Statistics and Computing (2025) 35:43
https://doi.org/10.1007/s11222-025-10569-y

ORIGINAL PAPER

Anisotropic multidimensional smoothing using Bayesian tensor product P-splines

Paul Bach[1] · **Nadja Klein**[2]

Received: 7 April 2024 / Accepted: 13 January 2025

Abstract

We introduce a highly efficient fully Bayesian approach for anisotropic multidimensional smoothing. The main challenge in this context is the Markov chain Monte Carlo (MCMC) update of the smoothing parameters as their full conditional posterior comprises a pseudo-determinant that appears to be intractable at first sight. As a consequence, existing implementations are computationally feasible only for the estimation of two-dimensional tensor product smooths, which is, however, too restrictive for many applications. In this paper, we break this barrier and derive closed-form expressions for the log-pseudo-determinant and its first and second order partial derivatives. These expressions are valid for arbitrary dimension and very fast to evaluate, which allows us to set up an efficient MCMC sampler with derivative-based Metropolis–Hastings (MH) updates for the smoothing parameters. We derive simple formulas for low-dimensional slices and averages to facilitate visualization and investigate hyperprior sensitivity. We show that our new approach outperforms previous suggestions in the literature in terms of accuracy, scalability and computational cost and demonstrate its applicability through an illustrating temperature data example from spatio-temporal statistics.

Keywords Functional ANOVA decomposition · Kronecker sum · Markov chain Monte Carlo · Multivariate smoothing · Penalized splines · Spatio-temporal statistics

1 Introduction

There are numerous settings in statistics where measurements $(\boldsymbol{x}_i, y_i) \in \mathbb{R}^p \times \mathbb{R}$, $i = 1, \ldots, n$, $p \geq 2$, are available and a smooth surface estimate $\widehat{f}(\boldsymbol{x})$ with varying degree of smoothness in each dimension $1 \leq j \leq p$ is required. One example which we use for illustration later on is from spatio-temporal statistics: Here, the y_i are noisy temperature measurements and the $\boldsymbol{x}_i$ contain spatio-temporal information about these measurements. A smooth surface estimate allows one to predict the temperature at locations and time points where no measurements are available and to gain general insights into the spatio-temporal temperature dynamics. For this example, it is highly desirable to allow not only for a different amount of smoothing for the temporal dimension but also across the spatial dimensions. This is because the temperature profile cannot necessarily be assumed to be comparably smooth in the north–south direction (across different latitudes) as in the east–west direction (across different longitudes) due to varying climatological gradients.

A general key distinction in the context of multidimensional smoothing is that between isotropic and anisotropic smoothing: Isotropic smoothing means that there is a single smoothing parameter and that every coordinate receives the same amount of smoothing. Anisotropic smoothing, in contrast, means that there are $p = \dim(\boldsymbol{x}_i)$ smoothing parameters and that every coordinate receives its own amount of smoothing. The latter is generally desirable but much more challenging from a computational point of view.

Until recently, the popular Bayesian P-splines approach of Lang and Brezger (2004) has been limited to isotropic smoothing. The main challenge to achieve anisotropic smoothing is the MCMC update of the smoothing parameters. This is because their full conditional posterior comprises a pseudo-determinant that appears to be intractable at first sight.

✉ Nadja Klein
nadja.klein@kit.edu

[1] Department of Statistics, TU Dortmund University, Dortmund, Germany

[2] Scientific Computing Center, Karlsruhe Institute of Technology, Zirkel 2, 76131 Karlsruhe, Germany

Published online: 05 February 2025

As a consequence, existing implementations are unsatisfactory, either because of prohibitive runtimes or because they only allow for partially anisotropic smoothing: Wood (2016) introduced the function `jagam`, which allows for a seamless combination of the R package `mgcv` (Wood 2012) and the general purpose MCMC sampler `JAGS` (Plummer 2003). This approach works well for a two-dimensional tensor product smooth but it becomes extremely slow for dimension three or higher. The R package `bamlss` by Umlauf et al. (2018) also allows for anisotropic Bayesian smoothing and has been applied by Köhler et al. (2018) to estimate a two-dimensional tensor product smooth in a biomedical context. However, `bamlss` uses slice sampling with a stepping-out procedure (Neal 2003) to update the smoothing parameters. Similar to `jagam`, this becomes extremely slow for dimension three or higher. Kneib et al. (2019) introduced an alternative approach that relies on a discrete anisotropy parameter. This approach is implemented in `BayesX` (Belitz et al. 2015) and much faster than those of `bamlss` or `jagam` for a three-dimensional smooth. However, the approach breaks down for a four-dimensional smooth and, in addition to that, it only allows for partially anisotropic smoothing. Kneib et al. (2019) partition the coordinates into two groups which are both treated isotropically. This leads to inferior performance in simulations but is also unsatisfactory from a practical perspective. In a spatio-temporal context, for instance, the approach allows to treat space and time anisotropically but it does not allow for a different amount of smoothing across the spatial dimensions.

To the best of our knowledge, `Stan` (Carpenter et al. 2017) currently also does not offer a satisfactory solution. Approaches that implement tensor product P-splines using the `mgcv` constructor `te` do not seem to be readily available. The popular R package `rstanarm` (Goodrich et al. 2022), for instance, only supports the alternative constructor `t2` based on Wood et al. (2013), which uses a different roughness penalty. Wood et al. (2013) have shown that the alternative penalty is comparable in terms of estimation accuracy. We can confirm this result but we found that `rstanarm` becomes numerically unstable for a three-dimensional tensor product smooth and extremely slow for dimension four or higher.

The lack of efficient fully Bayesian approaches for anisotropic multidimensional smoothing stands in sharp contrast to tensor product spline smoothers that use restricted maximum likelihood (REML) for the selection of the smoothing parameters. Several efficient approaches have been developed (Wood 2011; Rodríguez-Álvarez et al. 2015; Wood and Fasiolo 2017) and are readily available in R packages such as `mgcv`. The fully Bayesian approach, however, has the advantage that the uncertainty of the smoothing parameters is taken into account in the estimation process and, in addition to that, it is relatively straightforward to incorporate various complications such as heteroscedasticity or missing data into the fully Bayesian approach (cf. Harezlak et al. 2018, Section 6.9).

In this paper, we close this gap and introduce a highly efficient fully Bayesian approach for anisotropic multidimensional smoothing. To overcome the obstacle posed by the pseudo-determinant we exploit a special representation of the anisotropic roughness penalty matrix. This representation is closely related to the mixed model representation of tensor product smooths (Wood 2006; Lee and Durbán 2011; Rodríguez-Álvarez et al. 2015) and allows us to derive closed-form expressions for the log-pseudo-determinant and its partial derivatives. These expressions are very fast to evaluate which allows us to set up an efficient MCMC sampler with derivative-based MH proposals for the smoothing parameters. In summary, our work makes the following major contributions:

- We introduce a highly efficient fully Bayesian approach for anisotropic multidimensional smoothing using Bayesian tensor product P-splines. Our approach allows for a different amount of smoothing for every coordinate and works well in estimating a function that depends on up to five coordinates.
- We derive efficient derivative-based MH updates for the smoothing parameters and show that our resulting algorithm outperforms previous suggestions in the literature by means of simulations. Our new approach is much faster and yields better performance in terms of mean squared error (MSE).
- We derive simple formulas for low-dimensional slices and averages facilitating the visualization of an estimated tensor product smooth of dimension three or higher.
- We demonstrate the applicability of our new approach by consideration of a temperature data set with $n = 12,672$ observations in a three-dimensional space-time setting.

The remainder of this paper is organized as follows: In Sect. 2 we introduce anisotropic multidimensional smoothing using Bayesian tensor product P-splines, whereas Sect. 3 details our new approach for efficient posterior sampling. In Sect. 4 we derive simple formulas for low-dimensional slices and averages. Section 5 presents empirical evidence and Sect. 6 concludes with a discussion. The supplement contains several appendices with technical details, proofs of our theoretical results, background information for the temperature data set as well as another real data example.

2 Bayesian anisotropic P-spline model

Throughout, we consider the p-dimensional nonparametric regression model

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i,\ \epsilon_i \overset{iid}{\sim} N_1(0, \sigma^2),\ i = 1, \ldots, n, \tag{1}$$

where $f : [0,1]^p \longrightarrow \mathbb{R}$ is an unknown function to be estimated and $N_1(0, \sigma^2)$ denotes the univariate normal distribution with mean zero and variance σ^2. Without loss of generality we assume that $\boldsymbol{x}_i \in [0,1]^p,\ i = 1, \ldots, n$, which can always be achieved through a simple linear transformation. Furthermore, we assume that the unknown function f can be approximated by tensor product B-splines, i.e.,

$$f(\boldsymbol{x}) \approx \sum_{j=1}^{D} B_j(\boldsymbol{x})\beta_j, \tag{2}$$

where $\boldsymbol{\beta} \in \mathbb{R}^D$ is an unknown coefficient vector to be estimated. Moreover, the $B_j(\boldsymbol{x}),\ j = 1, \ldots, D$, are tensor product B-splines of the form

$$\begin{aligned} B_1(\boldsymbol{x}) &= \prod_{j=1}^{p} \widetilde{B}_1(x_j), \\ B_2(\boldsymbol{x}) &= \prod_{j=1}^{p-1} \widetilde{B}_1(x_j)\widetilde{B}_2(x_p),\ \ldots\ , \\ B_D(\boldsymbol{x}) &= \prod_{j=1}^{p} \widetilde{B}_{d_j}(x_j), \quad \boldsymbol{x} \in [0,1]^p. \end{aligned}$$

Thereby, the p marginal bases $\{\widetilde{B}_1(x_1), \ldots, \widetilde{B}_{d_1}(x_1)\}, \ldots, \{\widetilde{B}_1(x_p), \ldots, \widetilde{B}_{d_p}(x_p)\}$ are cubic B-spline bases of dimensions $d_j \geq 4,\ j = 1, \ldots, p$, each covering the unit interval $[0,1]$. Following the Bayesian P-splines approach of Lang and Brezger (2004), we use a relatively large number of equidistant spline knots for the marginal B-spline bases. To prevent overfitting, we endow the tensor product B-spline coefficient vector $\boldsymbol{\beta} \in \mathbb{R}^D$ with a smoothness prior to be detailed in Sect. 2.1 below. The basis expansion (2) allows us to express the nonparametric regression model (1) in the form of a multiple linear regression model

$$\boldsymbol{y} = \boldsymbol{B}\boldsymbol{\beta} + \boldsymbol{\epsilon},\ \boldsymbol{\epsilon} \sim N_n(\boldsymbol{0}, \sigma^2\boldsymbol{I}_n),$$

where $\boldsymbol{y} = (y_1, \ldots, y_n)^\top$ is the n-dimensional vector of observations, $\boldsymbol{B}$ is the $n \times D$ tensor product B-spline design matrix, and $N_n(\boldsymbol{0}, \sigma^2\boldsymbol{I}_n)$ denotes the n-variate normal distribution with mean vector zero and covariance matrix $\sigma^2\boldsymbol{I}_n$. The overall dimension $D = \prod_{j=1}^{p} d_j$ of the tensor product spline space and thus the dimension of the coefficient vector $\boldsymbol{\beta}$ can be very large (see Table 1).

Table 1 Shown is the overall dimension $D = \prod_{j=1}^{p} d_j$ of the tensor product spline space for different numbers of covariates $p = \dim(\boldsymbol{x}_i)$ when using five-dimensional ($d_j = 5$) or ten-dimensional ($d_j = 10$) marginal B-spline bases

D	$p = 2$	$p = 3$	$p = 4$	$p = 5$
$d_j = 5$	25	125	625	3,125
$d_j = 10$	100	1,000	10,000	100,000

Example 2.1 To clarify the construction of the tensor product B-splines and the roles of p, d_j and D, consider the example of two-dimensional smoothing, where $p = \dim(\boldsymbol{x}_i) = 2$. We fix the dimensions $d_1 = 10$ and $d_2 = 10$ of the marginal B-spline bases and denote them by $\{\widetilde{B}_1(x_1), \ldots, \widetilde{B}_{10}(x_1)\}$ and $\{\widetilde{B}_1(x_2), \ldots, \widetilde{B}_{10}(x_2)\}$, respectively. With this, there are $D = d_1 d_2 = 10^2 = 100$ tensor product B-splines, which are simply all possible products of the marginal B-splines. The first tensor product B-spline is $B_1(\boldsymbol{x}) = \widetilde{B}_1(x_1)\widetilde{B}_1(x_2)$ and the second tensor product B-spline is $B_2(\boldsymbol{x}) = \widetilde{B}_1(x_1)\widetilde{B}_2(x_2)$. The third tensor product B-spline is $B_3(\boldsymbol{x}) = \widetilde{B}_1(x_1)\widetilde{B}_3(x_2)$ and so on. Finally, the 100th tensor product B-spline is $B_{100}(\boldsymbol{x}) = \widetilde{B}_{10}(x_1)\widetilde{B}_{10}(x_2)$; see Fig. 1 for an illustration.

2.1 Anisotropic smoothness prior

To obtain a smooth estimate $\widehat{f}$, we introduce a vector $\boldsymbol{\tau}^2 = (\tau_1^2, \ldots, \tau_p^2)^\top$ of positive smoothing variances and endow the tensor product B-spline coefficients with the partially improper Gaussian prior

$$p(\boldsymbol{\beta} \mid \boldsymbol{\tau}^2) \propto Det(\boldsymbol{K}(\boldsymbol{\tau}^2))^{1/2} \exp\left(-\frac{1}{2}\boldsymbol{\beta}^\top \boldsymbol{K}(\boldsymbol{\tau}^2)\boldsymbol{\beta}\right),\ \boldsymbol{\beta} \in \mathbb{R}^D. \tag{3}$$

Thereby, $Det(\cdot)$ is the pseudo-determinant (also known as generalized determinant), which is defined as the product of nonzero eigenvalues and $\boldsymbol{K}(\boldsymbol{\tau}^2)$ is the anisotropic roughness penalty matrix of the form

$$\boldsymbol{K}(\boldsymbol{\tau}^2) = \frac{\boldsymbol{K}_1}{\tau_1^2} + \cdots + \frac{\boldsymbol{K}_p}{\tau_p^2}, \tag{4}$$

where $\boldsymbol{K}_j = \boldsymbol{I}_{d_1} \otimes \cdots \otimes \boldsymbol{I}_{d_{j-1}} \otimes \widetilde{\boldsymbol{K}}_j \otimes \boldsymbol{I}_{d_{j+1}} \otimes \cdots \otimes \boldsymbol{I}_{d_p},\ j = 1, \ldots, p$, and $\otimes$ denotes the Kronecker product (cf. Eilers and Marx 2003). Furthermore, $\widetilde{\boldsymbol{K}}_j \in \mathbb{R}^{d_j \times d_j}$ are second order difference penalty matrices corresponding to the j-th marginal B-spline bases (see Lang and Brezger 2004, for details). By using an entire vector $\boldsymbol{\tau}^2 = (\tau_1^2, \ldots, \tau_p^2)^\top$ instead of a single smoothing variance in (3) and (4), we allow for a different amount of smoothing for each coordinate $x_j,\ j = 1, \ldots, p$, which is crucial to achieve satisfactory estimation accuracy (Rodríguez-Álvarez et al. 2015).

Motivated by their popularity in the context of additive models, we consider two different choices for the hyperprior

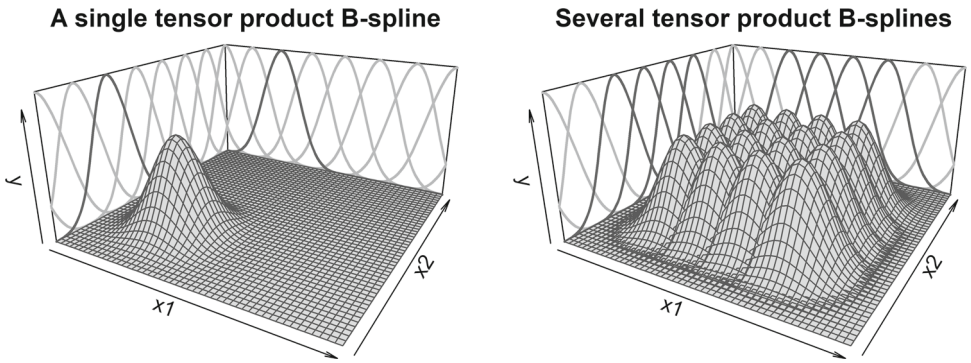

Fig. 1 Two-dimensional tensor product B-splines. The tensor product B-splines are defined as products of the marginal B-splines. The left plot shows a single tensor product B-spline, while the right plot shows several tensor product B-splines. The marginal B-splines are shown in the background

$p(\boldsymbol{\tau}^2)$ of the smoothing variances. First, we consider independent inverse gamma priors

$$\tau_j^2 \sim IG(a_j, b_j), \quad j = 1, \ldots, p, \tag{5}$$

and second, Weibull priors with shape 1/2 (cf. Klein and Kneib 2016), that is,

$$\tau_j^2 \sim Weibull(1/2, \lambda_j), \quad j = 1, \ldots, p. \tag{6}$$

To complete the prior specification, we place the Jeffreys' prior $p(\sigma^2) \propto 1/\sigma^2$ on the unknown residual variance σ^2.

Remark 2.2 The second order difference penalty matrices $\widetilde{\boldsymbol{K}}_j$ are commonly used for Bayesian P-splines estimation of univariate component functions in additive models (Lang and Brezger 2004). The main idea of the approach is to combine relatively many B-splines with a smoothness prior to prevent overfitting. The second order difference penalty matrices $\widetilde{\boldsymbol{K}}_j$ correspond to a second order random walk on the B-spline coefficients. The random walk introduces correlation among the coefficients, which prevents overly wiggly function estimates. It is well-known that the $\widetilde{\boldsymbol{K}}_j$ do not penalize linear functions. As a consequence, the $\widetilde{\boldsymbol{K}}_j$ only have rank $d_j - 2$ and two eigenvalues are equal to zero.

The anisotropic smoothness prior (3) extends the Bayesian P-splines approach to multivariate function estimation with tensor product B-splines. The smoothing variances τ_j^2 control the amount of wiggliness in the different coordinate directions. The anisotropic penalty matrix (4) does not penalize functions of the form $x_1^{j_1} \ldots x_p^{j_p}$ with $j_1, \ldots, j_p \in \{0, 1\}$. As a consequence, the matrix $\boldsymbol{K}(\boldsymbol{\tau}^2)$ is singular and 2^p eigenvalues are zero. Hence, (3) is not a regular Gaussian prior but a partially improper Gaussian prior. For these priors it is common to use the pseudo-determinant instead of the usual determinant (see, e.g., Rue and Held 2005). This is because the usual determinant $det(\cdot)$ vanishes for singular matrices. In the present context, for instance, we have $det(\boldsymbol{K}(\boldsymbol{\tau}^2)) = 0$ for all $\boldsymbol{\tau}^2 \in (0, \infty)^p$.

3 Efficient posterior sampling

In this section we derive a highly efficient MCMC sampler that avoids the limitations of the existing implementations mentioned in the introduction. To this end, first note that by Bayes' rule the joint posterior of $(\boldsymbol{\beta}, \boldsymbol{\tau}^2, \sigma^2) \in \mathbb{R}^D \times (0, \infty)^p \times (0, \infty)$ is proportional to

$$p(\boldsymbol{\beta}, \boldsymbol{\tau}^2, \sigma^2 \mid \boldsymbol{y}) \propto p(\boldsymbol{y} \mid \boldsymbol{\beta}, \sigma^2)\, p(\boldsymbol{\beta} \mid \boldsymbol{\tau}^2)\, p(\boldsymbol{\tau}^2)\, p(\sigma^2),$$

which does not correspond to a known probability distribution. Therefore, we use MCMC methods to sample from the posterior. In what follows, we first address the MCMC update of the tensor product B-spline coefficients $\boldsymbol{\beta}$ and the residual variance σ^2. Then we address the MCMC update of the vector of smoothing variances $\boldsymbol{\tau}^2$.

3.1 Updating the regression coefficients and residual variance

The full conditional posterior of the tensor product B-spline coefficients is a D-variate Gaussian distribution

$$\boldsymbol{\beta} \mid \boldsymbol{\tau}^2, \sigma^2, \boldsymbol{y} \sim N_D\left([\boldsymbol{B}^\top \boldsymbol{B}/\sigma^2 + \boldsymbol{K}(\boldsymbol{\tau}^2)]^{-1} \boldsymbol{B}^\top \boldsymbol{y}/\sigma^2, \; [\boldsymbol{B}^\top \boldsymbol{B}/\sigma^2 + \boldsymbol{K}(\boldsymbol{\tau}^2)]^{-1}\right), \tag{7}$$

which is straightforward to sample from. To increase computational efficiency, we use a sparse Cholesky decomposition of the precision matrix $\boldsymbol{B}^\top \boldsymbol{B}/\sigma^2 + \boldsymbol{K}(\boldsymbol{\tau}^2)$, which we combine with a blockwise updating scheme if the dimension D is very large (see Web Appendix A for further details). The full conditional posterior of the residual variance is inverse gamma

$$\sigma^2 \mid \boldsymbol{\beta}, \boldsymbol{y} \sim IG\left(n/2, \|\boldsymbol{y} - \boldsymbol{B}\boldsymbol{\beta}\|_2^2/2\right), \tag{8}$$

which is straightforward to sample from. A derivation of the full conditional posterior distributions (7) and (8) is provided in Web Appendix A.

3.2 Updating the smoothing variances

The full conditional posterior of the vector of smoothing variances $\boldsymbol{\tau}^2$ is proportional to

$$p(\boldsymbol{\tau}^2 \mid \boldsymbol{\beta}) \propto Det(\boldsymbol{K}(\boldsymbol{\tau}^2))^{1/2} \exp\left(-\frac{1}{2}\boldsymbol{\beta}^\top \boldsymbol{K}(\boldsymbol{\tau}^2)\boldsymbol{\beta}\right) p(\boldsymbol{\tau}^2), \tag{9}$$

which does not correspond to a known probability distribution, irrespective of the choice of the prior $p(\boldsymbol{\tau}^2)$. As a consequence, one cannot use Gibbs steps for the vector of smoothing variances $\boldsymbol{\tau}^2$. Moreover, deriving efficient MH updates for $\boldsymbol{\tau}^2$ is challenging because any MH update necessarily involves the repeated computation of the pseudo-determinant

$$Det(\boldsymbol{K}((\boldsymbol{\tau}^2)^*)) \tag{10}$$

at a proposed value $(\boldsymbol{\tau}^2)^*$, which generally has a high computational burden: The most obvious approach to compute the pseudo-determinant (10) is to perform an eigendecomposition of the $D \times D$ penalty matrix $\boldsymbol{K}((\boldsymbol{\tau}^2)^*)$. However, despite the sparsity of the penalty matrix, the eigendecomposition has computational complexity $\mathcal{O}(D^3)$. Therefore, a naive MH update of $\boldsymbol{\tau}^2$ is in fact much more expensive than the update of $\boldsymbol{\beta} \in \mathbb{R}^D$ even though $p \ll D$. To address this challenge, we exploit the following simple yet previously unrecognized expressions for the anisotropic penalty matrix and its pseudo-determinant.

3.2.1 Simple expressions for the penalty matrix and its determinant

Theorem 3.1 *[Penalty matrix decomposition] Let* $\widetilde{\boldsymbol{K}}_j = \widetilde{\boldsymbol{Q}}_j \widetilde{\boldsymbol{\Gamma}}_j \widetilde{\boldsymbol{Q}}_j^\top$, $j = 1, \ldots, p$, *be eigendecompositions of the marginal penalty matrices. Let* $\boldsymbol{Q} = \widetilde{\boldsymbol{Q}}_1 \otimes \cdots \otimes \widetilde{\boldsymbol{Q}}_p$ *and* $\boldsymbol{\Gamma}_j = \boldsymbol{I}_{d_1} \otimes \cdots \otimes \boldsymbol{I}_{d_{j-1}} \otimes \widetilde{\boldsymbol{\Gamma}}_j \otimes \boldsymbol{I}_{d_{j+1}} \otimes \cdots \otimes \boldsymbol{I}_{d_p}$, $j = 1, \ldots, p$. *Then, for all* $\boldsymbol{\tau}^2 \in (0, \infty)^p$, *it holds:*

$$\boldsymbol{K}(\boldsymbol{\tau}^2) = \boldsymbol{Q}\left(\frac{\boldsymbol{\Gamma}_1}{\tau_1^2} + \cdots + \frac{\boldsymbol{\Gamma}_p}{\tau_p^2}\right)\boldsymbol{Q}^\top. \tag{11}$$

Theorem 3.1 follows from the definition of the anisotropic roughness penalty matrix (4) and the properties of the Kronecker product. A detailed proof is provided in Web Appendix B. Theorem 3.1 implies the following convenient expression for the log-pseudo-determinant.

Corollary 3.2 (Log-determinant) *Let* $\gamma_{j,l}$, $j = 1, \ldots, p$, $l = 1, \ldots, D$, *denote the diagonal entries of the* $\boldsymbol{\Gamma}_j$, $j = 1, \ldots, p$, *and let the set* $\mathcal{D}^+ \subseteq \{1, \ldots, D\}$ *contain those indices for which at least one* $\boldsymbol{\Gamma}_j$ *has a positive diagonal entry, i.e.,* $l \in \mathcal{D}^+ \iff \exists\, j \in \{1, \ldots, p\} : \gamma_{j,l} > 0$. *Then, for all* $\boldsymbol{\tau}^2 \in (0, \infty)^p$, *it holds:*

$$\begin{aligned}\log Det(\boldsymbol{K}(\boldsymbol{\tau}^2)) &= \log Det\left(\frac{\boldsymbol{\Gamma}_1}{\tau_1^2} + \cdots + \frac{\boldsymbol{\Gamma}_p}{\tau_p^2}\right) \\ &= \sum_{l \in \mathcal{D}^+} \log\left(\frac{\gamma_{1,l}}{\tau_1^2} + \cdots + \frac{\gamma_{p,l}}{\tau_p^2}\right).\end{aligned} \tag{12}$$

A proof of Corollary 3.2 is provided in Web Appendix B. Corollary 3.2 reduces the computational cost of the pseudo-determinant (10) from cubic complexity $\mathcal{O}(D^3)$ to linear complexity $\mathcal{O}(D)$. As a consequence, the evaluation of the full conditional posterior (9) becomes much cheaper and efficient MH updates for $\boldsymbol{\tau}^2$ become feasible.

3.2.2 Taylored MH updates

A key feature of our new method is to exploit expression (12) to derive efficient derivative-based MH updates for the smoothing parameters. The basic idea of these updates, known as Taylored or iteratively weighted least squares (IWLS) updates in the literature (cf. Geweke and Tanizaki 2003; Klein and Kneib 2016), is to approximate the target density locally by a multivariate Gaussian density. In the present context, the target density is the full conditional posterior of the log-smoothing variances. We work with the log-smoothing variances $\rho_j = \log(\tau_j^2)$, $j = 1, \ldots, p$, as these are unconstrained.

By the density transformation formula and (9), the full conditional posterior of the log-smoothing variances $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_p)^\top \in \mathbb{R}^p$ is proportional to

$$p(\boldsymbol{\rho} \mid \boldsymbol{\beta}) \propto Det(\boldsymbol{K}(e^{\boldsymbol{\rho}}))^{1/2} \exp\left(-\frac{1}{2}\boldsymbol{\beta}^\top \boldsymbol{K}(e^{\boldsymbol{\rho}})\boldsymbol{\beta}\right) q(\boldsymbol{\rho}),$$

where $q(\boldsymbol{\rho}) \propto p(e^{\boldsymbol{\rho}}) \prod_{j=1}^p e^{\rho_j}$ is a kernel of the prior of the log-smoothing variances $\boldsymbol{\rho}$ and $e^{\boldsymbol{\rho}} = (e^{\rho_1}, \ldots, e^{\rho_p})^\top$. By Corollary 3.2, the log-full conditional posterior $\log p(\boldsymbol{\rho} \mid \boldsymbol{\beta})$ is up to an irrelevant additive constant equal to

$$\begin{aligned}&\frac{1}{2}\sum_{l \in D^+} \log\left(\gamma_{1,l}e^{-\rho_1} + \cdots + \gamma_{p,l}e^{-\rho_p}\right) \\ &\quad - \frac{1}{2}\boldsymbol{\beta}^\top \boldsymbol{K}(e^{\boldsymbol{\rho}})\boldsymbol{\beta} + \log q(\boldsymbol{\rho}).\end{aligned} \tag{13}$$

To employ the MH updates, we need the gradient vector $\boldsymbol{u}(\boldsymbol{\rho})$ and the Hessian matrix $\boldsymbol{H}(\boldsymbol{\rho})$ of the log-full conditional posterior (13). The corresponding first and second order partial derivatives are stated in the subsequent Proposition 3.3.

Proposition 3.3 (Partial derivatives) *For $\boldsymbol{\rho} \in \mathbb{R}^p$ it holds*

$$\begin{aligned}
\partial_{j_0} \log p(\boldsymbol{\rho} \mid \boldsymbol{\beta}) &= -\frac{1}{2} \sum_{l \in D^+} \frac{\gamma_{j_0,l} e^{-\rho_{j_0}}}{\gamma_{1,l} e^{-\rho_1} + \cdots + \gamma_{p,l} e^{-\rho_p}} \\
&\quad + \frac{1}{2} \boldsymbol{\beta}^\top \boldsymbol{K}_{j_0} \boldsymbol{\beta}\, e^{-\rho_{j_0}} + \partial_{j_0} \log q(\boldsymbol{\rho}), \\
\partial^2_{j_0} \log p(\boldsymbol{\rho} \mid \boldsymbol{\beta}) &= -\frac{1}{2} \sum_{l \in D^+} \left\{ \left(\frac{\gamma_{j_0} e^{-\rho_{j_0}}}{\gamma_{1,l} e^{-\rho_1} + \cdots + \gamma_{p,l} e^{-\rho_p}} \right)^2 \right. \\
&\quad \left. - \frac{\gamma_{j_0} e^{-\rho_{j_0}}}{\gamma_{1,l} e^{-\rho_1} + \cdots + \gamma_{p,l} e^{-\rho_p}} \right\} \\
&\quad - \frac{1}{2} \boldsymbol{\beta}^\top \boldsymbol{K}_{j_0} \boldsymbol{\beta}\, e^{-\rho_{j_0}} + \partial^2_{j_0} \log q(\boldsymbol{\rho}), \\
\partial_{j_0} \partial_{k_0} \log p(\boldsymbol{\rho} \mid \boldsymbol{\beta}) &= -\frac{1}{2} \sum_{l \in D^+} \frac{\gamma_{j_0,l} e^{-\rho_{j_0}} \gamma_{k_0,l} e^{-\rho_{k_0}}}{(\gamma_{1,l} e^{-\rho_1} + \cdots + \gamma_{p,l} e^{-\rho_p})^2} \\
&\quad + \partial_{j_0} \partial_{k_0} \log q(\boldsymbol{\rho}), \quad \text{if } j_0 \neq k_0.
\end{aligned}$$

We omit a proof of Proposition 3.3 as the results follow directly from (13) and elementary rules of differentiation like the chain rule. While the expressions in Proposition 3.3 appear complicated at first sight, it is important to realize that they can be evaluated very efficiently. Next we explain how these expressions can be used to generate the MH updates for the log-smoothing variances $\boldsymbol{\rho}$. Let therefore $\boldsymbol{\beta}^{(t+1)} \in \mathbb{R}^D$ and $\boldsymbol{\rho}^{(t)} \in \mathbb{R}^p$ be the current values of the the B-spline coefficients and the log-smoothing variances, respectively. Then, a single MH step for $\boldsymbol{\rho}$ goes as follows:

1. Generate the MH proposal $\boldsymbol{\rho}^*$ from the p-variate Gaussian distribution

   $$N_p(\boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})$$

   with mean $\boldsymbol{\mu}^{(t)} = \boldsymbol{\rho}^{(t)} - \boldsymbol{H}^{-1}(\boldsymbol{\rho}^{(t)}) \boldsymbol{u}(\boldsymbol{\rho}^{(t)})$ and covariance matrix $\boldsymbol{\Sigma}^{(t)} = -\boldsymbol{H}^{-1}(\boldsymbol{\rho}^{(t)})$.
2. Compute the MH acceptance probability

   $$\alpha^* = \min \left\{ 1, \frac{p(\boldsymbol{\rho}^* \mid \boldsymbol{\beta}^{(t+1)}) N_p(\boldsymbol{\rho}^{(t)}; \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*)}{p(\boldsymbol{\rho}^{(t)} \mid \boldsymbol{\beta}^{(t+1)}) N_p(\boldsymbol{\rho}^*; \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})} \right\},$$

   where $N_p(\boldsymbol{\rho}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the density of a p-variate Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ evaluated at $\boldsymbol{\rho}$. Moreover, $\boldsymbol{\mu}^* = \boldsymbol{\rho}^* - \boldsymbol{H}^{-1}(\boldsymbol{\rho}^*) \boldsymbol{u}(\boldsymbol{\rho}^*)$ and $\boldsymbol{\Sigma}^* = -\boldsymbol{H}^{-1}(\boldsymbol{\rho}^*)$.
3. Set $\boldsymbol{\rho}^{(t+1)} = \boldsymbol{\rho}^*$ with probability α^* and $\boldsymbol{\rho}^{(t+1)} = \boldsymbol{\rho}^{(t)}$ with probability $1 - \alpha^*$.

To implement the MH updates for the two priors (5) and (6) we need the log-prior kernels $\log q(\boldsymbol{\rho})$, as well as the corresponding first and second order partial derivatives. These are provided in Web Appendix E.

3.2.3 Hessian modification

For the Taylored MH updates to be well-defined we need both Hessians $\boldsymbol{H}(\boldsymbol{\rho}^{(t)})$ and $\boldsymbol{H}(\boldsymbol{\rho}^*)$ to be negative definite (otherwise the MH acceptance probability α^* is not well-defined). To ensure that this is the case, we follow Section 3.4 of Nocedal and Wright (2006) in the context of Newton's method and modify the eigenvalues of the Hessian, if they are not already sufficiently small. To this end, we replace the eigenvalues λ_j, $j = 1, \ldots, p$, of the Hessian $\boldsymbol{H}$ by $\widetilde{\lambda}_j = \min(\lambda_j, -\delta)$, $j = 1, \ldots, p$, where $\delta > 0$ is a fixed positive constant that is chosen by the user. Denoting the modified Hessian matrix by $\widetilde{\boldsymbol{H}}$, we thus use the matrices $-\widetilde{\boldsymbol{H}}(\boldsymbol{\rho}^{(t)})^{-1}$ and $-\widetilde{\boldsymbol{H}}(\boldsymbol{\rho}^*)^{-1}$ in our MCMC scheme. Similar modifications of the Hessian are also common for REML based approaches (see, e.g., Wood 2011, Section 3).

By default we use $\delta = 1/\pi$ for the threshold, which ensures that the Hessians are negative definite and, in addition to that, limits the maximal step size to a reasonable range. To see the latter note that the modification confines the eigenvalues of $-\widetilde{\boldsymbol{H}}(\boldsymbol{\rho}^{(t)})^{-1}$ and $-\widetilde{\boldsymbol{H}}(\boldsymbol{\rho}^*)^{-1}$ to the interval $(0, 1/\delta]$. For $\delta = 1/\pi$, the eigenvalues are thus confined to the interval $(0, \pi]$, which is a reasonable range as we work on the log-scale. We also investigated different values and found that the precise value of δ is not important. Similar values like $\delta = 1/3$ or $\delta = 1/2$ work as well.

3.3 Algorithmic details

In summary, combining the MH steps for $\boldsymbol{\rho}$ with Gibbs steps for $\boldsymbol{\beta}$ and σ^2 from (7) and (8), respectively, we obtain a MCMC sample from the joint posterior $(\boldsymbol{\beta}, \boldsymbol{\tau}^2, \sigma^2) \mid \boldsymbol{y}$ as follows:

Step 0 Initialize $\boldsymbol{\beta}^{(0)}, \boldsymbol{\rho}^{(0)}, (\sigma^2)^{(0)}$. Compute $(\boldsymbol{\tau}^2)^{(0)} = \exp(\boldsymbol{\rho}^{(0)})$.
Then, for $t = 0, \ldots, T-1$ repeat

Step 1 Generate $\boldsymbol{\beta}^{(t+1)} \mid (\boldsymbol{\tau}^2)^{(t)}, (\sigma^2)^{(t)}, \boldsymbol{y}$.
Step 2 Generate $(\sigma^2)^{(t+1)} \mid \boldsymbol{\beta}^{(t+1)}, (\boldsymbol{\tau}^2)^{(t)}, \boldsymbol{y}$.
Step 3 Generate $\boldsymbol{\rho}^{(t+1)} \mid \boldsymbol{\beta}^{(t+1)}, (\sigma^2)^{(t+1)}, \boldsymbol{y}$. Compute $(\boldsymbol{\tau}^2)^{(t+1)} = \exp(\boldsymbol{\rho}^{(t+1)})$.

Return $\{\boldsymbol{\beta}^{(1)}, (\boldsymbol{\tau}^2)^{(1)}, (\sigma^2)^{(1)}, \ldots, \boldsymbol{\beta}^{(T)}, (\boldsymbol{\tau}^2)^{(T)}, (\sigma^2)^{(T)}\}$ optionally omitting burn-in samples.

4 Visualization through slices and averages

The MCMC sampler introduced in the previous section allows for efficient Bayesian estimation of a tensor product smooth $\widehat{f} = \widehat{f}(x_1, \ldots, x_p)$ of moderate dimension

$p \in \{2, 3, 4, 5\}$. An important question in practice is how such a smooth can be visualized for $p \geq 3$. This is not completely obvious because for $p \geq 3$ the function graph cannot be plotted anymore. One option facilitating visualization are slice plots. Thereby, we fix some of the coordinates and regard $\widehat{f}$ as a function of the remaining coordinates only (see, e.g., Friedman 1991). Another option that is closely related to the functional ANOVA decomposition (Gu 2013) are plots of low-dimensional averages, where we integrate some of the coordinates out.

In the following we derive simple formulas for low-dimensional slices and averages using Kronecker calculus. While the derivation of these formulas is relatively straightforward, we were unable to find them in the literature. To facilitate the exposition, it will be convenient to first introduce additional notation. Let

$$\vec{\boldsymbol{B}}_j(x_j) = (\widetilde{B}_1(x_j), \ldots, \widetilde{B}_{d_j}(x_j))^\top \in \mathbb{R}^{d_j}, \ j = 1, \ldots, p,$$

denote the column vectors of marginal B-spline basis function evaluations. Let further

$$\vec{\boldsymbol{B}}(\boldsymbol{x}) = \vec{\boldsymbol{B}}_1(x_1) \otimes \cdots \otimes \vec{\boldsymbol{B}}_p(x_p) \in \mathbb{R}^D$$

denote the column vector of tensor product B-spline evaluations. With this, we can express any tensor product spline f in the form

$$f(\boldsymbol{x}) = \vec{\boldsymbol{B}}(\boldsymbol{x})^\top \boldsymbol{\beta}, \ \boldsymbol{x} \in [0, 1]^p.$$

Proposition 4.1 *(Slices) Let $f(\boldsymbol{x}) = \vec{\boldsymbol{B}}(\boldsymbol{x})^\top \boldsymbol{\beta}$, $\boldsymbol{x} \in [0, 1]^p$, be a tensor product spline. Then it holds:*

i) Let $\boldsymbol{x}_{-j} = (x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_p)^\top \in [0, 1]^{p-1}$ be fixed. Then, the one-dimensional slice $x_j \mapsto f(\boldsymbol{x})$ is a spline with coefficient vector

$$\Big(\vec{\boldsymbol{B}}_1(x_1)^\top \otimes \cdots \otimes \vec{\boldsymbol{B}}_{j-1}(x_{j-1})^\top \otimes \boldsymbol{I}_{d_j} \otimes \vec{\boldsymbol{B}}_{j+1}(x_{j+1})^\top \cdots \otimes \vec{\boldsymbol{B}}_p(x_p)^\top\Big)\boldsymbol{\beta}.$$

ii) Let $\boldsymbol{x}_{-(j,k)} = (x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_{k-1}, x_{k+1}, \ldots, x_p)^\top \in [0, 1]^{p-2}$ be fixed. Then, the two-dimensional slice $(x_j, x_k) \mapsto f(\boldsymbol{x})$ is a tensor product spline with coefficient vector

$$\Big(\vec{\boldsymbol{B}}_1(x_1)^\top \otimes \cdots \otimes \vec{\boldsymbol{B}}_{j-1}(x_{j-1})^\top \otimes \boldsymbol{I}_{d_j} \otimes \vec{\boldsymbol{B}}_{j+1}(x_{j+1})^\top \otimes \cdots \otimes \vec{\boldsymbol{B}}_{k-1}(x_{k-1})^\top \otimes \boldsymbol{I}_{d_k} \otimes \vec{\boldsymbol{B}}_{k+1}(x_{k+1})^\top \otimes \cdots \otimes \vec{\boldsymbol{B}}_p(x_p)^\top\Big)\boldsymbol{\beta}.$$

A proof of Proposition 4.1 is provided in Web Appendix C. Similar formulas can be established for low-dimensional averages, where we integrate some of the coordinates out instead of fixing them to certain values. To this end, let

$$\begin{aligned}\boldsymbol{A}_j &= \int_0^1 \vec{\boldsymbol{B}}_j(x_j)dx_j \\ &= \left(\int_0^1 \widetilde{B}_1(x_j)dx_j, \ldots, \int_0^1 \widetilde{B}_{d_j}(x_j)dx_j\right)^\top \\ &\quad \in \mathbb{R}^{d_j}, \ j = 1, \ldots, p,\end{aligned}$$

denote column vectors containing the averages of the marginal B-splines.

Proposition 4.2 (Averages) *Let $f(\boldsymbol{x}) = \vec{\boldsymbol{B}}(\boldsymbol{x})^\top \boldsymbol{\beta}$, $\boldsymbol{x} \in [0, 1]^p$, be a tensor product spline. Then it holds:*

i) The one-dimensional average $x_j \mapsto \int_{[0,1]^{p-1}} f(\boldsymbol{x})d\boldsymbol{x}_{-j}$ is a spline with coefficient vector

$$\Big(\boldsymbol{A}_1^\top \otimes \cdots \otimes \boldsymbol{A}_{j-1}^\top \otimes \boldsymbol{I}_{d_j} \otimes \boldsymbol{A}_{j+1}^\top \cdots \otimes \boldsymbol{A}_p^\top\Big)\boldsymbol{\beta}.$$

ii) The two-dimensional average $(x_j, x_k) \mapsto \int_{[0,1]^{p-2}} f(\boldsymbol{x})d\boldsymbol{x}_{-(j,k)}$ is a tensor product spline with coefficient vector

$$\Big(\boldsymbol{A}_1^\top \otimes \cdots \otimes \boldsymbol{A}_{j-1}^\top \otimes \boldsymbol{I}_{d_j} \otimes \boldsymbol{A}_{j+1}^\top \otimes \cdots \otimes \boldsymbol{A}_{k-1}^\top \otimes \boldsymbol{I}_{d_k} \otimes \boldsymbol{A}_{k+1}^\top \otimes \cdots \otimes \boldsymbol{A}_p^\top\Big)\boldsymbol{\beta}.$$

A proof of Proposition 4.2 is provided in Web Appendix C.

Remark 4.3 The above results are highly useful. In our spatio-temporal temperature data application, for instance, Proposition 4.1 allows us to plot temperature curves for fixed locations and temperature surfaces for fixed time points. In addition, Proposition 4.2 allows us to plot the average temperature over time and space (see Fig. 4). To this end, we simply apply the corresponding formulas to the estimated coefficient vector $\widehat{\boldsymbol{\beta}}$. We can also obtain credible intervals by applying the formulas to the entire MCMC sample $\boldsymbol{\beta}^{(t)}$, $t = 1, \ldots, T$.

5 Empirical evidence

In this section we provide empirical evidence for our new approach. First we conduct a simulation study, then we consider a real data example. All computations were conducted in R on a regular desktop PC with 3.5 GHz and 32 GB RAM.

5.1 Simulation study

Our simulation study is divided into two parts. Part a) investigates the computational efficiency (runtime) of our MCMC sampler relative to the competitors `bamlss`, `BayesX`,

`jagam` and `rstanarm` for different combinations of the dimensions $p \in \{2, 3, 4, 5\}$ and $d_j \in \{5, 10\}$. Part b) focuses on estimation accuracy of our approach and the benchmark methods in terms of MSE when the true function is isotropic or anisotropic. For the simulations we use 1,200 MCMC iterations from which we discard 200 as burn-in. These are the default settings in the R package `bamlss` and we found them to be suitable. However, for real data applications we generally recommend a much larger number of MCMC iterations. For the temperature data example presented in the subsequent Sect. 5.2 we used for instance 100,000 MCMC iterations and 5,000 as burn-in.

Part a) Computational efficiency (runtime)

We consider the isotropic test function $f_1(\boldsymbol{x}) = \sin(2\pi \|\boldsymbol{x}\|_2)$, $\boldsymbol{x} \in [0, 1]^p$. We use a sample of size $n = 10^4$ and the residual variance σ^2 is set to $(1/2)^2$. The design points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are sampled iid and uniformly on the p-dimensional unit cube $[0, 1]^p$. We increase the dimension of the domain $p \in \{2, 3, 4, 5\}$ and record the time needed to generate 1, 200 MCMC samples for the following five methods:

- new-WB: Our new approach with iid unit rate Weibull priors for the smoothing variances, i.e. $\tau_j^2 \overset{iid}{\sim} Weibull(1/2, 1)$, $j = 1, \ldots, p$.
- bamlss: The function `bamlss` in the R package `bamlss` with sampler `sam_GMCMC`.
- BayesX: The function `bamlss` in the R package `bamlss` with sampler `sam_BayesX`.
- jagam: The function `jagam` in the R package `mgcv`.
- rstanarm: The function `stan_gamm4` in the R package `rstanarm`.

For all five methods we consider either $d_j = 5$ or $d_j = 10$ for the dimensions of the marginal B-spline bases. Table 2 reports the runtime for each of the competitors in minutes.

Conclusion. For a two-dimensional tensor product smooth ($p = 2$) our new approach is a few seconds slower than some of the competitors. However, for dimension $p = 3$ or higher, our new approach is magnitudes faster than previous fully Bayesian approaches allowing for anisotropic multidimensional smoothing. This is true for five-dimensional marginal bases ($d_j = 5$) and in particular for ten-dimensional marginal bases ($d_j = 10$).

Table 2 Runtime in minutes to generate 1,200 MCMC samples

	Method	$p = 2$	$p = 3$	$p = 4$	$p = 5$
$d_j = 5$	new-WB	0.34	**0.74**	**2.71**	**28.70**
	bamlss	1.19	6.34	151.27	> 600
	BayesX	0.57	1.51	> 600	
	jagam	**0.22**	4.23	346.83	> 600
	rstanarm	0.37	2.82	72.03	> 600
$d_j = 10$	new-WB	0.43	**2.70**	**26.32**	**259.53**
	bamlss	3.15	> 600	> 600	> 600
	BayesX	0.58	7.97	> 600	
	jagam	2.42	> 600	> 600	> 600
	rstanarm	**0.21**	24.62	> 600	> 600

The rows show the different methods, while across the columns, the dimension p of the tensor product smooth increases. The MCMC sampling was interrupted after 10 h (marked with > 600). Two values are missing in the rightmost column because `BayesX` currently does not support five-dimensional tensor product smooths.
The lowest runtime for each configuration is highlighted in bold

Part b) Estimation accuracy (MSE)

Next we fix the dimension of the domain $p = 3$ and thus only consider three-dimensional tensor product smooths. In addition to the isotropic test function f_1 we consider the anisotropic test function $f_2(\boldsymbol{x}) = \sin\left(2\pi\sqrt{3x_1^2 + x_2^2 + x_3^2/3}\right)$, $\boldsymbol{x} \in [0, 1]^3$. We vary the sample size $n \in \{100, 250, 500, 10^3, 10^4\}$ and compute the MSE given by $1/n \sum_{i=1}^n (\widehat{f}(\boldsymbol{x}_i) - f(\boldsymbol{x}_i))^2$. The design points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are sampled iid and uniformly on the three-dimensional unit cube $[0, 1]^3$ and the residual variance σ^2 is set to $(1/2)^2$ as before. We use 1, 200 MCMC iterations and discard the first 200 as burn-in. We consider our new approach with four different parameter settings:

i) Inverse gamma priors $\tau_j^2 \overset{iid}{\sim} IG(0.001, 0.001)$ and $d_j = 5$ (denoted by new-IG-5).
ii) Weibull priors $\tau_j^2 \overset{iid}{\sim} Weibull(1/2, 1)$ and $d_j = 5$ (denoted by new-WB-5).
iii) Weibull priors $\tau_j^2 \overset{iid}{\sim} Weibull(1/2, \lambda)$ with λ determined via prior scaling and $d_j = 5$ (denoted by new-WB-PS-5). The key idea of the prior scaling approach is to set λ such that prior function draws have a reasonable scale (see Web Appendix D for details).
iv) Weibull priors $\tau_j^2 \overset{iid}{\sim} Weibull(1/2, \lambda)$ with λ determined via prior scaling and $d_j = 10$ dimensional marginal bases (denoted by new-WB-PS-10).

As further competitors we consider `bamlss` and `jagam` with $d_j = 5$ as well as `BayesX` and `rstanarm` with $d_j = 5$ or $d_j = 10$. We do not include $d_j = 10$ for `bamlss` and `jagam` because of the excessive runtime established before (see Table 2). Figure 2 shows boxplots of the logarithmic MSE based on $R = 50$ replicates for each configuration of $f \in \{f_1, f_2\}$ and $n \in \{100, 250, 500, 10^3, 10^4\}$.

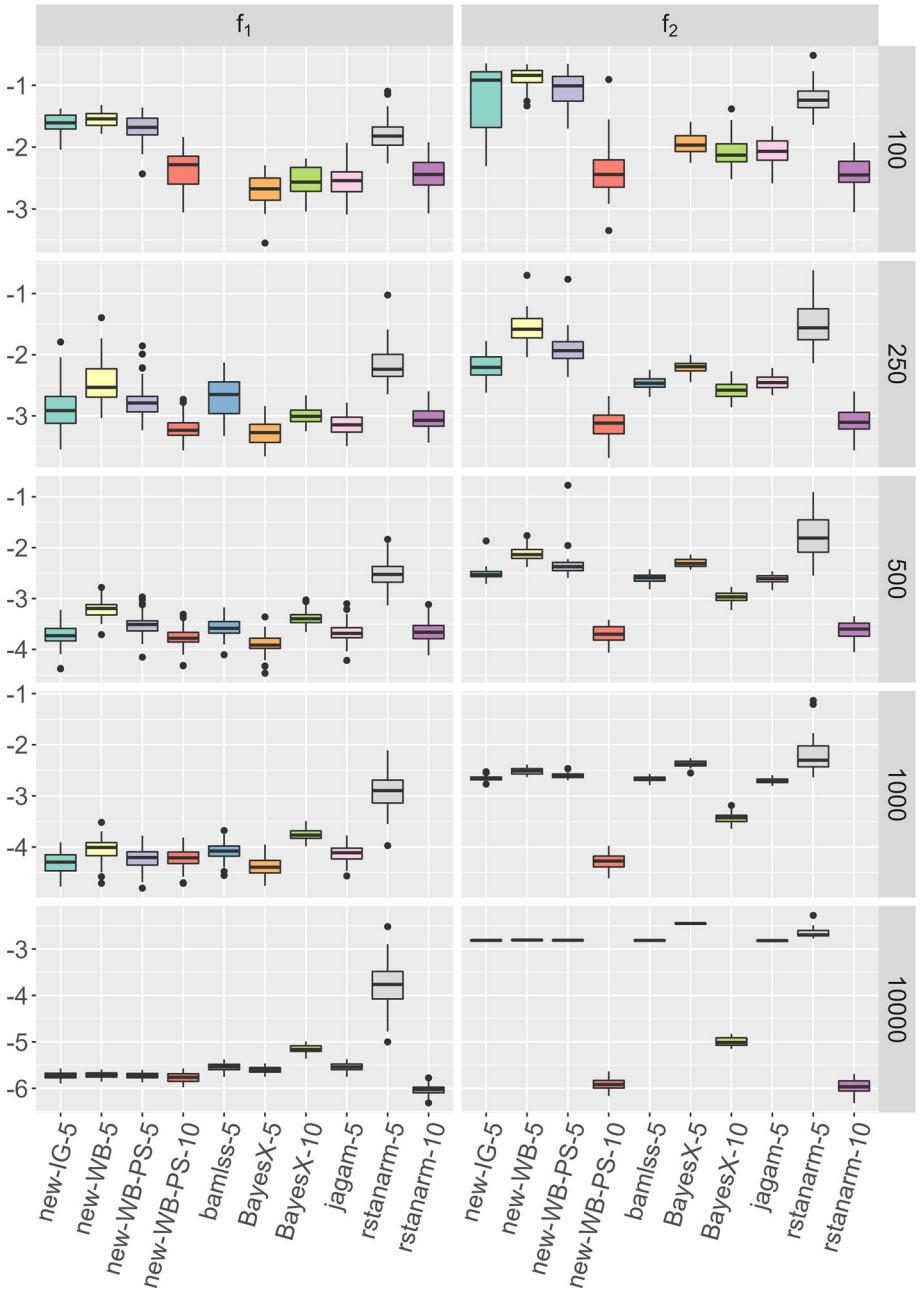

Fig. 2 Shown are the log MSEs for the different methods. The five plots on the left show the results for the isotropic test function f_1, the five plots on the right show the results for the anisotropic test function f_2. The rows show the results for the different sample sizes $n \in \{100, 250, 500, 10^3, 10^4\}$. The suffix in the labels indicates the dimension of the marginal B-spline bases, e.g. $d_j = 5$ for `rstanarm-5` and $d_j = 10$ for `rstanarm-10`

Conclusion. Altogether, parameter setting iv) with label "new-WB-PS-10" in Fig. 2 works best for our new approach, i.e., Weibull priors $\tau_j^2 \overset{iid}{\sim} \textit{Weibull}(1/2, \lambda)$ with λ determined via prior scaling and $d_j = 10$ dimensional marginal bases yield the best performance. Therefore, we opt for this setting as default. With this setting, we are slightly worse than some of the competitors for the isotropic test function f_1. However, we outperform most of the competitors for the anisotropic test function f_2. The only method that can keep up is `rstanarm-10`. However, we found `rstanarm-10` to be unstable in the sense that the MCMC sampler typically got stuck in the warm-up phase and did not enter the sampling phase for the sample size $n = 10^3$. This is also the reason why the corresponding boxes in the fourth row of Fig. 2 are missing for `rstanarm-10`. According to the documentation of the package `rstanarm`, it may be possible to get the sampler running properly, e.g. by adjusting the parameters `adapt_delta` or `max_treedepth`. However, our attempts were not successful. In contrast to that, we did not encounter any numerical issues for our new approach. Interestingly, the modification of the Hessian (see Sect. 3.2.3) was only necessary for the inverse gamma prior $\tau_j^2 \overset{iid}{\sim} IG(0.001, 0.001)$ but not for the Weibull prior. More specifically, the eigenvalue modification allowed us to avoid

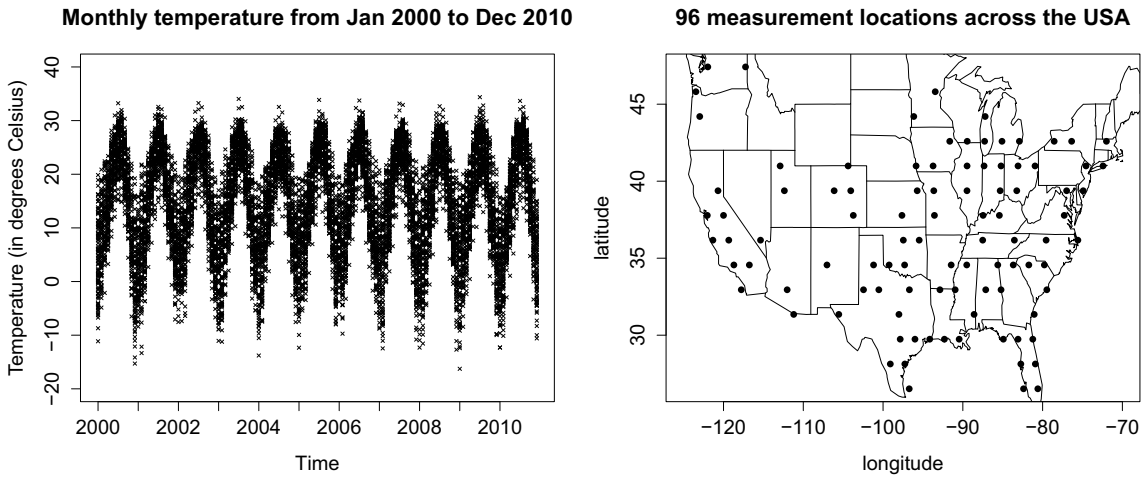

Fig. 3 Left: Monthly temperature from January 2000 to December 2010 for 96 measurement locations across the USA. Right: 96 measurement locations across the USA

numerical issues such as an indefinite Hessian in about 10% of the runs for the inverse gamma prior. For the Weibull prior, however, the modification was never exerted. This can be explained by the much lighter tails of the Weibull prior which ensure that the parameters stay within a reasonable range during MCMC sampling. The key message is that the Weibull prior offers better numerical stability compared to the inverse gamma prior. This finding is in line with the observations of others (see, e.g., Ghosh et al. 2018).

Overall summary. In summary, the simulation study shows that our new approach is much faster than previous Bayesian approaches allowing for anisotropic multidimensional smoothing. Moreover, the new approach is numerically stable and performs equally well or even better in terms of MSE.

5.2 Real data example

In this section we apply our new approach to analyze a publicly available temperature data set. The data set comprises $n = 12,672$ records of the monthly temperature from January 2000 to December 2010 for 96 measurement locations across the USA. The temperature data set is part of a large climate data base that was compiled by the Berkeley Earth project (`www.berkeleyearth.org`). Further information about the data set and our preprocessing steps are provided in Web Appendix F. Figure 3 visualizes the data set.

To gain insights into the spatio-temporal temperature dynamics, we consider the spatio-temporal model

$$temperature = f(time, longitude, latitude) + \epsilon,\ \epsilon \sim N_1(0, \sigma^2). \quad (14)$$

We model f as a three-dimensional tensor product smooth using the following parameters: We use $(d_1, d_2, d_3) = (40, 10, 10)$ for the dimensions of the marginal B-spline bases. The relatively high number of basis functions for the temporal dimension is necessary to capture the seasonal pattern visible in Fig. 3. The overall dimension of the tensor product spline space is thus $D = 40 \times 10 \times 10 = 4,000$. For the smoothing variances we use independent Weibull priors $\tau_j^2 \overset{iid}{\sim} Weibull(1/2, \lambda),\ j = 1, 2, 3$, where τ_1^2 refers to time, τ_2^2 refers to longitude and τ_3^2 refers to latitude. The rate parameter $\lambda \approx 38.37$ was determined via prior scaling (see Web Appendix D for details).

We ran the MCMC sampler introduced in Sect. 3 for $T = 100,000$ iterations and discarded the first 5,000 iterations as burn-in. Table 3 reports MCMC summaries as well as MCMC convergence diagnostics. Figure 4 shows selected functional effect estimates using the formulas from Sect. 4, while Fig. 5 shows trace plots for selected coefficients. The supplementary material contains a short video clip illustrating the estimated temperature dynamics using a daily temporal resolution.

Conclusion. From Fig. 4 we can see that the Rocky Mountains have a strong effect on the temperature and that the effect of the seasons is more pronounced in the north of the USA than in the south. Note that the latter finding could not be established using an additive model of the form $f_1(time) + f_2(longitude, latitude)$. This demonstrates the advantage of the more complex model (14) as it allows for a spatio-temporal interaction. From Fig. 5 and Table 3 we see that the posteriors of the log-smoothing variances differ significantly, which underlines the need for anisotropic estimation.

The runtime is acceptable with 100,000 MCMC iterations taking less than nine hours (`rstanarm`, for comparison,

Table 3 MCMC summaries and convergence diagnostics

	mean	median	sd	mad	q_5	q_{95}	$\widehat{R}$	ess (bulk)	ess (tail)
β_{1000}	2.21	2.23	25.76	25.80	−40.13	44.51	1.00	93588.60	94107.27
β_{2000}	−1.34	−1.28	25.76	25.77	−43.77	40.93	1.00	94636.39	93509.84
σ^2	2.64	2.64	0.04	0.04	2.58	2.71	1.00	11252.77	37410.87
ρ_1	4.81	4.80	0.11	0.11	4.63	5.00	1.00	6667.45	5076.53
ρ_2	4.73	4.73	0.21	0.21	4.37	5.07	1.00	2510.79	4464.76
ρ_3	−1.01	−1.01	0.17	0.17	−1.29	−0.74	1.00	723.93	1362.43

The numbers were computed using the R package `posterior` (Bürkner et al. 2022) and correspond to posterior means, medians, standard deviations, median absolute deviations, 5% and 95% quantiles, an improved version of the Gelman-Rubin $\widehat{R}$ as well as two versions of the effective sample size, one for the bulk of the distribution and one for the tails (see Vehtari et al. 2021, for details)

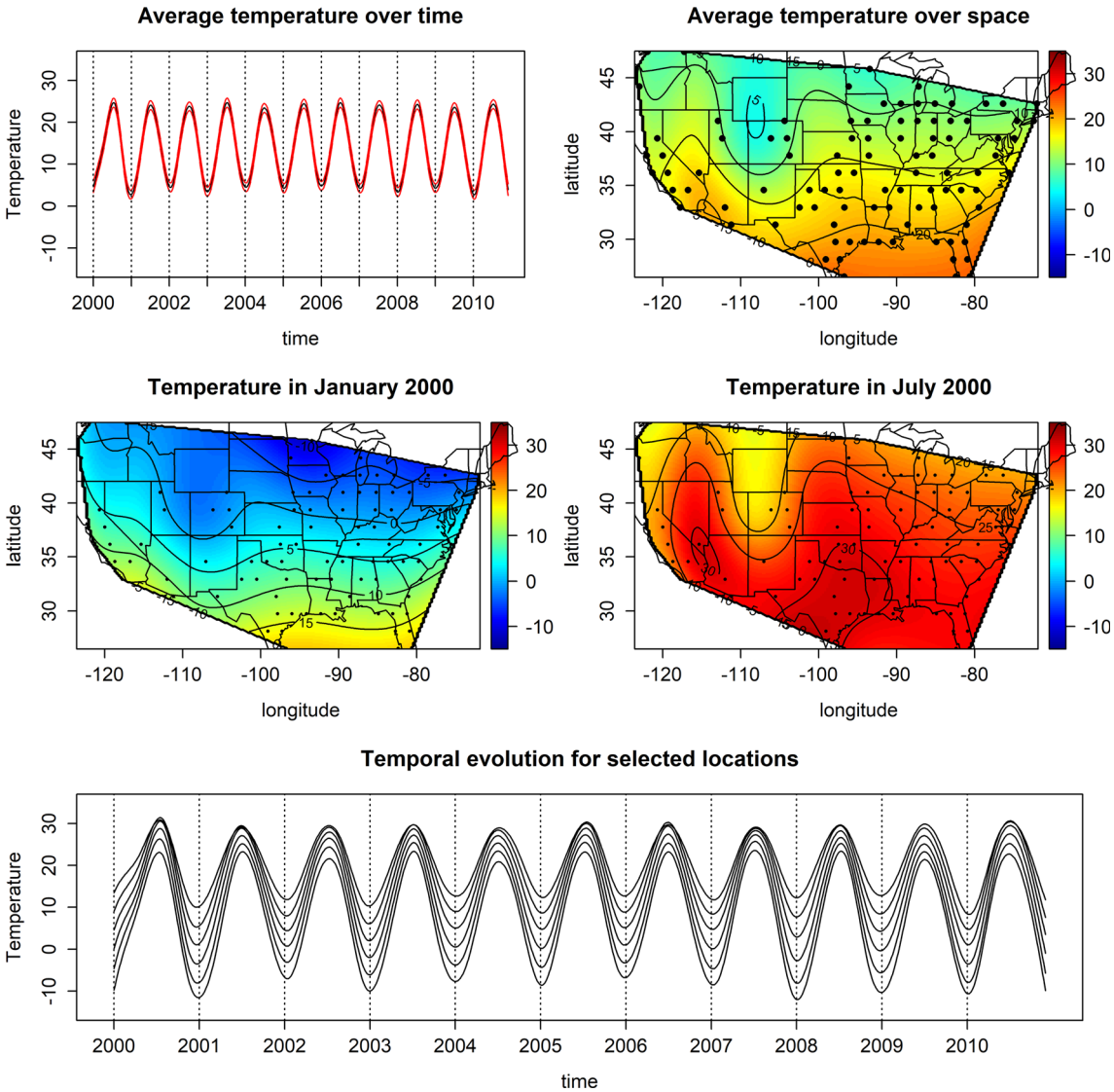

Fig. 4 The plots in the first row show the average temperature over time together with 95% posterior credible intervals (left) and the average temperature over space (right). The remaining plots are slice plots. In the middle row, we fix the time (January 2000, July 2000) and plot the temperature surface. In the final plot, we fix the location (longitude=-95, latitude=30,33,36,39,42,45) and plot the temporal evolution. The dashed vertical lines in the first and last plot indicate the month January for each of the years 2000 to 2010

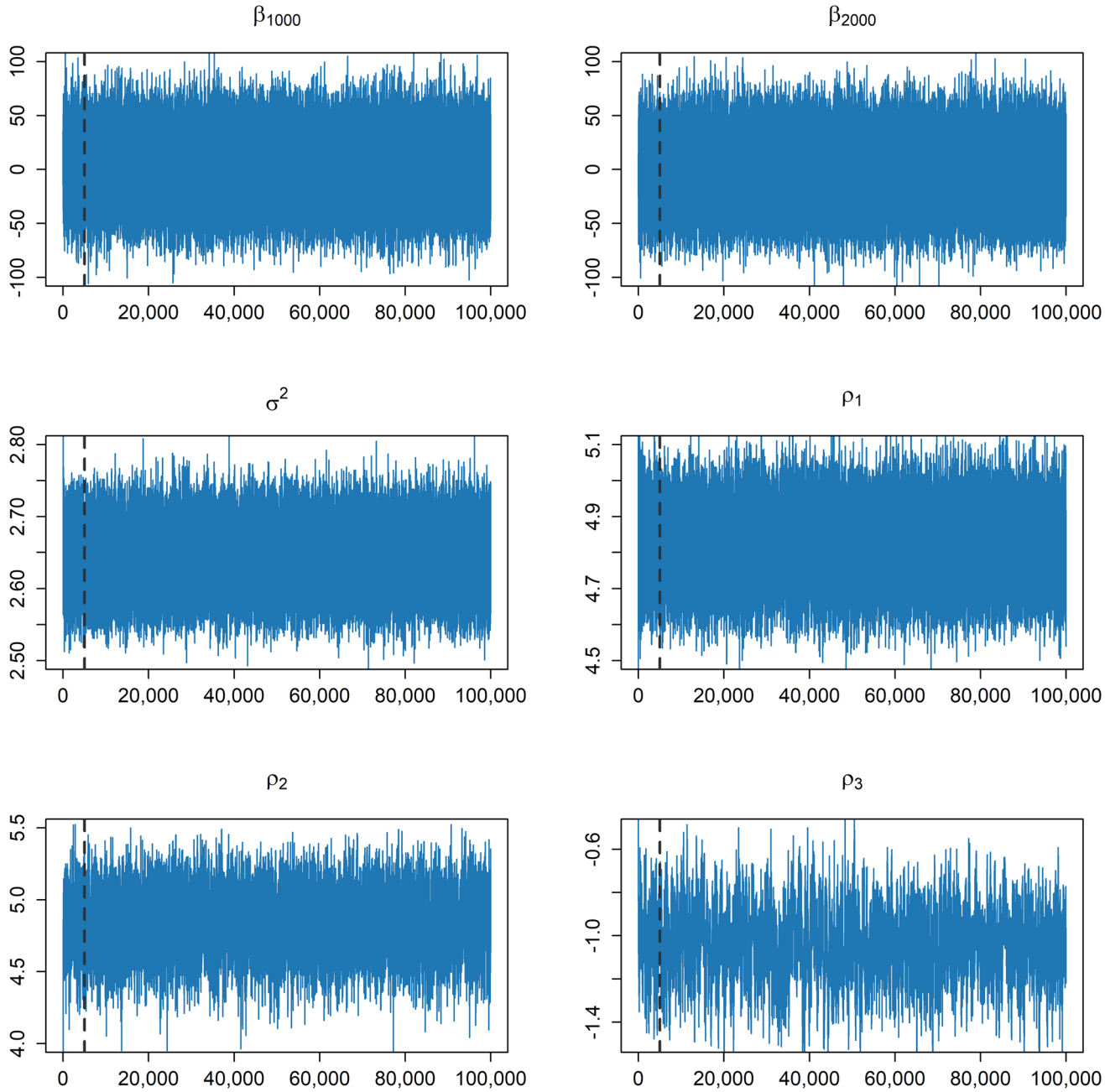

Fig. 5 Shown are trace plots for two tensor product B-spline coefficients ($\boldsymbol{\beta}_{1000}$ and $\boldsymbol{\beta}_{2000}$), the residual variance σ^2 and the log-smoothing variances ρ_1, ρ_2, ρ_3. We generated 100, 000 MCMC samples and discarded the first 5, 000 as burn-in, which is indicated by the dashed gray line on the left of each plot

has not even finished the warm-up phase of 200 iterations by then). Moreover, the MCMC mixing for the smoothing parameters is reasonably good. Vehtari et al. (2021) recommend that $\widehat{R}$ should be less than 1.01 and that the effective sample size should exceed 400, which are both satisfied (see Table 3). The MH acceptance rate for the vector $\boldsymbol{\rho} = (\rho_1, \rho_2, \rho_3)^\top$ was about 64%.

Overall summary. In summary, the temperature data example shows that our new approach is very well applicable to analyze real data. Through the visualization of low-dimensional slices and averages, the method allows us to gain interesting insights into complex multidimensional functions.

6 Discussion

In this paper, we have introduced a highly efficient fully Bayesian approach for anisotropic multidimensional smoothing. The cornerstone of our new approach are efficient derivative-based MH updates for the log-smoothing variances. These updates are possible because of the special representation (11) of the anisotropic roughness penalty matrix, which relies on the Kronecker sum structure of the penalty matrix. We have shown that our new approach outperforms previous suggestions in the literature and demonstrated its applicability through a real data example from spatio-temporal statistics.

Two possible directions for future research are as follows: First, while we have focused on the p-dimensional nonparametric regression model (1), it is straightforward to embed a p-dimensional tensor product smooth into a larger additive predictor. In this case, it is beneficial to introduce centering constraints for the tensor product smooth, one may for instance use empirical centering constraints of the form $\sum_{i=1}^{n} f(\boldsymbol{x}_i) = 0$ (cf. Lang et al. 2014). These centering constraints can easily be realized in the MCMC sampler through conditioning by Kriging (Rue and Held 2005, Section 2.3.3). Crucially, the update of the smoothing parameters is not affected by the centering constraints so that the approach introduced in Sect. 3.2.2 can easily be carried over. Second, while we have focused on a Gaussian response model our approach can easily be carried over to non-Gaussian response models. Bayesian P-splines have often been applied in additive non-Gaussian models using IWLS proposals for the B-spline coefficients (see, e.g., Klein et al. 2015). In the present setting, one can use (blockwise) IWLS proposals for the tensor product B-spline coefficients $\boldsymbol{\beta} \in \mathbb{R}^D$. Crucially, the update of the log-smoothing variances $\boldsymbol{\rho}$ does not depend on the likelihood so that the approach introduced in Sect. 3.2.2 can directly be carried over to a non-Gaussian response setting.

Following the suggestion of a reviewer, we close with a comparison of Bayesian tensor product P-splines and Gaussian processes. From a theoretical perspective, Bayesian tensor product P-splines are closely related to Gaussian processes. More specifically, one can show that the prior that is induced by (2) and (3) in function space can be represented as a convolution of an improper uniform prior and a particular Gaussian process. The kernel or covariance function of the latter is influenced by the dimensions of the marginal B-spline bases and it also incorporates the anisotropic roughness penalty (4). To convey the strengths and limitations of Bayesian tensor product P-splines from a practical perspective, we next compare them to the commonly used Gaussian process with squared exponential kernel. Anisotropic estimation for the Gaussian process with squared exponential kernel can be achieved through the introduction of multiple length scale parameters. However, this approach suffers from cubic computational complexity $\mathcal{O}(n^3)$ in the sample size n (see, e.g., Rasmussen and Williams 2006). Therefore, it quickly becomes untenable unless approximations such as the generalized Vecchia approximation are used (Katzfuss and Guinness 2021). Bayesian tensor product P-splines, in contrast, are almost independent of the sample size n and therefore well-suited for large samples. However, the number of basis functions D grows exponentially in the dimension of the smooth p (see Table 1). Therefore, Bayesian tensor product P-splines are best suited for two or three-dimensional smooths and they become computationally infeasible for dimension $p > 5$, which can be regarded as their main limitation. Overall, the properties of Bayesian tensor product P-splines can be summarized as follows:

- They allow for fully Bayesian estimation of anisotropic functions of a moderate number of covariates $p \in \{2, 3, 4, 5\}$.
- The computational cost is almost independent of the sample size n, which means that very large sample sizes pose no problem.
- Lower-dimensional structures of three or higher-dimensional smooths can easily be visualized by exploiting the tensor product structure.

In the light of these properties, we think that Bayesian tensor product P-splines provide a valuable addition to the statistician's toolbox for multivariate function estimation.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11222-025-10569-y.

Acknowledgements We thank the editor and two anonymous reviewers whose comments led to a substantial improvement of the initial submission.

Funding Open Access funding enabled and organized by Projekt DEAL. This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) through the Emmy Noether grant KL 3037/1-1.

References

Belitz, C., Brezger, A., Klein, N., Kneib, T., Lang, S., Umlauf, N.: BayesX—Software for Bayesian inference in structured additive regression models. http://www.bayesx.org. Version 3.0.2 (2015)

Bürkner, P.-C., Gabry, J., Kay, M., Vehtari, A.: posterior: Tools for Working with Posterior Distributions. R package version 1.3.1 (2022)

Carpenter, B., Gelman, A., Hoffman, M.D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., Riddell, A.: Stan: A probabilistic programming language. J. Stat. Softw. **76**(1), 1–32 (2017)

Eilers, P.H.C., Marx, B.D.: Multivariate calibration with temperature interaction using two-dimensional penalized signal regression. Chemom. Intell. Lab. Syst. **66**(2), 159–174 (2003)

Friedman, J.H.: Multivariate adaptive regression splines. Ann. Stat. **19**(1), 1–67 (1991)

Geweke, J., Tanizaki, H.: Note on the sampling distribution for the Metropolis-Hastings algorithm. Commun. Stat. Theory Methods **32**(4), 775–789 (2003)

Ghosh, J., Li, Y., Mitra, R.: On the use of Cauchy prior distributions for Bayesian logistic regression. Bayesian Anal. **13**(2), 359–383 (2018)

Goodrich, B., Gabry, J., Ali, I., Brilleman, S.: rstanarm: Bayesian applied regression modeling via Stan. R package version 2.21.3. (2022)

Gu, C.: Smoothing Spline ANOVA Models, 2nd edn. Springer, New York, NY (2013)

Harezlak, J., Ruppert, D., Wand, M.P.: Semiparametric Regression with R. Springer, New York, NY (2018)

Katzfuss, M., Guinness, J.: A general framework for Vecchia approximations of Gaussian processes. Stat. Sci. **36**(1), 124–141 (2021)

Klein, N., Kneib, T.: Scale-dependent priors for variance parameters in structured additive distributional regression. Bayesian Anal. **11**(4), 1071–1106 (2016)

Klein, N., Kneib, T., Lang, S.: Bayesian generalized additive models for location, scale, and shape for zero-inflated and overdispersed count data. J. Am. Stat. Assoc. **110**(509), 405–419 (2015)

Kneib, T., Klein, N., Lang, S., Umlauf, N.: Modular regression—a Lego system for building structured additive distributional regression models with tensor product interactions. TEST **28**(1), 1–39 (2019)

Köhler, M., Umlauf, N., Greven, S.: Nonlinear association structures in flexible Bayesian additive joint models. Stat. Med. **37**(30), 4771–4788 (2018)

Lang, S., Brezger, A.: Bayesian P-splines. J. Comput. Graph. Stat. **13**(1), 183–212 (2004)

Lang, S., Umlauf, N., Wechselberger, P., Harttgen, K., Kneib, T.: Multilevel structured additive regression. Stat. Comput. **24**(2), 223–238 (2014)

Lee, D.-J., Durbán, M.: P-spline ANOVA-type interaction models for spatio-temporal smoothing. Stat. Model. **11**(1), 49–69 (2011)

Neal, R.M.: Slice sampling. Ann. Stat. **31**(3), 705–767 (2003)

Nocedal, J., Wright, S.: Numerical Optimization, 2nd edn. Springer, New York, NY (2006)

Plummer, M.: JAGS: a program for analysis of Bayesian graphical models using Gibbs sampling. In: Hornik, K., Leisch, F., Zeileis, A. (eds.) Proceedings of the 3rd International Workshop on Distributed Statistical Computing. Technische Universität Wien, Vienna (2003)

Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge, MA (2006)

Rodríguez-Álvarez, M.X., Lee, D.-J., Kneib, T., Durbán, M., Eilers, P.: Fast smoothing parameter separation in multidimensional generalized P-splines: the SAP algorithm. Stat. Comput. **25**(5), 941–957 (2015)

Rue, H., Held, L.: Gaussian Markov Random Fields: Theory and Applications. Chapman & Hall/CRC Press, Boca Raton, FL (2005)

Umlauf, N., Klein, N., Zeileis, A.: bamlss: Bayesian additive models for location, scale, and shape (and beyond). J. Comput. Graph. Stat. **27**(3), 612–627 (2018)

Vehtari, A., Gelman, A., Simpson, D., Carpenter, B., Bürkner, P.-C.: Rank-normalization, folding, and localization: an improved R for assessing convergence of MCMC (with discussion). Bayesian Anal. **16**(2), 667–718 (2021)

Wood, S.N.: Low-rank scale-invariant tensor product smooths for generalized additive mixed models. Biometrics **62**(4), 1025–1036 (2006)

Wood, S.N.: Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **73**(1), 3–36 (2011)

Wood, S.N.: mgcv: Mixed GAM Computation Vehicle with GCV/AIC/REML smoothness estimation (2012)

Wood, S.N.: Just another Gibbs additive modeler: interfacing JAGS and mgcv. J. Stat. Softw. **75**(7), 1–15 (2016)

Wood, S.N., Fasiolo, M.: A generalized Fellner-Schall method for smoothing parameter optimization with application to Tweedie location, scale and shape models. Biometrics **73**(4), 1071–1081 (2017)

Wood, S.N., Scheipl, F., Faraway, J.J.: Straightforward intermediate rank tensor product smoothing in mixed models. Stat. Comput. **23**(3), 341–360 (2013)