Umlauf, Nadja Klein, and Achim Zeileis. BAMLSS: Bayesian Additive Models for Location, Scale, and Shape (and Beyond). *Journal of Computational and Graphical Statistics*, 27(3):612–627, July 2018. ISSN 1061-8600. DOI: 10.1080/10618600.2017.1407325.

Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood. An Introduction to Probabilistic Programming. arXiv, October 2021. DOI: 10.48550/arXiv.1809.10756.

Gabriele Villarini, James A. Smith, Francesco Serinaldi, Jerad Bales, Paul D. Bates, and Witold F. Krajewski. Flood frequency analysis for nonstationary annual peak records in an urban drainage basin. *Advances in Water Resources*, 32(8):1255–1266, August 2009. ISSN 0309-1708. DOI: 10.1016/j.advwatres.2009.05.003.

Simon N. Wood. *Generalized Additive Models: An Introduction with R, Second Edition*. Chapman and Hall/CRC, New York, second edition, May 2017. ISBN 978-1-315-37027-9. DOI: 10.1201/9781315370279.

Simon N. Wood. mgcv: Mixed GAM Computation Vehicle with Automatic Smoothness Estimation, March 2023. URL: https://cran.r-project.org/package=mgcv.

Thomas W. Yee. *Vector Generalized Linear and Additive Models: With an Implementation in R*. Springer Series in Statistics. Springer, New York, 2015. ISBN 978-1-4939-2818-7. DOI: 10.1007/978-1-4939-2818-7.

Appendix A

Liesel: A Probabilistic Programming Framework for Developing Semi-Parametric Regression Models and Custom Bayesian Inference Algorithms

Liesel: A Probabilistic Programming Framework for Developing Semi-Parametric Regression Models and Custom Bayesian Inference Algorithms

A Preprint

Hannes Riebl[*]
Chair of Statistics
University of Göttingen
Humboldtallee 3, 37073 Göttingen, Germany
hriebl@uni-goettingen.de

Paul F.V. Wiemann[*]
Chair of Statistics
University of Göttingen
Humboldtallee 3, 37073 Göttingen, Germany
pwiemann@uni-goettingen.de

Thomas Kneib
Chair of Statistics
University of Göttingen
Humboldtallee 3, 37073 Göttingen, Germany
tkneib@uni-goettingen.de

September 23, 2022

Abstract

Liesel is a probabilistic programming framework focusing on but not limited to semi-parametric regression. It comprises a graph-based model building library, a Markov chain Monte Carlo (MCMC) library with support for modular inference algorithms combining multiple kernels (both implemented in Python), and an R interface (RLiesel) for the configuration of semi-parametric regression models. Each component can be used independently of the others, e.g. the MCMC library also works with third-party model implementations. Our goal with Liesel is to facilitate a new research workflow in computational statistics: In a first step, the researcher develops a model graph with pre-implemented and well-tested building blocks as a base model, e.g. using RLiesel. Then, the graph can be manipulated to incorporate new research ideas, before the MCMC library can be used to run and analyze a default or user-defined MCMC procedure. The researcher has the option to combine powerful MCMC algorithms such as the No U-Turn Sampler (NUTS) with self-written kernels. Various tools for chain post-processing and diagnostics are also provided. Considering all its components, Liesel enables efficient and reliable statistical research on complex models and estimation algorithms. It depends on JAX as a numerical computing library. This way, it can benefit from the latest machine learning technology such as automatic differentiation, just-in-time (JIT) compilation, and the use of high-performance computing devices such as tensor processing units (TPUs).

Keywords Bayesian models · Markov chain Monte Carlo · probabilistic graphical models · Python · R · semi-parametric regression

1 Introduction

In this article, we introduce Liesel,[2] a probabilistic programming framework for the development and estimation of a broad range of Bayesian models in Python. The framework, named after a fountain in its birth city Göttingen, Germany, allows the user to represent statistical models as directed acyclic graphs (DAGs) and to implement tailor-made Markov

[*] Hannes Riebl and Paul F.V. Wiemann contributed equally to the development of the Liesel software and to this preprint.

[2] https://liesel-project.org

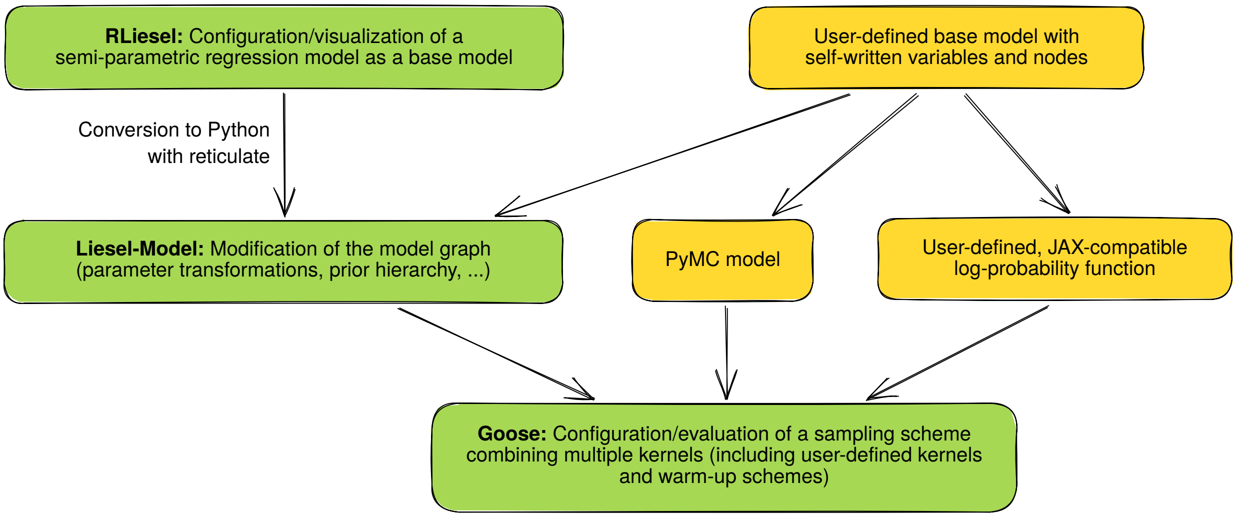

RLiesel: Configuration/visualization of a semi-parametric regression model as a base model
User-defined base model with self-written variables and nodes
Conversion to Python with reticulate
Liesel-Model: Modification of the model graph (parameter transformations, prior hierarchy, ...)
PyMC model
User-defined, JAX-compatible log-probability function
Goose: Configuration/evaluation of a sampling scheme combining multiple kernels (including user-defined kernels and warm-up schemes)

Figure 1: The standard workflow using the Liesel framework. When working with semi-parametric regression models, the first step is usually to create a Liesel model with the help of RLiesel. Then, the model graph is manipulated to accommodate newly developed features, and finally, Goose is used to develop an MCMC algorithm combining standard components like NUTS with user-defined kernels if required. The framework is, however, very flexible. The Liesel-Model library is not limited to semi-parametric regression models but can handle any Bayesian network expressed as a DAG. Goose communicates with the model via an interface which is also available for PyMC models or even self-written, JAX-compatible model representations.

chain Monte Carlo (MCMC) algorithms. Liesel provides many default components for these tasks, which are easy to extend and liberate the researcher from the time-consuming duty of re-implementing the basic components of their models and inference algorithms, giving them the opportunity to focus on the novel aspects of their research. This way, Liesel meets the requirements of many computational statisticians working on new methods or extensions of existing ones. Currently, the framework is particularly useful for developing semi-parametric regression models, since it includes all components required for this model class, but it can easily be extended beyond these models.

The Liesel framework consists of three main components: Goose, an MCMC library, Liesel-Model, a class system for representing statistical models as DAGs, and RLiesel, an R interface to conveniently set up semi-parametric regression models. The components and their relationships are illustrated in Figure 1. A standard workflow with Liesel involves the following steps: First, a semi-parametric regression model is configured with RLiesel, returning a Liesel-Model object. Second, the Liesel-Model object is modified for the research question at hand if required. Third, MCMC estimation is performed with Goose, potentially with different sampling schemes. In the end, Goose's utility functions can be used for model and estimation diagnostics.

Before we proceed to describe the three components of Liesel in more detail, we would like to point out that Liesel is by no means limited to semi-parametric regression models. In fact, the Liesel-Model library can be used to represent any model falling into the category of Bayesian networks, including, for example, regression models, spatial models, change-point models, Gaussian process models or Bayesian neural networks. For this rich model class, which may involve discrete model parameters, there is, to the best of our knowledge, no one-size-fits-all MCMC algorithm. For this reason, Goose encourages the researcher to use their expertise to design an optimal sampling scheme for their specific problem by providing a set of building blocks, which can be used to extend and replace standard MCMC algorithms. Moreover, Goose is not limited to the Liesel-Model library. As indicated in Figure 1, the Liesel framework is designed to be modular, which allows Goose to be agnostic about the concrete model implementation. Goose can also be used to estimate PyMC models or user-defined, JAX-compatible model implementations.

1.1 Software components

Liesel-Model The model building library of Liesel (called Liesel-Model in this article to distinguish it from the Liesel probabilistic programming framework as a whole) facilitates the development of complex statistical models allowing the user to represent them as directed acyclic graphs (DAGs). DAGs are easy to reason about and to manipulate. In Liesel, each node of a DAG represents either data or a computation. The edges indicate data flow or, put differently, how the value of a node depends on the other nodes. Hence, the relationship between the model parameters and the conditional distributions of the model naturally translates to a DAG.

Liesel provides methods to alter, remove or replace subgraphs of a model. This way, the user can extend or modify a given model, for example, a semi-parametric regression model created with RLiesel. More specifically, a prior in the model hierarchy can be replaced by updating the corresponding subgraph. This feature makes Liesel especially well-suited for the development of new statistical models, and in combination with RLiesel, it can simplify research on semi-parametric regression models significantly.

Goose Liesel's MCMC library is called Goose. To perform MCMC estimation, one needs to construct a Markov chain with an equilibrium distribution that matches the target distribution, i.e. the posterior distribution. The chain is simulated for a given certain number of iterations, and the draws from the chain are used to approximate the posterior distribution. While a valid MCMC algorithm is mathematically guaranteed to converge to the posterior distribution, the convergence can be slow in practice. For this reason, most MCMC algorithms need to be tuned, i.e. they need to learn some hyperparameters during a warmup phase to work efficiently.

Goose supports the user in building an MCMC algorithm for their estimation target by offering a broad range of well-tested kernels that can be combined in flexible ways to construct problem-specific MCMC algorithms. In this context, a kernel is an algorithm to transition a part of the parameter vector to a new state within an MCMC iteration. Most kernels in Goose also implement an automatic tuning procedure, which guarantees a high computational efficiency without requiring a manual adjustment of the kernel hyperparameters. The user can combine standard kernels like the No-U-Turn Sampler (NUTS) provided by Liesel with self-implemented ones, e.g. specific Gibbs updates. Of course, Goose also supports using a single kernel like NUTS for the full parameter vector as in Stan.

RLiesel The RLiesel package for R is built on top of the Liesel-Model library. It can be used to configure semi-parametric regression models with the convenient R formula notation. The models are represented as DAGs using the Liesel node and model classes and can be manipulated to incorporate new developments, e.g. new predictor components or prior hierarchies. Finally, the user can take advantage of a default sampler setup or build a custom MCMC algorithm for their model using Goose. RLiesel is based on the `reticulate` package, which allows for a seamless integration of Python and R. With RLiesel, we strive to make Liesel accessible to the statistics community, where R is the predominant language, and to allow for the integration of Liesel with many popular R-based post-sampling utilities.

RLiesel does not only demonstrate how Liesel can be used to implement complex statistical models, but it can also serve as a solid basis for further methodological research on the popular model class of semi-parametric regression. Semi-parametric regression has received a lot of attention among applied statisticians in recent years and is closely related to the concepts of structured additive distributional regression (Klein et al., 2015a) and generalized additive models for location, scale and shape (GAMLSS, Rigby and Stasinopoulos, 2005). These models allow the researcher to explore complex relationships between explanatory and response variables including linear, non-linear, random and spatial effects. Many of them are also multi-predictor models, where different features of the response distribution such as variance, skewness or kurtosis can be related to covariate information. Due to its generality, semi-parametric regression can be understood as an "umbrella" model class comprising many interesting models, which pose a broad range of statistical and computational challenges. RLiesel and Liesel allow the statistician to address these issues with a set of well-tested building blocks, an intuitive graph-based model representation and API, and a modular library for MCMC inference. This is particularly important due to the complexity of the model class, which would make an implementation from scratch a very time-consuming task.

1.2 Related software

Most statistical software packages for Bayesian inference can be classified into software for a specific model class on the one hand and general probabilistic programming languages (PPLs) on the other hand. Liesel and RLiesel try to cover a middle ground between these two approaches: RLiesel facilitates the definition of semi-parametric models, while Liesel-Model and Goose are capable of expressing and estimating a broad range of statistical models. Hence, Liesel has similar capabilities as general-purpose PPLs like Stan (Stan Development Team, 2022), JAGS (Plummer, 2022), NIMBLE (the successor to BUGS, de Valpine et al., 2017) or PyMC (Salvatier et al., 2016). Unlike these software projects, however, Liesel features a graph representation allowing for the modification of the model before estimation. Furthermore, with Liesel, users have full control of the estimation algorithm. Stan and JAGS provide only very limited options to customize the MCMC algorithm. In Stan, NUTS or HMC can be used, or alternatively a mean-field variational inference method. Certain parameters of the samplers, e.g. the initial step size or the target acceptance rate, can be configured. However, block-based sampling is not possible and user-implemented samplers cannot be integrated. Moreover, discrete parameters cannot be modeled with Stan, since it relies on gradient-based samplers.

Compared to Stan, NIMBLE allows for a more detailed configuration of the MCMC algorithm. For example, the default samplers can be reordered or replaced, even with user-defined samplers. In contrast to Liesel, NIMBLE misses capabilities for automatic differentiation and consequently does not provide any gradient-based samplers. Moreover, NIMBLE restricts the compilation of user-defined functions to a subset of the R programming language, which makes third-party libraries difficult to use, while Liesel can wrap code of other JAX-based libraries. PyMC also offers some options to customize the MCMC algorithm but does not go as far as Liesel, and similar to other general-purpose PPLs, does not feature a mutable model object.

For complex models or large datasets, general-purpose PPLs may be slow or unable to sample the model at all. In these situations, model-specific software remains important, and modeling frameworks with customizable MCMC algorithms like Liesel or PyMC may serve as a basis for the implementation of model-specific solutions.

Its flexible model building library sets Liesel apart from other more specialized software. Similar to `brms` (Bürkner, 2017), which provides an interface for various types of multi-level models in Stan, RLiesel provides an interface for semi-parametric regression models in Liesel. RLiesel's features are comparable to other software in the field like `mgcv` (Wood, 2022), `gamlss` (Stasinopoulos et al., 2017), `GJRM` (Marra and Radice, 2022), BayesX (Brezger et al., 2005) and `bamlss` (Umlauf et al., 2021). Its approach is different, however, in that the intermediate graph-based model representation can be modified and extended, allowing for the implementation of new models that are derived from a base model. BayesX was one of the first software packages for fast MCMC inference in semi-parametric regression models with spatial covariate effects. The software cannot be extended easily, however, restricting the user to the pre-defined predictor components (i.e. linear, non-linear, spatial covariate effects, etc.). `bamlss` is another Bayesian software that allows the user to define their own predictor components, which need to be linear in a basis expansion of the covariates, and the corresponding regression coefficients need to follow a (potentially degenerate) multivariate normal prior. In that regard, the model graph of Liesel is more expressive and more flexible. The inference procedure in `bamlss` can be configured with the `optimizer` and `sampler` arguments, but a comprehensive collection of MCMC kernels as in Goose is missing. Automatic differentiation and high-performance computing hardware are also not supported in `bamlss`. Finally, the packages `mgcv` and `GJRM` are not primarily focused on Bayesian inference, although `mgcv` offers an interface to JAGS using the `jagam()` function. In contrast to Liesel, both packages have an exclusive focus on semi-parametric regression using basis function approaches.

1.3 Technology stack

Liesel uses a modern machine learning technology stack for the efficient implementation of the model graph and the MCMC kernels. In particular, Liesel depends on the Python packages NumPy (Harris et al., 2020), JAX (Bradbury et al., 2022), BlackJAX (Lao and Louf, 2022) and TensorFlow Probability (Dillon et al., 2017). JAX, a library for scientific computing with support for automatic differentiation (AD) and just-in-time (JIT) compilation, is of particular importance for Liesel, since its features enable the implementation of computationally efficient inference algorithms. For example, when using reverse-mode AD, the value and the gradient of the log-posterior of a model can both be evaluated in the same amount of time – up to a constant. Furthermore, JAX supports using CUDA-enabled graphics cards for its computations, and running them on even more powerful tensor processing units (TPUs) or networks of those.

Liesel runs on Linux, macOS and with some limitations on Windows,[3] and can be used on laptops, desktop computers and servers. Liesel's development is hosted on GitHub,[4] where bugs can be reported and new features can be requested. The latest release of Liesel, 0.1.3 at the time of writing, is also available on the Python Package Index (PyPI).

The remainder of this article is organized as follows: In Section 2, the Liesel-Model library is discussed. Section 3 describes Liesel's MCMC library Goose, its main design goals, and the interfaces that allow the user to implement their own MCMC kernels and warmup schemes. RLiesel, the R interface for semi-parametric and distributional regression is covered in Section 4 together with some theoretical background on these model classes. Finally, Section 5 describes a case study showing how the components of the Liesel framework can be used together to evaluate different MCMC algorithms on a semi-parametric regression model. The article concludes with a discussion in Section 6.

[3] JAX, one of Liesel's dependencies, does not provide official builds for Windows. However, JAX can either be built by the user or run using the Windows Subsystem for Linux (WSL).

[4] `https://github.com/liesel-devs/liesel`

2 Liesel: Developing probabilistic graphical models

Please note: *The model building library of Liesel is going to receive a major update in version 0.2, which we plan to release in fall 2022. The arXiv preprint will be updated after the release to reflect the changes in version 0.2. For this reason, we focus on the abstract concepts and do not present any code examples in the current version of this section.*

The model building library of Liesel allows the user to express a broad range of (typically Bayesian) statistical models as probabilistic graphical models (PGMs). Particular attention is paid to the representation of semi-parametric regression models, which are described in Section 4, and for which a number of convenience functions are provided. In general, however, almost any statistical model can be expressed with Liesel. The PGM representation allows for a convenient factorization of the log-probability of the model (or the unnormalized log-posterior in a Bayesian context). It is also the basis for the user interface that can be used to update the nodes in a natural way and to modify the structure of the graph (e.g. by adding or removing nodes or edges).

2.1 Probabilistic graphical models and directed acyclic graphs

A PGM uses a graph to express the conditional dependence and independence between a set of random variables. For Bayesian models, one typically relies on directed acyclic graphs (DAGs) to represent hierarchical structures without any loops or circular dependencies, permitting the factorization of the joint probability into a product of conditional probabilities. More precisely, if $M = (X, E)$ is a DAG with nodes $x \in X$ representing random variables and edges $e \in E$ representing conditional dependencies between them, the joint probability of M can be written as

$$\prod_{x \in X} p\big(x \mid \mathrm{Inputs}(x)\big),$$

i.e. the product of the probabilities of the individual nodes conditional on their inputs (or parents). The inputs of a node $x \in X$ are all nodes $x' \in X$ for which x and x' are not conditionally independent given the other nodes of the model.

2.2 Nodes and models in Liesel

Liesel uses Python classes to implement and enrich the mathematical concept of a node in a PGM. A node has two important properties: a value and a log-probability, which is the evaluation of the log-probability density or mass function of the node at its value. To keep both properties in sync, i.e. to avoid an inconsistent state, the node class comes with methods for setting its value and updating its state. The model class, on the other hand, represents a PGM and can hold a number of nodes. It provides methods for the evaluation of the model log-probability and for updating the nodes in a topological order. The model graph can also be visualized conveniently.

The nodes are able to cache their value and log-probability, meaning that the model graph is stateful. The results of expensive mathematical operations can be stored directly in the graph, enabling performance improvements for MCMC sampling, especially if multiple parameter blocks are used. If required, the user can implement new types of nodes and models due to the modular and extensible design of Liesel. More details on the key features of the nodes and models are provided in the following paragraphs.

Nodes Liesel extends the concept of a node in a PGM, where nodes are used to represent random variables, and adds a distinction between so-called "strong" and "weak" nodes. Strong nodes have a value that is either fixed or set by an inference algorithm such as a sampler or optimizer. With some rare exceptions, the random variables of a model are strong nodes and can represent observed data (e.g. the response of a regression model) or a model parameter (in a Bayesian context). Conversely, not all strong nodes are random variables. Hyperparameters or design matrices are examples of nodes strong without an associated probability distribution.

In contrast, weak nodes represent functions of their inputs. These functions are usually deterministic and describe the mappings between the random variables of a model and their probability distributions. Weak nodes can also represent pseudo-random functions, in which case however, they require the state of the PRNG (stored in a strong node) as one of their inputs. The weak nodes can always be recomputed from the strong nodes, and hence, the state of a model is uniquely defined by the strong nodes. Weak nodes can be used to cache the results of expensive computations, because their value only needs to be updated when their inputs have changed. Node subclasses can implement weak nodes representing commonly used functions. By default, Liesel comes with a number of weak nodes facilitating the development of semi-parametric regression models.

If a node has a probability distribution, its log-probability is the evaluation of its probability mass or density function at its current value. For convenience, the log-probability of a node without a distribution is defined to be zero. Summing up the node log-probabilities gives the model log-probability, which can be interpreted as the unnormalized log-posterior

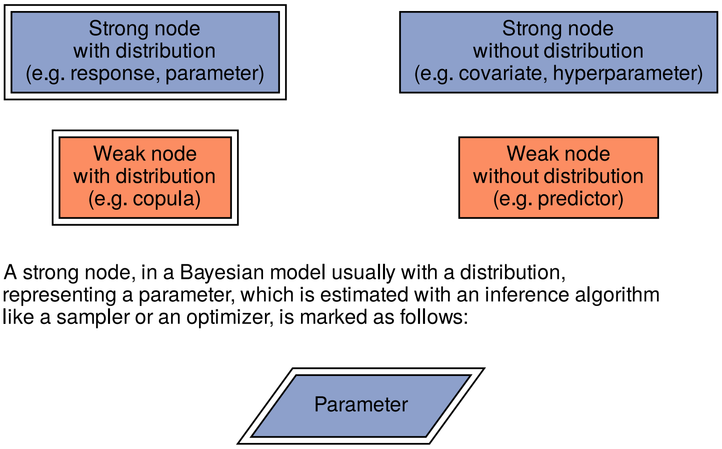

Strong node
with distribution
(e.g. response, parameter)
Strong node
without distribution
(e.g. covariate, hyperparameter)
Weak node
with distribution
(e.g. copula)
Weak node
without distribution
(e.g. predictor)
A strong node, in a Bayesian model usually with a distribution, representing a parameter, which is estimated with an inference algorithm like a sampler or an optimizer, is marked as follows:
Parameter

Figure 2: The nodes of a Liesel model can be strong (blue) or weak (orange), and can have a probability distribution (double border) or not (single border). Weak nodes are functions of their inputs and can always be recomputed from the strong nodes of the model. Nodes with a distribution have a log-probability that is part of the model log-probability. For a graphical representation of a concrete semi-parametric regression model, see Figure 5.

in a Bayesian context. The log-posterior can be decomposed into the log-likelihood (considering only the observed nodes) and the log-prior (considering only the parameter nodes).

Liesel supports probability distributions that follow the class interface from TensorFlow Probability (TFP). Thus, all distributions from TFP can be used with Liesel and new ones can be implemented. One feature of TFP that is particularly useful for Bayesian statistics is the possibility to transform distributions with bijectors. When defining a transformed distribution, TFP automatically adjusts the log-probability with the log-determinant of the Jacobian of the bijector. For an overview of the different node types – strong and weak, with and without a probability distribution – see Figure 2.

Finally, we provide a concrete example and describe which node types would be used to represent a generalized linear model (GLM) in Liesel: The response vector $\boldsymbol{y}$ and the design matrix $\mathbf{X}$ of a GLM are the observed data and would be two strong nodes. While the design matrix is fixed, the response is assumed to follow a probability distribution from the exponential family such as a Poisson or gamma distribution. The vector of regression coefficients $\boldsymbol{\beta}$ is the only model parameter and would be another strong node. In a Bayesian context, the regression coefficients are assigned a prior distribution, whose hyperparameters would again be strong nodes. In contrast, the linear predictor $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta}$ would be a weak node representing a simple matrix-vector product. The expected value of the response $\boldsymbol{\mu} = h(\boldsymbol{\eta})$ is the element-wise evaluation of the response (or inverse link) function h at the linear predictor $\boldsymbol{\eta}$ and would be encoded in separate weak node.

Models A Liesel model is a collection of nodes with properties for the model log-probability, the log-likelihood and the log-prior. Upon initialization, the model computes and stores a topological order of the nodes, which is required for updating the model. The API allows the user to extract and set the state of the model, that is, the values and log-probabilities of the nodes. If some of the nodes have a random seed as an input, the model can manage the PRNG state by splitting and distributing a JAX PRNG key.

The key feature of the model is its update mechanism, which also supports partial updates. If the value of a strong node is modified, its outputs (i.e. nodes that have the modified node as one of their inputs) are recursively flagged as outdated. By calling the update method on the outdated nodes in a topological order, a consistent state can be restored. This is exactly how the update mechanism of the model works. For situations when only a subset of the nodes is of interest and a full update of the model graph is unnecessary, a partial update can be triggered through the model by specifying the target nodes of the update.

The nodes and the model in Liesel follow a stateful, object-oriented approach, which is incompatible with JAX's requirement for pure, stateless functions. To take full advantage of JAX's and Goose's features for JIT compilation, the computations need to be separated from the state of the model. For this purpose, Liesel provides helpers to extract pure functions from the model, which can be used to compute the log-probability and to update the state. These functions are also used in the model interface that can connect the model with Goose.

2.3 Benefits of using Liesel

Goose, the MCMC library that comes with Liesel, can be used independently of the model building library. When using Goose, the user can decide whether their model is best represented with Liesel, PyMC or a self-written log-probability function. Comparing these different approaches, we see the following particular benefits of using Liesel:

Caching Weak nodes can be used to cache the results of expensive computations. This feature is particularly useful for efficient MCMC sampling with multiple parameter blocks, as supported by Goose. Using weak nodes as a cache, the results from the other branches of a tree-like model graph can be recycled when updating the branches individually. Further performance improvements can be achieved with Liesel's partial updates of the model graph, allowing the user to compute only those quantities that are relevant for a given operation.

Graph manipulations The graph of a Liesel model can be modified, allowing for a workflow with a base model, which can be customized to implement new variants of the model. This approach is most convenient if the base model is a semi-parametric regression model that can be configured with RLiesel (Section 4). RLiesel provides many model components for semi-parametric regression, e.g. different spline bases, penalties and response distributions.

Hackability Liesel tries to get out of the way of the user who is extending a model or implementing a new one. The design of the node and model classes is simple and follows the principle of least astonishment. When in doubt, less surprising behavior is favored over more convenience. New operations for a model can be implemented as weak nodes using JAX, which provides a familiar, NumPy-like user interface.

Visualization The graph of a Liesel model is composed of statistically meaningful nodes with values and log-probabilites. It is a wrapper around the computational graph of the model and can be plotted using the functions provided by Liesel. The visualization of the model graph can be useful for various purposes, including debugging or strengthening the intuition about the underlying statistical model.

3 Goose: A toolbox for modular MCMC algorithms

The Liesel framework includes a library named Goose for tailoring MCMC algorithms to specific estimation problems. Goose provides the means for statisticians to develop their own MCMC algorithms that fit the models they are working on better than generic samplers. Goose assists the statistician in three ways: First, by using Goose, they are freed from tedious bookkeeping tasks like storing the sample chains, managing the PRNG state or parallelizing the code to run multiple chains. Second, Goose provides the building blocks of an MCMC algorithm called kernels. A kernel is an algorithm that transitions the parameter vector or (in a blocked sampling scheme) a part of it within an MCMC iteration. Kernels can also define warmup procedures allowing them to learn their hyperparameters and thus removing the need to set them by hand. Third, a well-defined interface allows the combination of user-implemented problem-specific kernels with the default kernels in case the kernels that are shipped with Goose are not sufficient for the estimation problem.

All in all, Goose enables users to construct entirely new algorithms but also to use existing building blocks and combine them in new ways to match the estimation problem at hand. Statisticians using Goose can focus on how one MCMC transition should be performed. In this section, we introduce Goose in detail and our key design choices. Some implementation details are also discussed.

3.1 The primary design goals

The general goal of providing a modular framework for MCMC inference for statistical models can further be broken down into the following more specific design goals:

- Goose should free the user from monotonous tasks that are repeatedly encountered when implementing MCMC algorithms. Among these are storing the intermediate states, multi-chain management, tracking errors and debug messages, and calling tuning algorithms at the right time.
- Goose should allow the user to decide how to transition the model parameters from one to the next MCMC iteration. In Goose, we do that by letting the user combine multiple transition kernels. Each kernel moves a part of the parameter vector or, if only one kernel is used, the entire parameter vector using a valid MCMC transition.
- Goose should have a mechanism to tune the transition kernels automatically during a warmup phase and should thereby avoid that the user needs to tune the kernel hyperparameters by hand.

- The user should have full control over the combined MCMC algorithm. That means, in particular, that all defaults must be changeable, but even more importantly, Goose must allow the implementation of user components. Therefore, the framework should be based on a collection of modular components with well-documented interfaces. The user should be able to compose and extend the components in a flexible, yet straightforward way.
- Goose must support continuous and discrete model parameters.
- Liesel models should be first-class citizens and easy to set up with Goose. However, Goose should be a general MCMC framework that can be used with any JAX model, e.g. a PyMC model or a hand-coded model by the user.
- Goose strives to be convenient to use and fast. To achieve these goals, Goose provides pre-implemented components of popular MCMC algorithms like HMC and NUTS. Furthermore, Goose makes heavy use of JAX's capabilities for automatic differentiation (sparing the user the implementation of derivatives) and just-in-time compilation (speeding up the repeated evaluation of the log-probability of the model). For this reason, the models and the components of the MCMC algorithms need to be expressed in JAX.
- Whenever possible, Goose should wrap well-tested MCMC kernels from other libraries such as the NUTS and HMC kernels from BlackJAX. This way, we can avoid re-implementing complex algorithms, which would be unnecessarily error-prone, while extending the user base of existing projects like BlackJAX.

However, there are also aspects that are outside the scope of Goose. For instance, Goose does not check the mathematical correctness of the sampling schemes. It is up to the user to design a valid MCMC algorithm. The results from Goose should generally be reproducible on the same system. However, reproducibility between different hardware cannot be guaranteed due to small differences in the floating point arithmetic. These differences may add up to observable differences during many MCMC iterations using modern MCMC algorithms.[5]

3.2 Main components of Goose

Goose is composed of many classes and interfaces. The design boils down to a few central pieces users must understand to successfully use Goose as their tool to create MCMC algorithms in a few steps. A deeper understanding is required to write extensions. The most important building blocks and their relationships are illustrated in Figure 3. We describe their roles here. Note that we sometimes refer to the model parameters as the "position".

Engine The `Engine` class is the central part of Goose and acts as a coordinating entity, hiding a big part of the complexity from the user. In particular, after the user has decided how the transitions should be executed, it makes sure that the right functions and methods are called at the right time guaranteeing that the transitions of the position happen as requested. Moreover, the engine keeps track of the sampling history and advances through the different sampling phases (e.g. the warmup and posterior phase). It also coordinates the PRNG state and provides the top-level user interface.

Kernel A kernel object performs a single MCMC transition, i.e. an update of the position or some elements of the position. The update must be a valid MCMC transition, for example based on the Metropolis-Hastings algorithm. The `Kernel` interface describes how the engine can interact with the kernels. The user can either use pre-implemented kernels or implement new kernel classes adhering to the kernel interface.

Epoch An epoch is a series of MCMC iterations. The `EpochConfig` class describes an epoch. Epochs are used to communicate to the kernels which phase of the MCMC algorithm they are in and which operations they are allowed to perform in this phase. More specifically, we divide the sampling process into a warmup and a posterior phase. Samples from the posterior phase are expected to be valid MCMC samples. In contrast, during the warmup phase, the chain may not yet have converged, and during the so-called adaptation epochs, the Markov property may be violated. This way, we allow the kernels to learn their hyperparameters during the adaptation epochs in the warmup phase. If done right, this can spare the user the manual tuning of the kernel hyperparameters, and it can lead to more efficient sampling in the posterior phase.

The simplest setup would contain only two epochs: a burn-in epoch (part of the warmup phase) and a posterior epoch (part of the posterior phase), each containing multiple MCMC iterations. On the other hand, a more complex setup can include multiple adaptation epochs in the warmup phase.

[5] Exact reproducibility is limited for many modern computational tools. See for example Stan's reference manual (`https://mc-stan.org/docs/reference-manual/reproducibility.html`) or the corresponding section in Liesel's tutorial book (`https://liesel-devs.github.io/liesel-tutorials/reproducibility.html`).

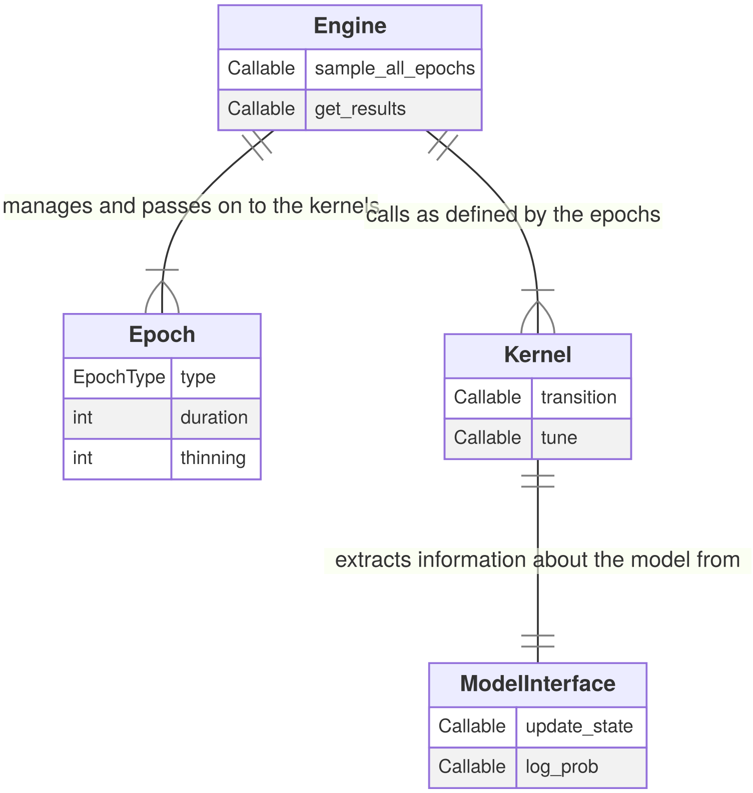


Figure 3: Entity-relationship diagram of Goose's main components. Only the most important classes, fields and methods are shown here.

ModelInterface The `ModelInterface` describes how the components of Goose can communicate with the model. Most importantly, it describes how the unnormalized log-posterior can be evaluated for a given position. By defining the model interface as an abstraction layer, Goose can easily be used with different model backends.

To set up an MCMC algorithm, the user needs to combine the different components of Goose into one valid engine object that handles the communication between them. However, the constructor of the engine is quite complex. To ease the creation of an engine object, the `EngineBuilder` class can be used. It provides a step-by-step interface for the configuration of an engine.

Using the engine builder, Goose leaves the user with only a few tasks to set up an MCMC sampler. These are: (i) Select the appropriate kernels such that every part of the position is moved and add the kernels to the builder. (ii) Supply the builder with an instance of a model interface so that the engine knows how to communicate with the model. (iii) Set the initial values for the position. (iv) Define a sequence of epochs with the desired warmup scheme and the right number of posterior samples. Goose provides a helper function for this task. (v) Additionally, the user must initialize the random number generator and decide how many chains should be run in parallel. Afterwards, the engine is ready to be used for sampling.

3.3 Some implementation details

To enable a deeper understanding of Goose, we describe how the sampling is performed on an implementation level. We explain in detail how the engine communicates with the kernels and provide an overview of the sequence of these interactions. A simplified sequence diagram of the sampling process is shown in Figure 4. Before the sampling is started with the method `sample_all_epochs()`, the user has to create an engine object as described above. That means a sequence of kernels and a sequence of epochs must be defined, the engine must be connected to the model via the model interface, and the initial position must be set. In the following, we assume that only one kernel is used. However, the extension to multiple kernels is straightforward and described later.

The sampling process is divided into multiple phases, which we call "epochs". Each epoch has a duration, i.e. the number of MCMC iterations that are performed in the epoch, and a type. At the beginning of each epoch, the kernel

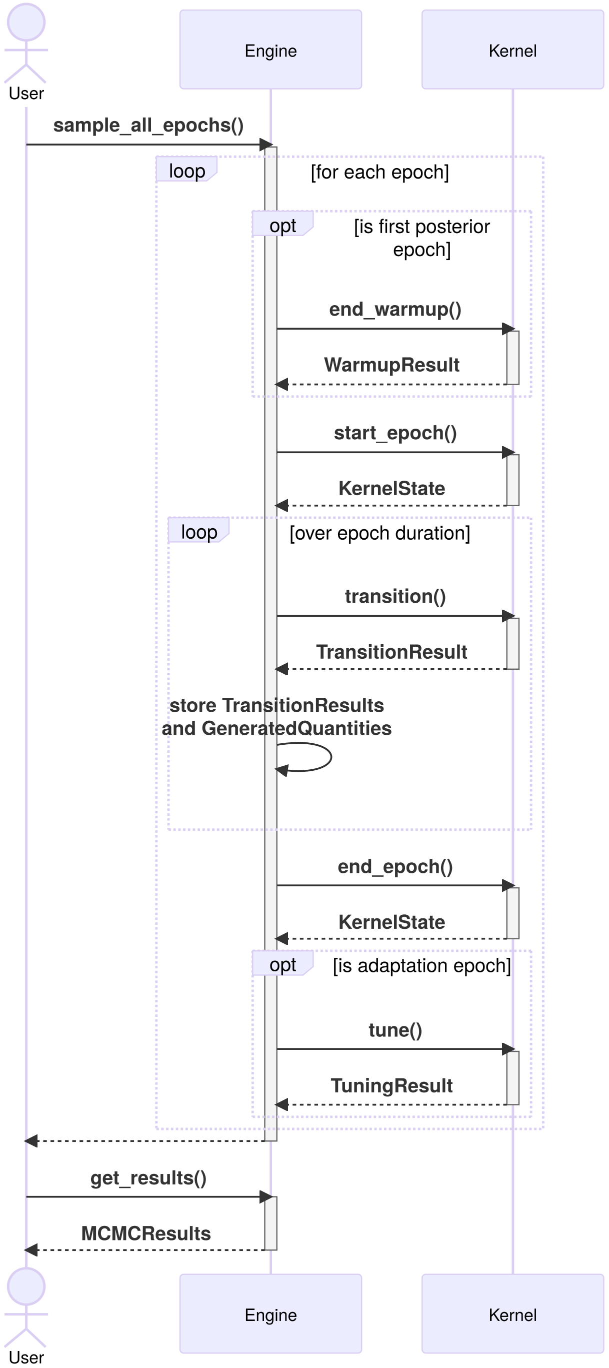


Figure 4: Sequence diagram of the communication between the engine and a kernel. For simplification, we show only one kernel here. However, the extension to multiple kernels is natural by calling the kernel methods in a sequence, which can be achieved by wrapping the kernels in a `KernelSequence` object. The engine provides additional methods to run the epochs one by one and to append epochs, which are not shown here. These methods allow for an interactive use of the engine, while the diagram illustrates a "one-shot" run of an already configured engine.

method `start_epoch()` is called informing the kernel about the new epoch and allowing it to modify the kernel state. The kernel state is a data structure used to store parameters defining the behavior of the kernel. It may be modified during the warmup. The scale of the proposal distribution (also known as the step size) of a random walk kernel serves as an example in this section. The kernel state can also include a cache required to calculate the actual parameters that affect the transitions. Allowing the kernel to change its state at the beginning of an epoch enables it to prepare for the subsequent operations.

Afterwards, the control is handed back to the engine, and it calls the kernel method `transition()` for each MCMC iteration in the current epoch. The transition method is supposed to move the position and return the new position together with additional information (which would typically include whether the position changed, how large the acceptance probability was, whether an error occurred, etc.) to the engine. The engine takes care of storing the position and the additional information. Note that during the warmup phase, the kernels are allowed to change their state in the transition method, which allows for on-the-fly tuning of kernel parameters and updates of the cache. This is required, for example, for the dual averaging algorithm (Nesterov, 2009; Hoffman and Gelman, 2014, Section 3.2) or for Welford's online algorithm for calculating the empirical variance of an element of the position (Welford, 1962).

Once all transitions defined in the current epoch have been carried out, the kernel method `end_epoch()` is called. Again, the kernel can change its state and prepare for the following tuning. To invoke the tuning, the kernel method `tune()` is called if the current epoch is an adaptation epoch. In the adaptive random walk kernel, this method would be the place to calculate the new step size based on Welford's algorithm and update it in the kernel state. The kernel is allowed to request the history of the positions visited in the current epoch. Having the history available facilitates the implementation of certain tuning algorithms.

The outlined process is repeated for each epoch. As soon as the first epoch of the posterior phase is encountered, the kernel method `end_warmup()` is called before the call to `start_epoch()`. It informs the kernel that the warmup phase is over, and subsequent to this call, the kernel must respect the Markov property.

Finally, the user can request the sampling results from the engine and inspect them. The results do not only contain the chain of the visited positions but also meta-information and an error log (e.g. an error is reported if the log-posterior evaluates to $-\infty$). Liesel also provides some utilities for the inspection of the chains.

A more interactive approach is also possible. The user can always add more epochs to continue sampling. One restrictions is that Goose does not allow posterior epochs to be followed by epochs of any other type. The interactive approach is facilitated by the engine methods `append_epoch()` and `sample_next_epoch()`. The user can run a few warmup epochs, inspect the chains, decide if they have reached the typical set and converged, add more warmup epochs if necessary or move on to the posterior epoch otherwise.

Everything that has been said so far can easily be generalized to multiple kernels. In that case, each method call is carried out in a loop over the sequence of kernels defined by the user. Note that the kernels cannot share their state.

If users want to work with custom MCMC transition or tuning methods or extend Goose's collection of kernels, they have to implement a new class that is required to follow the `KernelInterface`. The two most important methods to do so are `transition()` and `tune()`. We describe them in more detail and also provide more information on the implementation of the engine, which is useful to understand the requirements for the kernel methods.

The engine. As described above, the engine orchestrates the sampling process and provides the top-level user interface. It also hides some complexity that arises from using JAX and JIT-compiled functions. Using JAX comes with many benefits, e.g. automatic differentiation (AD) and just-in-time (JIT) compilation. Furthermore, JAX programs can be executed on high-performance devices like GPUs and TPUs. For efficient sampling, the engine automatically groups multiple MCMC iterations into one compilation unit and uses JAX's `jit()` function to compile them together. Thus, the MCMC iterations are performed together on the computing device without the need for communication with the host. This ensures a better performance, especially if the computing device is not the CPU.

One drawback, or rather one limitation, is the requirement of "pureness"[6] for functions to be compiled with JAX. Pureness is not necessarily a disadvantage, because pure functions are easier to reason about for humans and for the compiler. This can result in faster execution times compared to non-pure functions.

Goose needs to guarantee that the compiled functions are pure. This implies that the engine must manage the PRNG state – we use JAX's splittable Threefry counter-based PRNG – as well as the kernel states. Goose requires all kernel

[6] A pure function is a function whose value depends solely on the values of its arguments and which furthermore has no side effects. In JAX and Goose, the concept of pureness is a bit weaker. A function may depend on variables in the environment. However, the values of those variables are then compiled into the function, and therefore, the behavior of the function does not change if the variables are updated later. Consequently, the compiled function is pure.

methods called within the compiled functions (e.g. `transition()` and `tune()`) to be pure, meaning that the kernels cannot store values changing over time in fields but must pass them back to the engine via a `KernelState` object, and receive them again from the engine together with the PRNG state for the next transition.

The transition method. The two most important methods every kernel needs to implement are the `transition()` and the `tune()` method. These methods are called by the engine and need to be pure and jittable.

The purpose of the transition method is to move the position or parts of it using a valid MCMC step, e.g. a Metropolis-Hasting algorithm. The position is a subset of the model state. Through the standardized model interface, the kernel can extract the position from the model state.

The signature of the `transition()` method is as follows:

```
Py> class Kernel:
+       # ...
+
+       def transition(
+           self,
+           prng_key: KeyArray,
+           kernel_state: KernelState,
+           model_state: ModelState,
+           epoch: EpochState,
+       ) -> TransitionResult[KernelState, TransitionInfo]:
+           # ...
+
+       # ...
```

Since the `transition()` method must be pure and MCMC transitions generally involve the generation of random numbers, the state of the PRNG needs to be provided as an argument. In addition, the `transition()` method receives the kernel state, the model state and the epoch state as arguments, and returns a `TransitionResult` object, which wraps the new kernel state, the new model state and some meta-information about the transition, e.g. an error code or the acceptance probability (in a `TransitionInfo` object). An error code of zero indicates that the transition did not produce an error.

All inputs and outputs must be valid "pytrees" (i.e. arrays or nested lists, tuples or dicts of arrays). The structure of these objects, e.g. the shape of the arrays in the kernel state, must not change between transitions. This allows the kernels to have specialized `KernelState` and `TransitionInfo` classes.

Tuning a kernel. The sampling process can be divided into epochs of four types: fast and slow adaptation epochs, burn-in epochs and posterior epochs. The adaptation and burn-in epochs are so-called warmup epochs. During the adaptation epochs, the kernels are allowed to learn their hyperparameters from the history. Samples from the adaptation epochs are usually invalid as MCMC samples, because the Markov property of the chain is violated. In contrast, during a burn-in epoch, the kernels should no longer adjust their hyperparameters and the Markov property should be respected, but the chain may still require some more time to converge. Finally, when reaching the first posterior epoch, the chain should have converged, all transitions should be valid, e.g. there should be no divergent transitions, and hence, the samples should approximate the target distribution appropriately.

The kernel method `tune()` is supposed to update the kernel hyperparameters at the end of an adaptation epoch. The method receives the PRNG state, the model state, the kernel state, the epoch state and optionally the "history", i.e. the samples from the previous epoch, as arguments. It returns a `TuningResult` object that wraps the new kernel state and some meta-information about the tuning process, e.g. an error code. As for the transition, the `TuningInfo` class can be kernel-specific but must be a valid pytree.

The signature of the `tune()` method is as follows:

```
Py> class Kernel:
+       # ...
+
+       def tune(
+           self,
+           prng_key: KeyArray,
+           kernel_state: KernelState,
+           model_state: ModelState,
+           epoch: EpochState,
+           history: Position | None,
+       ) -> TuningResult[KernelState, TuningInfo]:
+           # ...
+
+       # ...
```

Debugging. The engine can be configured to store more information about the sampling process, e.g. for debugging purposes. The extra information can include the log-posterior, log-likelihood, log-prior or any other quantity that can be computed from the model state by a quantity generator. Debugging is further facilitated with the option to store the kernel states for each iteration. Moreover, the engine can store information about the transitions and the tuning such as the acceptance probabilities or the proposals. In any case, the `transition()` and `tune()` methods of the kernels need to return an error code and inform the engine about non-fatal errors and warnings. The engine keeps a log and warns the user about potential problems. Goose's diagnostic tools can further aid the detection of potential sampling issues.

3.4 Standard kernels in Goose

Goose provides several kernels that can be used directly with many models. We discuss some of them here:

RandomWalkKernel The RandomWalkKernel implements a Gaussian proposal distribution and a Metropolis-Hastings acceptance step. The kernel is self-tuning and uses the dual average algorithm to adjust the step size (i.e. to scale the proposal distribution) during fast and slow adaptation epochs, such that a user-defined target acceptance rate, by default of 0.234 (Gelman et al., 1997), is reached.

HMCKernel and NUTSKernel The HMCKernel and NUTSKernel use the gradient of the log-posterior to generate MCMC chains with a low autocorrelation. The implementation of the `transition()` method is based on BlackJAX's implementations of the HMC (Neal, 2011) and NUTS (Hoffman and Gelman, 2014; Lao et al., 2020; Phan et al., 2019) algorithms. Both kernels are able to tune the step size during fast and slow adaptation epochs using the dual averaging algorithm. After slow adaptation epochs, the mass vector or matrix of the impulse is adjusted based on the empirical variance-covariance of the samples from the previous epoch.

IWLSKernel The IWLSKernel is named after the method proposed by Gamerman (1997), which is often used for Bayesian distributional regression models (Brezger et al., 2005). However, Liesel's implementation is also inspired by the roughly equivalent Metropolis-adjusted Langevin algorithm (MALA) with the Riemann metric (Girolami and Calderhead, 2011). This approach allows us to add a step size parameter in a straightforward way, which can then be tuned with the dual averaging algorithm during fast and slow adaptation epochs. More precisely, the IWLSKernel employs a Metropolis-Hastings correction and a Gaussian proposal density, where the mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$ depend on the gradient (score) and the Hessian (Hess) of the log-posterior, i.e.

$$\boldsymbol{\mu} = \boldsymbol{\theta} + {}^{s^2}\!/_{2}\,\mathrm{Hess}(\boldsymbol{\theta})^{-1}\,\mathrm{score}(\boldsymbol{\theta}), \qquad \boldsymbol{\Sigma} = s^2\,\mathrm{Hess}(\boldsymbol{\theta})^{-1},$$

where s denotes the step size and $\boldsymbol{\theta}$ the position vector. The factor $1/2$ that is multiplied with s^2 in the mean vector comes from the Langevin diffusion, which is the basis of the MALA algorithm.

GibbsKernel The GibbsKernel can wrap a user-defined function generating samples from a full conditional into a Goose-compatible kernel. With a Gibbs sampler, no tuning is necessary or possible, and therefore, the GibbsKernel has a trivial `tune()` method returning an empty kernel state.

MHKernel Similar to the GibbsKernel, the MHKernel implements a Metropolis-Hastings sampler as a wrapper around a user-defined function generating proposals based on the current state. If the proposal distribution is asymmetric, the function must also return the Metropolis-Hastings correction factor. An optional step size argument is also provided, which is tuned with the dual averaging algorithm if used.

3.5 Beyond pre-implemented kernels

The default Goose kernels are sufficient to estimate many statistical models with MCMC. However, Goose was specifically designed for cases when specialized kernels are needed. In these situations, new kernel classes adhering to the kernel interface can be implemented. The developer does not need to start from scratch, however. Goose comes with some building blocks that facilitate the implementation of new kernel classes. For example, if a kernel should support dual averaging, Goose can extend the kernel state with the necessary fields. It also comes with functions to calculate the error sum and to adjust the step size. A mixin for Metropolis-Hastings kernels is provided as well.

4 RLiesel: An R interface for semi-parametric regression

In this section, we discuss semi-parametric and distributional regression, the model classes Liesel offers first-class support for, before introducing RLiesel, an R interface that assists the user with the configuration of these regression models in Liesel. We also describe a natural workflow for RLiesel using R Markdown and Quarto.

4.1 Semi-parametric regression

Semi-parametric regression models combine parametric (usually linear) and non-parametric (usually spline-based) covariate effects. The standard semi-parametric regression model is given by

$$y_i = \beta_0 + \boldsymbol{x}_{i1}'\boldsymbol{\beta}_1 + f_2(\boldsymbol{x}_{i2}, \boldsymbol{\beta}_2) + \cdots + f_L(\boldsymbol{x}_{iL}, \boldsymbol{\beta}_L) + \varepsilon_i, \qquad \varepsilon_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2), \tag{1}$$

where the response y_i is modeled as a function of the covariates $\boldsymbol{x}_{i1}$ with parametric effects and the covariates $\boldsymbol{x}_{il}$ with the non-parametric effects $f_l(\boldsymbol{x}_{il}, \boldsymbol{\beta}_l)$ for $l = 2, \dots, L$. The regression coefficients are the intercept β_0, the slope coefficients $\boldsymbol{\beta}_1$ and the spline coefficients $\boldsymbol{\beta}_l$. Fitting the model requires the estimation of the regression coefficients and the variance of the additive Gaussian error term ε_i.

One typical example of a non-parametric covariate effect is the B-spline $f(\boldsymbol{x}_i, \boldsymbol{\beta}) = \boldsymbol{b}(x_i)'\boldsymbol{\beta}$, where $\boldsymbol{b}(x_i)$ is the vector of B-spline basis functions for a fixed set of knots evaluated at x_i. For better readability, the index l is omitted in the remainder of this section. The given B-spline representation is linear in the spline coefficients $\boldsymbol{\beta}$, allowing for a straightforward evaluation of the log-likelihood and the use of efficient estimation techniques. To avoid overfitting, certain smoothness properties can be encouraged through regularization, giving rise to the concept of penalized B-splines, also known as P-splines (Eilers and Marx, 1996; Lang and Brezger, 2004).

In Bayesian statistics, regularization is achieved through informative priors, such as the multivariate normal distribution with the density

$$p(\boldsymbol{\beta} \mid \tau^2) \propto \left(\frac{1}{\tau^2}\right)^{\operatorname{rk}(\mathbf{K})/2} \exp\left(-\frac{1}{2\tau^2}\boldsymbol{\beta}'\mathbf{K}\boldsymbol{\beta}\right), \tag{2}$$

where τ^2 is the variance (or inverse smoothing) parameter, and $\mathbf{K}$ is a (potentially rank-deficient) penalty matrix. For P-splines with equidistant knots, it is common to penalize the second differences of the spline coefficients using the penalty matrix $\mathbf{K} = \mathbf{D}_2'\mathbf{D}_2$, where $\mathbf{D}_2$ is the second-order difference matrix such that $\mathbf{D}_2\boldsymbol{\beta} = \Delta^2\boldsymbol{\beta}$. In this case, the penalty matrix is in fact rank-deficient, implying that additional constraints, usually a sum-to-zero constraint, are required for the identification of the spline coefficients.

The hyperprior on the variance parameter τ^2 is typically weakly informative with support on the non-negative real line. Lang and Brezger (2004) suggest to use the conjugate inverse gamma prior with the hyperparameters $a = b = 0.01$ (or some other small number), allowing us to draw directly from the full conditional. However, priors like the half-Cauchy distribution or half-normal distribution might have better statistical properties in practice (Gelman, 2006; Klein and Kneib, 2016b).

The concept of semi-parametric regression also encompasses other effect types that can be expressed as the inner product of a vector of basis function evaluations and a vector of regression coefficients, e.g. random effects for clustered data or spatial effects. The structure of the penalty matrix $\mathbf{K}$ in the multivariate normal prior (2) depends on the desired effect type. For a random effect, we have $\mathbf{K} = \mathbf{I}$, for an (intrinsic) Gaussian Markov random field, $\mathbf{K}$ arises from the neighborhood structure (Rue and Held, 2005), and for more general spatial effects, Vecchia approximations can be used to construct $\mathbf{K}$ (Katzfuss and Guinness, 2021). Note that the linear effect $\boldsymbol{x}_i'\boldsymbol{\beta}$ also fits into this framework by setting $\mathbf{K} = \mathbf{0}$, reducing the multivariate normal prior (2) to a flat prior. Consequently, parametric and non-parametric covariate effects can be treated the same way in this framework, and are generically referred to as predictor components or smooth terms. Semi-parametric regression is sometimes (perhaps more accurately, but also more verbosely) called structured additive regression. Consult Fahrmeir et al. (2013, Chapters 8 and 9) for more information on predictor components and structured additive regression.

4.2 Distributional regression

Semi-parametric or structured additive regression predictors are often used in the context of distributional regression. These models are also known as generalized additive models for location, scale and shape (GAMLSS) and combine multiple regression predictors for different response parameters, that is,

$$p(y_i \mid \boldsymbol{x}_i, \boldsymbol{\beta}) = p(y_i \mid \theta_1(\boldsymbol{x}_{i1}, \boldsymbol{\beta}_1), \dots, \theta_K(\boldsymbol{x}_{iK}, \boldsymbol{\beta}_K)), \tag{3}$$

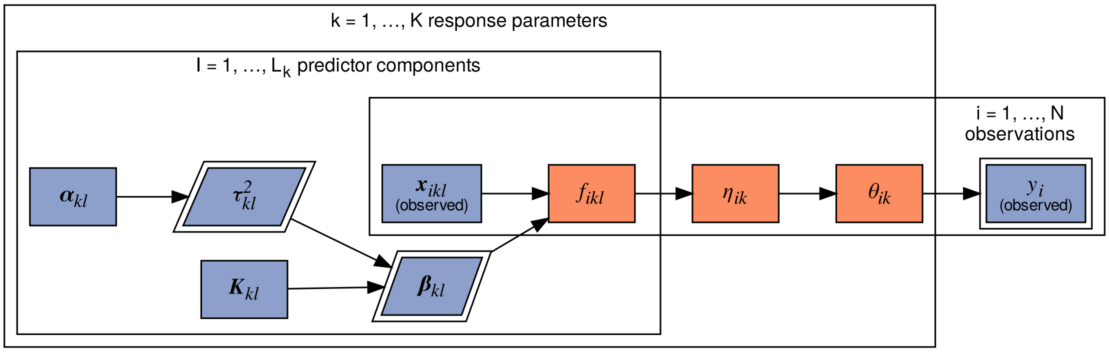

k = 1, ..., K response parameters
l = 1, ..., L_k predictor components
i = 1, ..., N observations
$\boldsymbol{\alpha}_{kl}$
τ^2_{kl}
$\boldsymbol{K}_{kl}$
$\boldsymbol{\beta}_{kl}$
$\boldsymbol{x}_{ikl}$ (observed)
f_{ikl}
η_{ik}
θ_{ik}
y_i (observed)

Figure 5: One possible DAG representation of the semi-parametric distributional regression model (3). The different node types are described in Figure 2: Strong nodes are blue, weak nodes are orange. Nodes with double borders have a probability distribution, and oblique nodes are model parameters. Plate notation is used to indicate the range of the variable indices.

where the response y_i follows a probability distribution with the parameters θ_k for $k = 1, \dots, K$, each of which is modeled as a function of the covariates $\boldsymbol{x}_{ik}$ and the regression coefficients $\boldsymbol{\beta}_k$. In contrast to generalized linear models (GLMs), the response distribution is not limited to the exponential family but can be of any parametric type, including for example non-negative continuous distributions like the Weibull or Pareto distribution. Distributional regression models for count data can take zero-inflation and overdispersion into account (Klein et al., 2015b), while fractional responses (i.e. single or multiple percentages) can be analyzed with the beta or Dirichlet distribution (Klein et al., 2015a). With mixed discrete-continuous distributions, we can add points with a non-zero probability mass to the support of a continuous response distribution. Finally, the distributional regression framework allows us to study multivariate response vectors using either conventional multivariate distributions (Michaelis et al., 2018) or copulas to describe complex dependence structures with arbitrary marginal distributions (Klein and Kneib, 2016a).

In distributional regression, each parameter of the response distribution is modeled with a semi-parametric regression predictor η_{ik} (just as the one in Model (1) in the previous section) and a response (or inverse link) function h_k, such that

$$\theta_k(\boldsymbol{x}_{ik}, \boldsymbol{\beta}_l) = h_k(\eta_{ik}) = h_k(\beta_{k0} + \boldsymbol{x}'_{ik1}\boldsymbol{\beta}_{k1} + f_{k2}(\boldsymbol{x}_{ik2}, \boldsymbol{\beta}_{k2}) + \dots + f_{kL_k}(\boldsymbol{x}_{ikL_k}, \boldsymbol{\beta}_{kL_k})). \tag{4}$$

The response function h_k is a one-to-one mapping of the predictor η_{ik} from the real line to the appropriate parameter space. For positive-valued response parameters, the exponential function is typically used as a response function, and for parameters on the unit interval, the logistic function is a common choice.

The distributional regression model (3) with the semi-parametric predictor (4) is a Bayesian hierarchical model, where the posterior can be factorized as $p\left(\bigcup_{k,l}\{\boldsymbol{\beta}_{kl}, \tau^2_{kl}\} \mid \bigcup_i\{y_i\}\right) = \prod_i p\left(y_i \mid \bigcup_{k,l}\{\boldsymbol{\beta}_{kl}\}\right) \cdot \prod_{k,l} p(\boldsymbol{\beta}_{kl} \mid \tau^2_{kl}) \cdot p(\tau^2_{kl})$. The model graph is a DAG with a tree-like structure, making it a good fit for software like Liesel, PyMC or Stan.

4.3 DAG representations of semi-parametric regression models

One possible DAG representation of the semi-parametric distributional regression model is shown in Figure 5. The strong node $\boldsymbol{\alpha}_{kl}$ denotes the fixed hyperparameters of the prior of the variance parameter τ^2_{kl}. Typically, $\boldsymbol{\alpha}_{kl} = (a_{kl}, b_{kl})' = (0.01, 0.01)'$ in the case of an inverse gamma prior. The choice of the weak nodes is essentially arbitrary: The nodes f_{ikl}, η_{ik} and θ_{ik} could also be merged into a single weak node. In Liesel, we encourage a structure of the model graph that resembles the mathematical formulation of the semi-parametric distributional regression model in Equation (3) and (4). This allows us to provide a number of pre-defined nodes for the components of the model class, which can be combined by the user in different ways.

The DAG representation can also be modified to improve the computational efficiency of the model. In the DAG as shown in Figure 5, the evaluation of the log-probability of $\boldsymbol{\beta}_{kl}$, i.e. the evaluation of the multivariate normal prior (2), requires computing the rank of the penalty matrix $\mathbf{K}_{kl}$. Given that the penalty matrix is usually a fixed hyperparameter, it is wasteful to repeat this expensive operation every time $\boldsymbol{\beta}_{kl}$ or τ^2_{kl} are updated. The performance of the model can be improved by adding a strong node with the pre-computed rank of $\mathbf{K}_{kl}$. This node can then be used as an input for the probability distribution of $\boldsymbol{\beta}_{kl}$, hence avoiding the repeated computation of the matrix rank.

4.4 Setting up semi-parametric regression models with RLiesel

RLiesel is an R interface for Liesel, which can be used to configure semi-parametric distributional regression models. It is implemented as a thin wrapper around the `mgcv` package (Wood, 2022). The entry point to the package is the `liesel()` function, which requires the user to pass in the response data and distribution, and the predictors as arguments. The predictors are specified as R formulas with the extensions from `mgcv` to define non-parametric predictor components. They are passed on to the `gam()` function from `mgcv`, which initializes the design and penalty matrices. Finally, the Liesel model graph is built and filled with the data from `mgcv`. A concrete example how a model can be specified in RLiesel is given in the case study in Section 5.

`mgcv` is the state-of-the-art package for semi-parametric regression in R. It is extremely powerful, supports many different response distributions and predictor components, and is installed with R by default. Other notable features of `mgcv` are the automatic smoothness selection (Wood, 2004) and various multivariate smooth terms. To the best of our knowledge, no package with a comparable set of features exists in Python. Most newer R packages in the domain of semi-parametric regression modeling depend on `mgcv` in one way or another. With our implementation of RLiesel, we follow the same approach and leverage the features of `mgcv` for the use with JAX and Liesel, avoiding the need to re-implement all predictor components in Python.

RLiesel configures the model graph, but does not automatically run an estimation procedure. Goose can be used for MCMC-based estimation, but needs to be configured in Python. For a seamless integration of RLiesel and Goose, we recommend Quarto (Scheidegger et al., 2022) and `reticulate` (Ushey et al., 2022). Quarto allows the user to write and render dynamic documents in Markdown with embedded R and Python code cells, and using `reticulate`, objects can be shared between the R and Python processes at runtime. With this setup, the model can be configured using RLiesel in a R code cell, then exchanged with the Python process, before an MCMC algorithm is developed in another code cell. Finally, the estimation results can be visualized either in Python or R, depending on the user's preferences.

5 Case study: Comparing different sampling schemes

In this case study, we show how RLiesel and Goose can be used to set up and compare different sampling schemes on a simple semi-parametric distributional regression model. Often, a one-size-fits-all MCMC algorithm does not work too well with a specific model. In these cases, one can try to reparametrize the model to improve the performance of the MCMC algorithm, or alternatively, one can try to develop a more suitable sampling scheme. The second approach is the particular strength of Liesel and Goose. Goose facilitates building custom samplers for specific estimation problems, allowing the user to combine different pre-defined and self-written kernels.

We use a dataset of LIDAR measurements, which was collected to determine the mercury concentration in the atmosphere, to evaluate the performance of five sampling schemes combining IWLS, Gibbs, NUTS and HMC kernels in different parameter blocks. For a detailed description of the experiment, see Holst et al. (1996). Two lasers with different wavelengths were emitted by the LIDAR device, and the log-ratio between the signals (the amount of reflected light, y_i) was recorded for each range (the distance the light traveled, x_i). The data is shown in Figure 6 together with an estimate of the mean function. The derivative of the mean function is proportional to the desired estimate of the mercury concentration.

5.1 Gaussian location-scale regression in RLiesel

From Figure 6, the non-linearity and heteroscedasticity of the LIDAR measurements becomes apparent. The semi-parametric Gaussian location-scale regression model

$$y_i \sim \mathcal{N}(\beta_0 + f(x_i), (\exp(\gamma_0 + g(x_i))^2) \tag{5}$$

is able to accommodate these properties of the data. Here, β_0 and γ_0 are the intercepts, and $f(x_i)$ and $g(x_i)$ are P-splines as described in Section 4.1. For the P-splines, we use a cubic B-spline basis and a second-order difference penalty on the regression coefficients. The model belongs to the distributional regression framework as defined in Equation (3), using a Gaussian response distribution and a log-link for the standard deviation.

With RLiesel, we can set up Model (5) as follows:

```
R> library(SemiPar)
R> data(lidar)
R>
R> library(rliesel)
R> use_liesel_venv()
```

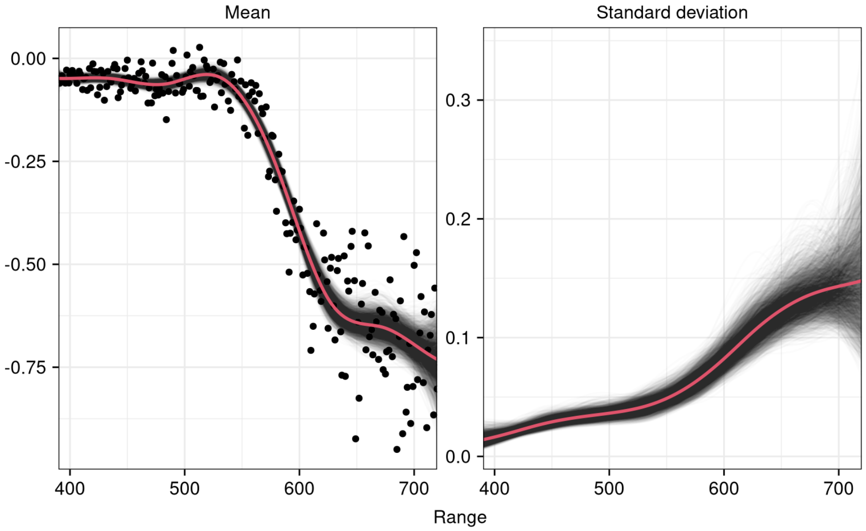


Figure 6: The log-ratio of the LIDAR signals for each range on top of a MCMC sample of 4000 estimated mean functions (left) and 4000 estimated standard deviation functions (right). The red lines mark the posterior mean, the sample was obtained with the IWLS-Gibbs scheme described in Section 5.2.

```
R>
R> model <- liesel(
+    response = lidar$logratio,
+    distribution = "Normal",
+    predictors = list(
+      loc = predictor(~s(range, bs = "ps"), inverse_link = "Identity"),
+      scale = predictor(~s(range, bs = "ps"), inverse_link = "Exp")
+    ),
+    data = lidar
+  )
```

The response variable and distribution, and the semi-parametric regression predictors are passed as arguments to the `liesel()` function. The predictors are specified as one-sided R formulas, where we can use the `s()` function from the `mgcv` package to define spline-based predictor components with the multivariate normal prior (2). The argument `bs = "ps"` indicates that we are using a P-spline. As Liesel depends on TensorFlow Probability (TFP) to represent probability distributions, we need to use the same class and parameter names. Here, the argument `distribution = "Normal"` refers to the class of the same name in TFP, which has the parameters `loc` and `scale` for the mean and the standard deviation of the normal distribution.

5.2 Sampling schemes with different kernels in Goose

For the LIDAR model, we are using the IWLS-within-Gibbs sampling scheme as a benchmark. This scheme is provided as the default in RLiesel and has been propagated in the literature on semi-parametric distributional regression for several years (Klein et al., 2015b). It combines one IWLS kernel for the regression coefficients $\boldsymbol{\beta}$ with one Gibbs kernel for the smoothing parameter τ^2 of each predictor component. Thus, in complex models with many predictor components, it results in a high number of parameter blocks, and sometimes in MCMC chains with a high autocorrelation. Furthermore, the use of the observed Fisher information in the IWLS kernel can cause numerical instabilities. Software packages like BayesX and `bamlss` replace the observed with the expected Fisher information whenever possible to mitigate these problems, but this workaround is model-specific and not possible with automatic differentiation.

Given the shortcomings of the IWLS-within-Gibbs scheme, it is interesting to compare its performance with gradient-based MCMC methods that do not require second derivatives such as HMC or NUTS. Relying only on the gradient, these kernels make it computationally feasible – also in complex models – to update large parameter blocks or the entire parameter vector. HMC and NUTS have been popularized with software like Stan (Stan Development Team, 2022) and PyMC (Salvatier et al., 2016), and are known to work well in many applications (MacKay, 2003, Chapter 30). In the

LIDAR model, the smoothing parameters τ_f^2 and τ_g^2 need to be log-transformed if sampled with HMC or NUTS to guarantee an unconstrained parameter space. The configuration of all five sampling schemes is described in Table 1.

Table 1: The sampling schemes for the LIDAR model. The IWLS kernel was used with the observed Fisher information as a metric (obtained through automatic differentiation). The NUTS kernel was configured with a maximum tree depth of 10 and a diagonal metric (tuned based on the empirical variances of the warmup samples). The HMC kernel was used with 64 integration steps and a diagonal metric. A smaller number of integration steps would have resulted in an insufficient exploration of the posterior distribution. The step size of the IWLS, NUTS and HMC kernels was calibrated with the dual averaging algorithm during the warmup epochs.

	β_0	β_f	τ_f^2 or $\log(\tau_f^2)$	γ_0	γ_g	τ_g^2 or $\log(\tau_g^2)$
IWLS-Gibbs	IWLS	IWLS	Gibbs	IWLS	IWLS	Gibbs
NUTS-Gibbs	NUTS	NUTS	Gibbs	NUTS	NUTS	Gibbs
NUTS1	NUTS					
NUTS2	NUTS			NUTS		
HMC2	HMC			HMC		

Setting up sampling schemes and parameter blocks is straightforward with Goose. To facilitate the configuration of an MCMC engine, a builder class can be used. Through the builder, kernels can be assigned to one or more parameters, the model and initial values can be set, as well as the number of MCMC iterations. Finally, the engine can be built and run. The following code snippet illustrates the procedure for the NUTS2 scheme, but the setup of the other schemes works analogously:

```
Py> builder = gs.EngineBuilder(seed=1337, num_chains=4)
Py>
Py> k1 = ["loc_p0_beta", "loc_np0_beta", "loc_np0_tau2_transformed"]
Py> k2 = ["scale_p0_beta", "scale_np0_beta", "scale_np0_tau2_transformed"]
Py> builder.add_kernel(gs.NUTSKernel(k1))
Py> builder.add_kernel(gs.NUTSKernel(k2))
Py>
Py> builder.set_model(lsl.GooseModel(model))
Py> builder.set_initial_values(model.state)
Py>
Py> builder.set_duration(warmup_duration=1000, posterior_duration=1000)
Py>
Py> engine = builder.build()
Py> engine.sample_all_epochs()
```

5.3 Run time and effective sample size

All sampling schemes from Table 1 converged to the same posterior distribution shown in Figure 6, so we can focus on comparing their efficiency rather than the parameter estimates. The MCMC algorithms were compiled and run on an Intel i7-1185G7 CPU with 8 cores and 3 GHz. The compilation was generally much more expensive than the generation of one chain with 1000 warmup and 1000 posterior iterations (Figure 7). The IWLS-Gibbs and NUTS-Gibbs schemes were particularly slow to compile, presumably because combining two types of kernels means more work for the compiler, while the sampling schemes involving one or two NUTS kernels took most time to run.

The reason for the performance issues with NUTS was that the maximum tree depth of 10 was reached in about 90% of the posterior iterations for the NUTS1 scheme, and in 75% for NUTS2. The problem did not occur with the NUTS-Gibbs scheme, where we split the regression coefficients β and the smoothing parameters τ^2 into separate blocks. We tried to improve the performance of the NUTS1 and NUTS2 schemes with a non-centered parameterization as recommended by the Stan Development Team (2022, User's Guide, Section 25.7) by diagonalizing the penalty matrices of the P-splines as described by Wood (2017, Section 5.4), but did not achieve an efficiency improvement. Other reparametrizations or the use of a Riemann metric (Girolami and Calderhead, 2011) might help to speed up the NUTS kernels, but we did not explore these options further in this case study.

The efficiency of an MCMC algorithm cannot be assessed based on the run time alone, but the quality of the samples needs to be taken into account as well. We use the effective sample size (ESS, Gelman et al., 2013) for this purpose. The ESS estimates the size an independent sample would need to have to contain the same amount of information as

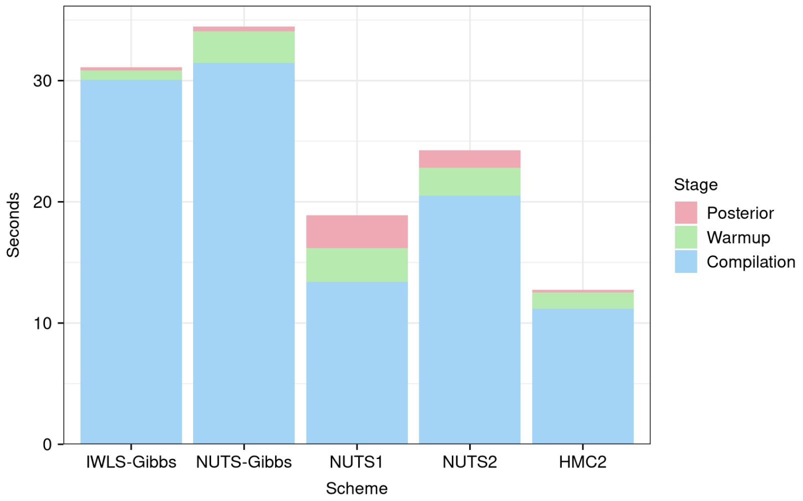


Figure 7: The compile and run time of the sampling schemes. The timings are obtained on an Intel i7-1185G7 CPU with 8 cores and 3 GHz for one MCMC chain with 1000 warmup and 1000 posterior iterations. The IWLS-Gibbs and NUTS-Gibbs schemes are most expensive to compile (because they combine two types of kernels), while the NUTS1 and NUTS2 schemes are most expensive to run (due to the high tree depth).

the correlated MCMC sample. An MCMC chain with a high autocorrelation generally has a low ESS. For the LIDAR model, the NUTS-Gibbs scheme has the highest ESS with a median of 318.67 per 1000 iterations, and the HMC2 scheme has the lowest ESS with a median of 25.56 (Table 2). The table also shows the ESS per second, which takes both the quality of the samples and the run time into account. By that measure, the two schemes involving a Gibbs kernel perform best, with a median of 869.05 for NUTS-Gibbs and 325.21 for IWLS-Gibbs.

Table 2: The bulk ESS and bulk ESS per second of the sampling schemes. 30 MCMC chains are generated per scheme, and the summary statistics are computed pooling all 22 parameters of the LIDAR model. The ESS per second is computed based on the run time of the posterior iterations, not taking the compilation and the warmup iterations into account. The NUTS-Gibbs scheme is the most efficient, both in terms of ESS and ESS per second.

		5%	25%	**Median**	75%	95%
Bulk ESS	**IWLS-Gibbs**	33.53	70.01	**91.17**	114.61	269.58
	NUTS-Gibbs	122.97	205.34	**318.67**	482.70	939.08
	NUTS1	7.80	46.30	**92.78**	347.65	945.62
	NUTS2	25.90	72.04	**140.14**	456.46	866.49
	HMC2	1.62	6.81	**25.56**	291.10	1249.13
Bulk ESS/s	**IWLS-Gibbs**	119.61	249.72	**325.21**	408.80	961.55
	NUTS-Gibbs	335.37	560.01	**869.05**	1316.41	2561.02
	NUTS1	2.88	17.11	**34.30**	128.52	349.57
	NUTS2	17.96	49.95	**97.16**	316.48	600.77
	HMC2	7.56	31.88	**119.55**	1361.79	5843.52

6 Discussion

In this article, we introduced the probabilistic programming framework Liesel, which allows the user to express Bayesian models as directed acyclic graphs and to build custom MCMC algorithms. With our software, established MCMC algorithms can be combined in new ways, and the user can implement problem-specific kernels and warmup schemes. Goose, Liesel's MCMC library, is independent of Liesel's graph-based model representation and can also be used with other JAX-compatible software, for example PyMC or user-defined log-posterior functions.

Models expressed in Liesel can be modified through a programmer-friendly API. A base model can be generated with RLiesel, a tool to configure semi-parametric regression models, and new ideas can be explored with little effort by modifying the base model. Using state-of-the-art technology like just-in-time compilation, automatic differentiation and cluster computing, which is possible with JAX, Liesel allows for a fast development and testing cycle in Python while maintaining good computational performance.

The development of Liesel will be continued in the coming years. Liesel uses many libraries that are under active development and whose API changes must be reflected in our software. We also plan to integrate new features and other enhancements of these libraries into Liesel. Based on JAX's experimental module for sparse linear algebra, for example, we will improve the performance of different models using efficient decomposition algorithms for matrices with band structures or more general sparsity patterns.

The next major update of the software, Liesel 0.2, is planned for fall 2022. It will feature an improved model representation, making manipulations and extensions of the model graph easier and safer. In the new version, the graph of the statistical variables in the model will be built on top of a graph of computational nodes. This approach will result in an interface that is more convenient in standard use cases and more "hackable" in advanced use cases. The new interface aims to be simple and transparent with a small number of classes that do not surprise the developer with any "magic" behavior.

Liesel will also be extended with more model components and new MCMC kernels. The new building blocks in the modeling library will facilitate the rapid development of new types of models, thus speeding up research. In particular, RLiesel will be extended with the functionality to build non-linear models that overcome the typical additive predictor structure of semi-parametric regression, or models that involve covariates that are themselves assigned a model specification such as measurement error models or more general structural equation models. These extensions will also serve as a demonstration of the functionality and flexibility that Liesel offers for the development of Bayesian (regression) models.

Liesel's technology stack facilitates the implementation of gradient-based methods. Having automatic differentiation available will allow us to use general optimization algorithms to implement variational inference methods. Stochastic gradient MCMC (SG-MCMC) is a relatively new class of Monte Carlo algorithms that scale well to large datasets. Compared to traditional MCMC, these algorithms reduce the computational costs by using subsamples of the original dataset, while maintaining a high accuracy of the parameter estimates. Tools like Stan, PyMC and NIMBLE that enabled the broad success of Bayesian methods in many application areas are still missing SG-MCMC methods, although the first steps have been made (e.g. in the R package `sgmcmc`). We plan to implement SG-MCMC kernels and non-traditional tuning methods for SG-MCMC in Liesel in the near future.

References

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: Composable transformations of Python+NumPy programs, version 0.3.17, August 2022. URL `https://github.com/google/jax`.

Andreas Brezger, Thomas Kneib, and Stefan Lang. BayesX: Analyzing Bayesian structured additive regression models. *Journal of Statistical Software*, 14(11):1–22, 2005. doi:10.18637/jss.v014.i11.

Paul-Christian Bürkner. brms: An R package for Bayesian multilevel models using Stan. *Journal of Statistical Software*, 80(1):1–28, 2017. doi:10.18637/jss.v080.i01.

Perry de Valpine, Daniel Turek, Christopher J. Paciorek, Clifford Anderson-Bergman, Duncan Temple Lang, and Rastislav Bodik. Programming with models: Writing statistical algorithms for general model structures with NIMBLE. *Journal of Computational and Graphical Statistics*, 26(2):403–413, 2017. doi:10.1080/10618600.2016.1172487.

Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matthew D. Hoffman, and Rif A. Saurous. TensorFlow distributions, November 2017.

Paul H. C. Eilers and Brian D. Marx. Flexible smoothing with B-splines and penalties. *Statistical Science*, 11(2): 89–102, May 1996. doi:10.1214/ss/1038425655.

Ludwig Fahrmeir, Thomas Kneib, Stefan Lang, and Brian Marx. *Regression: Models, Methods and Applications*. Springer, Heidelberg, 2013. doi:10.1007/978-3-642-34333-9.

Dani Gamerman. Sampling from the posterior distribution in generalized linear mixed models. *Statistics and Computing*, 7(1):57–68, March 1997. doi:10.1023/A:1018509429360.

A. Gelman, W. R. Gilks, and G. O. Roberts. Weak convergence and optimal scaling of random walk Metropolis algorithms. *The Annals of Applied Probability*, 7(1):110–120, February 1997. doi:10.1214/aoap/1034625254.

Andrew Gelman. Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian Analysis*, 1(3):515–534, September 2006. doi:10.1214/06-BA117A.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, New York, November 2013. doi:10.1201/b16018.

Mark Girolami and Ben Calderhead. Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214, March 2011. doi:10.1111/j.1467-9868.2010.00765.x.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi:10.1038/s41586-020-2649-2.

Matthew D. Hoffman and Andrew Gelman. The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(47):1593–1623, 2014.

Ulla Holst, Ola Hössjer, Claes Björklund, Pär Ragnarson, and Hans Edner. Locally weighted least squares kernel regression and statistical evaluation of LIDAR measurements. *Environmetrics*, 7(4):401–416, July 1996. doi:10.1002/(SICI)1099-095X(199607)7:4<401::AID-ENV221>3.0.CO;2-D.

Matthias Katzfuss and Joseph Guinness. A general framework for Vecchia approximations of Gaussian processes. *Statistical Science*, 36(1):124–141, February 2021. doi:10.1214/19-STS755.

Nadja Klein and Thomas Kneib. Simultaneous inference in structured additive conditional copula regression models: a unifying Bayesian approach. *Statistics and Computing*, 26(4):841–860, July 2016a. doi:10.1007/s11222-015-9573-6.

Nadja Klein and Thomas Kneib. Scale-dependent priors for variance parameters in structured additive distributional regression. *Bayesian Analysis*, 11(4):1071–1106, December 2016b. doi:10.1214/15-BA983.

Nadja Klein, Thomas Kneib, Stephan Klasen, and Stefan Lang. Bayesian structured additive distributional regression for multivariate responses. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 64(4):569–591, August 2015a. doi:10.1111/rssc.12090.

Nadja Klein, Thomas Kneib, and Stefan Lang. Bayesian generalized additive models for location, scale, and shape for zero-inflated and overdispersed count data. *Journal of the American Statistical Association*, 110(509):405–419, 2015b. doi:10.1080/01621459.2014.912955.

Stefan Lang and Andreas Brezger. Bayesian P-splines. *Journal of Computational and Graphical Statistics*, 13(1): 183–212, 2004. doi:10.1198/1061860043010.

Junpeng Lao and Rémi Louf. BlackJAX: A sampling library for JAX, version 0.8.3, August 2022. URL `https://github.com/blackjax-devs/blackjax`.

Junpeng Lao, Christopher Suter, Ian Langmore, Cyril Chimisov, Ashish Saxena, Pavel Sountsov, Dave Moore, Rif A. Saurous, Matthew D. Hoffman, and Joshua V. Dillon. tfp.mcmc: Modern Markov chain Monte Carlo tools built for modern hardware, February 2020.

David J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, September 2003.

Giampiero Marra and Rosalba Radice. GJRM: Generalised joint regression modelling, version 0.2-6, April 2022. URL `https://CRAN.R-project.org/package=GJRM`.

Patrick Michaelis, Nadja Klein, and Thomas Kneib. Bayesian multivariate distributional regression with skewed responses and skewed random effects. *Journal of Computational and Graphical Statistics*, 27(3):602–611, 2018. doi:10.1080/10618600.2017.1395343.

Radford M. Neal. MCMC using Hamiltonian dynamics. In Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng, editors, *Handbook of Markov Chain Monte Carlo*, pages 139–188. Chapman and Hall/CRC, New York, 2011. doi:10.1201/b10905-10.

Yurii Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120(1):221–259, August 2009. doi:10.1007/s10107-007-0149-x.

Du Phan, Neeraj Pradhan, and Martin Jankowiak. Composable effects for flexible and accelerated probabilistic programming in NumPyro, December 2019.

Martyn Plummer. JAGS: Just another Gibbs sampler, version 4.3.1, April 2022. URL `https://sourceforge.net/projects/mcmc-jags`.

R. A. Rigby and D. M. Stasinopoulos. Generalized additive models for location, scale and shape. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 54(3):507–554, June 2005. doi:10.1111/j.1467-9876.2005.00510.x.

Havard Rue and Leonhard Held. *Gaussian Markov Random Fields: Theory and Applications*. Chapman and Hall/CRC, New York, February 2005. doi:10.1201/9780203492024.

John Salvatier, Thomas V. Wiecki, and Christopher Fonnesbeck. Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*, 2:e55, 2016. doi:10.7717/peerj-cs.55.

Carlos Scheidegger, Charles Teague, Christophe Dervieux, J. J. Allaire, and Yihui Xie. Quarto: Open-source scientific and technical publishing system built on pandoc, version 1.2.127, September 2022. URL `https://github.com/quarto-dev/quarto-cli`.

Stan Development Team. Stan modeling language users guide and reference manual, version 2.30.0, July 2022. URL `https://mc-stan.org`.

Mikis D. Stasinopoulos, Robert A. Rigby, Gillian Z. Heller, Vlasios Voudouris, and Fernanda De Bastiani. *Flexible Regression and Smoothing: Using GAMLSS in R*. Chapman and Hall/CRC, New York, May 2017. doi:10.1201/b21973.

Nikolaus Umlauf, Nadja Klein, Thorsten Simon, and Achim Zeileis. bamlss: A Lego toolbox for flexible Bayesian regression (and beyond). *Journal of Statistical Software*, 100(4):1–53, 2021. doi:10.18637/jss.v100.i04.

Kevin Ushey, J. J. Allaire, and Yuan Tang. reticulate: Interface to 'Python', version 1.26, August 2022. URL `https://CRAN.R-project.org/package=reticulate`.

B. P. Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, August 1962. doi:10.1080/00401706.1962.10490022.

Simon Wood. mgcv: Mixed GAM computation vehicle with automatic smoothness estimation, version 1.8-40, March 2022. URL `https://CRAN.R-project.org/package=mgcv`.

Simon N. Wood. Stable and efficient multiple smoothing parameter estimation for generalized additive models. *Journal of the American Statistical Association*, 99(467):673–686, 2004. doi:10.1198/016214504000000980.

Simon N. Wood. *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC, New York, second edition, May 2017. doi:10.1201/9781315370279.

Appendix B

Modelling Intra-Annual Tree Stem Growth with a Distributional Regression Approach for Gaussian Process Responses

(With Supplement)

Journal of the Royal Statistical Society Series C: Applied Statistics, 2023, **72**, 414–433
https://doi.org/10.1093/jrsssc/qlad015
Advance access publication 22 March 2023
Original Article

ROYAL STATISTICAL SOCIETY
DATA | EVIDENCE | DECISIONS
Journal of the Statistics Society
Series C
Applied Statistics

Modelling intra-annual tree stem growth with a distributional regression approach for Gaussian process responses

Hannes Riebl[1], Nadja Klein[2] and Thomas Kneib[1]

[1]Chair of Statistics, Georg-August-Universität Göttingen, Göttingen, Germany
[2]Chair of Statistics and Data Science, Humboldt-Universität zu Berlin, Berlin, Germany

Address for correspondence: Hannes Riebl, Professur für Statistik, Georg-August-Universität Göttingen, Humboldtallee 3, 37073 Göttingen, Germany. Email: hriebl@uni-goettingen.de

Abstract

High-resolution circumference dendrometers measure the irreversible growth and the reversible shrinking and swelling due to the water content of a tree stem. We propose a novel statistical method to decompose these measurements into a permanent and a temporary component, while explaining differences between the trees and years by covariates. Our model embeds Gaussian processes with parametric mean and covariance functions as response structures in a distributional regression framework with structured additive predictors. We discuss different mean and covariance functions, connections with other model classes, Markov chain Monte Carlo inference, and the efficiency of our sampling scheme.

Keywords: generalised additive model for location, scale, and shape, growth curve model, Markov chain Monte Carlo simulation, Matérn covariance function, spatio-temporal regression, structured additive predictor

1 Introduction

Tree growth, and the growth of tree stems in particular, is a process that is of strong ecological and economic interest. Together with the height growth, the growth in the stem girth drives timber production, and at the same time, plays a key role in the global carbon cycle (Mencuccini et al., 2017). Unfortunately, it is difficult to measure the formation of new wood and bark cells in the cambium resulting in permanent stem growth, and while electronic dendrometers can record the variation of the stem circumference on small time scales of a few minutes (Klepper et al., 1971), these measurements also capture the reversible shrinking and swelling of the stem due to changes in its water content. Researchers have used additional measurement equipment such as sap flow sensors and controlled irrigation experiments to gain a better understanding of the permanent and temporary components of tree stem growth (Mencuccini et al., 2017), but these experiments are either expensive or not feasible under open field conditions.

We describe a novel statistical method for the analysis of high-resolution dendrometer measurements that does not require additional data about other tree-physiological processes. The method permits us to decompose the dendrometer measurements into a permanent and a temporary component through stochastic assumptions and explanatory variables. Our dataset contains 85 deciduous trees from Germany and the growing seasons 2012 and 2013. Figure 1 shows a subsample of the recorded growth curves between April 1 and September 30, each of which is assumed to be a realisation of a Gaussian process (GP). The GPs are conditionally independent from each other given a set of explanatory variables. We observe that the coloured ash grows primarily between mid-April and mid-July, while the coloured beech grows later and more during the

Received: March 4, 2022. **Revised:** December 5, 2022. **Accepted:** January 20, 2023

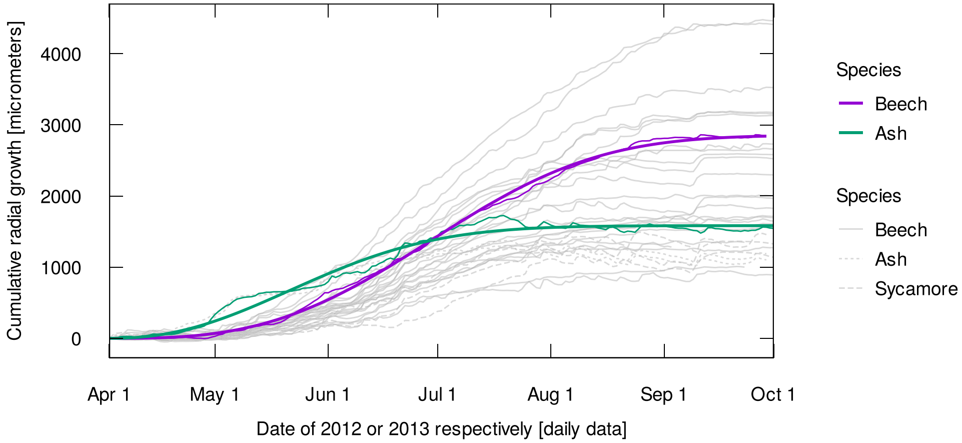


Figure 1. The cumulative radial growth of a subsample of the trees from our dataset over the course of one growing season. The coloured lines represent two exemplary trees, one beech and one ash, while the grey lines illustrate the diversity of the growth patterns in the dataset. The reversible shrinking and swelling is more pronounced for the coloured ash than the beech. The bold lines show the estimated irreversible growth.

vegetation period. These different patterns are captured in the estimated sigmoid mean functions of the GPs, which represent the irreversible growth of the tree stems. On the other hand, the temporary shrinking and swelling, which is more pronounced in the ash than the beech, is described by the within-season covariance functions. The tree species is one factor that we condition the GPs on. Other possible explanatory variables include the diameter at breast height (DBH) and the geographical location of the trees.

The way we use GPs is different from the standard approaches in machine learning or spatial statistics. Models from those fields typically assume a single latent GP. For instance, in many supervised learning problems in machine learning, GPs are used as priors over the hypothesis space of possible functions from the input to the output space (Rasmussen & Williams, 2006). GPs also play a key role in spatial statistics, where they are used to capture the spatial correlation of the data, and to avoid invalid, underestimated confidence intervals (Cressie, 1993). In contrast, we assume multiple *observed*, *conditionally independent* GPs as response structures in a regression model.

The fact that we link multiple properties of the mean and covariance functions of the GPs to explanatory variables puts our model in the domain of the so-called distributional regression models, also known as generalised additive models for location, scale, and shape (GAMLSS, Rigby & Stasinopoulos, 2005). Usually, this model class admits multiple structured additive predictors for different parameters of the conditional distribution of a response variable. Standard distributional regression models use univariate or low-dimensional multivariate response variables. Klein and Kneib (2016b) discuss distributional regression models with copula-based bivariate response distributions in a Bayesian setting, and Filippou et al. (2017) propose a trivariate probit model, which they estimate with a frequentist penalised likelihood method. A number of bivariate and trivariate response distributions is also available in the vector generalised additive model framework (Yee, 2015). Following this line of thought, we show that the distributional regression approach also works for more general, continuous response structures such as GPs.

The distributional regression literature offers different approaches to statistical inference. We build on the work of Klein et al. (2015), who propose a general Markov chain Monte Carlo (MCMC) algorithm for Bayesian inference in distributional regression models. To assess the posterior distribution of the model parameters, they use a Metropolis-within-Gibbs sampler with iterative weighted least squares (IWLS) proposals (Gamerman, 1997). The algorithm was implemented by Umlauf et al. (2018) in the R package `bamlss`. However, `bamlss` blocks the model parameters in a way that performs very poorly with GPs as response structures. We describe a more efficient way of blocking the model parameters in Section 3, and discuss the problem in more detail in the simulation study in Section 4.

All computations for this paper were performed using the R software environment for statistical computing (R Core Team, 2020). The relevant code is available under the MIT open source license as a supplement and on GitHub (https://github.com/hriebl/gp-responses).

The remainder of this paper is structured as follows: in the next section, we give the precise definition of our model, as well as some examples of mean and covariance functions. Structured additive predictors and the connection with a number of related statistical model classes (mixed models, functional data, etc.) are also discussed. In Section 3, we provide the specifics of the MCMC algorithm for the posterior estimation. The simulation study with three different scenarios is presented in Section 4, while Section 5 addresses the application to intra-annual tree stem growth in full detail. Finally, Section 6 concludes and discusses possible extensions and further applications of the model.

2 Model specification

2.1 Gaussian processes as response structures

We consider GPs $\{Y_i(\boldsymbol{t}); \boldsymbol{t} \in T\}$ as response structures in structured additive distributional regression, where the observation index i runs from 1 to N and the index set T is a metric space that can represent time, space, or space-time. The GPs are assumed to be conditionally independent given the covariate vectors $\boldsymbol{x}_i$,

$$\{Y_i(\boldsymbol{t}); \boldsymbol{t} \in T\} \mid \boldsymbol{x}_i \overset{\text{ind.}}{\sim} \mathcal{GP}(m_{\boldsymbol{x}}(\boldsymbol{t}; \boldsymbol{x}_i), c_{\boldsymbol{x}}(\boldsymbol{t}, \boldsymbol{t}'; \boldsymbol{x}_i)), \tag{1}$$

where $\boldsymbol{t}, \boldsymbol{t}' \in T$. As a specific feature of distributional regression, the mean function $m_{\boldsymbol{x}}$ and the covariance function $c_{\boldsymbol{x}}$ both depend on the covariates $\boldsymbol{x}_i$, which differ between the observations 1 to N but are constant within the index set T. An extension of the model to time or space-varying covariates is given below.

More precisely, the mean and the covariance function are linked to the covariates $\boldsymbol{x}_i$ via their respective parameter vectors $\boldsymbol{\theta}^m$ and $\boldsymbol{\theta}^c$,

$$m_{\boldsymbol{x}}(\boldsymbol{t}; \boldsymbol{x}_i) = m(\boldsymbol{t}; \boldsymbol{\theta}^m(\boldsymbol{x}_i)) \quad \text{and} \quad c_{\boldsymbol{x}}(\boldsymbol{t}, \boldsymbol{t}'; \boldsymbol{x}_i) = c(\boldsymbol{t}, \boldsymbol{t}'; \boldsymbol{\theta}^c(\boldsymbol{x}_i)).$$

For better readability, we use a subscript i as an abbreviation for the dependence of a variable on the covariates $\boldsymbol{x}_i$. Let $\boldsymbol{\theta}_i = [(\boldsymbol{\theta}_i^m)^\top, (\boldsymbol{\theta}_i^c)^\top]^\top = [\boldsymbol{\theta}^m(\boldsymbol{x}_i)^\top, \boldsymbol{\theta}^c(\boldsymbol{x}_i)^\top]^\top$ be the vector of all parameters of the GPs and K its dimension. In the terminology of distributional regression, $\boldsymbol{\theta}_i$ is the vector of the K distributional parameters. Each parameter θ_{ki} is linked to a structured additive predictor via a strictly monotonic link function (Section 2.3).

One important extension of Model (1) deals with the inclusion of time or space-varying covariates $z_i : T \to S$, which change within the index set T depending on the coordinates $\boldsymbol{t}$ (as opposed to the previously discussed covariates $\boldsymbol{x}_i$). The covariates z_i may be understood as a mapping of the coordinates $\boldsymbol{t}$ into a different, more abstract metric space S, which no longer represents only space or time. The mean and the covariance function now depend on the coordinates $\boldsymbol{t}$ via this mapping,

$$\{Y_i(\boldsymbol{t})\} \mid z_i, \boldsymbol{x}_i \overset{\text{ind.}}{\sim} \mathcal{GP}(m_{\boldsymbol{x}}(z_i(\boldsymbol{t}); \boldsymbol{x}_i), c_{\boldsymbol{x}}(z_i(\boldsymbol{t}), z_i(\boldsymbol{t}'); \boldsymbol{x}_i)). \tag{2}$$

The simulation study in Section 4 includes a proof of concept for time-varying covariates.

In practice, each GP $\{Y_i(\boldsymbol{t})\}$ can only be observed at a finite number of points $\boldsymbol{t}_j \in T$, for $j = 1, \ldots, n_i$. The collection of random variables at these points has a multivariate normal distribution,

$$[Y_i(\boldsymbol{t}_1), \ \ldots, \ Y_i(\boldsymbol{t}_{n_i})]^\top \mid z_i(\boldsymbol{t}_1), \ \ldots, z_i(\boldsymbol{t}_{n_i}), \boldsymbol{x}_i \overset{\text{ind.}}{\sim} \mathcal{N}_{n_i}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{3}$$

where the elements of the mean vector $\boldsymbol{\mu}_i$ and the covariance matrix $\boldsymbol{\Sigma}_i$ are the evaluations of the mean function m and the covariance function c at the observed points,

$$\boldsymbol{\mu}_i = [\mu_{i,j}] = m(z_i(\boldsymbol{t}_j); \boldsymbol{\theta}_i^m) \quad \text{and} \quad \boldsymbol{\Sigma}_i = [\sigma_{i,j,j'}] = c(z_i(\boldsymbol{t}_j), z_i(\boldsymbol{t}_{j'}); \boldsymbol{\theta}_i^c) \tag{4}$$

for $j, j' = 1, \ldots, n_i$. The number of observed values does not necessarily need to be the same for all GPs, i.e., potentially $n_i \neq n_{i'}$ for some $i, i' \in \{1, \ldots, N\}$. Furthermore, our construction leaves the basic structure of the covariance function untouched, such that even after including dependence on covariates, it is ensured that the covariance function and therefore also the resulting covariance matrices $\boldsymbol{\Sigma}_i$ are positive definite.

2.2 Examples of mean and covariance functions

The most important condition for a GP to be valid is that the covariance function needs to be positive semi-definite. For Model (1), this means that $\sum_{t,t' \in U} a_t c(\boldsymbol{t}, \boldsymbol{t}') a_{t'}$ needs to be non-negative for all $U \subset T$ and the weights $a_t \in \mathbb{R}$ of each linear combination $\sum_{t \in U} a_t Y_i(\boldsymbol{t})$ (Adler, 1990, Section 1.1). When considering Model (2) with time or space-varying covariates, the positive semi-definiteness of the covariance function needs to hold on the index set S instead of T. The requirements for the mean function of a GP are less restrictive: Essentially any function $m: T \to \mathbb{R}$ or $m: S \to \mathbb{R}$ is a valid mean function of a GP.

The mean and the covariance function of a GP determine its continuity and differentiability properties. For example, a GP is mean-square continuous if and only if its mean and its covariance function are continuous. Mean-square continuity, however, does not imply sample continuity (Rasmussen & Williams, 2006, Section 4.1.1). The concept of sample continuity is discussed in a rigorous and abstract fashion in Adler (1990). For most applied modelling problems, a continuous mean function will be a reasonable assumption, but the same is not necessarily true for the covariance function: One reason for a discontinuous covariance function might be an idiosyncratic error term for each measurement. This idiosyncratic error term ‘conceals’ the GP of interest and is known as the ‘nugget effect’ in spatial statistics. It is usually modelled as an additive i.i.d. GP, rendering the resulting sum of two GPs discontinuous, even in the mean-square sense.

Note that we omit the observation index i in the following discussion of the mean and covariance functions for the sake of simplicity.

2.2.1 Mean functions

Linear Mean Function. The linear mean function is defined as the dot product of the covariates $z_i(\boldsymbol{t})$ and the parameters $\boldsymbol{\theta}^m$,

$$m^l(z(\boldsymbol{t}); \boldsymbol{\theta}^m) = z(\boldsymbol{t})^\top \boldsymbol{\theta}^m. \tag{5}$$

The linear mean function is mathematically convenient and provides considerable flexibility for statistical modelling. Polynomials or B-splines can be used, among others, to represent large classes of functions as linear combinations of basis functions. To do so, we choose $z(\boldsymbol{t}) = [b_1(\boldsymbol{t}), \ldots, b_M(\boldsymbol{t})]^\top$, where $b_1, \ldots, b_M$ are the aforementioned basis functions and M is the number of basis functions or, equivalently, the number of distributional parameters of the mean function. This approach gives rise to *non-parametric* mean functions with very flexible shapes.

Weibull Growth Curve. The flexibility of a linear mean function with polynomials or B-splines comes at a cost: it requires a large number of parameters without an obvious interpretation. In many applications, however, interpretable distributional parameters are desirable. If prior knowledge about the shape of the response processes is available, a *parametric* mean function might be a more natural choice. For example, the intra-annual tree growth curves in Section 5 have a sigmoid shape. Any sigmoid function such as the logistic function or the hyperbolic tangent could serve as a mean function in this case, but we follow Metz et al. (2020) and use the Weibull growth curve for this purpose. The Weibull growth curve is a scaled version of the cumulative distribution function of the Weibull distribution,

$$m^w(z(\boldsymbol{t}) = t; \boldsymbol{\theta}^m = [l, a, b]^\top) = l \times \left[1 - \exp\left(-\left(\frac{t}{b}\right)^a\right)\right], \tag{6}$$

where $t \geq 0$ is the point in time since the start of the growing season, and the parameters are the limit $l > 0$, the shape $a > 0$, and the scale $b > 0$. The scale parameter describes for how long a tree continues to grow during the summer, while the shape parameter represents the steepness of the growth curve. As all parameters of the Weibull growth curve need to be positive, we use a log-link for these parameters.

2.2.2 *Covariance functions*

Throughout this paper, we use the Matérn covariance function and relate the standard deviation σ and the range ϕ to covariates. Without any time or space-varying covariates, it has the form

$$c^m(z(\boldsymbol{t}) = \boldsymbol{t}, z(\boldsymbol{t}') = \boldsymbol{t}'; \boldsymbol{\theta}^c = [\sigma, \phi]^\top) = \sigma^2 \times \rho\left(\frac{d(\boldsymbol{t}, \boldsymbol{t}')}{\phi}; \nu\right), \tag{7}$$

where ρ is the Matérn correlation function with the smoothness parameter ν, and $d(\boldsymbol{t}, \boldsymbol{t}')$ is a distance function. For $\nu = 1/2$, the Matérn correlation function simplifies to the exponential correlation function, and for $\nu \to \infty$, it converges to the squared exponential or Gaussian correlation function. GPs with the Matérn covariance function are $\lfloor \nu \rfloor$ times mean-square differentiable and even have differentiable sample paths (Rasmussen & Williams, 2006, Section 4.2.1; Paciorek, 2003, Section 2.5.4). Different values for ν can be used for different models, but we do not treat it as a distributional parameter in this paper. For the standard deviation and the range, we use a log-link, as these parameters need to be positive.

The Matérn covariance function (or any other covariance function) can be modified to include an additive i.i.d. measurement error, giving rise to the covariance function

$$c^*(\boldsymbol{t}, \boldsymbol{t}'; \boldsymbol{\theta}^c = [\sigma, \phi, \delta]^\top) = c^m(\boldsymbol{t}, \boldsymbol{t}'; \boldsymbol{\theta}^c = [\sigma, \phi]^\top) + \delta^2 \times \mathbb{I}(d(\boldsymbol{t}, \boldsymbol{t}') = 0),$$

where δ is the standard deviation of the idiosyncratic error, and $\mathbb{I}$ is the indicator function. In the distributional regression framework, δ can be interpreted as an additional distributional parameter of the covariance function. The estimation procedure discussed in Section 3 can be applied to δ in the same way as to any other distributional parameter.

It is important to note that the validity of a covariance function depends on the metric space it is defined on, i.e., on the distance function $d(\boldsymbol{t}, \boldsymbol{t}')$. While the Matérn covariance function is valid on the Euclidean space of any dimension, the situation on the sphere with the great circle distance is more complicated: Gneiting (2013) investigates the validity of different commonly used covariance functions on the one- to three-dimensional sphere and finds that, in this case, the Matérn covariance function is only valid for $0 < \nu \leq 1/2$.

There are no restrictions on the families of covariance functions that can be used in our model framework, and the number of covariance parameters can be greater than for the Matérn covariance function. Examples of alternative covariance functions include the power exponential, the rational quadratic, and the spherical covariance functions (Rasmussen & Williams, 2006, Section 4.2). Some index sets with special interpretations might require more elaborate covariance functions such as non-stationary or non-separable space-time covariance functions (Gneiting, 2002).

2.3 Structured additive predictors and effect priors

In the structured additive regression framework (Fahrmeir et al., 2004; Wood, 2017), each predictor η_{ki} can be expressed as a sum of L_k smooth terms f_{kl},

$$\eta_{ki} = \sum_{l=1}^{L_k} f_{kl}(\boldsymbol{x}_i; \boldsymbol{\beta}_{kl}),$$

where each function f_{kl} is expanded from a basis representation as $f_{kl} = \sum_{d=1}^{D_{kl}} B_{kld}(\boldsymbol{x}_i)\beta_{kld}$, and $\boldsymbol{\beta}_{kl}$ are the regression coefficients to be estimated. The predictor η_{ki} can attain any real value and needs to be mapped to the (possibly constrained) parameter space of the distributional parameter θ_{ki} with a strictly monotonic link function h_k, i.e., $h_k(\theta_{ki}) = \eta_{ki}$ or $\theta_{ki} = h_k^{-1}(\eta_{ki})$.

A smooth term usually depends on one or two elements of the covariate vector $\boldsymbol{x}_i$, but it can also be of a higher dimension, e.g., in the case of simple linear covariate effects. The interpretation of a smooth term depends on the choice of the basis functions and the prior of the regression coefficients. In many cases, a (proper or improper) normal prior is assumed for the regression coefficients,

$$p(\boldsymbol{\beta}_{kl} \mid \tau_{kl}) \propto \exp\left(-\frac{1}{2\tau_{kl}^2}\boldsymbol{\beta}_{kl}^\top \mathbf{P}_{kl}\boldsymbol{\beta}_{kl}\right),$$

where τ_{kl} is a hyperparameter that controls the smoothness of the covariate effect, and $\mathbf{P}_{kl}$ is a penalty matrix. The hierarchical prior of the parameters of smooth term l in predictor k is given by $p(\boldsymbol{\beta}_{kl}, \tau_{kl}) = p(\boldsymbol{\beta}_{kl} \mid \tau_{kl}) \times p(\tau_{kl})$, where $p(\tau_{kl})$ is often an inverse gamma distribution with fixed hyperparameters $a = b = 0.0001$ or some other small value. In more complex scenarios, we might also have covariate effects that depend on more than just one single, scalar hyperparameter (e.g., to achieve adaptive smoothness) or another hierarchical prior layer that connects the hyperparameters (e.g., to control the overall model complexity, Klein & Kneib, 2016a), but we will stick to the simple case in this article. Notationally, we will assume a vector of hyperparameters $\boldsymbol{\tau}_{kl}$ for better generality, including the scalar case as a special case.

The smooth terms in a structured additive predictor can represent a broad range of covariate effects (linear, random, non-linear, spatial, etc.). A simple non-linear effect of a single covariate can be constructed using a polynomial basis. Cubic or B-splines provide a numerically more stable and flexible alternative to standard polynomials. In Section 5, we use a kriging smooth to model a spatial effect, which we can also represent as a linear combination of basis function evaluations. See Fahrmeir et al. (2013) for more details on smooth terms and structured additive predictors.

2.4 Related model classes

Mixed models with within-group correlation structures

If the index set T reduces to a finite set, the GPs become finite collections of dependent observations. For grouped data like this, where the groups could represent longitudinal observations on one person, mixed models are the standard tool. Distributional regression models with GP responses are closely related to the marginal distribution implied by mixed models. For example, the random intercepts model corresponds to

$$[Y_{i,1}, \ldots, Y_{i,n_i}]^\top \mid \boldsymbol{x}_i \overset{\text{ind.}}{\sim} \mathcal{N}_{n_i}(\boldsymbol{\mu}_i, \xi^2\mathbf{E}_{n_i} + \sigma^2\mathbf{I}_{n_i}), \tag{8}$$

where $[Y_{i,1}, \ldots, Y_{i,n_i}]^\top$ is the vector of the n_i measurements on group i, $\mathbf{E}_i$ is a $(n_i \times n_i)$-dimensional matrix of ones, $\mathbf{I}_{n_i}$ is the n_i-dimensional identity matrix, and ξ^2 represents the random effect variance. In this case, there are two (co-)variance parameters ξ^2 and σ^2. Adding random slopes to the model would increase the number of covariance parameters.

Some mixed model implementations (such as the `nlme` package for R, Pinheiro et al., 2020) allow for within-group correlation structures of the residuals, which means that the identity matrix $\mathbf{I}_{n_i}$ in Equation (8) can be replaced with a more complex correlation matrix. This model extension is useful for temporally or spatially correlated data. As in our model, the correlation matrix is defined in terms of a parametric correlation function such as the Matérn correlation function, but to the best of our knowledge, its parameters (and the further covariance parameters of mixed models) are usually not linked to group-specific covariates. We are also not aware of any software package that supports this model structure out of the box. Moreover, the `nlme` package does not support structured additive predictors, but other mixed model software packages like `gamm4` (Wood & Scheipl, 2020) do.

Functional data

Our model also has a close conceptual relation with GP regression models for functional data (Shi & Choi, 2011). For example, consider the functional response model

$$Y_i(\boldsymbol{t}) \overset{\text{ind.}}{\sim} \mathcal{N}(\mu_i(\boldsymbol{t}) + \omega_i(\boldsymbol{x}_i(\boldsymbol{t})), \sigma^2),$$

where $\boldsymbol{x}_i(\boldsymbol{t})$ are functional covariates, while a GP prior is assumed for $\omega_i(\boldsymbol{x}_i(\boldsymbol{t}))$, which defines the covariance structure of $Y_i(\boldsymbol{t})$. As a major difference, Shi and Choi (2011) do not allow for the inclusion of group-specific covariate effects on the (hyper-)parameters of the covariance functions of $\omega_i(\boldsymbol{x}_i(\boldsymbol{t}))$ and $Y_i(\boldsymbol{t})$.

Another aspect where the work of Shi and Choi (2011) is similar to that of Greven and Scheipl (2017) or Scheipl et al. (2015) but different from ours is the way the mean functions $\mu_i(\boldsymbol{t})$ are modelled: while our motivation is to use mean functions whose parameters serve as distributional parameters and are linked to covariates, Greven and Scheipl (2017) focus on (non-parametric) linear representations of the mean functions via suitable basis expansions. Given this difference, our model can be considered more realistic and more stable, at least in situations where prior knowledge on the shape of the mean functions such as the sigmoid shape of the Weibull growth curves in Section 5 is available. On the other hand, parametric mean functions might be too restrictive for some applications. Finally, our index set T can represent different (potentially non-Euclidean) metric spaces, while the functional data literature is typically concerned with time-indexed data.

3 Posterior estimation

We stack all regression coefficients in a vector $\boldsymbol{\beta}$, all hyperparameters in a vector $\boldsymbol{\tau}$, and all covariates in a matrix $\mathbf{X}$ consisting of the N rows $\boldsymbol{x}_i$. The unnormalised log-posterior is then given by

$$\log \pi(\boldsymbol{\beta}, \boldsymbol{\tau} \mid \boldsymbol{y}_1, \ldots, \boldsymbol{y}_N, \mathbf{X}) \propto \sum_{i=1}^{N} \log p_{\mathcal{N}}(\boldsymbol{y}_i \mid \boldsymbol{x}_i, \boldsymbol{\beta}) + \sum_{k=1}^{K} \sum_{l=1}^{L_k} \log p_{kl}(\boldsymbol{\beta}_{kl}, \boldsymbol{\tau}_{kl}).$$

The first term on the right-hand side is the log-likelihood of the regression coefficients $\boldsymbol{\beta}$, where the observed values for the ith GP are denoted by $\boldsymbol{y}_i$ and the density of the multivariate normal distribution by $p_{\mathcal{N}}$. The second term on the right-hand side is the joint log-prior of all parameters $\boldsymbol{\beta}$ and $\boldsymbol{\tau}$, where p_{kl} is the density of the prior distribution of the regression coefficients and hyperparameters of smooth term l in predictor k.

We perform fully Bayesian inference with an adjusted version of the generic MCMC sampler of Umlauf et al. (2018), which uses inverse gamma priors and Gibbs updates for each scalar element of the hyperparameters $\boldsymbol{\tau}$, and Metropolis–Hastings updates with locally adaptive IWLS proposals (Gamerman, 1997) for the regression coefficients $\boldsymbol{\beta}$. As the IWLS proposals involve the observed or expected Fisher information matrix, the regression coefficients are sampled in blocks for numerical stability and efficiency. Typically, one block consists of the regression coefficients of one smooth term, and the blocks are sampled in a nested loop over the distributional parameters first and the smooth terms second. As discussed in the next section, we sample the parameters of certain smooth terms in one joint block, which can reduce the autocorrelation of the MCMC chains substantially.

The required full conditionals for the Gibbs updates of the hyperparameters are independent of the specific response structure of a distributional regression model and are given e.g., in Umlauf et al. (2018).

3.1 Model-specific scores and Fisher information

For a general distributional regression model, the score and the Fisher information of the regression coefficients $\boldsymbol{\beta}_{kl}$ are given by

$$s(\boldsymbol{\beta}_{kl}) = \sum_{i=1}^{N} s(\theta_{ki}) \frac{\partial \theta_{ki}}{\partial \boldsymbol{\beta}_{kl}} \quad \text{and} \quad \mathcal{I}(\boldsymbol{\beta}_{kl}) = \sum_{i=1}^{N} \mathcal{I}(\theta_{ki}) \frac{\partial \theta_{ki}}{\partial \boldsymbol{\beta}_{kl}} \left(\frac{\partial \theta_{ki}}{\partial \boldsymbol{\beta}_{kl}} \right)^{\top},$$

where the distributional parameters θ_{ki} are functions of the regression coefficients $\boldsymbol{\beta}_{kl}$ composed of smooth terms, structured additive predictors, and inverse link functions. The derivatives $\partial\theta_{ki}/\partial\boldsymbol{\beta}_{kl}$ are usually easy to compute. For this reason, we only give $s(\theta_{ki})$ and $\mathcal{I}(\theta_{ki})$, the score and the Fisher information of the distributional parameters with respect to the response distribution, for the particular case of GP responses in the distributional regression framework.

Using the definitions of the mean vector $\boldsymbol{\mu}_i$ and the covariance matrix $\boldsymbol{\Sigma}_i$ from Equation (4), the unnormalised log-likelihood contribution of the ith GP is

$$\log p_{\mathcal{N}}(\boldsymbol{y}_i \mid \boldsymbol{x}_i, \boldsymbol{\beta}) \propto -\frac{1}{2}\left(\log|\boldsymbol{\Sigma}_i| + (\boldsymbol{y}_i - \boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{y}_i - \boldsymbol{\mu}_i)\right).$$

For better readability, we omit the observation index i in the following formulas. Let θ_k be a distributional parameter of the *mean function*, then the score and the Fisher information of θ_k are

$$s(\theta_k) = \left(\frac{\partial\boldsymbol{\mu}}{\partial\theta_k}\right)^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{y} - \boldsymbol{\mu}) \quad \text{and} \quad \mathcal{I}(\theta_k) = \left(\frac{\partial\boldsymbol{\mu}}{\partial\theta_k}\right)^\top \boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\mu}}{\partial\theta_k}.$$

Now, let θ_k be a distributional parameter of the *covariance function*, then the score of θ_k is

$$s(\theta_k) = -\frac{1}{2}\left[\operatorname{tr}\left(\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\Sigma}}{\partial\theta_k}\right) - (\boldsymbol{y} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\Sigma}}{\partial\theta_k}\boldsymbol{\Sigma}^{-1}(\boldsymbol{y} - \boldsymbol{\mu})\right]$$

and the corresponding Fisher information is

$$\mathcal{I}(\theta_k) = \frac{1}{2}\operatorname{tr}\left(\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\Sigma}}{\partial\theta_k}\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\Sigma}}{\partial\theta_k}\right).$$

The derivatives $\partial\boldsymbol{\mu}/\partial\theta_k$ and $\partial\boldsymbol{\Sigma}/\partial\theta_k$ depend on the specific mean and covariance function of the GPs. Typically, one of the distributional parameters of the covariance function will be the standard deviation σ (or the variance σ^2, depending on the parameterisation). In this case, the covariance matrix is given by $\boldsymbol{\Sigma} = \sigma^2\mathbf{R}$, where $\mathbf{R}$ is the correlation matrix, and the score and the Fisher information of σ simplify to

$$s(\sigma) = -\frac{1}{\sigma}\left(n - (\boldsymbol{y} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{y} - \boldsymbol{\mu})\right) \quad \text{and} \quad \mathcal{I}(\sigma) = \frac{2n}{\sigma^2}.$$

3.2 Sampling the covariance parameters in one block

To sample the regression coefficients of the smooth terms l and $\tilde{l}$ in the predictors k and $\tilde{k}$ in one block, their joint Fisher information is required,

$$\mathcal{I}\left(\begin{bmatrix}\boldsymbol{\beta}_{kl} \\ \boldsymbol{\beta}_{\tilde{k}\tilde{l}}\end{bmatrix}\right) = \begin{bmatrix}\mathcal{I}(\boldsymbol{\beta}_{kl}) & \operatorname{Cov}(s(\boldsymbol{\beta}_{kl}), s(\boldsymbol{\beta}_{\tilde{k}\tilde{l}})) \\ \operatorname{Cov}(s(\boldsymbol{\beta}_{\tilde{k}\tilde{l}}), s(\boldsymbol{\beta}_{kl})) & \mathcal{I}(\boldsymbol{\beta}_{\tilde{k}\tilde{l}})\end{bmatrix},$$

where

$$\operatorname{Cov}(s(\boldsymbol{\beta}_{kl}), s(\boldsymbol{\beta}_{\tilde{k}\tilde{l}})) = \sum_{i=1}^{N}\operatorname{Cov}(s(\theta_{ki}), s(\theta_{\tilde{k}i}))\frac{\partial\theta_{ki}}{\partial\boldsymbol{\beta}_{kl}}\left(\frac{\partial\theta_{\tilde{k}i}}{\boldsymbol{\beta}_{\tilde{k}\tilde{l}}}\right)^\top.$$

Specifically, we want to improve the sampling performance for the covariance parameters, so we need the covariance of the score of the standard deviation and the range, which is given by

$$\operatorname{Cov}(s(\sigma), s(\phi)) = \frac{1}{\sigma}\operatorname{tr}\left[\mathbf{R}^{-1}\frac{\partial\mathbf{R}}{\partial\phi}\right].$$

4 Simulation study

We designed a simulation study with three scenarios: the first scenario shows that the sampling scheme from Section 3 can greatly improve the performance of the ‘standard’ IWLS sampler with separate blocks for each distributional parameter and smooth term. Scenario II resembles the real-world application to intra-annual tree stem growth in Section 5, extending it with an artificial time-varying covariate as a proof of concept. In the third scenario, we use GPs on a sphere, which can be understood as shapes of tree crowns. While this simulation is not immediately linked to the application of analysing tree stem radial growth, it underlines that the index set of the GPs does not need to be one-dimensional or Euclidean. To communicate a clear message with each scenario, we refrained from adding unnecessary complexity: All scenarios use normal priors with mean zero and standard deviation 1,000 for the regression coefficients, 100 replications of the data-generating process, and MCMC chains of length 1,000 after a burn-in of 200 iterations.

4.1 Scenario I: joint sampling

In this scenario, we use a constant mean function and the Matérn covariance function c^m from Equation (7). The smoothness parameter ν of the Matérn covariance function is fixed to 1.5, and the predictors and inverse link functions are defined to be

$$\mu_i = x_{i1}, \quad \sigma_i = \exp(x_{i2} + x_{i4}), \quad \text{and} \quad \phi_i = \exp(\beta_0 + x_{i3} + x_{i4}),$$

where $x_{i1}, x_{i2}, x_{i3}, x_{i4} \overset{\text{ind.}}{\sim} \mathcal{U}(0, 1)$ are the covariates. The observation index i runs from 1 to N, where $N = 30$ is the number of GPs, and $n_i = n = 30$ is the number of observed values per GP. The unit interval serves as the index set of the GPs.

For β_0, the intercept for the range ϕ_i, we use two different values: −3, such that $\phi_i \in [0.049, 0.368]$, and 0, such that $\phi_i \in [1, 7.39]$. We call $\beta_0 = -3$ the ‘small-range scenario’ and $\beta_0 = 0$ the ‘large-range scenario.’ Figure 2 shows 30 simulated GPs from one exemplary replication of the simulation scenario. The realisations of the GPs in the large-range scenario seem almost linear, while the realisations in the small-range scenario are much more wiggly.

We ran 100 replications of this setup, both with a small and a large range. In a next step, we sampled the (correctly specified) model for each replication and range, one time with separate blocks for the regression coefficients for the standard deviation and the range, and another time with one joint block for these coefficients. The bias of the posterior mean estimates is negligible for both samplers, but in terms of the autocorrelation of the chains, there are substantial differences between them. While the autocorrelation is similar for the regression coefficients for x_{i2} and x_{i3}, the joint sampler beats the one with separate blocks by far for the intercept and the regression coefficients for x_{i4} (the covariate with an effect on both the standard deviation and the range; see Figure 3 for a comparison of the trace plots). With a small range, the performance gap is less extreme but still very apparent.

We conclude from Scenario I that using a joint sampler for the covariance parameters is much more efficient. In contrast, with the sampling scheme with separate parameter blocks for the covariance parameters, the resulting MCMC chains have a high autocorrelation and must be inspected carefully. For Scenarios II and III, similar performance differences between the samplers can be observed, which we demonstrate in the supplementary material to this article.

4.2 Scenario II: time-varying covariates

The mean function of the GPs in this scenario is constructed as the sum of the Weibull growth curve from Equation (6) and the linear mean function from Equation (5), yielding

$$m(z_i(\boldsymbol{t}) = [t, u_i(t)]^\top; \boldsymbol{\theta}_i^m = [l_i, a_i, b_i, w_i]^\top) = m^w(t; [l_i, a_i, b_i]^\top) + m^l(u_i(t); w_i),$$

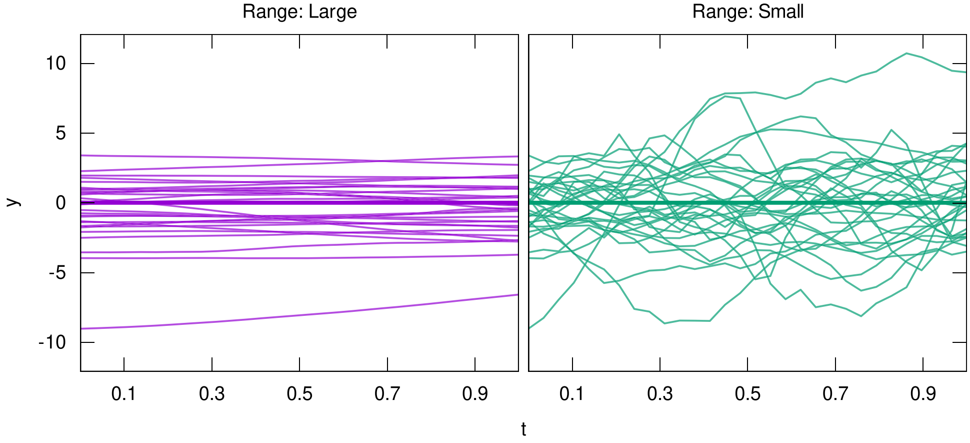


Figure 2. The GPs from one exemplary replication of Simulation Scenario I. The range takes values between 1 and 7.39 for the GPs on the left-hand side and between 0.049 and 0.368 for those on the right-hand side.

where $u_i(t)$ is a univariate time-varying covariate, and the parameter w_i determines the effect size of $u_i(t)$. The covariance function of the GPs is

$$c^s(z_i(\boldsymbol{t}) = t, z_i(\boldsymbol{t}') = t'; \boldsymbol{\theta}_i^c = [\sigma_i, \phi_i]^\top) = q(t) \times q(t') \times c^m(t, t'; [\sigma_i, \phi_i]^\top), \tag{9}$$

where c^m is the Matérn covariance function from Equation (7) with the smoothness parameter $v = 1.5$. The auxiliary function $q(t)$ is defined as

$$q(t) = \begin{cases} 0.1 + \frac{0.9}{30} \times t & \text{if } 0 \leq t < 30, \\ 1 & \text{otherwise}, \end{cases} \tag{10}$$

and scales the standard deviation of the GPs over time, such that it increases linearly on the interval [0, 30] and remains constant afterwards. The motivation for this step is that the growth curves in the application in Section 5 are defined to start at zero on April 1 of each year (i.e., at the beginning of each growing season) and have little variability in the first couple of weeks after that.

We define the predictors and inverse link functions $l_i = 1500 \approx \exp(7.313)$, $a_i = 3.5 \approx \exp(1.253)$, $b_i = 100 \approx \exp(4.605)$, $w_i = x_i$, $\sigma_i = 40 \approx \exp(3.689)$, and $\phi_i = 2 \approx \exp(0.693)$. The only explanatory variable x_i, where $i = 1, \ldots, N$, is independent and uniformly distributed on the interval [1, 2]. The number of GPs takes the values $N = 30$, 60, or 120, and the number of observed values per GP is $n_i = n = 60$ or 120. As the index set of the GPs, we use the interval [0, 182], representing the days during one growing season.

The focus of this scenario is on the time-varying covariate $u_i(t)$, which we simulate as i.i.d. GPs with mean zero and a squared exponential covariance function, scaled over time with the auxiliary function $q(t)$ from Equation (10). In the context of intra-annual tree stem growth, the time-varying covariate $u_i(t)$ could, for example, be a mean-centred moving average over the precipitation at the location of a tree i. How much the precipitation affects the growth dynamics of tree i depends on the explanatory variable x_i, which might represent the soil conditions around that tree. Figure 4 illustrates this simulation scenario.

For each possible N-n-combination, we ran 100 replications and found that the sampling scheme from Section 3 works very reliably. The posterior mean estimates do not show any systematic bias (Figure 5). They have very little variability around the true value for the parameters of the Weibull growth curve, while more variability is observed for the parameter of the time-varying covariate and the covariance parameters. As expected, the quality of the estimates improves with the sample size, where N has a stronger effect on the quality than n, because additional independent response GPs are more informative than additional dependent observations within each GP. In

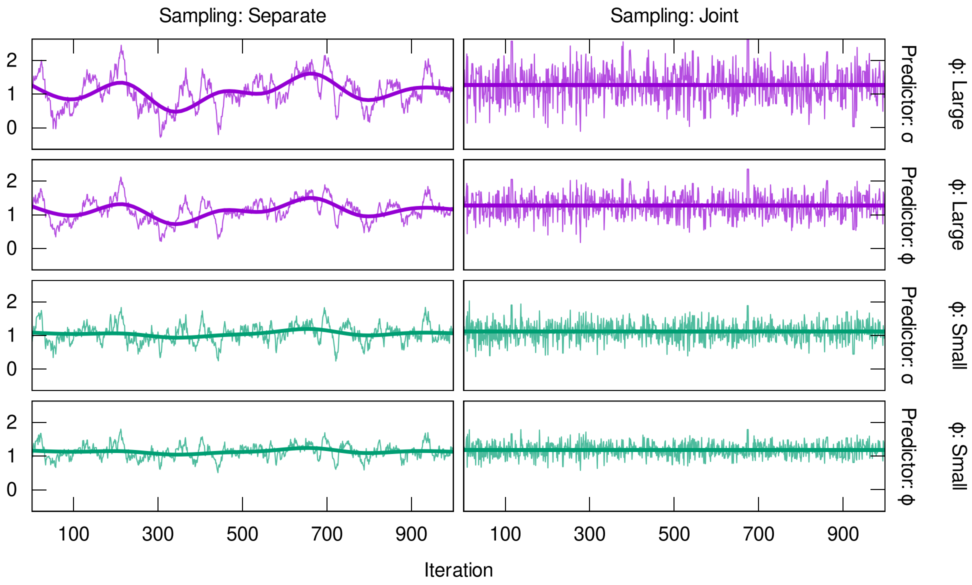


Figure 3. Trace plots for the regression coefficients for x_{i4} from one exemplary replication of Simulation Scenario I. The left-hand side shows the sampler with separate blocks for the regression coefficients for the covariance parameters, whereas the right-hand side shows the sampler with one joint block for these coefficients. No thinning was applied to the chains.

summary, this simulation scenario shows that our sampler has the expected properties for a correctly specified model, even if time-varying covariates are included.

Finally, we highlight that the boxplots in Figure 5 show the performance of the sampler under the assumption that the true smoothness parameter ν is known, which is usually not the case in practice. However, a misspecified smoothness parameter ($\nu = 0.5$ or 2.5 instead of 1.5 in this scenario) does not seem to have a strong adverse effects on the inferences drawn from such a model. In the supplementary material to this article, we show that a misspecified smoothness parameter is mostly compensated for by the estimated range parameter ϕ, while the other parameters remain essentially unaffected. Rather, we found that a reasonable choice for ν can increase the model stability and improve the MCMC mixing, as we will investigate in Section 5.

4.3 Scenario III: processes on a sphere

In this scenario, we show how our model can accommodate spatial or spatio-temporal processes as response structures. Generally speaking, the GPs can be defined on a one- or higher-dimensional Euclidean space, or even a non-Euclidean metric space when employing appropriate distances. The processes in this specific scenario are defined on a sphere, resembling shapes of tree crowns, and we use the great circle distance for quantifying distances. Figure 6 illustrates the design: the object on the left is an ‘average’ tree crown, from which we simulated a more realistic shape as a realisation of a GP with an exponential covariance function, shown on the right. In an application, the tree species or the light availability could be used as covariates to explain the properties of the mean and the covariance function of the crown shapes. The mean properties are, among others, the average radius, and the vertical elongation, while the covariance properties are the size and the persistence of the deviations from the mean.

The mean function in this scenario is defined in terms of the linear mean function from Equation (5), which is applied to a transformed coordinate vector as follows:

$$
\begin{aligned}
&m(z_i(\boldsymbol{t}) = [t_1,\, t_2]^\top;\, \boldsymbol{\theta}_i^m = [r_i,\, h_i,\, v_i]^\top)\\
=&m^l([1,\ \cos(t_2)(\cos(t_1)+1),\, t_2+\pi/2]^\top;\, [r_i,\, h_i,\, v_i]^\top)\\
=&r_i + \cos(t_2)(\cos(t_1)+1)\times h_i + (t_2+\pi/2)\times v_i.
\end{aligned}
$$

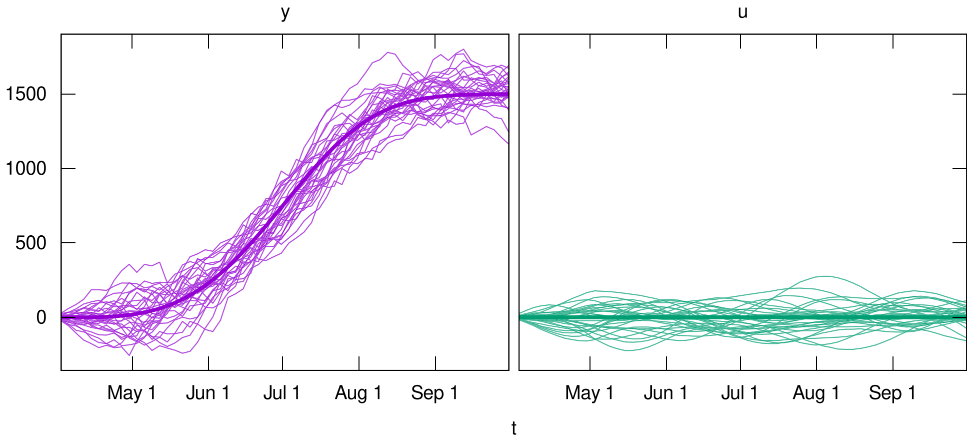


Figure 4. The response GPs $Y_i(t)$ (on the left-hand side) and the time-varying covariates $u_i(t)$ (on the right-hand side) from one exemplary replication of Simulation Scenario II with $N = 30$ and $n = 60$. The large-scale deviations of $Y_i(t)$ from the Weibull growth curve are mainly driven by mean shifts due to $u_i(t)$, while the small-scale variation comes from the covariance structure of $Y_i(t)$.

The coordinates $t_1 \in [-\pi, \pi]$ and $t_2 \in [-\pi/2, \pi/2]$ are the longitude and the latitude on the sphere in radians. The parameters are r_i, the minimum radius of the tree crown, h_i, the horizontal elongation towards the south, and v_i, the vertical elongation. Furthermore, we use the Matérn covariance function c^w from Equation (7) with the smoothness parameter $\nu = 0.5$, i.e., an exponential covariance function.

The predictors and inverse link functions are defined as $r_i = \exp(x_{i1})$, $s_i = \exp(x_{i2})$, $h_i = \exp(1 + x_{i3})$, $\sigma_i = \exp(x_{i4})$, and $\phi_i = \exp(x_{i5})$, where the explanatory variables x_{i1}, x_{i2}, x_{i3}, x_{i4}, and x_{i5} are independent and uniformly distributed on the unit interval and the observation index is $i = 1, \ldots, N$. The number of GPs is set to $N = 30$, and the number of observed values per GP is $n_i = n = 379$. A regular longitude–latitude grid is used for the observations of the GPs.

With a maximum value of 0.014 for the covariate effect of x_{i1} on the radius r_i, the average bias is negligible for all posterior mean estimates. The average mean squared errors (MSEs) are also very small, especially for the regression coefficients for the vertical elongation v_i, the standard deviation σ_i, and the range ϕ_i. Among these regression coefficients, the maximum average MSE is 0.006 for the covariate effect of x_{i3} on v_i. The average MSEs for the regression coefficients for r_i and the horizontal elongation h_i are higher but still uncritical with values between 0.02 and 0.26 (results not shown graphically for this scenario). These numbers indicate that we are able to estimate the model parameters reliably with our sampling scheme, despite the fact that the GPs are defined on a non-Euclidean space.

5 Intra-annual tree stem growth

In this section, we apply our method to the intra-annual stem growth of 72 beeches, 6 ashes, and 7 sycamores from three different regions in Germany. For each tree, the growing seasons 2012 and 2013 were recorded in a high temporal resolution using electronic circumference dendrometers. The original purpose of the data was a study on the effect of the neighbourhood identity on the growth patterns of beech trees in pure and mixed stands (Metz et al., 2020), which was conducted in the *Biodiversity Exploratories* (Fischer et al., 2010). The dataset can be downloaded from the information system of the project (*BExIS*, Metz & Ammer, 2018; Ostrowski et al., 2016), thanks to the open data policy of the *Biodiversity Exploratories*. Our analysis is fully replicable with the code in the supplements.

A first outlook on the dataset was given in Figure 1. We compared a beech to an ash and observed differences in the overall annual growth, the starting time of the growth process, and the shrinking and swelling. With our model, we formalise these observations and assess the effect of explanatory variables such as the DBH or the geographical location. We show that our model

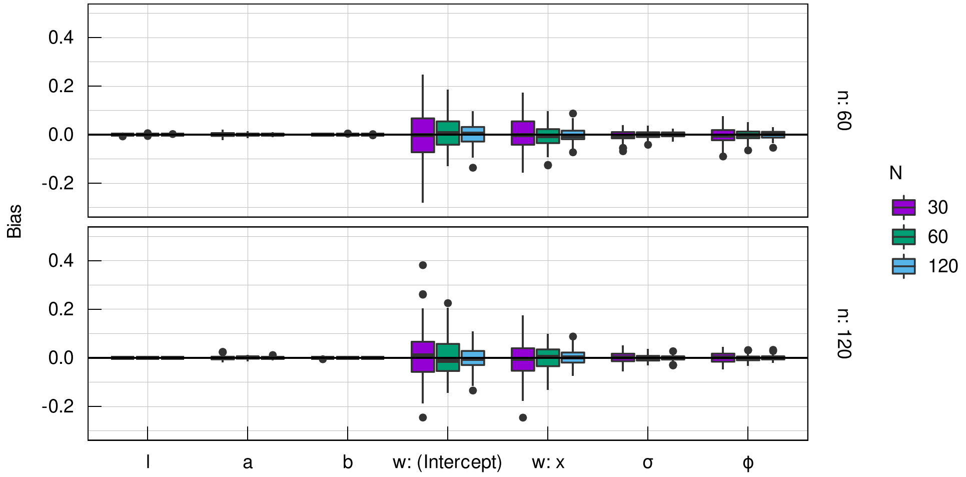


Figure 5. Boxplots of the bias of the posterior mean estimates in Simulation Scenario II. Each boxplot represents 100 replications for one combination of N, the number of response GPs, and n, the number of observed values per GP.

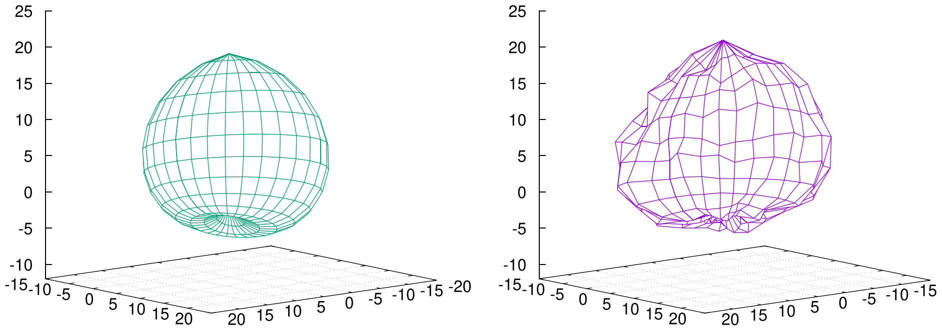


Figure 6. An exemplary mean function, shown on the left, and a corresponding realisation of a GP with an exponential covariance function, shown on the right, from one replication of Simulation Scenario III. The objects are designed to resemble shapes of tree crowns. The properties of the shapes can be related to explanatory variables such as the tree species or the light availability.

is instructive when applied to high-resolution dendrometer data. Our statistical approach is different from the one used by Metz et al. (2020), who estimate one Weibull curve per observed curve, each fitted individually by non-linear least squares, and then model the estimated parameters and other derived quantities. One downside of this two-step procedure is that the estimation uncertainty of the parameters is not systematically taken into account in the second step. We solve this problem with an explicit assumption about the probability distribution of the stochastic process of intra-annual tree stem growth and a one-step inference algorithm.

5.1 Model specification

Our analysis is based on the Weibull growth curve $m^w(t)$ from Equation (6) as a mean function and the scaled Matérn covariance function $c^s(t, t')$ from Equation (9). Consequently, we have the following five distributional parameters: the limit l, the shape a, and the scale b of the Weibull growth curve, and the standard deviation σ and the range ϕ of the covariance function.

The covariance function describes the shrinking and swelling of the tree stems and does not include any additional parameters to account for potential measurement errors from the dendrometers. The motivation for this approach is that electronic dendrometers are high-precision devices, and the individual measurement errors are on a very small scale. Large errors occur only when the dendrometers are touched, e.g., by an animal or a researcher. These errors were already corrected in our dataset with a simple post-processing step. Finally and most importantly, the data were originally recorded in a very high temporal resolution of 30 min. As we are only working with daily data, a potential additive error per half-hourly measurement would average out over the 48 measurements on one day. From a theoretical perspective, however, including a measurement error in the model would be straightforward, see the discussion in Section 2.2.2.

The predictors and inverse link functions are defined as

$$\begin{aligned}
l_i &= \exp(\beta_{l0} + (\text{Tree} * \text{Year})_i \times \boldsymbol{\beta}_{l1}),\\
a_i &= \exp(\beta_{a0} + \text{Species}_i \times \boldsymbol{\beta}_{a1} + \text{DBH}_i \times \beta_{a2} + (\text{Site} * \text{Year})_i \times \boldsymbol{\beta}_{a3}),\\
b_i &= \exp(\beta_{b0} + \text{Species}_i \times \boldsymbol{\beta}_{b1} + \text{DBH}_i \times \beta_{b2} + (\text{Site} * \text{Year})_i \times \boldsymbol{\beta}_{b3}),\\
\sigma_i &= \exp(\beta_{\sigma0} + \text{Species}_i \times \boldsymbol{\beta}_{\sigma1} + \text{DBH}_i \times \beta_{\sigma2} + f_{\text{Year}_i}(x_i, y_i; \boldsymbol{\beta}_{\sigma3})),\\
\phi_i &= \exp(\beta_{\phi0} + \text{Species}_i \times \boldsymbol{\beta}_{\phi1} + \text{DBH}_i \times \beta_{\phi2} + (\text{Site} * \text{Year})_i \times \boldsymbol{\beta}_{\phi3}),
\end{aligned}$$

where $\beta_{\bullet,0}$ and $\beta_{\bullet,2}$ are scalar regression coefficients, while $\boldsymbol{\beta}_{\bullet,1}$ and $\boldsymbol{\beta}_{\bullet,3}$ are vectors of regression coefficients, and Species$_i$ denotes the entries of the design matrix for the dummy variable for the species of the tree where the ith growth curve was recorded,

$$\text{Species}_i = \begin{cases} [0, 0] & \text{if the } i\text{th growth curve is of a beech,}\\ [1, 0] & \text{if it is of a ash; and}\\ [0, 1] & \text{if it is of a sycamore.} \end{cases}$$

Similarly, $(\text{Tree} * \text{Year})_i$ and $(\text{Site} * \text{Year})_i$ are the entries of the design matrix for the interaction of two dummy variables: in the first case, of the individual tree and the year, and in the second case, of the field site and the year. Finally, f_{Year_i} denotes a year-specific spatial kriging smooth.

The tree-year-interaction in the predictor for the limit parameter implies one degree of freedom for the limit of each growth curve. The other variables in the dataset do not explain the overall annual growth sufficiently, but the limit is identified well enough for each growth curve that these parameters can be estimated without problems. All other predictors include the species and the DBH as covariates. For the shape, the scale, and the range, the site-year-interaction captures the spatial and temporal differences between the field sites and the years. For the standard deviation, the smooth term f_{Year_i} serves this purpose and illustrates the flexibility of structured additive predictors when used with covariance parameters in the GP framework.

For the regression coefficients $\boldsymbol{\beta}$, we used uninformative $\mathcal{N}(0, 1000)$ priors, and an inverse gamma prior with fixed hyperparameters $a = b = 0.0001$ for the smoothing parameter τ^2 of the spatial kriging smooth f_{Year_i}.

5.2 Tree physiology

The dendrometer measurements of the stem radius of a tree are composed of an irreversible growth component and temporary shrinking and swelling dynamics, which can be further divided into water potential-driven and osmotic processes (Chan et al., 2016). Water potential-driven changes in the stem radius are caused by sap moving radially within the xylem or between the xylem and the phloem from areas with a higher to areas with a lower water potential. This process give rise to an approximately periodic fluctuation of the dendrometer measurements over the course of 24 hr, while osmotic changes occur more gradually, for example, if the tree draws water from the roots.

In our case, the mean curves can be interpreted as the irreversible growth and the fluctuations around them as temporary shrinking and swelling. These fluctuations are characterised by the covariance function of the GPs. As the employed Matérn covariance function is stationary, the GPs keep returning to their mean. How fast they return depends on the range parameter, which is estimated to be relatively small for most trees in the dataset. As we analyse the dendrometer measurements in a daily resolution, the water potential-driven changes are averaged out from the data for the most part, and the remaining fluctuations are primarily osmotic. In fact, the deviations of the growth curves from the estimated mean curves that we observe in the data do typically last a few days or weeks, as expected for the osmotic processes in tree stems.

If a growth curve increases faster than the mean curve, we interpret this as the tree drawing more water from the roots than required for the formation of new cells and the irreversible growth at a given moment. Conversely, if a growth curve increases slower than the mean curve, more water is consumed by the irreversible growth than drawn from the roots. Finally, if a growth curve decreases, water is released from the stem. These processes are reflected in the stochastic part of the model, which is characterised by the covariance parameters: the standard deviation quantifies the magnitude of the osmotic changes in the stem radius, and the range parameter their persistence. For example, some tree species might store more water in the stem than others (which would imply a higher standard deviation), or they might store it for a longer time (which would imply a higher range parameter).

Different approaches for the decomposition of dendrometer measurements have been proposed, among others, by Zweifel et al. (2005) and Chan et al. (2016). Zweifel et al. (2005), on the one hand, use a linear interpolation of the local maxima of the observed growth curves as an approximation of the irreversible growth and interpret the difference between the interpolation and the observed curves as the tree water deficit. They find that this measure of tree water deficit is explained well by soil water potential and vapour pressure deficit for pine, oak, and spruce under different environmental conditions in Switzerland. Chan et al. (2016), on the other hand, compute an estimate of the sum of the irreversible growth and the osmotic changes in the stem radius from dendrometer measurements of both the whole stem and the xylem radial thickness. The irreversible growth is then obtained as the difference of the minima of this estimate on two consecutive days.

Despite the apparent similarities between our model framework and the approaches of Zweifel et al. (2005) and Chan et al. (2016), the scope of the methods is quite different: While the other approaches are motivated from ecophysiological considerations, our model takes advantage of statistical assumptions about the parametric form of the mean and covariance function of the GPs in a regression setting. It can be used to decompose dendrometer measurements into a permanent and a temporary component for any given point in time, but this is not our primary goal, and the decomposition is likely to be less accurate than the ones from the other, more specialised methods. Instead, our model does focus on the relationship between structural patterns of both the irreversible growth and the temporary shrinking and swelling throughout the vegetation period and a set of explanatory variables. As mentioned before, such structural patterns could be for example the magnitude or persistence of the osmotic changes in the stem radius. In particular, we would like to point out the following advantages of our approach:

- The model has minimal data requirements: A single circumference dendrometer per tree is sufficient. Zweifel et al. (2005) and Chan et al. (2016) use one or two point dendrometers, respectively. The model is also agnostic about the temporal resolution of the data, while the method of Chan et al. (2016) requires multiple measurements per day.
- The model can be estimated with one integrated MCMC algorithm, which estimates the decomposition of the dendrometer measurements into irreversible growth and temporary shrinking and swelling, and the effects of the explanatory variables on the characteristics of the growth curves at the same time. The advantage is that the estimation uncertainty can be assessed in a sound statistical framework, and a two-step procedure can be avoided.
- The structural patterns of the growth curves that can be explained by covariates are not limited to the osmotic changes in the stem radius, but include characteristics of the overall intra-annual sigmoid growth curves as well, such as the total growth of a tree over an entire vegetation period or the steepness of a growth curve; for details, see Section 2.2.1.

The overall aim of our article is to present a model *framework* rather than one specific model for tree stem growth. The application in this section should illustrate how the different components of the framework can be interpreted in the context of tree stem growth. For the sake of simplicity, a relatively low temporal resolution and a limited number of covariates are used, but the model could easily be refined with a different mean or covariance function, more covariates, or a higher temporal resolution. In the view of this, the methods of Zweifel et al. (2005) and Chan et al. (2016) should not be considered as competitors of our model, but their studies could serve as a basis for the development of more specific models in our framework that could provide further insights into the ecophysiological process of tree stem growth. In fact, recent studies by Zweifel et al. (2021) have confirmed vapour pressure deficit as an important driver of tree stem growth, so including it as a time-varying covariate in the mean or covariance function could be particularly instructive. In our application, the site-year-interactions in the predictors serve as proxies for the average weather conditions at a given site in a given year and ensure that the species and DBH effects do not suffer from an omitted variable bias, but the explicit inclusion of vapour pressure deficit or soil water potential would give rise to a more direct model.

Recent articles have addressed the question during which time of the day trees and other plants grow the most (Wiese et al., 2007; Zweifel et al., 2021). To investigate this problem in our model framework, the growth curves would need to be processed in a higher, e.g., hourly, temporal resolution, which would increase the computational cost of our model and require some adjustments of the covariance and possibly the mean function of the GPs: to account for the daily pattern of the water potential-driven changes in the tree stem radius, an additive periodic kernel could be included in the covariance function, for example, an exponential sine squared kernel (Rasmussen & Williams, 2006, Section 4.2). To address the question whether the irreversible growth primarily occurs during the day or the night, the mean function could be modified to represent a sequence of daily smooth step functions, where a parameter could be introduced to estimate the time of the day around which the steps are centred.

5.3 Estimation results

We used the sampling scheme from Section 3 to estimate our tree growth model with results presented in this section based on MCMC samples from the posterior distribution with a sample size of 10,000, excluding the 2,000 burn-in iterations. No thinning was applied to the chains. The effective sample size ranges from 703.169 to 8,793.660, as for some regression coefficients in the predictors for the mean parameters, the chains exhibit moderate to strong autocorrelation.

Table 1 summarises the posterior samples of the species effect of ash and the effect of DBH on the predictors. The reference category for the species effect is beech, so the negative effect on the scale implies that, on average, an ash stops growing earlier during the vegetation period than a beech. As different species allocate resources differently throughout the growing season, this is an expected result. The positive effect of DBH on the growth duration during the vegetation period might be due to the fact that trees with a greater DBH are more likely to be dominant in the stand. When the light availability decreases in the fall, the dominant trees still continue to grow, while the smaller trees cannot keep up their growth.

In terms of the covariance parameters, ash has a positive effect on the standard deviation. This is plausible because ash has a thicker bark than beech, which means it can store more water in the bark. The same argument applies for the effect of DBH on the standard deviation, as larger trees have a thicker bark. The effect of ash on the range parameter is estimated to be negative, which means that the osmosis-induced changes in the stem radius are less persistent for ash than for beech, possibly because the water is stored for shorter periods of time. Finally, the effect of DBH on the range is slightly positive, but the 95% credible interval does not exclude a zero effect.

The spatial kriging smooth in the predictor for the standard deviation is displayed in Figure 7. The six rectangles represent the three study regions Schwäbische Alb, Hainich-Dün, and Schorfheide-Chorin of the *Biodiversity Exploratories* and the years 2012 and 2013. The locations of the field sites in the study regions are marked with small crosses. A brighter colour indicates a higher standard deviation of the growth curves of the trees at a given location. The figure shows that the differences in the standard deviation are greater between than within the study regions: The trees on the Schwäbische Alb have the highest standard deviation, followed by those in the Hainich-Dün and the Schorfheide-Chorin. This pattern is stable over the years and very likely a result of differences in the precipitation, which are greater on a large than on a small scale. Note that the estimated effects are extrapolated considerably beyond areas supported by the observations, such that these parts of the rectangles in Figure 7 should be interpreted with care.

To check the robustness of the results with respect to the smoothness assumption of the correlation function, we also estimated the model with the smoothness parameters $\nu = 0.5$ and 2.5 instead of 1.5. We found that all three models produce similar results, both in terms of the model fit and the

Table 1. Summary statistics of the posterior samples of the species effect of ash (vs. beech) and the effect of DBH on the predictors

Ash	Coefficient	Mean	2.5%	Median	97.5%
Shape	$\beta_{a1,1}$	−0.289	−0.363	−0.289	−0.218
Scale	$\beta_{b1,1}$	−0.426	−0.452	−0.426	−0.400
Std. dev.	$\beta_{\sigma 1,1}$	0.602	0.529	0.602	0.678
Range	$\beta_{\phi 1,1}$	−0.101	−0.157	−0.101	−0.043
DBH	**Coefficient**	**Mean**	**2.5%**	**Median**	**97.5%**
Shape	β_{a2}	−0.027	−0.044	−0.027	−0.009
Scale	β_{b2}	0.050	0.045	0.050	0.055
Std. dev.	$\beta_{\sigma 2}$	0.085	0.064	0.085	0.106
Range	$\beta_{\phi 2}$	0.011	−0.004	0.011	0.027

estimated covariate effects. For $v = 0.5$, we had to use informative standard normal priors for the regression coefficients $\boldsymbol{\beta}$ instead of the default $\mathcal{N}(0, 1000)$ priors, and still some parameters appeared to be poorly identified, deteriorating the mixing of the MCMC chains. Comparing the results for $v = 1.5$ and 2.5, we obtained the better DIC for $v = 1.5$, making this model our preferred specification. The estimated covariate effects, particularly their signs and sizes, were comparable for all three models, such that the interpretation of the model is unaffected by the specific choice of the smoothness parameter. For more details, see the supplementary material to this article.

The discussion of the model results shows that our framework allows us to relate different properties of the tree growth curves to explanatory variables and complex covariate effects (such as the spatial kriging smooth in this example) in a very direct way. Using the methods of Zweifel et al. (2005) or Chan et al. (2016), similar results could be obtained by decomposing the dendrometer data, defining measures for the phenomena of interest, e.g., the persistence of the osmotic changes in the stem radius, based on the decomposition, and finally using these measures as response variables in different regression models. Both approaches have benefits and drawbacks depending on the goal of the analysis, but if the focus is on the effect of the explanatory variables, our model is arguably more comprehensive.

6 Discussion

In this paper, we embedded GPs as response structures into the framework of structured additive distributional regression as described by Klein et al. (2015) to study intra-annual tree stem growth and to decompose high-resolution dendrometer measurements into irreversible growth and temporary shrinking and swelling. It is of particular interest for the physiological understanding of stem growth that our model can explain certain properties of both components of the dendrometer measurements by covariates such as the tree species or the DBH. Based on a dataset of 85 individual trees from Germany, for which the variations in the stem radius were recorded during the growing seasons 2012 and 2013, we could identify different growth patterns for three deciduous tree species: for instance, ash grows more gradually and earlier during the vegetation period than beech, and its thick bark gives rise to a more pronounced temporary shrinking and swelling. Our model can quantify these differences between tree species and conditional on other explanatory variables with a sound and unified statistical approach.

We want to point out that the design of the structured additive predictors, that is the selection of the explanatory variables and their effect type, requires special care for the proposed type of model. Theoretical considerations and subject-matter expertise must be taken into account to build models with meaningful interpretations for the research questions at hand. Concerning the analysis of intra-annual tree stem growth, additional, more detailed models including precipitation data and other time-varying explanatory variables could be a promising next step, especially since

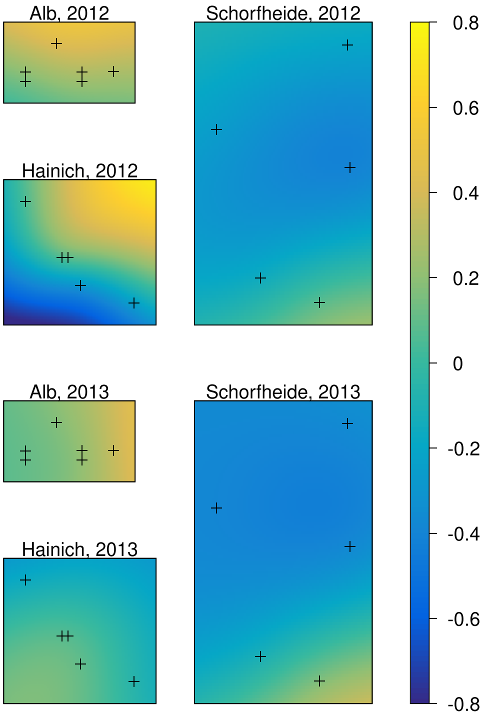


Figure 7. Posterior mean of the spatial kriging smooth f_{Year_i} in the predictor for the standard deviation. The field sites are marked with small crosses. Lighter colours indicate a higher standard deviation.

in the light of global climate change, a comprehensive understanding of the trees' reaction to drought stress is becoming more and more indispensable.

While the original use case for our model class is the analysis of intra-annual tree stem growth, the flexibility, and versatility of the model framework was discussed throughout the paper and demonstrated in particular in the simulation study. The general model class certainly deserves further investigation in the future. To explore its full potential in many other applications, it will be necessary to study the various members of the model class arising from specific choices of index sets, mean functions, and covariance functions, and to develop applications for research questions in different fields.

Other aspects that deserve further attention are both theoretical and software-related. We implemented an R package that can fit the models described in this paper but does not yet support arbitrary mean and covariance functions. The challenge will be to keep the performance cost of these generalisations as small as possible. To reach a greater audience, the package will also need a more complete documentation and a better user interface. In terms of open theoretical questions, the propriety of the posterior distribution and the ergodicity of the MCMC chains comes to mind. Finally, following up on the discussion in Sections 3 and 4, a more thorough investigation of the correlation structure of the model parameters could guide the development of more efficient MCMC sampling schemes.

Acknowledgments

We would like to thank Jérôme Metz, Peter Annighöfer, and Christian Ammer for providing us with the tree growth dataset, which was originally collected for the DFG Priority Program 1374: Biodiversity Exploratories, and two anonymous reviewers and the associate editor for their helpful comments.

Conflict of interest: None declared.

Funding statement

The work of H.R. on this article was funded by the German Research Foundation (DFG) through the Research Training Group 2300: Enrichment of European Beech Forests with Conifers (316045089/GRK2300). N.K. was funded by the German Research Foundation (DFG) through the Emmy Noether grant KL 3037/1-1.

Data availability

All data that were used in the article is freely available online and listed in the references.

Supplementary material

Supplementary material on the smoothness parameter of the Matérn correlation function and the different sampling schemes presented in the article are available at *Journal of the Royal Statistical Society: Series C* online. The code to replicate the results of the article is also available on GitHub (https://github.com/hriebl/gp-responses, https://github.com/hriebl/bamlssGP).

References

Adler R. J. (1990). *An introduction to continuity, extrema, and related topics for general gaussian processes*. Lecture Notes – Monograph Series, Vol. 12, Institute of Mathematical Statistics. http://www.jstor.org/stable/4355563.

Chan T., Hölttä T., Berninger F., Mäkinen H., Nöjd P., Mencuccini M., & Nikinmaa E. (2016). Separating water-potential induced swelling and shrinking from measured radial stem variations reveals a cambial growth and osmotic concentration signal. *Plant, Cell & Environment*, *39*(2), 233–244. https://doi.org/10.1111/pce.12541

Cressie N. A. C. (1993). *Statistics for spatial data*. Wiley.

Fahrmeir L., Kneib T., & Lang S. (2004). Penalized structured additive regression for space-time data: A Bayesian perspective. *Statistica Sinica*, *14*(3), 731–761.

Fahrmeir L., Kneib T., Lang S., & Marx S. (2013). *Regression: Models, methods and applications*. Springer.

Filippou P., Marra G., & Radice R. (2017). Penalized likelihood estimation of a trivariate additive probit model. *Biostatistics*, *18*(3), 569–585. https://doi.org/10.1093/biostatistics/kxx008

Fischer M., Bossdorf O., Gockel S., Hänsel F., Hemp A., Hessenmöller D., Korte G., Nieschulze J., Pfeiffer S., Prati D., Renner S., Schöning I., Schumacher U., Wells K., Buscot F., Kalko E. K. V., Linsenmair K. E., Schulze E.-D., & Weisser W. W. (2010). Implementing large-scale and long-term functional biodiversity research: The biodiversity exploratories. *Basic and Applied Ecology*, *11*(6), 473–485. https://doi.org/10.1016/j.baae.2010.07.009

Gamerman D. (1997). Sampling from the posterior distribution in generalized linear mixed models. *Statistics and Computing*, *7*(1), 57–68. https://doi.org/10.1023/A:1018509429360

Gneiting T. (2002). Nonseparable, stationary covariance functions for space-time data. *Journal of the American Statistical Association*, *97*(458), 590–600. https://doi.org/10.1198/016214502760047113

Gneiting T. (2013). Strictly and non-strictly positive definite functions on spheres. *Bernoulli*, *19*(4), 1327–1349. https://doi.org/10.3150/12-BEJSP06

Greven S., & Scheipl F. (2017). A general framework for functional regression modelling. *Statistical Modelling*, *17*(1–2), 1–35. https://doi.org/10.1177/1471082X16681317

Klein N., & Kneib T. (2016a). Scale-dependent priors for variance parameters in structured additive distributional regression. *Bayesian Analysis*, *11*(4), 1071–1106. https://doi.org/10.1214/15-BA983

Klein N., & Kneib T. (2016b). Simultaneous inference in structured additive conditional copula regression models: A unifying Bayesian approach. *Statistics and Computing*, *26*(4), 841–860. https://doi.org/10.1007/s11222-015-9573-6

Klein N., Kneib T., Lang S., & Sohn A. (2015). Bayesian structured additive distributional regression with an application to regional income inequality in Germany. *The Annals of Applied Statistics*, *9*(2), 1024–1052. https://doi.org/10.1214/15-AOAS823

Klepper B., Browning V. D., & Taylor H. M. (1971). Stem diameter in relation to plant water status. *Plant Physiology*, *48*(6), 683–685. https://doi.org/10.1104/pp.48.6.683

Mencuccini M., Salmon Y., Mitchell P., Hölttä T., Choat B., Meir P., O'Grady A., Tissue D., Zweifel R., Sevanto S., & Pfautsch S. (2017). An empirical method that separates irreversible stem radial growth from bark water content changes in trees: Theory and case studies. *Plant, Cell & Environment*, *40*(2), 290–303. doi:10.1111/pce.12863

Metz J., & Ammer C. (2018). Dendrometer data of trees, neighbor, 1 MIP, 2012–2013. Biodiversity Exploratories Information System. Dataset. https://www.bexis.uni-jena.de/ddm/data/Showdata/17766?version=12.

Metz J., Annighöfer P., Westekemper K., Schall P., Schulze E.-D., & Ammer C. (2020). Less is more: Effects of competition reduction and facilitation on intra-annual (basal area) growth of mature European beech. *Trees*, *34*(1), 17–36. https://doi.org/10.1007/s00468-019-01894-7

Ostrowski A., Nieschulze J., Schulze E.-D., Fischer M., Ayasse M., Weisser W., & König-Ries B. (2016). Basic information and coordinates of field plots of the biodiversity exploratories project. Biodiversity Exploratories Information System. Dataset. https://www.bexis.uni-jena.de/ddm/data/Showdata/1000?version=7.

Paciorek C. J. (2003). *Nonstationary Gaussian processes for regression and spatial modelling* [Ph.D. thesis]. Carnegie Mellon University.

Pinheiro J., Bates D., DebRoy S., & Sarkar D., & R Core Team (2020). *nlme: Linear and nonlinear mixed effects models*. R package version 3.1-147. https://CRAN.R-project.org/package=nlme.

Rasmussen C. E., & Williams C. K. I. (2006). *Gaussian processes for machine learning*. MIT Press.

R Core Team (2020). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org.

Rigby R. A., & 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. doi:10.1111/j.1467-9876.2005.00510.x

Scheipl F., Staicu A.-M., & Greven S. (2015). Functional additive mixed models. *Journal of Computational and Graphical Statistics*, *24*(2), 477–501. https://doi.org/10.1080/10618600.2014.901914

Shi J. Q., & Choi T. (2011). *Gaussian process regression analysis for functional data*. CRC Press.

Umlauf N., Klein N., & Zeileis A. (2018). Bamlss: Bayesian additive models for location, scale, and shape (and beyond). *Journal of Computational and Graphical Statistics*, *27*(3), 612–627. https://doi.org/10.1080/10618600.2017.1407325

Wiese A., Christ M. M., Virnich O., Schurr U., & Walter A. (2007). Spatio-temporal leaf growth patterns of arabidopsis thaliana and evidence for sugar control of the diel leaf growth cycle. *New Phytologist*, *174*(4), 752–761. https://doi.org/10.1111/j.1469-8137.2007.02053.x

Wood S., & Scheipl F. (2020). *gamm4: Generalized additive mixed models using 'mgcv' and 'lme4'*. R package version 0.2-6. https://CRAN.R-project.org/package=gamm4.

Wood S. N. (2017). *Generalized additive models: An introduction with R* (2nd ed.), Chapman & Hall/CRC Texts in Statistical Science. CRC Press.

Yee T. W. (2015). *Vector generalized linear and additive models: With an implementation in R*. Springer.

Zweifel R., Sterck F., Braun S., Buchmann N., Eugster W., Gessler A., Häni M., Peters R. L., Walthert L., Wilhelm M., Ziemińska K., & Etzold S. (2021). Why trees grow at night. *New Phytologist*, *231*(6), 2174–2185. https://doi.org/10.1111/nph.17552

Zweifel R., Zimmermann L., & Newbery D. M. (2005). Modeling tree water deficit from microclimate: an approach to quantifying drought stress. *Tree Physiology*, *25*(2), 147–156. https://doi.org/10.1093/treephys/25.2.147

Modeling Intra-Annual Tree Stem Growth – Supplement

Hannes Riebl

Chair of Statistics, Georg-August-Universität Göttingen, Germany

E-mail: hriebl@uni-goettingen.de

Nadja Klein

Chair of Statistics and Data Science, Humboldt-Universität zu Berlin, Germany

E-mail: nadja.klein@hu-berlin.de

Thomas Kneib

Chair of Statistics, Georg-August-Universität Göttingen, Germany

E-mail: tkneib@uni-goettingen.de

Summary. Supplementary material on the misspecification of the smoothness parameter of the Matérn correlation function and the performance of the different sampling schemes discussed in the article "Modeling Intra-Annual Tree Stem Growth with a Distributional Regression Approach for Gaussian Process Responses".

Keywords: Generalized additive model for location, scale, and shape; growth curve model; Markov chain Monte Carlo simulation; Matérn covariance function; spatio-temporal regression; structured additive predictor

1. Misspecification of the smoothness parameter of the Matérn correlation function

Throughout the paper, we use the Matérn correlation function with a fixed smoothness parameter $\nu = 0.5$, 1.5 or 2.5. In spatial statistics, it is common practice to fix the parameter ν to a half-integer value, as the correlation function, which generally involves the gamma function and the Bessel function of the second kind, reduces to a product of a polynomial and an exponential in these cases. Treating ν as a continuous parameter is computationally challenging, and the parameter is often difficult to identify in practice. Typically, ν is chosen based on assumptions about the smoothness of the Gaussian process (GP), since it is intrinsically connected to the differentiability of the GP. Another aspect to consider is the mathematical validity of the correlation function on a given metric space, e.g. the Matérn correlation function is only valid for $0 < \nu \leq 0.5$ on the one to three-dimensional sphere (Section 2.2.2 of the paper).

1.1. Simulation study

To investigate the effect of a misspecified smoothness parameter ν on the inference, we estimated the model from Scenario II of the simulation study (Section 4.2 of the paper) with the wrong parameters $\nu = 0.5$ and 2.5 (the true parameter is $\nu = 1.5$). The effect of the model misspecification on the bias of the posterior mean is shown in Figure 1 and 2. As expected, the strongest bias can be observed for the covariance parameters, particularly for the range parameter ϕ.

For half-integer values of the parameter ν, we parametrize the Matérn correlation function as

$$\begin{aligned}\rho(\tilde{d}, \nu = 0.5) &= \exp(-\tilde{d}),\\ \rho(\tilde{d}, \nu = 1.5) &= (1 + \tilde{d}) \exp(-\tilde{d}),\\ \rho(\tilde{d}, \nu = 2.5) &= (1 + \tilde{d} + \tilde{d}^2/3) \exp(-\tilde{d}),\end{aligned}$$

where $\tilde{d} = d/\phi$ and d is the distance between two observations. The parameter ϕ can be compared between the different correlation functions in the sense that it scales the distance that is used in the correlation functions, but the estimates cannot be expected to match each other perfectly given the different terms in the correlation functions.

The range parameter can also be interpreted as a smoothness measure of the GP – not in the sense of differentiability but describing the persistence of the deviations from the mean. A GP with a smaller parameter ϕ has less persistent deviations from the mean, i.e. the GP is more wiggly (Figure 2 in the paper). For this reason, it is not surprising that the parameter ϕ seems to compensate for the misspecification of the smoothness parameter: In Figure 1, where the smoothness parameter is too small, the parameter ϕ is estimated to be larger than for the correct

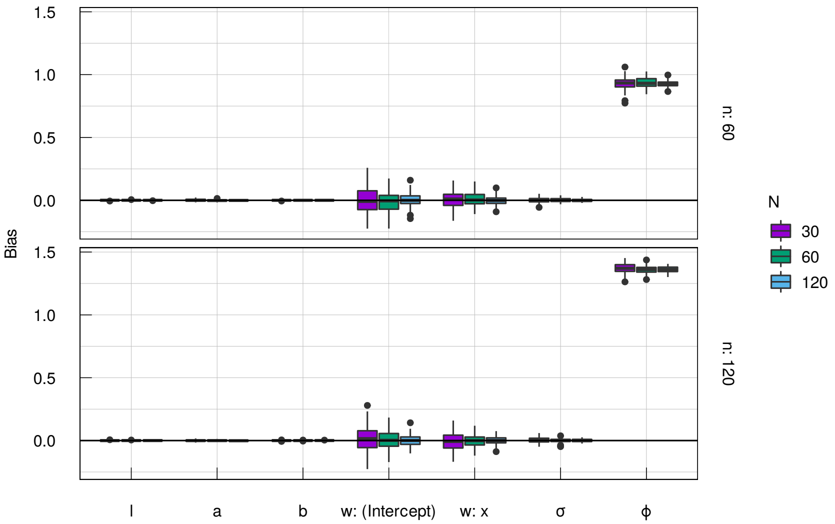


Fig. 1. The bias of the posterior mean in Simulation Scenario II estimated with the misspecified **smoothness parameter** $\boldsymbol{\nu = 0.5}$ (instead of the true $\nu = 1.5$). Each boxplot summarizes 100 replications for one combination of N (the number of GPs) and n (the number of observations per GP).

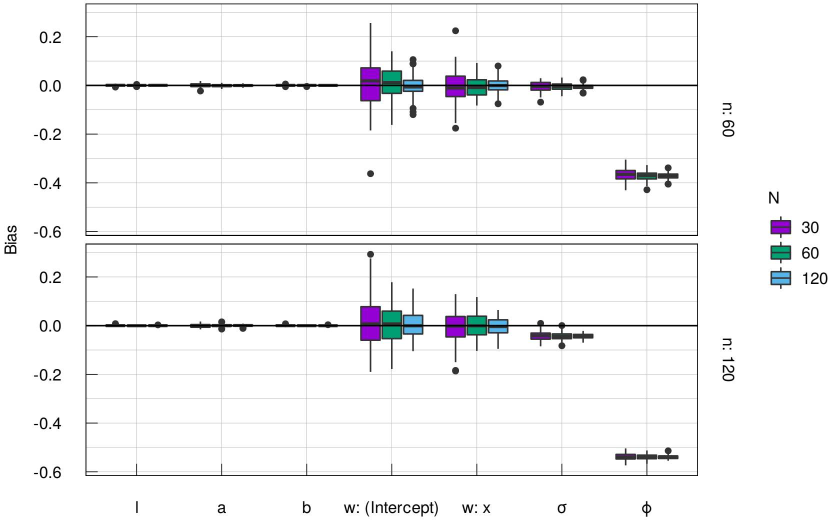


Fig. 2. The bias of the posterior mean in Simulation Scenario II estimated with the misspecified **smoothness parameter** $\boldsymbol{\nu = 2.5}$ (instead of the true $\nu = 1.5$). Each boxplot summarizes 100 replications for one combination of N (the number of GPs) and n (the number of observations per GP).

smoothness parameter, hence reducing the wiggliness of the GP. On the other hand, the opposite behavior can be observed in Figure 2.

Another conclusion from the simulation study is that the parameters other than the range are barely affected by a misspecified smoothness parameter: No bias was observed for the mean parameters, and only a very small bias for the standard deviation in some configurations of the simulation study. In the context of distributional regression with GP responses, which we focus on in the paper, it is also more relevant to consider the effect of a misspecified smoothness parameter on the estimated covariate effects rather than the intercepts. This last aspect is discussed in the next section for the tree stem growth model from Section 5 of the paper.

1.2. Intra-annual tree stem growth

To check the robustness of the results from the tree stem growth model from Section 5 of the paper, we also estimated the same model with the smoothness parameters $\nu = 0.5$ and 2.5. We found that all three models produce very similar results, both in terms of the model fit and the estimated covariate effects.

To compare the model fit, we used the deviance information criterion (DIC), which is shown in Table 1. Generally, the deviance $D(\bar{\theta})$ is lower for smaller smoothness parameters, which seems plausible because larger smoothness parameters express more restrictive assumptions about the smoothness of the GP. At the same time, the effective number of parameters p_D is comparable for all three models, yielding the lowest DIC for the model with $\nu = 0.5$, followed by $\nu = 1.5$ and $\nu = 2.5$.

Table 1: The **DIC of the tree stem growth model** from Section 5 of the paper for different smoothness parameters ν. The DIC is defined as DIC $= D(\bar{\theta}) + 2p_D$, where $D(\bar{\theta})$ is the deviance at the posterior mean of the parameters, and p_D is the effective number of parameters. The results for $\nu = 0.5$ are in parentheses, because some parameters appear to be underidentified in this case, deteriorating the mixing of the MCMC chains.

	$D(\bar{\theta})$	p_D	DIC
$\nu = 0.5$	(209187.4)	(267.351)	(209722.1)
$\nu = 1.5$	210850.9	276.261	211403.5
$\nu = 2.5$	216793.3	275.828	217345.0

To estimate the model with $\nu = 0.5$, however, we had to tighten the priors of the regression coefficients for the limit and scale parameter of the mean function from an uninformative prior to a standard normal prior. Even with the standard normal prior, we still did not achieve a satisfactory mixing of the MCMC chains for the model with $\nu = 0.5$. As larger smoothness parameters can stabilize the estimation of distributional regression models with GP responses, we propose the following approach to model selection: (1) Estimate the model with a number of different smoothness parameters. (2) Among the smoothness parameters that are large enough for a stable estimation of the model, choose the one with the best value of your preferred model selection criterion. In our case, this is the tree stem growth model with $\nu = 1.5$, which is presented in the paper.

Table 2 shows some of the estimated covariate effects from the tree stem growth model with $\nu = 2.5$, comparing them to the model from the paper with $\nu = 1.5$. The results are almost identical for the mean parameters and very similar for the covariance parameters. In particular, the sign pattern of the estimated covariate effects remains unchanged. The most notable difference between the two models is the estimated intercept for the range: For $\nu = 2.5$, the posterior mean is 0.119, instead of 0.875 for $\nu = 1.5$. The differences from the model with $\nu = 0.5$ are more pronounced: The posterior mean of the intercept for the range is 3.85 for $\nu = 0.5$, and the size of some estimated covariate effects also changes. Nonetheless, the sign pattern is again the same, except for the species effect of ash on the range, where the posterior mean flips from negative to positive.

Table 2: Summary statistics of the posterior samples of the species effect of ash (vs. beech) and the effect of DBH on the predictors of the tree stem growth model from Section 5 of the paper with the **smoothness parameter $\boldsymbol{\nu = 2.5}$**. For comparison, the summary statistics for $\nu = 1.5$ are reported in parentheses.

Ash	Coefficient	Mean	2.5%	Median	97.5%
Shape	$\beta_{a1,1}$	-0.285	-0.341	-0.285	-0.228
		(-0.289)	(-0.363)	(-0.289)	(-0.218)
Scale	$\beta_{b1,1}$	-0.430	-0.450	-0.429	-0.409
		(-0.426)	(-0.452)	(-0.426)	(-0.400)
Std. dev.	$\beta_{\sigma1,1}$	0.636	0.573	0.635	0.699
		(0.602)	(0.529)	(0.602)	(0.678)
Range	$\beta_{\phi1,1}$	-0.058	-0.091	-0.058	-0.025
		(-0.101)	(-0.157)	(-0.101)	(-0.043)
DBH	Coefficient	Mean	2.5%	Median	97.5%
Shape	β_{a2}	-0.027	-0.040	-0.027	-0.013
		(-0.027)	(-0.044)	(-0.027)	(-0.009)
Scale	β_{b2}	0.050	0.046	0.050	0.054
		(0.050)	(0.045)	(0.050)	(0.055)
Std. dev.	$\beta_{\sigma2}$	0.101	0.084	0.101	0.119
		(0.085)	(0.064)	(0.085)	(0.106)
Range	$\beta_{\phi2}$	0.010	0.001	0.010	0.018
		(0.011)	(-0.004)	(0.011)	(0.027)

2. Comparison of the different sampling schemes for all simulation scenarios

In Simulation Scenario I, we compare the performance of two different MCMC algorithms for distributional regression with GP responses: The conventional sampling scheme uses separate parameter blocks for each distributional parameter and covariate effect, while we propose to sample the intercepts and covariate effects of the standard deviation and range in one joint parameter block. We find that separate blocks for the standard deviation and range can lead to MCMC chains with a very high autocorrelation, and that the performance can be improved substantially by blocking these parameters together.

Figure 3 and 4 show exemplary trace plots for Scenario II and III to verify that the same pattern can also be observed in these cases. For Scenario III (the GP on the sphere, Figure 4), the performance difference is very apparent from the trace plots for the unmodified configuration from the paper. The autocorrelation of the MCMC chains produced by the joint sampler is substantially lower.

For Scenario II (the GP that resembles the tree stem growth model, Figure 3), the green trace plots correspond to the configuration from the paper. In this case, the true range parameter is $\phi = 2$, and the trace plots show no clear performance difference. As discussed in the paper, the separate parameter blocks are particularly prone to produce samples with a high autocorrelation if the range is large. To illustrate this in Scenario II, we ran the same scenario but with the true range parameter set to $\phi = 20$, giving rise to the expected performance difference shown in the purple trace plots.

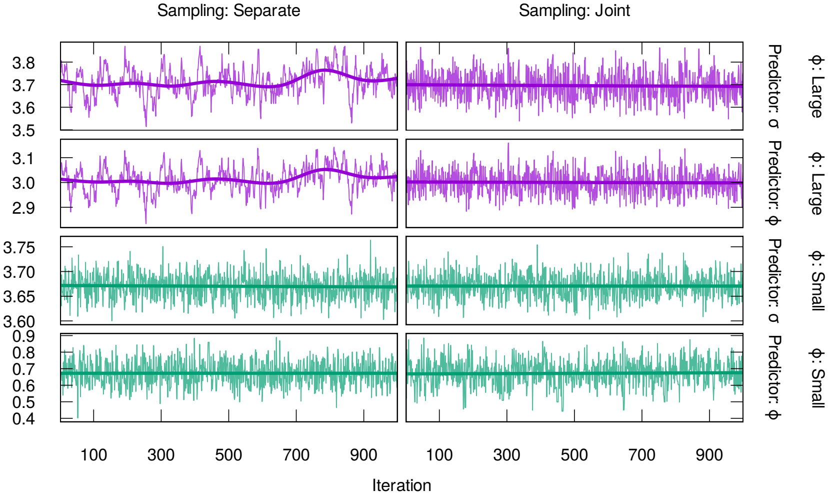


Fig. 3. Trace plots for the standard deviation and range from one exemplary replication of **Simulation Scenario II**. The left-hand side shows the sampler with separate blocks for the covariance parameters, while the right-hand side shows the sampler with one joint block for these coefficients. No thinning was applied to the chains.

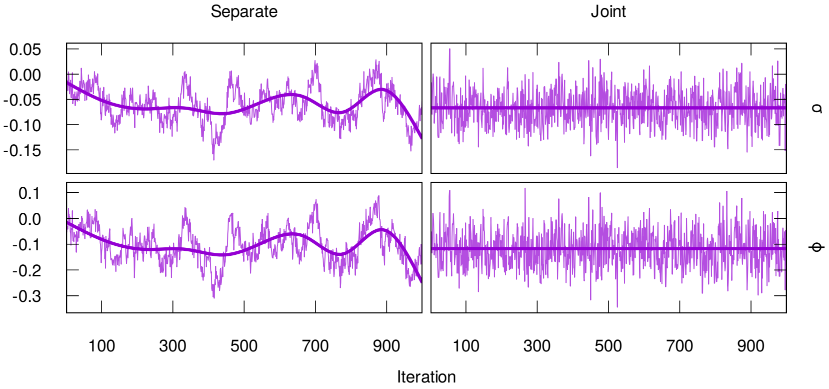


Fig. 4. Trace plots for the standard deviation and range from one exemplary replication of **Simulation Scenario III**. The left-hand side shows the sampler with separate blocks for the covariance parameters, while the right-hand side shows the sampler with one joint block for these coefficients. No thinning was applied to the chains.

Appendix C

A Structured Additive Multi-Species Count Model for Assessing the Relation Between Site Conditions and Species Diversity

(With Supplement)

A Structured Additive Multi-Species Count Model for Assessing the Relation Between Site Conditions and Species Diversity

A Preprint

Hannes Riebl[*]
Chair of Statistics
University of Göttingen
Humboldtallee 3, 37073 Göttingen, Germany
hriebl@uni-goettingen.de

Jonas Glatthorn
Bestandesdynamik und Waldbau
Eidgenössische Forschungsanstalt WSL
Zürcherstrasse 111, 8903 Birmensdorf, Switzerland
jonas.glatthorn@wsl.ch

Thomas Kneib
Chair of Statistics
University of Göttingen
Humboldtallee 3, 37073 Göttingen, Germany
tkneib@uni-goettingen.de

Abstract

We propose the multi-species count model (MSCM), a semi-parametric regression model with a novel response structure, to assess the relationship between site conditions and species diversity. The model can be applied to a broad range of problems, including the analysis of different species diversity indices and taxa. It belongs to the class of Bayesian hierarchical models, allowing us to incorporate structured additive predictors with linear, non-linear, random and spatial effects for the site conditions. The connections with several related model classes such as zero-inflated Poisson regression and multi-species occupancy models are discussed in the article, as well as a number of interesting model extensions and generalizations. We describe a robust and efficient MCMC algorithm to perform fully Bayesian inference, which we implement in Python using the probabilistic programming framework Liesel. The performance of the algorithm is illustrated in a simulation study, where we also pinpoint problematic parameter constellations in which the estimates are not well-identified. Finally, we apply the MSCM to data from the Research Training Group (RTG) 2300, a large-scale ecological research project conducted in Lower Saxony, northwest Germany, where we investigate the impact of admixing Norway spruce and Douglas fir in European beech forests, accounting for the spatial correlation of the field sites of the RTG.

Keywords Generalized additive model for location, scale, and shape · Multi-species occupancy model · Markov chain Monte Carlo simulation · Zero-inflated data · Spatial regression · Structured additive predictor

[*]The work of Hannes Riebl on this article was funded by the German Research Foundation (DFG) through the Research Training Group 2300 “Enrichment of European Beech Forests with Conifers” (316045089/GRK2300) and the grant “LIESEL – A Software Framework for Bayesian Semi-Parametric Distributional Regression” (KN 922/11-1).

1 Introduction

Loss of biodiversity due to human overuse of natural resources is one of the most pressing environmental issues of our time, as emphasized by the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES) in its global assessment report from 2019. It directly impacts the integrity of ecosystems, and hence human well-being. As such, there is an urgent need for robust and effective statistical models to assess biodiversity and its drivers on local, regional and global scales. To address these questions, researchers typically collect data on the abundance of a variety of species at different field sites, together with environmental variables such as temperature, precipitation and land-use type. Analyzing such data can be a challenging task, however, as non-linear relationships and complex interactions need to be taken into account.

In large-scale ecological research projects, the assessment of biodiversity across different taxa at multiple field sites is often of primary interest. One approach to perform this analysis is to estimate the species diversity for each taxon and field site separately, based on the observed abundances, and then relate these diversity estimates to site-specific covariates using a regression model (Glatthorn et al., 2023). This two-step procedure has the drawback, however, that it is difficult to take the uncertainty of the diversity estimates into account in the regression model. Alternatively, a more comprehensive modeling approach is to use multi-species occupancy models (MSOMs, MacKenzie et al., 2004; Dorazio and Royle, 2005), which are a class of models that estimate the occurrence, abundance and detection probabilities of various species simultaneously, possibly allowing for interactions between them. As MSOMs involve the estimation of detection probabilities, they require repeated surveys at each field site. In meta-studies, this level of detailed data is often not available for all relevant taxa.

To address these difficulties and to provide a common model for different taxa that are monitored in a research project, we propose the novel multi-species count model (MSCM) that can be used to estimate various species diversity indices. The MSCM is formulated as a Bayesian hierarchical model, allowing us to integrate structured additive predictors with linear, non-linear, random and spatial effects of the environmental conditions at the field sites. The MSCM has a few advantages over MSOMs and other existing model classes: First, it offers a good compromise between the simplicity of a two-step analysis and the high data requirements of MSOMs, making it a suitable tool for meta-studies. Second, the formulation of the model as a directed acyclic graph (DAG) makes it straightforward to include derived quantities, i.e. quantities that are computed from other variables in the model graph such as species diversity indices. Finally, considering the model in a Bayesian context straightforwardly allows us to compare different model specifications and assess the reliability of predictions.

We perform fully Bayesian inference using a Markov chain Monte Carlo (MCMC) algorithm, allowing us to estimate the parameters of the multi-species count model and to make predictions in a flexible and computationally efficient way. The model and inference scheme are implemented using Liesel, a probabilistic programming framework, and the corresponding MCMC library Goose (Riebl et al., 2022). The software is based on the high-performance machine learning library JAX (Bradbury et al., 2023) for Python. The robustness of the inference scheme is evaluated in an extensive simulation study. Some data-generating processes (DGPs) under which the model parameters are difficult to identify are also discussed.

Finally, we apply the MSCM to assess the species diversity of different taxa in mixed forest stands in Lower Saxony, demonstrating the flexibility of our model with a complex structured additive predictor combining parametric and non-parametric covariate effects. This way, we find that the more favorable environmental conditions at the field sites in southern Lower Saxony are reflected in a higher community diversity across all taxa. Furthermore, the diversity of the vegetation and small mammals tends to increase with the proportion of coniferous tree species at a given site.

The remainder of this article is organized as follows: Section 2 defines the multi-species count model, including a description of structured additive predictors and the species diversity indices that can be computed from the model. Section 3 explores the connections between the MSCM and other established model classes such as zero-inflated Poisson regression and multi-species occupancy models. Section 4 provides details on our fully Bayesian inference scheme, followed by a presentation of our simulation study with three scenarios in Section 5. In Section 6, we apply the MSCM to assess the species diversity of different taxa in mixed forest stands in Lower Saxony, before providing concluding remarks in Section 7.

2 The structured additive multi-species count model

The multi-species count model can be used with a response matrix $\mathbf{Y}$, where the entry in the i-th row and the j-th column describes how often species $j = 1, \dots, M$ was observed on experimental plot $i = 1, \dots, N$ (or more generally, on observation unit i). Moreover, the covariate vectors $\boldsymbol{x}_i$ contain information on the plots such as their geographic location, the composition of tree species or the climate. Given this sort of data, we define the multi-species count model as the following Bayesian hierarchical model with

$$\begin{aligned}
\text{the occupancy intercept of species } j, \quad & \gamma_j \sim \text{Normal}(0, 10),\\
\text{the probability that species } j \text{ occupies plot } i, \quad & \psi_{ij} = \text{InvLogit}(\gamma_j + \eta_i),\\
\text{the unobserved indicator whether species } j \text{ occupies plot } i, \quad & z_{ij} \sim \text{Bernoulli}(\psi_{ij}),\\
\text{the expected abundance of species } j, \quad & \mu_j \sim \text{HalfNormal}(0, 10),\\
\text{the total number of observations on plot } i, \quad & n_i \sim \text{CountDistribution}(z_{ij}, \mu_j),\\
\text{the relative expected abundances per species on plot } i, \quad & \boldsymbol{p}_i = \frac{1}{\sum_{j=1}^{M} z_{ij}\mu_j} \times (z_{i1}\mu_1,\ \ldots,\ z_{iM}\mu_M),\\
\text{the number of observations per species on plot } i, \quad & \boldsymbol{y}_i \sim \text{Multinomial}(n_i, \boldsymbol{p}_i).
\end{aligned}$$

Here, the η_i are the structured additive predictors for the plots combining different covariate effects computed from the covariate vectors $\boldsymbol{x}_i$ (Section 2.1). In the standard case, $\text{CountDistribution}(z_{ij}, \mu_j) = \text{Poisson}(\sum_j \lambda_{ij})$, where $\lambda_{ij} = z_{ij}\mu_j$, but other count distributions can be used to account for specific properties of the data, e.g. the negative binomial distribution in the case of overdispersion or the Yule distribution for heavy-tailed data. Section 6 presents an application where an MSCM is estimated with three different count distributions. A comparison of the models using the widely applicable information criterion (WAIC) suggests that count distributions other than the Poisson distribution often result in a better model fit. Figure 1 shows the graph of the MSCM from the application including a structured additive predictor and two species diversity indices that are derived from the model (Section 2.2).

2.1 Structured additive predictors

The structured additive predictor η_i combines parametric covariate effects $\boldsymbol{x}_{i1}'\boldsymbol{\beta}_1$ and non-parametric covariate effects $f_k(\boldsymbol{x}_{ik}, \boldsymbol{\beta}_k)$, for $k = 1, \ldots, K$, i.e.

$$\eta_i = \beta_0 + \boldsymbol{x}_{i1}'\boldsymbol{\beta}_1 + \sum_{k=2}^{K} f_k(\boldsymbol{x}_{ik}, \boldsymbol{\beta}_k),$$

where β_0 is the intercept and the non-parametric effects f_k are centered around zero (Fahrmeir et al., 2004; Wood, 2017, Chapter 4). In the MSCM, the global intercept β_0 is omitted in favor of the species-specific occupancy intercepts γ_j to ensure identifiability. The functions f_k are usually modeled as linear basis expansions of the covariate vectors $\boldsymbol{x}_{ik}$ (Hastie et al., 2009, Chapter 5). Depending on the choice of the basis and the prior of the regression coefficients $\boldsymbol{\beta}_k$, the functions can represent non-linear, random and spatial effects, among others.

To avoid overfitting, certain smoothness properties of the non-parametric effects can be enforced through regularization, e.g. using P-splines that penalize (second) differences between coefficients of neighboring B-splines (Eilers and Marx, 1996; Lang and Brezger, 2004). In a Bayesian context, regularization is accomplished using informative priors, e.g. the multivariate normal prior

$$p(\boldsymbol{\beta} \mid \tau^2) \propto \tau^{-\text{rk}(\mathbf{K})} \exp(-0.5\tau^{-2}\boldsymbol{\beta}'\mathbf{K}\boldsymbol{\beta}),$$

where τ^2 is the variance (or smoothing) parameter, $\mathbf{K}$ is a (potentially rank-deficient) penalty matrix, and the index k is omitted for better readability. For P-splines, the penalty matrix is given by $\mathbf{K} = \mathbf{D}_2'\mathbf{D}_2$, where $\mathbf{D}_2$ is the second-order difference matrix, such that $\mathbf{D}_2\boldsymbol{\beta} = (\beta_1 - 2\beta_2 + \beta_3,\ \beta_2 - 2\beta_3 + \beta_4,\ \ldots)$. In this case, the penalty matrix is, in fact, rank-deficient.

Other common effect types include random effects, where the penalty matrix reduces to $\mathbf{K} = \mathbf{I}$, (intrinsic) Gaussian Markov random field, where $\mathbf{K}$ is determined by the underlying neighborhood structure (Rue and Held, 2005), and other spatial effects, where Vecchia approximations can be used to construct the penalty matrix (Katzfuss and Guinness, 2021). Moreover, note that the linear effect $\boldsymbol{x}_{i1}'\boldsymbol{\beta}_1$ can also be embedded in this framework by setting the penalty matrix to $\mathbf{K} = \mathbf{0}$, which results in a flat prior. Therefore, parametric and non-parametric effects are two sides of the same coin, and are sometimes generically referred to as predictor components or smooth terms.

For the smoothing parameter τ^2, it is common to assume a weakly informative hyperprior. Lang and Brezger (2004) propose the conjugate inverse gamma prior with hyperparameters $a = b = 0.01$, or some other small value, which enables direct Gibbs updates by sampling from the full conditional. Practically, however, other priors such as the half-Cauchy distribution or the half-normal distribution may have better statistical properties (Gelman, 2006; Klein and Kneib, 2016a). For a comprehensive treatment of structured additive predictors, refer to Fahrmeir et al. (2013), specifically Chapter 8 and 9.

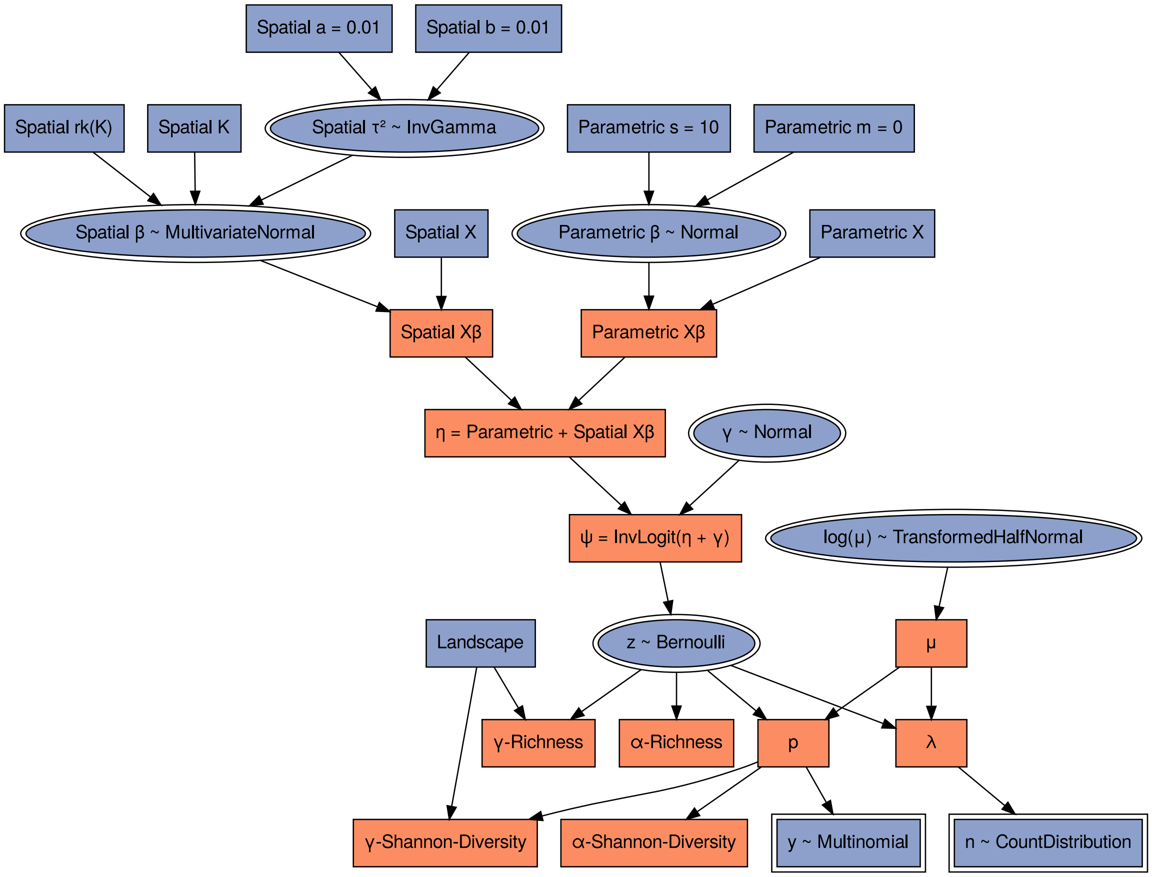


Figure 1: The graph of the multi-species count model from the application in Section 6, implemented using the Liesel probabilistic programming framework. Blue variables are strong (constant or sampled), orange variables are weak (computed from other variables). A double contour line indicates an associated probability distribution, and a round shape indicates a model parameter. The lower part of the graph represents the response structure of the MSCM, where n follows an arbitrary count distribution such as the Poisson, negative binomial or Yule distribution. The structured additive predictor η is composed of parametric and non-parametric covariate effects. In this case, the non-parametric effect is a Gaussian process (GP) representing a spatial effect. From the response structure of the MSCM, various diversity indices such as the species richness and the Shannon index can be derived.

2.2 Species diversity indices

From the multi-species count model, different biodiversity measures can be derived and estimated simultaneously with an MCMC algorithm based on one itegrated model. Among the most common biodiversity measures are the species richness and the Shannon index, both of which we assess on a plot and landscape-level in the application in Section 6.

The species richness R_i of experimental plot i is the total number of species that are present on that plot, i.e.

$$R_i = \sum_{j=1}^{M} z_{ij}.$$

As the z_{ij}, the indicators whether species j is present at plot i, are unobserved and sampled with an MCMC algorithm, the species richness R_i is another random variable, whose posterior distribution can be assessed from the MCMC samples. Generally speaking, species richness is a simple biodiversity measure that is easy to compute and interpret. A high species richness can be indicative of a healthy and diverse ecosystem, but as it does not take the abundances of the species into account, it can only provide an incomplete picture of biodiversity (Hill, 1973; Colwell, 2009).

The Shannon index H_i', on the other hand, does take into account both the number of species and their abundances. It is computed as

$$H_i' = -\sum_{j=1}^{M} p_{ij} \log(p_{ij}),$$

where, by definition, some p_{ij} may become zero, in which case we define $p_{ij} \log(p_{ij}) = 0$. The Shannon index H_i' ranges from 0 to $\log(M)$, where M is the total number of species, and provides a more complex and comprehensive biodiversity measure than the species richness R_i (Shannon, 1948). A higher Shannon index signals a greater complexity and diversity of an ecosystem.

Regardless of the specific index, species diversity can be considered on different spatial scales. The concept of α-diversity is a measure of local biodiversity, referring to a single habitat or small area, e.g. a 50×50 m^2 plot in the application in Section 6. In contrast, the concept of γ-diversity is defined for a larger geographic area or region, e.g. northern or southern Lower Saxony. It is a measure of regional biodiversity and reflects the diversity of species across multiple habitats or ecosystems (Whittaker, 1960; Whittaker et al., 2001).

Figure 1 shows how the species richness and the Shannon index on a plot and landscape-level can be integrated into the MSCM model graph. The landscapes are defined via an $L \times N$ binary selection matrix $\mathbf{S}$, where the entry in the l-th row and the i-th column is one if plot i belongs to landscape l and zero otherwise. Based on the matrix $\mathbf{S}$, the γ-richness can be computed as the number of non-zero entries per row of the matrix $\mathbf{T} = \mathbf{SZ}$, where $\mathbf{Z}$ is the $N \times M$ matrix containing the indicators z_{ij}. The γ-Shannon index can be computed in an analogous way.

3 Related work

The multi-species count model is closely related to several well-known statistical and ecological model classes such as zero-inflated Poisson regression, structured additive distributional regression and multi-species occupancy models, which we discuss in more detail in this section.

3.1 Zero-inflated Poisson regression

Zero-inflated Poisson regression is a statistical modeling technique that can be used to analyze count data with an excess of zeros (Cameron and Trivedi, 2013). Mathematically, zero-inflated Poisson regression is defined as a mixture model combining a point mass at zero and a Poisson regression model. Hence, there are two possible sources of the observed zeros: the point mass and the Poisson regression model. To model the unobserved source of the observations, latent dichotomous variables are introduced, which again can be related to covariates and modeled as outcomes of another binary regression model.

Zero-inflated count data models originally were proposed by Mullahy (1986), together with hurdle models, another type of mixture models where zeros come from a point mass at zero as the only possible source. Lambert (1992) was the first to prominently apply zero-inflated Poisson regression to model defects in manufacturing. In addition to the covariates of the Poisson regression model, they use a logit regression model with covariates for the latent dichotomous variables describing whether the manufacturing equipment is properly aligned or not. With properly aligned equipment, defects are almost impossible, while they follow a Poisson distribution when the equipment is misaligned.

The data-generating process of the MSCM naturally falls into the category of zero-inflated count data models, as there are two possible sources of zeros: A zero occurs for sure if a plot is not occupied by a species, but it may also occur because a species is not detected despite being present at a plot. While Mullahy (1986) notes that zero-inflated and hurdle models are equivalent if no covariates are used, this is not the case for the MSCM, where covariates are used for the occupancy probabilities.

More formally, it can be shown that the response structure of the MSCM with CountDistribution(z_j, μ_j) = Poisson$(\sum \lambda_j)$, for $\lambda_j = z_j \mu_j$ and the species index $j = 1, \dots, M$, is, in fact, equivalent to zero-inflated Poisson regression. For one experimental plot and hence omitting the plot index $i = 1, \dots, N$, the joint probability of the observed responses and the latent occupancies of the MSCM is given by

$$\begin{aligned} p(\boldsymbol{y}, n, \boldsymbol{z}) &= p(\boldsymbol{y} \mid n, \boldsymbol{z}) \times p(n \mid \boldsymbol{z}) \times p(\boldsymbol{z}) \\ &= \frac{n!}{y_1! \dots y_M!} \left(\frac{\lambda_1}{\sum \lambda_j} \right)^{y_1} \dots \left(\frac{\lambda_M}{\sum \lambda_j} \right)^{y_M} \times \frac{(\sum \lambda_j)^n}{n!} e^{-\sum \lambda_j} \times \psi_1^{z_1} (1 - \psi_1)^{1 - z_1} \dots \psi_M^{z_M} (1 - \psi_M)^{1 - z_M} \\ &= \psi_1^{z_1} (1 - \psi_1)^{1 - z_1} \frac{\lambda_1^{y_1}}{y_1!} e^{-\lambda_1} \times \dots \times \psi_M^{z_M} (1 - \psi_M)^{1 - z_M} \frac{\lambda_M^{y_M}}{y_M!} e^{-\lambda_M} \times \cancel{\frac{n! (\sum \lambda_j)^n}{n! (\sum \lambda_j)^{y_1} \dots (\sum \lambda_j)^{y_M}}}. \end{aligned}$$

This is the product of the probabilities of M zero-inflated Poisson variables y_j with the latent dichotomous variables z_j. As $z_j = 0 \implies y_j = \lambda_j = 0$ and $y_j = 1 \implies z_j = 0$, marginalizing out z_j yields the standard probability mass function of the zero-inflated Poisson distribution, i.e.

$$\psi_j^{z_j} (1 - \psi_j)^{1 - z_j} \frac{\lambda_j^{y_j}}{y_j!} e^{-\lambda_j} = \begin{cases} (1 - \psi_j) + \psi_j e^{-\lambda_j} & \text{if } y_j = 0, \\ \psi_j \frac{\lambda_j^{y_j}}{y_j!} e^{-\lambda_j} & \text{if } y_j > 0. \end{cases}$$

Note that the equivalence depends on how the terms of the multinomial and the Poisson distribution can be rearranged. Similar operations are not possible for other count distributions such as the negative binomial or the Yule distribution, so that the equivalence does not hold in those cases.

3.2 Structured additive distributional regression

Zero-inflated Poisson regression and the multi-species count model belong to the so-called distributional regression model class. These models are also known as generalized additive models for location, scale and shape (GAMLSS, Rigby and Stasinopoulos, 2005). In contrast to standard generalized linear models (GLMs, Nelder and Wedderburn, 1972), the response distribution of GAMLSS is not limited to the exponential family but can come from any parametric family. While the acronym GAMLSS suggests that the first three moments of the response distribution are related to covariates, this is not generally the case. For example, the two parameters of the zero-inflated Poisson distribution that are related to covariates in a regression context are the rate and zero-inflation parameter, neither of which directly represents the mean or variance of the distribution. Usually in GAMLSS, the relationship between response and explanatory variables is modeled using multiple structured additive predictors (Section 2.1). If a parameter of the response distribution is constrained, e.g. to $(0, \infty)$ or $(0, 1)$, it needs to be transformed to $\mathbb{R}$ before being linked to a structured additive predictor.

Generally, GAMLSS have the capability to accommodate a broad range of response distributions, including discrete, continuous and mixed distributions (Rigby et al., 2019). The use of various count data distributions beyond the zero-inflated Poisson distribution within the GAMLSS framework is discussed by Klein et al. (2015b). For non-negative continuous data, distributions like the Pareto or Weibull distribution may be a fitting choice. Regarding fractional responses, such as single or multiple percentages, Klein et al. (2015a) consider the beta and Dirichlet distributions. Finally, the GAMLSS framework can also be employed to study multivariate responses, using either conventional multivariate distributions (Michaelis et al., 2018) or copulas to describe complex dependence structures with arbitrary marginal distributions (Klein and Kneib, 2016b).

In a recent review paper, Stasinopoulos et al. (2018) provide an overview of the scientific fields where GAMLSS have been applied since their introduction by Rigby and Stasinopoulos in 2005. The fields include biology (Hawkins et al., 2013), economics (Voudouris et al., 2015), environmental science (Villarini et al., 2009), genomics (Khondoker et al., 2007), management science (Budge et al., 2010) and medicine (Rodrigues et al., 2009). A Bayesian workflow for GAMLSS is described by Umlauf and Kneib (2018) together with an application on German weather data.

3.3 Multi-species occupancy models

As the name suggests, the multi-species count model is also closely linked to multi-species occupancy models, a widely-used method in ecology for estimating the probabilities of occurrence and detection of multiple species at the same time. Multi-species occupancy models (MSOMs) were first introduced by MacKenzie et al. (2004) and Dorazio and Royle (2005) as an extension of single-species occupancy models, and they have gained significant popularity in recent years. MSOMs are commonly used to study the composition and diversity of species communities, as the joint analysis of data on multiple species is often more effective and informative than modeling each species separately (Devarajan et al., 2020).

As MSOMs involve the estimation of detection probabilities, they require data from multiple surveys per field site (Devarajan et al., 2020). This is an essential difference between MSOMs and our model, which relies on a single count per species and field site and hence does not require repeated surveys. As the data requirements of our model are relatively low, it is a good tool for meta-studies on various taxa, even if the species are sampled according to different protocols for each taxon. On the other hand, our model is not designed to disentangle if differences in the recorded counts are due to differences in the abundances or the detection probabilities of the species. Hence, when species diversity indices are computed from our model, the implicit assumption is that the counts are proportional to the abundances, i.e. that the detection probability is constant across all species.

Some variants of MSOMs can take biotic interactions between species into account. Correlations between occurrence probabilities of different species are often a result of shared habitat requirements and similar responses to relevant environmental factors. By including such dependencies in the model, the accuracy of the occupancy estimates can be improved, and the mechanisms driving co-occurrence patterns can be better understood. Specific MSOMs with a focus on species co-occurrence and biotic interactions have been developed by MacKenzie et al. (2004), Waddle et al. (2010) and Rota et al. (2016).

In the form presented in Section 2, our model cannot take biotic interactions between species into account. However, it would be straightforward to equip the model with this feature, e.g. by introducing a multivariate normal prior for the species-specific occupancy intercepts γ_j and the expected abundances λ_j. This way, a suitable correlation structure between the species could be enforced. The researcher could either fix or estimate the correlations, depending on the specific parameterization of the model. One drawback of estimating the correlations would be an increased number of parameters, which could potentially result in identification issues and increase the computational cost of the model. If the correlations were specified in the most naive way, the number of parameters would increase quadratically with the number of species, i.e. sparse parameterizations would become necessary for more than four or five species. In fact, most MSOM variants with a focus on co-occurrence patterns are limited to a relatively small number of species (Devarajan et al., 2020).

Some MSOMs can also be used to estimate the size of an unobserved meta-community. For this purpose, Dorazio and Royle (2005) propose a parameter-expanded data augmentation technique for MCMC inference, where some all-zero columns representing potentially unobserved species are added to the response matrix $\mathbf{Y}$. The size of the meta-community is then assessed by estimating the number of extra columns (Kéry and Schaub, 2012). Our model as described in this article lacks the ability to quantify the size of the meta-community, i.e. all relevant species must be added to the response matrix $\mathbf{Y}$ by the researcher.

Due to the high degree of flexibility in terms of the model specification, presenting MSOMs concisely can be a challenging task for researchers. Devarajan et al. (2020) provide a review of 92 studies using MSOMs that were published between 2009 and 2018, spanning 27 countries and various taxa. They observe a consistent pattern of underreporting on aspects as diverse as the spatial and temporal scope of the data, the field methods and the type of detectors, as well as the covariates and the statistical tools. The insufficient reporting undermines the robustness of the inferences and the reproducibility of the studies, and could potentially have an adverse effect on conservation and management efforts.

Many studies using MSOMs also lack an explicit discussion of the model assumptions. Devarajan et al. (2020) note, for example, that monitoring is usually geared towards one focal species, and other species are only recorded as bycatch. If MSOMs are used with such data, the assumptions are unlikely to transfer seamlessly between the focal and the bycatch species. Despite the biases and errors associated with MSOMs involving bycatch data, only about a quarter of the studies reviewed by Devarajan et al. mention the presence of bycatch species in the study area.

4 Bayesian inference

A variety of approaches exist for Bayesian inference in distributional regression with structured additive predictors. To assess the posterior distribution of the model parameters, Klein et al. (2015b) propose a MCMC algorithm using a Metropolis-within-Gibbs scheme with iterative weighted least squares (IWLS, Gamerman, 1997) proposals for the parametric and non-parametric regression coefficients β. The IWLS proposals are locally adaptive and make use of the expected or observed Fisher information to construct proposal densities that approximate the curvature of the posterior.

For this reason, they are useful for complex posteriors, and at the same time, they do not require the user to tune any hyperparameters of the MCMC algorithm.

In combination with the IWLS proposals, the algorithm of Klein et al. (2015b) uses conjugate inverse gamma priors for the smoothing parameters τ^2 of the non-parametric covariate effects. Assuming an inverse gamma prior with the hyperparameters a and b for τ^2, the smoothing parameter can be sampled directly from the full conditional $\tau^2 \mid \cdot \sim \text{InvGamma}(a^*, b^*)$, where $a^* = 0.5 \times \text{rk}(\mathbf{K}) + a$, $b^* = 0.5 \times \boldsymbol{\beta}'\mathbf{K}\boldsymbol{\beta} + b$, $\mathbf{K}$ is the penalty matrix, and $\boldsymbol{\beta}$ are the regression coefficients of the covariate effect.

As the IWLS proposals involve second derivatives, they tend to become computationally expensive and numerically unstable if many parameters are sampled together. A popular alternative is the Hamiltonian Monte Carlo (HMC, Neal, 2011) algorithm that simulates the evolution of a Hamiltonian system, defined by a potential and a kinetic energy function, using numerical integration methods to generate posterior samples. To simulate this motion of particles, only the gradient of the log-posterior but no second derivatives are required. Based on the trajectory, new values for the model parameters are proposed, which are finally accepted or rejected in a Metropolis-Hastings step.

To explore the posterior distribution efficiently, the user needs to find a suitable value for the number of leapfrog steps of the HMC algorithm. To eliminate the requirement of tuning the number of leapfrog steps, the No-U-Turn Sampler (NUTS) was developed as a variant of HMC by Hoffman and Gelman (2014). NUTS uses a recursive algorithm to build a binary tree of possible states and to adjust the trajectory of the particles in a dynamic way responding to the curvature of the posterior. For this reason, NUTS is typically easier to use and more efficient than HMC or other traditional MCMC methods.

Our approach to estimate the MSCM combines a Metropolis-within-Gibbs scheme with NUTS, using the same way to block the parameter vector as Klein et al. (2015b) but exchanging the IWLS updates of the parametric and non-parametric regression coefficients β with NUTS updates. For the smoothing parameters τ^2 of the non-parametric covariate effects, we use conjugate inverse gamma priors, allowing us to sample directly from the full conditionals. Additionally, the occupancy states z are sampled in a Gibbs update, and for the species-specific expected abundances $\log(\mu)$ and the occupancy intercepts γ, two separate NUTS updates are performed. More systematically, our MCMC algorithm can be described as follows:

1. Initialize the model parameters as $z = \mathbf{1}(y > 0)$, where $\mathbf{1}$ is the indicator function, $\mu = [\sum_{i=1}^{N} y_i \times \mathbf{1}(y_i > 0)]/[\sum_{i=1}^{N} \mathbf{1}(y_i > 0)]$, i.e. the mean of the non-zero counts for one species over the sites, $\gamma = 0$, $\beta = 0$ and $\tau^2 = 10.000$.
2. For each iteration of the algorithm:
   - Sample the occupancy states z from the binary full conditional in a Gibbs update.
   - Update the species-specific expected abundances $\log(\mu)$ using NUTS.
   - Update the species-specific occupancy intercepts γ using NUTS.
   - Update the site-specific parametric regression coefficients β using NUTS.
   - For each site-specific non-parametric covariate effect:
     - Update the non-parametric regression coefficients β using NUTS.
     - Sample the smoothing parameter τ^2 from the inverse gamma full conditional in a Gibbs update.
3. Repeat step 2 for a great number of iterations to obtain a representative sample from the posterior distribution of the model parameters.

The proposed sampling scheme is designed to iterate over the parameter variables in the model graph from the bottom to the top, so that the highest level of the prior hierarchy is sampled last. The scheme is illustrated in Figure 2 for the MSCM used in the application in Section 6. To implement the scheme, we use Goose, the MCMC library of the probabilistic programming framework Liesel, which among other MCMC kernels, provides the NUTS kernel and an abstract Gibbs kernel (Riebl et al., 2022). Using Goose, only the methods to sample from the full conditionals of the model parameters z and τ^2 need to be implemented manually.

To improve the performance of the sampling scheme, we run it in two phases: a warmup and a posterior phase. During the warmup, the NUTS kernels are allowed to tune their hyperparameters, i.e. the step size and the mass matrix (also called metric). To tune the step size in an adaptive way, we use the dual averaging algorithm, which increases or decreases the step size based on the acceptance probabilities of the previous iterations, hence improving the convergence rate and reducing the dependence on user-defined hyperparameters (Hoffman and Gelman, 2014; Nesterov, 2009). The mass matrix, on the other hand, is supposed to capture correlations in the posterior and is tuned based on the empirical covariance of the warmup samples. A well-adjusted mass matrix can improve the mixing and convergence rate of the HMC and NUTS algorithms substantially in many situations (Betancourt, 2018).

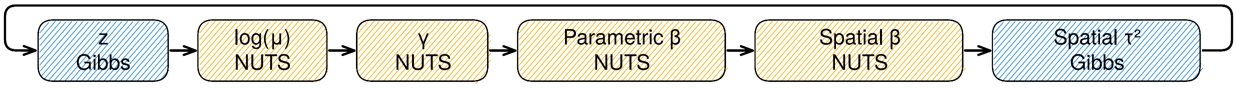


Figure 2: The proposed MCMC sampling scheme for the multi-species count model, illustrated for the model graph in Figure 1. The algorithm iterates over the parameter variables in the model graph from the bottom to the top, so that the highest level of the prior hierarchy is sampled last. Generally, the structured additive predictor η of an MSCM can include more than one non-parametric covariate effect, i.e. there could be additional NUTS and Gibbs kernels for the β and τ^2 parameters of all non-linear, random and spatial effects.

In our experience, the proposed sampling scheme is robust and efficient for many different parameters of the data-generating process of the MSCM. More details on the accuracy and performance of the MCMC algorithm are given in the simulation study in the following section.

5 Simulation study

We examine our sampling scheme in a simulation study with three scenarios: In the first scenario, we confirm that the MCMC algorithm is able to recover the true model parameters reliably in most practically relevant situations. The second scenario is designed to produce more extreme data, i.e. data where the model parameters are closer to the boundaries of the parameter space. We use this data to evaluate the performance of the MCMC algorithm under more challenging conditions. Finally, we demonstrate that the sampling scheme also works with structured additive predictors and spatial effects in Scenario III.

To conduct the simulation study, we generate data from the MSCM, adopting the priors of the model parameters to produce more or less extreme parameters and therefore data. For each scenario, we run 1000 replications, generating four independent chains with 1000 warmup and 1000 post-warmup samples per replication. No thinning is applied to the chains before they are used to compute the summary statistics of the posterior. Various metrics such as the bias, the root-mean-square error (RMSE) and the coverage probability are used to assess the performance of the sampling scheme.

We find that our method works accurately under realistic conditions with and without structured additive predictors. If the occupancy probability ψ is close to zero or one and the expected abundance μ of a species is small, the estimated posterior mean of the species-specific occupancy intercept γ tends to show some bias. However, the posterior standard deviation also increases drastically in these cases, indicating a high uncertainty about the estimate, so that the issue can easily be identified from the MCMC output.

5.1 Scenario I: Stable MCMC estimation

In the first scenario, we demonstrate that our sampling scheme is able to recover the true parameters of the MSCM. For this purpose, we simulate data from the MSCM as defined in Section 2, exchanging the weakly informative priors of the species-specific occupancy intercepts γ and the expected abundances μ with narrower sampling distributions as follows:

$$\beta, \gamma \sim \text{Normal}(\mu = 0,\ \sigma = 0.5),$$
$$\mu \sim \text{Gamma}(\alpha = 10,\ \beta = 1).$$

We also simulate two independent site-specific covariates from a uniform distribution on the unit interval. This configuration implies that the true occupancy probabilities ψ are between 17.6% and 82.3% and the true expected abundances μ are between 4.1 and 18.8 with 99% probability (see Table 1). Despite the modified data-generating process, we use the weakly informative priors from Section 2 for the estimation to encode less knowledge about the DGP in the priors and to estimate the model in the exact same way as in the application in Section 6.

For each of the sample sizes of 40 and 80 sites, and 26 and 52 species, our method is able to recover the true parameters without problems. The estimated posterior means computed from the MCMC chains are unbiased on average, and the bias shrinks with the sample size (see Figure 3). A greater sample size affects the estimation of the various model parameters in different ways: The regression coefficients β benefit from more sites and more species, because they are shared between the sites and the species. On the other hand, the estimation of the occupancy intercepts γ and the expected abundances $\log(\mu)$ can only be improved with more sites, not more species, because they are not shared between the species.

Table 1: The quantiles of the simulated true model parameters in the different scenarios of the simulation study. Scenario I and III are designed to produce realistic data with and without structured additive predictors. In Scenario II, the performance of the sampling scheme is assessed when the model parameters are close to the boundaries of the parameter space.

		Scenario I		Scenario II		Scenario III	
Parameter		1%	99%	1%	99%	1%	99%
Parametric regression coefficients	β	−1.175	1.178	−2.241	2.399	−1.152	1.174
Spatial regression coefficients	β^s	—	—	—	—	−2.312	2.327
Species-specific occupancy intercepts	γ	−1.166	1.170	−2.337	2.328	−1.164	1.204
Species-specific expected abundances	μ	4.141	18.768	0.142	26.029	4.185	18.781
Occupancy probabilities	ψ	0.176	0.823	0.047	0.957	0.120	0.883

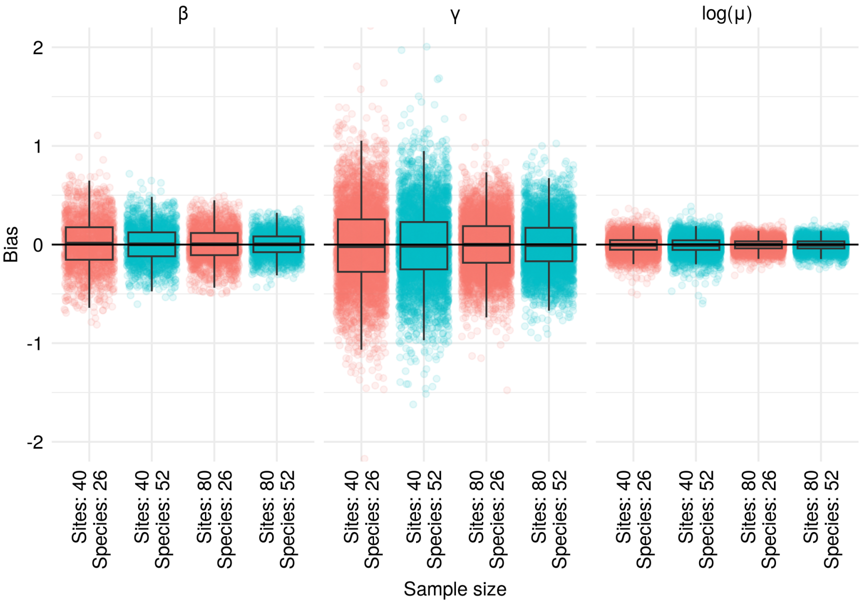


Figure 3: The bias of the estimated posterior mean of the model parameters for 1000 replications of Simulation Scenario I. The average bias is close to zero for all model parameters and sample sizes. Most variability is observed in the bias of the species-specific occupancy intercepts γ, followed by the site-specific regression coefficients β and the species-specific expected abundances $\log(\mu)$. The bias of the regression coefficients β decreases with both the number of sites and species, while the estimation of the occupancy intercepts γ and the expected abundances $\log(\mu)$ only benefits from more sites but not more species.

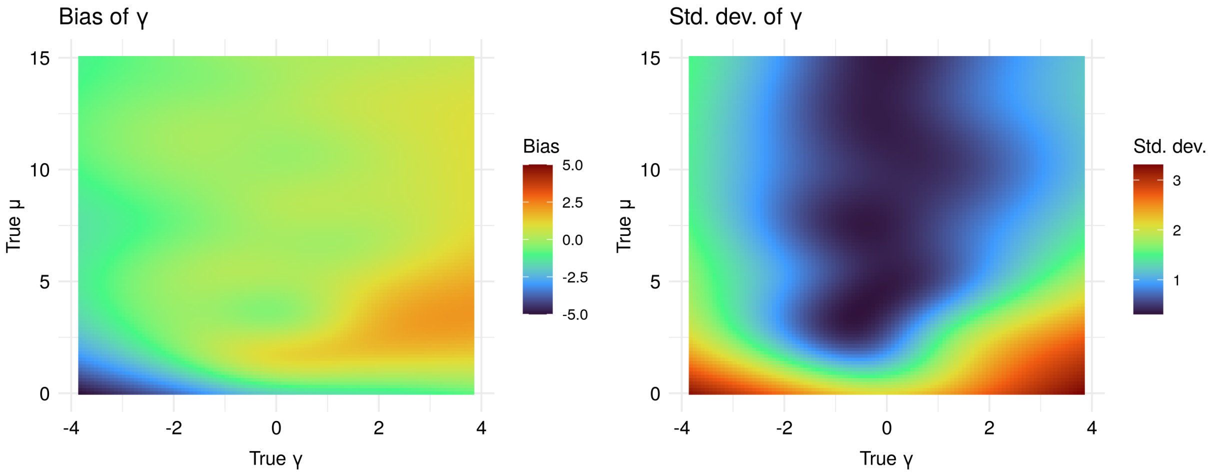


Figure 4: The bias of the estimated posterior mean (left) and the estimated posterior standard deviation (right) of the species-specific occupancy intercept γ as a function of the true occupancy intercept γ and the true expected abundance μ in Simulation Scenario II. If the abundance and the occupancy are small, γ tends to be underestimated, while it tends to be overestimated if the abundance is small but the occupancy is large. In these cases, the posterior standard deviation also increases drastically, indicating a high uncertainty about the estimate.

5.2 Scenario II: Problematic parameter constellations

This simulation scenario includes more cases where the model parameters are close the boundaries of the parameter space, posing a more challenging estimation problem to our MCMC algorithm. To generate this kind of data, we sample the model parameters from the distributions

$$\beta, \gamma \sim \text{Normal}(\mu = 0,\ \sigma = 1),$$
$$\mu \sim \text{HalfNormal}(\sigma = 10).$$

Again, we simulate two site-specific covariates on the unit interval using a uniform distribution. In this scenario, the true occupancy probabilities ψ are between 4.7% and 95.7% and the true expected abundances μ are between 0.1 and 26.0 with 99% probability (Table 1). The usual weakly informative priors are used for the estimation.

From 1000 replications with 40 sites and 26 species, we find that the average bias of the estimated posterior mean is still close to zero for all model parameters, but for the species-specific occupancy intercepts γ and the expected abundances $\log(\mu)$, it increases substantially compared to Scenario I (0.025 vs. -0.010 for γ, and 0.049 vs. -0.006 for $\log(\mu)$). In particular, there are several cases when γ is strongly underestimated (when its true value and the true μ are small), and when it is strongly overestimated (when its true value is large but the true μ is small, see Figure 4). In these cases, the posterior standard deviation also increases drastically, indicating a high uncertainty about the estimate. For this reason, the coverage rates of the 90% credible intervals remain relatively stable with 86.5% for γ and 86.7% for $\log(\mu)$.

The difficulties with these parameter constellations are to be expected considering their interpretation: A low expected abundance μ implies that few or no individuals are observed at each site, no matter if the site was occupied by the species or not. Under these circumstances, it is hard to disentangle whether an observed zero is the result of a low occupancy probability or a low expected abundance, and hence the species-specific occupancy intercept γ is not well-identified. Finally, it is worth mentioning that despite the bias in the estimation of γ, the occupancy probabilities ψ including the site-specific regression coefficients β are still estimated quite accurately on the unit interval after the inverse logit transformation.

5.3 Scenario III: Structured additive predictors

In the last scenario, we verify that our sampling scheme also works with the model structure used in Section 6, i.e. with a structured additive predictor and a spatial effect. The spatial effect is modeled as a latent Gaussian process with a Matérn correlation function and a fixed smoothness parameter $\nu = 1.5$. The simulated data comprises 40 sites and 26 species. The 40 sites are geographically clustered in the same way as in Section 6, resulting in a similar correlation matrix as in Figure 5. In addition to the spatial effect, this scenario also includes two continuous covariates on the unit interval with parametric linear effects. The spatial regression coefficients are sampled from an improper multivariate normal distribution with the smoothing parameter $\tau^2 = 1$, and all other model parameters are sampled from the same distributions as in Scenario I.

Although there are substantially more parameters than in Scenario I, our method is able to estimate the spatial effect reliably. The average bias of the estimated posterior mean of the spatial regression coefficients (0.005) is on a similar scale as the bias of the parametric regression coefficients (−0.001). The bias of the parametric regression coefficients does not increase compared to Scenario I (−0.001 vs. 0.011 in Scenario I). Moreover, the coverage rates of the 90% credible intervals of the spatial regression coefficients are remarkably accurate (90.1%). Only the smoothing parameter τ^2 of the spatial effect is slightly overestimated with an average bias of 0.312 of the estimated posterior mean, which is an effect of the heavy-tailed inverse gamma prior.

6 Application: Species diversity in mixed forest stands in Lower Saxony, Germany

In this section, we apply our model to assess the diversity of communities of various taxa within the flora, fauna and microorganisms in pure and mixed forest stands located in Lower Saxony, a state in northwest Germany. The study design of the Research Training Group (RTG) 2300, which collected the data, is described, followed by the specification of a structured additive predictor for the occupancy probabilities. Finally, we present and discuss the estimation results obtained from the model.

6.1 Research Training Group 2300

The Research Training Group 2300, which is based at the University of Göttingen, is devoted to investigating the ecosystem functions of pure and mixed forest stands comprising European beech, Norway spruce and Douglas fir. The cultivation of non-native species, e.g. Douglas fir, in managed forests in central Europe is believed to potentially alleviate the effects of climate change (Glatthorn et al., 2023). To gain a more comprehensive understanding of the possible implications of introducing a foreign species into the native ecosystem, the RTG examines various functional traits of the tree species and associated organisms at eight field sites and 40 experimental plots.

The field sites of the RTG are distributed across Lower Saxony in northwest Germany, where the climate is temperate. Each site comprises five experimental plots of square or rectangular shape and 0.25 ha in size. The plots are located in even-aged, state-owned forests (Ammer et al., 2020). Four of the eight field sites are in the uplands of Lower Saxony, specifically in the Solling and Harz mountain ranges, and the remaining four in the lowlands in the north. Due to the high precipitation and clay content in the soil, the environmental conditions on the upland plots generally tend to be more favorable (Foltran et al., 2022).

At each field site, three experimental plots in pure stands and two in mixed stands were established, featuring stand ages that vary from 42 to 130 years, with an average of 80 years. The pure stands are either dominated by native broadleaved European beech, native coniferous Norway spruce or non-native coniferous Douglas fir. The mixed stands are composed of beech and one of the conifers, specifically mixtures of beech and Douglas fir as well as beech and spruce. After the original plots were established in 2017, seven of them had to be relocated following a windthrow in early 2018.

In the context of the RTG's research theme, Glatthorn et al. (2023) study the abundance and diversity of multiple taxa that are relevant to ecosystem functioning, such as fungi, plants, arthropods and small mammals. Their findings indicate that pure stands of Douglas fir provide habitats that can accommodate communities of equal or greater diversity than those in beech or spruce stands. At the same time, the diversity of communities in mixed stands of beech in combination with spruce or Douglas fir is not generally improved compared to pure stands.

Glatthorn et al. (2023) employ a two-step procedure to analyze the data: First, they estimate various abundance and diversity indices for each experimental plot, and then relate them to the stand types using different mixed models. We aim to replicate parts of Glatthorn et al.'s work with a more comprehensive approach based on the proposed multi-species count model. Specifically, we focus on three taxa: collembola, small mammals and vegetation. Collembola were sampled between November 2017 and January 2018 by collecting one soil core with a diameter of 5 cm per plot. The arthropods were extracted using high-gradient heat extraction (Macfadyen, 1961) and identified to the species level (Lu, 2021).

From July to September of 2018, 2019 and 2020, small mammals were surveyed using 64 Sherman traps per plot, arranged in an 8×8 grid with 10 m between the traps. Simultaneous surveys were conducted for all plots at one site over four consecutive nights per year. The captured animals were identified to the species level and individually marked to identify future recaptures (Glatthorn et al., 2023). The cover abundance of plant species was visually estimated on 100 m^2 subplots in May and June of 2020 following Braun-Blanquet (1951). The three selected taxa – collembola,

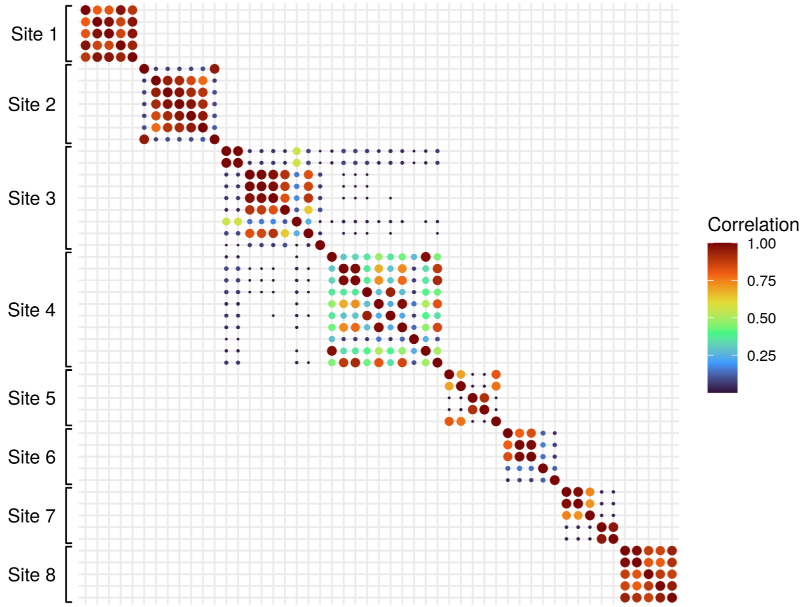


Figure 5: The assumed prior correlation matrix of the experimental plots of the RTG based on their geographical distance. The eight field sites of the RTG are located in Lower Saxony in northwest Germany. Originally, each site consisted of five experimental plots, but some of the plots had to be replaced due to storm and bark beetle damage. The correlation matrix is computed from the Matérn correlation function with range $\rho \approx 0.008$ (so that the correlation effectively decays to zero after 5 km) and smoothness $\nu = 1.5$. It is used with the latent Gaussian process representing the spatial effect in the structured additive predictor of the RTG multi-species count model.

small mammals and vegetation – were chosen to demonstrate the versatility of the MSCM, which can be applied to data from sampling methods as different as soil coring, mark-recapture and visual vegetation assessment.

6.2 Predictor and model specification

As described in Section 2.1, the probability for a species to occupy a plot is modeled with a structured additive predictor combining parametric and non-parametric covariate effects. In the particular case of the RTG-MSCM, the parametric covariate effects are the species-specific occupancy intercepts γ and the composition of tree species on the experimental plots. The composition of tree species is measured in terms of the area potentially available (APA, Glatthorn, 2021) for the two coniferous species Norway spruce and Douglas fir. The APA is computed from a weighted Voronoi tessellation of the plot, approximating the growing space that each tree can exploit. As European beech, Norway spruce and Douglas fir are the three dominant species on the plots, their APA is defined on an approximate simplex, i.e. $\text{APA}_{\text{Beech}} + \text{APA}_{\text{Spruce}} + \text{APA}_{\text{Douglas}} \approx 1$.

The non-parametric covariate effect is a spatial effect modeled as a Gaussian process, assuming a correlation structure between the experimental plots based on their geographical distance. The spatial effect accounts for unmeasured environmental and biological factors, hence capturing the otherwise unexplained variability in the data. The correlation matrix of the GP is computed from the Matérn correlation function with range $\rho \approx 0.008$ (so that the correlation effectively decays to zero after 5 km) and smoothness $\nu = 1.5$. Figure 5 shows how the correlation matrix reflects the study design of the RTG with its eight field sites and five experimental plots per site. Most plots that are part of the same site are strongly correlated, while the correlation between the sites is effectively zero in most cases. In total, there are more than 40 plots, because some of the original plots had to be replaced due to storm and bark beetle damage.

Gaussian processes are stochastic processes, i.e. collections of random variables, where every finite subset of the random variables has a joint multivariate normal distribution. They are popular both in machine learning and spatial statistics (Rasmussen and Williams, 2005). In the context of spatial statistics, they are often used for kriging, a method for interpolating continuously indexed spatial data (Cressie, 1993). Using a GP with the Matérn correlation function, a fixed range ρ and the smoothness parameter $\nu = 1.5$, as we do in this application, is common practice when working with kriging and geoadditive models (Kammann and Wand, 2003). One benefit of this approach is that only the variance

Table 2: The widely applicable information criterion of the RTG multi-species count model for different taxa and count distributions. Note that the WAIC is reported on the log-score scale rather than the deviance scale, i.e. higher values indicate a better predictive accuracy. For all taxa, the data shows some degree of overdispersion, making the location-scale parameterization of the negative binomial distribution a better fit than the Poisson distribution. The heavy-tailed Yule distribution performs better than the Poisson but worse than the negative binomial distribution for all studied taxa.

	Count distribution	WAIC	Std. err.	Eff. param.
Collembola	Negative binomial	**−3281.88**	351.51	132.68
	Poisson	−3566.92	396.68	166.45
	Yule	−3384.62	356.98	129.74
Small mammals	Negative binomial	**−2966.08**	237.28	31.11
	Poisson	−3330.91	281.24	61.55
	Yule	−3102.36	239.58	29.90
Vegetation	Negative binomial	**−19022.28**	3378.80	1028.34
	Poisson	−21016.58	3627.85	1156.56
	Yule	−19174.55	3390.47	1049.61

(or smoothing) parameter τ^2 needs to be estimated. Generally, the Matérn correlation function is considered to provide a good balance between smoothness and flexibility of the estimated processes.

For each taxon, we estimate the MSCM with the aforementioned structured additive predictor using three different count distributions for the total number of observations per experimental plot: the Poisson distribution, the negative binomial distribution in a location-scale parameterization (Rigby et al., 2019, Chapter 22.2.3), and the Yule distribution (Rigby et al., 2019, Chapter 22.1.4). The negative binomial distribution can account for potential overdispersion of the data compared to the standard Poisson distribution, while the Yule distribution is heavy-tailed. Comparing the different models by the WAIC, the negative binomial distribution shows the best performance across all taxa (Table 2). For the vegetation data, the predictive accuracy of the heavy-tailed Yule distribution is almost as high as that of the negative binomial distribution, while the Poisson distribution seems to fit worst in all cases. For this reason, the presentation of the estimation results in the remainder of this section is only based on the MSCM with the negative binomial distribution.

6.3 Estimation results

For each taxon and count distribution, the model was configured using Liesel and estimated using Goose, following the sampling scheme proposed in Section 4. Four chains were sampled in parallel with 1000 warmup and 1000 posterior iterations. While sampling, the species richness and Shannon index on the plot and landscape-level were tracked to assess their posterior distribution. No thinning was applied to the chains before computing the summary statistics of the posterior distribution. For the collembola and vegetation data, the model was estimated without any errors and with a good effective sample size (ESS), while for the small mammal data, some divergent transitions of the NUTS kernels were observed. This is a consequence of the small mammal data being substantially smaller than the others with only seven species in total, four of which are very rare.

Our findings regarding the spatial effect on the species diversity are in line with Glatthorn et al. (2023). For all three taxa, species richness and Shannon index are consistently estimated to be higher in southern Lower Saxony than in the north. Figure 6 shows the posterior distribution of both diversity indices on the plot and landscape-level for collembola. The differences between the northern and southern plots are generally very pronounced for collembola and for the vegetation. For small mammals, the posterior distribution also indicates a slightly higher species diversity in the south, but as the number of observed small mammal species is generally very low (posterior mean of 7.0 of the landscape species richness in the south, 5.32 in the north), the difference is quite hard to identify.

The impact of the combination of tree species on the species diversity remains somewhat ambiguous. The diversity of collembola shows a trend towards a reduced species richness and Shannon index with a higher APA for spruce and Douglas fir. However, the posterior variance of the effect is high, indicating a substantial estimation uncertainty. In contrast, the diversity of small mammals and the vegetation appears to increase with a higher APA for the coniferous species. While for small mammals, the estimation uncertainty remains high, the positive effect on the vegetation is rather pronounced, as shown in Figure 7. Glatthorn et al. (2023) report a similar pattern across the taxa collembola, small mammals and vegetation.

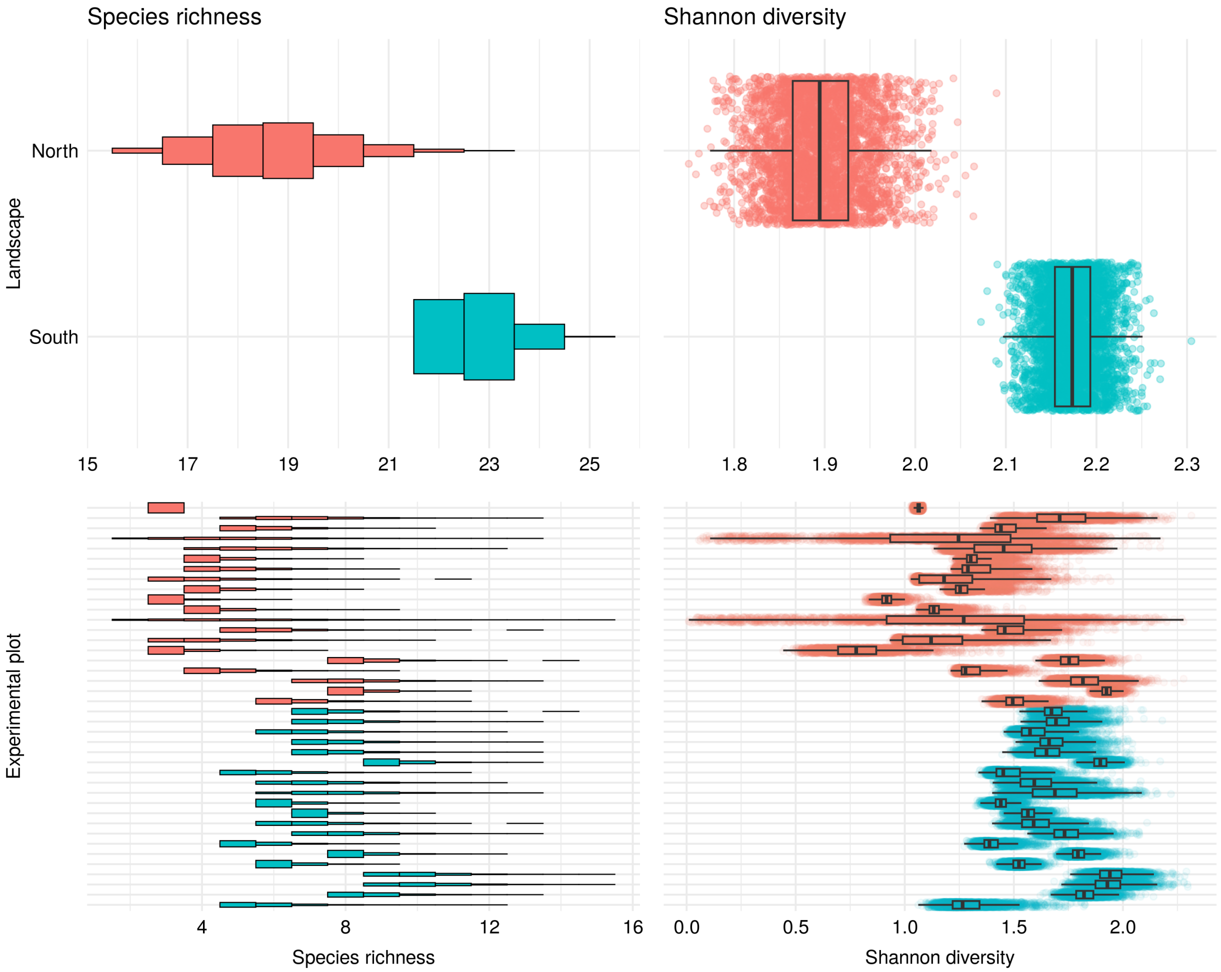


Figure 6: The posterior distribution of the α and γ-diversity (on the plot and landscape-level) of collembola depending on the geographic location. The generally more favorable environmental conditions on the experimental plots in southern Lower Saxony are clearly reflected in a higher average species richness and Shannon index of collembola.

The model specification presented in this section only allows for the estimation of the species effect of spruce and Douglas fir relative to beech. The effect of mixed stands compared to pure stands cannot be assessed with this parameterization. To address this question, a non-linear effect of the APA of spruce and Douglas fir on the structured additive predictor for the occupancy probabilities would need to be assumed and modeled using e.g. polynomials or P-splines. The figures illustrating the spatial effect and the effect of the composition of tree species on the diversity of the taxa omitted in the article may be found in the supplementary material.

7 Conclusion

In conclusion, this study introduces the multi-species count model as a new model class for assessing the relationship between site conditions and species diversity. The model allows us to incorporate a structured additive predictor with linear, non-linear, random and spatial effects describing the site conditions. It can be estimated with a fully Bayesian inference scheme based on an efficient MCMC algorithm, which we evaluate in a simulation study with different scenarios and apply to data from a large-scale ecological research project in Lower Saxony, Germany.

We use the model to study the effect of admixing two coniferous species on the ecosystem in European beech forests, accounting for the spatial correlation between the field sites. This application demonstrates the usefulness of the model in real-world scenarios, where it can provide insights into the complex relationships between species diversity and site conditions. Generally, the model can be applied to a broad range of problems, including the analysis of different species

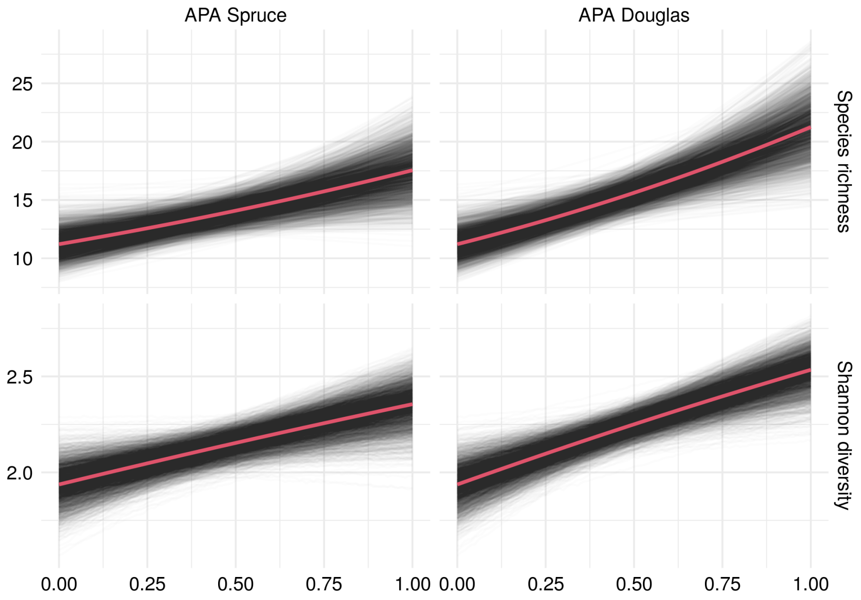


Figure 7: The posterior distribution of the effect of the composition of tree species on the α-diversity (on the plot-level) of the overall vegetation on the experimental plots. The composition of tree species is measured in terms of the area potentially available (APA, computed from a weighted Voronoi tessellation of the plot) for the different species. A higher APA for the coniferous species Norway spruce and Douglas fir corresponds with a higher average species richness and Shannon index of the vegetation on the plots. This effect is even more pronounced for Douglas fir than for Norway spruce.

diversity indices and taxa. It can be used to estimate both the occupancy probabilities and the expected abundances of the studied species. Due to its low data requirements, it is a particularly useful tool for meta-studies across various taxa that are sampled according to different study designs.

Modeling species compositions and their driving environmental factors via species occupancy probabilities, and aggregating them in ecologically meaningful indices in a Bayesian framework offers unique flexibility in terms of research questions that can be addressed in a consistent way. For example, analyses of contrasts between different species groups (rare vs. common, specialists vs. generalists, etc.) or of trait compositions of species communities depending on environmental factors can all be derived from the same model. Through sampling from the posterior predictive distribution conditional on environmental factors, simulation studies about the impact of different landscapes on the overall species composition can be carried out easily.

Several aspects deserve further attention in future research: One key aspect to consider is the extension of the model specification with another structured additive predictor for the expected abundances of the species, potentially introducing a functional relationship between the new and the old predictor for the occupancy probabilities. Moreover, alternative parameterizations and count distributions for the total number of observations per experimental plot could be explored. It should be noted that, unlike multi-species occupancy models, our model lacks the ability to disentangle the detection probabilities and expected abundances of the species. The current version of the model also cannot estimate the size of a meta-community accounting for unobserved species. To overcome these limitations, more complex versions of the model could be developed in the future.

Overall, the proposed multi-species count model provides a flexible and powerful framework for the analysis of species diversity data, and can be used to identify key environmental drivers of biodiversity patterns. Our study contributes to the growing body of literature highlighting the potential of Bayesian hierarchical modeling in ecology. The multi-species count model can be applied to many different datasets, and we hope it will stimulate further research in the field of community ecology.

References

Christian Ammer, Peter Annighöfer, Niko Balkenhol, Dietrich Hertel, Christoph Leuschner, Andrea Polle, Norbert Lamersdorf, Stefan Scheu, and Jonas Glatthorn. RTG 2300 – Study design, location, topography and climatic conditions of research plots in 2020. PANGAEA, September 2020. doi:10.1594/PANGAEA.923125.

Michael Betancourt. A Conceptual Introduction to Hamiltonian Monte Carlo. arXiv, July 2018. doi:10.48550/arXiv.1701.02434.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: Composable transformations of Python+NumPy programs, 2023.

Josias Braun-Blanquet. *Pflanzensoziologie: Grundzüge der Vegetationskunde*. Springer, Vienna, 1951. ISBN 978-3-7091-4079-6 978-3-7091-4078-9. doi:10.1007/978-3-7091-4078-9.

Susan Budge, Armann Ingolfsson, and Dawit Zerom. Empirical Analysis of Ambulance Travel Times: The Case of Calgary Emergency Medical Services. *Management Science*, 56(4):716–723, 2010. ISSN 0025-1909.

Adrian Colin Cameron and Pravin K. Trivedi. *Regression Analysis of Count Data*. Econometric Society Monographs. Cambridge University Press, Cambridge, second edition, 2013. ISBN 978-1-107-01416-9. doi:10.1017/CBO9781139013567.

Robert K. Colwell. Biodiversity: Concepts, Patterns, and Measurement. In Simon A. Levin, Stephen R. Carpenter, H. Charles J. Godfray, Ann P. Kinzig, Michel Loreau, Jonathan B. Losos, Brian Walker, and David S. Wilcove, editors, *The Princeton Guide to Ecology*, pages 257–263. Princeton University Press, Princeton, July 2009. ISBN 978-1-4008-3302-3. doi:10.1515/9781400833023.257.

Noel A. C. Cressie. *Statistics for Spatial Data*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., Hoboken, NJ, USA, September 1993. ISBN 978-1-119-11515-1 978-0-471-00255-0. doi:10.1002/9781119115151.

Kadambari Devarajan, Toni Lyn Morelli, and Simone Tenan. Multi-species occupancy models: Review, roadmap, and recommendations. *Ecography*, 43(11):1612–1624, 2020. ISSN 1600-0587. doi:10.1111/ecog.04957.

Robert M. Dorazio and Jeffrey Andrew Royle. Estimating Size and Composition of Biological Communities by Modeling the Occurrence of Species. *Journal of the American Statistical Association*, 100(470):389–398, June 2005. ISSN 0162-1459. doi:10.1198/016214505000000015.

Paul H. C. Eilers and Brian D. Marx. Flexible smoothing with B-splines and penalties. *Statistical Science*, 11(2): 89–121, May 1996. ISSN 2168-8745. doi:10.1214/ss/1038425655.

Ludwig Fahrmeir, Thomas Kneib, and Stefan Lang. Penalized Structured Additive Regression for Space-Time Data: A Bayesian Perspective. *Statistica Sinica*, 14(3):731–761, 2004. ISSN 1017-0405.

Ludwig Fahrmeir, Thomas Kneib, Stefan Lang, and Brian Marx. *Regression: Models, Methods and Applications*. Springer, Berlin, Heidelberg, 2013. ISBN 978-3-642-34333-9. doi:10.1007/978-3-642-34333-9.

Estela Covre Foltran, Christian Ammer, and Norbert Lamersdorf. Douglas fir and Norway spruce admixtures to beech forests along in Northern Germany – Are soil nutrient conditions affected? bioRxiv, March 2022. doi:10.1101/2020.09.25.313213.

Dani Gamerman. Sampling from the posterior distribution in generalized linear mixed models. *Statistics and Computing*, 7(1):57–68, March 1997. ISSN 1573-1375. doi:10.1023/A:1018509429360.

Andrew Gelman. Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian Analysis*, 1(3):515–534, September 2006. ISSN 1936-0975, 1931-6690. doi:10.1214/06-BA117A.

Jonas Glatthorn. A spatially explicit index for tree species or trait diversity at neighborhood and stand level. *Ecological Indicators*, 130:108073, November 2021. ISSN 1470-160X. doi:10.1016/j.ecolind.2021.108073.

Jonas Glatthorn, Scott Appleby, Niko Balkenhol, Peter Kriegel, Likulunga Emmanuel Likulunga, Jing-Zhong Lu, Dragan Matevski, Andrea Polle, Hannes Riebl, Carmen Alicia Rivera Pérez, Stefan Scheu, Alexander Seinsche, Peter Schall, Andreas Schuldt, Severin Wingender, and Christian Ammer. Species diversity of forest floor biota in non-native Douglas-fir stands is similar to that of native stands. *Ecosphere*, 14(7):e4609, 2023. ISSN 2150-8925. doi:10.1002/ecs2.4609.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, New York, 2009. ISBN 978-0-387-84858-7. doi:10.1007/978-0-387-84858-7.

Ed Hawkins, Thomas E. Fricker, Andrew J. Challinor, Christopher A. T. Ferro, Chun Kit Ho, and Tom M. Osborne. Increasing influence of heat stress on French maize yields from the 1960s to the 2030s. *Global Change Biology*, 19 (3):937–947, 2013. ISSN 1365-2486. doi:10.1111/gcb.12069.

M. O. Hill. Diversity and Evenness: A Unifying Notation and Its Consequences. *Ecology*, 54(2):427–432, 1973. ISSN 0012-9658. doi:10.2307/1934352.

Matthew D. Hoffman and Andrew Gelman. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(47):1593–1623, 2014. ISSN 1533-7928.

Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services. Global assessment report on biodiversity and ecosystem services of the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services. Technical report, Zenodo, May 2019.

Erin E. Kammann and Matthew P. Wand. Geoadditive models. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 52(1):1–18, 2003. ISSN 1467-9876. doi:10.1111/1467-9876.00385.

Matthias Katzfuss and Joseph Guinness. A General Framework for Vecchia Approximations of Gaussian Processes. *Statistical Science*, 36(1):124–141, February 2021. ISSN 2168-8745. doi:10.1214/19-STS755.

Marc Kéry and Michael Schaub. *Bayesian Population Analysis Using WinBUGS: A Hierarchical Perspective*. Elsevier, 2012. ISBN 978-0-12-387020-9. doi:10.1016/C2010-0-68368-4.

Mizanur R. Khondoker, Chris A. Glasbey, and Bruce J. Worton. A Comparison of Parametric and Nonparametric Methods for Normalising cDNA Microarray Data. *Biometrical Journal*, 49(6):815–823, 2007. ISSN 1521-4036. doi:10.1002/bimj.200610338.

Nadja Klein and Thomas Kneib. Scale-Dependent Priors for Variance Parameters in Structured Additive Distributional Regression. *Bayesian Analysis*, 11(4):1071–1106, December 2016a. ISSN 1936-0975, 1931-6690. doi:10.1214/15-BA983.

Nadja Klein and Thomas Kneib. Simultaneous inference in structured additive conditional copula regression models: A unifying Bayesian approach. *Statistics and Computing*, 26(4):841–860, July 2016b. ISSN 1573-1375. doi:10.1007/s11222-015-9573-6.

Nadja Klein, Thomas Kneib, Stephan Klasen, and Stefan Lang. Bayesian Structured Additive Distributional Regression for Multivariate Responses. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 64(4):569–591, August 2015a. ISSN 0035-9254. doi:10.1111/rssc.12090.

Nadja Klein, Thomas Kneib, and Stefan Lang. Bayesian Generalized Additive Models for Location, Scale, and Shape for Zero-Inflated and Overdispersed Count Data. *Journal of the American Statistical Association*, 110(509):405–419, January 2015b. ISSN 0162-1459. doi:10.1080/01621459.2014.912955.

Diane Lambert. Zero-Inflated Poisson Regression, with an Application to Defects in Manufacturing. *Technometrics*, 34 (1):1–14, 1992. ISSN 0040-1706. doi:10.2307/1269547.

Stefan Lang and Andreas Brezger. Bayesian P-Splines. *Journal of Computational and Graphical Statistics*, 13(1): 183–212, March 2004. ISSN 1061-8600. doi:10.1198/1061860043010.

Jingzhong Lu. *Community Structure and Guild Patterns of Soil Decomposers in Pure and Mixed Forests of European Beech, Norway Spruce and Douglas Fir*. PhD thesis, University of Göttingen, October 2021. doi:10.53846/goediss-8908.

Amyan Macfadyen. Improved Funnel-Type Extractors for Soil Arthropods. *Journal of Animal Ecology*, 30(1):171–184, 1961. ISSN 0021-8790. doi:10.2307/2120.

Darryl I. MacKenzie, Larissa L. Bailey, and James. D. Nichols. Investigating species co-occurrence patterns when species are detected imperfectly. *Journal of Animal Ecology*, 73(3):546–555, 2004. ISSN 1365-2656. doi:10.1111/j.0021-8790.2004.00828.x.

Patrick Michaelis, Nadja Klein, and Thomas Kneib. Bayesian Multivariate Distributional Regression With Skewed Responses and Skewed Random Effects. *Journal of Computational and Graphical Statistics*, 27(3):602–611, July 2018. ISSN 1061-8600. doi:10.1080/10618600.2017.1395343.

John Mullahy. Specification and testing of some modified count data models. *Journal of Econometrics*, 33(3):341–365, December 1986. ISSN 0304-4076. doi:10.1016/0304-4076(86)90002-3.

Radford M. Neal. MCMC Using Hamiltonian Dynamics. In *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC, New York, 2011. ISBN 978-0-429-13850-8.

John A. Nelder and Robert W. M. Wedderburn. Generalized Linear Models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384, 1972. ISSN 0035-9238. doi:10.2307/2344614.

Yurii Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120(1):221–259, August 2009. ISSN 1436-4646. doi:10.1007/s10107-007-0149-x.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. November 2005. doi:10.7551/mitpress/3206.001.0001.

Hannes Riebl, Paul F. V. Wiemann, and Thomas Kneib. Liesel: A Probabilistic Programming Framework for Developing Semi-Parametric Regression Models and Custom Bayesian Inference Algorithms. arXiv, September 2022. doi:10.48550/arXiv.2209.10975.

Robert A. Rigby and Mikis D. Stasinopoulos. Generalized additive models for location, scale and shape. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 54(3):507–554, 2005. ISSN 1467-9876. doi:10.1111/j.1467-9876.2005.00510.x.

Robert A. Rigby, Mikis D. Stasinopoulos, Gillian Z. Heller, and Fernanda De Bastiani. *Distributions for Modeling Location, Scale, and Shape: Using GAMLSS in R*. Chapman and Hall/CRC, New York, October 2019. ISBN 978-0-429-29854-7. doi:10.1201/9780429298547.

Josemar Rodrigues, Mário de Castro, Vicente G. Cancho, and Narayanaswamy Balakrishnan. COM–Poisson cure rate survival models and an application to a cutaneous melanoma data. *Journal of Statistical Planning and Inference*, 139 (10):3605–3611, October 2009. ISSN 0378-3758. doi:10.1016/j.jspi.2009.04.014.

Christopher T. Rota, Marco A. R. Ferreira, Roland W. Kays, Tavis D. Forrester, Elizabeth L. Kalies, William J. McShea, Arielle W. Parsons, and Joshua J. Millspaugh. A multispecies occupancy model for two or more interacting species. *Methods in Ecology and Evolution*, 7(10):1164–1173, 2016. ISSN 2041-210X. doi:10.1111/2041-210X.12587.

Havard Rue and Leonhard Held. *Gaussian Markov Random Fields: Theory and Applications*. Chapman and Hall/CRC, New York, February 2005. ISBN 978-0-429-20882-9. doi:10.1201/9780203492024.

Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(4):623–656, October 1948. ISSN 0005-8580. doi:10.1002/j.1538-7305.1948.tb00917.x.

Mikis D. Stasinopoulos, Robert A. Rigby, and Fernanda De Bastiani. GAMLSS: A distributional regression approach. *Statistical Modelling*, 18(3-4):248–273, June 2018. ISSN 1471-082X. doi:10.1177/1471082X18759144.

Ole Tange. GNU parallel 20230322 ('arrest warrant'). Zenodo, March 2023. doi:10.5281/zenodo.7761866.

Nikolaus Umlauf and Thomas Kneib. A primer on Bayesian distributional regression. *Statistical Modelling*, 18(3-4): 219–247, June 2018. ISSN 1471-082X. doi:10.1177/1471082X18759140.

Gabriele Villarini, James A. Smith, Francesco Serinaldi, Jerad Bales, Paul D. Bates, and Witold F. Krajewski. Flood frequency analysis for nonstationary annual peak records in an urban drainage basin. *Advances in Water Resources*, 32(8):1255–1266, August 2009. ISSN 0309-1708. doi:10.1016/j.advwatres.2009.05.003.

Vlasios Voudouris, Robert Ayres, Andre Cabrera Serrenho, and Daniil Kiose. The economic growth enigma revisited: The EU-15 since the 1970s. *Energy Policy*, 86:812–832, November 2015. ISSN 0301-4215. doi:10.1016/j.enpol.2015.04.027.

James Hardin Waddle, Robert M. Dorazio, Susan C. Walls, Kenneth G. Rice, Jeff Beauchamp, Melinda J. Schuman, and Frank J. Mazzotti. A new parameterization for estimating co-occurrence of interacting species. *Ecological Applications*, 20(5):1467–1475, 2010. ISSN 1939-5582. doi:10.1890/09-0850.1.

Robert H. Whittaker. Vegetation of the Siskiyou Mountains, Oregon and California. *Ecological Monographs*, 30(3): 279–338, 1960. ISSN 0012-9615. doi:10.2307/1943563.

Robert J. Whittaker, Katherine J. Willis, and Richard Field. Scale and species richness: Towards a general, hierarchical theory of species diversity. *Journal of Biogeography*, 28(4):453–470, 2001. ISSN 1365-2699. doi:10.1046/j.1365-2699.2001.00563.x.

Simon N. Wood. *Generalized Additive Models: An Introduction with R, Second Edition*. Chapman and Hall/CRC, New York, second edition, May 2017. ISBN 978-1-315-37027-9. doi:10.1201/9781315370279.

Supplementary material

In this supplementary material, we present additional figures that are not included in the article "A Structured Additive Multi-Species Count Model for Assessing the Relation Between Site Conditions and Species Diversity". The figures show the estimation results for the taxa that are omitted from Section 6, i.e. the application on species diversity in mixed forest stands in Lower Saxony, Germany. Specifically, the spatial effect, i.e. the posterior distribution of species diversity depending on the geographic location, and the tree species effect, i.e. the posterior distribution of the effect of the composition of tree species on species diversity, are shown. See Section 6.3 for more details on the interpretation of the figures.

Spatial effect

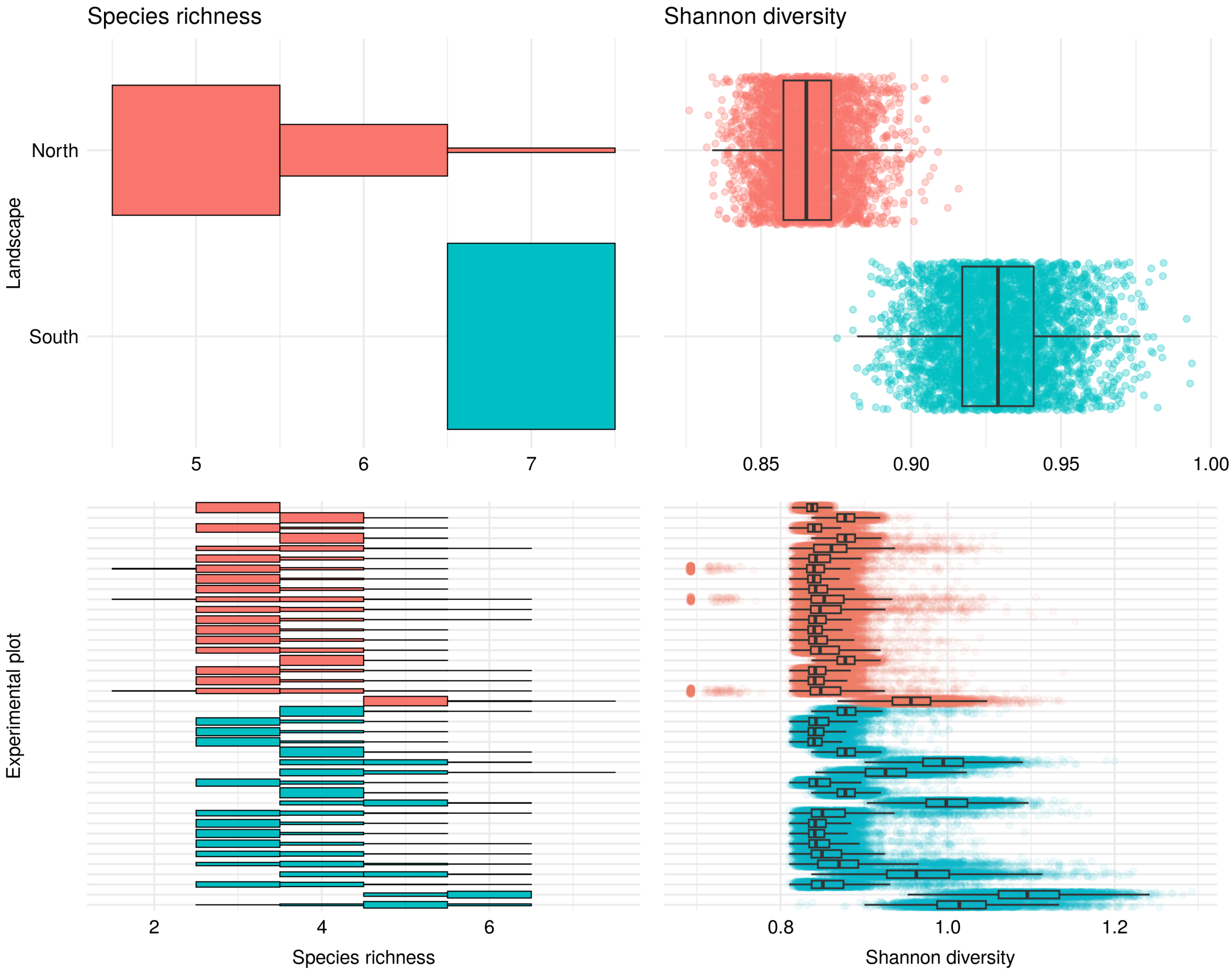


Figure 8: The posterior distribution of the α and γ-diversity (on the plot and landscape-level) of **small mammals** depending on the geographic location.

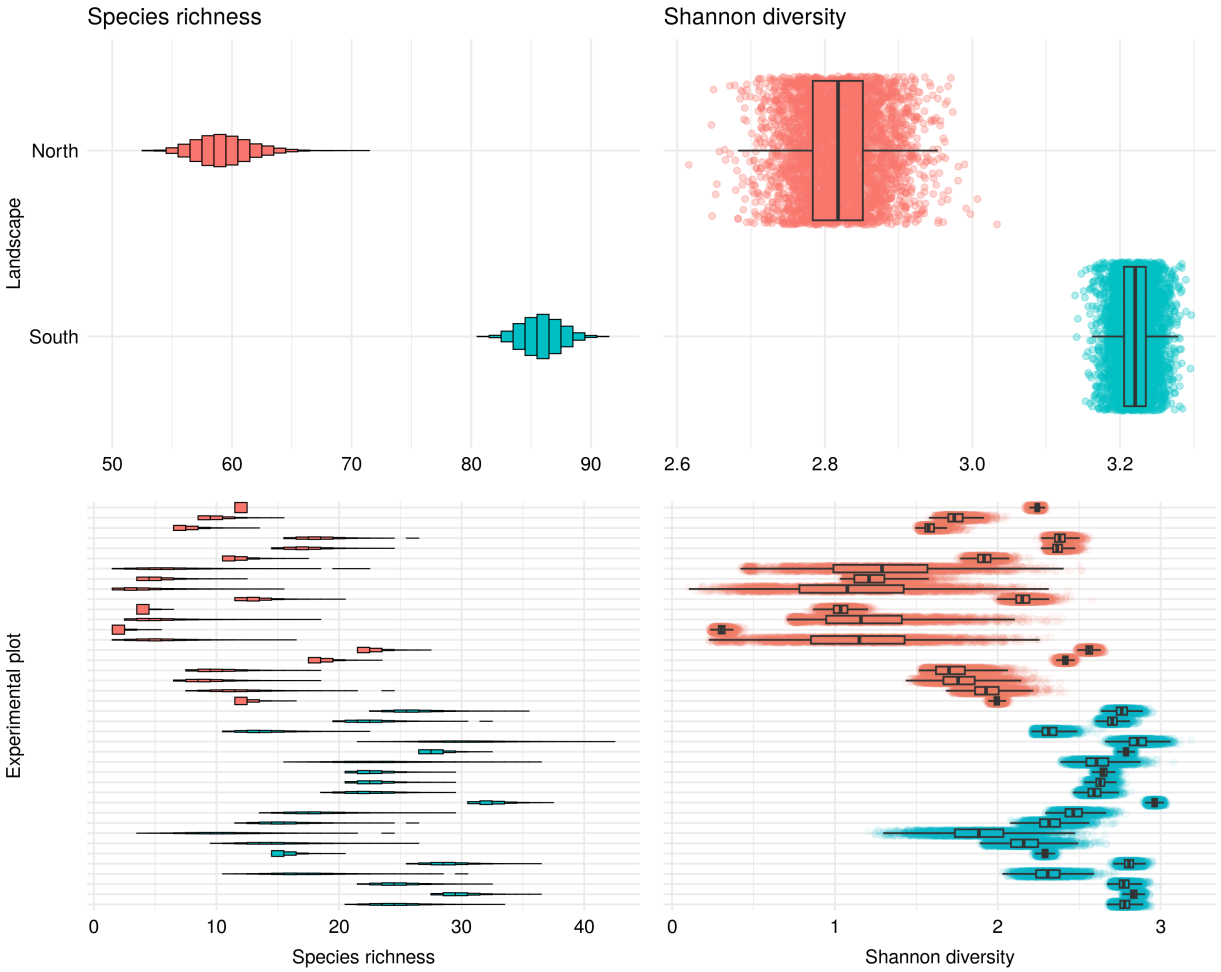


Figure 9: The posterior distribution of the α and γ-diversity (on the plot and landscape-level) of the **vegetation** depending on the geographic location.

Tree species effect

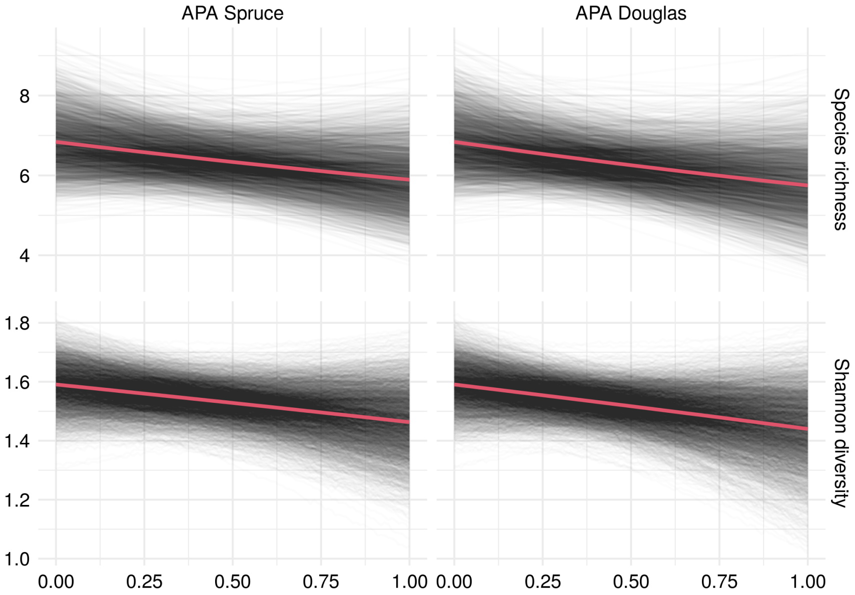


Figure 10: The posterior distribution of the effect of the composition of tree species on the α-diversity (on the plot-level) of **collembola** on the experimental plots.

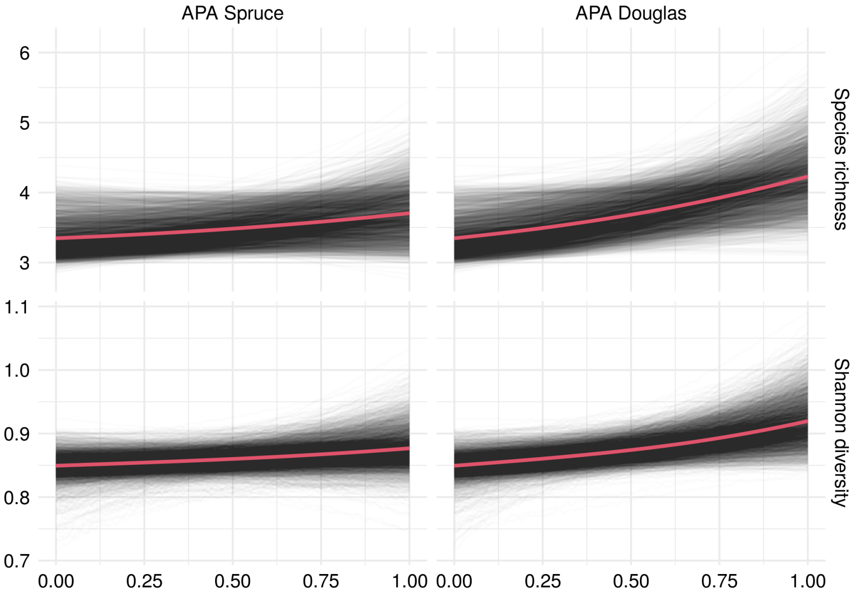


Figure 11: The posterior distribution of the effect of the composition of tree species on the α-diversity (on the plot-level) of **small mammals** on the experimental plots.