Clara Maria Happ

Statistical Methods for Data with Different Dimensions

Multivariate Functional PCA and Scalar-on-Image Regression

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

Eingereicht am 12.07.2017

Erste Berichterstatterin: Prof. Dr. Sonja Greven
Zweiter Berichterstatter: Prof. Dr. Volker Schmid
Dritter Berichterstatter: Prof. Dr. Jan Gertheiss

Tag der Disputation: 22.09.2017

If you are receptive and humble, mathematics will lead you by the hand. Again and again, when I have been at a loss how to proceed, I have just had to wait until I have felt the mathematics led me by the hand. It has led me along an unexpected path, a path where new vistas open up, a path leading to new territory, where one can set up a base of operations, from which one can survey the surroundings and plan future progress.

Paul Dirac

Acknowledgements

This thesis would not have been possible without the help and support of many people:

First of all, I would like to thank Prof. Dr. Sonja Greven and Prof. Dr. Volker Schmid for giving me this unique opportunity and for jointly supervising this thesis. Thanks for sharing your experience, knowledge and enthusiasm for functional data and (Bayesian) image analysis. Thank you, Sonja, for proofreading all my drafts so quickly and thoroughly and thank you, Volker, for the great collaboration in both teaching and research.

I would also like to thank Prof. Dr. Jan Gertheiss, who kindly agreed to serve as the third reviewer and to Prof. Dr. Christian Heumann and Prof. Dr. Anne-Laure Boulesteix for being part of the doctoral committee.

In particular, I would like to thank Prof. Dr. Francesca Biagini for being my mentor in the LMUMentoring program. Thank you also for the financial support that gave me the chance to present my work at several national and international conferences.

My special thanks go to our collaborators, Prof. Dr. Michael Ewers and Dr. Miguel Àngel Araque Caballero for the fruitful collaboration and interesting insights into the early development of Alzheimer's disease, and to all my colleagues, particularly those from the Biostatistics/FDA and Financial Econometrics groups.

Thank you also to all people who helped me to improve this thesis by carefully proofreading parts of it.

When you try to reach for the stars, you have to stand on a solid ground. I would like to thank all professors who taught me much more than just what I needed to get good grades. In particular, I would like to thank Prof. Dr. W. Balser and Prof. Dr. P. Müller for showing me the beauty of mathematics and Prof. Dr. F. Schweiggert and Prof. Dr. U. Schöning for teaching me the art of computer science. Both were extremely helpful for this thesis.

My particular thanks go to Eva, Serene and Cristina, to Anna, Almut and many others from Theresianum and to Caro and Sven. Thank you for reminding me of what really matters in life, for *Radler*, tea and hot chocolate, for sharing everyday life and unforgettable moments.

Finally, I would like to thank my family and Malte for their infinite-dimensional support for this girl *chi è nata con la testa così*. My special thanks go to Malte and Lucas for pushing me forward when I needed it. Without you, the most important parts of this thesis would never have been written. Thank you.

Summary

This thesis addresses the joint analysis of data with different dimensions, such as scalars, vectors, functions and images. This is of high practical and methodological relevance, as in the course of the technical progress, data with increasing complexity and dimensionality becomes available, requiring the extension of statistical models to new types of data and leading to the development of completely new statistical methods.

In the first part of the thesis, multivariate functional principal component analysis (MFPCA) is developed for functional data on different dimensional domains. This is a novel method, as existing approaches for MFPCA are restricted to multivariate functional data on the same, one-dimensional interval. Using the new approach, principal components for data consisting e.g. of functions and images (i.e. functions on a two-dimensional domain) can be obtained, taking potential covariation in the elements into account. The thesis constructs a thorough theoretical basis for multivariate functional data on different dimensional domains and derives a theoretical relationship between univariate and multivariate functional principal component analysis for finite sample sizes. The results can be used to estimate multivariate functional principal components, eigenvalues and scores based on their univariate counterparts. It is shown how the method can be extended to univariate elements in general basis representations and to a weighted version of MFPCA to correct for differences in domain, range or variation of the elements. The approach is also applicable for sparse data or data with measurement error. The finite sample performance of the new method is evaluated in a simulation study with different levels of complexity. Moreover, asymptotic properties for large sample sizes are derived in two theorems, using results from perturbation theory and showing consistency of the proposed estimators. The estimation algorithm has been implemented in a publicly available **R**-package **MFPCA**, together with another **R**-package **funData** for representing functional data in an object-oriented manner. The thesis provides an introduction to the software and the underlying concepts. The new approach is illustrated in an application to a neuroimaging dataset. The aim here is to examine the relationship between trajectories of a neuropsychological test score over time and FDG-PET brain scans at baseline, that can be interpreted as functions on a three-dimensional domain, as the latter might be predictive of subsequent cognitive decline. The results show that estimates obtained from the new MFPCA method are meaningful from a medical point of view and provide new insights into the data.

The second part of the thesis is concerned with scalar-on-image regression. This class of statistical methods models the relation of a scalar outcome and an image predictor, hence data with different dimensions and a complex dependence structure. It is representative for a broad class of statistical models for complex data, which intrinsically is unidentifiable, as in general the number of observations will be low compared to the number of pixels in the image. Strong model assumptions are thus required to obtain a unique solution,

which is of course conditional on the hypotheses made on the true coefficient image. In the thesis, different models for scalar-on-image regression with different assumptions are compared with respect to their ability to give reliable and interpretable estimates. To this end, new measures for quantifying the influence of model assumptions are developed and analyzed in a simulation study for nine different scalar-on-image models. The relevance of the topic is illustrated in a practical neuroimaging application. It is shown that different models with different assumptions can lead to results that share common patterns, but can differ substantially in their details, as model assumptions can have a strong influence on the estimates. This can entail the risk of over-interpreting effects that are mainly driven by the model assumptions.

Zusammenfassung

Diese Doktorarbeit beschäftigt sich mit der gemeinsamen Analyse von Daten unterschiedlicher Dimension, wie beispielsweise Skalare, Vektoren, Funktionen und Bilder. Dies ist sowohl aus praktischer als auch aus methodischer Sicht relevant, da im Zuge des technischen Fortschritts Daten mit zunehmender Komplexität und Dimensionalität zur Verfügung stehen, die einerseits eine Erweiterung von statistischen Modellen auf neue Datentypen erfordern und andererseits zur Entwicklung völlig neuer statistischer Methoden führen.

Im ersten Teil der Arbeit wird eine multivariate funktionale Hauptkomponentenanalyse (engl. multivariate functional principal component analysis, MFPCA) für funktionale Daten auf unterschiedlich-dimensionalen Trägern entwickelt. Es handelt sich hier um eine neuartige Methode, da bestehende Ansätze für MFPCA auf multivariate funktionale Daten auf einem gemeinsamen eindimensionalen Intervall beschränkt sind. Mit dem neu entwickelten Ansatz können Hauptkomponenten für Daten bestimmt werden, die z.B. aus Funktionen und Bildern (d.h. Funktionen auf einem zwei-dimensionalen Träger) bestehen, womit eventuell vorhandene Kovariation in den Elementen berücksichtigt werden kann. In der Arbeit werden die theoretischen Grundlagen für multivariate funktionale Daten auf unterschiedlich-dimensionalen Trägern gelegt. Für den Fall einer endlichen Stichprobe wird anschließend einen theoretischen Zusammenhang zwischen univariater und multivariater funktionaler Hauptkomponentenanalyse hergeleitet. Das Ergebnis kann zur Schätzung multivariater funktionaler Hauptkomponenten, Eigenwerte und Scores auf Basis der univariaten Analoga genutzt werden. Es wird gezeigt, wie die Methode auf univariate Elemente in allgemeinen Basisdarstellungen erweitert werden kann. Weiterhin wird eine gewichtete Version der MFPCA vorgestellt, mithilfe derer für Unterschiede im Träger, Wertebereich oder Variation der einzelnen Elemente korrigiert werden kann. Der neue Ansatz eignet sich auch für funktionale Daten mit wenig Beobachtungspunkten (engl. sparse data) oder Daten, die mit Messfehlern erhoben wurden. Für den Fall endlicher Stichproben wird die Leistungsfähigkeit der neuen Methode im Rahmen einer Simulationsstudie mit unterschiedlichen Komplexitätsgraden untersucht. Darüberhinaus werden die asymptotischen Eigenschaften für große Stichproben in zwei Theoremen unter Verwendung von Resultaten aus der Perturbationstheorie hergeleitet und es wird bewiesen, dass die vorgeschlagenen Schätzer konsistent sind. Der Schätzalgorithmus ist in dem öffentlich verfügbaren **R**-Paket **MFPCA** implementiert, gemeinsam mit einem weiteren **R**-Paket **funData** zur objektorientierten Darstellung funktionaler Daten. Die Arbeit enthält eine Einführung in die Software und die zugrundeliegenden Konzepte. Die neue Methode wird in einem Anwendungskapitel anhand eines Neuroimaging Datensatzes illustriert. Ziel der Untersuchung ist es, einen Zusammenhang zwischen den Ergebnissen eines neuropsychologischen Tests über den Studienverlauf und FDG-PET Gehirnscans herzustellen, die zu Beginn der Studie aufgenommen wurden, da Letztere prädiktiv für eine anschließende Verschlechterung der kognitiven Fähigkeiten

sein können. Die Scans können dabei als Funktionen auf einem drei-dimensionalen Träger aufgefasst werden. Die Ergebnisse zeigen, dass die von der neuen MFPCA Methode gefundenen Schätzer medizinisch sinnvoll sind und neue Einblicke in die Daten ermöglichen.

Der zweite Teil der Arbeit beschäftigt sich mit Skalar-auf-Bild Regression. Diese statistische Modellklasse beschreibt den Zusammenhang einer skalaren Zielgröße und einer Einflussgröße in Form eines Bildes, also Daten mit unterschiedlicher Dimension und einer komplexen Abhängigkeitsstruktur. Sie steht stellvertretend für eine breite Klasse statistischer Modelle für komplexe Daten, die von sich aus nicht identifizierbar ist, da im Allgemeinen die Anzahl der Beobachtungen im Verhältnis zur Anzahl der Pixel in einem Bild sehr klein ist. Es sind also starke Modellannahmen vonnöten, um eine eindeutige Lösung zu erhalten, die selbstverständlich durch die Annahmen an das wahre Koeffizientenbild bedingt wird. In dieser Arbeit werden unterschiedliche Modelle für Skalar-auf-Bild Regression mit unterschiedlichen Annahmen in Bezug auf ihre Fähigkeit, zuverlässige und interpretierbare Ergebnise zu erzielen, untersucht. Zu diesem Zweck werden neue Maße zur Quantifizierung des Einflusses von Modellannahmen entwickelt und in einer Simulationsstudie für neun verschiedene Skalar-auf-Bild Regressionsmodelle untersucht. Die Bedeutung der Thematik wird wiederum in einer praktischen Anwendung aus dem Neuroimaging-Bereich veranschaulicht. Es wird gezeigt, dass unterschiedliche Modelle mit unterschiedlichen Annahmen zu Ergebnissen führen können, die zwar ähnliche Muster aufweisen, sich in Details aber zum Teil deutlich unterscheiden, da die Modellannahmen einen starken Einfluss auf die Schätzungen haben können. Dies bringt die mögliche Gefahr mit sich, Effekte zu überinterpretieren, die hauptsächlich von den Modellannahmen getrieben sind.

Contents

1. Introduction

In recent years there has been a veritable boom concerning the collection and storage of enormous amounts of data. Nowadays, data is recorded, collected and analyzed automatically in nearly all areas of life, resulting in what has been called *big data*. In general, the data is multivariate, i.e. one has many, potentially interdependent observations for each observation unit. The available data has increased not only in quantity, but also in complexity, such that for each unit one may have a collection of values, among which data with a strong inherent structure. Examples for structured data include measurements that have been collected in regular or irregular time intervals (longitudinal / functional data), or images, which have a clear spatial structure. In the statistical analysis of this kind of data, these different sources of dependence and complexity have to be taken into account. This can be achieved by adapting existing models to the new data situation or by developing new and specifically tailored statistical models. The present thesis contributes two aspects to the statistical analysis of data with different dimensions.

The first part of the thesis introduces multivariate functional principal component analysis for data observed on potentially different dimensional domains. This new approach allows to find patterns of joint variation in multivariate functional data, i.e. data with different elements of potentially different dimensions. Each of these elements can be seen as a discrete realization of a (smooth) function. Examples for such data are e.g. combinations of functions and images or even functions and three-dimensional images. The innovation in the proposed method concerns the fact that all existing approaches for multivariate functional principal component analysis are available only for one-dimensional data and the different elements are restricted to have the same, one-dimensional domain. With the new approach, by contrast, functions on different domains can be taken into account. The thesis provides the theoretical foundations for multivariate functional data on different dimensional domains and defines a principal component analysis for this kind of data based on a multivariate version of the Karhunen-Loève Theorem. The resulting principal components are seen to have the same structure as the data. This means that if the data consists e.g. of functions and images, the principal component will also do. The multivariate functional principal component analysis thus offers a natural representation of this very complex type of data and yields individual scalar scores, which characterize the weight of each principal component for each observation. The proposed estimation algorithm for the multivariate

functional principal component analysis is based on univariate functional principal component analysis or general basis expansions. Therefore, it can be customized to the data at hand. The asymptotic properties of the resulting estimators are investigated, showing consistency under a given set of assumptions. Moreover, an efficient implementation of the method is presented which builds upon an object-oriented representation of (multivariate) functional data. The relevance of the new multivariate functional principal component approach is demonstrated in a neuroimaging application. Here it allows to find joint patterns of variation in bivariate functional data, consisting of functions over time combined with two- or three-dimensional images. Within the application it is shown how multivariate functional principal component analysis can be used to calculate predictions for new data which have not been included in the multivariate functional principal component analysis.

For the statistical analysis of complex, high-dimensional data one frequently has to make strong assumptions in order to make a model identifiable at all. This is the starting point for the second part of this thesis. The issue is discussed by means of scalar-on-image regression. This statistical model aims at combining scalar response values with images plus potentially additional scalar covariates, hence high-dimensional data with different dimensions and complexity. First of all, an overview of the most important methods for scalar-on-image regression is given, with a special focus on their assumptions. In a subsequent simulation study and application to real data, the extent of the problem is evaluated and quantified by means of newly introduced measures. The results show that model assumptions can indeed have a considerable impact on the estimates, which manifests in quite different estimated coefficient images for the same data. This of course raises the question how to interpret them. In the thesis, it is discussed how problematic settings can be identified in a specific application and how one can characterize features that can be estimated at all using the chosen methods and the data at hand.

In this way, this thesis contributes to the further development of statistical methodology and software. At the same time it is shown that statistical methods for high-dimensional data require particular caution concerning their application and interpretation, as they frequently involve strong model assumptions.

Outline and Contributing Manuscripts

Chapter 2 discusses concepts of dimensionality for structured, high-dimensional data. It gives an introduction to functional data analysis and Bayesian statistics, which form the methodological basics of the methods developed in this thesis. Further, the chapter gives an overview of the existing literature and open research questions, from which the new results have emerged.

The following chapters are reprints of the contributing manuscripts of this thesis. The beginning of each chapter contains a full reference to the original article and software, copyright information (where applicable), a declaration of the individual contributions of each author, acknowledgements and the abstract. The (online) appendices of each article are given in the appendices A to D.

Chapter 3 introduces multivariate functional principal component analysis for data observed on different dimensional domains. The properties of the estimators are investigated theoretically as well as numerically for real and simulated data.

Chapter 4 contains a more advanced application of the multivariate functional principal component analysis to bivariate data consisting of trajectories of a neuropsychological test over time and three-dimensional brain scans at baseline. Further, it is shown how predictions for new data can be obtained on the basis of multivariate functional principal component analysis.

Chapter 5 describes the object-oriented representation of functional data in the **R**-package **funData**, including many usage examples. In addition, the implementation of the estimation algorithm for multivariate functional principal component analysis in the **R**-package **MFPCA** is described, which is based on the **funData** package.

Chapter 6 discusses the impact of model assumptions in scalar-on-image regression. It provides an overview of different approaches for this model class with a special focus on the assumptions made in each model. The impact of the assumptions is studied for real and synthetic data, showing that the respective assumptions can indeed have a strong influence on the results.

The thesis concludes with a short summary and an outlook to potential aspects of future research in Chapter 7.

Software

If not stated otherwise, all calculations have been carried out in **R** (R Core Team, 2015).

2. Statistical Concepts for Structured High-Dimensional Data

This chapter introduces the methodological basis for the articles that contribute to this thesis. It gives an overview of the existing literature and explains the scientific context and open research questions from which the contributing papers have emerged. Section 2.1 is a general discussion of concepts of dimensionality for structured, high-dimensional data. Section 2.2 introduces functional data analysis, which is used in both parts of the thesis. Scalar-on-image regression is in the focus of part II of the thesis. Here Bayesian methods have proven useful to deal with the complex structure of image data. The underlying concepts are presented in Section 2.3.

2.1. Concepts of Dimensionality

In standard statistical settings, one often assumes to observe data from $N \in \mathbb{N}$ statistical units, that may for instance be a group of people in a medical study (such as in Chapters 3, 4 and 6), weather stations (as in the examples in Chapter 5) or any other clearly defined entity. For each unit, one usually investigates the same $p \in \mathbb{N}$ variables. The relevant dimensions of the data are thus given by N and p, which determine important model properties. As an example, (generalized) linear models are identifiable only if the number of explaining variables ($p-1$, if one of the p observed variables is the response) is lower or equal to the number of observations N (Nelder and Wedderburn, 1972).

In recent years, methods for $N \ll p$ settings, where the number of variables clearly exceeds the number of observations, have grown in importance, as the technical progress in many fields of applications allows to collect more and more data without much effort. In this case of high-dimensional data, the p variables often have an inherent structure, being caused e.g. by repeated measurements over time (longitudinal data; note that here the number of variables per observation unit may vary) or space (spatial data). This structure can help to overcome the "curse of dimensionality" and turn it into a "blessing" (Wang, Chiou, et al., 2016). At the same time, it adds a new form of dimensionality to the data, describing the time interval or the spatial dimension of the data. Images for example will typically

form regular, two-dimensional lattices. In the context of neuroimaging, however, they can become three- (cf. the FDG-PET scans in Chapter 4) or even four-dimensional (fMRI data, e.g. Brezger et al., 2007; Smith and Fahrmeir, 2007).

When the temporal or spatial resolution of the data becomes finer, meaning that one has many observations within a short time period or a small spatial distance, the observed data can be considered as discrete measurements of an underlying stochastic process $X\colon \mathcal{T} \to \mathbb{R}$ on a domain $\mathcal{T} \subset \mathbb{R}^d$ with $d \in \mathbb{N}$, which is continuous in time ($d = 1$) or space ($d = 2, 3, 4, \ldots$). This is the key to functional data analysis, which is introduced in Section 2.2. The process can be considered as infinite-dimensional, as one could – at least theoretically – increase the resolution of the observation points to become arbitrarily fine, which would lead to arbitrarily many observed values per observation unit. At the same time, the dimensionality d of the domain $\mathcal{T}$ determines whether the realizations of the process are functions ($d = 1$), images ($d = 2$) or even higher-dimensional.

Finally, one can combine multiple stochastic processes with potentially different dimensional domains to form a multivariate stochastic process. This is the data structure which is in the focus of Chapters 3 to 5. Here, the number of elements can be seen as an additional kind of dimensionality.

In summary, structured, high-dimensional data can have different forms of dimensionality. The statistical methods discussed in this thesis aim at combining information from data with different dimensions, which becomes increasingly available. In the case of multivariate functional principal component analysis (part I), the objective is to find the important modes of joint variation in multivariate functional data, having elements with potentially different dimensional domains, such as functions and images. The second part of the thesis focuses on scalar-on-image regression. This statistical model class aims at finding a relationship between a scalar response and an image covariate, possibly supplemented by some additional scalar covariates. The data observed for one statistical unit hence consists of a scalar (a single variable), an image, which is high-dimensional with a regular spatial structure, and a vector of scalar covariates. In order to obtain unique and interpretable coefficients, that combine the scalar and image covariates with the response, strong model assumptions are required, whose impact is discussed in Chapter 6.

2.2. Functional Data Analysis

Functional data analysis (Ramsay, 1982) is concerned with the analysis of data that can be interpreted as discrete samples of functions. In this section, we define univariate and multivariate functional data and give an overview of important techniques for functional data that are used in the following chapters. A general overview of functional data analysis

can be found e.g. in Ramsay and Silverman (2005), Ferraty and Vieu (2006), Horváth and Kokoszka (2012), and Hsing and Eubank (2015) or in a more recent review of Wang, Chiou, et al. (2016).

2.2.1. Functional Data

As mentioned in Section 2.1, the fundamental idea of functional data analysis is to interpret data, which has a natural ordering (e.g. in time or space), as discrete measurements of an underlying continuous stochastic process $X\colon \mathcal{T} \to \mathbb{R}$. Realizations $x_1, \ldots, x_N$ of the process will thus be functions over their domain $\mathcal{T}$. Important tools for describing functional data are the pointwise mean and covariance functions

$$\mu(t) = \mathbb{E}(X(t)), \quad c(s,t) = \mathrm{Cov}(X(s), X(t)), \quad s,t \in \mathcal{T}. \tag{2.1}$$

In most cases, the process X is assumed to have rather smooth realizations and to lie in the Hilbert space $L^2(\mathcal{T})$ of square integrable functions over $\mathcal{T}$ (for a definition, see e.g. Reed and Simon, 1980, Chapter III). The Hilbert space structure is associated with the existence of a scalar product, which is mostly taken as the standard inner product in $L^2(\mathcal{T})$,

$$\langle f, g \rangle_2 = \int_{\mathcal{T}} f(t)g(t)\mathrm{d}t, \qquad f, g \in L^2(\mathcal{T}), \tag{2.2}$$

and provides an induced norm $\|f\|_2 = \langle f, f \rangle_2^{1/2}$, $f \in L^2(\mathcal{T})$. This allows to transfer many concepts of multivariate statistics involving projections, angles or distances to the functional case. Important examples include principal component analysis (see Section 2.2.3) or regression models (see Section 2.2.4). The – at least theoretically – infinite dimensionality of the data, however, poses additional methodological challenges, as one has e.g. infinitely many principal components and the unknown coefficient in regression models becomes infinite-dimensional, too. One way to overcome these issues is to expand the data in a finite number of basis functions, which is discussed in the next section.

The first part of the thesis is concerned with *multivariate functional data*. As indicated before, this means that the data generating process X in this case is a vector of p stochastic processes $X^{(j)} : \mathcal{T}_j \to \mathbb{R}^{d_j}$, $d_j \in \mathbb{N}$, $j = 1, \ldots, p$, which we call the *elements* of X. They are again assumed to have realizations $x_i^{(j)} \in L^2(\mathcal{T}_j)$, $i = 1, \ldots, N$. Up to now, methods for multivariate functional data have mainly focused on processes on the same, one-dimensional domain $\mathcal{T}_1 = \ldots = \mathcal{T}_p \subset \mathbb{R}$ (e.g. Ramsay and Silverman, 2005; Berrendero et al., 2011; Jacques and Preda, 2014; Chiou, Yang, et al., 2014; Chiou and Müller, 2014). Chapter 3 formally introduces multivariate functional data on different dimensional domains and defines multivariate analogues of the scalar product, the mean and covariance functions. Subsequently, their theoretical properties are investigated, which form the basis of multivariate functional principal component analysis.

More generally, univariate and multivariate functional data can be seen as special cases of so-called *object data*, which is studied in object-oriented data analysis (Wang and Marron, 2007). This is the motivation for the object-oriented software implementation of functional data in the package **funData** (Happ, 2017a), which relates the statistical concept of object orientation with the object-oriented programming paradigm in computer science (Booch et al., 2007; Meyer, 1988). More details are given in Chapter 5.

2.2.2. Basis Function Approaches

Basis function approaches are widely used in functional data analysis and related fields, such as semiparametric regression. The central idea here is to approximate a realization $x \in L^2(\mathcal{T})$ of a stochastic process (when working with functional data) or an unknown coefficient function $\beta \in L^2(\mathcal{T})$ (in the context of regression) in terms of a finite number of given basis functions $B_1, \dots, B_K \in L^2(\mathcal{T})$. The term *basis functions* is somewhat misleading in this context, as a finite number of functions cannot entirely span the infinite-dimensional space $L^2(\mathcal{T})$. The functions $B_1, \dots, B_K$ in general are linearly independent and thus are elements of an infinite basis $\{B_k \colon k \in \mathbb{N}\}$ of $L^2(\mathcal{T})$. Moreover, they may be orthogonal or orthonormal. The approximation of x and β is given by their projection on the span of the basis functions, i.e.

$$x \approx \sum_{k=1}^{K} \theta_k B_k, \quad \beta \approx \sum_{k=1}^{K} b_k B_k. \tag{2.3}$$

This means of course an immense dimension reduction, as functions in the infinite-dimensional space $L^2(\mathcal{T})$ are now represented by elements of the K-dimensional subspace $\text{span}\{B_1, \dots, B_K\}$ of $L^2(\mathcal{T})$. Given observed data $x(t_s),\ s = 1, \dots, S$ with $\{t_1, \dots, t_S\} \subset \mathcal{T}$, the approximation can be interpreted as

$$x(t_s) = \sum_{k=1}^{K} \theta_k B_k(t_s) + \varepsilon_s, \quad \varepsilon_s \overset{\text{iid}}{\sim} \text{N}(0, \sigma^2),$$

which is a standard linear model in the coefficients θ_k and can be solved via least squares

$$(\tilde{x} - B\theta)^\top (\tilde{x} - B\theta) \to \min_{\theta} \tag{2.4}$$

with $\tilde{x} = (x(t_1), \dots, x(t_S))$[§], $B \in \mathbb{R}^{S \times K}$ with entries $b_{sk} = B_k(t_s)$ and $\theta = (\theta_1, \dots, \theta_K)$. A unique solution of (2.4) can be found if and only if $K \leq S$, i.e. if the number of basis functions does not exceed the number of observation points. The coefficients θ do not have a clear interpretation in the regression model, this is why basis function approaches can be considered as being part of the broad class of semiparametric methods (Ruppert et al., 2003).

[§]If not stated otherwise, $x = (x_1, \dots, x_p)$ denotes a p-dimensional column vector $x \in \mathbb{R}^p$.

Assuming that the approximation error is negligible, the functions in (2.3) can be represented by the K-dimensional coefficient vector θ or $b = (b_1, \ldots, b_K)$. This opens the broad range of statistical methods for multivariate data to the functional case, by simply applying them to the coefficient vectors and transforming the result back to the original space. Examples for this generic approach include e.g. principal component analysis (Ramsay and Silverman, 2005, Chapter 8), clustering (e.g. Abraham et al., 2003; James and Sugar, 2003; Jacques and Preda, 2013) or regression models (Marx and Eilers, 1999; Cardot et al., 2003; Müller and Stadtmüller, 2005). The goodness of the results certainly depends on the accuracy of the approximation, which is influenced by the type and the number of basis functions. In Chapter 6 of this thesis it is investigated how the assumption of a basis representation can influence the results in the special case of scalar-on-image regression.

In the following, P-spline bases and wavelets are presented in more detail as two typical examples for basis functions used in the context of functional data analysis (cf. Chapters 3 and 6). While both of them provide basis functions with local support, the non-orthogonal P-splines are mostly used to represent smooth functions (Eilers and Marx, 1996), whereas wavelets are suitable for modeling sharp, highly localized features and make use of orthonormality (Daubechies, 1988). Alternative choices of basis functions include all other forms of splines (e.g. thin plate regression splines Wood, 2003), Fourier bases and discrete cosine bases as their real counterparts (Ramsay and Silverman, 2005; Frigo and Johnson, 2005) and many more. The principal components found by functional principal component analysis (see Section 2.2.3) represent a special case of basis functions that depend on the data. Finally, random field methods, as discussed in Section 2.3, can be seen as basis function approaches, too, with the basis being formed by indicator functions for each region.

P-Splines

Spline basis functions are very popular in semiparametric statistics and functional data analysis. They are defined as piecewise polynomial functions, smoothly connected at so-called *knots* (for a precise definition see e.g. Quarteroni et al., 2007, Chapter 7). *B-Splines* (introduced in Schoenberg, 1946a; Schoenberg, 1946b) are commonly used due to their mathematical and computational advantages: On the one hand, they have attractive mathematical properties, as they have local support and yield continuous and differentiable functions, as long as the polynomial degree is sufficiently high. On the other hand, there exist fast and efficient algorithms for evaluating the B-spline basis functions at observation points $\{t_1, \ldots, t_S\} \subset \mathcal{T}$ (De Boor, 1972).

The popularity of B-splines in the statistical community has grown substantially with the seminal paper of Eilers and Marx (1996). They proposed a penalized approach (*P-splines*) to circumvent the non-trivial problem of choosing appropriate knots by using many basis functions and at the same time penalizing squared differences of the coefficients in order to

obtain a smooth function. This is achieved by minimizing the penalized version of the least squares criterion (2.4)

$$(\tilde{x} - B\theta)^\top (\tilde{x} - B\theta) + \lambda \theta^\top P \theta \to \min_\theta \tag{2.5}$$

with P a *penalty matrix*, e.g. for squared first differences of neighbouring coefficients,

$$\theta^\top P \theta = \sum_{k=2}^{K} (\theta_k - \theta_{k-1})^2. \tag{2.6}$$

The *smoothing parameter* $\lambda > 0$ controls the influence of the penalty term $\theta^\top P \theta$ on the solution of (2.5). If $\lambda = 0$, the penalty term drops out and the penalized least squares criterion equals the unpenalized version in (2.4). For $\lambda \to \infty$, the penalty term dominates the criterion and in the limiting case only parameter vectors θ in the null space of P are considered, as they are not penalized. In the case of (2.6), these are all constant vectors θ with $\theta_k = c$ for all $k = 1, \ldots, K$ and a constant $c \in \mathbb{R}$. An optimal value for λ can be found e.g. by (generalized) cross-validation, AIC or using a restricted maximum likelihood (REML) approach (Wood, 2006, Chapters 4 and 6).

For functions on higher-dimensional domains, Marx and Eilers (2005) suggest to use tensor products of univariate B-splines as basis functions. This is relevant in the case of images, which can be interpreted as discretely observed functions on a two- or higher dimensional domain, as it is repeatedly the case in this thesis. A function $x\colon \mathcal{T} \to \mathbb{R}$ with $\mathcal{T} \subset \mathbb{R}^2$ evaluated at $t = (t_x, t_y)$ can thus be approximated by

$$x(t) \approx \sum_{k_x=1}^{K_x} \sum_{k_y=1}^{K_y} \theta_{k_x,k_y} B_{k_x}(t_x) B_{k_y}(t_y).$$

The unknown coefficients $\theta = \{\theta_{k_x,k_y}\colon k_x = 1, \ldots, K_x, k_y = 1, \ldots, K_y\}$ can again be found by a penalized least squares criterion with a penalty that penalizes differences between neighbouring coefficients both in x- and y- direction. For more details, see the discussion in Chapter 6 and the paper of Marx and Eilers (2005).

Wavelets

Wavelets provide an alternative choice of fixed basis functions. While having some similarities with Fourier basis functions, they can be chosen to have a local support, which makes it easier to model functions with abrupt changes (e.g. edges in images). They have become very popular e.g. in imaging and signal processing since the fundamental papers of Daubechies (1988) and Mallat (1989). Wavelets are constructed as orthonormal basis functions from a multiresolution analysis, which is a family of closed subspaces $\{V_m \subset L^2(\mathbb{R})\colon m \in \mathbb{Z}\}$ of the space $L^2(\mathbb{R})$ of square-integrable functions on $\mathbb{R}$, such that

$$\{0\} \subset \ldots \subset V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \subset \ldots \subset L^2(\mathbb{R})$$

and

$$f \in V_m \Leftrightarrow f^* \in V_{m-1},$$

with $f^*(t) = f(2t)$, i.e. the spaces have a different resolution. Moreover, there exists a so-called scaling function $\phi \in V_0$ such that the translated and dilatated versions $\phi_{m,n}(t) = 2^{-m/2}\phi(2^{-m}t - n),\ n \in \mathbb{Z}$ of ϕ form a basis of V_m for all $m \in \mathbb{Z}$. Without loss of generality, ϕ can be chosen such that the $\phi_{m,n}$ even form an orthonormal basis of V_m. Define now W_m as the orthogonal complement of V_m in V_{m-1}, i.e.

$$V_m \perp W_m,\ V_{m-1} = V_m \oplus W_m.$$

Then the spaces W_m are mutually orthogonal with direct sum $L^2(\mathbb{R})$. One can further construct a function $\psi \in W_0$ from ϕ (the so-called mother wavelet), such that the functions $\psi_{m,n}(t) = 2^{-m/2}\psi(2^{-m}t - n)$ form an orthonormal basis of W_m for all $m \in \mathbb{Z}$. Combining these results yields $L^2(\mathbb{R}) = V_{M_0} \oplus \left(\bigoplus_{m=-\infty}^{M_0} W_m\right)$ for some resolution $M_0 \in \mathbb{Z}$, i.e. a function $x \in L^2(\mathbb{R})$ can be written as

$$x(t) = \sum_{n\in\mathbb{Z}} c_{M_0,n}\phi_{M_0,n}(t) + \sum_{m=-\infty}^{M_0}\sum_{n\in\mathbb{Z}} d_{m,n}\psi_{m,n}(t)$$

with $c_{M_0,n} = \langle x, \phi_{M_0,n}\rangle_2$ and $d_{m,n} = \langle x, \psi_{m,n}\rangle_2$. In practical applications, x will be observed on a finite grid $\{t_1, \ldots, t_S\}$, and thus the infinite sums will be truncated, which corresponds to the approximation in (2.3) with the coefficient vector θ being a combination of the coefficients $c_{M_0,n}$ and $d_{m,n}$. If S is a power of 2, Mallat's pyramid algorithm (Mallat, 1989) describes how to efficiently calculate $c_{M_0,n}$ and $d_{m,n}$.

As in the case of splines, functions on higher-dimensional domains can be represented as combinations of univariate bases. Given a one-dimensional multi-resolution analysis, a two-dimensional version can be defined via $V_m^2 := V_m \oplus V_m$ with a scaling function $\phi^{(2)}(t_x, t_y) = \phi(t_x)\phi(t_y)$ that yields basis functions

$$\phi^{(2)}_{m,n}(t_x, t_y) = 2^{-m}\phi^{(2)}(2^{-m}t_x - n_x, 2^{-m}t_y - n_y) = \phi_{m,n_x}(t_x)\phi_{m,n_y}(t_y)$$

of V_m^2 with $n = (n_x, n_y)$. Further $V_{m-1}^2 = V_m^2 \oplus [(V_m \oplus W_m) \oplus (W_m \oplus V_m) \oplus (W_m \oplus W_m)]$, i.e. the orthogonal complement W_m^2 of V_m^2 in V_{m-1}^2 is spanned by orthonormal basis functions

$$\psi^{(2,l)}_{m,n}(t_x, t_y) = 2^{-m}\psi^{(l)}(2^{-m}t_x - n_x, 2^{-m}t_y - n_y)$$

with $l = 1, 2, 3$ and $\psi^{(1)}(t_x, t_y) = \phi(t_x)\psi(t_y), \psi^{(2)}(t_x, t_y) = \psi(t_x)\phi(t_y)$ and $\psi^{(3)}(t_x, t_y) = \psi(t_x)\psi(t_y)$. An efficient algorithm for calculating the basis coefficients is also derived in Mallat (1989).

As the basis functions are defined locally and have different resolution, it is possible to represent most functions, even those with sharp, highly localized features, well using only

a few basis functions. The coefficients of all other basis functions are thus close to zero and model more or less "noise". This is the starting point e.g. for regression models that represent an unknown function in terms of wavelets and apply variable selection in order to achieve a representation that is as sparse as possible (e.g. Wand and Ormerod, 2011; Reiss, Huo, et al., 2015). The effect of such assumptions in the context of scalar-on-image regression is also discussed in Chapter 6.

2.2.3. Functional Principal Component Analysis

Functional principal component analysis (FPCA) is a key concept in functional data analysis and forms the basis for many functional data methods. It builds on the *covariance operator* V of X, which is defined by

$$(Vf)(t) = \int_{\mathcal{T}} c(s,t)f(s)\mathrm{d}s, \quad f \in L^2(\mathcal{T}),$$

with $c(s,t)$ the covariance function of the process X, as in (2.1). It can be shown that V is a symmetric, positive (in the sense of having non-negative eigenvalues) Hilbert-Schmidt operator in $L^2(\mathcal{T})$, if $\mathbb{E}(\|X\|_2^{\,2}) < \infty$ (see e.g. Horváth and Kokoszka, 2012, Chapter 2.3). In this case, Mercer's Theorem (Mercer, 1909) gives

$$c(s,t) = \sum_{m=1}^{\infty} \lambda_m \phi_m(s)\phi_m(t),$$

where $\lambda_1 \geq \lambda_2 \geq \ldots \geq 0$ are the eigenvalues of V and $\{\phi_m \colon m \in \mathbb{N}\}$ the associated orthonormal eigenfunctions, or *functional principal components*, which are uniquely defined up to a sign change. The pairs (λ_m, ϕ_m) are thus solutions to the eigenequation

$$V\phi_m = \lambda_m \phi_m, \quad m \in \mathbb{N}. \tag{2.7}$$

It is easy to show that ϕ_1 maximizes $\langle V\phi, \phi\rangle_2$ subject to $\|\phi\|_2 = 1$, i.e. ϕ_1 explains the most important mode of variation in X, and the subsequent principal components ϕ_m maximize $\langle V\phi, \phi\rangle_2$ subject to $\|\phi\|_2 = 1$ and $\langle \phi, \phi_k\rangle_2$ for all $k < m$, provided that the corresponding eigenvalues are distinct (e.g. Horváth and Kokoszka, 2012, Chapter 3.1.). FPCA is thus a natural extension of multivariate principal component analysis to the functional case (Horváth and Kokoszka, 2012; Ramsay and Silverman, 2005).

The Karhunen-Loève Theorem, named after Karhunen (1947) and Loève (1946), expands the random process X in its principal components

$$X(t) = \mu(t) + \sum_{m=1}^{\infty} \xi_m \phi_m(t), \quad t \in \mathcal{T}$$

with random variables $\xi_m = \langle X, \phi_m \rangle_2$ that satisfy $\mathbb{E}(\xi_m) = 0,\ \mathrm{Var}(\xi_m) = \lambda_m$. In practice, one often observes that the eigenvalues λ_m decrease quite rapidly, meaning that the first few principal components contribute a large proportion of the variability and thus the information in X. Realization $x_1, \ldots, x_N$ of X can hence be written as

$$x_i(t) = \mu(t) + \sum_{m=1}^{\infty} \xi_{i,m}\phi_m(t) \approx \mu(t) + \sum_{m=1}^{M} \xi_{i,m}\phi_m(t), \quad t \in \mathcal{T},\ i = 1, \ldots, N \tag{2.8}$$

for some truncation lag M. The approximation in (2.8) can be seen as special basis function representation of $x_i - \mu$ as in (2.3) where the basis functions, which in this case are the functional principal components, depend on the process X through $c(s,t)$. The *functional principal component scores* $\xi_{i,m}$ represent individual weights for each observation x_i and each principal component ϕ_m.

In practice, the covariance function $c(s,t)$ – and thus the principal components, the associated eigenvalues and the principal component scores – are unknown and have to be estimated based on observed data $x_i(t_{i,s})$, where the observation points $t_{i,s} \in \mathcal{T},\ s = 1, \ldots, S_i$ can be the same for all $i = 1, \ldots, N$ or differ between the realizations.

One possible estimation strategy is to expand the observed data in a fixed basis and reduce the estimation of the FPCA to a matrix eigenanalysis problem, possibly including numerical integration, if the basis is not orthonormal (cf. Ramsay and Silverman, 2005, Chapter 8.4.). This gives estimated eigenvalues $\hat{\lambda}_m$ and principal components $\hat{\phi}_m$. The maximum number of principal components that can be estimated this way is equal to K, the number of basis functions used. The functional principal component scores can be found by numerically approximating

$$\hat{\xi}_{i,m} = \langle x_i, \hat{\phi}_m \rangle_2 = \int_{\mathcal{T}} x_i(t)\hat{\phi}_m(t)\mathrm{d}t. \tag{2.9}$$

The goodness of the estimated FPCA clearly depends on the accuracy of the basis function representation for x_i and also on the number and location of observation points $t_{i,s}$, as this influences the quality of the numerical integration, particularly needed in (2.9).

An alternative estimation approach, which is especially suitable for sparse functional data, having only a few observations for each realization x_i at potentially different observation points $t_{i,s}$, has been proposed in Yao et al. (2005). This approach makes use of "borrowing strength" across the realizations by smoothly estimating the mean and the covariance function from the pooled data, e.g. by using local polynomial smoothers (Fan and Gijbels, 1997), as originally proposed in Yao et al. (2005), or based on a penalized spline approach (Di et al., 2009), as implemented in the R-package `refund` (Goldsmith, Scheipl, et al., 2016). The proposed method can also handle data that is observed with measurement error:

$$\tilde{x}_{i,s} = x_i(t_{i,s}) + \varepsilon_{i,s}, \quad s = 1, \ldots, S_i,\ i = 1, \ldots, N$$

with $\varepsilon_{i,s} \overset{\text{iid}}{\sim} \mathrm{N}(0, \sigma^2)$ for some $\sigma^2 > 0$. In this case, the diagonal of the covariance function will show some additional "spikes" due to the independence assumption of the errors $\varepsilon_{i,s}$ and thus has to be treated separately. More details are given in Yao et al. (2005). For the estimation of the principal component scores, the authors propose the PACE (principal components analysis through conditional expectation) approach, which is based on the assumption that the scores $\xi_{i,m}$ and the measurement errors $\varepsilon_{i,s}$ are jointly Gaussian. In this case, the best prediction of the principal component score $\xi_{i,m}$ conditional on the observed data for observation unit i, is given by

$$\tilde{\xi}_{i,m} = \mathbb{E}\left(\xi_{i,m} | \tilde{x}_i\right) = \lambda_m \Phi_{i,m}^\top \Sigma_i^{-1} (\tilde{x}_i - \mu_i)$$

with $\tilde{x}_i = (\tilde{x}_{i,1}, \ldots, \tilde{x}_{i,S_i})$, $\Phi_{i,m} = (\phi_m(t_{i,1}), \ldots, \phi_m(t_{i,S_i}))$, $\Sigma_i \in \mathbb{R}^{S_i \times S_i}$ with entries $\Sigma_{i,rs} = c(t_{i,r}, t_{i,s}) + \sigma^2 \mathbb{1}\{r = s\}$ and $\mu_i = (\mu(t_{i,1}), \ldots, \mu(t_{i,S_i}))$. If the Gaussianity assumption does not hold, $\tilde{\xi}_{i,m}$ is still the best linear prediction of $\xi_{i,m}$ given the observed data from subject i (Yao et al., 2005). Estimated scores can be found by replacing all theoretical quantities by their empirical counterparts. For a detailed discussion of the theoretical properties of the estimators, see Yao et al. (2005).

For functional data on higher-dimensional domains, Allen (2013) has introduced a method for finding smooth principal components, following ideas in Huang, Shen, et al. (2009). General asymptotic theory for principal component analysis can e.g. be found in the book of Bosq (2000) or in Hall and Hosseini-Nasab (2006).

In the case of multivariate functional data, several principal component methods have been proposed in recent years. All of these approaches are restricted to functions observed on the same one-dimensional interval: Ramsay and Silverman (2005) briefly outline *multivariate functional principal component analysis* (MFPCA) for bivariate functional data by an illustrative example of gait data. They suggest to stack the observations of the two functions and perform a standard (univariate) FPCA for the combined observations. The estimated principal components are then separated into two parts, corresponding to the original two functions. Jacques and Preda (2014) introduce MFPCA in the context of clustering of multivariate functional data with p elements. Their approach for MFCPA is somewhat related to the ideas in Ramsay and Silverman (2005), as they propose to concatenate the observed functions to a vector of functions. MFPCA is then based on the associated covariance operator, using theory developed in Saporta (1981). The authors propose an estimation procedure for multivariate functional principal components and principal component scores by expanding each element of the multivariate functional data in terms of a given basis. The basis functions can be chosen non-orthonormal and may also differ for each element. In case that the different elements of the multivariate functional data are measured e.g. in different units, the authors propose normed MFPCA by replacing the covariance operator by a normalized version of it. The approach of Chiou, Yang, et al. (2014) for normalized MFPCA goes in the same direction, but uses a slightly different normalization approach

that leads to a normalized Karhunen-Loève representation of the data. Inspired by Yao et al. (2005), they give an estimation method of normalized MFPCA for data that may be observed with measurement error and discuss asymptotic properties of the approach. An alternative approach to MFPCA is introduced by Berrendero et al. (2011), who perform a standard multivariate PCA for each observation point, and then interpolate the results. Hence, each observation can be characterized by one or a few score functions, rather than a score vector. This approach differs considerably from the previously described methods, as it does not aim at a Karhunen-Loève representation of the multivariate functional data.

The new MFPCA approach proposed in Chapter 3 overcomes the restriction that the domain of all elements has to be one-dimensional and the same for all elements. In this way, it becomes possible to calculate principal components of multivariate functional data consisting e.g. of functions and images (cf. Chapters 3 to 5).

2.2.4. Functional Regression Models

One main goal of statistics is to find relationships between a dependent variable based on independent variable information in terms of regression models (Ramsay and Silverman, 2005, Chapter 1). The standard linear and generalized linear models can easily be extended to functional data, that can take the role of an independent/explaining variable (*scalar-on-function regression*), the depending/response variable (*function-on-scalar regression*) or both (*function-on-function regression*) (see e.g. Morris, 2015; Greven and Scheipl, 2017, for this terminology). In the simplest case of linear regression models, this yields the linear scalar-on-function regression model for scalar responses y_i and functional covariates x_i

$$y_i = \int_{\mathcal{T}} x_i(s)\beta(s)\mathrm{d}s + \varepsilon_i, \quad i = 1, \ldots, N, \tag{2.10}$$

the linear function-on-scalar regression for functional responses y_i and scalar covariates x_i

$$y_i(t) = x_i\beta(t) + \varepsilon_i(t), \quad t \in \mathcal{T},\ i = 1, \ldots, N$$

and the linear function-on-function regression model for both functional responses y_i and covariates x_i

$$y_i(t) = \int_{\mathcal{T}} x_i(s)\beta(s,t)\mathrm{d}s + \varepsilon_i(t), \quad i = 1, \ldots, N.$$

The relation to standard linear models is immediately seen as only the scalar product between the covariates x_i and the coefficient β has to be replaced by its counterpart in $L^2(\mathcal{T})$ when moving from vectors to functions.

In the past few years, there has been a broad interest in these types of regression models, resulting in a vast literature on functional regression. An overview can for example be found in some recent review papers that cover functional data analysis in general (Wang, Chiou,

et al., 2016), functional regression models (Morris, 2015) or scalar-on-function regression models (Reiss, Goldsmith, et al., 2016+). The discussion paper of Greven and Scheipl (2017) shows how many of the existing functional regression models can be comprised into a flexible overall framework for functional regression. In the following, the focus is on the scalar-on-function regression model (2.10), which can be seen as a generalized version of scalar-on-image regression, that is discussed in Chapter 6 with particular attention on the impact that model assumptions can have on the estimates.

A very common approach for estimating the unknown coefficient function β in (2.10) is to expand it in a given basis (Ramsay and Silverman, 2005, Chapter 15), as in (2.3). This already implies an approximation, as discussed before:

$$y_i = \int_{\mathcal{T}} x_i(t)\beta(t)\mathrm{d}t + \varepsilon_i \approx \sum_{k=1}^{K} b_k \int_{\mathcal{T}} x_i(t)B_k(t)\mathrm{d}t + \varepsilon_i, \quad i = 1, \ldots, N.$$

The right hand side of this equation is in effect a standard linear regression model in the coefficients b_k and covariates $z_{i,k} = \int_{\mathcal{T}} x_i(t)B_k(t)\mathrm{d}t$, that have to be approximated numerically based on the observed values $x_i(t_{i,s})$. Given an estimate $\hat{b}$ for the coefficient vector b, the estimated coefficient function is obtained by a simple plug-in principle:

$$\hat{\beta}(t) = \sum_{k=1}^{K} \hat{b}_k B_k(t), \quad t \in \mathcal{T}.$$

The model is identifiable if the number of observations N is larger or equal to the number of basis functions K. A possible remedy in the case of non-identifiability is to use e.g. P-splines with an appropriate penalty in order to obtain a smooth coefficient function β (Marx and Eilers, 1999; Cardot et al., 2003). As long as the kernel of the covariance operator V, which is the space spanned by eigenfunctions of V associated with the eigenvalue $\lambda = 0$, is null or has no overlap with the space of unpenalized coefficient functions β, parametrized by coefficient vectors b in the kernel of the penalty matrix, the model remains identifiable even if $N < K$. A detailed discussion of this issue including diagnostic tools and proposals for countermeasures is given in Happ (2013) for scalar-on-function regression and more general in Scheipl and Greven (2016) for function-on-function regression.

Alternatively, the functional covariates x_i can be expanded in the same basis functions as β, which is particularly useful if the basis functions are orthonormal, as it is the case for functional principal components (Cardot et al., 1999; Müller and Stadtmüller, 2005). In this case, model (2.10) can be approximated as

$$y_i = \int_{\mathcal{T}} x_i(t)\beta(t)\mathrm{d}t + \varepsilon_i \approx \sum_{k=1}^{K} b_k \sum_{m=1}^{M} \xi_{i,m} \int_{\mathcal{T}} \phi_k(t)\phi_m(t)\mathrm{d}t + \varepsilon_i = \sum_{m=1}^{\min(M,K)} b_m \xi_{i,m} + \varepsilon_i, \quad i = 1, \ldots, N,$$

where the last equality is due to the orthonormality of the principal components. The scalar-on-function regression model hence reduces to a standard linear model in the coefficients b_m and the functional principal component scores $\xi_{i,m}$ and does not require numerical

integration as it makes use of the orthonormality of the principal components. In practice, the functional principal components ϕ_m and the scores $\xi_{i,m}$ have to be replaced by their estimates found via FPCA (cf. Section 2.2.3). The model is identifiable if the number of observation units N is larger or equal to $\min(M, K)$. In applications one often observes that the data can be represented well by the first few principal components, meaning that with $M < 5$ one can often explain more than 95% of the variance in the data (as e.g. in the univariate FPCA for the ADAS-Cog trajectories in Chapters 3 and 4) and hence the model is almost always identifiable. At the same time, however, this results in an immense dimension reduction and is the reason for a frequent criticism of this last FPCA based approach: The coefficient function β is expanded in the first eigenfunctions of the covariance operator V of X, which implicitly assumes that β has the same leading modes of variation as the data (Goldsmith, Bobb, et al., 2011; Happ, 2013). Truncating the data at $M = 3$, say, means that the unknown coefficient function is in effect assumed to lie in the three-dimensional space spanned by the first three eigenfunctions, as only the first $\min(M, K)$ coefficients b_m enter the final regression model. When estimating a larger number of principal components, the principal components of higher order tend to be often quite wiggly, which of course transfers to the β coefficient functions. Penalized versions which aim at reducing the influence of the higher order principal components (James and Silverman, 2005; Happ, 2013) seem to provide promising solutions in this case.

2.3. Bayesian Methods for Image Data

Bayesian methods have a long and successful history in statistical image analysis (Geman and Geman, 1984; Besag et al., 1991) and provide an alternative approach to include image data in a statistical model. Here, the images are no longer treated as discrete evaluations of two-dimensional functions, but as the collection of all pixels, including information about their spatial structure. In the scalar-on-image regression model, which is discussed in Chapter 6, observed images $x_1, \ldots, x_N$ are used to explain response values $y_1, \ldots, y_N \in \mathbb{R}$. The relationship is established through an unknown coefficient image β that has the same structure as the observed images

$$y_i = \sum_{l=1}^{L} \beta_l x_{i,l} + \varepsilon_i, \quad i = 1, \ldots, N,$$

where L denotes the number of pixels. For the error terms one uses the standard assumption $\varepsilon_i \overset{\text{iid}}{\sim} \mathrm{N}(0, \sigma_\varepsilon^2)$, where the notation σ_ε emphasizes that this is the variance of ε_i. The model can be extended to include some more scalar covariates, contained in a vector $w_i \in \mathbb{R}^p$ for each observation $i = 1, \ldots, N$ with a corresponding coefficient vector α:

$$y_i = w_i^\top \alpha + \sum_{l=1}^{L} \beta_l x_{i,l} + \varepsilon_i, \quad i = 1, \ldots, N.$$

A reformulation of the model, which is common in the Bayesian literature, stresses the dependency of the response y_i on both the data and the parameters (Goldsmith, Huang, et al., 2014):

$$y_i \Big| w_i, x_i, \alpha, \beta, \sigma_\varepsilon^2 \overset{\text{iid}}{\sim} \mathrm{N}\left(w_i^\top \alpha + \sum_{l=1}^{L} \beta_l x_{i,l}, \sigma_\varepsilon^2 \right), \quad i = 1, \ldots, N. \tag{2.11}$$

The following sections give a short overview of Bayesian methods in general and explain how they can be applied to the special case of image data. A general introduction to Bayesian methods can for example be found in the textbooks of Carlin and Louis (2009) or Gelman, Carlin, et al. (2014). Statistical methods for image analysis with a focus on Bayesian models are presented e.g. in Winkler (2003) or Meyer-Baese and Schmid (2014).

2.3.1. The Bayesian Paradigm

Consider a generic statistical model with data $\mathcal{X}$ and parameters $\theta \in \Theta$. In the case of scalar-on-image regression, the data would be given by $\mathcal{X} = \{(y_i, w_i, x_i) \colon i = 1, \ldots, N\}$ and the parameters can be combined into a vector $\theta = (\alpha, \beta, \sigma_\varepsilon^2) \in \mathbb{R}^p \times \mathbb{R}^L \times (0, \infty)$. In a frequentist approach, the parameters are treated as fixed and conclusions about θ are mostly drawn from the *likelihood* $p(\mathcal{X}|\theta)$, a term introduced by Fisher (1922), which is the density of the data, considered as a function of the parameters.

The Bayesian approach, in contrast, treats the parameter vector θ as a random variable, having a distribution F_θ, which is called the *prior* distribution of θ with a density $p(\theta)$. The prior can be chosen based on expert knowledge, e.g. from previous similar experiments. If no reliable prior information on θ is available, it can be chosen to contain as little information as possible (*uninformative prior*). Alternatively, the prior can be chosen in a mathematically convenient way (so-called *conjugate prior*). The advantages and disadvantages of the different choices are for example discussed in Carlin and Louis (2009, Chapter 2). As it is seen later on, the prior needs not necessarily be a *proper* density, meaning that it can integrate to other values than 1.

The prior is combined with the likelihood according to Bayes' Theorem, dating back to Bayes (1763),

$$p(\theta|\mathcal{X}) = \frac{p(\mathcal{X}|\theta)\, p(\theta)}{\int_\Theta p(\mathcal{X}|\vartheta)\, p(\vartheta) \mathrm{d}\vartheta}, \quad \theta \in \Theta. \tag{2.12}$$

The distribution of θ given the data $\mathcal{X}$ is called the *posterior* and provides the basis for the statistical analysis, such as e.g. inference or model selection. The marginal

$$m(\mathcal{X}) = \int_\Theta p(\mathcal{X}|\vartheta)\, p(\vartheta) \mathrm{d}\vartheta$$

ensures that the posterior integrates to 1 and thus forms a valid probability density, as long as prior and likelihood contain enough information. In particular, *improper* priors can still lead to a proper posterior provided that the likelihood integrates to some finite value (Carlin and Louis, 2009, Chapter 2). As the marginal $m(\mathcal{X})$ is constant with respect to θ, one often writes shorthand

$$p\left(\theta|\mathcal{X}\right) \propto p\left(\mathcal{X}|\theta\right) p(\theta), \tag{2.13}$$

as the right hand side uniquely defines $m(\mathcal{X})$ and thus the posterior.

The prior distribution for θ may of course depend on other parameters $\eta \in H$, the so-called *hyperparameters*. These values can either be treated as fixed or they can also be assumed to be random variables, having a distribution and a density $p(\eta)$. In this case, model (2.12) becomes (Carlin and Louis, 2009, Chapter 2)

$$p\left(\theta, \eta|\mathcal{X}\right) = \frac{p\left(\mathcal{X}|\theta\right) p\left(\theta|\eta\right) p(\eta)}{\int_\Theta \int_H p\left(\mathcal{X}|\vartheta\right) p\left(\vartheta|\zeta\right) p(\zeta) \mathrm{d}\zeta \, \mathrm{d}\vartheta}, \quad \theta \in \Theta, \ \eta \in H$$

where the notation $p(\theta|\eta)$ emphasizes that the prior for θ depends on η. The *hyperprior* $p(\eta)$ can again depend on parameters, that in principle could be integrated into the model in an analogous way, contributing their own priors and so forth. This leads to so-called *hierarchical models*, where each additional prior contributes a new level in the hierarchy. In practice, the number of levels is mostly restricted to a small number such as two or three, as each new level contributes additional parameters to the model with decreasing impact on the posterior (Carlin and Louis, 2009, Chapter 2). In the remainder of this section, there is no distinction between the parameters θ and the hyperparameters η, which means that θ denotes the vector of all parameters in a model that are not treated as fixed.

Especially if models contain many parameters, the posterior becomes high-dimensional and thus difficult to handle. In this case, one often considers the so-called *full conditional* of a single parameter or a subgroup of parameters θ_* of θ, meaning the full conditional posterior density of this parameter given the data $\mathcal{X}$ and all other parameters θ_{-*}

$$p\left(\theta^*|\cdot\right) := p\left(\theta_*|\mathcal{X}, \theta_{-*}\right).$$

The full conditional is easily seen to be proportional to the joint posterior of all parameters (Carlin and Louis, 2009, Chapter 3), i.e.

$$p\left(\theta_*|\mathcal{X}, \theta_{-*}\right) \propto p\left(\theta|\mathcal{X}\right). \tag{2.14}$$

This relation is used in some Markov-Chain-Monte-Carlo methods to draw samples from the posterior based on the full conditionals (see Section 2.3.4).

2.3.2. Random Fields as Latent Priors in Hierarchical Bayesian Models

Going back to the scalar-on-image regression model (2.11), an appropriate prior distribution is needed for the parameter vector $\theta = (\alpha, \beta, \sigma_\varepsilon^2)$. It is common to assume the model parameters to be independent a priori, such that $p(\theta) = p(\alpha)p(\beta)p(\sigma_\varepsilon^2)$ (Gelman, Carlin, et al., 2014, Chapter 14). For the coefficient vector α of the scalar parameters and the error variance σ_ε^2 one can use standard choices such as $p(\alpha) \propto 1$, which is an improper prior, and $\sigma_\varepsilon^2 \sim \mathrm{IG}(\delta_\varepsilon^{(1)}, \delta_\varepsilon^{(2)})$, which is (semi-)conjugate, as it can be shown that the full conditional distribution of σ_ε^2 given the data and the other parameters is again an element of the family of inverse-gamma distributions (e.g. Carlin and Louis, 2009, Chapter 2).

The parameter β is an image with L pixels, having a strong spatial structure, which should be reflected appropriately in the prior. *(Markov) random fields* (Besag, 1974) and in particular *Gaussian Markov random fields* (GMRFs, for an introduction see e.g. Rue and Held, 2005) have proven to be suitable priors for image data (e.g. Besag et al., 1991; Winkler, 2003; Meyer-Baese and Schmid, 2014). The Markov property (Besag, 1974) assumes that the value of the coefficient image in a pixel l depends only on the values of β in the neighbourhood of this pixel and is independent of all other values:

$$p\left(\beta_l | \beta_{-l}\right) = p\left(\beta_l \middle| \beta_{\delta(l)}\right).$$

Here β_{-l} denotes the set of all pixels in β without β_l and $\beta_{\delta(l)}$ is the set of all neighbouring coefficients, i.e. $\beta_{\delta(l)} = \{\beta_j \colon j \sim l\}$, where $j \sim l$ means that the pixels j and l are neighbours. A neighbourhood in this case means for example the set of all pixels sharing an edge with l. In particular, a neighbourhood structure must be symmetric (i.e. if $j \sim l$, then l must also be a neighbour of j) and the neighbourhood must not contain the pixel itself ($l \not\sim l$) (Winkler, 2003, Chapter 3). Some typical examples for neighbourhoods in the case of two-dimensional images are shown in Fig. 2.1. With the Markov property, the choice of the neighbourhood defines the dependence structure in β.

In an (intrinsic) GMRF, the conditional distribution of β_l given the values of β in the neighbourhood $\beta_{\delta(l)}$ and an additional variance parameter σ_β^2 is assumed to be Gaussian

$$\beta_l \middle| \beta_{\delta(l)}, \sigma_\beta^2 \sim \mathrm{N}\left(\frac{1}{d_l}\sum_{j \sim l} \beta_j, \frac{\sigma_\beta^2}{d_l}\right)$$

with $d_l = \#\{j = 1, \ldots, L \colon j \sim l\}$ the number of neighbours of l (cf. Besag, 1974; Rue and Held, 2005). The prior assumption for β can be rewritten in unconditional form using the Hammersley-Clifford Theorem (Besag, 1974; Rue and Held, 2005)

$$p\left(\beta \middle| \sigma_\beta^2\right) \propto (\sigma_\beta^2)^{-\operatorname{rank}(P)/2} \exp\left(-\frac{1}{2\sigma_\beta^2}\beta^\top P \beta\right)$$

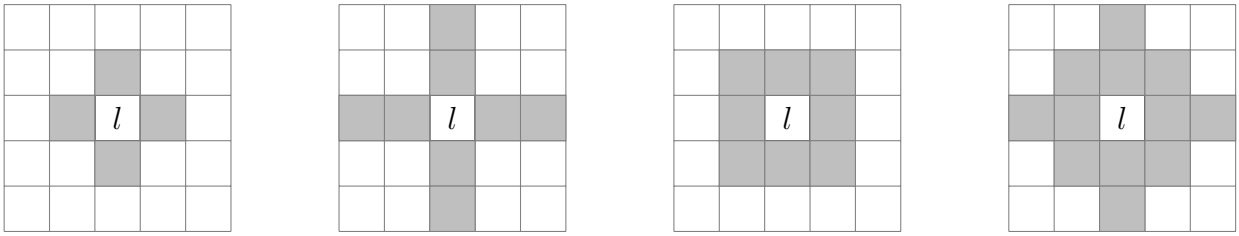


Figure 2.1.: Typical examples for neighbourhood structures in the case of two-dimensional images. The gray areas mark the neighbourhood of the pixel l in the center (boundary cases not shown).

with $P \in \mathbb{R}^{L \times L}$ the so-called *neighbourhood matrix* with

$$p_{jl} = \begin{cases} d_l & j = l \\ -1 & j \sim l \\ 0 & \text{else.} \end{cases}$$

This is not a proper distribution, as P does not have full rank ($\text{rank}(P) = L - 1$, Rue and Held, 2005, Chapter 3). However, this prior assumption leads to a proper posterior distribution in most cases of interest and to a Gaussian full conditional for β.

The variance parameter σ_β^2 in the prior for β is a hyperparameter. It can either be treated as fixed or e.g. assumed to have an inverse-gamma distribution, which can be shown to be conjugate in this case (Meyer-Baese and Schmid, 2014, Chapter 2):

$$\sigma_\beta^2 \sim \text{IG}(\delta_\beta^{(1)}, \delta_\beta^{(2)}).$$

2.3.3. Relation to Penalized Basis Function Approaches

The Bayesian approach for scalar-on-image regression (or more general models) with GMRF priors has an interesting relation to penalized spline methods, as indicated in Section 2.2.2. The natural logarithm of the full conditional of β is easily shown to be of the form

$$-\frac{1}{2\sigma_\varepsilon^2}\left[\sum_{i=1}^{N}\left(y_i - \sum_{j=1}^{p} w_{i,j}\alpha_j - \sum_{l=1}^{L} x_{i,l}\beta_l\right)^2 + \frac{\sigma_\varepsilon^2}{\sigma_\beta^2}\sum_{j\sim l}(\beta_j - \beta_l)^2\right] + C$$

with some constant C. Maximizing this quantity, which gives the posterior mode as a point estimate for β, is equivalent to minimizing the squared residuals $y_i - \sum_{j=1}^{p} w_{i,j}\alpha_j - \sum_{l=1}^{L} x_{i,l}\beta_l$ with respect to a quadratic first order difference penalty on the coefficients with some smoothing parameter $\lambda = \frac{\sigma_\varepsilon^2}{\sigma_\beta^2}$. The GMRF approach thus corresponds to a penalized basis function approach with constant local basis functions $\mathbb{1}_l$ for each pixel.

2.3.4. Bayesian Inference and Markov-Chain-Monte-Carlo Methods

Bayesian inference is based on the posterior, i.e. the density of all parameters given the data. As this might be a high-dimensional distribution and thus hard to interpret, one can use Bayesian versions of point estimates, interval estimates and hypothesis tests to summarize the information in the posterior (Carlin and Louis, 2009, Chapter 2).

Examples for Bayesian point estimates for θ are the posterior expectation

$$\mathbb{E}\left(\theta|\mathcal{X}\right) = \int_{\Theta} \theta \; p\left(\theta|\mathcal{X}\right) \mathrm{d}\theta \tag{2.15}$$

or, analogously, the posterior mode or the posterior median. *Credible intervals* (more general: credible sets) are the Bayesian counterparts to confidence intervals. A set $C \subset \Theta$ is said to be a $100 \cdot (1-\alpha)$ percent credible set for the parameter θ, if

$$P\left(C|\mathcal{X}\right) = \int_{C} p\left(\theta|\mathcal{X}\right) \mathrm{d}\theta \geq 1-\alpha.$$

In contrast to a frequentist $100 \cdot (1-\alpha)$ percent confidence interval, C can be easily interpreted as the probability for $\theta \in C$ being greater or equal to $1-\alpha$, given the observed data $\mathcal{X}$. More details can be found e.g. in Carlin and Louis (2009, Chapter 2).

For simple models with only one or two parameters, the posterior can often be found analytically. Particularly for parameters with conjugate priors it is then easy to derive e.g. point estimates, as the posterior is a known distribution in this case and thus the posterior mean is mostly just a function of the updated parameters of the posterior distribution (with respect to the prior). For more complex models such as scalar-on-image regression, the posterior is high-dimensional and thus the theoretical calculation of point estimates for instance is hardly feasible, as it involves integration (posterior expectation) or optimization (posterior mode) in a high-dimensional space. Instead, the theoretical point estimates or credible intervals can be approximated by their empirical counterparts using a high number of samples from the posterior. In the case of the posterior expectation, for example, this means approximating the integral in (2.15) by so-called *Monte Carlo integration* (Carlin and Louis, 2009, Chapter 3)

$$\int_{\Theta} \theta \; p\left(\theta|\mathcal{X}\right) \mathrm{d}\theta \approx \frac{1}{J} \sum_{j=1}^{J} \theta_j \quad \text{with } \theta_j \overset{\text{iid}}{\sim} p\left(\theta|\mathcal{X}\right), \; j = 1, \ldots, J. \tag{2.16}$$

This of course requires the availability of efficient sampling methods for easily generating hundreds or thousands of samples from the posterior, which are the condition for a good approximation in (2.16).

In this context, Markov-Chain-Monte-Carlo methods (MCMC) have proven extremely useful and have helped to pave the way to a broad applicability of high-dimensional Bayesian

models (Geyer, 2011; Gilks et al., 1996). Starting from some given initial values for the parameters, they construct a Markov chain having the posterior as the *stationary* or *invariant* distribution. For a review of Markov chain theory with a special focus on MCMC, see for example Tierney (1996) or Geyer (2011).

The *Metropolis-Hastings algorithm*, named after Metropolis et al. (1953) and Hastings (1970), describes how to construct a Markov chain with a given stationary distribution π, which in the case of Bayesian approaches will always be the posterior, thus $\pi(\theta) = p(\theta|\mathcal{X})$. Starting from $\theta^{(0)} \in \Theta$ with $\pi(\theta^{(0)}) > 0$ and given the state of the chain $\theta^{(j)}$ in iteration $j \geq 0$, the state in the following iteration $j+1$ is generated as follows:

1. Sample a candidate θ^* from a proposal distribution q, that may depend on the current state $\theta^{(j)}$
$$\theta^* \sim q\left(\cdot\middle|\theta^{(j)}\right).$$

2. Accept θ^* with probability $\alpha(\theta^{(j)}, \theta^*)$, where
$$\alpha(\theta, \theta^*) = \min\left(1, \frac{\pi(\theta^*) q\left(\theta|\theta^*\right)}{\pi(\theta) q\left(\theta^*|\theta\right)}\right). \tag{2.17}$$

Acceptance means that $\theta^{(j+1)} = \theta^*$, otherwise $\theta^{(j+1)} = \theta^{(j)}$. Note that for the calculation of $\alpha(\theta, \theta^*)$ the normalizing constant in π cancels out, as one considers only the ratio $\frac{\pi(\theta^*)}{\pi(\theta)}$. It is therefore sufficient to know the posterior only up to the normalizing constant, as in (2.13).

After an initial phase (*burnin*), in which the values of the chain still depend on the starting values, the states of the chain can be considered as samples from the limiting distribution, and thus as samples from the posterior (Gilks et al., 1996). By construction, the samples are not independent from each other, which can be problematic if mixing is slow and one has to save a lot of highly correlated samples. To this end, one often uses *thinning*, meaning that one saves only each n-th state of the chain, with e.g. $n = 10$ or $n = 20$, which also reduces the dependence among the samples (Gelman and Shirley, 2011). Both concepts of burnin and thinning have been the subject of criticism, as they throw away information in form of samples. For a general discussion, see e.g. Geyer (2011).

The proposal distribution q can be chosen quite freely, but one should bear in mind that sampling from this distribution should be easily achievable, as for each state of the chain, a sample has to be drawn from q (Gilks et al., 1996) and usually, the number of iterations will be in the order of thousands or millions. Moreover, the relationship between q and the posterior π is crucial for convergence, as a higher acceptance rate in (2.17) and good *mixing* properties lead to a better exploration of the posterior (Gilks et al., 1996). For regularity conditions for q, see e.g. Geyer (2011).

The update scheme can be modified by updating sub-blocks of θ at a time. Let $\theta = (\theta_1, \ldots, \theta_h)$ be a division of all model parameters in blocks, where each block θ_k might be one- or higher-dimensional. In the case of scalar-on-image regression as in (2.11), the parameter θ might be split in $h = 3$ blocks, namely $\theta_1 = \alpha \in \mathbb{R}^p, \theta_2 = \beta \in \mathbb{R}^L$ and $\theta_3 = \sigma_\beta^2 \in (0, \infty)$. Then the update from iteration j to iteration $j+1$ comprises h updating steps, one for each block. Let $\theta_k^{(j)}$ be the state of the k-th block after iteration j and denote $\theta_{-k}^{(j)} = (\theta_1^{(j+1)}, \ldots, \theta_{k-1}^{(j+1)}, \theta_{k+1}^{(j)}, \ldots, \theta_h^{(j)})$, i.e. all blocks except for θ_k, where the first $k-1$ blocks have already been updated. A candidate θ_k^* for the next state is sampled from the k-th proposal distribution $q_k\left(\cdot \middle| \theta_k^{(j)}, \theta_{-k}^{(j)}\right)$, which depends on the current state of θ_k, but also on the states of the other blocks in θ_{-k}. The proposal is accepted with probability

$$\alpha(\theta_k^{(j)}, \theta_{-k}^{(j)}, \theta_k^*) = \min\left(1, \frac{\pi\left(\theta_*^{(j)} \middle| \theta_{-k}^{(j)}\right) q\left(\theta_k^{(j)} \middle| \theta_k^*, \theta_{-k}^{(j)}\right)}{\pi\left(\theta_k^{(j)} \middle| \theta_{-k}^{(j)}\right) q\left(\theta_k^* \middle| \theta_k^{(j)}, \theta_{-k}^{(j)}\right)}\right).$$

Here $\pi\left(\theta_k | \theta_{-k}\right) = p\left(\theta_k | \theta_{-k}, \mathcal{X}\right)$ is the full conditional of θ_k, given the parameters of all other blocks and the data. This block-wise update scheme is known as *single-component Metropolis-Hastings* (Gilks et al., 1996) or *variable-at-a-time Metropolis-Hastings* (Geyer, 2011). It is particularly useful for hierarchical models with many parameters, as one can specify the proposals separately for each block.

Special cases of the Metropolis-Hastings algorithms are given by the *Metropolis algorithm* (Metropolis et al., 1953), which uses a symmetric proposal density, i.e. $q\left(\theta^* | \theta\right) = q\left(\theta | \theta^*\right)$. This reduces the acceptance probability to $\alpha(\theta, \theta^*) = \min(1, \frac{\pi(\theta^*)}{\pi(\theta)})$. The so-called *Gibbs sampler* (Geman and Geman, 1984; Gelfand and Smith, 1990) is a special case of the single-component Metropolis-Hastings algorithm described before, where the proposal distribution of the k-th block is set to the full conditional of this block, i.e. $q_k\left(\cdot | \theta_k, \theta_{-k}\right) = p\left(\theta_k | \theta_{-k}, \mathcal{X}\right)$. This choice yields $\alpha \equiv 1$, hence the proposals are always accepted. At the same time, it requires all full conditionals to be known distributions and to have appropriate sampling algorithms available.

A frequent criticism of MCMC methods is that they are, in a sense, "black box" algorithms (Geyer, 2011). This means that for example determining convergence of the chain is hardly feasible, although some heuristics have been proposed (cf. Gelman and Shirley, 2011, and references therein). In addition, MCMC methods usually take very long to converge, particularly for high-dimensional models with a strong dependence structure. In this context, Rue, Martino, et al. (2009) have proposed integrated nested Laplace approximation (INLA), which can be applied in the important special case of structured additive regression models with latent Gaussian fields and a few hyperparameters. This broad class of models covers for example spatial models involving Gaussian Markov random fields, as some of the methods for scalar-on-image regression used in Chapter 6. The basic idea here is to approximate posterior marginals based on a repeated application of the Laplace approximation, which

reduces the computation time from hours or days (MCMC) to seconds and minutes (INLA), as argued in Rue, Martino, et al. (2009).

The method of Goldsmith, Huang, et al. (2014) for scalar-on-image regression alternatively proposes to combine a Gibbs sampler for the unknown coefficient image β with K-fold cross-validation for all hyperparameters in the model. They include two parameters of a latent Ising field for implicit variable selection. These parameters do not have a known conjugate distribution and hence Gibbs sampling is not applicable for them. The use of cross-validation, however, has its own pitfalls: First, the computation time considerably increases, as for each parameter combination that is to be considered a full Gibbs sampler has to be run for each of the K folds. In the model of Goldsmith, Huang, et al. (2014), there are in total four hyperparameters. If for each parameter M values have to be tested, this results in running in total $K \cdot M^4$ Gibbs samplers for β in order to find the optimal choice of hyperparameters. Consequently, the number M of values per hyperparameter needs to be extremely small, which is a second drawback (in the simulation in Chapter 6 we use $M = 3, K = 5$ and extremely short Gibbs samplers with only 250 iterations as suggested in Goldsmith, Huang, et al. (2014), but still one fit takes more than 2.5 hours for a moderate number of $N = 250$ individuals and relatively small images of size 64×64). Choosing from only a few values corresponds to a discrete prior distribution with M possible values for each parameter. This is of course highly informative if M is small and can be seen as a quite strong model assumption.

An alternative approach to the model in Goldsmith, Huang, et al. (2014), using conjugate priors for the variance parameters $\sigma_\varepsilon^2, \sigma_\beta^2$ and an auxiliary variable approach (Møller et al., 2006) based on *coupling from the past* (Propp and Wilson, 1996) for the parameters of the latent Ising field, performed rather poorly (not shown here). The results indicated that the choice of the priors and thus of the model assumptions are crucial in complex statistical models such as scalar-on-image regression. This was the starting point for the systematic survey in Chapter 6.

Part I.

Multivariate Functional Principal Component Analysis

3. Multivariate Functional Principal Component Analysis for Data on Different (Dimensional) Domains

This chapter is a reprint[§] of:

C. Happ and S. Greven for the Alzheimer's Disease Neuroimaging Initiative (ADNI) (2017+). "Multivariate Functional Principal Component Analysis for Data Observed on Different (Dimensional) Domains". *Journal of the American Statistical Association*. To appear. DOI: 10.1080/01621459.2016.1273115

Copyright:

Author contributions:

Clara Happ prepared a first draft of the manuscript. Sonja Greven added valuable input, particularly on the inclusion of weights, the application and some parts of the proofs. The asymptotic properties were derived in close collaboration of both authors. All implementations were done by Clara Happ.

Acknowledgements:

The authors thank the Alzheimer's Disease Neuroimaging Initiative (ADNI) Committee for providing access to the data. Financial support from the German Research Foundation through Emmy Noether grant GR 3793/1-1 is gratefully acknowledged.

[§]Up to minor modifications in order to ensure consistency of terminology, e.g. non-zero vs. nonzero.

Abstract:

Existing approaches for multivariate functional principal component analysis are restricted to data on the same one-dimensional interval. The presented approach focuses on multivariate functional data on different domains that may differ in dimension, e.g. functions and images. The theoretical basis for multivariate functional principal component analysis is given in terms of a Karhunen-Loève Theorem. For the practically relevant case of a finite Karhunen-Loève representation, a relationship between univariate and multivariate functional principal component analysis is established. This offers an estimation strategy to calculate multivariate functional principal components and scores based on their univariate counterparts. For the resulting estimators, asymptotic results are derived. The approach can be extended to finite univariate expansions in general, not necessarily orthonormal bases. It is also applicable for sparse functional data or data with measurement error. A flexible `R` implementation is available on CRAN. The new method is shown to be competitive to existing approaches for data observed on a common one-dimensional domain. The motivating application is a neuroimaging study, where the goal is to explore how longitudinal trajectories of a neuropsychological test score covary with FDG-PET brain scans at baseline. Supplementary material, including detailed proofs, additional simulation results and software is available online.

3.1. Introduction

Statistical methods for functional data have become increasingly important in recent years. Functional principal component analysis (FPCA) is one of the key techniques in functional data analysis, as it provides an easily interpretable exploratory analysis of the data. Further, it is an important building block for many statistical models (see e.g. Ramsay and Silverman, 2005). The technical progress in many fields of application allows the collection of more and more data with functional features, often several kinds per observation unit. This encourages the study of multivariate functional data and new methods are required to reveal e.g. joint variation in the different elements.

As a simple motivating example, consider the gait cycle data (Ramsay and Silverman, 2005) shown in Fig. 3.1. It contains 39 observations of hip and knee angle during a gait cycle on a standardized time interval. Both elements of this bivariate data can be described separately by their first three univariate eigenfunctions that explain 94.4% (hip) and 87.5% (knee) of the total variability in the data. The associated functional principal component scores, however, reveal that there is a non negligible correlation between almost all score pairs of the two elements. The separate FPCA thus captures joint variation between hip and knee angles only indirectly, which makes the interpretation of the FPCA results difficult. Correlated scores can also lead to multicollinearity issues in a subsequent regression analysis (functional principal component regression, e.g. Müller and Stadtmüller, 2005). Multivariate FPCA, by contrast, directly adresses potential covariation between the hip and knee elements. The first three bivariate principal components shown in Fig. 3.1, which explain 85.3% of the variability in the data, give insight into the main modes of joint variation in the overall gait movement. The corresponding scores do not only allow a more parsimonious representation of the data (one score value per bivariate principal component and per observation), but they are also uncorrelated by construction. Finally, the multivariate functional principal components are more natural to represent multivariate functional data in the sense that they have the same structure as each observation. The extension of FPCA to multivariate functional data is hence of high practical relevance.

Existing approaches for multivariate functional principal component analysis (MFPCA) are restricted to functions observed on the same finite, one-dimensional interval (Ramsay and Silverman, 2005; Jacques and Preda, 2014; Chiou, Yang, et al., 2014; Berrendero et al., 2011). Except for Berrendero et al. (2011), they all aim at a multivariate functional Karhunen-Loève representation of the data. For data measured e.g. in different units, Jacques and Preda (2014) and Chiou, Yang, et al. (2014) also discuss normalized versions of MFPCA based on a normalized covariance operator.

The key motivation for this paper is that in practical applications, multivariate functional data are neither restricted to lie on the same interval nor to have one-dimensional domains,

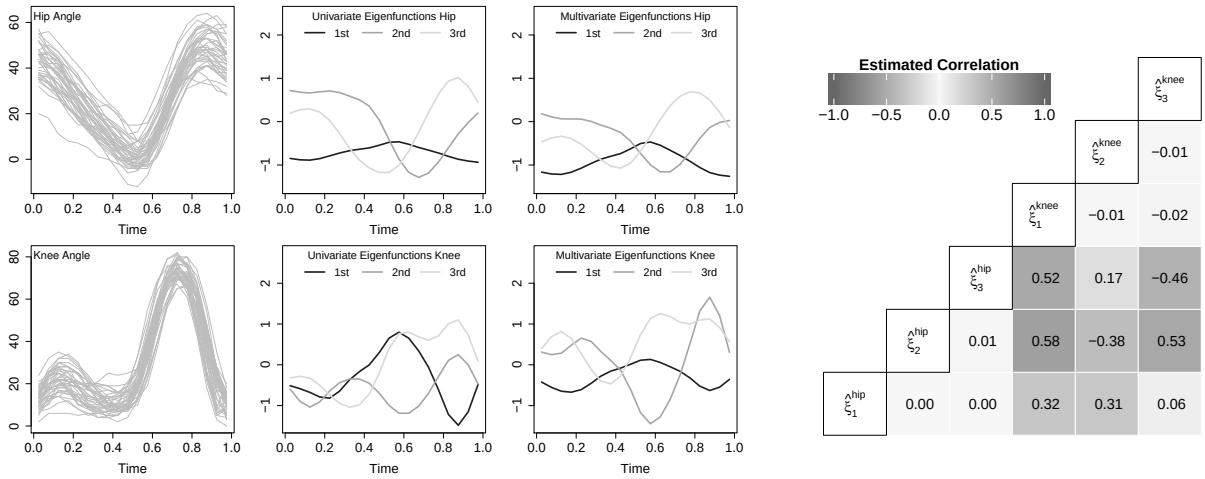


Figure 3.1.: Univariate and multivariate FPCA for the gait cycle data. 1st column: Original data. 2nd column: Results for univariate FPCA, calculated separately. The functions have been reflected, if necessary, and rescaled to have the same norm as the multivariate eigenfunctions for comparison purposes. 3rd column: Results for multivariate FPCA, calculated with the new approach. 4th column: Empirical correlation of the univariate FPCA scores for hip and knee.

e.g. data that consists of functions and images, as in our neuroimaging application. We start by extending the notion of multivariate functional data to the case of different (dimensional) domains for the different elements. Next, the theoretical foundations of MFPCA are provided in terms of a Karhunen-Loève Theorem. For the practically relevant case of a finite or truncated Karhunen-Loève representation, we establish a direct theoretical relationship between univariate and multivariate FPCA. This suggests a simple estimation strategy for multivariate functional principal components and scores based on their univariate counterparts. For data on higher dimensional domains (tensor data, e.g. images), principal component methods have originally been developed in the context of psychometrics (e.g. Tucker, 1966; Carroll and Chang, 1970) and have become particularly important in the machine learning literature (Coppi and Bolasco, 1989; Lu et al., 2013). Recent approaches for functional or smooth principal component analysis for tensor data have been proposed e.g. in Allen (2013). All these methods can be used as univariate building blocks for MFPCA. The resulting estimators for MFPCA are shown to be consistent under a given set of assumptions. In contrast to most of the existing methods for MFPCA, our new approach can be applied to sparse functional data and data with measurement error. It can be generalized to data available in arbitrary basis expansions and hence includes the MFPCA procedure proposed by Jacques and Preda (2014) as a special case. The new method further allows to incorporate weights for the elements, if they differ in domain, range or variation.

The paper is organized as follows. Section 3.2 introduces multivariate functional data and gives the theoretical basis for MFPCA. In Section 3.3 we derive the estimation algorithm

for MFPCA based on univariate basis expansions and investigate asymptotic properties of the resulting estimators. The performance of the new method is evaluated in Section 3.4 in a simulation with different levels of complexity. Section 3.5 contains the analysis of the motivating neuroimaging dataset. The paper concludes with a discussion and an outlook in Section 3.6. Supplementary material, containing detailed proofs of all propositions, more simulation results and `R` code is available online.

3.2. Theoretical Foundations of Multivariate Functional Data

3.2.1. Data Structure and Notation

This paper is concerned with *multivariate functional data*, i.e. each observation consists of $p \geq 2$ functions $X^{(1)}, \ldots, X^{(p)}$. They may be defined on different domains $\mathcal{T}_1, \ldots, \mathcal{T}_p$ with possibly different dimensions. Technically, $\mathcal{T}_j$ must be compact sets in $\mathbb{R}^{d_j}$, $d_j \in \mathbb{N}$ with finite (Lebesgue-) measure and each element $X^{(j)} \colon \mathcal{T}_j \to \mathbb{R}$ is assumed to be in $L^2(\mathcal{T}_j)$.

In analogy to other approaches for multivariate functional data, the different functions are combined in a vector X with

$$X(\boldsymbol{t}) = \left(X^{(1)}(t_1), \ldots, X^{(p)}(t_p)\right) \in \mathbb{R}^p.$$

Note that $\boldsymbol{t} := (t_1, \ldots, t_p) \in \mathcal{T} := \mathcal{T}_1 \times \cdots \times \mathcal{T}_p$ is a p-tuple of $d_1, \ldots, d_p$-dimensional vectors and not a scalar. This is a main difference to earlier approaches, as it allows each element $X^{(j)}$ to have a different argument t_j, even in the case of a common one-dimensional domain. In the following, it will be further assumed that

$$\mu(\boldsymbol{t}) := \mathbb{E}\left(X(\boldsymbol{t})\right) = \left(\mathbb{E}\left(X^{(1)}(t_1)\right), \ldots, \mathbb{E}\left(X^{(p)}(t_p)\right)\right) = \boldsymbol{0} \quad \forall\, \boldsymbol{t} \in \mathcal{T}.$$

For $\boldsymbol{s}, \boldsymbol{t} \in \mathcal{T}$, define the matrix of covariances $C(\boldsymbol{s}, \boldsymbol{t}) := \mathbb{E}\left(X(\boldsymbol{s}) \otimes X(\boldsymbol{t})\right)$ with elements

$$C_{ij}(s_i, t_j) := \mathbb{E}\left(X^{(i)}(s_i)X^{(j)}(t_j)\right) = \mathrm{Cov}(X^{(i)}(s_i), X^{(j)}(t_j)), \quad s_i \in \mathcal{T}_i,\ t_j \in \mathcal{T}_j. \tag{3.1}$$

As noted in Ramsay and Silverman (2005, Chapter 8.5.), a suitable inner product is the basis of all approaches for principal component analysis. For functions $f = (f^{(1)}, \ldots, f^{(p)})$ with elements $f^{(j)} \in L^2(\mathcal{T}_j)$ define the space $\mathcal{H} := L^2(\mathcal{T}_1) \times \ldots \times L^2(\mathcal{T}_p)$ and

$$\langle\langle f, g \rangle\rangle := \sum_{j=1}^{p} \langle f^{(j)}, g^{(j)} \rangle_2 = \sum_{j=1}^{p} \int_{\mathcal{T}_j} f^{(j)}(t_j) g^{(j)}(t_j) \mathrm{d}t_j, \qquad f, g \in \mathcal{H}. \tag{3.2}$$

Proposition 1. *$\mathcal{H}$ is a Hilbert space with respect to the scalar product $\langle\langle \cdot, \cdot \rangle\rangle$.*

Proofs for all propositions are given in the online appendix. The norm induced by $\langle\langle\cdot,\cdot\rangle\rangle$ is denoted by $|||\cdot|||^*$. Next, define the covariance operator $\Gamma\colon \mathcal{H} \to \mathcal{H}$ with the j-th element of $\Gamma f,\ f \in \mathcal{H}$ given by

$$(\Gamma f)^{(j)}(t_j) := \sum_{i=1}^{p} \int_{\mathcal{T}_i} C_{ij}(s_i, t_j) f^{(i)}(s_i) \mathrm{d}s_i = \langle\langle C_{\cdot j}(\cdot, t_j), f\rangle\rangle, \quad t_j \in \mathcal{T}_j. \tag{3.3}$$

The setting can be generalized to a weighted scalar product on $\mathcal{H}$, i.e.

$$\langle\langle f, g\rangle\rangle_w := \sum_{j=1}^{p} w_j \langle f^{(j)}, g^{(j)}\rangle_2, \qquad f, g \in \mathcal{H} \tag{3.4}$$

for some positive weights $w_1, \ldots, w_p$, cf. Ramsay and Silverman (2005, Chapter 10.3. in the context of hybrid data) or Chiou, Yang, et al. (2014). The associated weighted covariance operator Γ_w is given by its elements $(\Gamma_w f)^{(j)}$ with $f \in \mathcal{H}$ and

$$(\Gamma_w f)^{(j)}(t_j) = \langle\langle C_{\cdot,j}(\cdot, t_j), f\rangle\rangle_w, \quad t_j \in \mathcal{T}_j.$$

The use of weights may be necessary if the elements have quite different domains or ranges or if they exhibit different amounts of variation, in order to obtain multivariate functional principal components that have a meaningful interpretation (Chiou, Yang, et al., 2014). A weighted scalar product corresponds to a (global) rescaling of the elements by $w_j^{1/2}$. An alternative approach would be pointwise rescaling, e.g. by the inverse of the square root of the pointwise variance $C_{jj}(t_j, t_j)$. This can be seen as normalizing the covariance operator (Chiou, Yang, et al., 2014; Jacques and Preda, 2014). However, this second approach does not consider the size of the different domains $\mathcal{T}_j$ and would give equal variation per observation point t_j rather than per element j. Moreover, rescaling with the pointwise variance would downweight areas in $\mathcal{T}_j$ with stronger variation, hence areas that might contribute relevant information to the functional principal components. Therefore, only global rescaling by means of a weighted scalar product is considered in the following. The weights have to be chosen prior to the analysis. They can be specified based on expert knowledge or estimated from the data, e.g. based on the variation in each element (see references in Chiou, Yang, et al., 2014). A sensible choice will always depend on the specific application and the question of interest. One possible solution that is analogous to standardization in multivariate PCA is proposed in the application in Section 3.5. For the sake of better readability, all following theoretical results are derived for $w_1 = \ldots = w_p = 1$, but remain valid in the more general case of different weights. For the estimation algorithm discussed in Section 3.3.2, MFPCA based on the weighted scalar product is addressed again.

*The L^2-norm induced by $\langle\cdot,\cdot\rangle_2$ on each $L^2(\mathcal{T}_j)$ is denoted by $\|\cdot\|_2$. Further, $\|\cdot\|$ is the Euclidean norm for vectors and $\|\cdot\|_{\mathcal{T}}$ denotes a norm on $\mathcal{T}$ with $\|\boldsymbol{t}\|_{\mathcal{T}}^2 = \sum_{j=1}^{p} \|t_j\|^2$ for $t_j \in \mathcal{T}_j \subset \mathbb{R}^{d_j},\ j = 1, \ldots, p$.

3.2.2. A Karhunen-Loève Theorem for Multivariate Functional Data

In the following it is shown that under mild conditions, Γ has the same properties as the covariance operator in the univariate case and therefore a Karhunen-Loève representation for multivariate functional data exists. The main difference to existing approaches for data with elements observed on the same (one-dimensional) domain is that in this special case, Γ is an integral operator with positive definite kernel $C(s,t)$. This directly gives all of the desired properties (Saporta, 1981). In the more general case of elements observed on different domains, this is not obviously the case and the properties are shown explicitly.

Proposition 2. *The covariance operator Γ defined in (3.3) is a linear, self-adjoint and positive operator. If further for all $i, j = 1, \ldots, p$ there exist $K_{ij} < \infty$ with*

$$\|C_{ij}(\cdot, t_j)\|_2^{\,2} = \int_{\mathcal{T}_i} C_{ij}(s_i, t_j)^2 \mathrm{d}s_i < K_{ij} \quad \forall\ t_j \in \mathcal{T}_j, \tag{3.5}$$

and C_{ij} is uniformly continuous in the sense that

$$\forall\ \varepsilon > 0\ \exists\ \delta_{ij} > 0 : \quad \left\|t_j - t_j^*\right\| < \delta_{ij} \quad \Rightarrow \quad \left|C_{ij}(s_i, t_j) - C_{ij}(s_i, t_j^*)\right| < \varepsilon \quad \forall\ s_i \in \mathcal{T}_i,$$

then Γ is a compact operator.

In the remainder of this paper, it is assumed that the C_{ij} satisfy all conditions of Prop. 2 and hence Γ can always be assumed to be a compact positive operator on $\mathcal{H}$. By the Hilbert-Schmidt Theorem (e.g. Reed and Simon, 1980, Thm. VI.16) it follows that there exists a complete orthonormal basis of eigenfunctions $\psi_m \in \mathcal{H},\ m \in \mathbb{N}$ of Γ such that

$$\Gamma\psi_m = \nu_m\psi_m \quad \text{and} \quad \nu_m \to 0 \quad \text{for } m \to \infty.$$

In particular, since Γ is a positive operator, it may be assumed w.l.o.g. that $\nu_1 \geq \nu_2 \geq \ldots \geq 0$. Since $\psi_m,\ m \in \mathbb{N}$ is an orthonormal basis of $\mathcal{H}$ and Γ is self-adjoint, by the Spectral Theorem (e.g. Werner, 2011, Thm. VI.3.2.) it holds that

$$\Gamma f = \sum\nolimits_{m=1}^{\infty} \nu_m \langle\langle f, \psi_m \rangle\rangle \psi_m \quad \forall\ f \in \mathcal{H}.$$

The following proposition is a multivariate version of Mercer's Theorem (Mercer, 1909). It plays a key role in the proof of the Karhunen-Loève Theorem (Prop. 4).

Proposition 3 (Mercer's Theorem)**.** *For $j = 1, \ldots, p$ and $s_j, t_j \in \mathcal{T}_j$ it holds that*

$$\mathrm{Cov}\left(X^{(j)}(s_j), X^{(j)}(t_j)\right) = C_{jj}(s_j, t_j) = \sum\nolimits_{m=1}^{\infty} \nu_m \psi_m^{(j)}(s_j)\psi_m^{(j)}(t_j),$$

where the convergence is absolute and uniform.

Proposition 4 (Multivariate Karhunen-Loève Theorem)**.** *Under the assumptions of Prop. 2,*

$$X(\boldsymbol{t}) = \sum_{m=1}^{\infty} \rho_m \psi_m(\boldsymbol{t}), \quad \boldsymbol{t} \in \mathcal{T}, \tag{3.6}$$

with zero mean random variables $\rho_m = \langle\langle X, \psi_m \rangle\rangle$ *and* $\mathrm{Cov}(\rho_m, \rho_n) = \nu_m \delta_{mn}$*. Moreover*

$$\mathbb{E}\left(\left\| X(\boldsymbol{t}) - \sum_{m=1}^{M} \rho_m \psi_m(\boldsymbol{t}) \right\|^2 \right) \to 0 \quad \text{for } M \to \infty$$

uniformly for $\boldsymbol{t} \in \mathcal{T}$*.*

The multivariate Karhunen-Loève representation has an analogous interpretation as in the univariate case (Ramsay and Silverman, 2005, Chapter 8.2.). The eigenvalues ν_m represent the amount of variability in X explained by the single *multivariate functional principal components* ψ_m, while the *multivariate functional principal component scores* ρ_m serve as weights of ψ_m in the Karhunen-Loève representation of X. As the eigenvalues ν_m decrease towards 0, leading eigenfunctions reflect the most important features of X. Truncated Karhunen-Loève expansions, optimal M-dimensional approximations to X,

$$X_{\lceil M \rceil}(\boldsymbol{t}) := \sum_{m=1}^{M} \rho_m \psi_m(\boldsymbol{t}), \quad \boldsymbol{t} \in \mathcal{T}, \tag{3.7}$$

are often used in practice. Single observations x_i of X can then be characterized by their score vectors $(\rho_{i,1}, \ldots, \rho_{i,M})$ with $\rho_{i,m} = \langle\langle x_i, \psi_m \rangle\rangle$ for further analysis, e.g. for regression (Müller and Stadtmüller, 2005) or clustering (Jacques and Preda, 2014).

3.3. Multivariate FPCA

3.3.1. Relationship Between Univariate and Multivariate FPCA for Finite Karhunen-Loève Decompositions

Given the Karhunen-Loève representation of multivariate functional data X as in (3.6), a natural question is how this representation relates to the univariate Karhunen-Loève representations of the single elements $X^{(j)}$. The following proposition establishes a direct relationship between these two representations if they are both finite, based on the theory of integral equations (Zemyan, 2012).

Proposition 5. *The multivariate functional vector* $X = \left(X^{(1)}, \ldots, X^{(p)}\right)$ *in* (3.6) *has a finite Karhunen-Loève representation if and only if all univariate elements* $X^{(1)}, \ldots, X^{(p)}$*, have a finite Karhunen-Loève representation. In this case, it holds:*

1. *Given the multivariate Karhunen-Loève representation* (3.6)*, the positive eigenvalues* $\lambda_1^{(j)} \geq \ldots \geq \lambda_{M_j}^{(j)} > 0$*,* $M_j \leq M$ *of the univariate covariance operator* $\Gamma^{(j)}$ *associated with* $X^{(j)}$ *correspond to the positive eigenvalues of the matrix* $\boldsymbol{A^{(j)}} \in \mathbb{R}^{M\times M}$ *with entries*

$$A_{mn}^{(j)} = (\nu_m \nu_n)^{1/2} \langle \psi_m^{(j)}, \psi_n^{(j)} \rangle_2, \quad m, n = 1, \ldots, M.$$

The eigenfunctions of $\Gamma^{(j)}$ *are given by*

$$\phi_m^{(j)}(t_j) = \left(\lambda_m^{(j)}\right)^{-1/2} \sum_{n=1}^{M} \nu_n^{1/2} [\boldsymbol{u_m^{(j)}}]_n \psi_n^{(j)}(t_j), \quad t_j \in \mathcal{T}_j,\ m = 1, \ldots, M_j,$$

where $\boldsymbol{u_m^{(j)}}$ *denotes an (orthonormal) eigenvector of* $\boldsymbol{A^{(j)}}$ *associated with eigenvalue* $\lambda_m^{(j)}$ *and* $[\boldsymbol{u_m^{(j)}}]_n$ *denotes the n-th entry of this vector. For the univariate scores*

$$\xi_m^{(j)} = \langle X^{(j)}, \phi_m^{(j)} \rangle_2 = \left(\lambda_m^{(j)}\right)^{-1/2} \sum_{n=1}^{M} \nu_n^{1/2} \left[\boldsymbol{u_m^{(j)}}\right]_n \sum_{k=1}^{M} \rho_k \langle \psi_n^{(j)}, \psi_k^{(j)} \rangle_2.$$

2. *Assuming the univariate Karhunen-Loève representation* $X^{(j)} = \sum_{m=1}^{M_j} \xi_m^{(j)} \phi_m^{(j)}$ *with* $\Gamma^{(j)} \phi_m^{(j)} = \lambda_m^{(j)} \phi_m^{(j)}$ *for each element* $X^{(j)}$ *of* X*, the positive eigenvalues* $\nu_1 \geq \ldots \geq \nu_M > 0$ *of* Γ *with* $M \leq \sum_{j=1}^{p} M_j =: M_+$ *correspond to the positive eigenvalues of the matrix* $\boldsymbol{Z} \in \mathbb{R}^{M_+ \times M_+}$ *consisting of blocks* $\boldsymbol{Z^{(jk)}} \in \mathbb{R}^{M_j \times M_k}$ *with entries*

$$Z_{mn}^{(jk)} = \mathrm{Cov}\left(\xi_m^{(j)}, \xi_n^{(k)}\right), \quad m = 1, \ldots, M_j,\ n = 1, \ldots, M_k,\ j, k = 1, \ldots, p.$$

The eigenfunctions of Γ *are given by their elements*

$$\psi_m^{(j)}(t_j) = \sum_{n=1}^{M_j} [\boldsymbol{c_m}]_n^{(j)} \phi_n^{(j)}(t_j), \quad t_j \in \mathcal{T}_j,\ m = 1, \ldots, M,$$

where $[\boldsymbol{c_m}]^{(j)} \in \mathbb{R}^{M_j}$ *denotes the j-th block of an (orthonormal) eigenvector* $\boldsymbol{c_m}$ *of* $\boldsymbol{Z}$ *associated with eigenvalue* ν_m*. The scores are given by*

$$\rho_m = \sum_{j=1}^{p} \sum_{n=1}^{M_j} [\boldsymbol{c_m}]_n^{(j)} \xi_n^{(j)}.$$

Extensions: The second part of Prop. 5 can be extended in a natural way if univariate elements are expanded in finitely many, not necessarily orthonormal basis functions $b_m^{(j)}$ with coefficients $\theta_m^{(j)}$, i.e.

$$X^{(j)}(t_j) = \sum_{m=1}^{K_j} \theta_m^{(j)} b_m^{(j)}(t_j), \quad t_j \in \mathcal{T}_j. \tag{3.8}$$

This is a very likely situation in practice, e.g. due to presmoothing of noisy observations. Following analogous steps as in the proof of Prop. 5 results in an eigenanalysis problem $\boldsymbol{BQc} = \nu \boldsymbol{c}$ as starting point for the MFPCA. Here $\boldsymbol{B} \in \mathbb{R}^{K_+ \times K_+}$ with $K_+ = \sum_{j=1}^{p} K_j$ is a block diagonal matrix of scalar products $\langle b_m^{(j)}, b_n^{(j)} \rangle_2$ of univariate basis functions associated with each element $X^{(j)}$. In the special case that all univariate bases are orthonormal (e.g. when using the univariate principal component bases as in Prop. 5), $\boldsymbol{B}$ equals the identity

matrix. The symmetric block matrix $\boldsymbol{Q}$ with entries $Q_{mn}^{(jk)} = \text{Cov}(\theta_m^{(j)}, \theta_n^{(k)})$ corresponds to $\boldsymbol{Z}$ in Prop. 5. Although $\boldsymbol{BQ}$ is in general not symmetric, its eigenvectors $\boldsymbol{c}_m$ and eigenvalues ν_m, which are at the same time the eigenvalues of Γ, are real. This can be easily shown using the Cholesky decomposition of the symmetric matrix $\boldsymbol{B} = \boldsymbol{R}\boldsymbol{R}^\top$ and solving $\boldsymbol{R}^\top\boldsymbol{Q}\boldsymbol{R}\tilde{\boldsymbol{c}} = \nu\tilde{\boldsymbol{c}}$ with $\tilde{\boldsymbol{c}} = \boldsymbol{R}^{-1}\boldsymbol{c}$. The estimation algorithm for principal components ψ_m and associated scores ρ_m based on this general basis expansion is presented in the next section combined with the case of a weighted scalar product.

3.3.2. Estimation of Multivariate FPCA

Estimation based on univariate FPCA: The second part of Prop. 5 suggests a simple and natural approach for estimating the MFPCA. After calculation of univariate FPCAs for each element, the estimates can be plugged into the formulae given in Prop. 5. Given demeaned samples $x_1, \ldots, x_N$ of X, the proposed estimation procedure for MFPCA consists of four steps:

1. For each element $X^{(j)}$ estimate a univariate FPCA based on the observations $x_1^{(j)}, \ldots, x_N^{(j)}$. This results in estimated eigenfunctions $\hat{\phi}_m^{(j)}$ and scores $\hat{\xi}_{i,m}^{(j)},\ i = 1, \ldots, N,\ m = 1, \ldots, M_j$ for suitably chosen truncation lags M_j. As there exist numerous estimation procedures, e.g. for irregularly sampled and sparse data with measurement error (Yao et al., 2005), the multivariate method is also applicable to this kind of data.

2. Define the matrix $\boldsymbol{\Xi} \in \mathbb{R}^{N \times M_+}$, where each row $(\hat{\xi}_{i,1}^{(1)}, \ldots, \hat{\xi}_{i,M_1}^{(1)}, \ldots, \hat{\xi}_{i,1}^{(p)}, \ldots, \hat{\xi}_{i,M_p}^{(p)})$ contains all estimated scores for a single observation. An estimate $\hat{\boldsymbol{Z}} \in \mathbb{R}^{M_+ \times M_+}$ of the block matrix $\boldsymbol{Z}$ in Prop. 5 is given by $\hat{\boldsymbol{Z}} = (N-1)^{-1}\boldsymbol{\Xi}^\top\boldsymbol{\Xi}$.

3. Perform a matrix eigenanalysis for $\hat{\boldsymbol{Z}}$ resulting in eigenvalues $\hat{\nu}_m$ and orthonormal eigenvectors $\hat{\boldsymbol{c}}_{\boldsymbol{m}}$.

4. Estimates for the multivariate eigenfunctions are given by their elements

$$\hat{\psi}_m^{(j)}(t_j) = \sum_{n=1}^{M_j} [\hat{\boldsymbol{c}}_{\boldsymbol{m}}]_n^{(j)} \hat{\phi}_n^{(j)}(t_j), \quad t_j \in \mathcal{T}_j,\ m = 1, \ldots, M^+ \tag{3.9}$$

   and multivariate scores can be calculated via

$$\hat{\rho}_{i,m} = \sum_{j=1}^{p} \sum_{n=1}^{M_j} [\hat{\boldsymbol{c}}_{\boldsymbol{m}}]_n^{(j)} \hat{\xi}_{i,n}^{(j)} = \boldsymbol{\Xi}_{\boldsymbol{i},\cdot}\hat{\boldsymbol{c}}_{\boldsymbol{m}}. \tag{3.10}$$

Finding an appropriate truncation lag M_j in step 1 is a well-known issue in functional data analysis. Common approaches are based on the decrease of the estimated eigenvalues $\hat{\lambda}_m^{(j)}$ (scree-plot, Cattell, 1966) or the percentage of variance explained (e.g. Ramsay and Silverman, 2005, Chapter 8.2.). An optimal number $M \leq M_+$ of multivariate functional principal components can basically be chosen with the same techniques, while the importance of a

"correct" choice depends on the specific application: For simply exploratory aims it is less crucial than for subsequent analyses that ignore the information of the eigenvalues (and hence, the proportion of variance explained by the single components) and are based solely on multivariate eigenfunctions or scores, as e.g. clustering or functional principal component regression. For the latter, relevant eigenfunctions can also be selected using model-based approaches such as AIC or cross-validation. The goodness of the resulting MFPCA estimates of course depends on an appropriate choice of M_j, which can also be used as a sensitivity check: If the first M_j eigenfunctions capture all the relevant information in $X^{(j)}$, increasing M_j will add only little information and hence should have only little impact on the results. This relationship is analyzed in a simulation in the online appendix.

Extensions: The estimation algorithm can easily be extended to elements $X^{(j)}$ available in general basis expansions as in (3.8) and to MFPCA based on a weighted scalar product as in (3.4). Given weights $w_1, \ldots, w_p > 0$ and demeaned observations $x_1, \ldots, x_N$ of X with estimated basis function coefficients $\hat{\theta}_{i,m}^{(j)}$ for each element, the eigenanalysis problem to solve is

$$(N-1)^{-1}\boldsymbol{B}\boldsymbol{D}\boldsymbol{\Theta}^\top\boldsymbol{\Theta}\boldsymbol{D}\boldsymbol{c} = \nu\boldsymbol{c}. \tag{3.11}$$

The matrix $\boldsymbol{B}$ is the block diagonal matrix of basis scalar products as in Section 3.3.1 and $\boldsymbol{D} = \text{diag}(\boldsymbol{w_1}^{1/2}, \ldots, \boldsymbol{w_p}^{1/2}) \in \mathbb{R}^{K_+ \times K_+}$ accounts for the weights, where each $w_j^{1/2}$ is repeated K_j times to give $\boldsymbol{w_j}^{1/2}$. $\boldsymbol{\Theta} \in \mathbb{R}^{N \times K_+}$ with rows $(\hat{\theta}_{i,1}^{(1)}, \ldots, \hat{\theta}_{i,K_1}^{(1)}, \ldots, \hat{\theta}_{i,1}^{(p)}, \ldots, \hat{\theta}_{i,K_p}^{(p)})$ corresponds to the matrix $\boldsymbol{\Xi}$ defined in step 2 of the original algorithm and $(N-1)^{-1}\boldsymbol{\Theta}^\top\boldsymbol{\Theta}$ is an estimate for $\boldsymbol{Q}$ introduced in Section 3.3.1. Given eigenvectors $\hat{\boldsymbol{c}}_{\boldsymbol{m}}$ and eigenvalues $\hat{\nu}_m$ for (3.11), estimated orthonormal eigenfunctions $\hat{\psi}_m$ of Γ_w and associated scores $\hat{\rho}_{i,m}$ can be calculated in analogy to (3.9) and (3.10) with $\hat{\boldsymbol{Q}}_{\boldsymbol{w}} = (N-1)^{-1}\boldsymbol{D}\boldsymbol{\Theta}^\top\boldsymbol{\Theta}\boldsymbol{D}$:

$$\begin{aligned}
\hat{\psi}_m^{(j)}(t_j) &= \left(w_j \cdot \hat{\nu}_m \hat{\boldsymbol{c}}_{\boldsymbol{m}}^\top \hat{\boldsymbol{Q}}_{\boldsymbol{w}} \hat{\boldsymbol{c}}_{\boldsymbol{m}}\right)^{-1/2} \sum_{k=1}^{p} \sum_{l=1}^{K_j} \sum_{n=1}^{K_k} [\hat{\boldsymbol{Q}}_{\boldsymbol{w}}]_{ln}^{(jk)} [\hat{\boldsymbol{c}}_{\boldsymbol{m}}]_n^{(k)} b_l^{(j)}(t_j), \\
\hat{\rho}_{i,m} &= (\hat{\nu}_m)^{1/2} \left(\hat{\boldsymbol{c}}_{\boldsymbol{m}}^\top \hat{\boldsymbol{Q}}_{\boldsymbol{w}} \hat{\boldsymbol{c}}_{\boldsymbol{m}}\right)^{-1/2} \boldsymbol{\Theta}_{\boldsymbol{i},\cdot} \boldsymbol{D} \hat{\boldsymbol{c}}_{\boldsymbol{m}}.
\end{aligned}$$

Clearly, the original algorithm is obtained as a special case with $\boldsymbol{\Theta} = \boldsymbol{\Xi}$, $\boldsymbol{B} = \boldsymbol{I}$ (univariate FPCA for each element) and $\boldsymbol{D} = \boldsymbol{I}$ (all weights equal to 1). Moreover, the extended algorithm allows to flexibly combine univariate FPCA and general basis expansions for different elements of the multivariate functional data.

If all elements $X^{(j)}$ are defined on the same (one-dimensional) interval and $\boldsymbol{D} = \boldsymbol{I}$, expanding each element in a general basis is equivalent to the method of Jacques and Preda (2014). The approach proposed in this paper, however, is more general, as it allows for different intervals as well as for higher dimensional $\mathcal{T}_j$ and thus basis functions $b_m^{(j)}$.

Implementation: All presented variations of the MFPCA estimation algorithm are implemented in an `R` package `MFPCA` (Happ, 2017b). Univariate basis expansions include univariate FPCA (1D), smooth tensor PCA (2D), spline bases (1D/2D) and cosine bases

(2D/3D). New bases can be added easily and in a modular way. The `MFPCA` package is based on the package `funData` (Happ, 2017a) for representing (multivariate) functional data on potentially different dimensional domains.

3.3.3. Asymptotic Properties

The results of Prop. 5 and the estimators proposed in the previous section have been derived under the assumption of a finite sample size N and a finite Karhunen-Loève representation for each element $X^{(j)}$. This case is relevant in practice, since data is observable only in finite form (finitely many observations, finite resolution) and hence contains only finite information. In this case, the maximal number of principal components which can be estimated is limited to the number of observations N. For a growing number of observations, the truncation limits M_j and thus M_+ may increase with N. All asympotic examinations hence have to consider the approximation error caused by truncating the univariate Karhunen-Loève representations to finite sums as well as the estimation error. For the eigenfunctions (analogously for the eigenvalues and scores) one hence has the following decomposition:

$$\left|\left|\left|\psi_m - \hat{\psi}_m\right|\right|\right| \leq \left|\left|\left|\psi_m - \psi_m^{[M]}\right|\right|\right| + \left|\left|\left|\psi_m^{[M]} - \hat{\psi}_m\right|\right|\right|.$$

Here ψ_m is the true m-th eigenfunction of the covariance operator Γ and $\hat{\psi}_m$ is the estimator based on the assumption of a finite Karhunen-Loève representation in each element. This assumption is reflected in $\psi_m^{[M]}$, which denotes the m-th eigenfunction of the covariance operator $\Gamma^{[M]}$ associated with $X^{[M]}$ with elements equal to the truncated $X^{(j)}$. These are really the eigenfunctions targeted with the estimation algorithm presented in Section 3.3.2. The first term on the right hand side of the inequality can be seen as a bias term caused by truncation. It depends on N only implicitly via $M_1, \ldots, M_p$. The second term accounts for the estimation error, thus can be interpreted as a variance term.

Proposition 6 (Approximation Error)**.** *Let $\nu_m^{[M]}$, $m \in \mathbb{N}$ be the eigenvalues of the covariance operator $\Gamma^{[M]}$ associated with $X^{[M]}$ having truncated univariate elements $X^{[M](j)} = \sum_{m=1}^{M_j} \xi_m^{(j)} \phi_m^{(j)}$. Then the approximation error $\left|\left|\left|X^{[M]} - X\right|\right|\right|$ converges to 0 in probability for $M_1, \ldots, M_p \to \infty$. For each $m \in \mathbb{N}$, $\nu_m^{[M]}$ converges to ν_m including multiplicity and the total projection $P_m^{[M]}$ of $\mathcal{H}$ onto the eigenspace of $\Gamma^{[M]}$ associated with $\nu_m^{[M]}$ converges in norm to the total projection P_m of $\mathcal{H}$ onto the eigenspace of Γ associated with ν_m.*

In particular, if ν_m and $\nu_m^{[M]}$ both have multiplicity 1 with associated eigenfunctions ψ_m and $\psi_m^{[M]}$, such that $\langle\langle \psi_m, \psi_m^{[M]} \rangle\rangle \geq 0$, then

$$\left|\left|\left|\psi_m^{[M]} - \psi_m\right|\right|\right| \to 0 \quad \text{for } M_1, \ldots, M_p \to \infty.$$

The scores $\rho_m^{[M]} := \langle\langle X^{[M]}, \psi_m^{[M]} \rangle\rangle$ converge in probability to ρ_m for all $m \in \mathbb{N}$.

In the remainder of this section, all non-zero eigenvalues ν_m are assumed to have multiplicity 1, as then the eigenfunctions $\psi_m^{[M]}$ converge to ψ_m, if their orientation is chosen such that $\langle\langle \psi_m, \psi_m^{[M]} \rangle\rangle \geq 0$.

For the estimation error, consider the univariate elements $X^{(j)}$ of X with covariance operator $\Gamma^{(j)}$ and associated eigenvalues $\lambda_m^{(j)}$ and eigenfunctions $\phi_m^{(j)}$, $m = 1, \ldots, M_j$. In the following, let $X_1, \ldots, X_N$ be independent copies of X and assume for all $j = 1, \ldots, p$

$$\Delta_{M_j}^{(j)} := \sup_{m=1,\ldots,M_j} (\lambda_m^{(j)} - \lambda_{m+1}^{(j)})^{-1} < \infty \text{ for every finite } M_j \tag{A1}$$

$$\int_{\mathcal{T}_j} \int_{\mathcal{T}_k} \mathbb{E}\left(X^{(j)}(t_j)^2 X^{(k)}(s_k)^2\right) \mathrm{d}s_k \, \mathrm{d}t_j < \infty \quad \forall\, k = 1, \ldots, p \tag{A2}$$

$$\left\| \Gamma^{(j)} - \hat{\Gamma}^{(j)} \right\|_{\mathrm{op}} = O_p(r_N^{\Gamma}) \tag{A3}$$

$$\langle \phi_m^{(j)}, \hat{\phi}_m^{(j)} \rangle_2 \geq 0 \text{ for all } m = 1, \ldots, M_j \tag{A4}$$

$$\hat{\xi}_{i,m}^{(j)} = \langle X_i^{(j)}, \hat{\phi}_m^{(j)} \rangle_2 \text{ for all } m = 1, \ldots, M_j,\ i = 1, \ldots, N \tag{A5}$$

(A1) – (A2) concern theoretical properties of $X^{(j)}$ and $\Gamma^{(j)}$, while (A3) – (A5) depend on the univariate decompositions used. (A1) is a standard assumption in univariate FPCA (Bosq, 2000; Hall and Hosseini-Nasab, 2006). It guarantees that the first M_j univariate eigenvalues of each element all have multiplicity 1. With (A2), the integral operator with kernel $\hat{C}_{jk}(s,t) := N^{-1} \sum_{i=1}^N X_i^{(j)}(s) X_i^{(k)}(t)$ converges to the one with kernel $C_{jk}(s,t)$ with rate $N^{-1/2}$. (A2) is used in combination with (A5) to obtain a convergence rate for the maximal eigenvalue of $\boldsymbol{Z} - \hat{\boldsymbol{Z}}$, which, in turn, affects the convergence of the eigenvectors $\hat{\boldsymbol{c}}_{\boldsymbol{m}}$ to $\boldsymbol{c}_{\boldsymbol{m}}$ (Yu et al., 2015). (A3) ensures that the operator $\hat{\Gamma}^{(j)}$, which is the basis of the univariate FPCA, converges to $\Gamma^{(j)}$ in the operator norm $\|\cdot\|_{\mathrm{op}}$ induced by $\|\cdot\|_2$ with a given rate r_N^{Γ}. For fully observed data, Hall and Horowitz (2007) show $r_N^{\Gamma} = N^{-1/2}$, while the approach of Yao et al. (2005) yields $r_N^{\Gamma} = N^{-1/2} h^{-2}$ in the case of measurement error or irregularly sampled data for a certain bandwidth h. Together with (A1), r_N^{Γ} gives a convergence rate for the univariate eigenfunctions $\hat{\phi}_m^{(j)}$ (Bosq, 2000, Lemma 4.3). (A4) guarantees that $\hat{\phi}_m^{(j)}$ is an estimator for $\phi_m^{(j)}$ rather than for $-\phi_m^{(j)}$, as eigenfunctions are defined only up to a sign change (Bosq, 2000; Hall and Hosseini-Nasab, 2006). Finally, (A5) is used to formulate the convergence of the estimated scores in terms of convergence rates for the estimated eigenfunctions. If this assumption does not hold (e.g. in Yao et al., 2005), convergence results can still be obtained e.g. by assuming a convergence rate for $\hat{\xi}_{i,m}^{(j)}$ and replacing (A2) by an assumption on the rate of convergence for the maximal eigenvalue of $\boldsymbol{Z} - \hat{\boldsymbol{Z}}$.

Proposition 7 (Estimation Error)**.** *Assume* (A1) – (A5) *hold. Then for* $M_{\max} = \max_{j=1,\ldots,p} M_j$ *and* $\Delta_M := \max_{j=1,\ldots,p} \Delta_{M_j}^{(j)}$*, the maximal eigenvalue of* $\boldsymbol{Z} - \hat{\boldsymbol{Z}}$ *can be characterized by*

$$\lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}}) = O_p(M_{\max} \max(N^{-1/2}, \Delta_M r_N^{\Gamma})).$$

Using the same notation as in Prop. 6, it holds for all $m = 1, \ldots, M_+$ *that*

$$\begin{aligned}
\left|\nu_m^{[M]} - \hat{\nu}_m\right| &= O_p(M_{\max} \max(N^{-1/2}, \Delta_M r_N^\Gamma)), \\
\left\|\left|\psi_m^{[M]} - \hat{\psi}_m\right\|\right| &= O_p(M_{\max}^{3/2} \max(N^{-1/2}, \Delta_M r_N^\Gamma)), \\
\left|\rho_{i,m}^{[M]} - \hat{\rho}_{i,m}\right| &= O_p(M_{\max}^{3/2} \max(N^{-1/2}, \Delta_M r_N^\Gamma)), \\
\left\|\left|X_i^{[M]} - \hat{X}_i^{[M]}\right\|\right| &= O_p(M_{\max} \Delta_M r_N^\Gamma)
\end{aligned}$$

with $\hat{X}_i^{[M](j)} = \sum_{m=1}^{M_j} \hat{\xi}_{i,m}^{(j)} \hat{\phi}_m^{(j)}$.

When combining the results of Prop. 6 and Prop. 7, the analogy to bias and variance again becomes apparent: For fixed N, higher values of $M_1, \ldots, M_p$ will reduce the approximation error, but simultaneously increase the estimation error, as both $M_{\max}$ and Δ_M increase with M_j. If one assumes for example $M_1 = \ldots = M_p = M_{\max} = O(N^\beta)$, $r_N^\Gamma = N^{-1/2}$, and that the eigengaps fulfill $\lambda_m^{(j)} - \lambda_{m+1}^{(j)} \geq C^{-1} m^{-\alpha-1}$ with $\alpha > 1,\ C > 0$ (cf. Hall and Horowitz, 2007), the MFPCA estimators given in Section 3.3.2 are consistent for $0 < \beta < (2\alpha+5)^{-1}$.

3.4. Simulation

We illustrate the performance of our new MFPCA estimation procedure in three settings with increasing complexity:

1. Densely observed bivariate functional data on the same one-dimensional interval.
2. Trivariate functional data on different one-dimensional intervals with different levels of sparsity.
3. Bivariate functional data on different dimensional domains (images and functions).

The first two settings deal with multivariate functional data on one-dimensional domains and are presented together in Section 3.4.1. Setting 3 is discussed separately in Section 3.4.2. Examples for simulated data and estimation results for all three settings are given in the online appendix, which also includes two additional simulations (cf. Sections 3.3.2 and 3.5). Unless specified otherwise, the `MFPCA` package (Happ, 2017b, version 1.0-1) is used for all calculations.

Each setting is based on 100 datasets with $N = 250$ observations of the form

$$x_i(\boldsymbol{t}) = \sum_{m=1}^{M} \rho_{i,m} \psi_m(\boldsymbol{t}) + \boldsymbol{\varepsilon}_i(\boldsymbol{t}), \quad \boldsymbol{\varepsilon}_i(\boldsymbol{t}) \overset{\text{iid}}{\sim} \mathrm{N}_p(0, \sigma^2 \boldsymbol{I}), \quad \boldsymbol{t} \in \mathcal{T},\ i = 1, \ldots, N.$$

In each case, we consider data without ($\sigma^2 = 0$) and with ($\sigma^2 = 0.25$) measurement error. The scores $\rho_{i,m}$ are independent samples from $\mathrm{N}(0, \nu_m)$ for eigenvalues with exponential

($\nu_m^{\text{exp}} = \exp(-(m+1)/2)$) or linear ($\nu_m^{\text{lin}} = (M+1-m)/M$) decrease, while the choice of $\mathcal{T}, M$ and ψ_m varies between settings (see Sections 3.4.1 and 3.4.2). In all cases, we use unit weights ($w_j = 1$). The accuracy of the resulting estimates $\hat{\nu}_m$ and $\hat{\psi}_m$ is measured by the relative errors $\text{Err}(\hat{\nu}_m) = (\nu_m - \hat{\nu}_m)^2 / \nu_m^2$ and $\text{Err}(\hat{\psi}_m) = \left\| \psi_m - \hat{\psi}_m \right\|^2$. As functional principal components are defined only up to a sign change, the estimate $\hat{\psi}_m$ is reflected, i.e. multiplied by -1, if $\langle\langle \psi_m, \hat{\psi}_m \rangle\rangle < 0$. The goodness of the reconstructed observations $\hat{x}_i = \sum_{m=1}^{M} \hat{\rho}_{i,m} \hat{\psi}_m^{(j)}$ is evaluated by the mean relative squared error $\text{MRSE} = N^{-1} \sum_{i=1}^{N} \left(\|\|x_i - \hat{x}_i\|\|^2 / \|\|x_i\|\|^2 \right)$.

3.4.1. Multivariate Functional Data on One-Dimensional Domains

Setting 1: For the first setting, the first $M = 8$ Fourier basis functions on $[0, 2]$ are split into $p = 2$ parts. The pieces are shifted and multiplied by a random sign to form the elements $\psi_m^{(1)}$ and $\psi_m^{(2)}$ on $\mathcal{T}_1 = \mathcal{T}_2 = [0, 1]$ (for technical details, see online appendix). The observations x_i are sampled on an equispaced grid of $S_1 = S_2 = 100$ sampling points. The MFPCA is based on $M_1 = M_2 = 8$ univariate functional principal components that are calculated by the PACE algorithm (Yao et al., 2005) with penalized splines to smooth the covariance function, as implemented in the `R` package `refund` (Goldsmith, Scheipl, et al., 2016). In this simple setting of a common, one-dimensional domain, the new approach can be compared to the method of Ramsay and Silverman (2005), which is implemented in the `R` package `fda` (Ramsay, Wickham, et al., 2014) and in the following denoted by MFPCA_{RS}. This method involves presmoothing of the elements with $K = 15$ cubic spline basis functions. MFPCA_{RS} computes score values $\hat{\rho}_{i,m}^{(j)} = \langle x_i^{(j)}, \hat{\psi}_m^{(j)} \rangle_2$ for each observation i and each element j. Since they do not have the same interpretation as the scores in the multivariate Karhunen-Loève representation (Prop. 4), $\sum_{j=1}^{p} \hat{\rho}_{i,m}^{(j)} = \hat{\rho}_{i,m}$ is used for comparison purposes.

The results for the first setting are shown in Fig. 3.2 and Table 3.1. In total, the new approach can compete very well with the existing method of Ramsay and Silverman and gives nearly identical results for synthetic and real data (see online appendix for the gait cycle example). Both techniques mostly have higher errors in ψ_m for linearly decreasing eigenvalues, as in these cases, the eigenfunctions are more often confused, i.e. $\hat{\psi}_m$ is an estimate for e.g. ψ_{m-1} or ψ_{m+1} rather than for ψ_m. In the ideal case of no measurement error, MFPCA_{RS} yields lower MRSE values than the new approach, which might be an effect of MFPCA_{RS} expecting smooth or presmoothed data. For the practically relevant case of data with measurement error, both methods give almost the same prediction errors (cf. Table 3.1). Simulations based on Legendre polynomials gave very similar results (not shown here).

Setting 2: Here we consider trivariate functional data on $\mathcal{T}_1 = [-1, -0.5]$, $\mathcal{T}_2 = [0, 1]$, $\mathcal{T}_3 = [1.5, 2]$. The eigenfunctions are constructed according to the same scheme as in setting 1 by

Table 3.1.: Average MRSE (in %) for simulation settings 1 and 2, depending on eigenvalue decrease and measurement error.

		$\sigma^2 = 0$		$\sigma^2 = 0.25$	
Setting		ν_m^{exp}	ν_m^{lin}	ν_m^{exp}	ν_m^{lin}
1	MFPCA	0.006	0.009	0.740	0.355
1	MFPCA_{RS}	$< 10^{-3}$	$< 10^{-3}$	0.720	0.338
2	Full Data	0.004	0.007	0.778	0.367
2	Medium Sparsity	0.164	0.146	2.070	1.102
2	High Sparsity	5.755	4.568	15.365	10.824

splitting the first $M = 8$ Fourier basis functions on $[0, 2]$ into $p = 3$ parts, followed by a shift and multiplication with a random sign. The observations are sampled on equidistant grids with $S_1 = S_3 = 50$ and $S_2 = 100$ sampling points. We consider the dense observations as well as sparse variants with medium ($50-70\%$) and high ($90-95\%$ missings) sparsity. The sparsification mechanism is analogous to Yao et al. (2005) and applied to each observation and each element separately. The MFPCA is calculated in the same way as in setting 1, using the PACE approach to estimate $M_1 = M_2 = M_3 = 8$ functional principal components for each element. For data with high sparsity, we set $M_1 = M_3 = 3$ and $M_2 = 5$ to make computation of the univariate FPCA feasible.

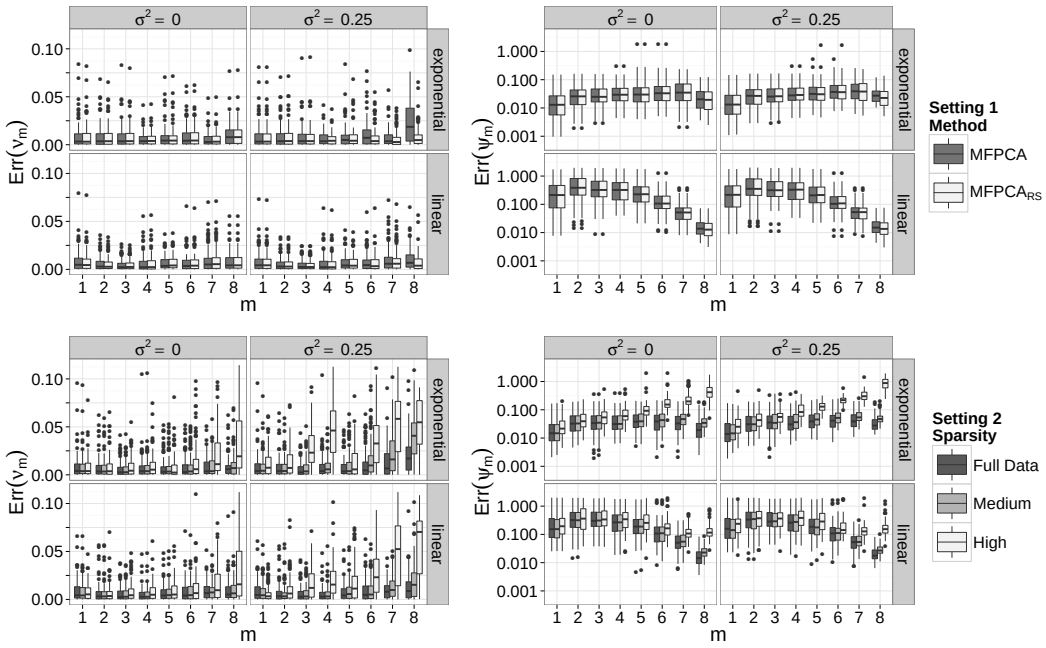


Figure 3.2.: Relative errors for estimated eigenvalues (left) and eigenfunctions (right, log-scale) for simulation settings 1 and 2, depending on eigenvalue decrease, measurement error and estimation method (setting 1) or sparsity (setting 2).

The results are given in Fig. 3.2 and Table 3.1. Here there is no available competitor. The performance of our MFPCA for full data is very similar to the simpler case of setting 1. Even for a moderate level of sparsity, the new method yields excellent results for most eigenvalues and eigenfunctions at the expense of somewhat higher reconstruction errors. For very sparse data, the leading eigenvalues and eigenfunctions are still estimated well, but the reconstruction error is considerably higher than for the full data. However, this is still acceptable (average MRSE is lower than 16% for all levels of sparsity), bearing in mind that data with high sparsity contains at most 10% of the original information. Again, simulations based on Legendre polynomials gave very similar results (not shown here).

3.4.2. Multivariate Functional Data Consisting of Functions and Images

Setting 3: Observations are generated based on $M = 25$ principal components, where the image elements $\psi_m^{(1)}$ are formed by tensor products of Fourier basis functions on $\mathcal{T}_1 = [0, 1] \times [0, 0.5]$ and $\psi_m^{(2)}$ are given by Legendre polynomials on $\mathcal{T}_2 = [-1, 1]$. The elements are weighted by random factors $\alpha^{1/2}$ and $(1 - \alpha)^{1/2}$, respectively, with $\alpha \in (0.2, 0.8)$ to ensure orthonormality. For the scores, only exponentially decreasing eigenvalues are used. The observations are discretized using a grid of $S_1 = 100 \times 50$ equidistant points for the image element and $S_2 = 200$ equidistant points for the functions.

We consider the new MFPCA approach based on univariate FPCA as well as non-orthogonal basis functions. In the first case, the eigendecomposition for the image data is calculated with the FCP-TPA algorithm for regularized tensor decomposition (Allen, 2013). The smoothing parameters for penalizing second differences in both image directions are chosen via generalized cross-validation in $[10^{-5}, 10^5]$ (Allen, 2013; Huang, Shen, et al., 2009). Multivariate FPCA is calculated based on $M_1 = 20$ eigenimages and $M_2 = 15$ univariate eigenfunctions. In the case of general basis functions, image elements are expanded in tensor products of $K_1 = 10 \times 12$ B-splines and the one-dimensional element is represented in terms of $K_2 = 15$ B-spline basis functions. In the presence of measurement error the univariate expansions are fit with appropriate smoothness penalties (Eilers and Marx, 1996).

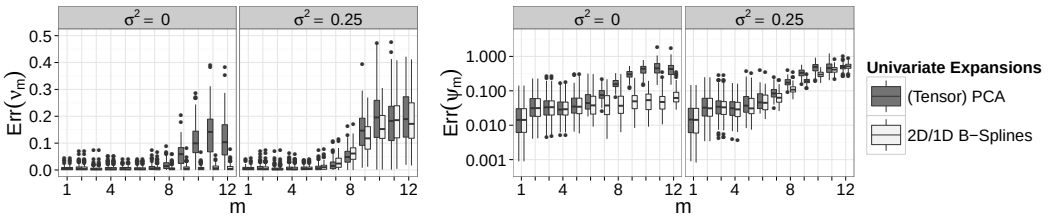


Figure 3.3.: Relative errors for estimated eigenvalues (left) and eigenfunctions (right, log-scale) for simulation setting 3, depending on measurement error and univariate expansions.

The overall results for the first $M = 12$ eigenvalue/eigenvector pairs are given in Fig. 3.3. Compared to the settings with one-dimensional domains, the errors are slightly higher, in particular for higher order eigenvalues and eigenfunctions. Exemplary results however, show that even in this case, the new approach is still able to capture the important features of the true eigenfunctions well (see online appendix). The results further show that the general approach with spline basis functions performs mostly better than the pure MFPCA approach. Moreover, the truncated Karhunen-Loève representation with $M = 12$ (true $M = 25$) estimated eigenfunctions and scores gives an excellent reconstruction of the original data. The average MRSE is 1.382%/0.398% (PCA/splines) for data without measurement error and 2.233%/2.048% (PCA/splines) for data with measurement error.

3.5. Application – ADNI Study

In this section, the new method is applied to data from the Alzheimer's Disease Neuroimaging Initiative study (ADNI), which aims at identifying biomarkers for accurate diagnosis of Alzheimer's disease (AD) in an early stage (Mueller et al., 2005). We use MFPCA to explore how longitudinal trajectories of a neuropsychological score (ADAS-Cog, a current standard for monitoring AD progression) covary with FDG-PET scans at baseline. The latter are used to assess the glucose metabolism in the brain, which is tightly coupled with neuronal function. As the brain images might be predictive of subsequent cognitive decline, common patterns between these two sources of information would be highly relevant.

Dataset: The dataset considered for MFPCA contains data from all $N = 483$ participants enrolled in ADNI1, having an FDG-PET scan at baseline and at least three ADAS-Cog measurements during follow-up. At baseline, 84 subjects were diagnosed with AD, 302 were suffering from mild cognitive impairment (MCI, in many cases an early stage of AD) and 97 were cognitively healthy elderly controls. The ADAS-Cog trajectories constitute the first element $X^{(1)}$, where high values indicate a high level of cognitive impairment. The measurements contain missings, mostly in the second half of the study period and thus are sparse. The second element $X^{(2)}$ is an axial slice of 93×117 pixels (139.5×175.5 mm^2) of FDG-PET scans, containing the Precuneus and temporo-parietal regions. Both are believed to show a strong relation between hypometabolism (reduced brain function) and AD (Blennow et al., 2006). Exemplary data is shown in Fig. 3.4.

Weighted scalar product: As the ADAS-Cog trajectories and FDG-PET scans differ considerably in domain, range and variation (cf. Fig. 3.4), we use a weighted MFPCA with

$$w_j = \left(\int_{\mathcal{T}_j} \hat{C}_{jj}(t_j, t_j) \mathrm{d}t_j \right)^{-1} = \left(\int_{\mathcal{T}_j} \widehat{\mathrm{Var}} \left(X^{(j)}(t_j) \right) \mathrm{d}t_j \right)^{-1}, \quad j = 1, 2,$$

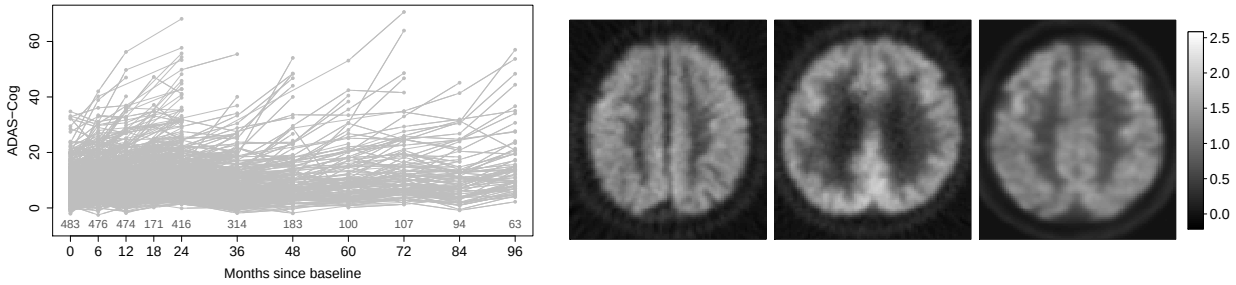


Figure 3.4.: Left: ADAS-Cog trajectories for all $N = 483$ subjects. Numbers above the x-axis give the total number of measurements for each visit. Right: FDG-PET scans for three randomly chosen male subjects (left to right: AD, MCI, normal; diagnosis at baseline).

where $\hat{C}_{jj}$ is estimated from the data. Using these weights, the integrated variance equals 1 for the rescaled elements $\tilde{X}^{(j)} = w_j^{1/2} X^{(j)}$. All elements thus contribute equal amounts of variation to the analysis, similarly to multivariate PCA, where the data is usually standardized before the analysis. We believe that this a sensible choice for many applications, but there may of course be situations, in which other weighting schemes may be preferable. For example, one could think of data that has two image elements, representing brain regions of different size for the same imaging modality. Here variability is naturally on the same scale and it might be better to keep the information of the site of the individual domains by setting both weights to one. On the other hand, if the images stem from different imaging modalities on the same domain, it might be necessary to correct solely for differences in variation. As a general rule, the weights should be chosen in close coordination with practitioners, considering the objective of the analysis and the data at hand.

Results: The results for the first two multivariate functional principal components, that account for 80.7% of the total weighted variance, are shown in Fig. 3.5. For the univariate expansions, we use FPCA for $X^{(1)}$ with $M_1 = 3$ principal components (explaining 99.2% of the univariate variance) and 20×15 tensor product B-splines for the images $X^{(2)}$. Fig. 3.5 further includes pointwise bootstrap confidence bands for the principal components based on 100 nonparametric bootstrap iterations on the level of subjects. The coverage of such confidence bands for data consisting of functions and images has been analyzed in a simulation study, which gave good results, even in the presence of measurement error (see online appendix). The entire analysis for the ADNI data took around 15 minutes on a standard laptop (2.7 GHz, 16 GB RAM) including the calculation of the bootstrap confidence bands and without parallelization.

Almost half of the variability in the data (46.7% of the weighted variance) is explained by the first functional principal component. The ADAS-Cog element – and hence the degree of cognitive impairment – is elevated relative to the mean and increases during follow-up. The FDG-PET element exhibits hypometabolism in the Precuneus and the temporo-parietal regions, i.e. this component reflects reduced brain activity in these regions already

at baseline. In total, the first eigenfunction seems to be interpretable as an AD related effect, as the pattern for positive scores perfectly agrees with medical knowledge about AD progression. This interpretation is supported by the estimated scores, which are mainly positive for people diagnosed with AD by their last visit, while scores of subjects who remained cognitively normal during follow-up are nearly all negative. Persons with MCI have intermediate score values, which is in line with the hypothesis that this diagnosis can constitute a transitional phase between normal ageing and AD.

For the second functional principal component (explains 33.9% of weighted variance), the ADAS-Cog element is nearly constant and has wide bootstrap confidence bands that include zero during the whole follow-up. In contrast, the FDG-PET element differs significantly from zero in almost all voxels (cf. Fig. 3.5). Hence, this principal component reflects variation in the FDG-PET scans at baseline. Plotting the overall mean plus or minus this component suggests that it can be interpreted as an effect of imperfect registration that manifests in different brain sizes, which are known to correlate with gender (Ruigrok et al., 2014). This hypothesis is supported by the boxplots of the estimated scores in Fig. 3.5, while scores do not differ notably by diagnosis (not shown here).

Discussion: The results show that MFPCA is able to capture important sources of variation in the data that have a meaningful interpretation from a medical and neuroimaging point of view. An important issue not addressed here is that for ADAS-Cog, there may well be an informative dropout of patients with high score values (cf. Fig. 3.4). While addressing informative missingness goes beyond the scope of this paper, interpretation of results should take this possibility into account. For instance, it is easily conceivable that $\hat{\psi}_1^{(1)}$ may be underestimating $\psi_1^{(1)}$ towards the end of the study period.

3.6. Discussion and Outlook

This paper introduces methodology and a practical estimation algorithm for multivariate functional principal component analysis. While other methods for MFPCA are restricted to observations on a common, one-dimensional interval, the new approach is suitable for data on different domains, which may also differ in dimension, such as functions and images. The key results are 1. a Karhunen-Loève Theorem, that establishes the theoretical basis for MFPCA (Prop. 4), 2. an explicit relation between multivariate and univariate FPCA, which serves as a starting point for the estimation (Prop. 5) and 3. asymptotic results for the estimators (Prop. 6 and 7). The estimation algorithm can be extended to expansions of the univariate elements in not necessarily orthonormal bases. This allows to flexibly choose an appropriate basis for each element depending on the data structure, in particular also mixtures of univariate FPCA and general bases. The algorithm is applicable to sparse data or data with measurement error, as well as to images. Notably, the proposed method can

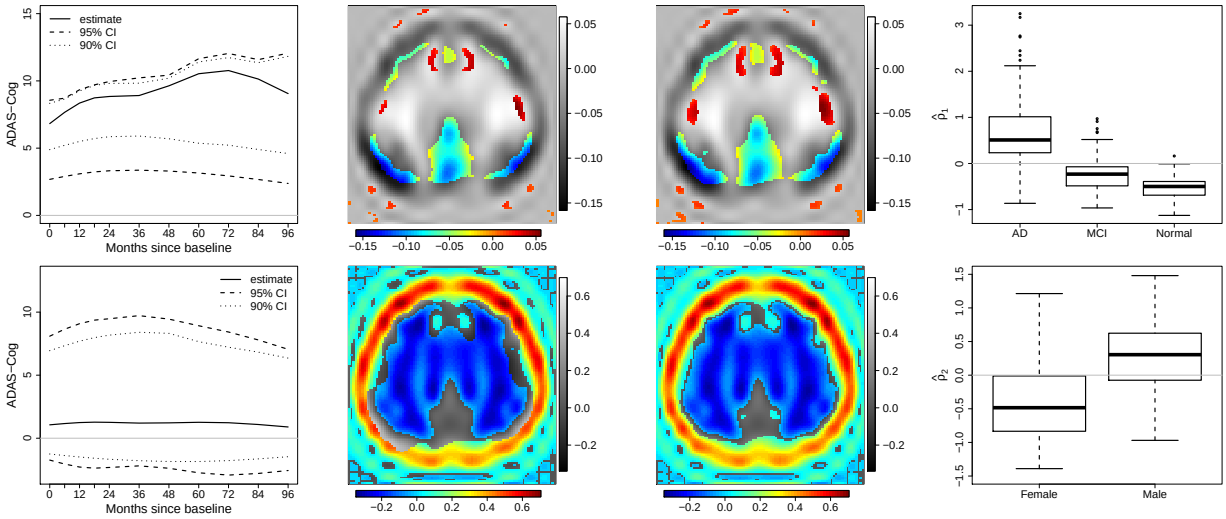


Figure 3.5.: The first two estimated multivariate functional principal components for the ADNI data (1st row: $\hat{\psi}_1$, 2nd row: $\hat{\psi}_2$). Estimates are given with pointwise 95% and 90% bootstrap confidence bands based on 100 nonparametric bootstrap iterations (ADAS-Cog, 1st column: Dashed lines; FDG-PET, 2nd and 3rd column: Pixels with pointwise 95% (left) and 90% (right) confidence bands not including zero in color). Boxplots of the scores (4th column) support the interpretation.

be used to calculate smooth univariate functional principal components for data on higher dimensional domains and is hence an alternative to existing methods for tensor PCA (Allen, 2013). The results of MFPCA give insights into simultaneous variation within the data and provide a natural tool for dimension reduction. Moreover, they can be used as a building block for further statistical analyses such as functional clustering methods or functional principal component regression with multiple covariates (cf. Müller and Stadtmüller, 2005, for the univariate case). If the elements differ in domain, range or variation, the new method can incorporate weights, which should be chosen with respect to the question of interest and the data at hand.

Possible extensions of the approach include normalization methods as an alternative to the weighted scalar product, following the ideas in Jacques and Preda (2014) or Chiou, Yang, et al. (2014) for functions observed on a common interval. However, one should take into account that the domains may have different dimensions and sizes. The concept of MFPCA could further be extended to *hybrid data*, i.e. data consisting of a functional and a vector part (Ramsay and Silverman, 2005, Chapter 10.3.). A natural starting point would be to extend the scalar product suggested by Ramsay and Silverman (2005) in this context to multivariate functional data as proposed in Prop. 1. However, transferring the results for MFPCA shown in this paper requires a careful revision of the concept of the covariance operator and related proofs. Finally, one could think of estimating the multivariate covariance operator directly without computing a univariate decomposition

for each element. This operator is typically high-dimensional, making smoothing as well as an eigendecomposition hardly feasible, which is avoided in our two-step approach.

4. Application

This chapter is based on:

Araque Caballero*, M. Á., C. Happ*, S. Greven, V. J. Schmid, and M. Ewers for the Alzheimer's Disease Neuroimaging Initiative (ADNI) (2017). "Multivariate Functional Principal Component Analysis for the Prediction of Longitudinal Cognitive Decline in Alzheimer's Disease". Manuscript in preparation.

Author contributions:

The manuscript was written by Miguel Ángel Araque Caballero and Clara Happ. Both authors contributed equally to this work. The FDG-PET scans were preprocessed by Miguel Ángel Araque Caballero. All computations concerning the correction of the data, the MFPCA and subsequent analyses were done by Clara Happ. Michael Ewers, Sonja Greven and Volker Schmid added valuable input and proofread the manuscript.

Acknowledgements:

The authors thank the Alzheimer's Disease Neuroimaging Initiative (ADNI) Committee for providing access to the data. Financial support from the German Research Foundation through Emmy Noether grant GR 3793/1-1 is gratefully acknowledged.

4.1. Introduction

In Chapter 3, multivariate functional principal component analysis was introduced for multivariate functional data on different dimensional domains. In the application to the ADNI data in Section 3.5 it was shown that MFPCA can give valuable insights into the joint variation of ADAS-Cog trajectories and FDG-PET scans, finding patterns that are known to be related to early signs of Alzheimer's disease (AD). In this chapter, the analysis of the ADNI data is deepened and extended by several aspects.

First of all, the full FDG-PET scans are considered for each subject, which means that the amount of data considerably increases from 2D images with $93 \times 117 \approx 1.1 \cdot 10^4$ pixels to 3D images with $121 \times 145 \times 121 \approx 2.1 \cdot 10^6$ voxels per subject. This is primarily a computational challenge, as the data can hardly be analyzed with standard computers (the raw data already requires 11 GB of memory). Moreover, a spline interpolation as in Section 3.5 would become computationally very demanding. For this reason, it is replaced by a discrete cosine transformation, for which fast and efficient algorithms have been developed (Frigo and Johnson, 2005). The second principal component found in Chapter 3 further suggests that there is a non negligible effect of imperfect registration in the images. In order to remove this effect, the images have been registered before this analysis, using MRI (magnetic resonance imaging) scans at baseline. We use data from the ADNI2 study only, as here MRI T1 scans with a field strength of 3 Tesla are available for all subjects.

Second, the ADAS-Cog scores in the previous chapter were assumed to be measured at regular intervals. This of course is a simplifying assumption, as one can expect a that the exact dates can deviate from the schedule of visits by some days or weeks. To this end, we consider ADAS-Cog at the exact visit dates for the analysis in this chapter. This does not only add more information, but may also help to stabilize the univariate FPCA (Yao et al., 2005).

In order to account for potentially confounding factors, both ADAS-Cog and FDG-PET were adjusted for age, gender and years of education in this analysis.

The resulting MFPCA is in line with the results in the previous chapter. For the first functional principal component, we find again an overall positive and increasing ADAS-Cog element, associated with an FDG-PET element showing negative values, particularly in the precuneus and lateral temporo-parietal regions. This is indicative of reduced global cognition with respect to the overall mean, becoming worse during follow-up, associated with below-average tracer uptake and thus hypometabolism in the affected brain areas at baseline. The second principal component does not show a registration effect as in Chapter 3. This demonstrates that registrations and adjustments for confounding factors were successful.

Another central aim of the analysis in this chapter is to use MFPCA for the long-term prediction of ADAS-Cog. For subjects whose data has been used for the calculation of the MFPCA (in-sample), the trajectories can easily be reconstructed based on the multivariate Karhunen-Loève theorem proven in the previous chapter. As scores with very high absolute values can yield reconstructions that exceed the natural range of ADAS-Cog (0 – 70), we propose to transform the data before the analysis and retransform the predictions back to the original scale. The reconstructed trajectories span the full study period of around 5.5 years, even if subjects have been measured only in the first one or two years after baseline.

For the prediction of ADAS-Cog for new subjects, a simple projection on the multivariate functional principal components is not applicable due to the sparse nature of the ADAS-Cog measurements. Instead, it is shown how individual scores can be calculated for each principal component, following the steps of the MFPCA algorithm on the basis of given MFPCA results combined with data from the new subject. Once the scores are given, predicted ADAS-Cog trajectories can be calculated as for the in-sample case. This new method is applied for calculating predicted ADAS-Cog trajectories for subjects with preclinical AD (HC-$A\beta$+) based on the MFPCA results for subjects with prodromal AD (MCI-$A\beta$+). Here also, a transformation of the data ensures that the predictions stay within the scale of ADAS-Cog.

The chapter is structured as follows: In Section 4.2 we describe the study design and the methods used in the analysis. An overview of all steps is given in Fig. 4.1. The results of the analysis are presented in Section 4.3. The chapter concludes with a summary and a discussion of the results in Section 4.4. Supplementary plots are given in the appendix (Appendix B).

4.2. Methods

4.2.1. Study Design

Subjects

We included 754 subjects from the Alzheimer's Disease Neuroimaging Initiative (ADNI) (Weiner et al., 2015). The inclusion criteria for this study were: FDG-PET, AV45-PET, MRI T1 and cognitive assessments (summary score the Alzheimer's Disease Assessment Scale – Cognitive Subscale (ADAS-Cog)) available at baseline and at least two additional cognitive assessments at later visits. The subjects were diagnosed at baseline as healthy controls (HC), mild cognitive impairment (MCI) and subjects with Dementia due to Alzheimer's Disease (AD) as per Petersen et al. (2010).

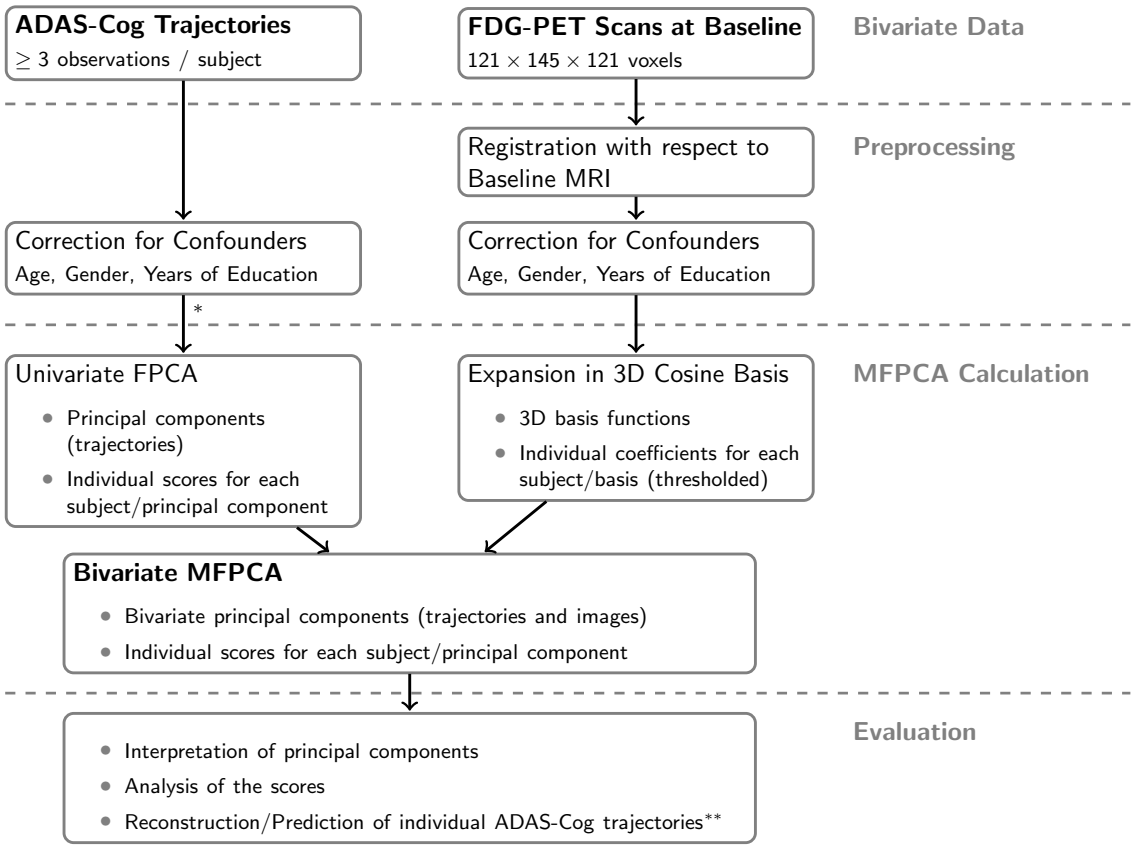


Figure 4.1.: Schematic overview of the full analysis. If a reconstruction/prediction was calculated, the ADAS-Cog trajectories have been transformed before the analysis (*) and the reconstructed/predicted values have been transformed back to the original scale (**).

Amyloid-β Status

AV45-PET scans were obtained 50 minutes after injection of 10±1.0 mCi of [^{18}F]-Florbetapir and consisted of four 300 s frames. All frames were averaged and attenuation-corrected and the resulting images were transformed to standard orientation, resolution and voxel-size. The ADNI PET Core at the University of Pittsburgh performed the image preprocessing and SUVR quantification (Jagust et al., 2010). All the details, ROIs and procedures can be found at the ADNI website (`http://adni.loni.usc.edu/data-samples/pet`).

Based on the global SUVR, we dichotomized the subjects into those with normal and abnormal levels of brain $A\beta$ ($A\beta-$ and $A\beta+$ respectively) using a cutoff value of 1.11 (Landau, Breault, et al., 2013). In order to exclude subjects potentially misdiagnosed with AD or with mixed dementias (Landau, Horng, et al., 2016; Chételat et al., 2016), we excluded 13 AD-$A\beta-$ subjects from all analyses, resulting in a final sample of 741.

FDG-PET Acquisition

FDG-PET scans were obtained 30 minutes after injection of 5 ± 0.5 mCi of [^{18}F]-Fluorodeoxyglucose and consisted of six 300 s frames. All frames were averaged and attenuation-corrected and the resulting images were transformed to standard orientation, resolution and voxel-size. As with AV45-PET, all these steps were performed by the ADNI PET Core and the scans used in this study were downloaded from the ADNI database (https://ida.loni.usc.edu/) with these basic preprocessing steps already performed (Jagust et al., 2010).

4.2.2. Preprocessing

FDG-PET Processing

The downloaded FDG-PET scans were normalized to standard MNI space with a pipeline described previously (Araque Caballero et al., 2015). Briefly, nonlinear normalization parameters were estimated on the basis of the T1 scans using high-dimensional diffeomorphic transformations from the DARTEL toolbox of SPM12 (Ashburner, 2007). To adjust the FDG-PET uptake to the same scale across subjects, the voxel SUVR values for each MNI-normalized image were divided with the average SUVR within the pons and vermis. These MNI-normalized, pons-and-vermis scaled FDG-PET scans were the input to our analysis.

Adjustment for Confounding Variables

The ADAS-Cog scores were adjusted for the effect of potentially confounding variables (age, gender, years of education) based on the HC group. We used the following generalized additive model (GAM) (Hastie and Tibshirani, 1986; Wood, 2006)

$$\begin{aligned}\mathrm{ADASCog}_{ij} = \alpha + f_{\mathrm{Time}}\left(t_{ij}\right) + \gamma_i + \beta_{\mathrm{Gender}}\mathbb{1}_{\mathrm{Male}}\left(\mathrm{Gender}_i\right) + \beta_{\mathrm{EDU}}\mathrm{EDU}_i \\ + f_{\mathrm{Age;m}}\left(\mathrm{Age}_{ij}\right)\mathbb{1}_{\mathrm{Male}}\left(\mathrm{Gender}_i\right) + f_{\mathrm{Age;f}}\left(\mathrm{Age}_{ij}\right)\mathbb{1}_{\mathrm{Female}}\left(\mathrm{Gender}_i\right) + \varepsilon_{ij}\end{aligned}$$

with $\mathrm{ADASCog}_{ij}$ the ADAS-Cog value and t_{ij} the years since baseline of the i-th subject at the j-th visit (other variables analogously) and EDU as years of education. Although the follow-up visits are scheduled at regular intervals, we use the exact visit dates for t_{ij}, in order to account for deviations from the schedule of visits and to stabilize the following principal component analysis (Yao et al., 2005). Time and age are modeled with a smooth effect, with age being gender-specific. An analogous model having a smooth effect for EDU resulted in a nearly linear function, showing that a linear approach is sufficient for capturing the effect of EDU in this context. The estimated smooth coefficient functions are shown in

Fig. B.2 in the appendix. The estimates $\widehat{\text{ADASCog}}_{ij}$ based on the HC group were then used for all subjects to predict residuals $\hat{\varepsilon}_{ij} = \text{ADASCog}_{ij} - \widehat{\text{ADASCog}}_{ij}$ and calculate adjusted values

$$\text{ADASCog}_{ij}^{\text{adj}} = \hat{\alpha} + \hat{f}_{\text{Time}}\left(t_{ij}\right) + \hat{\beta}_{\text{Gender}} + \hat{\beta}_{\text{EDU}}\text{EDU}_{\text{ref}} + \hat{f}_{\text{Age;m}}\left(\text{Age}_{\text{ref},0} + t_{ij}\right) + \hat{\varepsilon}_{ij}$$

with the reference subject being male, $\text{Age}_{\text{ref},0} = 72.4$ (the median age at baseline) and $\text{EDU}_{\text{ref}} = 16$ (the median years of education). Examples for adjusted trajectories are shown in Fig. B.1 in the appendix.

The preprocessed FDG-PET scans were adjusted for age, gender and EDU based on the HC group, similarly to the adjustment procedure applied to the ADAS-Cog trajectories. The following linear regression model was fit separately for each voxel based on the HC group

$$\begin{aligned}\text{PET}_{i\nu} = \alpha_\nu + \beta_{\text{Gender},\nu}\mathbb{1}_{\text{Male}}\left(\text{Gender}_i\right) + \beta_{\text{Age},\nu}\text{Age}_{i0} + \beta_{EDU,\nu}\text{EDU}_i \\ + \beta_{\text{Gender;Age},\nu}\text{Age}_{i0}\mathbb{1}_{\text{Male}}\left(\text{Gender}_i\right) + \varepsilon_{i\nu}\end{aligned}$$

with $\text{PET}_{i\nu}$ the FDG-PET value of subject i in voxel ν. The main difference between the models for ADAS-Cog and for FDG-PET is that only data from baseline has been considered (no time variable) and that the model includes only linear effects for all variables. Tests with nonlinear effects resulted in nearly linear effects in all cases. Based on the estimates found in the HC group, the adjustment for age, gender and EDU in all subjects was

$$\text{PET}_{i\nu}^{\text{adj}} = \hat{\alpha}_\nu + \hat{\beta}_{\text{Gender},\nu} + \hat{\beta}_{\text{Age},\nu}\text{Age}_{\text{ref},0} + \hat{\beta}_{\text{EDU},\nu}\text{EDU}_{\text{ref}} + \hat{\beta}_{\text{Gender;Age},\nu}\text{Age}_{\text{ref},0} + \hat{\varepsilon}_{i\nu}.$$

4.2.3. Multivariate Functional Principal Component Analysis

Introduction to MFPCA

Multivariate functional principal component analysis (MFPCA, Happ and Greven, 2017+) allows to calculate a principal component analysis for multivariate functional data with potentially different dimensional domains. The term "functional data" (Ramsay and Silverman, 2005) in this context means that the values of FDG-PET can be interpreted as discrete evaluations of a (smooth) function on a three-dimensional space (i.e. the brain), and similarly the repeatedly sampled ADAS-Cog scores can be described as a smooth function over time for each subject. Combining both modalities, the ADAS-Cog trajectories and the FDG-PET scans, results in a bivariate functional data set. As for the usual principal component analysis (PCA), the main aim of MFPCA is to find the most important modes of variation in the data. The advantage of MFPCA over the usual PCA, however, is two-fold: First, considering the data as functions takes the spatial (for FDG-PET) and

temporal (for ADAS-Cog) structure of the data into account. Second, it allows to consider potential covariation between the modalities. This is achieved by expanding each modality separately in a (univariate) basis and combining the information of the individual coefficients for each modality for the calculation of multivariate functional principal components. The principal components found by the MFPCA method have the same structure as the data, i.e. each principal component consists of an ADAS-Cog element and an FDG-PET element. Subject-specific principal component scores can be derived for each component, which are to be interpreted as weights for a particular principal component in representing the data of a particular subject. For technical details see Happ and Greven (2017+).

Whole Sample MFPCA

In a first analysis, MFPCA was calculated for the whole sample (N = 741), including subjects with AD, MCI and HC. The aim was to estimate joint patterns of baseline FDG-PET and cognitive decline across a wide clinical spectrum of AD. In particular, we assessed the ability of the individual MFPCA scores to differentiate between diagnostic groups. Moreover, the accuracy of the reconstructed ADAS-Cog trajectories was quantified. In order to ensure that the reconstructions stay within the total range of ADAS-Cog (0 – 70), the MFPCA was rerun with transformed ADAS-Cog values ($\log\left(\frac{\text{ADASCog}}{70-\text{ADASCog}}\right)$, where the factor 70 transforms the ADAS-Cog scale to a 0 – 1 range) and the predicted trajectories were transformed back to the original scale.

As a first step of the MFPCA, a univariate basis expansion was calculated for each element (cf. Fig. 4.1). For ADAS-Cog, functional principal component analysis (FPCA, Yao et al., 2005) was applied to the trajectories of the adjusted ADAS-Cog scores. The first two principal components explained 97.7% of the variability in the data and were included in the subsequent MFPCA. For the expansion of the FDG-PET elements, a 3D cosine basis was used (Frigo and Johnson, 2005). As the signal here concentrates on a small proportion of the cosine basis coefficients, they were thresholded according to their absolute values, keeping only the most extreme 2.5% of the values and setting the remaining 97.5% of the coefficients to zero. The resulting basis functions and individual coefficients for each subject were subsequently fed into the MFPCA for combined analysis. In order to ensure that both modalities contribute equal amounts of variation, the components were weighted by the inverse of the integrated pointwise variance as proposed in Happ and Greven (2017+). For ADAS-Cog, the weight was calculated based on the smooth variance estimate from the univariate FPCA, while for FDG-PET the variance was calculated on the basis of voxels. This resulted in weights $w_{\text{ADAS}} = 3.64 \cdot 10^{-3}$ for ADAS-Cog and $w_{\text{PET}} = 3.00 \cdot 10^{-5}$ for the FDG-PET. The MFPCA implementation in our `R` package `MFPCA` (Happ, 2017b) was used for the analysis. After loading the data (the full uncompressed images require approximately 11 GB space), the full MFPCA took roughly ten minutes.

Bootstrap Confidence Bands for Functional Principal Components

In order to quantify the estimation uncertainty in the MFPCA estimates, nonparametric 95% pointwise bootstrap confidence bands were calculated for the functional principal components. The resampling was done on the level of subjects and in a stratified manner, keeping the proportions of AD, MCI and healthy subjects at baseline constant over the 100 bootstrap iterations. The confidence bands were then calculated pointwise (for each time point and each voxel separately) using the empirical 2.5% and 97.5% quantiles as lower and upper bounds.

Reconstruction of ADAS-Cog Trajectories

The MFPCA results can be used to reconstruct ADAS-Cog trajectories based on the multivariate Karhunen-Loève representation (Happ and Greven, 2017+)

$$\widehat{\text{ADASCog}}_i(t) = \hat{\mu}^{\text{ADAS}}(t) + \sum_{m=1}^{M} \hat{\rho}_{i,m} \hat{\psi}_m^{\text{ADAS}}(t), \qquad t = \text{time since baseline.} \tag{4.1}$$

Here $\widehat{\text{ADASCog}}_i$ denotes the reconstructed trajectory for subject i, $\hat{\psi}_m^{\text{ADAS}}$ is the estimated ADAS-Cog element of the m-th principal component (we use $M = 5$ for reconstruction), $\hat{\rho}_{i,m}$ the associated subject-specific score and $\hat{\mu}^{\text{ADAS}}$ is the smoothed mean ADAS-Cog trajectory, which is calculated during the MFPCA.

The accuracy of the reconstruction was measured by the root mean of squared differences between corrected and predicted ADAS-Cog values for each individual (root mean squared error, RMSE):

$$\text{RMSE}_i = \sqrt{\frac{1}{N_i} \sum_{j=1}^{N_i} \left(\text{ADASCog}_{ij} - \widehat{\text{ADASCog}}(t_{ij})\right)^2}, \qquad i = 1, \ldots, N$$

with N_i the number of visits for subject i.

4.2.4. Out-of-Sample Prediction of ADAS-Cog Trajectories

From a clinical point of view, it would be particularly attractive if MFPCA results showed utility for predicting potential long-term cognitive decline in the preclinical phase of AD (HC-$A\beta$+). To this end, MFPCA was calculated within the MCI-$A\beta$+ subgroup only (training group) in order to yield an independent estimate of MFPCA for subsequent prediction for HC $A\beta$+ (test group). We chose MCI-$A\beta$+ as the training data set since AD related changes in FDG-PET are usually well present without severe global brain damage as observed in the AD dementia stage (Araque Caballero et al., 2015).

Calculating Predictions

We propose to calculate out-of-sample predictions in analogy to the reconstruction in (4.1). The key is hence to find good estimates ρ_m^* based on the MFPCA results and the data available for a new subject. We suggest to calculate the scores in analogy to the MFPCA algorithm (see Fig. 4.1). In a first step, the new data is demeaned by the in-sample means $\hat{\mu}^{\text{ADAS}}$ and $\hat{\mu}^{\text{PET}}$ and projected separately onto the univariate bases used for MFPCA. This results in univariate functional principal component scores for ADAS-Cog and thresholded cosine basis coefficients for FDG-PET. These univariate component scores are used together with the MFPCA results to calculate multivariate scores ρ_m^* for $m = 1, \ldots, M$, according to the formula in Happ and Greven (2017+)

$$\rho_m^* = \hat{\nu}_m^{1/2} \left(\hat{c}_m^\top \hat{Q}_w \hat{c}_m \right)^{-1/2} \theta^* D \hat{c}_m.$$

Here θ^* denotes the vector of univariate functional principal component scores for ADAS-Cog and the thresholded cosine basis coefficients for FDG-PET for a new subject. All other quantities stem from the original MFPCA: $\hat{\nu}_m$ is the eigenvalue associated with the m-th principal component, $\hat{c}_m$ is the m-th eigenvector of the matrix $\hat{Q}_w = (N-1)^{-1} D \Theta^\top \Theta D$ with N being the number of subjects in the sample for which the MFPCA was calculated, D a diagonal matrix of the weights $w_{\text{ADAS}}^{1/2}$ and $w_{\text{PET}}^{1/2}$, each repeated according to the number of univariate basis functions used. The matrix Θ contains the analogues of θ^* for the original sample in a row-wise manner.

Out-of-sample Predictions of ADAS-Cog in HC-$A\beta$+

In order to predict the ADAS-Cog trajectories of HC-$A\beta$+ over several years, we use the baseline FDG-PET scans and the ADAS-Cog values at baseline and in the first year of follow-up. This choice ensures that two univariate principal component scores can be obtained for ADAS-Cog and all HC-$A\beta$+ subjects, which are needed for the subsequent calculation of the multivariate scores that form the basis of the prediction.

As for the in-sample prediction, the ADAS-Cog values for both MCI-$A\beta$+ and HC-$A\beta$+ were transformed by the logit before the analysis and the final predictions of ADAS-Cog for HC-$A\beta$+ were retransformed to the original scale. Note that this transformation affects the interpretation of the principal components and scores obtained for MCI-$A\beta$+, which, however, is not crucial for prediction. The prediction accuracy was measured via RMSE.

4.3. Results

4.3.1. Subject Characteristics

Subject characteristics are presented in Table 4.1 for the full sample and for the HC-$A\beta+$ and MCI-$A\beta+$ groups separately. HC-$A\beta+$ had a higher proportion of females ($\chi^2 = 13.05 \sim \chi^2_1,\ p < 0.001$) than MCI-$A\beta+$ subjects. As expected, ADAS-Cog values at baseline for HC-$A\beta+$ were lower than for MCI-$A\beta+$ ($t = 10.65 \sim t_{213.7},\ p < 0.001$). Mean follow-up time was 2.92 years (Range 0.85 – 5.47).

Table 4.1.: Subject characteristics at baseline

	Full Dataset	**MCI-$A\beta+$**	**HC-$A\beta+$**
No. Subjects	741	230	73
Age – mean (SD) Range	72.4 (7.09) 55 – 91.4	72.87 (6.71) 55 – 87.8	74.12 (5.72) 59.7 – 84.8
Gender – F/M (% of females)	356 / 385 (48%)	106 / 124 (46%)	52 / 21 (71%)*
Years of education – mean (SD) Range	16.25 (2.64) 8 – 20	16.01 (2.8) 9 – 20	16.01 (2.65) 8 – 20
Diagnosis (baseline)	85 (AD) 418 (MCI) 238 (HC)	-	-
ADAS-Cog – mean (SD) Range	9.34 (5.99) 0 – 38	10.47 (4.68) 2 – 27	5.77 (2.7)* 1 – 15

* $p < 0.001$ (MCI-$A\beta+$ vs. HC-$A\beta+$) Abb.: SD = Standard Deviation, F = Female, M = Male

4.3.2. Whole Sample MFPCA

MFPCA Component Overview

In total, five principal components were calculated. The first two components, which explained 97.1% of the variability in the data, are shown in Fig. 4.2. The first principal component explains most of the variability (93.9%). Its ADAS-Cog element has positive values for the whole study period and is increasing over time, with a change in slope after around two years since baseline. Positive score values thus represent a trajectory of cognitive decline typical of AD. The 95% pointwise bootstrap confidence bands show that the first principal component differs significantly from zero throughout the trajectory. The FDG-PET element of the first principal component shows negative values in all voxels, indicative of below-average tracer uptake. The lowest values of the element are distributed

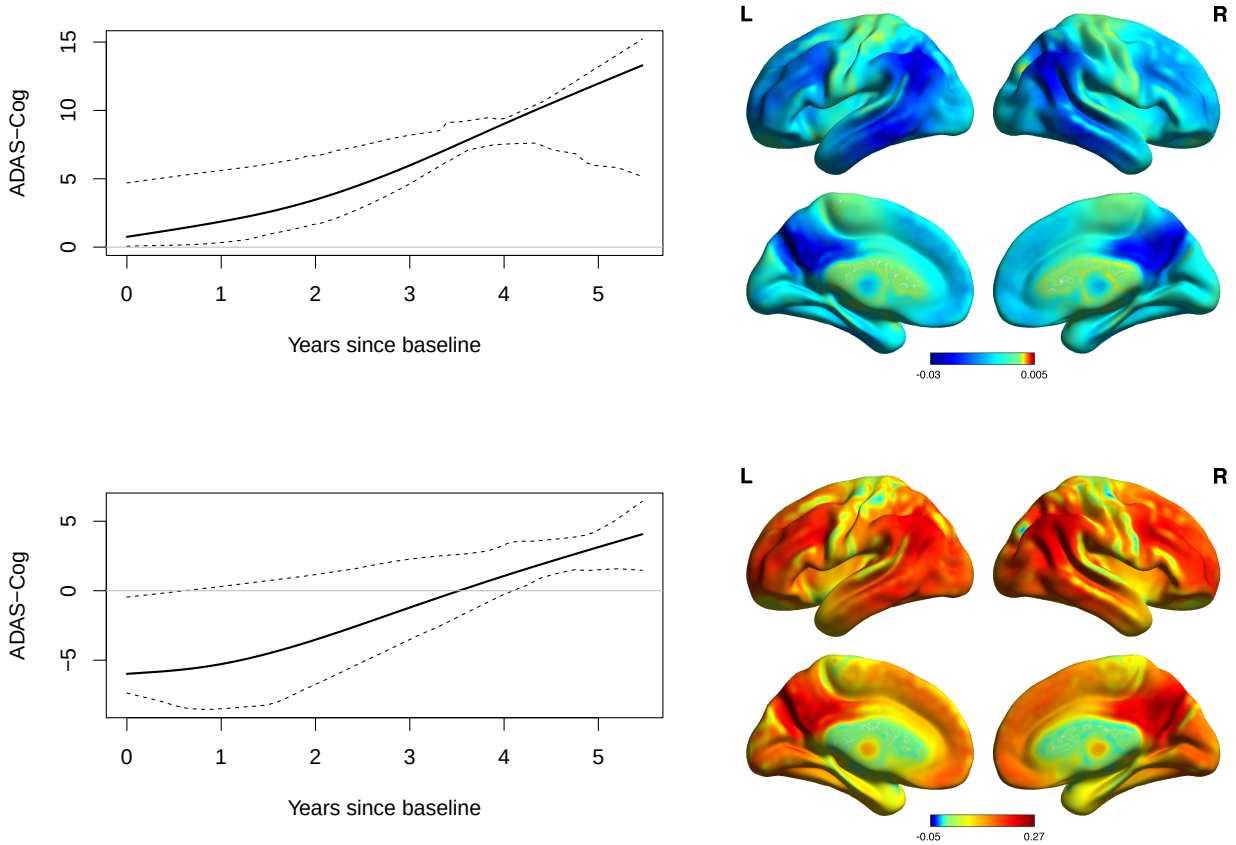


Figure 4.2.: The first two principal components for the whole sample MFPCA. Left: ADAS-Cog element for each component (solid lines), along with the bootstrapped 95% CIs of the estimate (dotted lines). The gray vertical line marks zero. Right: FDG-PET element of each of the two components projected on to the brain surface. The FDG-PET elements are masked such that any voxels having zero within the voxelwise bootstrapped 95% CIs were excluded, but only voxels outside the brain were non-significant.

within the precuneus and lateral temporo-parietal regions, which is consistent with the pattern of hypometabolism typically found in AD (La Joie et al., 2012). Virtually all of the voxels in the brain had bootstrapped 95% CIs excluding zero, indicating that they differ significantly from zero.

The second principal component explains 3.2% of the variability in the data. The ADAS-Cog element starts at a negative value and increases over time, becoming positive at about 3.5 years after baseline. Positive score values thus indicate that a subject starts with relatively benign ADAS-Cog values (i.e. above-average cognition) and progressively worsens. The confidence band includes zero except for two short periods at the beginning of the study and starting from four years after baseline, which means that the ADAS-Cog element

does not differ significantly from zero during most of the study. The FDG-PET element of the second principal component shows overall positive values, indicative of above-average tracer uptake. All the voxels in the brain had bootstrapped 95% CIs excluding zero, where the highest voxel values are found in the precuneus, lateral temporo-parietal and lateral frontal regions. That is, the FDG-PET element of the second component is representative of relatively well-preserved FDG-PET uptake in core AD regions.

Group Differences in MFPCA Scores

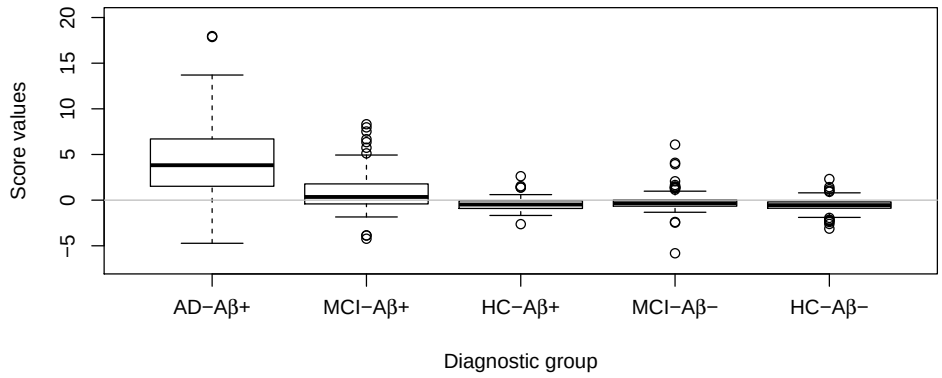


Figure 4.3.: MFPCA score values for the first principal component in the whole sample MFPCA depending on diagnostic groups. The gray horizontal line marks zero. An analogous plot for the scores of the second principal component is shown in the appendix in Fig. B.3.

The scores of the first principal component are shown in Fig. 4.3 depending on the diagnostic groups (one extreme value was removed from the plot). An analogous plot for the scores of the second principal component are shown in Fig. B.3 in the appendix.

The scores of the first principal component are mostly positive for the AD-$A\beta+$ group and partly also for MCI-$A\beta+$, which is consistent with the pattern of ADAS-Cog and reduced FDG-PET uptake associated with this component. The scores of the second principal component are mostly negative in the AD-$A\beta+$ group. The highest positive score values were found for healthy controls (HC-$A\beta+$ and HC-$A\beta-$), suggesting a slower increase in ADAS-Cog values and higher FDG-PET uptake present in the HC groups compared to the overall average.

Group-wise t-tests showed significant differences in the scores of AD-$A\beta+$ and MCI-$A\beta+$ compared to HC-$A\beta-$ for both principal components ($p < 0.001$ after Bonferroni adjustment). The scores were not dependent on gender or years of education, which reflects that

both the ADAS-Cog trajectories and the FDG-PET scans have been corrected for these confounders.

Fig. 4.4 shows the ADAS-Cog trajectories depending on the score value associated with the first principal component. An analogous plot for the second principal component is given in the appendix in Fig. B.4. The first component distinguishes quite well between subjects with constant trajectories and low ADAS-Cog values (negative score values) and trajectories with an increasing trend (positive score values). Notably, trajectories that decrease towards the last visit mainly also have negative scores.

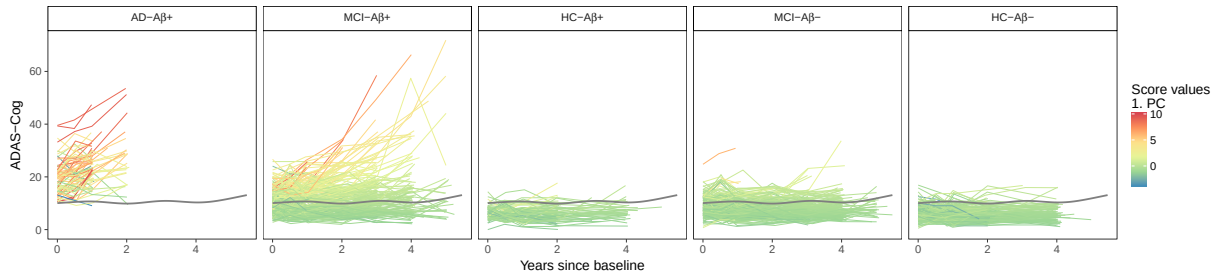


Figure 4.4.: Corrected ADAS-Cog trajectories by diagnostic group. The colors of the lines correspond to the score values for the first principal component in the whole sample MFPCA. For reasons of clarity, the 1% subjects having the highest and lowest score values, each, have been removed from the plot. The gray curve marks the smooth overall mean. An analogous plot for the scores of the second principal component is shown in the appendix in Fig. B.4.

In-sample Reconstruction of ADAS-Cog

Based on the estimated principal components for the MFPCA with transformed ADAS-Cog trajectories and the resulting individual score values, ADAS-Cog trajectories were reconstructed for each subject (see Fig. B.5 in the appendix). Overall, the reconstructed trajectories fit the observed ones quite well, with an average predictive accuracy of RMSE = 1.56 (standard dev. 1.11), meaning that the average deviation between the reconstructed and the original ADAS-Cog values per subject is 1.56 units on the ADAS-Cog scale. For subjects with a high variability in the score values over time, the predicted trajectories correspond to a global smooth trend of the observed values. Boxplots of the RMSE depending on the diagnostic group are shown in Fig. 4.5.

4.3.3. Out-of-Sample Predictions of ADAS-Cog in HC-$A\beta+$

In the first step, the MFPCA was calculated in the MCI-$A\beta+$ group based on the baseline FDG-PET and the transformed ADAS-Cog trajectories. The results were used to calculate

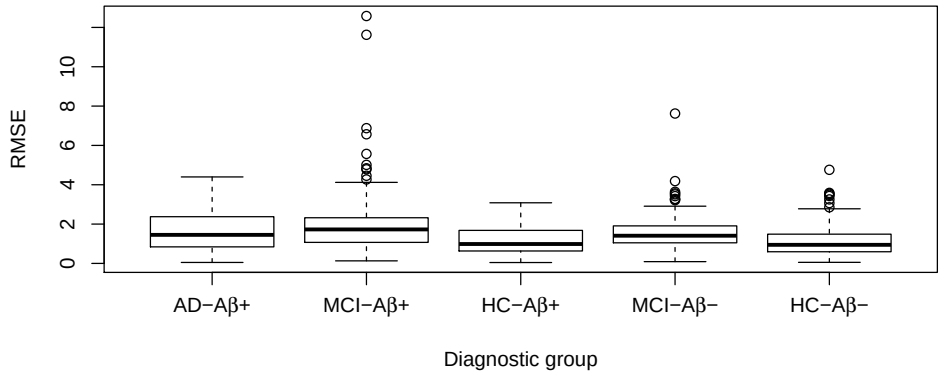


Figure 4.5.: Goodness of fit in the whole sample MFPCA: The boxplots show the root mean squared error (RMSE) depending on the diagnostic groups.

individual predictions of ADAS-Cog for HC-$A\beta+$ based on the observed FDG-PET scans of this group together with a minimal follow-up of 1.3 years for ADAS-Cog. Examples for the predictions are shown in Fig. B.6 in the appendix.

The prediction of the ADAS-Cog trajectories for HC-$A\beta+$ in the first year of baseline, i.e. reconstruction of the trajectory within the interval used for calculating the predictions, gives very good results, similar to the in-sample prediction for the full dataset. The average predictive accuracy in the first study period was 0.76 (standard dev.: 0.51). The prediction of ADAS-Cog in the second study period poses a much greater challenge. In this second period, the mean RMSE was 5.24 (standard dev.: 6.71). Here, the best and intermediate predictions in terms of RMSE were obtained for subjects where the additional measurements are consistent with the trend of data in the first year since baseline. Subjects with the worst predictions are seen to have a higher variation and structural breaks in ADAS-Cog between the two periods, which makes it harder to predict them.

4.4. Discussion

4.4.1. Whole Sample MFPCA

The whole sample MFPCA shows that the method is able to give interpretable results that agree with medical knowledge. The results are in line with those of Section 3.5 in the previous chapter. The first principal component, which explains by far the most of the variance in the data (93.7%), can be interpreted as an AD related effect. It shows reduced global

cognition with respect to the global average, which becomes worse during the study, associated with below-average tracer uptake in the precuneus and lateral temporo-parietal brain regions at baseline. The resulting scores showed a significant difference between healthy controls (HC-$A\beta-$) compared to MCI-$A\beta+$ and AD-$A\beta+$. The in-sample reconstructions resulted in an average RMSE of 1.56, meaning that the data can be precisely reconstructed using the MFPCA results. For the calculation of the reconstructions, MFPCA was rerun with transformed data, retransforming the results back to the original scale. While this ensures that the reconstructed trajectories stay within the possible range of ADAS-Cog (0 – 70), the principal components and scores from the transformed data are less interpretable, which, however, is not crucial if one aims only at the reconstruction.

4.4.2. Out-of-Sample Prediction

The newly developed method for calculating principal component scores for out-of-sample prediction of ADAS-Cog was applied to HC-$A\beta+$ using MCI-$A\beta+$ as the reference group. Based on the MFPCA results for MCI-$A\beta+$ combined with the observed FDG-PET scans at baseline and a short follow-up of ADAS-Cog of 1.3 years for HC-$A\beta+$, we obtained scores for all HC-$A\beta+$ subjects. Predictions were then calculated on the basis of these scores. The results show that while the reconstruction of the observed ADAS-Cog values in the first 1.3 years since follow-up gave excellent results with an average RMSE of 0.75, the prediction of the second study period from 1.3 years since baseline until the end of the study poses a much greater challenge. Here the average RMSE was 5.24.

4.4.3. Strengths and Limitations

MFPCA is a new method that allows to model temporal profiles and images together by taking covariation between these quite different types of data into account. Up to now, methods for MFPCA were restricted to multivariate functional data on a common one-dimensional domain and therefore not applicable to images. By contrast, our newly proposed and theoretically founded method can include 2D and even 3D images, as in this analysis, in a very flexible way. In particular, this means that the images enter the analysis as a whole and not in a pixelwise manner. The results give insights into the joint variation of ADAS-Cog over time and FDG-PET at baseline. An open source software implementation is available that allows an efficient calculation of the MFPCA and convenient utility functions to handle the data and display the results.

A limitation is that images up to now have to be defined as rectangles or cubes. For FDG-PET this means that the data has many zero values outside the brain that increase the memory load and the computation time. Moreover, the cosine basis expansion used in the

analysis can lead to artifacts, producing non-zero principal component values outside the brain area. The results of the analysis, however, indicate that this has only marginal influence on the estimates, while the FDG-PET element of the leading two principal components was significantly different from zero in nearly all pixels within the brain. Another point is that the algorithm used for the calculation of the scores in the out-of-sample prediction requires at least two observations of ADAS-Cog for each subject, as the data is projected on the univariate bases, which consists of the leading two principal components in the case of ADAS-Cog. For this reason, a minimum of three visits was required as one of the inclusion criteria. This means that the sample might not be completely representative, if subjects with less than three visits differ systematically from the rest. A calculation of the scores based on a mixed-models approach along the lines of the PACE approach proposed by Yao et al. (2005) did not give satisfactory results (not shown here). Future research might aim at a direct prediction method that can use e.g. only FDG-PET and ADAS-Cog at baseline.

In addition, the prediction based on the Karhunen-Loève Theorem does not allow for extrapolation in the sense that the maximal time frame for the prediction of ADAS-Cog values is the maximal follow-up time to which subjects have been observed. For the data used in this analysis, this means that one can calculate predictions until roughly 5.5 years after baseline. Given that the average follow-up in the data set is 2.92 years, this means that a reconstructed trajectory can be used for the prediction of ADAS-Cog from the last measurement until the end of the study period, which is around 2.5 years per subject on average.

4.4.4. Conclusion

Overall, MFPCA is a valuable method for the combined analysis of ADAS-Cog trajectories and FDG-PET scans that gives insights into patterns of joint variation. Moreover, it can be used to calculate out-of-sample predictions of ADAS-Cog trajectories given the FDG-PET scans at baseline together with a short follow-up of ADAS-Cog. The principal component scores represent individual weights for each subject with respect to each principal component and allow thus a very parsimonious representation of this complex kind of data. They might for example be helpful for the early identification of subjects who have an increased risk of converting from MCI to AD. In future research it might be interesting to investigate this relationship.

5. Object-Oriented Software for Functional Data

This chapter is a reprint of:

C. Happ (2017). "Object-Oriented Software for Functional Data". arXiv: 1707.02129

The article presents the following software packages:

C. Happ (2017a). *funData: An S4 Class for Functional Data*. R package version 1.1

C. Happ (2017b). *MFPCA: Multivariate Functional Principal Component Analysis for Data Observed on Different Dimensional Domains*. R package version 1.1

Abstract:

This paper introduces the **funData R** package as an object-oriented implementation of functional data. It implements a unified framework for dense univariate and multivariate functional data on one- and higher dimensional domains as well as for irregularly sampled functional data. The aim of this package is to provide a user-friendly, self-contained core toolbox for functional data, including important functionalities for creating, accessing and modifying functional data objects, that can serve as a basis for other packages. The package further contains a full simulation toolbox, which is a useful feature when implementing and testing new methodological developments.

Based on the theory of object-oriented data analysis, it is shown why it is natural to implement functional data in an object-oriented manner. The classes and methods provided by **funData** are illustrated in many examples using two freely available datasets. The **MFPCA** package, which implements multivariate functional principal component analysis, is presented as an example for an advanced methodological package that uses the **funData** package as a basis, including a case study with real data. Both packages are publicly available on GitHub and CRAN.

5.1. Introduction

Functional data analysis is a branch of modern statistics that has seen a rapid growth in recent years. The technical progress in many fields of application allows to collect data in increasingly fine resolution, e.g. over time or space, such that the observed datapoints form quasi-continuous, possibly noisy, samples of smooth functions and are thus called *functional data*. One central aspect of functional data analysis is that the focus of the analysis is not a single data point, but the entirety of all datapoints that are considered to stem from the same curve. Researchers in functional data analysis have developed many new statistical methods for the analysis of this type of data, linking the concept of functional data also to related branches of statistics, such as the study of longitudinal data, which can be seen as sparse and often also irregular samples of smooth functions, or image data, that can be represented as functions on two-dimensional domains. New approaches focus on even more generalized functional objects (*next generation* functional data analysis, Wang, Chiou, et al., 2016).

When it comes to the practical application of new methods to real data, appropriate software solutions are needed to represent functional data in an adequate manner and ideally in a way that new theoretical developments can be implemented easily. The most widely used **R** package for functional data is **fda** (Ramsay, Wickham, et al., 2014), which is related to the popular textbook of Ramsay and Silverman (2005). There are many other **R** packages for functional data that build on it, e.g. **Funclustering** (Soueidatt, 2014) or **funFEM** (Bouveyron, 2015) or **funHDDC** (Bouveyron and Jacques, 2014) or provide interfaces to **fda**, e.g. **fda.usc** (Febrero-Bande and Oviedo de la Fuente, 2012) or **refund** (Goldsmith, Scheipl, et al., 2016). The **fda** package contains a class `fd` for representing dense functional data on one-dimensional domains together with many functionalities for `fd` objects, such as plotting or summaries. It implements a variety of functional data methods, for example principal component analysis, regression models or registration. The `fd` objects represent the data as a finite linear combination of given basis functions, such as splines or Fourier bases, i.e. they store the basis functions and the individual coefficients for each curve. This representation of course works best if the underlying function is smooth and can be represented well in the chosen basis. Moreover, the data should be observed with only a small degree of noise.

Alternatively to the basis function representation, the raw, observed data can be saved directly. There are two different approaches for organizing the observations: Many packages use matrices, that contain the data in a row-wise (e.g. **fda.usc**, **refund**) or column-wise (e.g. **rainbow**, Shang and Hyndman (2016)) manner. This representation is most suitable for rather densely sampled data, where missing values can be coded via `NA`, which is supported by most of the packages. When it comes to irregular data, this way of storing functional data becomes quite inefficient, as the matrices then contain mostly missing values. Alternative

solutions for sparse data or single points in 2D are list solutions (e.g. **fdapace**, Dai et al. (2017)) or `data.frame` based versions containing the data in a long format (e.g. **fpca**, Peng and Paul (2011), **fdaPDE**, Lila et al. (2016) or **sparseFLMM**, Cederbaum (2017)). Some packages also accept different formats (**funcy**, Yassouridis (2017) or **FDboost**, Brockhaus and Ruegamer (2017)).

Some packages also support image data, i.e. functions on two-dimensional domains (**refund** and **refund.wave**, Huo et al. (2014) or **fdasrvf**, Tucker (2017)). Some others, as e.g. **fda** and **fda.usc** implement image objects, but use them rather for representing covariance or coefficient surfaces from function-on-function regression than for storing data in form of images. The majority of the **R** packages for functional data, however, are restricted to single functions on one-dimensional domains.

Methods for multivariate functional data, consisting of more than one function per observation unit, have also become relevant in recent years. The **roahd** package (Tarabelloni et al., 2017) provides a special class for this type of data, while some others simply combine the data from the different functions in a list (e.g. **fda.usc**, **Funclustering** or **RFgroove**, Gregorutti (2016)). For all of these packages, the elements of the multivariate functional data must be observed on one-dimensional domains, which means that combinations of functions and images for example are not supported. In addition, the one-dimensional observation grid must be the same for most of the implementations.

In summary, there exist already several software solutions for functional data, but there is still need for a unified, flexible representation of functional data, univariate and multivariate, on one- and higher dimensional domains and for dense and sparse functional data. The **funData** package (Happ, 2017a), which is in the main focus of this article, attempts to fill this gap. It provides a unified framework to represent all these different types of functional data together with utility methods for handling the data objects. In order to take account of the particular structure of functional data, the implementation is organized in an object-oriented manner. In this way, a link is established between the broad methodological field of object-oriented data analysis (Wang and Marron, 2007), in which functional data analysis forms an important special case, and object-oriented programming (e.g. Meyer, 1988), which is a fundamental concept in modern software engineering. It is shown why it is natural and reasonable to combine these two concepts for representing functional data. In contrast to most **R** packages mentioned above, the **funData** package is not related to a certain type of methodology, such as regression, clustering or principal component analysis. Instead, it aims at providing a flexible and user-friendly core toolbox for functional data, which can serve as a basis for other packages, similarly to the **Matrix** package for linear algebra calculations for matrices (Bates and Maechler, 2017). It further contains a complete simulation toolbox for generating functional data objects, which is fundamental for testing new functional data methods.

The **MFPCA** package (Happ, 2017b), which is also presented in this article, is an example of a package that depends on **funData**. It implements a new methodological approach – multivariate functional principal component analysis for data on potentially different dimensional domains (Happ and Greven, 2017+) – that allows to calculate principal components and individual score values for e.g. functions and images, taking covariation between the elements into account. All implementation aspects that relate to functional data, i.e. input data, output data and all calculation steps involving functions are implemented using the object-oriented functionalities of the **funData** package. Both packages are publicly available on GitHub (`https://github.com/ClaraHapp`) and CRAN (`https://CRAN.R-project.org`).

The structure of this article is as follows: Section 5.2 contains a short introduction to the concept of object orientation in statistics and computer science and discusses how to adequately represent functional data in terms of software objects. The next section presents the object-oriented implementation of functional data in the **funData** package. Section 5.4 introduces the **MFPCA** package as an example on how to use the **funData** package for the implementation of new functional data methods. The final section 5.5 contains a discussion and an outlook to potential future extensions.

5.2. Object Orientation and Functional Data

Concepts of object orientation exist both in computer science and statistics. In statistics, the term *object-oriented data analysis* (OODA) has been introduced by Wang and Marron (2007). They define it as "the statistical analysis of complex objects" and draw their attention on what they call the *atom* of the analysis. While in many parts of statistics these atoms are numbers or vectors (multivariate analysis), Wang and Marron (2007) argue that they can be much more complex objects such as images, shapes, graphs or trees. Functional data analysis (Ramsay and Silverman, 2005) is an important special case of object-oriented data analysis, where the atoms are functions. In most cases, they can be assumed to lie in $L^2(\mathcal{T})$, the space of square integrable functions on a domain $\mathcal{T}$. This space has infinite dimension, but being a Hilbert space, its mathematical structure has many parallels to the space $\mathbb{R}^p$ of p-dimensional vectors, which allows to transfer many concepts of multivariate statistics to the functional case in a quite straightforward manner.

In computer science, *object orientation* (Booch et al., 2007; Armstrong, 2006; Meyer, 1988) is a programming paradigm which has profoundly changed the way how software is structured and developed. The key concept of object-oriented programming (OOP) is to replace the until then predominant procedural programs by computer programs made of objects, that can interact with each other and thus form, in a way, the "atoms" of the program. These objects usually consist of two blocks. First, a collection of data, which may have

different data types, such as numbers, character strings or vectors of different length and is organized in fields. Second, a collection of methods, i.e. functions for accessing and/or modifying the data and for interacting with other objects. The entirety of all objects and their interactions forms the final program. A typical example for an object-oriented programming language is **Java**. But also in many other languages, concepts of object-oriented programming have been included, for instance in **MATLAB** (The MathWorks, Inc.) and **R** (R Core Team, 2015). The taxonomy given in Armstrong (2006) identifies eight fundamental concepts of object-oriented programming (see Table 5.1). The first five concepts are related to structure. Apart from the concept of objects, that contain data and methods and the concept of classes, that provide abstract designs for concrete objects, the taxonomy also lists abstraction, i.e. the fact that objects represent simplified version of reality, together with encapsulation, the principle of limited access to the data via specified methods, and inheritance, i.e. the inclusion of specialized classes into more general ones. The second group contains three concepts concerning behavior. One is the existence of methods. As discussed before, this means that the objects contain functionalities that access or modify their data. The other two concepts are message passing, i.e. objects can interact via messages that are produced and understood by the individual object methods, and polymorphism, meaning that one and the same message may be interpreted differently by the objects of different classes.

Table 5.1.: Taxonomy of object-oriented programming in Armstrong (2006).

Concepts related to structure	
Abstraction	Creating classes to simplify aspects of reality using distinctions inherent to the problem.
Class	A description of the organization and actions shared by one or more similar objects.
Encapsulation	Designing classes and objects to restrict access to the data and behavior by defining a limited set of messages that an object can receive.
Inheritance	The data and behavior of one class is included in or used as the basis for another class.
Object	An individual, identifiable item, either real or abstract, which contains data about itself and the descriptions of its manipulations of the data.
Concepts related to behavior	
Message Passing	An object sends data to another object or asks another object to invoke a method.
Method	A way to access, set or manipulate an object's information.
Polymorphism	Different classes may respond to the same message and each implements it appropriately.

The main idea of the **funData** package is to combine the concepts of object orientation that exist in computer science and in statistics for the representation of functional data. The atom of the statistical analysis should thus be represented by the "atom" of the software program. The package therefore provides classes to organize the observed data in an appropriate manner. The class methods implement functionalities for accessing and modifying the data and for interaction between objects. As functional data objects represent mathematical structures, interaction primarily means mathematical operations. The object orientation is realized in **R** via the S4 object system (Chambers, 2008). This system fulfills most of the concepts in Table 5.1 and is thus more rigorous than **R**'s widely used S3 system, which is used e.g. by **fda** or **fda.usc**. In particular, it checks for example if a given set of observation (time) points matches the observed data before constructing the functional data object.

For the theoretical analysis of functional data, the functions are mostly considered as elements of a function space such as $L^2(\mathcal{T})$, as discussed before. For the practical analysis, the data can of course only be obtained in finite resolution. Data with functional features therefore will always come in pairs of the form (t_{ij}, x_{ij}) with

$$x_{ij} = x_i(t_{ij}), \quad j = 1, \dots, S_i, \ i = 1, \dots, N$$

for some functions $x_1, \dots, x_N$ that are considered as realizations of a random process $X : \mathcal{T} \to \mathbb{R}$. The domain $\mathcal{T} \subset \mathbb{R}^d$ here is assumed to be a compact set with finite (Lebesgue-) measure and in most cases, the dimension d will be equal to 1 (functions on one-dimensional domains) sometimes also 2 (images) or 3 (3D images). The observation points $t_{ij} \in \mathcal{T}$ in general can differ in their number and location between the individual functions.

When implementing functional data in an object-oriented way, it is thus natural to collect the data in two fields: the observation points $\{(t_{i1}, \dots, t_{iS_i}) \colon \ i = 1, \dots, N\}$ on one hand and the set of observed values $\{(x_{i1}, \dots, x_{iS_i}) \colon \ i = 1, \dots, N\}$ on the other hand. Both fields form the data block of the functional data object as an inseparable entity. This is a major advantage compared to non object-oriented implementations, that can consider the observation points and the observed values as parameters in their methods, but can not map the intrinsic dependency between both of them.

In the important special case that the functions are observed on a one-dimensional domain and that the arguments do not differ across functions, they can be collected in a single vector $(t_1, \dots, t_S)$ and the observed values can be stored in a matrix X with entries $x_{ij}, \ i = 1, \dots, N, \ j = 1, \dots, S$. The matrix-based concept can be generalized to data observed on common grids on higher dimensional domains. In this case, the observation grid can be represented as a matrix (or array) or, in the case of a regular and rectangular grid, as a collection of vectors that define the marginals of the observation grid. The observed data is collected in an array with three or even more dimensions.

In recent years, the study of multivariate functional data that takes multiple functions at the same time into account, has led to new findings. Each observation unit hence consists of a fixed number of functions p, that can also differ in their domain (e.g. functions and images, Happ and Greven, 2017+). Technically, the observed values are assumed to stem from a random process $X = (X^{(1)}, \ldots, X^{(p)})$, with random functions $X^{(k)} \colon \mathcal{T}_k \to \mathbb{R},\ \mathcal{T}_k \in \mathbb{R}^{d_k},\ k = 1, \ldots, p$, that we call the *elements* of X. Realizations $x_1, \ldots, x_N$ of such a process all have the same structure as X. If for example $p = 2$ and $d_1 = 1,\ d_2 = 2$, the realizations will all be bivariate functions with one functional and one image element. As data can only be obtained in finite resolution, observed multivariate functional data is of the form

$$(t_{ij}^{(k)},\ x_{ij}^{(k)}) \quad j = 1, \ldots, S_i^{(k)},\ i = 1, \ldots, N,\ k = 1, \ldots, p.$$

Each element thus can be represented separately by its observation points and the observed values, and the full multivariate sample constitutes the collection of all the p elements.

5.3. The funData Package

The **funData** package implements the object-oriented approach for representing functional data in **R**. It provides three classes for functional data on one- and higher dimensional domains, multivariate functional data and irregularly sampled data, which are presented in Section 5.3.1. The second Section 5.3.2 presents the methods associated with the functional data classes based on two example datasets.

5.3.1. Three Classes for Functional Data

For the representation of functional data in terms of abstract classes – which, in turn, define concrete objects – we distinguish three different cases.

1. Class `funData` for dense functional data of "arbitrary" dimension (in most cases the dimension of the domain will is $d \in \{1, 2, 3\}$) on a common set of observation points $t_1, \ldots, t_S$ for all curves. The curves may have missing values coded by `NA`.

2. Class `irregFunData` for irregularly sampled functional data with individual sampling points $t_{ij},\ j = 1, \ldots, S_i,\ i = 1, \ldots, N$ for all curves. The number S_i and the location of observation points can vary across individual observations. At the moment, only data on one-dimensional domains is implemented.

3. Class `multiFunData` for multivariate functional data, which combines p elements of functional data that may be defined on different dimensional domains (e.g. functions and images).

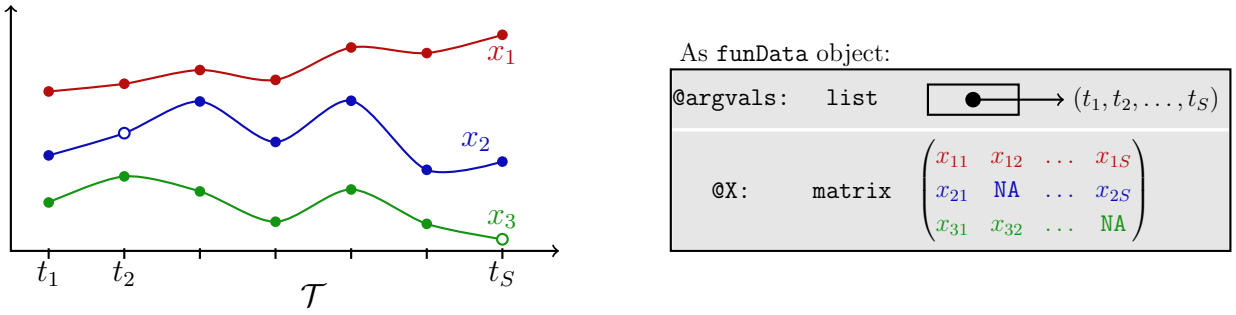


Figure 5.1.: Left: An example of $N = 3$ observations of functional data on a one-dimensional domain $\mathcal{T}$, observed on a common discrete grid $(t_1, \ldots, t_S)$, where the observed values $x_{ij} = x_i(t_j)$ are represented by solid circles. The functions x_2 and x_3 have one missing value, each (open circles). Right: Representation of the data in a `funData` object. The `@argvals` slot is a list of length one, containing the observation grid as a vector. The `@X` slot is a matrix of dimension $N \times S$ that contains the observed values in row-wise format. Missing values are coded with `NA`.

In the case of data on one-dimensional domains, the boundaries between the `funData` and the `irregFunData` class may of course be blurred in practice. The conceptual difference is that in 1. all curves are ideally supposed to be sampled on the full grid $T = \{t_1, \ldots, t_S\} \subset \mathcal{T}$ and differences in the number of observation points per curve are mainly driven by anomalies or errors in the sampling process, such as missing values, which can be coded by `NA`. In contrast, case 2. expects a priori that the curves can be observed at different observation points t_{ij}, and that the number of observations per curve may vary.

For `funData` and `irregFunData`, the data is organized in two fields or *slots*, as they are called for S4 classes (Chambers, 2008): The slot `@argvals` contains the observation points and the slot `@X` contains the observed data. For `funData`, the `@argvals` slot is a list, containing the common sampling grid for all functions and `@X` is an array containing all observations. In the simplest case of functions defined on a one-dimensional domain and sampled on a grid with S observation points, `@argvals` is a list of length one, containing a vector of length S and `@X` is a matrix of dimension $N \times S$, containing the observed values for each curve in a row-wise manner. For an illustration, see Fig. 5.1. If the `funData` object is supposed to represent N images with $S_x \times S_y$ pixels (i.e. S_x pixels long and S_y pixels high), `@argvals` is a list of length 2, containing two vectors with S_x and S_y entries, respectively, that represent the sampling grid. The slot `@X` is an array of dimension $N \times S_x \times S_y$, cf. Fig. 5.2. For the `irregFunData` class, only functions on one-dimensional domains are currently implemented. The `@argvals` slot here is a list of length N, containing in its i-th entry the vector $(t_{i1}, \ldots, t_{iS_i})$ with all observation points for the i-th curve. The `@X` slot organizes the observed values analogously, i.e. it is also a list of length N with the i-th entry containing the vector $(x_{i1}, \ldots, x_{iS_i})$. An illustration is given in Fig. 5.3. The `multiFunData` class, finally, represents multivariate functional data with p elements. An object of this class is simply a list of p `funData` objects, representing the different elements.

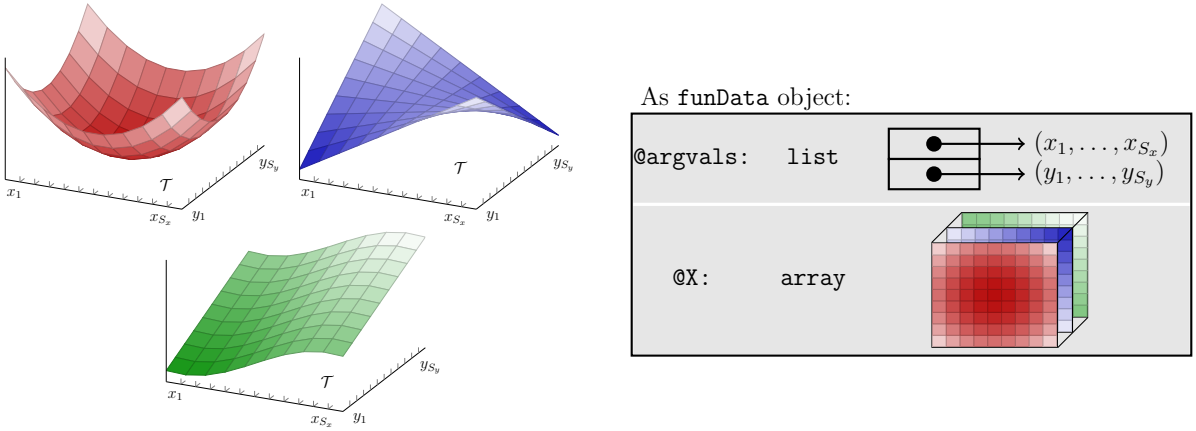


Figure 5.2.: Left: An example of $N = 3$ observations of functional data on a two-dimensional domain $\mathcal{T}$. The functions are observed on a common discrete grid having S_x points in x- and S_y points in y-direction, i.e. each observation forms an image with $S_x \times S_y$ pixels. Right: Representation of the data in a `funData` object. The `@argvals` slot is a list of length 2, containing the marginal sampling points. The slot `@X` is an array of dimension $N \times S_x \times S_y$.

For an illustration, see Fig. 5.4. Given specific data, the realizations of such classes are called `funData`, `irregFunData` or `multiFunData` objects. We will use the term *functional data object* in the following for referring to objects of all three classes.

5.3.2. Methods for Functional Data Objects

Essential methods for functional data objects include the creation of an object from the observed data, methods for modifying and extracting the data, plotting, arithmetics and other functionalities. The methods in the **funData** package are implemented such that they work on functional data objects as the atoms of the program, i.e. the methods accept functional data objects as input and/or have functional data objects as output. Moreover, all functions are implemented for the three different classes with appropriate sub-methods. This corresponds to the principle of polymorphism in Armstrong (2006), as different classes have their own implementation e.g. of a plot function. Using the S4 framework (Chambers, 2008), this is achieved by defining abstract, so-called *generic functions* and implementing associated *methods* for each class. In most cases, the methods for `multiFunData` objects will simply call the corresponding method for each element and concatenate the results appropriately.

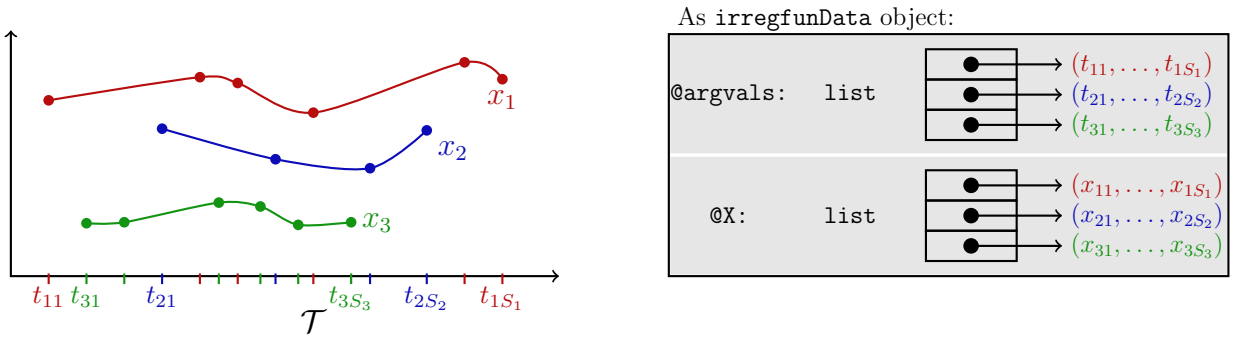


Figure 5.3.: Left: An example of $N = 3$ irregular observations of functional data on a one-dimensional domain $\mathcal{T}$. The observation points for each function differ in number and location. Right: Representation of the data in an `irregFunData` object. Both the `@argvals` and the `@X` slot are a list of length N, containing the observation points t_{ij} and the observed values x_{ij}.

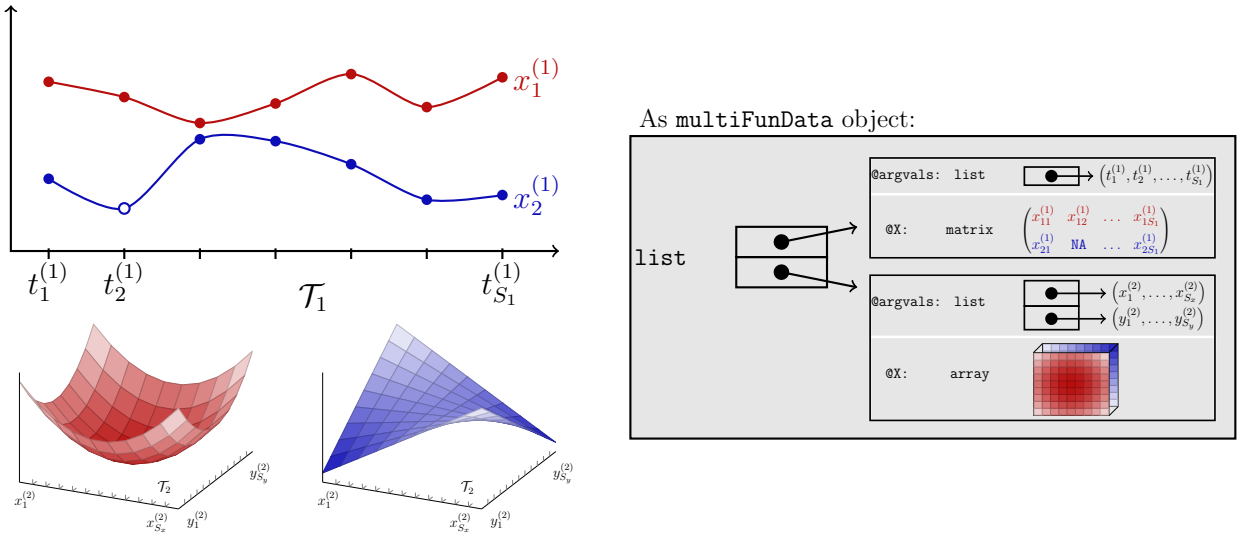


Figure 5.4.: Left: An example of $N = 2$ observations of bivariate functional data on different domains, i.e. each observation (red/blue) consists of two elements, a curve and an image. Right: Representation of the data as a `multiFunData` object. As the data is bivariate, the `multiFunData` object is a list of length 2, containing the two elements as `funData` objects.

Data Used in the Examples

The code examples use the Canadian weather data (Ramsay and Silverman, 2005), that are available e.g. in the **fda** package and the CD4 cell count data (Goldsmith, Greven, et al., 2013) from the **refund** package. In both cases, the data is observed on a one-dimensional domain. Examples for image data are included in the description of the simulation toolbox at the end of this section.

The Canadian weather data contains daily and monthly observations of temperature (in °C) and precipitation (in mm) for $N = 35$ Canadian weather stations, averaged over the years 1960 to 1994. We will use the daily temperature as an example for dense functional data on a one-dimensional domain. Moreover, the monthly precipitation data will be used together with the daily temperature to construct multivariate functional data with elements on different domains ($\mathcal{T}_1 = [1, 365]$ for the temperature and $\mathcal{T}_2 = [1, 12]$ for the precipitation).

The CD4 cell count data reports the number of CD4 cells per milliliter of blood for $N = 366$ subjects who participated in a study on AIDS (MACS, Multicenter AIDS Cohort Study). CD4 cells are part of the human immune system and are attacked in the case of an infection with HIV. Their number thus can be interpreted as a proxy for the disease progression. For the present data, the CD4 counts were measured roughly twice a year and centered at the time of seroconversion, i.e. the time point when HIV becomes detectable. In total, the number of observations for each subject varies between 1 and 11 in the period of 18 months before and 42 months after seroconversion. The individual time points do also differ between subjects. The dataset thus serves as an example for irregular functional data. For more information on the data, please see Goldsmith, Greven, et al. (2013).

Creating New Objects and Accessing an Object's Information

The following code creates `funData` objects for the Canadian temperature and precipitation data:

```
R> dailyTemp <- funData(argvals = 1:365,
                        X = t(CanadianWeather$dailyAv[,, 'Temperature.C']))
R> monthlyPrec <- funData(argvals = 1:12,
                          X = t(CanadianWeather$monthlyPrecip))
```

It is then very easy to create a bivariate `multiFunData` object, containing the daily temperature and the monthly precipitation for the 35 weather stations:

```
R> canadWeather <- multiFunData(dailyTemp, monthlyPrec)
```

The `cd4` data in the **refund** package is stored in a `data.frame` with 366×61 entries, containing the CD4 counts for each patient on the common grid of all sampling points.

Missing values are coded as `NA`. Since each patient has at least 1 and at most 11 observations, more than 90% of the dataset consists of missings. The `irregFunData` class stores only the observed values and their time points and is therefore more parsimonious. The following code extracts both separately as lists and then constructs an `irregFunData` object:

```
R> allArgvals <- seq(-18, 42)
R> argvalsList <- apply(cd4, 1, function(x){allArgvals[complete.cases(x)]})
R> obsList <- apply(cd4, 1, function(x){x[complete.cases(x)]})
R> cd4Counts <- irregFunData(argvals = argvalsList, X = obsList)
```

When a functional data object is called in the command line, some basic information is printed to standard output. For the `funData` object containing the Canadian temperature data:

```
R> dailyTemp

Functional data object with 35 observations of 1 - dimensional support
argvals:
        1 2 ... 365                (365 sampling points)
X:
        array of size 35 x 365
```

The `multiFunData` version lists the information of the different elements:

```
R> canadWeather

An object of class "multiFunData"
[[1]]
Functional data object with 35 observations of 1 - dimensional support
argvals:
        1 2 ... 365                (365 sampling points)
X:
        array of size 35 x 365

[[2]]
Functional data object with 35 observations of 1 - dimensional support
argvals:
        1 2 ... 12                (12 sampling points)
X:
        array of size 35 x 12
```

For `irregFunData` objects there is some additional information about the total number of observations:

```
R> cd4Counts

Irregular functional data object with 366 observations of 1 - dimensional support
argvals:
```

```
        Values in -18 ... 42.
X:
        Values in 10 ... 3184.
Total:
        1888 observations on 60 different x-values (1 - 11 per observation).
```

The slots can be accessed directly via `@argvals` or `@X`. The preferable way of accessing and modifying the data in the slots, however, is via the get/set methods, following the principle of limited access (encapsulation, see Table 5.1), as an example:

```
R> getArgvals(monthlyPrec)

[[1]]
 [1]  1  2  3  4  5  6  7  8  9 10 11 12
```

The method `nObs` returns the number of observations (functions) in each object:

```
R> nObs(dailyTemp)

[1] 35

R> nObs(cd4Counts)

[1] 366

R> nObs(canadWeather)

[1] 35
```

The number of observation points is given by `nObsPoints`. Note that `dailyTemp` and `canadWeather` are densely sampled and therefore return a single number or two numbers (one for each element). For the irregularly sampled data in `cd4Counts`, the method returns a vector of length $N = 366$, containing the individual number of observations for each subject:

```
R> nObsPoints(dailyTemp)

[1] 365

R> nObsPoints(cd4Counts)

 [1] 3 4 8 4 8 3 4 7 2 6 8 3
 ... [output truncated] ...

R> nObsPoints(canadWeather) # 365 (temp.) and 12 (prec.) observation points
```

```
[[1]]
[1] 365

[[2]]
[1] 12
```

The dimension of the domain can be obtained by the `dimSupp` method:

```
R> dimSupp(dailyTemp)

[1] 1

R> dimSupp(cd4Counts)

[1] 1

R> dimSupp(canadWeather) # both elements have one-dimensional support

[1] 1 1
```

Finally, a subset of the data can be extracted using the function `extractObs`. We can for example extract the temperature data for the first five weather stations:

```
R> extractObs(dailyTemp, obs = 1:5)

Functional data object with 5 observations of 1 - dimensional support
argvals:
        1 2 ... 365                (365 sampling points)
X:
        array of size 5 x 365
```

or the CD4 counts of the first 8 patients before seroconversion (i.e. until time 0):

```
R> extractObs(cd4Counts, obs = 1:8, argvals = -18:0)

Irregular functional data object with 8 observations of 1 - dimensional support
argvals:
        Values in -17 ... -3.
X:
        Values in 429 ... 1454.
Total:
        15 observations on 6 different x-values (1 - 3 per observation).
```

In both cases, the method returns an object of the same class as the argument with which the function was called (`funData` for `dailyTemp` and `irregFunData` for `cd4Counts`), which is seen by the output.

Plotting

The more complex the data, the more important it is to have adequate visualization methods. The **funData** package comes with two plot methods for each class, one based on **R**'s standard plotting engine (`plot.default` and `matplot`) and one based on the `ggplot2` implementation of the grammar of graphics (package **ggplot2**, Wickham (2009) and Wickham et al. (2016)). The `plot` function inherits all parameters from the `plot.default` function from the **graphics** package, i.e. colors, axis labels and many other options can be set as for the usual `plot.default` function. The following code plots all 35 curves of the Canadian temperature data:

```
R> plot(dailyTemp, main = "Daily Temperature Data",
        xlab = "Day of Year", ylab = "Temperature in °C")
```

and the CD4 counts of the first five patients on the log-scale:

```
R> plot(cd4Counts, obs = 1:5, xlim = c(-18, 45), log = "y",
        main = "CD4 Counts for Individuals 1-5",
        xlab = "Month since seroconversion",
        ylab = "CD4 cell count (log-scale)")
R> legend("topright", legend = 1:5, col = rainbow(5), lty = 1, pch = 20,
          title = "Individual")
```

For multivariate functional data, the different elements are plotted side by side, as shown here for the last ten Canadian weather stations:

```
R> plot(canadWeather, obs = 26:35, lwd = 2, log = c("", "y"),
        main = c("Temperature", "Precipitation (log-scale)"),
        xlab = c("Day of Year", "Month"),
        ylab = c("Temperature in °C", "Precipitation in mm"))
```

The resulting plots are shown in Fig. 5.5.

The optional `ggplot` function creates a `ggplot` object that can be further modified by the user by loading the **ggplot2** package and using the functionality provided therein. For plotting multivariate functional data, the **gridExtra** package (Auguie, 2016) is used to order plots of all elements side by side. The following codes produce analogous plots to the `plot` examples above for the Canadian temperature data:

```
R> library(ggplot2)
R>
R> tempPlot <- funData::ggplot(dailyTemp)
R> tempPlot + labs(title = "Daily Temperature Data",
                   x = "Day of Year", y = "Temperature in °C")
```

for the CD4 counts:

```
R> cd4Plot <- funData::ggplot(cd4Counts, obs = 1:5)
R> cd4Plot + geom_line(aes(colour = obsInd)) +
     labs(title = "CD4 Counts for Individuals 1-5", color = "Individual",
          x = "Month since seroconversion", y = "CD4 cell count (log-scale)") +
     scale_y_log10(breaks = seq(200,1000,200))
```

and for the bivariate Canadian weather data:

```
R> weatherPlot <- funData::ggplot(canadWeather, obs = 26:35)
R>
R> # customize the univariate plots
R> weatherPlot[[1]] <- weatherPlot[[1]] + geom_line(aes(colour = obsInd)) +
     labs(title = "Temperature", colour = "Weather Station",
          x = "Day of Year", y = "Temperature in °C")
R>
R> weatherPlot[[2]] <- weatherPlot[[2]] + geom_line(aes(colour = obsInd)) +
     labs(title = "Precipitation (log-scale)", colour = "Weather Station",
          x = "Month", y = "Precipitation in mm") +
     scale_x_continuous(breaks = 1:12) +
     scale_y_log10(breaks = c(0.1,0.5,1,5,10))
R>
R> # plot all elements of the multivariate functional data
R> gridExtra::grid.arrange(grobs = weatherPlot, nrow = 1)
```

The corresponding plots are shown in Fig. 5.6.

Coercion

As discussed earlier, there is no clear boundary between the `irregFunData` class and the `funData` class for functions on one-dimensional domains. The package thus provides coercion methods to coerce `funData` objects to `irregFunData` objects, as seen in the output:

```
R> as.irregFunData(dailyTemp)

Irregular functional data object with 35 observations of 1 - dimensional support
argvals:
        Values in 1 ... 365.
X:
        Values in -34.8 ... 22.8.
Total:
        12775 observations on 365 different x-values (365 - 365 per observation).
```

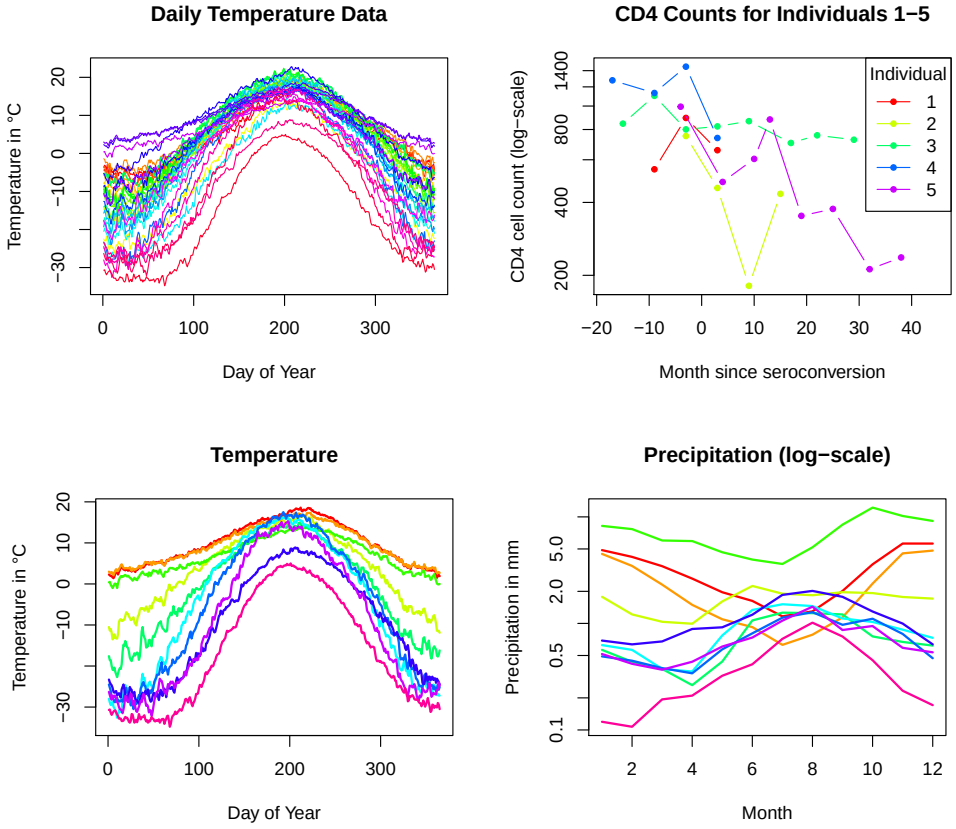


Figure 5.5.: Results of the `plot` commands for functional data objects. First row: The daily temperature in 35 Canadian weather stations (`funData` object, left) and the CD4 counts for the first five individuals (`irregFunData` object, right). Second row: The Canadian weather data for ten weather stations (`multiFunData` object). See text for the commands used; all other options were kept as defaults.

and vice versa, using the union of all observation points of all subjects as the common one and coding missing values with `NA`:

```
R> as.funData(cd4Counts)

Functional data object with 366 observations of 1 - dimensional support
argvals:
        -18 -17 ... 42                (60 sampling points)
X:
        array of size 366 x 60
```

Further, `funData` objects can also be coerced to `multiFunData` objects with only one element:

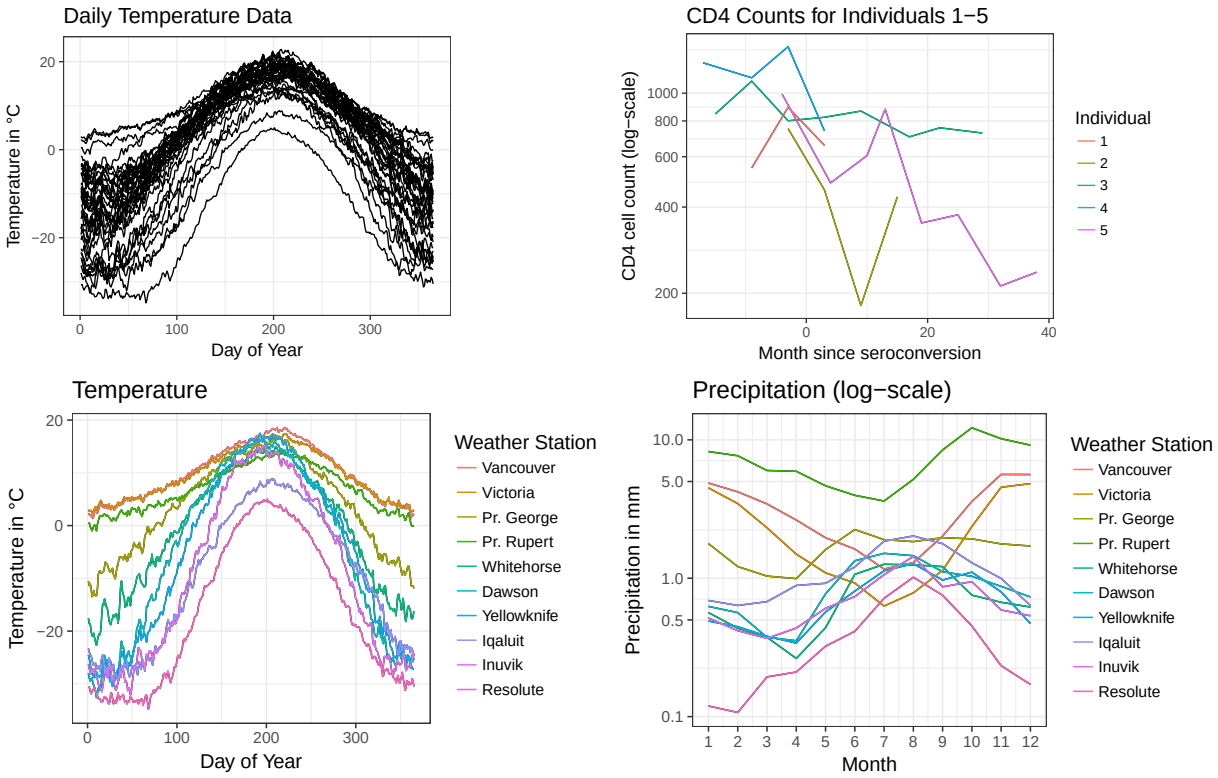


Figure 5.6.: Results of the `ggplot` commands for functional data objects. First row: The daily temperature in 35 Canadian weather stations (`funData` object, left) and the CD4 counts for the first five individuals (`irregFunData` object, right). Second row: The Canadian weather data for ten weather stations (`multiFunData` object). See text for the commands used. For all plots the option `theme_bw()` has been added for optimal print results; all other parameters were kept as defaults.

```
R> as.multiFunData(dailyTemp)

An object of class "multiFunData"
[[1]]
Functional data object with 35 observations of 1 - dimensional support
argvals:
        1 2 ... 365                (365 sampling points)
X:
        array of size 35 x 365
```

Mathematical Operations for Functional Data Objects

With the **funData** package, mathematical operations such as sums, products, calculating the absolute value or the logarithm can directly be applied to functional data objects, with

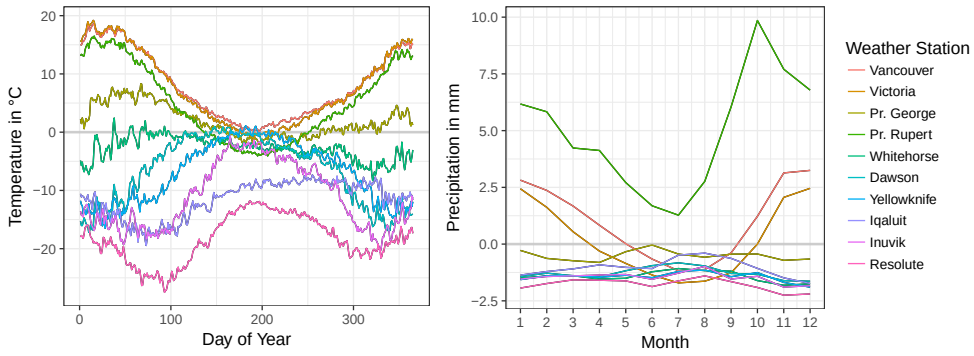


Figure 5.7.: Demeaned versions of the ten `canadianWeather` observations shown in Fig. 5.6. The curves have been obtained by `canadWeather - meanFunction(canadWeather)`, i.e. the bivariate mean function of all 35 weather stations has been subtracted from each observation. The horizontal gray lines mark zero, corresponding to the original mean function.

the calculation made pointwise and the return being again an object of the same class. The operations build on the `Arith` and `Math` *group generics* for S4 classes. We can for example convert the Canadian temperature data from Celsius to Fahrenheit:

```
R> 9/5 * dailyTemp + 32
```

or calculate the logarithms of the CD4 count data:

```
R> log(cd4Counts)
```

Arithmetic operations such as sums or products are implemented for scalars and functional data objects as well as for two functional data objects. Note that in the last case, the functional data objects must have the same number of observations (in this case, the calculation is done with the i-th function of the first object and the i-th function of the second object) or one object may have only one observation (in this case, the calculation is made with each function of the other object). This is particularly useful e.g. for subtracting a common mean from all functions in an object as in the following example, using the special `meanFunction` method:

```
R> canadWeather - meanFunction(canadWeather)
```

Some of the demeaned curves are shown in Fig. 5.7. Note that the functions also need to have the same observation points, which is especially important for `irregFunData` objects.

The `tensorProduct` function allows to calculate tensor products of functional data objects f_1, f_2 on one-dimensional domains $\mathcal{T}_1, \mathcal{T}_2$, respectively, i.e.

$$f_{\text{Tens}}(t_1, t_2) = f_1(t_1) f_2(t_2) \quad t_1 \in \mathcal{T}_1, t_2 \in \mathcal{T}_2.$$

The following code calculates the tensor product of the Canadian weather data and the output shows that the result is a `funData` object on a two-dimensional domain:

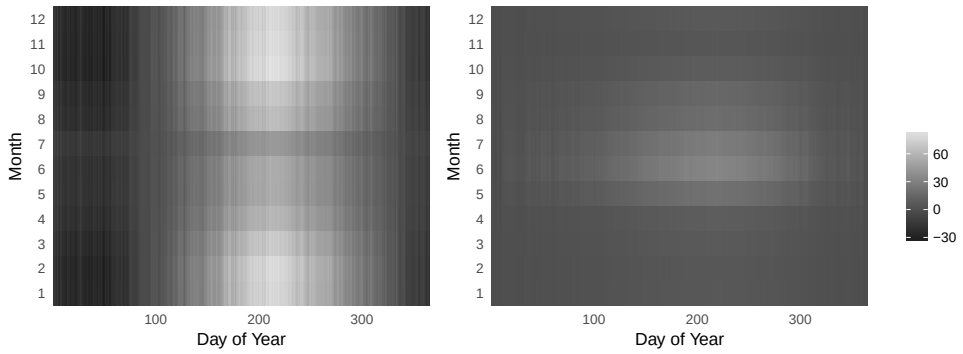


Figure 5.8.: Two observations of the tensor product of daily temperature and monthly precipitation from the Canadian weather data, calculated via `tensorProduct(dailyTemp, monthlyPrec)`. As seen in the plot, the domains of the functions have to be one-dimensional, but can be different. The result is an object of class `funData` on the two-dimensional domain $[1, 356] \times [1, 12]$ with $35^2 = 1225$ observations, from which two are shown here.

```
R> tensorData <- tensorProduct(dailyTemp, monthlyPrec)
R> tensorData

Functional data object with 1225 observations of 2 - dimensional support
argvals:
        1 2 ... 365                (365 sampling points)
        1 2 ... 12                (12 sampling points)
X:
        array of size 1225 x 365 x 12
```

Two observations in `tensorData` are shown in Fig. 5.8. Note that for image data, a single observation has to be specified for plotting.

Another important aspect when working with functional data is integration. The **funData** package implements two quadrature rules, `"midpoint"` and `"trapezoidal"` (the default). The data is always integrated over the full domain and in the case of multivariate functional data, the integrals are calculated for each element and the results are added:

```
R> integrate(dailyTemp, method = "trapezoidal")

 [1]  1715.70  2249.75  2015.30  2488.10  1916.15  1929.65
 [7] -1824.20  1145.40   834.95  1500.95  1514.00  2246.65
 ... [output truncated] ...

R> integrate(canadWeather)

 [1]  1759.7695  2292.5877  2058.6627  2525.2938  1951.8074
 [6]  1963.3119 -1799.6470  1172.5376   863.3241  1537.4718
[11]  1547.7194  2275.1507
 ... [output truncated] ...
```

For irregular data, the integral can be calculated on the observed points or they can be extrapolated linearly to the full domain. For the latter, curves with only one observation point are assumed to be constant.

Based on integrals, one defines the usual scalar product on $L^2(\mathcal{T})$

$$\langle f, g \rangle_2 = \int_{\mathcal{T}} f(t)g(t)\mathrm{d}t$$

and the induced norm $\|f\|_2 = \langle f, f \rangle_2^{1/2}$ for $f, g \in L^2(\mathcal{T})$. For multivariate functional data on domains $\mathcal{T}_1 \times \ldots \times \mathcal{T}_p$, the scalar product can be extended to

$$\langle\langle f, g \rangle\rangle = \sum_{j=1}^{p} f^{(j)}(t)g^{(j)}(t)\mathrm{d}t$$

with the induced norm $|||f||| = \langle\langle f, f \rangle\rangle^{1/2}$. The multivariate scalar product can further be generalized by introducing weights $w_j > 0$ for each element (cf. Happ and Greven, 2017+):

$$\langle\langle f, g \rangle\rangle_w = \sum_{j=1}^{p} w_j \langle f^{(j)}, g^{(j)} \rangle_2. \tag{5.1}$$

Scalar products and norms are implemented for all three classes in the **funData** package. Here also, the scalar product can be calculated for pairs of functions $f_1, \ldots, f_N$ and $g_1, \ldots, g_N$, hence $\langle f_i, g_i \rangle_2$, or for a sample $f_1, \ldots, f_N$ and a single function g, returning $\langle f_i, g \rangle_2$. The `norm` function accepts some additional arguments, such as `squared` (logical, should the squared norm be calculated), `obs` (the indices of curves for which the norm should be calculated) or `weight` (a vector containing the weights $w_1, \ldots, w_p$ for multivariate functional data):

```
R> scalarProduct(dailyTemp, meanFunction(dailyTemp))

 [1] 30737.63 38602.51 36314.96 32467.36 40324.06 43842.06
 [7] 46235.71 48665.02 48404.69 46850.19 44177.40 47596.52
 ... [output truncated] ...

R> all.equal(scalarProduct(dailyTemp, dailyTemp),
              funData::norm(dailyTemp, squared = TRUE))

[1] TRUE

R> # norm for the first 4 observations, extrapolated to full domain
R> funData::norm(cd4Counts, obs = 1:4, fullDom = TRUE)

[1]  15562439  30772035  32405620 109583314

R> # give temperature double weight
R> funData::norm(canadWeather, weight = c(2,1))
```

```
 [1]  54484.78  84524.45  74968.58  71400.87  85221.47
 [6]  96849.21 136065.38 109946.26 107856.02 103943.63
[11]  93596.27 116518.97
 ... [output truncated] ...
```

Note that there also exists a function `norm` in the **base** package and thus it is better to use `funData::norm` in this case.

Simulation Toolbox

The **funData** package comes with a full simulation toolbox for univariate and multivariate functional data, which is a very useful feature when implementing and testing new methodological developments. The data is simulated based on a truncated Karhunen-Loève representation of functional data, as for example in the simulation studies in Scheipl and Greven (2016) or Happ and Greven (2017+). All examples in the following text use `set.seed(1)` before calling a simulation function for reasons of reproducibility. Example plots are given in the appendix.

For univariate functions $x_i \colon \mathcal{T} \to \mathbb{R}$, the Karhunen-Loève representation of a function x_i truncated at $M \in \mathbb{N}$ is given by

$$x_i(t) = \mu(t) + \sum_{m=1}^{M} \xi_{i,m}\phi_m(t), \quad i = 1, \ldots, N,\ t \in \mathcal{T} \tag{5.2}$$

with a common mean function $\mu(t)$ and principal component functions $\phi_m,\ m = 1, \ldots, M$. The individual functional principal component scores $\xi_{i,m} = \langle x_i, \phi_m \rangle_2$ are realizations of random variables ξ_m with $\mathbb{E}(\xi_m) = 0$ and $\text{Var}(\xi_m) = \lambda_m$ with eigenvalues $\lambda_m \geq 0$ that decrease towards 0. This representation is valid for domains of arbitrary dimension, hence also for $\mathcal{T} \subset \mathbb{R}^2$ (images) or $\mathcal{T} \subset \mathbb{R}^3$ (3D images).

The simulation algorithm constructs new data from a system of M orthonormal eigenfunctions $\phi_1 \ldots \phi_M$ and scores $\xi_{i,m}$ according to the Karhunen-Loève representation (5.2) with $\mu(t) \equiv 0$. For the eigenfunctions, the package implements Legendre Polynomials, Fourier basis functions and eigenfunctions of the Wiener process including some variations (e.g. Fourier functions plus an orthogonalized version of the linear function). The scores are generated via

$$\xi_{i,m} \overset{\text{iid}}{\sim} \text{N}(0, \lambda_m),\ m = 1, \ldots, M,\ i = 1, \ldots, N. \tag{5.3}$$

For the eigenvalues λ_m, one can choose between a linear ($\lambda_m = \frac{M-m+1}{M}$) or exponential decrease ($\exp(-\frac{m+1}{2})$) or those of the Wiener process. New eigenfunctions and eigenvalues can be added to the package in an easy and modular manner.

The next code chunk simulates $N = 8$ curves on the one-dimensional observation grid $\{0, 0.01, 0.02, \ldots, 1\}$ based on the first $M = 10$ Fourier basis functions on $[0, 1]$ and eigenvalues with a linear decrease:

```
R> simUniv1D <- simFunData(N = 8, argvals = seq(0,1,0.01),
                           eFunType = "Fourier", eValType = "linear", M = 10)
```

The function returns a list with 3 entries: The simulated data (`simData`, a `funData` object shown in Fig. C.1 in the appendix), the true eigenvalues (`trueVals`) and eigenfunctions (`trueFuns`, also as a `funData` object).

For simulating functional data on a two- or higher dimensional domain, `simFunData` constructs eigenfunctions based on tensor products of univariate eigenfunctions. The user thus has to supply the parameters that relate to the eigenfunctions as a list (for `argvals`) or as a vector (`M` and `eFunType`), containing the information for each marginal. The total number of eigenfunctions equals to the product of the entries of M. The following example code simulates $N = 5$ functions on $\mathcal{T} = [0, 1] \times [-0.5, 0.5]$. The eigenfunctions are calculated as tensor products of $M_1 = 10$ eigenfunctions of the Wiener process on $[0, 1]$ and $M_2 = 12$ Fourier basis functions on $[-0.5, 0.5]$. In total, this leads to $M = M_1 \cdot M_2 = 120$ eigenfunctions. For each eigenfunction and each observed curve, the scores $\xi_{i,m}$ are generated as in (5.3) with linearly decreasing eigenvalues:

```
R> argvals <- list(seq(0,1,0.01), seq(-0.5,0.5, 0.01))
R> simUniv2D <- simFunData(N = 5, argvals = argvals,
           eFunType = c("Wiener", "Fourier"), eValType = "linear", M = c(10,12))
```

The first simulated image is shown in Fig. C.1 in the appendix. As for functions on one-dimensional domains, the function returns the simulated data together with the true eigenvalues and eigenfunctions.

For multivariate functional data, the simulation is based on the multivariate version of the Karhunen-Loève Theorem (Happ and Greven, 2017+) for multivariate functional data $x_i = (x_i^{(1)}, \ldots, x_i^{(p)})$ truncated at $M \in \mathbb{N}$

$$x_i(t) = \mu(t) + \sum_{m=1}^{M} \rho_{i,m} \psi_m(t), \quad i = 1, \ldots, N, \; t = (t_1, \ldots, t_p) \in \mathcal{T}_1 \times \ldots \times \mathcal{T}_p \tag{5.4}$$

with a multivariate mean function μ and multivariate eigenfunctions ψ_m that have the same structure as x_i (i.e. if x_i consists of a function and an image, for example, then μ and ψ_m will also consist of a function and an image). The individual scores $\rho_{i,m} = \langle\langle x_i, \psi_m \rangle\rangle$ for each observation x_i and each eigenfunction ψ_m are real numbers and have the same properties as in the univariate case, i.e. they are realizations of random variables ρ_m with $\mathbb{E}(\rho_m) = 0$ and $\text{Var}(\rho_m) = \nu_m$ with eigenvalues $\nu_m \geq 0$ that again form a decreasing sequence that converges towards 0. As in the univariate case, the multivariate functions

are simulated based on eigenfunctions and scores according to (5.4) with $\mu(t) \equiv 0$. The scores are sampled independently from a $\mathrm{N}(0, \nu_m)$ distribution with decreasing eigenvalues ν_m, analogously to (5.3). For the construction of multivariate eigenfunctions, Happ and Greven (2017+) propose two approaches based on univariate orthonormal systems, which are both implemented in the `simMultiFunData` function.

If all elements of the multivariate functional data are to be defined on one-dimensional domains, the multivariate eigenfunctions can be obtained by starting from orthonormal functions on a big domain, splitting them into p pieces and shifting the pieces to where the elements should be defined. As an example, consider that one would like to simulate bivariate functional data with one element defined on $\mathcal{T}_1 = [-0.5, 0.5]$ and the other on $\mathcal{T}_2 = [0, 1]$ (note that each element is defined on a one-dimensional domain, not to be confused with univariate functional data on a two-dimensional domain with margins $[-0.5, 0.5]$ and $[0, 1]$). For the orthonormal system on the big domain choose e.g. a Fourier basis on $[0, 2]$. The functions that form this big basis are now cut at $t = 1$, with the first pieces shifted to the left by 0.5 and the second pieces to the left by 1 such that the first part of a split function is defined on $[-0.5, 0.5]$ and the second part is defined on $[0, 1]$. The functions on $[-0.5, 0.5]$ then form the first element of the bivariate eigenfunctions and the functions on $[0, 1]$ form the second element, after multiplication with a random sign for each element. Orthonormality of the multivariate eigenfunctions is ensured by construction. For more technical details, see Happ and Greven (2017+).

When calling `simMultiFunData` with the option `"split"`, the user has to supply the observation points for the elements as a list, the type and number M of basis function for the big orthonormal system and of course the number N of observations to be generated and how the eigenvalues should decrease. The following code simulates $N = 7$ bivariate functions on $[-0.5, 0.5]$ and $[0, 1]$ as in the example, based on $M = 10$ Fourier basis functions and linearly decreasing eigenvalues. The parameter `type = "split"` indicates that the splitting algorithm is used for obtaining the eigenfunctions, as just described:

```
R> argvals <- list(seq(-0.5,0.5,0.01), seq(0,1,0.01))
R> simMultiSplit <- simMultiFunData(N = 7, argvals = argvals,
                        eFunType = "Fourier", eValType = "linear", M = 10,
                        type = "split")
```

The result contains the simulated data as well as the eigenfunctions and eigenvalues, as for the univariate simulation. The simulated functions are shown in Fig. C.2 in the appendix.

As an alternative, multivariate eigenfunctions can be constructed as weighted versions of univariate eigenfunctions, i.e. the elements of a multivariate eigenfunction ψ_m are given by $\psi_m^{(j)} = \alpha_j^{1/2} f_m^{(j)}$ with functions $f_m^{(j)}$ that form an orthonormal system for each $j = 1, \ldots, p$ and weights $\alpha_j \in [0, 1]$ that sum up to 1. This choice ensures orthonormality of the multivariate functions ψ_m. With this approach, one can also simulate multivariate functional data on

different dimensional domains, e.g. functions and images. It is implemented in **funData**'s `simMultiFunData` method using the option `type = "weighted"`. In this case, the user has to supply the information for all univariate eigenfunctions in form of lists, i.e. `M` (the number of univariate eigenfunctions), `eFunType` (the type of univariate eigenfunctions) and `argvals` (the domains) have all to be supplied as a list with p elements with the j-th entry specifying the information for the j-th univariate eigenfunctions. For elements on higher dimensional domains, the corresponding list entry has to be a vector (for `M` and `eFunType`) or a list (`argvals`) in order to construct higher dimensional basis functions based on tensor products. The weights are generated randomly with $\alpha_j = a_j / \sum_{i=1}^{p} a_i$ and in $a_j \sim \mathrm{U}([0.2, 0.8])$. This choice guarantees that for moderate p, the weights α_j cannot become arbitrarily small, which might cause issues in a later analysis. As for the `"split"` algorithm, the number of observations and the type of eigenvalue decrease have to be supplied in `simMultiFunData`.

The following code simulates $N = 5$ bivariate functions on $\mathcal{T}_1 = [-0.5, 0.5]$ and $\mathcal{T}_2 = [0, 1] \times [-1, 1]$. The first elements of the eigenfunctions are derived from $M_1 = 12$ Fourier basis functions on $\mathcal{T}_1$ and the second elements of the eigenfunctions are constructed from tensor products of 4 eigenfunctions of the Wiener process on $[0, 1]$ and 3 Legendre polynomials on $[-1, 1]$, which give together $M_2 = 12$ eigenfunctions on $\mathcal{T}_2$. The scores are sampled using exponentially decreasing eigenvalues:

```
R> argvals <- list(seq(-0.5,0.5,0.01), list(seq(0,1,0.01), seq(-1,1,0.01)))
R> simMultiWeight <- simMultiFunData(N = 5, argvals = argvals,
                          eFunType = list("Fourier", c("Wiener", "Poly")),
                          eValType = "exponential",  M = list(12, c(4,3)),
                          type = "weighted")
```

As before, the result contains the simulated data as well as the eigenfunctions and eigenvalues. The first observation is shown in Fig. C.3 in the appendix.

Once simulated, the data can be further processed by adding noise or by artificially deleting measurements (sparsification). The latter is done in analogy to Yao et al. (2005) by first randomly generating the number of observed values that is to be retained for each function and then selecting the observation points randomly from the full grid. Note that this is currently implemented only for `funData` and `multiFunData` objects on one-dimensional domains. For the `addError` function, that adds pointwise noise $\varepsilon(t) \overset{\text{iid}}{\sim} \mathrm{N}(0, \sigma^2)$ (respectively $\varepsilon^{(j)}(t_j) \overset{\text{iid}}{\sim} \mathrm{N}(0, \sigma_j^2)$ for each element of multivariate functional data), the user can supply the standard deviation as a non-negative number σ for univariate data or as vector $\sigma = (\sigma_1, \ldots, \sigma_p)$ for the multivariate case. The `sparsify` function, that creates sparse data, accepts a `minObs` and a `maxObs` argument for the minimal and maximal number of values to be retained, which are again scalar for the univariate case and vectors for the multivariate case:

```
R> # univariate data
R> addError(simUniv1D$simData, sd = 0.5)
R> sparsify(simUniv1D$simData, minObs = 5, maxObs = 10)
R>
R> # multivariate data
R> addError(simMultiWeight$simData, sd = c(0.5, 0.3))
R> sparsify(simMultiSplit$simData, minObs = c(5, 50), maxObs = c(10, 80))
```

Both functions return an object of the same class as the input data. The modified functions of the code example are shown in the appendix (Fig. C.4 for the univariate case and in Fig. C.5 for the multivariate case).

5.4. The MFPCA Package

The **MFPCA** package implements multivariate functional principal component analysis (MFPCA) for data on potentially different dimensional domains (Happ and Greven, 2017+). It heavily builds upon the **funData** package, i.e. all functions are implemented as functional data objects. The **MFPCA** package thus illustrates the use of **funData** as a universal basis for implementing new methods for functional data. Section 5.4.1 gives a short review of the MFPCA methodology. The second section 5.4.2 describes the implementation including a detailed description of the main functions and a practical case study. For theoretical details, please refer to Happ and Greven (2017+).

5.4.1. Methodological Background

The basic idea of MFPCA is to extend functional principal component analysis to multivariate functional data on different dimensional domains. The data is assumed to be iid samples $x_1, \ldots, x_N$ of a random process $X = (X^{(1)}, \ldots, X^{(p)})$ with p elements $X^{(j)} \in L^2(\mathcal{T}_j)$ on domains $\mathcal{T}_j \subset \mathbb{R}^{d_j}$ with potentially different dimensional dimensions $d_j \in \mathbb{N}$. In Happ and Greven (2017+), the theoretical properties of the process X and the associated covariance operator are discussed. Moreover, the article provides an algorithm to estimate multivariate functional principal components and scores based on their univariate counterparts. The algorithm starts with demeaned samples $x_1, \ldots, x_N$ and consists of four steps:

1. Calculate a univariate functional principal component analysis for each element $j = 1, \ldots, p$. This results in principal component functions $\hat{\phi}_1^{(j)}, \ldots, \hat{\phi}_{M_j}^{(j)}$ and principal component scores $\hat{\xi}_{i,1}^{(j)}, \ldots, \hat{\xi}_{i,M_j}^{(j)}$ for each observation unit $i = 1, \ldots, N$ and suitably chosen truncation lags M_j.

2. Combine all coefficients into one big matrix $\Xi \in \mathbb{R}^{N \times M_+}$ with $M_+ = M_1 + \ldots + M_p$, having rows
$$\Xi_{i,\cdot} = \left(\hat{\xi}_{i,1}^{(1)}, \ldots, \hat{\xi}_{i,M_1}^{(1)}, \ldots, \hat{\xi}_{i,1}^{(p)}, \ldots, \hat{\xi}_{i,M_p}^{(p)}\right)$$
and estimate the joint covariance matrix $\hat{Z} = \frac{1}{N-1}\Xi^\top \Xi$.

3. Find eigenvectors $\hat{c}_m$ and eigenvalues $\hat{\nu}_m$ of $\hat{Z}$ for $m = 1, \ldots, M$ for some truncation lag $M \leq M_+$.

4. Calculate estimated multivariate principal component functions $\hat{\psi}_m$ and scores $\hat{\rho}_{i,m}$ based on the results from steps 1 and 3:
$$\hat{\psi}_m^{(j)} = \sum_{n=1}^{M_j} [\hat{c}_m]_n^{(j)} \hat{\phi}_n^{(j)}, \qquad \hat{\rho}_{i,m} = \sum_{j=1}^{p} \sum_{n=1}^{M_j} [\hat{c}_m]_n^{(j)} \hat{\xi}_{i,n}^{(j)} = \Xi_{i,\cdot} \hat{c}_m, \quad m = 1, \ldots, M.$$

The advantage of MFPCA with respect to univariate FPCA for each component can be seen in steps 2 and 3: The multivariate version takes covariation between the different elements into account, by using the joint covariance of the scores of all elements.

As discussed in Section 5.3.2, the multivariate principal component functions will have the same structure as the original samples, i.e. $\hat{\psi}_m = \left(\hat{\psi}_m^{(1)}, \ldots, \hat{\psi}_m^{(p)}\right)$ with $\hat{\psi}_m^{(j)} \in L^2(\mathcal{T}_j)$ for $m = 1, \ldots, M$. The scores $\hat{\rho}_{i,m}$ give the individual weight of each observation x_i for the principal component $\hat{\psi}_m$ in the empirical version of the truncated multivariate Karhunen-Loève representation:
$$x_i \approx \hat{\mu} + \sum_{m=1}^{M} \hat{\rho}_{i,m} \hat{\psi}_m, \tag{5.5}$$
with $\hat{\mu}$ being an estimate for the multivariate mean function, cf. (5.4).

In some cases, it might be of interest to replace the univariate functional principal component analysis in step 1 by a representation in terms of fixed basis functions $B_1^{(j)}, \ldots, B_{K_j}^{(j)}$, such as splines. In Happ and Greven (2017+) it is shown how the algorithm can be extended to arbitrary basis functions in $L^2(\mathcal{T}_j)$. Mixed approaches with some elements expanded in principal components and others for instance in splines are also possible. Another very likely case is that the elements of the multivariate functional data differ in their domain, range or variation. For this case, Happ and Greven (2017+) develop a weighted version of MFPCA with weights $w_j > 0$ for the different elements $j = 1, \ldots, p$. The weights have to be chosen depending on the data and the question of interest. One possible choice is to use the inverse of the integrated pointwise variance, as proposed in Happ and Greven (2017+). The weighted MFPCA is then based on the weighted scalar product (5.1) with weights $w_j = \left(\int_{\mathcal{T}_j} \widehat{\mathrm{Var}}(X^{(j)}(t))\mathrm{d}t\right)^{-1}$ and the associated weighted covariance operator.

5.4.2. MFPCA Implementation

The main function in the **MFPCA** package is `MFPCA`, that calculates the multivariate functional principal component analysis. It requires as input arguments a `multiFunData` object for which the MFPCA should be calculated, the number of principal components `M` to calculate and a list `uniExpansions` specifying the univariate representations to use in step 1.

Case Study: Calculating the MFPCA for the Canadian Weather Data

The following example calculates a multivariate functional principal component analysis for the bivariate Canadian weather data with three principal components, using univariate FPCA with five principal components for the daily temperature (element 1) and univariate FPCA with four principal components for the monthly precipitation (element 2). The univariate expansions are specified in a list with two list entries (one for each element) and are then passed to the main function:

```
R> uniExpansions <- list(list(type = "uFPCA", npc = 5), # temperature element
                         list(type = "uFPCA", npc = 4)) # precipitation element
R> MFPCAweather <- MFPCA(canadWeather, M = 3, uniExpansions = uniExpansions)
```

The full analysis takes roughly nine seconds on a standard laptop, with most time spent for the univariate decompositions (if the elements are e.g. expanded in penalized splines, the total calculation time reduces to one second, see also the next section).

The result of the `MFPCA` function is a list containing the multivariate mean function (`meanFunction`, as the data is demeaned automatically before the analysis), the empirical multivariate principal component functions (`functions`) and scores (`scores`) as well as the estimated eigenvalues (`values`). Additionally, it returns the eigenvectors $\hat{c}_m$ (`vectors`) and normalization factors (`normFactors`, arise if non-orthogonal basis functions or weights are used), that can be used for predicting scores out-of-sample. All functions are represented as functional data objects and can thus for example be plotted using the methods provided by the **funData** package (see Fig. 5.9). The mean function of the temperature element is seen to have low values below -10°C in the winter and a peak at around 15°C in the summer, while the mean of the monthly precipitation data is slightly increasing over the year. The first principal component has negative values for both elements, i.e. weather stations with positive scores will in general have lower temperatures and less precipitation than the mean. The difference is more pronounced in the winter than in the summer, as both the temperature as well as the precipitation element of the first principal component has more negative values in the winter period. This indicates that there is covariation between both elements, that can be captured by the MFPCA approach. In total, the first bivariate eigenfunction

can be associated with arctic and continental climate, characterized by low temperatures (especially in the winter) and less precipitation than on average. Weather stations with negative score values will show an opposite behavior, with higher temperatures and more rainfall than on average, particularly in the winter months. This is typical for maritime climate.

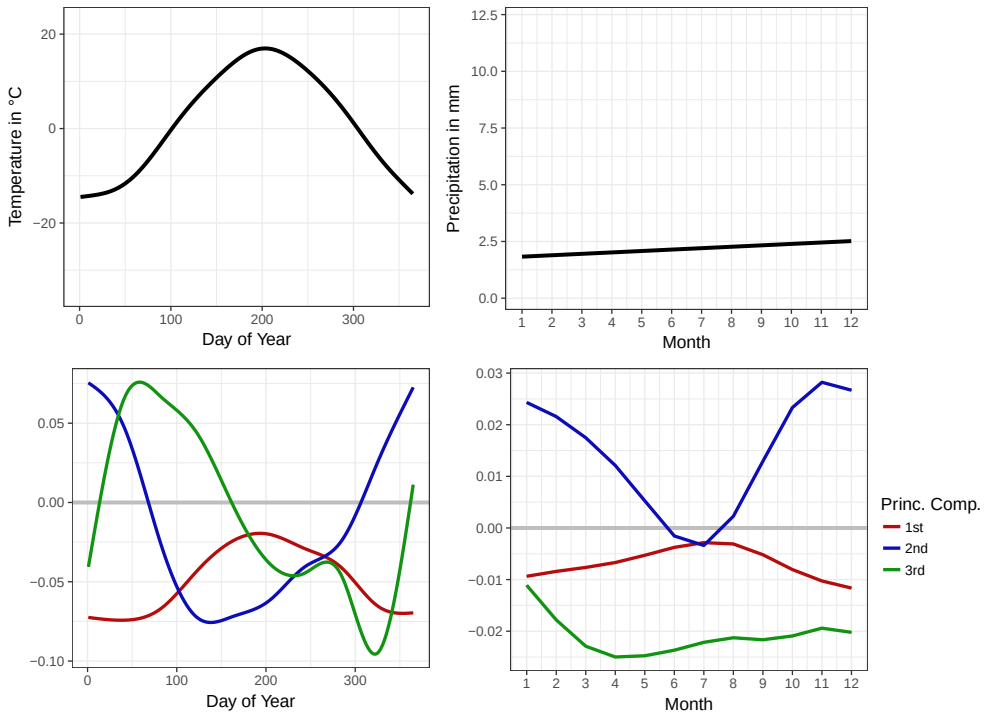


Figure 5.9.: MFPCA results for the Canadian weather data. First row: The bivariate mean function, which is subtracted from the data before calculating the MFPCA. Second row: The first three bivariate functional principal components. The gray horizontal lines in the principal component plots mark zero.

The estimated scores for the first principal component support this interpretation, as weather stations in arctic and continental areas mainly have positive scores, while stations in the coastal areas have negative values in most cases (see Fig. 5.10). Moreover, weather stations in the arctic and pacific regions are seen to have more extreme score values than those in continental areas and on the Atlantic coast, meaning that the latter have a more moderate climate. The eigenvalues, that also give the variance of the scores, are rapidly decreasing, i.e. the first principal component explains most of the variability in the data:

```
R> MFPCAweather$values

[1] 15541.684  1481.960   330.112
```

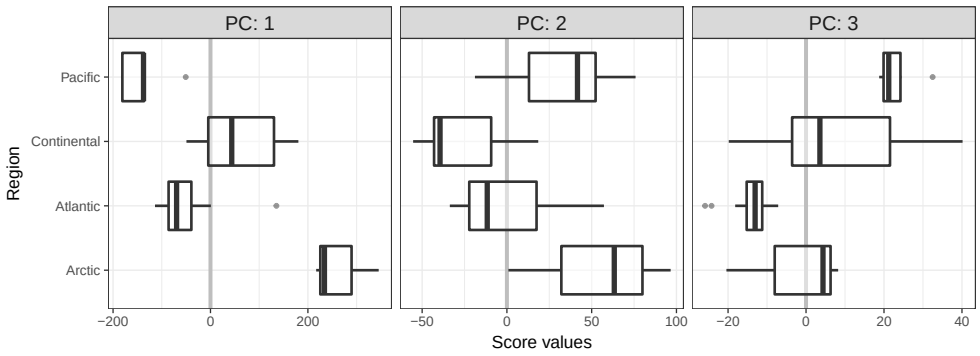


Figure 5.10.: Scores of the first three bivariate functional principal components (PCs) for the Canadian weather data depending on the region of each weather station. The gray vertical lines mark zero.

Univariate Basis Expansions

In the above example, both univariate elements have been decomposed in univariate functional principal components in step 1. The **MFPCA** package implements some further options for the univariate expansions, that can easily be extended in a modular way. Currently implemented basis expansions are:

- `uFPCA`: Univariate functional principal component analysis for data on one-dimensional domains. This option was used in the previous example. The list entry for one element has the form:

  ```
  R> list(type = "uFPCA", nbasis, pve, npc, makePD, cov.weight.type)
  ```

  The implementation is based on the PACE approach (Yao et al., 2005) with the mean function and the covariance surface smoothed with penalized splines (Di et al., 2009), following the implementation in the **refund** package. The `MFPCA` function returns the smoothed mean function, while for all other options, the mean function is calculated pointwise. Options for this expansion include the number of basis functions `nbasis` used for the smoothed mean and covariance functions (defaults to 10; for the covariance this number of basis functions is used for each marginal); `pve`, a value between 0 and 1, giving the proportion of variance that should be explained by the principal components (defaults to 0.99); `npc`, an alternative way to specify the number of principal components to be calculated explicitly (defaults to `NULL`, otherwise overrides `pve`); `makePD`, an option to enforce positive definiteness of the covariance surface estimate (defaults to `FALSE`) and `cov.weight.type`, which characterizes the weighting scheme for the covariance surface (defaults to `"none"`).

- `spline1D` and `spline1Dpen`: These options calculate a spline representation of functions on one-dimensional domains using the `gam` function in the **mgcv** package (Wood,

2011; Wood, 2017). When using this option, the `uniExpansions` entry for one element is of the form:

```
R> list(type = "splines1D", bs, m, k)
R> list(type = "splines1Dpen", bs, m, k, parallel)
```

For `spline1Dpen`, the coefficients are found by a penalization approach, while for `spline1D` the observations are simply projected on the spline space without penalization. Thus, the `spline1Dpen` option will in general lead to smoother representations than `spline1D`. Possible options passed for these expansions are `bs`, the type of basis functions to use (defaults to `"ps"` for possibly penalized B-spline functions); `m`, the order of the spline basis (defaults to `NA`, i.e. the order is chosen automatically); `k`, the number of basis functions to use (default value is `-1`, which means that the number of basis functions is chosen automatically). For the penalized version, there is an additional option `parallel` which, if set to `TRUE`, calculates the spline coefficients in parallel. In this case, a parallel backend must be registered before (defaults to `FALSE`).

- `spline2D` and `spline2Dpen`: These are analogue options to `spline1D` and `spline1Dpen` for functional data on two-dimensional domains (images):

```
R> list(type = "splines2D", bs, m, k)
R> list(type = "splines2Dpen", bs, m, k, parallel)
```

The parameters `bs`, `m` and `k` for the type, order and number of basis functions can be either a single number/character string that is used for all marginals or a vector with the specifications for all marginals. For the penalized version, the function `bam` in **mgcv** is used to speed up the calculations and reduce memory load. Setting `parallel=TRUE` enables parallel calculation of the basis function coefficients. As for the one-dimensional case, this requires a parallel backend to be registered before.

- `FCP_TPA`: This option uses the *Functional CP-TPA* algorithm of Allen (2013) to compute an eigendecomposition of image observations, which can be interpreted as functions on a two-dimensional domain. The algorithm assumes a CANDECOMP/PARAFRAC (CP) representation of the data tensor $X \in \mathbb{R}^{N \times S_x \times S_y}$ containing all observations x_i with $S_x \times S_y$ pixels, each:

$$X = \sum_{m=1}^{M} d_m u_m \circ v_m \circ w_m$$

Here, d_m is a scalar, $u_m \in \mathbb{R}^N, v_m \in \mathbb{R}^{S_x}, w_m \in \mathbb{R}^{S_y}$ are vectors and $\circ$ denotes the outer product. We can thus interpret $v_m \circ w_m$ as the m-th univariate eigenfunction $\hat{\psi}_m$ evaluated at the same pixels as the originally observed data. The vector $d_m \cdot u_m \in \mathbb{R}^N$ can in turn be interpreted as the score vector containing the scores for the m-th principal component function and each observation. Note that this interpretation does neither

imply that the eigenimages are normed functions nor that they are ordered such that the associated eigenvalues form a decreasing sequence. The algorithm proposed in Allen (2013) includes smoothing parameters $\lambda_u, \lambda_v, \lambda_w \geq 0$ to smooth along all dimensions, extending the approach of Huang, Shen, et al. (2009) from one-dimensional to two-dimensional functions. As smoothing along the observations $u_m \in \mathbb{R}^N$ is not required in the given context, the parameter λ_u is fixed to zero and the smoothing is implemented only for the v and w directions. When decomposing images with this algorithm, the user has to supply a list of the following form for the corresponding element:

```
R> list(type = "FCP_TPA", npc, smoothingDegree, alphaRange,
        orderValues, normalize)
```

Required options are `npc`, the number of eigenimages to be calculated, and `alphaRange`, the range of the smoothing parameters. The latter must be a list with two entries named `v` and `w`, giving the possible range for λ_v, λ_w as vectors with the minimal and maximal value, each (e.g. `alphaRange = list(v = c(10^-2,10^2), w = c(10^-3,10^3))` would enforce $\lambda_v \in [10^{-2}, 10^2]$ and $\lambda_w \in [10^{-3}, 10^3]$). Optimal values for λ_v and λ_w are found by numerically optimizing a GCV criterion (cf. Huang, Shen, et al., 2009, in the one-dimensional case). Further options are the smoothing degree, i.e. the type of differences that should be penalized in the smoothing step (`smoothingDegree`, defaults to second differences for both directions) and two logical parameters concerning the ordering of the principal components and their normalizations: If `orderValues` is `TRUE`, the eigenvalues and associated eigenimages and scores are ordered decreasingly (defaults to `TRUE`), i.e. the first eigenimage corresponds to the highest eigenvalue that has been found, the second eigenimage to the second highest eigenvalue and so on. The option `normalize` specifies whether the eigenimages should be normalized (defaults to `FALSE`).

- `UMPCA`: This option implements the UMPCA (Uncorrelated Multilinear Principal Component Analysis, Lu et al., 2009) algorithm for finding uncorrelated eigenimages of two-dimensional functions (images). Essentially, this implements the UMPCA toolbox for MATLAB (Lu, 2012) in **R**:

  ```
  R> list(type = "UMPCA", npc)
  ```

  The number of eigenimages that are calculated has to be supplied by the user (`npc`). Note that this algorithm aims more at uncorrelated features than at an optimal reconstruction of the images and thus may lead to unsatisfactory results for the MFPCA approach.

- `DCT2D`/`DCT3D`: This option calculates a representation of functional data on two- or three-dimensional domains in a tensor cosine basis. For speeding up the calculations,

the implementation is based on the **fftw3 C**-library (Frigo and Johnson, 2005, developer version). If the **fftw3-dev** library is not available during the installation of the **MFPCA** package, the `DCT2D` and `DCT3D` options are disabled and throw an error. After installing **fftw3-dev** on the system, **MFPCA** has to be re-installed to activate `DCT2D`/`DCT3D`. The `uniExpansions` entry for a cosine representation of 2D/3D elements is:

```
R> list(type = "DCT2D", qThresh, parallel)
R> list(type = "DCT3D", qThresh, parallel)
```

The discrete cosine transformation is a real-valued variant of the fast Fourier transform (FFT) and usually results in a huge number of non-zero coefficients that mostly model "noise" and can thus be set to zero without affecting the representation of the data. The user has to supply a threshold between 0 and 1 (`qThresh`) that defines the proportion of coefficients to be thresholded. Setting e.g. `qThresh = 0.9` will set 90% of the coefficients to zero, leaving only the 10% of the coefficients with the highest absolute values. The coefficients are stored in a `sparseMatrix` (package **Matrix**) object to reduce the memory load for the following computations. The calculations can be run in parallel for the different observations by setting the parameter `parallel` to `TRUE` (defaults to `FALSE`), if a parallel backend has been registered before.

Further Options for `MFPCA`

With the mean function, the principal components and the individual scores calculated in the `MFPCA` function, the observed functions $x_1, \ldots, x_N$ can be reconstructed based on the truncated Karhunen-Loève representation with plugged-in estimators as in (5.5). The reconstructions can be obtained by setting the option `fit = TRUE`, which adds a multivariate functional data object `fit` with N observations to the result list, where the i-th entry corresponds to the reconstruction $\hat{x}_i$ of an observation x_i. For a weighted version of MFPCA, the weights can be supplied to the `MFPCA` function in form of a vector `weights` of length p, containing the weights $w_j > 0$ for each element $j = 1, \ldots, p$. Both options are used in the following example for the `CanadWeather` data, which uses the weights based on the integrated pointwise variance, as discussed in Happ and Greven (2017+):

```
R> # calculate pointwise variance
R> varTemp <- funData(argvals = canadWeather[[1]]@argvals,
                      X = matrix(apply(canadWeather[[1]]@X, 2, var), nrow = 1))
R> varPrec <- funData(argvals = canadWeather[[2]]@argvals,
                      X = matrix(apply(canadWeather[[2]]@X, 2, var), nrow = 1))
R>
R> # compute weights based on variance functions
```

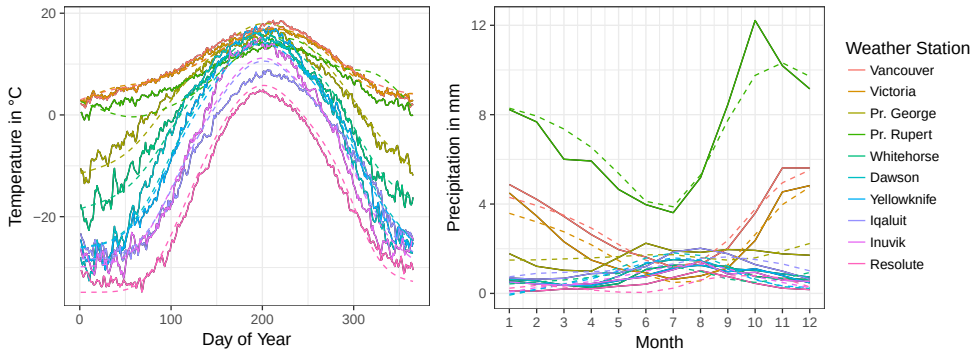


Figure 5.11.: The ten observations of the bivariate Canadian weather data shown in Fig. 5.6 (solid lines) and their reconstruction (dashed lines) based on the truncated Karhunen-Loève representation with estimates found by a weighted version of MFPCA (`MFPCAweatherFit`).

```
R> weightWeather <- c(1/integrate(varTemp), 1/integrate(varPrec))
R>
R> # weighted MFPCA with reconstruction for each observation
R> MFPCAweatherFit <- MFPCA(canadWeather, M = 3, uniExpansions = uniExpansions,
                            weights = weightWeather, fit = TRUE)
```

Fig. 5.11 shows some original functions of the `canadWeather` data and their reconstructions saved in `MFPCAweatherFit`.

If elements are expanded in fixed basis functions, the number of basis functions that are needed to represent the data well will in general be quite high, particularly for elements with higher dimensional domains. As a consequence, the covariance matrix of all scores in step 2 can become large and the eigendecompositions in step 3 can get computationally very demanding. By setting the option `approx.eigen = TRUE`, the eigenproblem is solved approximately using the augmented implicitly restarted Lanczos bidiagonalization algorithm (IRLBA, Baglama and Reichel, 2005) implemented in the **irlba** package (Baglama, Reichel, and Lewis, 2017). If the number M of principal components is low with respect to the number of observations N and the total number of univariate basis functions for all elements, the approximation will in general work very well. If in contrast M is small with respect to N or the total number of univariate basis functions, the approximation may be inappropriate. Following the recommendations in **irlba**, the approximation is not used if M is larger than $\frac{1}{2}\min(N, M_+)$ and a warning is thrown. On the other hand, if `approx.eigen = FALSE` and the total number of univariate basis functions exceeds 1000, `MFPCA` throws a warning, but continues calculating the exact eigendecomposition.

Bootstrap Confidence Bands

The **MFPCA** function implements nonparametric bootstrap on the level of functions to quantify the uncertainty in the estimation of the MFPCA (cf. Happ and Greven, 2017+). It calculates pointwise bootstrap confidence bands for the principal component functions and bootstrap confidence bands for the associated eigenvalues. For elements with fixed basis functions, that do not depend on the data (i.e. no principal component-related methods for the univariate decomposition), the algorithm efficiently uses the results of step 1 by resampling from the scores on the level of curves. For all other elements, the principal components and associated scores are recalculated for each bootstrap sample.

The confidence bands are calculated by setting the parameter `bootstrap` to `TRUE`. In this case, the user has to supply a parameter `nBootstrap`, that gives the number of bootstrap iterations and `bootstrapAlpha`, which specifies the significance level for the bootstrap confidence intervals. This can be a number or a vector with multiple levels, if confidence bands for more than one level should be calculated at a time. It defaults to 0.05. The bootstrap confidence bands for the principal components are returned in an element `CI`, which has the same length as `bootstrapAlpha`, the vector of significance levels. For each level, the corresponding entry in `CI` is named e.g. `alpha_0.05` for a confidence level of 0.05. Each is a list of length 2, containing the upper (`upper`) and the lower bound (`lower`) as `multiFunData` objects. The confidence intervals for the eigenvalues are returned in an entry `CIvalues`, which is a list of length 2, containing the vectors of the lower or upper confidence bounds, respectively.

5.5. Summary and Outlook

The **funData** package implements functional data in an object-oriented manner. The aim of the package is to provide a flexible and unified toolbox for dense univariate and multivariate functional data with different dimensional domains as well as irregular functional data. The package implements basic utilities for creating, accessing and modifying the data, upon which other packages can be built. This distinguishes the **funData** package from other packages for functional data, that either do not provide a specific data structure together with basic utilities or mix this aspect with the implementation of advanced methods for functional data.

The **funData** package implements three classes for representing functional data based on the observed values and without any further assumptions such as basis function representations. The classes follow a unified approach for representing and working with the data,

which means that the same methods are implemented for all the three classes (polymorphism). The package further includes a full simulation toolbox for univariate and multivariate functional data on one- and higher dimensional domains. This is a very useful feature when implementing and testing new methodological developments.

The **MFPCA** package is an example for an advanced methodological package, which builds upon the **funData** functionalities. It implements a new approach, multivariate functional principal component analysis for data on different dimensional domains (Happ and Greven, 2017+). All calculations relating to the functional data, data input and output use the basic **funData** classes and methods.

Both packages, **funData** and **MFPCA**, are publicly available on CRAN (`https://CRAN.R-project.org`) and GitHub (`https://github.com/ClaraHapp`). They come with a comprehensive documentation, including many examples. Both of them use the **testthat** system for unit testing (Wickham, 2011), to make the software development more safe and stable and reach a code coverage of over 95%.

Potential future extensions of the **funData** package include interfaces to other packages, e.g. **fda**, **fda.usc** or **refund**. This could open all methods for functional data implemented in these packages to functional data objects of the **funData** package. At the same time, the results of these methods could be transformed back to functional data objects of the **funData** package for using the basic utility functions, such as e.g. plotting based on the **ggplot2** graphics engine. Further, one may think of extending the existing classes to represent an even broader group of functional data, by e.g. allowing the `irregFunData` class to have observation points in a higher dimensional space or by providing appropriate plotting methods for one-dimensional curves in 2D or 3D space. For the **MFPCA**, new basis functions as e.g. wavelets could be implemented.

Part II.

Scalar-on-Image Regression

6. The Impact of Model Assumptions in Scalar-on-Image Regression

This chapter is a reprint of:

C. Happ, S. Greven, and V. J. Schmid for the Alzheimer's Disease Neuroimaging Initiative (ADNI) (2017). "The Impact of Model Assumptions in Scalar-on-Image Regression". arXiv: 1707.02233

Author contributions:

The project was carried out in close collaboration of all authors. Clara Happ did the implementation, simulations and data analysis, and wrote the manuscript. Sonja Greven and Volker Schmid supervised the writing, added valuable comments and proofread the paper.

Acknowledgements:

The authors thank the Alzheimer's Disease Neuroimaging Initiative (ADNI) Committee for providing access to the data as well as Michael Ewers and Miguel Ángel Araque Caballero for registering the FDG-PET scans. The `R` code for the *bumpy* function was kindly provided by Philip T. Reiss. Financial support from the German Research Foundation through Emmy Noether grant GR 3793/1-1 is gratefully acknowledged.

Abstract:

Complex statistical models often require strong assumptions to overcome the issue of non-identifiability. While in theory it is well understood that model assumptions can strongly influence the results, this seems to be underappreciated in practice. We address the impact of model assumptions in the case of scalar-on-image regression.

This paper gives a systematic overview of the main approaches and their assumptions. We propose to categorize the latter into underlying and parametric assumptions and develop measures to quantify the degree to which specific assumptions are met. The extent of the problem is investigated in a simulation study and in a practical application to neuroimaging data. The results show that while the predictive performance is similar across models, different assumptions can indeed lead to quite different estimates, raising the question of their interpretability.

We thus recommend to carefully identify the assumptions made in a model. Moreover, it seems helpful to compare the obtained estimates to others, found by models with different assumptions, in order to find common patterns. These might help understanding which features in the estimate are mostly driven by the data, dominated by the model assumptions or supported by both. Systematic simulation studies based on observed data and hypothetic coefficient images, as used in this paper, may lead to further insights about the features in coefficient images that can be found with the data at hand.

6.1. Introduction

With the increasing availability of complex data, highly specialized statistical methods have been developed in recent years, which account for the particular characteristics of such data. Image data, which are the focus of this article, are an example of data that require specifically tailored statistical methods. Their development has largely been driven by the technological progress in medical imaging and related fields (e.g. Friston et al., 2007; Smith and Fahrmeir, 2007). Given today's high-resolution imaging techniques, the number of pixels (or voxels) and therefore parameters in an image is usually much higher than the number of observed images. This makes the statistical analysis of image data a typical example of an $n \ll p$ problem, where the number of variables is higher than the number of observations. Whereas for e.g. genetic data variable selection methods have proven very useful (Zou and Hastie, 2005), statistical methods for image data can make use of the spatial structure in the data. This is achieved by making structural model assumptions, that translate e.g. the hypothesis of a smooth image into restrictions on the model parameters. However, as stated in Coombs (1964) "we buy information with assumptions". This comes at the price that the estimate found by a certain model contains not only information from the data, but also from the assumptions made in the model. It is thus fundamental to consider the role of the model assumptions when analyzing the results.

Non-identifiability in scalar-on-image regression: In this paper, we address the issue of how model assumptions influence the estimates in the case of scalar-on-image regression. The goal here is to find a relationship between a scalar response and an image covariate. In contrast to the widely used and very popular pixelwise or voxelwise methods, as for example statistical parametric mapping (SPM, Friston et al., 2007), the pixels enter the scalar-on-image regression models all at once. Consequently, the number of variables in principle equals the number of pixels in the image, plus potentially further effects of other covariates. The model is hence inherently unidentified and requires strong assumptions on the coefficients to overcome the issue of non-identifiability. While this is less problematic for prediction (different coefficient images may give similar predictions), it remains an issue for the estimation and particularly for the interpretability of the coefficient image. Although all this is well understood from a theoretical point of view, we consider it an underappreciated problem in practice, which entails the risk of over-interpreting effects that are mainly driven by the model assumptions.

Aims of this paper: The aim of this paper is three-fold. First, we provide an overview of different conceptual approaches for scalar-on-image regression, including their assumptions and currently available implementations. We systematically compare the models from a theoretical point of view as well as in simulations and in a real case study based on neuroimaging data from a study on Alzheimer's disease (Mueller et al., 2005; Weiner et al.,

2015). Second, due to the inherent non-identifiability of scalar-on-image regression models, we investigate the influence of the model assumptions on the coefficient estimates and examine the extent of the problem in practice. We show that different assumptions can indeed lead to quite different estimates, raising the question of interpretability of the resulting estimates. Finally, we give recommendations and develop measures that may help to make modeling and interpretation decisions in practice.

Considered models and their assumptions: Overall, we discuss eight models that represent the principal approaches for scalar-on-image regression. Some reduce the complexity by means of basis function representations of the coefficient image, such as penalized splines (Marx and Eilers, 2005), wavelets (Daubechies, 1988) or functional principal components (Allen, 2013) and can therefore be related to the broad field of scalar-on-function regression methods (Reiss, Goldsmith, et al., 2016+; Müller and Stadtmüller, 2005; Cardot et al., 1999). Others apply dimension reduction methods, such as principal component analysis or partial least squares, partly combined with a basis function expansion (Reiss and Ogden, 2010; Reiss, Huo, et al., 2015). Finally, we also consider methods that formulate the model assumptions in terms of spatial Gaussian Markov random field priors (Besag, 1974; Goldsmith, Huang, et al., 2014). We argue that the structural assumptions made in the different models can come in different levels of abstraction and propose to distinguish between underlying and parametric model assumptions. In a spline regression model, for example, the underlying assumption is smoothness, while the parametric assumption puts a difference penalty on neighbouring spline coefficients. Across all scalar-on-image regression models considered in this paper, we find smoothness and sparsity to be the two key concepts. One or both of them are considered in all discussed models. Prior variability and projection onto a subspace are further central concepts. For all types of assumptions, we propose interpretable measures to quantify the extent to which the model assumptions are met. They can help to understand the influence of the assumptions on the estimation result and are an important contribution for the appropriate interpretation of the results.

Conclusions from simulated and real data: The influence of the model assumptions is studied in more detail in a simulation study with four different coefficient images that reflect the four main assumptions of the models. The results show that while the predictive performance is mostly similar among the models, there is a wide variability in the goodness of the estimation, depending both on the "true" coefficient image and the assumptions made in the model. We therefore highly recommend to carefully choose the model assumptions in scalar-on-image regression and to take them into account when interpreting resulting estimates. In the simulation study we demonstrate how numerical experiments with the real data and hypothetic coefficient images can be used to determine which types of features in a coefficient image can be found using models with appropriate assumptions and the observed data. Moreover, comparing the estimate with results from similar models with different assumptions may help to distinguish between features that are mostly driven by

the data, those that mainly reflect the model assumptions and others that are supported by both.

Outline: The article is structured as follows: Section 6.2 introduces the scalar-on-image regression model and the different estimation methods. Particular emphasis is put on the model assumptions. Section 6.3 compares and categorizes the assumptions, distinguishing between underlying and parametric model assumptions. Further, measures are developed that allow to characterize to what extent the assumptions of a certain model are met. Section 6.4 contains the simulation study and Section 6.5 presents the neuroimaging application. The paper concludes with a short discussion and an outlook to potential future research in Section 6.6.

6.2. Overview of Methods for Scalar-on-Image Regression

This section introduces the scalar-on-image regression model (Section 6.2.1) and provides a systematic overview of the approaches considered in this paper (Section 6.2.2 for basis function approaches and Section 6.2.3 for random field methods). The presented models have been selected to represent the most important assumptions and all relevant model classes in scalar-on-image regression. In addition, we have focused on easily accessible methods, for which software solutions are already available or can be implemented without much effort. An overview of the implementations for the studied methods is given in Section 6.2.4 and in the code supplement of this article (available on GitHub: `https://github.com/ClaraHapp/SOIR`).

6.2.1. The Scalar-on-Image Regression Model

The scalar-on-image regression model studied in this paper is assumed to be of the following form:

$$y_i = \sum_{j=1}^{p} w_{i,j}\alpha_j + \sum_{l=1}^{L} x_{i,l}\beta_l + \varepsilon_i, \quad i = 1, \ldots, N. \tag{6.1}$$

The observed data for each of the $N \in \mathbb{N}$ observation units (e.g. subjects in a medical study) consist of a scalar response y_i, an image covariate x_i with $L \in \mathbb{N}$ pixels (demeaned over all observations) and scalar covariates $w_i \in \mathbb{R}^p$, including an intercept term. As in the standard linear model, the vector $\alpha \in \mathbb{R}^p$ contains the coefficients for w_i and the error term ε_i is assumed to be i.i.d. Gaussian with variance $\sigma_\varepsilon^2 > 0$. The coefficient image β relates the observed images x_i to the response and therefore has the same size as x_i. Alternatively, the model can be written in matrix-form

$$y = W\alpha + X\beta + \varepsilon \tag{6.2}$$

with $y = (y_1, \ldots, y_N)$, $W \in \mathbb{R}^{N \times p}$ the matrix of scalar covariates (row-wise), $X \in \mathbb{R}^{N \times L}$ the matrix of vectorized image covariates, $\beta \in \mathbb{R}^L$ the vectorized coefficient image and $\varepsilon \sim \mathrm{N}(0, \sigma_\varepsilon^2 I_N)$ with $I_N \in \mathbb{R}^{N \times N}$ the identity matrix. Note that theoretically, the images x_i and therefore also the coefficient image β can be two-, three- or even higher dimensional. In practice, increasing the dimensionality of the images is frequently associated with a considerable computational burden and is not supported by all implementations, see Section 6.2.4. For reasons of simplicity and comparability, only 2D images are considered in the following analysis.

Model (6.1) is effectively a standard linear model with coefficients α and β. In most cases, however, the total number of coefficients $p + L$ will exceed the number of observation units N by far, i.e. model (6.1) will in general be unidentifiable. On the other hand, the coefficients β_l are known to form an image and thus will show dependencies between neighbouring pixels. It is therefore natural to make structural assumptions about β such as e.g. smoothness. These assumptions imply restrictions on the coefficients β_l and can thus help to overcome the issue of non-identifiability. As the true β coefficient is unknown, the structural assumptions on β have to be made prior to the analysis. They reflect prior beliefs about the unknown image and can be expected to have an influence on the result. If one assumes e.g. smoothness and thus enforces smoothness in the model estimation, then the result will be a more or less smooth image.

6.2.2. Basis Function Approaches

Basis function approaches start from the idea that the unknown coefficient image is generated by a function $\beta(\cdot)\colon \mathcal{T} \to \mathbb{R}$ with $\mathcal{T} \subset \mathbb{R}^2$. The function is evaluated at a rectangular grid of observation points $t_l \in \mathcal{T}$ (the pixels), such that $\beta_l = \beta(t_l)$, and assumed to lie in the span of K known basis functions $B_1, \ldots, B_K$, which is a K-dimensional space. Then (6.1) translates to

$$y_i = \sum_{j=1}^{p} w_{i,j}\alpha_j + \sum_{l=1}^{L} x_{i,l}\beta_l + \varepsilon_i = \sum_{j=1}^{p} w_{i,j}\alpha_j + \sum_{l=1}^{L} x_{i,l} \sum_{k=1}^{K} b_k B_k(t_l) + \varepsilon_i. \tag{6.3}$$

This assumption reduces the estimation of β from L coefficients β_l to K coefficients b_k, as usually the number of basis functions K is chosen much smaller than the number of pixels L. If further $p + K < N$, this solves the identifiability issue. Otherwise, one can make additional assumptions on the coefficients b_k, depending on the basis functions used.

If the basis functions B_k are orthonormal, it can be useful to interpret the observed images x_i as functions, too, and to expand them in the same basis functions as $\beta(\cdot)$ with coefficients $\theta_{i,m}$, as then

$$\begin{aligned} y_i &= \sum_{j=1}^{p} w_{i,j}\alpha_j + \sum_{l=1}^{L} x_{i,l}\beta_l + \varepsilon_i \approx \sum_{j=1}^{p} w_{i,j}\alpha_j + \sum_{m=1}^{K}\sum_{k=1}^{K} \theta_{i,m} b_k \sum_{l=1}^{L} B_m(t_l)B_k(t_l) + \varepsilon_i \\ &\approx \sum_{j=1}^{p} w_{i,j}\alpha_j + \sum_{m=1}^{K}\sum_{k=1}^{K} \theta_{i,m} b_k \int_{\mathcal{T}} B_m(t)B_k(t)\mathrm{d}t + \varepsilon_i = w_i^\top \alpha + \theta_i^\top b + \varepsilon_i, \end{aligned} \tag{6.4}$$

which is a standard linear regression with the covariate vectors $w_i = (w_{i,1}, \ldots, w_{i,p})$ and $\theta_i = (\theta_{i,1}, \ldots, \theta_{i,K})$ and the coefficient vectors α and $b = (b_1, \ldots, b_K)$. Note that the approximation in (6.4) in general must include integration weights to be valid. In most cases, however, the pixels are all equidistant and the weights can be set to one, at most changing the scale of $\beta(\cdot)$. Given an estimate $\hat{b}$, a simple plug-in estimate for β then is $\hat{\beta}_l = \sum_{k=1}^{K} \hat{b}_k B_k(t_l)$.

The choice of the basis functions has a considerable influence on the estimate $\hat{\beta}$. We divide the methods into three classes with fixed basis functions, data-driven basis functions or a combination of the two.

Fixed basis functions

(Penalized) B-Splines, Marx and Eilers (2005), in the following referred to as *Splines*: B-Splines (Eilers and Marx, 1996; De Boor, 1972) are a popular class of bases for representing smooth functions. In the case of a two-dimensional function evaluated on a grid of pixels, one can use tensor product splines (Marx and Eilers, 2005), giving

$$\beta(t) = \sum_{k_x=1}^{K_x}\sum_{k_y=1}^{K_y} b_{k_x,k_y} B_{k_x}(t_x)B_{k_y}(t_y)$$

for $t = (t_x, t_y)$. In the scalar-on-image regression model proposed by Marx and Eilers (2005), the unknown coefficients b and α are found by minimizing the penalized least squares criterion

$$\sum_{i=1}^{N}\left(y_i - \sum_{j=1}^{p} w_{i,j}\alpha_j - \sum_{l=1}^{L} x_{i,l} \sum_{k_x=1}^{K_x}\sum_{k_y=1}^{K_y} b_{k_x,k_y} B_{k_x}(t_{l,x})B_{k_y}(t_{l,y})\right)^2 + \lambda_x \operatorname{pen}_x(b) + \lambda_y \operatorname{pen}_y(b) \to \min_{\alpha,b},$$

where the penalty terms pen_x and pen_y usually penalize differences in b along the x- and y- axes in order to obtain a smooth function and an identifiable model. In Happ (2013) and more generally in Scheipl and Greven (2016), however, it is shown that identifiability remains an issue if the covariates do not contain sufficient variation in the penalty null

space. The penalty parameters $\lambda_x, \lambda_y > 0$ can be found e.g. by (generalized) cross-validation (Marx and Eilers, 2005) or using a restricted maximum likelihood (REML) approach (Wood, 2011).

The main assumption here is that the unknown coefficient function $\beta(\cdot)$ can be represented well by the $K_x \cdot K_y$ tensor product spline basis functions and that it has smooth variation.

Wavelets, Reiss, Huo, et al. (2015), *WNET*:
Given a so-called pair of mother and father wavelet functions ψ and ϕ, an arbitrary function f can be expressed as

$$f(t) = \sum_{n\in\mathbb{Z}} c_{M_0,n}\phi_{M_0,n}(t) + \sum_{m=-\infty}^{M_0}\sum_{n\in\mathbb{Z}} d_{m,n}\psi_{m,n}(t)$$

with coefficients $c_{M_0,n} = \langle f, \phi_{M_0,n}\rangle_2$ and $d_{m,n} = \langle f, \psi_{m,n}\rangle_2$. The basis functions $\phi_{m,n}$ and $\psi_{m,n}$ are orthonormal for a given resolution level m and derive from the original mother and father wavelets via dilatation and translation: $\psi_{m,n}(t) = 2^{-m/2}\psi(2^{-m}t - n)$ and $\phi_{m,n}(t) = 2^{-m/2}\phi(2^{-m}t - n)$ with $m, n \in \mathbb{Z}$ (see e.g. Daubechies, 1988). In practical applications, f will be observed on a finite grid $\{t_1, \ldots, t_L\}$, and thus the infinite sums will be truncated. For the two-dimensional case, one can again use a tensor-type approach, defining basis functions for the x-, y- and xy-directions. The basis coefficients can be obtained efficiently if the side length of the image is a power of 2 (Mallat, 1989).

In practice, one observes that only a few basis functions are needed to describe most functions well, even those with sharp, highly localized features, due to the different resolutions of the basis functions. The majority of the coefficients can therefore be set to 0 without affecting the important characteristics of the function. This is the basic idea of the scalar-on-image model proposed in Reiss, Huo, et al. (2015), where the expansion of the unknown coefficient function $\beta(\cdot)$ in wavelet basis functions is combined with a variable selection step. As the wavelet basis functions form an orthonormal basis, the observed images x_i are transformed to the wavelet space, according to (6.4), renaming the wavelet coefficients $c_{M_0,n}$ and $d_{m,n}$ associated with x_i to $\theta_{i,1}, \ldots, \theta_{i,K}$ and the corresponding wavelet basis functions to $B_1, \ldots, B_K$. The coefficients b_k thus represent the wavelet coefficients of $\beta(\cdot)$ with respect to the same basis. In general, the number of coefficients K will equal the number of pixels, i.e. the model is still non-identifiable. Here is where the variable selection step comes into play: Reiss, Huo, et al. (2015) propose to add a (naïve) elastic net penalty (Zou and Hastie, 2005), i.e. one minimizes

$$\sum_{i=1}^{N}\left(y_i - \sum_{j=1}^{p} w_{i,j}\alpha_j - \sum_{k=1}^{K}\theta_{i,k}b_k\right)^2 + \lambda\left[\eta\sum_{k=1}^{K}|b_k| + (1-\eta)\sum_{k=1}^{K}b_k^2\right] \to \min_{\alpha,b}$$

for a penalty parameter $\lambda > 0$ and a mixing parameter $\eta \in [0, 1]$. For $\eta = 0$ one obtains a Ridge-type penalty for the coefficients b_k and for $\eta = 1$, the penalty corresponds to a

LASSO approach (Tibshirani, 1996). For $0 < \eta < 1$, the penalty is a mixture of both, combining the smoothing property of Ridge regression with the variable selection made by LASSO. The algorithm can be extended by an additional screening step, retaining only the $K^* < K$ coefficients with the highest variance, following Johnstone and Lu (2009). This number K^*, together with λ and η, can be chosen e.g. by cross-validation.

The main assumption on $\beta(\cdot)$ is sparsity of the coefficients b_k, i.e. that the signal concentrates on a few basis functions. The preselection step further assumes that the non-zero coefficients in b are those corresponding to the highest variation in the observed images x_i.

Data-driven basis functions

Principal component regression, e.g. Müller and Stadtmüller (2005) and Allen (2013), *PCR2D*:
Principal component regression (e.g. Müller and Stadtmüller, 2005, for the functional case) expands the unknown function $\beta(\cdot)$ in principal components (functions or images), that are obtained from the data. They represent orthogonal modes of variation in the data and thus provide the most parsimonious representation of the data in terms of the number of basis functions needed to explain a given degree of variation in the data. Expanding the data and the unknown $\beta(\cdot)$ in the same K leading principal component functions and making use of their orthonormality yields (6.4) with covariates $\theta_{i,k} = \xi_{i,k}$ (individual principal component scores for each observation and each principal component) and coefficients b_k to be estimated. In most cases the total number of unknown variables $p + K$ will be much smaller than the number of observations N, thus the model is identifiable. Allen (2013) proposes an estimation algorithm to obtain smooth principal component images based on the CANDECOMP/PARAFRAC (CP) representation of a tensor $X \in \mathbb{R}^{N \times L_x \times L_y}$, containing all observed image covariates x_i in a "row-wise" manner

$$X = \sum_{k=1}^{K} d_k \cdot u_k \circ v_k \circ w_k$$

with weights $d_k \in \mathbb{R}$, norm-one vectors $u_k \in \mathbb{R}^N$, $v_k \in \mathbb{R}^{L_x}$, $w_k \in \mathbb{R}^{L_y}$ and $\circ$ the outer product. The expression $v_k \circ w_k \in \mathbb{R}^{L_x \times L_y}$ can be interpreted as eigenimages and $\xi_{i,k} = d_k \cdot u_{k,i}$ as the individual scores for image i and eigenimage k. The approach of Allen includes a smoothness penalty along all axes of X, controlled by smoothing parameters λ_u, λ_v and λ_w. Optimal values for these parameters can be found by a generalized cross-validation criterion following Allen (2013) and Huang, Shen, et al. (2009). Note that in the case of images, it makes sense to set $\lambda_u = 0$, as otherwise this would smooth along the observations $i = 1, \ldots, N$.

There are thus three main assumptions on the coefficient function $\beta(\cdot)$: First, it is assumed that $\beta(\cdot)$ is a linear combination of the first K principal components, i.e. that $\beta(\cdot)$ shares the same modes of variation as the observed images x_i. Second, the eigenfunctions are assumed to be representable as outer vector products $v_k \circ w_k$ in the case of Allen (2013). Third, the eigenfunctions are smoothed, i.e. $\beta(\cdot)$ is assumed to be a smooth function.

Combined Methods

The following methods combine a basis function expansion of $\beta(\cdot)$ with a subsequent data-dependent dimension reduction based on principal component analysis or partial least squares.

Principal component regression based on splines, Reiss and Ogden (2010), *FPCR*: As in Marx and Eilers (2005) $\beta(\cdot)$ is expanded in a spline basis and a smoothness penalty on the coefficients b is added to impose smoothness on $\beta(\cdot)$. The least squares criterion to minimize thus becomes

$$\|y - W\alpha - XBb\|_2^2 + \lambda b^\top P b \to \min_{\alpha, b} \tag{6.5}$$

with $B \in \mathbb{R}^{L \times K}$ the matrix of the basis functions $B_1, \ldots, B_K$ evaluated on the observation grid $\{t_1, \ldots, t_L\}$, $\lambda > 0$ a regularization parameter and $P \in \mathbb{R}^{K \times K}$ an appropriate penalty matrix, e.g. for penalizing first differences. This corresponds to a penalized linear model with design matrix XB for the coefficients b. In a next step, the singular value decomposition of XB is calculated: $XB = UDV^\top$ with $V \in \mathbb{R}^{K \times K}$ containing the principal components of XB. Reiss and Ogden (2010) then assume b to lie in the span of the leading $K_0 < K$ principal components of XB, i.e. $b = V_0 \tilde{b}$ with $V_0 \in \mathbb{R}^{K \times K_0}$ the matrix containing the first K_0 columns of V. Then (6.5) can be written as a model in $\tilde{b}$

$$\left\|y - W\alpha - XBV_0\tilde{b}\right\|_2^2 + \lambda \tilde{b} V_0^\top P V_0 \tilde{b} \to \min_{\alpha, \tilde{b}}.$$

Usually, K_0 will be much smaller than K, which makes the model identifiable in $\tilde{b}$ if $K_0 < n$. Once an estimate for $\tilde{b}$ is found, the estimated coefficient image is given by $\hat{\beta} = BV_0\tilde{b}$. Reiss and Ogden (2010) propose to find the optimal smoothing parameter λ via generalized cross-validation (GCV), the Akaike information criterion (AIC) or REML. The optimal number of principal components K_0 can be found e.g. by cross-validation.

In this approach, the coefficient function $\beta(\cdot)$ is assumed to lie in the span of a given spline basis with coefficients b and to be a smooth function, which is induced by a smoothness penalty. Moreover, the coefficient vector is assumed to lie in the span of the leading principal components of the matrix XB.

Principal component regression in wavelet space, Reiss, Huo, et al. (2015), *WCR*:
The principal-component based wavelet method in Reiss, Huo, et al. (2015) proposes to transform the unknown coefficient function $\beta(\cdot)$ to the wavelet space with coefficients $b = (b_1, \ldots, b_K)$. As the wavelet basis is orthonormal, this is equivalent to transforming the observed images x_i to the wavelet domain, giving coefficients $\theta_{i,k},\ k = 1, \ldots, K$ and solving (6.4). In a subsequent screening step, only the K^* coefficients $\theta_{i,k}$ with the highest sample-variance across the images are retained, giving a matrix $X^* \in \mathbb{R}^{N \times K^*}$ and the corresponding vector of unknown coefficients $b^* \in \mathbb{R}^{K^*}$ (cf. *WNET*). Next, the singular value decomposition of $X^* = U^* D^* {V^*}^\top$ is calculated with $V^* \in \mathbb{R}^{K^* \times K^*}$ containing the principal components of X^*. It is then assumed that b^* lies in the span of the first K_0 principal components of X^*, i.e. $b^* = V_0^* \tilde{b}^*$ with V_0^* the matrix containing the first K_0 columns of V^* as in the spline-based approach. An estimate for $\tilde{b}^*$ can be found by minimizing

$$\left\| y - W\alpha - X^* V_0^* \tilde{b}^* \right\|_2^2 \to \min_{\alpha, \tilde{b}^*}.$$

Given the estimated values $\tilde{b}^*$, the estimated coefficient function $\hat{\beta}(\cdot)$ can be obtained by calculating $b^* = V_0^* \tilde{b}^*$, setting all other coefficients in b to zero and retransforming b to the original space. The number K^* of wavelet coefficients to retain as well as the number K_0 of principal components can be chosen by cross-validation.

The coefficient function $\beta(\cdot)$ here is assumed to be representable by given wavelet basis functions, where only a small number K^* of wavelet coefficients are assumed to be non-zero, notably those coefficients which have the highest variation in the images (see also the wavelet approach with elastic net). Moreover, the coefficient vector is assumed to lie in the span of the leading principal components of the non-zero wavelet coefficients of the images.

Partial least squares in wavelet space, Reiss, Huo, et al. (2015), *WPLS*:
A variant of the last method is presented in Reiss, Huo, et al. (2015), where principal component analysis is replaced by partial least squares. While principal component analysis focuses on the most important modes of variation in the covariate images x_i or their wavelet coefficients $\theta_{i,k}$, respectively, partial least squares finds the components in x_i that are most relevant for predicting the outcome y_i. The approach thus transforms the images to the wavelet space and retains the K^* coefficients with the highest covariance with the response, giving a matrix $X^* \in \mathbb{R}^{N \times K^*}$. Next, the K_0 leading partial least squares components of the remaining coefficients and the response y_i are calculated and stored in a matrix R_0^*. The resulting least squares criterion is

$$\left\| y - W\alpha - X^* R_0^* \tilde{b}^* \right\|_2^2 \to \min_{\alpha, \tilde{b}^*}$$

Given an estimate for $\tilde{b}^*$, the non-zero coefficients b^* of $\beta(\cdot)$ in the wavelet space are given by $R_0^*\tilde{b}^*$. As in the principal component version, the number K^* of wavelet coefficients to retain and the number K_0 of PLS components can be chosen by cross-validation.

Similarly to the previous approach, $\beta(\cdot)$ is assumed to lie in the span of wavelets with a sparse coefficient vector b, having non-zero values only for those entries where the corresponding wavelet coefficients of the images have the highest covariation with the response. Moreover, the non-zero coefficients b^* are assumed to lie in the span of the leading principal least squares components of the corresponding wavelet coefficients $\theta_{i,k}$ of the observed images x_i with the response.

6.2.3. Random Field Methods

Random fields model the coefficient image β in a Bayesian framework. In contrast to the basis function approaches, β is modeled directly on the pixel level, i.e. the unknown coefficient is $\beta = (\beta_1, \ldots, \beta_L)$. Following the Bayesian paradigm, one assumes a prior distribution for all variables in model (6.2), assuming that α, β and σ_ε^2 are independent:

$$\begin{aligned} y\big|\alpha, \beta, \sigma_\varepsilon^2 &\sim \mathrm{N}(W\alpha + X\beta, \sigma_\varepsilon^2 I_N) \\ p(\alpha) \propto \mathrm{const} \qquad \beta \sim F_\beta \qquad \sigma_\varepsilon^2 &\sim \mathrm{IG}(\delta_\varepsilon^{(1)}, \delta_\varepsilon^{(2)}) \end{aligned}$$

for some $\delta_\varepsilon^{(1)}, \delta_\varepsilon^{(2)} > 0$. In this case, the full conditionals for α and σ_ε^2 are given by

$$\begin{aligned} \alpha|\cdot &\sim \mathrm{N}\left((W^\top W)^{-1} W^\top (y - X\beta), \sigma_\varepsilon^2 (W^\top W)^{-1}\right) \\ \sigma_\varepsilon^2\Big|\cdot &\sim \mathrm{IG}\left(\delta_\varepsilon^{(1)} + \frac{N}{2}, \delta_\varepsilon^{(2)} + \frac{1}{2}(y - W\alpha - X\beta)^\top (y - W\alpha - X\beta)\right), \end{aligned}$$

i.e. they are known distributions. Samples from the posterior distribution can thus be obtained by a simple Gibbs sampler. The prior F_β for β should be a random field, modeling the spatial dependence between pixels. Ideally, it should yield a relatively simple full conditional in order to facilitate sampling from the posterior.

Gaussian Markov Random Fields, e.g. Besag (1974) and Rue and Held (2005), *GMRF*: A commonly used class of priors for β are (intrinsic) Gaussian Markov Random Fields (GMRF), which can induce smoothness and constitute a conjugate prior for β. The value of β for a pixel l is assumed to depend only on the values of β in the neighbourhood (Markov property), which can be modeled as

$$\beta_l\Big|\beta_{\delta(l)}, \sigma_\beta^2 \sim \mathrm{N}\left(\frac{1}{d_l}\sum_{j\sim l}\beta_j, \frac{\sigma_\beta^2}{d_l}\right)$$

with $d_l = \#\{j = 1, \ldots, L \colon j \sim l\}$ the number of neighbours of l and $\beta_{\delta(l)}$ the set of all neighbouring coefficients, i.e. $\beta_{\delta(l)} = \{\beta_j \colon j \sim l\}$, where $j \sim l$ means that the pixels j and l are neighbours. (cf. Besag, 1974; Rue and Held, 2005). The choice of the neighbourhood thus models the dependence structure in β. The common variance parameter σ_β^2 is again assumed to have an $\text{IG}(\delta_\beta^{(1)}, \delta_\beta^{(2)}), \delta_\beta^{(1)}, \delta_\beta^{(2)} > 0$ distribution, which can be shown to be conjugate in this case. The prior assumption for β can be rewritten in an unconditional form

$$p\left(\beta \middle| \sigma_\beta^2\right) \propto (\sigma_\beta^2)^{-\operatorname{rank}(P)/2} \exp\left(-\frac{1}{2\sigma_\beta^2}\beta^\top P \beta\right)$$

with $P \in \mathbb{R}^{L \times L}$ the neighbourhood matrix with $p_{j,l} = d_l$ for $j = l$, $p_{j,l} = -1$ for $j \sim l$ and $p_{j,l} = 0$ otherwise. This is not a proper distribution, as P does not have full rank ($\operatorname{rank}(P) = L - 1$). However, this prior assumption yields a proper Gaussian full conditional for β if the data contains enough information, and hence samples from the posterior can be drawn by simple Gibbs sampling.

The Bayesian approach with Gaussian Markov random field priors has an interesting correspondence to penalized basis function methods with constant local basis functions $\mathbb{1}_l$ for each pixel, where the Gaussian prior corresponds to the quadratic penalty. The smoothing parameter is given by $\lambda = \frac{\sigma_\varepsilon^2}{\sigma_\beta^2}$.

The assumptions for the Bayesian GMRF models are given in terms of the priors. For the coefficient image β the GMRF prior induces smoothness.

Sparse Gaussian Markov Random Field, Goldsmith, Huang, et al. (2014), *SparseGMRF*:
The sparse GMRF method proposed in Goldsmith, Huang, et al. (2014) adds a variable selection aspect to the *GMRF* model to combine smoothness with sparsity. The basic idea here is that in general, not the full image x_i will show a relevant association with the response and thus major parts of the coefficient image β can be assumed to be zero. At the same time, the non-zero pixels of interest ideally should form smooth coherent clusters.

In Goldsmith, Huang, et al. (2014), this is modeled by combining the GMRF prior for β with a latent binary Ising prior γ. The corresponding priors are given as follows

$$\beta_l \middle| \beta_{\delta(l)}, \gamma_l, \sigma_\beta^2 \sim \begin{cases} \delta(0) & \gamma_l = 0 \\ \text{N}\left(\frac{1}{d_l}\sum_{j \sim l} \beta_j, \frac{\sigma_\beta^2}{d_l}\right) & \gamma_l = 1 \end{cases}$$
$$\gamma \sim \text{Ising}(a, b)$$

with $\delta(0)$ the Dirac measure centered at 0. The sparse GMRF model thus has an additional level γ in the hierarchical Bayesian model structure with parameters a and b. Depending on the value of the Ising field γ in a pixel l, the corresponding β coefficient is either set

to 0 (if $\gamma_l = 0$, pixel is not selected) or follows the GMRF prior distribution (if $\gamma_l = 1$, i.e. pixel is selected). Goldsmith, Huang, et al. (2014) show that samples from the joint full conditional of β and γ can be obtained by single-site Gibbs sampling. The authors propose to choose the hyperparameters $\sigma^2_\varepsilon, \sigma^2_\beta, a$ and b via cross-validation with extremely short MCMC chains (e.g. 250 iterations).

This model assumes the true β image to be sparse with a few coherent smooth areas of non-zero pixels, which is modeled by a combination of a GMRF and a latent Ising field.

6.2.4. Implementations

For most of the considered approaches, software implementations are available in existing `R`-packages or easily made available. This was one of the inclusion criteria.

The spline regression model (*Splines*) can be fit using the `gam` function for generalized additive models in the `R`-package `mgcv` (Wood, 2011; Wood, 2017). The implementation is very flexible and can handle 2D, 3D or even higher dimensional data and many different basis functions.

All wavelet-based approaches (*WCR*, *WPLS* and *WNET*) are implemented in the `refund.wave` package (Huo et al., 2014, *WCR* and *WPLS* in `wcr`, using different options and *WNET* in `wnet`), heavily building on the `wavethresh` package (Nason, 2016) for calculating the transformations into the wavelet space and the retransformations back to the original space. They all can handle 2D and 3D images with the restriction that the sidelength of the images must be the same power of 2 for all dimensions. For *WNET*, the `glmnet` package (Friedman et al., 2010) is used for the elastic net part.

The principal component regression approach based on splines (*FPCR*) is available in the function `fpcr` in the related package `refund` (Goldsmith, Scheipl, et al., 2016). The implementation currently accepts only 2D images, but without restrictions on the sidelengths of the images.

For the calculation of the eigenimages in *PCR2D* we use the implementation in the `MFPCA` package (Happ, 2017b), which at present works only for 2D images. The scalar-on-image regression model based on the scores can be fit with the standard `lm` function for linear models. The reconstruction of the coefficient image $\hat{\beta}$ using the estimated eigenimages and the regression coefficients can easily be done using the `expandBasisFunction` method in `MFPCA`. In principle, the MFPCA approach (Happ and Greven, 2017+) can be used to calculate eigenimages, too, interpreting the images as multivariate functional data with a single element on a two- or three-dimensional domain.

For *SparseGMRF*, Goldsmith, Huang, et al. (2014) provide an `R` implementation of the Gibbs sampler in the supplementary files. As the code turns out to be quite slow for larger

images, we use our own `C` implementation of the Gibbs sampler and implement the cross-validation in `R`. For the fully Bayesian *GMRF* model, we use a variant of our `C` code for *SparseGMRF*, without the latent Ising field γ, but including Gibbs sampling steps for the variance parameters σ_ε^2 and σ_β^2. Both models are currently implemented only for 2D images, but can easily be extended to the 3D case. The most important aspect here is to properly define the neighbourhood in the three-dimensional case.

Usage examples for all methods are given in the code supplement of this article (`https://github.com/ClaraHapp/SOIR`).

6.3. Discussion and Measures for Model Assumptions

As discussed in Section 6.2.1, the scalar-on-image regression model (6.1) in general is not identifiable, as the total number of model coefficients in most applications exceeds the number of observation units. It is therefore necessary to make structural assumptions on β to overcome the issue of non-identifiability and make estimation possible. However, all assumptions come at a price, as the estimated coefficient image will reflect the model assumptions, e.g. in terms of smoothness. It is hence important to be aware of the assumptions made and to understand how they influence the estimate.

6.3.1. Underlying and Parametric Model Assumptions

In the following, we distinguish between underlying and parametric model assumptions. The underlying assumptions describe the fundamental model assumptions, such as smoothness or sparsity, and the general class of coefficient images that a model can handle, e.g. linear combinations of splines or wavelets. The parametric model assumptions reflect restrictions of the model parameters in the estimation process, in terms of penalties or variable selection.

For the discussed models, the underlying and parametric model assumptions can be broadly divided into three categories each (cf. Tables 6.1 and 6.2). For the underlying assumptions, there are 1. smoothness, meaning that neighbouring pixels have similar values, 2. sparsity, that is a few coefficients dominate all others and 3. projection, which reflects the assumption that the true coefficient image β can be expanded in a finite number of given basis functions. For the parametric model assumptions, we have 1. smoothness via quadratic difference penalties of neighbouring coefficients, 2. sparsity via variable selection and 3. assumptions on the variability of β (σ_β^2 in the Bayesian GMRF based models).

If the true β image fulfils the underlying model assumptions, then the model should be able to find it, if enough data is provided. If the true β is not in the class of coefficient images

Table 6.1.: Underlying model assumptions for the considered models. The order of the models has been slightly rearranged with respect to the presentation in Section 6.2 according to their assumptions.

Method	Smoothness	Sparsity	Projection
Splines	image	-	spline basis
FPCR	image	PCs of XB	spline basis
PCR2D	-	PCs of images	PCs of images
WCR	-	wavelet coefficients	wavelet space
WPLS	-	wavelet coefficients	wavelet space
WNET	-	wavelet coefficients	wavelet space
SparseGMRF	image	pixels	-
GMRF	image	-	-

Table 6.2.: Parametric assumptions for the considered models.

Method	Smoothness (Penalty/Prior)	Sparsity (Variable Selection)	Prior Variability
Splines	spline coefficients	-	-
FPCR	spline coefficients	PCs of XB	-
PCR2D	-	PCs of images	-
WCR	-	wavelet coefficients & PCs	-
WPLS	-	wavelet coefficients & PLSCs	-
WNET	-	wavelet coefficients	-
SparseGMRF	pixels	pixels via Ising field	CV for σ^2_β
GMRF	pixels	-	IG-prior on σ^2_β

defined by the model, then the result will be an approximation of the true β within the given class, i.e. the underlying assumptions will introduce some sort of bias. Strong model assumptions will restrict the class of "estimable" coefficient images considerably and thus will in general lead to a stronger bias than mild assumptions. In general, this bias cannot be detected from in-sample prediction error due to non-identifiability of β, as different estimates $\hat{\beta}$ can give equally good predictions. Therefore, widely used methods such as cross-validation can not provide protection against this issue. It is therefore important to develop measures that quantify how well the underlying and parametric model assumptions are met, in order to understand how strongly the assumptions affect the estimate.

6.3.2. Measures for Quantifying the Impact of Model Assumptions

In the following, we develop measures for the underlying and parametric model assumptions discussed before. For better comparability, all measures take values between 0 and 1 with 0 meaning that the model assumptions are perfectly met and 1 meaning that the assumptions are not met at all.

Smoothness: Here we interpret smoothness as neighbouring pixels having similar values. The sum of squared differences between neighbours can thus be used as a measure of smoothness. For $\beta \in \mathbb{R}^L$ with a given neighbourhood structure (e.g. first differences), a natural smoothness measure is

$$\tilde{m}_{\text{Smoothness}}(\beta) = \sum_{i \sim j} (\beta_i - \beta_j)^2 = \beta^\top P \beta$$

for the symmetric and positive semidefinite neighbourhood matrix $P \in \mathbb{R}^{L \times L}$ (see GMRF in Section 6.2.3). By the theorem of Rayleigh-Ritz (Horn and Johnson, 1985, Thm. 4.2.2), it holds

$$\lambda_{\min}(P) \leq \frac{x^\top P x}{x^\top x} \leq \lambda_{\max}(P) \quad \text{for all } x \in \mathbb{R}^L \setminus \{0\}$$

and the equalities are fulfilled for the eigenvectors of P associated with the minimal and maximal eigenvalues of P, $\lambda_{\min}(P)$ and $\lambda_{\max}(P)$, respectively. For a constant vector x, the value of $\tilde{m}_{\text{Smoothness}}$ will be equal to 0, as P does not have full rank and $\lambda_{\min}(P) = 0$. Rescaling $\tilde{m}_{\text{Smoothness}}$ to

$$m_{\text{Smoothness}}(\beta) = \frac{\beta^\top P \beta}{\lambda_{\max}(P) \beta^\top \beta}$$

yields a smoothness measure between 0 (constant, i.e. extremely smooth image) and 1 (extremely nonsmooth images).

This measure can be used to assess the smoothness of an image as an underlying model assumption and also for the parametric smoothness assumptions made for the GMRF based models. For the approaches using splines, β has to be replaced by the vector of spline coefficients $(b_1, \ldots, b_K)$ and $P \in \mathbb{R}^{K \times K}$ has to be chosen as the associated penalty matrix to measure the parametric smoothness assumption.

Sparsity: Many scalar-on-image regression methods involve underlying sparsity assumptions, e.g. for the coefficient image (Goldsmith, Huang, et al., 2014) or for the wavelet coefficients (e.g. Reiss, Huo, et al., 2015). Hurley and Rickard (2009) compare different sparsity measures for images based on six criteria (*Robin Hood, Scaling, Rising Tide,*

Cloning, Bill Gates, Babies), partly introduced by Dalton (1920) in the context of measuring the inequity of incomes. The measure that fulfils all criteria and hence is a reasonable measure for sparsity of an image $\beta \in \mathbb{R}^L$ is the Gini index

$$G(\beta) = 1 - 2\sum_{l=1}^{L} \frac{\beta_{(l)}}{\|\beta\|_1} \left(\frac{L - l + \frac{1}{2}}{L} \right)$$

with $\beta_{(1)} \leq \beta_{(2)} \leq \ldots \leq \beta_{(L)}$ the ordered values of $|\beta_l|$, $l = 1, \ldots, L$ and $\|\beta\|_1 = \sum_{i=l}^{L} |\beta_l|$. Note that at least one entry of β must be non-zero for $G(\beta)$ to be well defined. For reasons of consistency, we define

$$m_{\text{Sparsity}}(\beta) = 1 - G(\beta)$$

with $m_{\text{Sparsity}}(\beta) = 0$ for complete inequality of β across all pixels (very sparse case) and $m_{\text{Sparsity}}(\beta) = 1$ indicating complete equality of β across all entries (non-sparse case). This measure can also be applied to a coefficient vector $b = (b_1, \ldots, b_K)$, e.g. of wavelet coefficients.

Sparsity can be induced by variable selection methods, setting many coefficients to zero. A measure for a parametric sparsity assumption for a coefficient vector $b \in \mathbb{R}^K$ obtained from a variable selection method is thus given by

$$m_{\text{Selection}}(b) = \frac{\#\{k = 1, \ldots, K : b_k \neq 0\}}{K},$$

i.e. the proportion of variables in b that are not set to zero. The measure takes values between 0 and 1, where 0 means extreme sparsity ($b \equiv 0$) and 1 means no sparsity. It corresponds to a normalized version of the ℓ^0 measure studied in Hurley and Rickard (2009). The sparse GMRF approach (Goldsmith, Huang, et al., 2014) assumes sparsity on the pixel level, i.e. here one can apply $m_{\text{Selection}}$ to the vectorized posterior mean of the Ising field γ, thresholded at 0.5.

Projection: Basis function approaches assume that the function $\beta(\cdot)$ generating the coefficient image lies in the span of some predefined basis functions $B_1, \ldots, B_K$, which can be splines, wavelets or principal component functions. A suitable measure for this assumption is

$$m_{\text{Projection}}(\beta) = \frac{\|P^{\perp}\beta\|^2}{\|\beta\|^2} = 1 - \frac{\|P\beta\|^2}{\|\beta\|^2}$$

with $P\beta$ the orthogonal projection of β onto the space spanned by $B_1, \ldots, B_K$, $P^{\perp}\beta$ the projection onto the orthogonal complement of that space and $\|\beta\|^2 = \sum_{l=1}^{L} \beta_l^2$. By the Pythagorean Theorem, this measure takes values between 0 and 1, where 1 means that β lies completely in the orthogonal complement of the basis functions and $m_{\text{Projection}}(\beta) = 0$, if β is indeed a linear combination of the basis functions.

While methods with given basis functions such as splines or wavelets will usually yield good approximations of the true functions, provided that K is large enough, principal component methods will be more restrictive, as here K is usually small and thus β is assumed to have similar modes of variation as the data. For a given model, the estimate $\hat{\beta}$ will always lie in the given model class, i.e. this measure for underlying assumptions makes sense only for the true β, which in general is not available. However, we will use this measure in the simulation in Section 6.4.2.

Prior variability: For Bayesian methods, parametric assumptions on the parameters are formulated as priors. In hierarchical models, one often defines independent or conditional priors for the individual parameters. The so-called full conditionals are the corresponding conditional posterior distributions of one parameter given all others and the data. It is thus natural to use the relation between the (conditional) prior of a parameter and its full conditional posterior as a measure for the prior impact. The Kullback-Leibler divergence is one possible way to measure the distance of a distribution with respect to a reference distribution. It has already been applied to the Bayesian case in the theory of Bayesian surprise (see e.g. Itti and Baldi, 2005), but for prior and posterior densities and not for the conditional case.

For the *GMRF* model, the prior variance of β, σ_β^2, has an inverse gamma prior. It is thus conjugate and leads to an inverse gamma full conditional. We choose the full conditional as the reference and calculate the Kullback-Leibler divergence to the prior, which yields

$$D = \log\left(\frac{b_{\text{post}}^{a_{\text{post}}}}{b_{\text{pri}}^{a_{\text{pri}}}}\right) + \log\left(\frac{\Gamma(a_{\text{pri}})}{\Gamma(a_{\text{post}})}\right) + (a_{\text{pri}} - a_{\text{post}})[\log(b_{\text{post}}) - \psi(a_{\text{post}})] + (b_{\text{pri}} - b_{\text{post}})\frac{a_{\text{post}}}{b_{\text{post}}}$$

with f_{pri} the density of the prior ($\text{IG}(a_{\text{pri}}, b_{\text{pri}})$), f_{post} the density of the full conditional ($\text{IG}(a_{\text{post}}, b_{\text{post}})$) and ψ the digamma function.

For the *SparseGMRF* model in Goldsmith, Huang, et al. (2014), σ_β^2 is chosen via cross-validation. Here, the prior can be seen as a discrete uniform distribution on the set of possible values $\{\sigma_1^2, \ldots, \sigma_K^2\}$ for σ_β^2, i.e. $f_{\text{pri}}(\sigma_\beta^2) = \frac{1}{K}\sum_{k=1}^{K} \mathbb{1}\{\sigma_\beta^2 = \sigma_k^2\}$ and the full conditional is a point measure on the optimal value σ_*^2 found by cross-validation: $f_{\text{post}}(\sigma_\beta^2) = \mathbb{1}\{\sigma_\beta^2 = \sigma_*^2\}$. For the Kullback-Leibler divergence, one obtains

$$D = \log\left(\frac{f_{\text{post}}(\sigma_*^2)}{f_{\text{pri}}(\sigma_*^2)}\right) f_{\text{post}}(\sigma_*^2) = \log(K).$$

The resulting distance D between prior and posterior grows logarithmically with K: increasing the number of possible values from 5 to 10, say, will have a much stronger impact than an increase from 105 to 110.

For numerical reasons, we divide D by 10 and transform the result to $[0, 1]$, which gives the measure for the prior variability

$$m_{\text{Prior}}(\beta) = 1 - \exp(-D/10).$$

Values close to 1 correspond to $D \to \infty$, i.e. situations where the information from the prior has little influence on the full conditional and thus model assumptions will in general not be met. By contrast, values close to 0 correspond to $D \approx 0$, hence prior and full conditional are very similar, which means that the full conditional fulfils the prior assumptions.

As mentioned, $m_{\text{Projection}}$ is sensible only for the true coefficient image, while m_{Prior} also requires information from the likelihood, and hence can be calculated only for the estimate $\hat{\beta}$. The other measures, $m_{\text{Smoothness}}$, m_{Sparsity} and $m_{\text{Selection}}$ can be used for true as well as for estimated coefficient images, even though $m_{\text{Selection}}$ mainly aims at estimates which are the result of a variable selection.

The measures are interpretable in the sense that they all range between 0 and 1 with values near 0 meaning that the coefficient image is very close to the assumptions made and values of approximately 1 that the coefficient image is far from the assumptions. In order to interpret the absolute values in practice, it might be helpful to create some hypothetic coefficient images, which should be motivated by the question of interest. The measures for the coefficient image obtained from the real data can then be compared, either directly to the measures for the hypothetic image or to measures for coefficient image estimates based on simulated data. For the latter, one could for example use the original image covariates x_i and the hypothetic coefficient images to generate new response values, similarly to the setting in our simulation study.

6.4. Simulation Study

In this section, the performance of different scalar-on-image regression approaches is analyzed for various coefficient images β, reflecting the assumptions in the different models, and real data from the Alzheimer's Disease Neuroimaging Initiative study (ADNI, Mueller et al., 2005; Weiner et al., 2015) that are also considered in the application in Section 6.5. On the one hand, this ensures that the image covariates have a realistic degree of complexity (cf. Reiss and Ogden, 2007; Reiss and Ogden, 2010, for similar settings). On the other hand, this allows to study systematically which features in the coefficient images can be found with the data at hand. We consider four different coefficient images (see Fig. 6.1):

- *bumpy*, an image with some high-peaked, clearly defined "bumps", as proposed in Reiss, Huo, et al. (2015) as a two-dimensional version of the *bump* function of Donoho

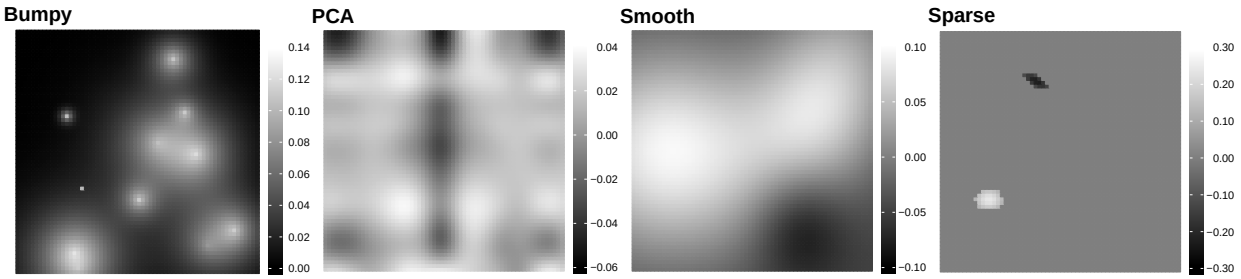


Figure 6.1.: Coefficient images β used for the simulation. From left to right: *bumpy*, *pca* (based on the first $N = 250$ images in the dataset), *smooth* and *sparse*. Note the individual scale for each image.

and Johnstone (1994), which has become a common benchmark for one-dimensional wavelet models. It is thus expected that the wavelet-based methods should be the most suitable ones for estimation.

- *pca*, an image constructed as a linear combination of the first $K = 5$ empirical principal components of the image covariates $x_1, \ldots, x_N$ with coefficients $b_k = (-1)^k \exp(-\frac{k}{5})$, $k = 1, \ldots, 5$. Obviously, the principal component based model should work very well in this case.

- *smooth*, a smooth image which corresponds to the smoothness assumption made in the spline-based models and for the Bayesian models using Gaussian Markov random fields. It is constructed as a mixture of three 2D normal densities.

- *sparse*, an image that is mostly zero with two small, smooth spikes (Goldsmith, Huang, et al., 2014). This image corresponds to the assumption made for the *SparseGMRF* model. It is a mixture of two normal densities, cut off at a certain threshold, setting all pixels with absolute values below this threshold to zero.

The performance of the different estimation methods is evaluated for all four coefficient images and for different sample sizes and signal-to-noise ratios. A sensitivity study with varying coefficient images gave similar results, showing that the spatial distribution of features in the coefficient images has only a marginal impact on the results (see appendix, Section D.4).

The image covariates stem from FDG-PET scans, which measure the glucose uptake in the brain. The original scans were co-registered to simultaneously measured MRI scans in order to reduce registration effects (Araque Caballero et al., 2015). We use 64×64 subimages of the first $N = 250$ or $N = 500$ images in the original data as covariates $x_1, \ldots, x_N$, see Fig. 6.2. The image size is determined by the wavelet-based methods, which require the

sidelength of the images to be a power of 2. The demeaned images take values between -1 and 1.24.

The response is constructed as

$$y_i = \alpha + \sum_{l=1}^{L} x_{i,l}\beta_l + \varepsilon_i, \quad i = 1, \ldots, N$$

with $\alpha = -1$ as intercept, a total number of $L = 64^2 = 4096$ pixels and ε_i chosen such that the signal-to-noise ratio

$$\text{SNR} = \frac{\widehat{\text{sd}}(\sum_{l=1}^{L} x_{i,l}\beta_l)}{\text{sd}(\varepsilon_i)}$$

is either equal to 4 or to 1 (see e.g. Goldsmith, Huang, et al., 2014, for an analogous approach), which corresponds to $R^2 = 0.94$ and 0.5 (cf. Reiss and Ogden, 2007). For each setting, the simulation and analysis is repeated 100 times, fitting in total nine different models to the data pairs $\{(x_i, y_i),\ i = 1, \ldots, N\}$.

The resulting estimates $\hat{\beta}$ and the fitted values

$$\hat{y}_i = \hat{\alpha} + \sum_{l=1}^{L} x_{i,l}\hat{\beta}_l, \quad i = 1, \ldots, N$$

are evaluated with respect to the relative estimation accuracy and the relative (in-sample) prediction error

$$\frac{\sum_{i=l}^{L}(\beta_l - \hat{\beta}_l)^2}{\sum_{i=l}^{L}(\beta_l - \bar{\beta})^2} \qquad \text{and} \qquad \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}.$$

with $\bar{\beta} = \frac{1}{L}\sum_{l=1}^{L}\beta_l$ and $\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$. Taking the relative errors allows to compare the results across coefficient images β and datasets $\{(x_i, y_i),\ i = 1, \ldots, N\}$ generated in different iterations of the study. A relative estimation error of 1 means that the estimated coefficient image gives equally good results as a constant image, taking the average value of the true β in each pixel. This corresponds to a simpler model in the mean of the image covariate over pixels. Analogously, a relative prediction error of 1 means that the prediction is comparable to a simple intercept model, not taking the image information into account. Relative errors above 1 therefore are indicators for poor performance. In addition, the underlying and parametric model assumptions from Section 6.3 are calculated for each estimate $\hat{\beta}$ and – for the underlying assumptions – compared with those of the true images. As computation time plays an important role for the practical usability of the models, it is also recorded.

6.4.1. Model Settings

Splines: The unknown β image is expanded in $K_x = K_y = 15$ cubic B-spline basis functions in each direction, penalizing the second squared differences of the corresponding coefficients.

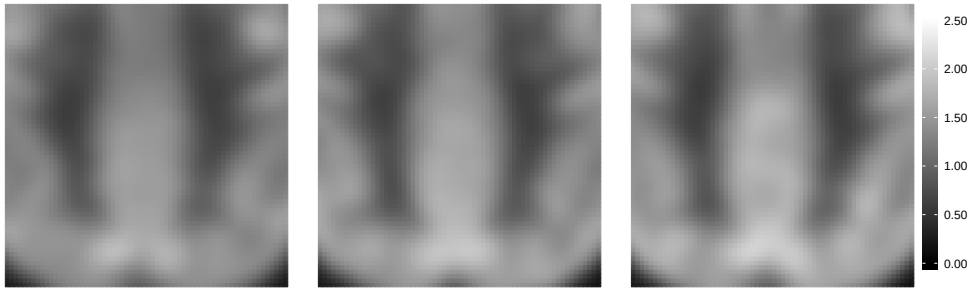


Figure 6.2.: The first three image covariates x_1, x_2, x_3 before demeaning.

The smoothing parameter λ is found via REML. The calculations can be done using the `gam` function in the `R`-package `mgcv` (Wood, 2011; Wood, 2017).

FPCR: As for the pure spline approach we use $K_x = K_y = 15$ basis functions for each marginal and choose the smoothing parameter via REML. The number K_0 of principal components retained for regression is chosen via five-fold cross-validation from $\{5, 10, 25, 50, 100, 150\}$. The model is fit using the function `fpcr` in the `R`-package `refund` (Goldsmith, Scheipl, et al., 2016).

PCR2D: We calculate 25 two-dimensional principal components of the observed images using the approach of Allen (2013) as implemented in the `MFPCA` package (Happ, 2017b) with second difference penalty for smoothing in each direction. The smoothing parameters λ_v, λ_w are chosen via GCV within the boundaries 10^{-4} and 10^2. The response y is regressed on the first $K \in \{1, 5, 10, 15, 20, 25\}$ score vectors to find the coefficients for the unknown coefficient image. An optimal choice of K is found via five-fold cross-validation.

WCR: We use the function `wcr` in the package `refund.wave` (Huo et al., 2014). The observed images are transformed to the wavelet space using Daubechies least-asymmetric orthonormal compactly supported wavelets with 10 vanishing moments. The resolution level M_0 is fixed to 3. Only the $K^* \in \{10, 25, 50, 100, 250, 500, 1000\}$ coefficients having the highest variance are retained. The response y is regressed on the leading $K_0 \in \{5, 10, 15, 25, 50, 75\}$ principal components of the remaining coefficients (restricting $K_0 \leq K^*$) and the result is transformed back to the original space. An optimal combination of K^* and K_0 is found via five-fold cross-validation.

WPLS: The wavelet-based principal least squares method is implemented in the same function `wcr` of the `refund.wave` package, using the option `method = "pls"`. For all parameters (M_0, K^*, K_0) we use the same specifications as for *WCR*.

WNET: Here also we use Daubechies least-asymmetric orthonormal compactly supported wavelets with 10 vanishing moments and a resolution level $M_0 = 3$ to obtain wavelet

coefficients from the observed images. The model is estimated using the `wnet` function in the `R`-package `refund.wave` (Huo et al., 2014). As for the other two wavelet methods, the number of wavelet coefficients that are retained for the regression is chosen from $K^* \in \{10, 25, 50, 100, 250, 500, 1000\}$. For the elastic net part, the mixing parameter η can take values in $\{0, 0.25, 0.5, 0.75, 1\}$, with 0 corresponding to the Ridge penalty and 1 giving the LASSO approach. Candidate values for the penalty parameter λ are automatically generated by the `glmnet` function. An optimal combination of K^* and η is chosen via five-fold cross-validation.

SparseGMRF: A constant prior for α is used. The hyperparameters are chosen via five-fold cross-validation from $a \in \{-4, -2, -0.5\}$, $b \in \{0.1, 0.5, 1.5\}$, $\sigma_\varepsilon^2, \sigma_\beta^2 \in \{10^{-5}, 10^{-3}, 10^{-1}\}$. For each parameter combination and each fold (in total $81 \cdot 5 = 405$ combinations), a short Gibbs sampling is run with 250 iterations, of which 100 are discarded as burnin (no thinning), following the settings in Goldsmith, Huang, et al. (2014). For the starting values, γ_l is sampled randomly from $\{0, 1\}$ and if $\gamma_l = 1$, β_l is sampled from $\mathrm{N}(0, \sigma_\beta^2)$, otherwise $\beta_l = 0$. The pixels are updated in random order.

GMRF: The prior for the unknown coefficient image β is chosen as an intrinsic GMRF with four neighbours and for α a constant prior is used. The priors for the variance parameters σ_ε^2 and σ_β^2 are chosen as conjugate inverse gamma distributions with $\sigma_\varepsilon^2, \sigma_\beta^2 \sim \mathrm{IG}(1, 1)$ (which is considered rather uninformative, but not entirely without controversy, see Gelman (2006); model *GMRF*) and $\sigma_\varepsilon^2, \sigma_\beta^2 \sim \mathrm{IG}(10, 10^{-3})$ (highly informative with a prior mean of 10^{-3} and a prior variance of 10^{-9}; model *GMRF2*). For both models, the Gibbs Sampler is run over 5000 iterations, of which 500 are discarded as burnin and saving each 20th step (thinning). As a starting value, β_l is initialized with $\mathrm{N}(0, \tilde{\sigma}_\beta^2)$ with $\tilde{\sigma}_\beta^2$ the prior mode. The pixels are updated in random order.

6.4.2. Results

Fig. 6.3 shows the results of the simulation study for $N = 250$ in terms of the relative prediction and estimation error. The computing time in seconds is shown in Fig. 6.4. Example plots for estimates and predictions of all models and all coefficient images can be found in the appendix, Section D.1. Boxplots of the measures for underlying and parametric model assumptions developed in Section 6.3 are shown in Fig. 6.5 (underlying assumptions, including measures for the true coefficient images) and Fig. 6.6 (parametric assumptions). The corresponding results for $N = 500$ are shown in the appendix, Section D.5.

The predictive accuracy for the different coefficient images is rather constant over all models (except *GMRF*) with values around 0.5 for SNR = 1, which means that the performance is somewhat better than a simple intercept model, and values close to 0.05 for SNR = 4, i.e. the models clearly perform better than the intercept model. This means that if the focus is

only on prediction, the different models and their assumptions lead to equally good results. The scalar-on-image regression model, however, also aims at an interpretable coefficient image $\hat{\beta}$, showing how the observed image covariates x_i influence the response y_i. The relative estimation error is thus of greater importance for assessing a model's ability of giving interpretable results.

As seen in Fig. 6.3 for $N = 250$ and similarly also in Fig. D.15 in the appendix for $N = 500$, the relative estimation errors vary around 1 for SNR = 1. Exceptions are found only for *pca* and *PCR2D* and *smooth* and *Splines, FPCR, SparseGMRF* and *GMRF2*, hence settings in which the true coefficient image meets the models assumptions very well. For SNR = 4, the errors become smaller, but particularly for *bumpy* and *sparse*, i.e. coefficient images with highly localized features, the methods still result in relatively high error rates. The median estimation error over all methods (except *GMRF*) for $N = 250$ and SNR = 4 are 0.70 (*bumpy*), 0.45 (*pca*), 0.28 (*smooth*) and 0.74 (*sparse*). Overall, *FPCR* gives the best results, as it is always among the best two models in terms of estimation accuracy and also by far the model with the shortest computation time. By contrast, the *GMRF* model with the IG(1, 1) prior is clearly seen to have the worst estimation accuracy of all models, yielding relative errors roughly around 100, i.e. it performs considerably worse than a simple intercept model. The *SparseGMRF* model is much slower than the other models, requiring around 85% of the total computation time of the study, although a relatively simple setting was chosen with only three possible values for each hyperparameter.

For the *pca* coefficient image, the *PCR2D* model gives the best results, as expected. This is due to the fact that the true coefficient image is completely spanned by the leading $K = 5$ principal components of the images (see the $m_{\text{Projection}}$ value in Table 6.3), and thus the model assumptions are perfectly met in *PCR2D*. It is followed by the other principal component methods, *FPCR* and *WCR*. Their measures for the parametric model assumptions show that a rather small proportion of principal components (in the spline or wavelet space) is used to calculate the estimate. For the remaining models, the smoothness assuming approaches perform mostly better than the wavelet-based methods *WPLS* and *WNET*. However, the estimates are relatively unsmooth, which is seen in the higher measures for smoothness assumptions.

Similarly, the *smooth* coefficient image is estimated best by the spline-based methods (*Splines*, *FPCR*) that assume smoothness, as the image can be represented well in the spline space (see Table 6.3). The underlying smoothness measures for both models are very low, indicating that the estimated images are actually smooth. They also have a high correlation with the true coefficient image (Fig. 6.7). The Bayesian *GMRF2* and *SparseGMRF* models, that also assume smoothness of the coefficient image, perform slightly worse, as here the smoothness is assumed on a pixelwise level and thus is more local. Notably, the estimation error for *SparseGMRF* becomes higher with an increasing signal-to-noise ratio. The wavelet methods, particularly *WPLS* and *WNET*, give rather poor results for SNR = 1.

The sparsity assumption measures show that the estimated images are too sparse in the wavelet space and thus the sparsity assumption dominates in the estimates.

For the *bumpy* image, the result is more surprising. One would expect the wavelet-based methods to perform best, as argued in Reiss, Huo, et al. (2015). By contrast, all wavelet methods are outperformed by *FPCR* and *GMRF2*, hence two methods that assume smoothness. This, however, is in line with the results of Reiss, Huo, et al. (2015), who found that wavelet methods did not clearly outperform non-wavelet methods for the *bumpy* coefficient image when compared to *sparse*. The measures for the underlying model assumptions give an explanation for this result: The *bumpy* image can be perfectly projected into the wavelet space, just as all coefficient images can (see Table 6.3), but has a similar sparsity in the wavelet space as *pca* and *smooth*, i.e. the sparsity assumption in *WCR*, *WPLS* and especially *WNET* has no advantage for the estimation. The smoothness measures for *FPCR* and *GMRF2* show that the resulting estimates are a bit too smooth, but they still yield better results than the wavelet-based methods.

The *sparse* coefficient image is the most difficult to estimate, as it has two rather spiky features and the rest of the image is equal to zero. Indeed, all models (except *GMRF*) have relative estimation errors close to 1, which means that the methods perform similarly as a pure intercept model which simply ignores the non-zero pixels in the image. Contrary to expectation, the *SparseGMRF* model does not clearly perform better than other models, although it involves a variable selection step and hence the possibility to set entire areas of the image to zero. The model measures for parametric assumptions show that it produces estimates that are smooth, but completely non-sparse (sparsity measure is approximately 1), which means that the sparsity assumption is more or less ignored in the estimation process. *SparseGMRF* hence behaves more as a non-sparse GMRF with the variance parameters chosen via cross-validation. This has also a high impact on the result, as in the simulation there were only three possible values for σ_β^2 due to computational reasons.

In order to check the agreement of the estimated coefficient images among each other and with the true β, correlations of the vectorized images were calculated (median for SNR = 4 is given in Fig. 6.7). They show that the smoothness inducing models (*Splines*, *FPCR* and *GMRF2*) are highly correlated for all coefficient images β, and also with the true *smooth* image. For the wavelet-based models, there is a high correlation between *WPLS* and *WNET* for *bumpy, pca* and *smooth*, whereas for *sparse*, *WNET* aims at an even more sparse representation in the wavelet space and here *WCR* and *WPLS* are highly correlated. Notably, for *sparse*, all estimated coefficient images show medium to high correlation among themselves, but rather low correlation with the true coefficient image. The Bayesian *GMRF*/*GMRF2* models show that the choice of the prior for the variance parameters matters, as *GMRF2* gives reasonably good estimates, while *GMRF* by far has the worst results. When looking at the corresponding measure for the prior, the *GMRF2* model with the highly informative prior for σ_β^2 yields much higher values than the model with the less informative prior

(*GMRF*). This seems reasonable, as the prior assumptions in the highly informative model are much stricter and therefore more difficult to meet than the uninformative ones in *GMRF*. The measure for *SparseGMRF* is quite low, reflecting that σ_β^2 is chosen via cross-validation (cf. Section 6.3.2).

In summary, the simulation results show that the assumptions made for the different models can lead to different results with different estimation quality. The estimation accuracy varies depending on the structure of the true coefficient image and the model, while the predictive performance is quite similar over all models. This shows that the different models can give equally good predictions while at the same time they result in considerably different estimates. This of course affects the interpretability of the results. For a higher SNR, the relative errors decrease, both for estimation and prediction, meaning that more information in the data leads to better results over all model classes and all coefficient images. Overall, *FPCR* seems to give the best results in this simulation. In particular, the combination of a spline basis representation and a principal component analysis for XB appears to be advantageous compared to the pure *Splines* models. Similarly, *WCR* performs better than the other wavelet basis methods in all settings considered in this study. As expected, *PCR2D* clearly outperforms all other methods for the *pca* coefficient image, which perfectly meets the assumptions made in this model. For all other coefficient images, *PCR2D* gives intermediate results. Finally, for the GMRF based models, the highly informative *GMRF2* model performs best, followed by *SparseGMRF*, which, however, does not make use of the integrated variable selection and is computationally very demanding. The *GMRF* model with an uninformative prior overall gives very poor results, performing substantially worse than an intercept model (relative prediction error > 1) or a model with a constant coefficient image (relative estimation error > 1).

Table 6.3.: Values of $m_{\text{Projection}}$ for the different coefficient images depending on the basis functions used.

	Coefficient Image			
Projection on	*bumpy*	*pca*	*smooth*	*sparse*
PCs	$1.22 \cdot 10^{-01}$	$3.65 \cdot 10^{-30}$	$7.05 \cdot 10^{-02}$	$8.54 \cdot 10^{-01}$
splines	$4.16 \cdot 10^{-03}$	$3.79 \cdot 10^{-04}$	$3.22 \cdot 10^{-08}$	$5.03 \cdot 10^{-01}$
wavelets	$1.73 \cdot 10^{-18}$	$2.43 \cdot 10^{-18}$	$2.36 \cdot 10^{-18}$	$4.39 \cdot 10^{-18}$

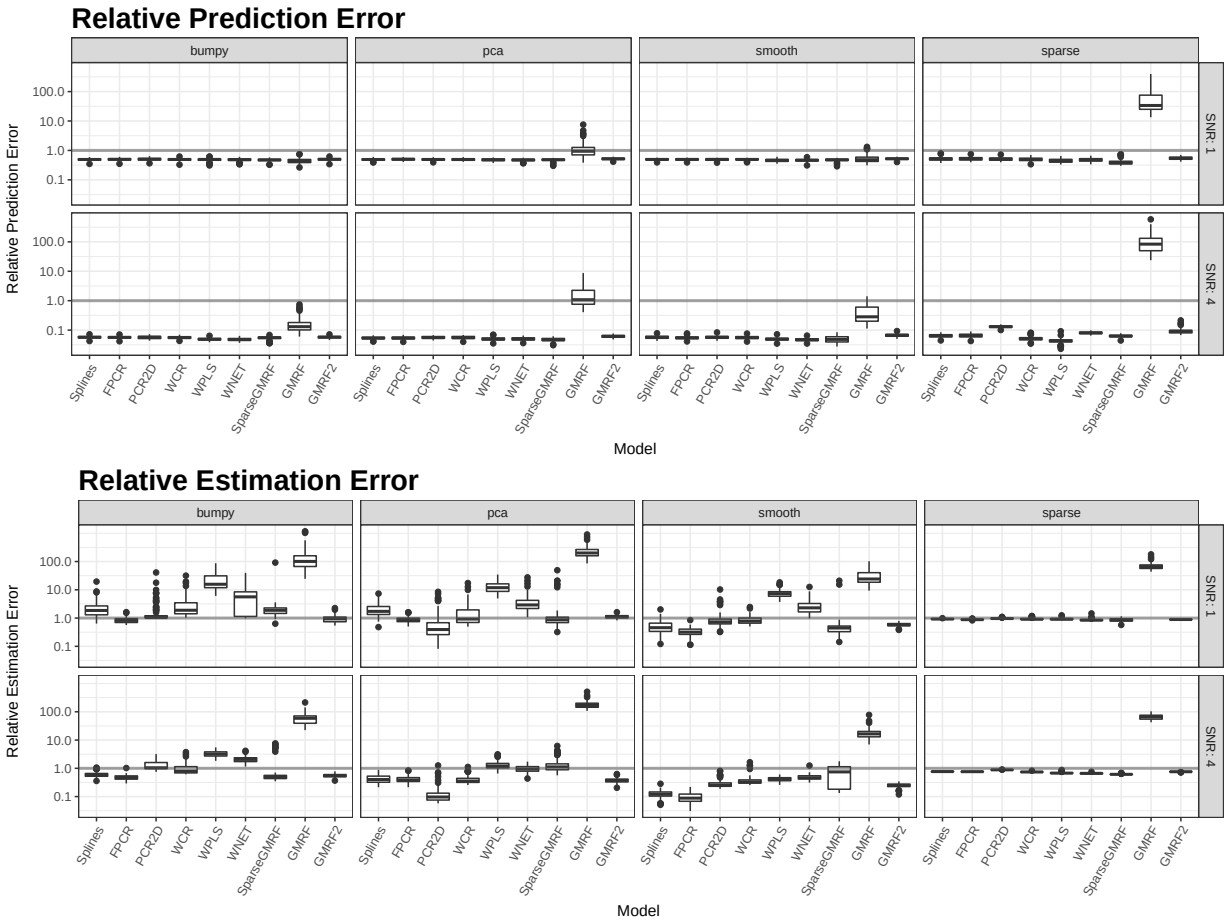


Figure 6.3.: Simulation results for $N = 250$ observations. Boxplots show the relative prediction and estimation error for all nine models depending on the coefficient image and the signal-to-noise ratio (SNR) over all 100 simulation runs. Gray horizontal lines mark 1, which corresponds to the simple intercept model (for prediction error) or to a constant coefficient image, having the average value of the true β image (estimation error).

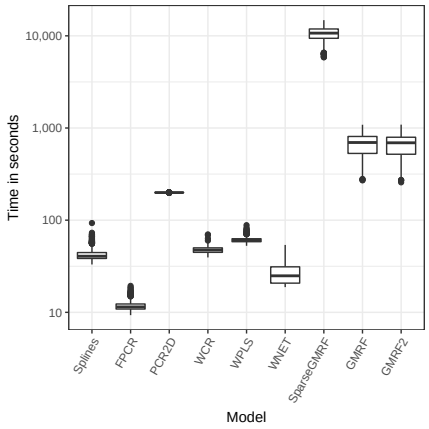


Figure 6.4.: Computation times per fit for all nine models and $N = 250$ observations. The boxplots contain the merged values for all coefficient images and signal-to-noise ratios over all 100 simulation runs.

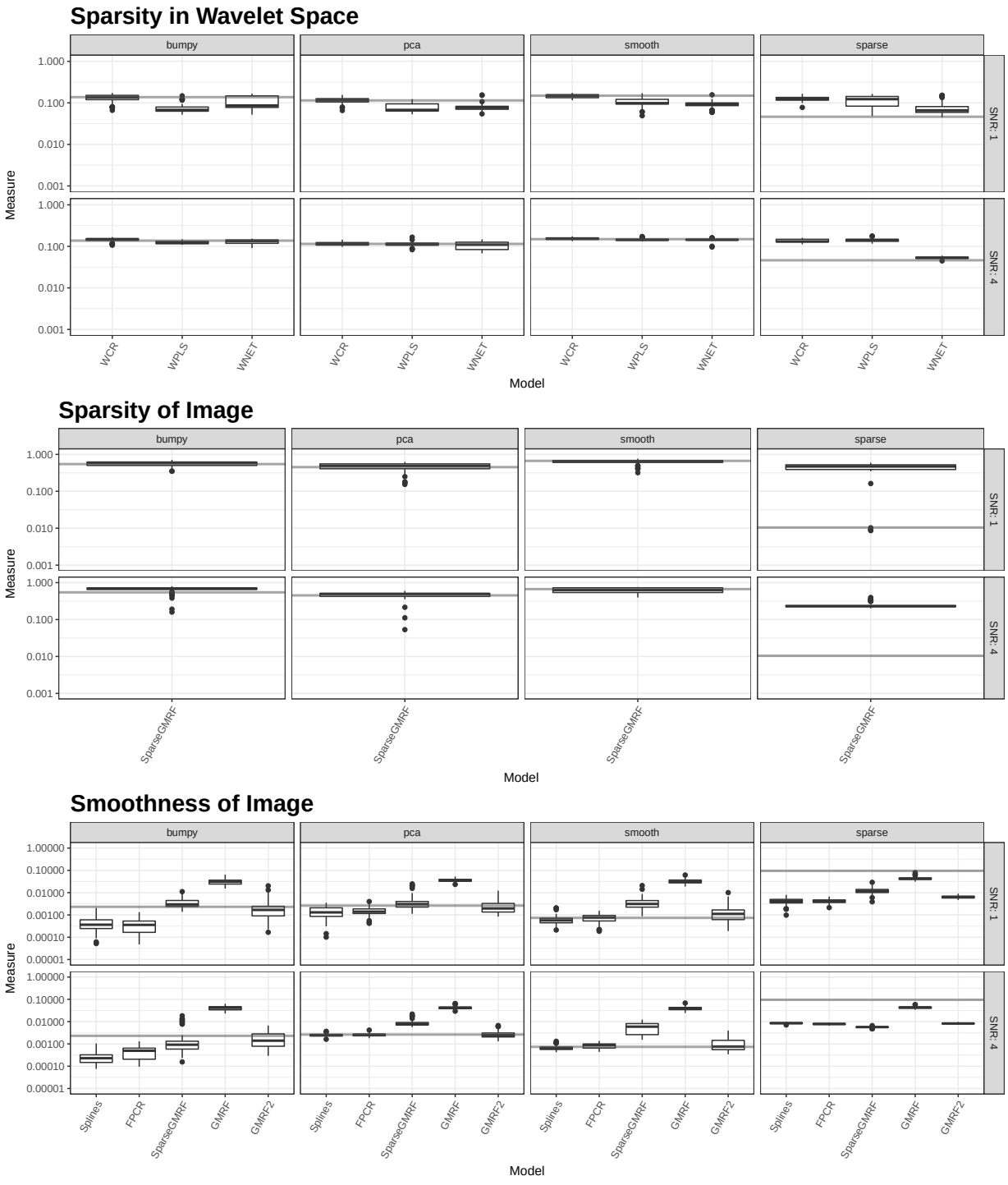


Figure 6.5.: Measures for underlying model assumptions in the simulation for $N = 250$ observations. Boxplots show the measures for the different models depending on the true coefficient image and the signal-to-noise ratio (SNR) over all 100 simulation runs. All values on log-scale. Gray horizontal lines correspond to the values for the true coefficient images.

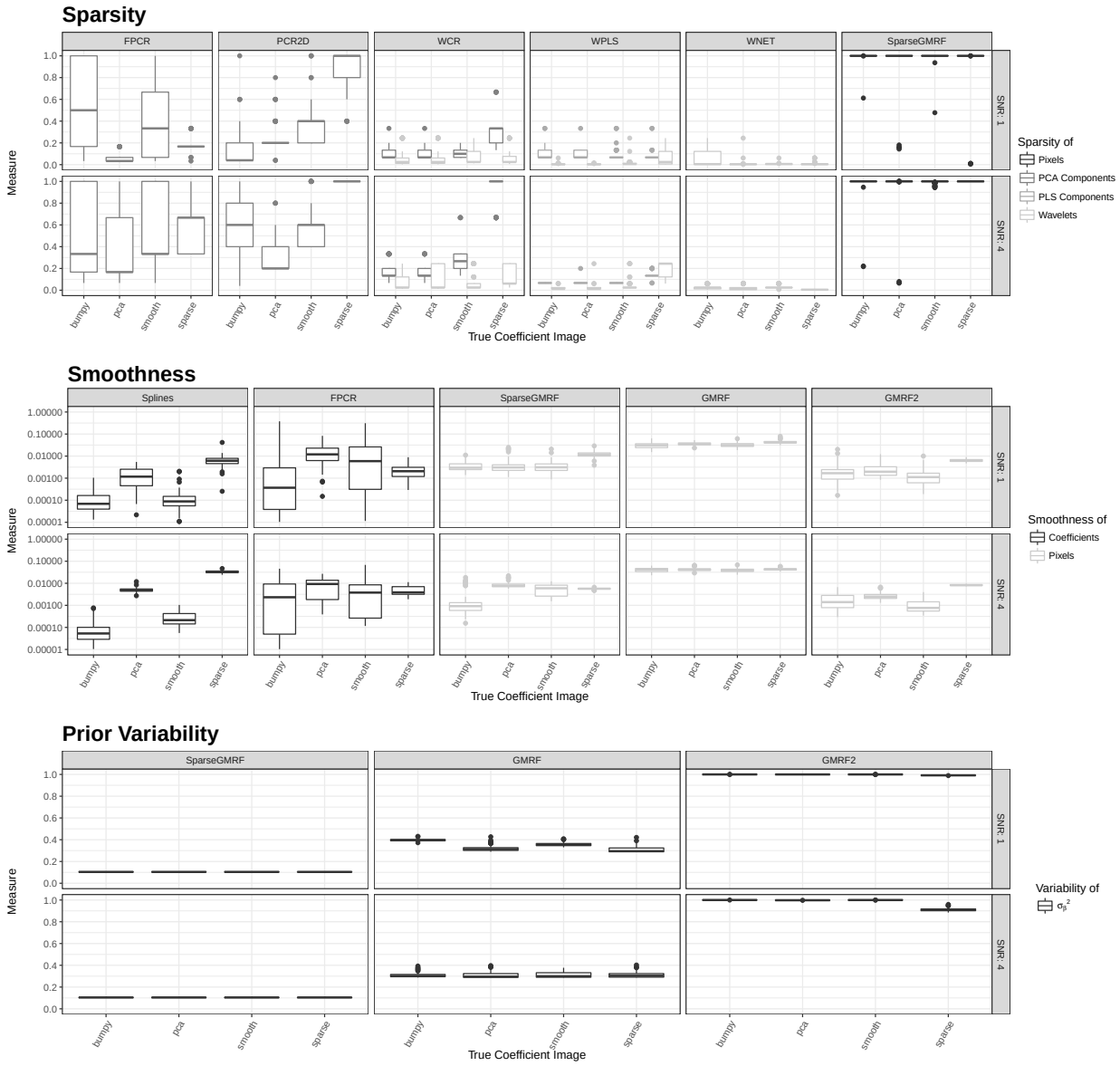


Figure 6.6.: Measures for parametric model assumptions in the simulation for $N = 250$ observations. Boxplots show the measures for the different coefficient images depending on the model used and the signal-to-noise ratio (SNR) over all 100 simulation runs. Measures with extremely low values on log-scale.

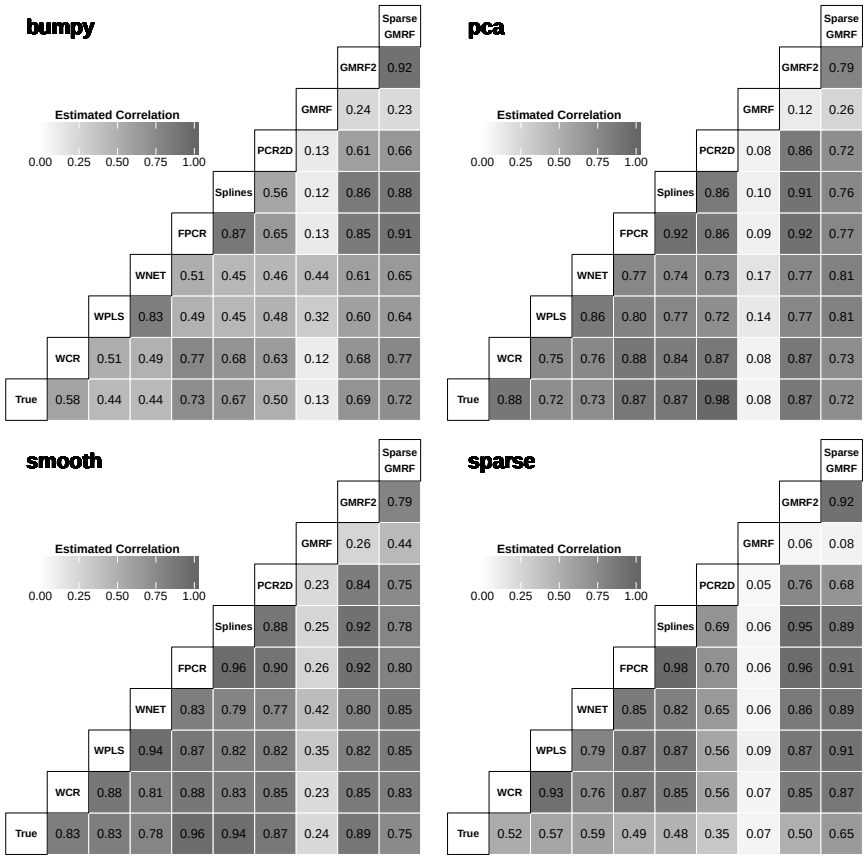


Figure 6.7.: Correlation between the true coefficient images and the estimates found by the different models in the simulation study with $N = 250$ observations and SNR $= 4$. The figures show the median correlation of the vectorized images over 100 simulation runs depending on the true images and the models used.

6.5. Application

In this section, the different scalar-on-image regression models are applied to a neuroimaging dataset with $N = 754$ subjects taken from the ADNI study (Mueller et al., 2005; Weiner et al., 2015), cf. Section 6.4.2. This example shows that different model assumptions can lead to quite different estimates for the coefficient image in practice, which of course has consequences on the interpretability of the results.

The aim of this analysis is to find a relation between a neuropsychological test (Alzheimer's disease assessment scale - cognitive subscale, ADAS-Cog: Rosen et al., 1984) and FDG-PET images, which measure the glucose uptake in the brain and thus reflect neural integrity. The ADAS-Cog values form the scalar response and the FDG-PET images are considered as image covariates together with age, gender and years of education as scalar covariates. All quantities are measured at baseline. ADAS-Cog takes values between 0 and 70, where higher values indicate worse global cognition and thus a higher risk of having Alzheimer's disease. The values are transformed via square root in order to obtain approximate residual normality, see Fig. D.6 in the appendix. For the image covariates, we use 64×64 subimages of the FDG-PET scans, which have also been used in the simulation study, see Fig. 6.2.

The resulting estimates for the coefficient image $\hat{\beta}$ are shown in Fig. 6.8 together with pointwise 95% credibility/confidence bands to illustrate the variability in the estimates (see appendix, Section D.3 for details on the calculation). The appendix contains an analogous plot for $\hat{\alpha}$ (Fig. D.7), measures for the underlying and parametric model assumptions as used in the simulation study (Table D.1) and an illustration of the goodness of fit (Fig. D.6).

Overall, *GMRF* gives quite poor results, which is in line with our findings in the simulation study. The $\text{IG}(1, 1)$ priors for the variance parameters obviously do not contribute enough information to give a reasonable fit. For all other models, the relative (in-sample) prediction error is very similar with values around 0.6 (cf. Fig. D.6 in the appendix). This also agrees with the results from the simulation, as different model assumptions may still lead to a similar predictive ability of the models.

The estimated coefficient images and the confidence bands have some common features: Most of the images have positive "bumps" in the upper left and right part as well as in the lower left corner, that are also flagged as "significant" by most models (i.e. the CI does not contain zero). Note that some pixels too many might be flagged, as the confidence bands do not account for e.g. variable selection via cross-validation and are mostly calculated on a pointwise basis. Areas with negative values are found especially in the center and the lower right part of the images. However, the influence of the assumptions made for the different models is clearly seen and the percentage of the "significant" pixels, their location and the shape of the "significant" regions differ considerably among the methods.

The results for *FPCR* and *Splines* are very smooth, which is reflected in the low values for the underlying sparsity assumption in Table D.1. The pixels flagged as "significant" by both models form mostly large, round shaped areas. The Bayesian models *GMRF2* and *SparseGMRF* do also assume smoothness, but on a pixel level rather than for basis function coefficients. The estimates thus are a bit more structured, which is also seen in the "significant" regions for *SparseGMRF*, that do also contain "non-significant" pixels and do not have clearly defined borders as for *Splines* or *FCPR*. *GRMF2* does not flag any pixel at all. When comparing the estimates from the two Bayesian models, the result of *SparseGMRF* seems to be a bit blurred compared to *GMRF2*, which might be due to the fact that the β coefficients might have been set to zero in some MCMC iterations, shrinking the posterior mean towards zero. The model assumption in *SparseGMRF* of many pixels being equal to zero, while all others form smooth clusters, however, is not achieved, which is also seen in the measure for the parametric sparsity assumption (it equals 1, i.e. the posterior mean of the latent Ising field γ is greater or equal to 0.5 in all pixels). The wavelet-based methods *WCR*, *WPLS* and *WNET* do not assume smoothness and hence have a more pronounced small-scale structure with more abrupt changes between positive and negative values. A very characteristic feature of the wavelet-based estimates are the negative values in the center of the images, which might have been oversmoothed by the other models. The main model assumption for *WCR*, *WPLS* and *WNET* is sparsity in the wavelet space. As seen in Table D.1, the *WNET* estimate has the highest sparsity, followed by *WPLS*. As seen in Fig. 6.8, higher sparsity translates to a more "spiky" estimate (note the different scale for *WPLS* and *WNET*). Finally, *PCR2D* assumes sparsity of β in the principal component space. As seen in Table D.1 in the appendix, the leading 20 of 25 possible eigenimages are selected (80%) to construct the estimate. What is striking here is the rectangular, "streaky" nature of the structures seen in the coefficient image. This is clearly caused by constructing the eigenimages as rank-one approximation (Allen, 2013).

The correlation matrix in Fig. D.8 in the appendix measures the similarity between all estimates. The highest correlation is found between models that assume smoothness (*SparseGMRF* and *FPCR*: 92%, *SparseGMRF* and *GMRF2*: 91%, *FPCR* and *Splines*: 90%). The *GMRF* estimate shows no correlation with any other method, which is also seen in Fig. 6.8.

The coefficient estimates $\hat{\alpha}$ for the intercept as well as the effects of age at baseline, gender and years of education are shown in Fig. D.7 in the appendix with the associated empirical credibility/confidence intervals. Here also, the result for the *GMRF* model clearly differs from all other models, particularly concerning the width of the confidence intervals. Except for *GMRF*, all covariates are seen to have a "significant" effect, as the confidence intervals do not include zero.

As for the coefficient images, there is again some common ground in the point estimates: The estimated intercepts and the effects for age at baseline are both positive, which makes

sense as ADAS-Cog takes positive values (note that the test scores have been transformed via square root before the analysis and now take values between 0 and 8.37) and age is known to be a main risk factor for Alzheimer's disease. For gender and years of education, the estimated coefficients are negative, which means that on average, being female and a longer period of education are associated with lower ADAS-Cog values and a lower risk of Alzheimer's disease. However, there is also a notable variation between the methods, as the confidence bands do not necessarily overlap or contain the point estimates of all other methods. Some of the differences in $\hat{\alpha}$ might be caused by the fact that different coefficient images $\hat{\beta}$ might lead to different parameter estimates $\hat{\alpha}$.

In total, the results of the application show that while some methods used here show some common patterns in their results, they can differ substantially in their details, as model assumptions can have a strong influence on the results. In practical applications, this can entail the risk of over-interpreting effects that are mainly driven by the model assumptions.

6.6. Discussion

Scalar-on-image regression is an inherently non-identifiable statistical problem due to the fact that the number of pixels – and therefore the number of coefficients – exceeds the number of observations, in many cases by far. In order to overcome the issue of non-identifiability, different approaches have been proposed in the literature, making different structural assumptions on the coefficient image. The most widely used assumptions for scalar-on-image regression include all forms and combinations of smoothness or sparsity, projection onto a subspace or assumptions on the variability of the coefficient image in terms of priors.

Whereas the beneficial aspects of making assumptions are well known and understood, their impact on the estimates seems underappreciated in practice. From a theoretical point of view it is obvious that models with different assumptions may lead to different estimates, as the scalar-on-image regression model inherently is not identifiable. In practical applications, however, it is not always clear which model is appropriate and to what extent the model assumptions influence the results. While this is less crucial for predictions, it strongly affects the interpretability of the coefficient image estimates, as one cannot identify whether features in the estimate are dominated by the model assumptions or driven by the data or are supported by both, as one would ideally assume. It is thus important to be aware of the assumptions made and the impact that they can have on the estimates.

In this paper, we have provided a systematic overview of the principal approaches to scalar-on-image regression and the assumptions made in the different models. The assumptions

have been characterized as underlying ones, that describe the fundamental assumptions of a model, and parametric assumptions, that are expressed in terms of penalties or priors for model parameters. The methods discussed in this paper do not completely represent all published scalar-on-image models, but largely cover all main classes and their assumptions. Variations include e.g. the LASSO-variant of *WNET* (Zhao et al., 2015, implemented in the `R` package `refund.wave`), all types of models that combine smoothness of the coefficient image with a sparsity assumption as in *SparseGMRF* (e.g. Huang, Goldsmith, et al. (2013), Shi and Kang (2015), Kang et al. (2016) or Li et al. (2015, provide a MATLAB implementation)) or methods for scalar-on-function regression that can easily be extended to the scalar-on-image case (e.g. the PLS variant of *FPCR*, both discussed in Reiss and Ogden (2007) in the case of functional data on one-dimensional domains). An overview of scalar-on-function regression models that can handle functional covariates on higher dimensional domains is provided in Reiss, Goldsmith, et al. (2016+).

Ideally, one would wish to have a diagnostic criterion that identifies problematic settings, i.e. settings in which the model assumptions dominate the estimate, in advance. This, however, seems very challenging, if even feasible. The measures proposed in this paper constitute a first step in this direction, as they can measure the degree to which the model assumptions are met. In Bayesian approaches, where model assumptions are formulated in terms of priors, alternative measures have been proposed, e.g. for prior-data conflict (Evans and Moshonov, 2006), prior informativeness (Müller, 2012) or prior data size with respect to the likelihood (Reimherr et al., 2014). Together with the measures proposed in this paper, they could serve as starting point for an overall measure for the appropriateness of model assumptions. However, most of these Bayesian measures are restricted to rather simple models and to proper priors. Further work is needed to be able to apply them to high-dimensional models such as scalar-on-image regression, improper priors such as the intrinsic *GMRF* priors or non-Bayesian models that include dimension reduction or variable selection steps.

For practical applications, we recommend to carefully check the assumptions in the models used. The measures proposed in this paper may help to interpret the results for real data, e.g. by relating values obtained for estimates from the original data to the values for hypothetic coefficient images, as in our simulation. The simulation results may also be indicative for the types of features that can be found with the chosen methods and the observed data. For the case of the FDG-PET images used in Sections 6.4 and 6.5, smooth coefficients and those lying in the span of the leading principal components were estimated quite well by methods with corresponding assumptions. At the same time, the coefficient images *bumpy* and *sparse*, which have highly localized features, are seen to be considerably more difficult to estimate.

Further, it seems helpful to compare the results with those of other approaches, making different assumptions, in order to find common patterns. These may help understanding

which features in the estimated coefficient image are mostly driven by the data, the model assumptions or combine both sources of information. Empirical confidence bands as in the application can serve as a first indicator, which regions of the estimated coefficient images might be of interest. For the ADNI data studied in Section 6.5, the empirical confidence bands agree most in the right upper and lower part of the images. Within these regions of interest, one could for example calculate the median correlation of the estimated coefficient images in order to check their agreement, which is an indicator of data-driven effects. The idea of combining different models is also adopted in ensemble methods, see e.g. the approach in Goldsmith and Scheipl (2014) for ensemble methods in scalar-on-function regression. A drawback of this approach, however, is that it is based on predictive performance in a cross-validation setting, which is not only associated with high computational costs, but also aims more at prediction and not at interpretability.

In summary, model assumptions are a necessary and helpful tool to overcome identifiability issues in complex models such as scalar-on-image regression. In practical applications, however, it is advisable to be aware of the assumptions made in a model and the impact that they can have on the coefficient estimates.

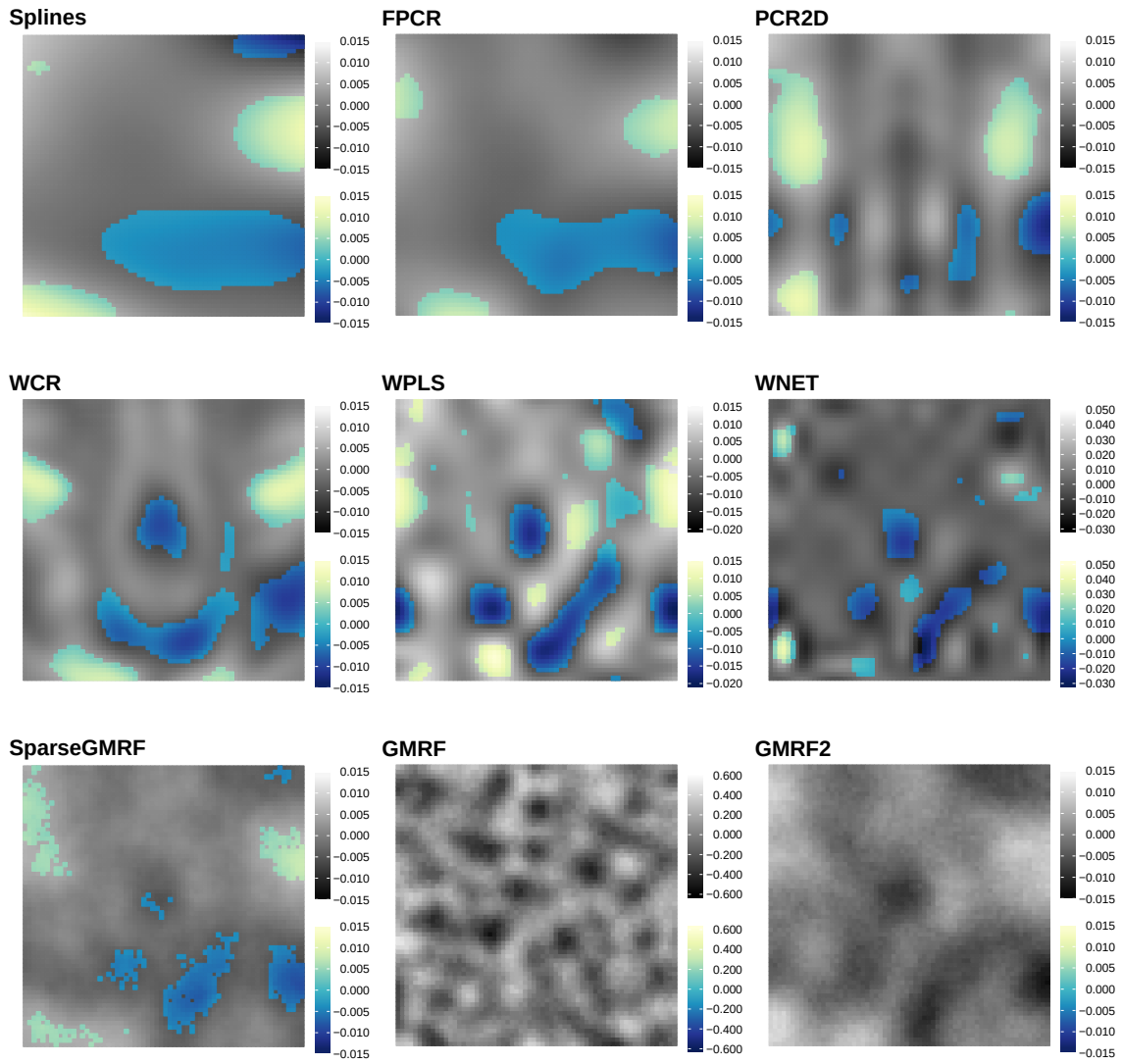


Figure 6.8.: Coefficient image estimates for the application with pointwise 95% confidence bands/credible intervals. Coloured pixels correspond to "significant" pixels, i.e. the confidence band/credible interval in this pixel does not contain zero. Note the different scales for *WPLS*, *WNET* and *GMRF*.

7. Concluding Remarks

Being able to combine the information of data available in different structures and dimensions becomes increasingly important in the *big data* era.

In the first part of this thesis, multivariate functional principal component analysis was introduced as a tool for the joint analysis of multivariate functional data on different dimensional domains. The new method extends existing concepts for MFPCA in two ways. First, the elements of the multivariate functional data need not necessarily be functions on one-dimensional domains. This opens the methodology to images (two-dimensional) or data on even higher dimensional domains, as the three-dimensional brain scans studied in Chapter 4. Second, one can now combine elements of different domains, such as functions and images, in one analysis. This provides new insights into the joint variation of data with different structures. At the same time, the MFPCA results allow a very parsimonious representation of the data in the form of individual scalar scores and principal components, which have the same structure as the data.

Based on a thorough theoretical foundation, the proposed estimators for the principal components, eigenvalues and scores were shown to be consistent under a given set of assumptions. The results of the simulation study further show that the new method is able to give good results in the case of a finite sample size, even in the presence of sparsity or measurement errors. In Chapter 4 it was demonstrated how MFPCA can be used to calculate out-of-sample predictions for new observations. The newly proposed method is easily applicable, as it is based on univariate basis expansions that can be chosen flexibly depending on the structure of the data. The current implementation of the algorithm in the **MFPCA** package allows users to choose between several univariate basis functions for data on up to three-dimensional domains. The implementation was done in an object-oriented manner, reflecting the strong inherent structure of the data.

The results of the neuroimaging applications in Chapters 3 and 4 indicate that the method yields principal components that are meaningful from a medical point of view and can lead to new insights into the data. Applications are of course not restricted to the medical area. For example, one could also think of meteorological data, combining webcam or satellite images with measurements of temperature or humidity over time.

From a methodological point of view, MFPCA can be used as a starting point for further statistical analysis of multivariate functional data, using the scores for instance in regression or clustering approaches. First findings indicate that the principal components obtained from MFPCA might lead to better results than combinations of univariate principal components in regression in the case of strong correlation among the functional covariates (Palma, 2017, co-supervised during this thesis). Future research might aim at integrating not only functions on different dimensional domains, but also vectors and more complicated objects such as e.g. warping functions, in a combined principal component analysis. At least for the warping functions, this would mean giving up the Hilbert space structure, which was a key instrument in the theoretical derivations of the MFPCA presented in this thesis.

Scalar-on-image regression is a statistical model class that aims at finding a relationship between scalar and image covariates and a scalar response. It thus combines data with different structure and complexity into one model. Due to the high number of covariates, the model is inherently non-identifiable and requires strong assumptions, whose impact was studied in the second part of this thesis. The benefits of making assumptions are well known and understood in statistical theory and practice. Their impact on the estimates, however, seems sometimes underappreciated. To this end, several models, representing the most important approaches for scalar-on-image regression, were analyzed and compared with respect to the assumptions made and their ability of finding relevant structures in a coefficient image. The results of the simulation study show that the model assumptions can indeed have a quite strong influence on the estimation results. Moreover, there is not one single model which dominates all others. Instead, the performance depends on how well the unknown coefficient image matches the model assumptions. Even in the rather unrealistic case of a signal-to-noise ratio of 4, there were still models that did not yield better estimates than a simpler model with a constant coefficient image. In addition, not all types of coefficient images are equally easy to estimate.

The findings of the study raise the question to what extent model assumptions influence the results in general, not only in scalar-on-image regression, and how they affect the interpretability of estimates obtained from complex models. At the same time, they open up new perspectives for methodological research: starting from the development of new methods with new assumptions, to the question of how to appropriately measure the discrepancy between a "true" coefficient image and its estimated version, to quantifying the impact of model assumptions on the estimates, to *borrowing strength* across several models by combining their results. The measures developed in Chapter 6 are a first attempt in this direction, but there is clearly room for improvements and extensions.

The results of this thesis illustrate that methodological statistical research is relevant, for the development of new methods as well as for the critical examination of existing approaches, also – and especially – in the era of *big data*.

Appendices

A. Appendix for Chapter 3

A.1. Proofs of Propositions

Proof of Prop. 1. $\mathcal{H}$ is a direct sum of the Hilbert spaces $L^2(\mathcal{T}_j),\ j = 1, \ldots, p$, with natural scalar product $\langle\langle \cdot, \cdot \rangle\rangle$ (cf. Reed and Simon, 1980, Chapter II.1.) □

Proof of Prop. 2.

1. Γ is linear: Follows from the linearity of the scalar product in (3.3).
2. Γ is self-adjoint: Follows from the symmetry $C_{ij}(s_i, t_j) = C_{ji}(t_j, s_i)$.
3. Γ is positive: Let $f \in \mathcal{H}$. Then

$$\begin{aligned}\langle\langle f, \Gamma f \rangle\rangle &= \sum_{j=1}^p \int_{\mathcal{T}_j} f^{(j)}(t_j) \sum_{i=1}^p \int_{\mathcal{T}_i} \mathbb{E}\left(X^{(i)}(s_i) X^{(j)}(t_j)\right) f^{(i)}(s_i) \mathrm{d}s_i \, \mathrm{d}t_j \\ &= \mathbb{E}\left(\sum_{j=1}^p \int_{\mathcal{T}_j} f^{(j)}(t_j) X^{(j)}(t_j) \mathrm{d}t_j\right)^2 \geq 0.\end{aligned}$$

4. Γ is compact: Let $\mathcal{B} := \{f \in \mathcal{H} : |||f|||^2 \leq B\}$ be a bounded family in $\mathcal{H}$ for some constant $0 < B < \infty$. Clearly, $\left\|f^{(j)}\right\|_2^2 \leq B$ for all $j = 1, \ldots, p$. Define the image of $\mathcal{B}$ under Γ by $\mathcal{Z} = \Gamma\mathcal{B} = \{g \in \mathcal{H} : \exists\ f \in \mathcal{B} \text{ such that } g = \Gamma f\}$, which has the following properties:

   - $\mathcal{Z}$ is uniformly bounded: Let $g \in \mathcal{Z}$ and $\boldsymbol{t} \in \mathcal{T}$. Define $K := \max_{i,j=1,\ldots,p} K_{ij}$ with K_{ij} as in (3.5). Then

$$\begin{aligned}\|g(\boldsymbol{t})\|^2 &\leq \sum_{j=1}^p \left(\sum_{i=1}^p \int_{\mathcal{T}_i} \left|C_{ij}(s_i, t_j) f^{(i)}(s_i)\right| \mathrm{d}s_i\right)^2 \\ &\overset{\text{Hölder}}{\leq} \sum_{j=1}^p \left(\sum_{i=1}^p \left[\int_{\mathcal{T}_i} C_{ij}(s_i, t_j)^2 \mathrm{d}s_i\right]^{1/2} \left[\int_{\mathcal{T}_i} f^{(i)}(s_i)^2 \mathrm{d}s_i\right]^{1/2}\right)^2 \\ &\leq \sum_{j=1}^p \left(\sum_{i=1}^p K^{1/2} B^{1/2}\right)^2 = p^3 K B < \infty.\end{aligned}$$

- $\mathcal{Z}$ is equicontinuous: Denote by $\lambda(\mathcal{T}_j)$ the Lebesgue measure of $\mathcal{T}_j$ and let $T = \max_{j=1,\dots,p} \lambda(\mathcal{T}_j)$. For $\varepsilon > 0$, define $\tilde{\varepsilon} := \frac{\varepsilon}{p(pTB)^{1/2}}$. By the continuity assumption for $C_{ij}(s_i, \cdot)$, there exist $\delta_{ij} > 0$ such that

$$\left\| t_j - t_j^* \right\| < \delta_{ij} \quad \Rightarrow \quad |C_{ij}(s_i, t_j) - C_{ij}(s_i, t_j^*)| < \tilde{\varepsilon}, \quad \forall\ s_i \in \mathcal{T}_i. \tag{A.1}$$

for all $i, j = 1, \dots, p$. Set $\delta := \min_{i,j=1,\dots,p} \delta_{ij}$ and let $\|\boldsymbol{t} - \boldsymbol{t}^*\|_{\mathcal{T}} < \delta$. Clearly, $\left\| t_j - t_j^* \right\| < \delta$ for all $j = 1, \dots, p$ and for $g \in \mathcal{Z}$ it holds

$$\begin{aligned}
\|g(\boldsymbol{t}) - g(\boldsymbol{t}^*)\|^2 &\leq \sum_{j=1}^p \left(\sum_{i=1}^p \left| \int_{\mathcal{T}_i} \left(C_{ij}(s_i, t_j) - C_{ij}(s_i, t_j^*) \right) f^{(i)}(s_i) \mathrm{d}s_i \right| \right)^2 \\
&\overset{\text{Hölder}}{\leq} \sum_{j=1}^p \left(\sum_{i=1}^p \left[\int_{\mathcal{T}_i} \left(C_{ij}(s_i, t_j) - C_{ij}(s_i, t_j^*) \right)^2 \mathrm{d}s_i \right]^{1/2} \left[\int_{\mathcal{T}_i} f^{(i)}(s_i)^2 \mathrm{d}s_i \right]^{1/2} \right)^2 \\
&\overset{(A.1)}{<} \sum_{j=1}^p \left(\sum_{i=1}^p \left(\int_{\mathcal{T}_i} \tilde{\varepsilon}^2 \mathrm{d}s_i \right)^{1/2} \left\| f^{(i)} \right\|_2 \right)^2 \leq p^3 TB \tilde{\varepsilon}^2 = \varepsilon^2.
\end{aligned}$$

By the Theorem of Arzelà-Ascoli (Reed and Simon, 1980, Thm. I.28. and related notes for Chapter I), for each sequence $\{f_n\}_{n\in\mathbb{N}}$ in $\mathcal{B}$ there exists a convergent subsequence $\{g_{n(i)} = \Gamma f_{n(i)}\}_{i\in\mathbb{N}}$ of the corresponding sequence $\{g_n\}_{n\in\mathbb{N}}$ in $\mathcal{Z}$, which implies that Γ is a compact operator (cf. Reed and Simon, 1980, Chapter VI.5.).

□

Lemma 1. *For fixed $m \in \mathbb{N}$ and $j \in \{1, \dots, p\}$, the j-th element $\psi_m^{(j)}$ of the eigenfunction ψ_m is continuous, if $\nu_m > 0$ and C_{ij} is uniformly continuous as in Prop. 2.*

Proof. Let $\varepsilon > 0$ and $\tilde{\varepsilon} := \varepsilon \nu_m \left(2 \sum_{j=1}^p \lambda(\mathcal{T}_j)^{1/2} \right)^{-1}$. By the uniform continuity assumption for C_{ij}, there exist $\delta_{ij} > 0$ such that for all $i = 1, \dots, p$ (A.1) holds. Let $\delta_j = \min_{i=1,\dots,p} \delta_{ij}$ and $\left\| t_j - t_j^* \right\| < \delta_j$. Then, as $\nu_m > 0$ and $\left\| \psi_m^{(i)} \right\|_2 \leq |||\psi_m||| = 1$,

$$\begin{aligned}
\left| \psi_m^{(j)}(t_j) - \psi_m^{(j)}(t_j^*) \right| &= \frac{1}{\nu_m} \left| \sum_{i=1}^p \int_{\mathcal{T}_i} \left[C_{ij}(s_i, t_j) - C_{ij}(s_i, t_j^*) \right] \psi_m^{(i)}(s_i) \mathrm{d}s_i \right| \\
&\leq \frac{1}{\nu_m} \tilde{\varepsilon} \sum_{i=1}^p \int_{\mathcal{T}_i} \left| \psi_m^{(i)}(s_i) \right| \mathrm{d}s_i \overset{\text{Hölder}}{\leq} \frac{\tilde{\varepsilon}}{\nu_m} \sum_{i=1}^p \left\| \psi_m^{(i)} \right\|_2 \lambda(\mathcal{T}_i)^{1/2} \\
&\leq \frac{\tilde{\varepsilon}}{\nu_m} \sum_{i=1}^p \lambda(\mathcal{T}_i)^{1/2} = \frac{\varepsilon}{2} < \varepsilon.
\end{aligned}$$

□

Proof of Prop. 3. The proof follows the idea in Werner (2011, Chapter VI.4.) for the proof of Mercer's Theorem in the univariate case. From the Spectral Theorem for compact self-adjoint operators (Werner, 2011, Thm. VI.3.2.), it is known that

$$\Gamma f = \sum_{m=1}^\infty \nu_m \langle\langle f, \psi_m \rangle\rangle \psi_m \quad \forall\ f \in \mathcal{H}.$$

For $M \in \mathbb{N}$ define

$$\Gamma_M f := \sum_{m=1}^{M} \nu_m \langle\langle f, \psi_m \rangle\rangle \psi_m \quad \forall\ f \in \mathcal{H}.$$

Then for all $f \in \mathcal{H}$: $\langle\langle \Gamma f, f \rangle\rangle - \langle\langle \Gamma_M f, f \rangle\rangle = \sum_{m=M+1}^{\infty} \nu_m \langle\langle f, \psi_m \rangle\rangle^2 \geq 0$. Let $j \in \{1, \ldots, p\}$ and $t^* \in \mathcal{T}_j$. Define $f = \left(0, \ldots, 0, f^{(j)}, 0, \ldots, 0\right)$ with $f^{(j)} = \lambda\left(B_{t^*}\left(\frac{1}{n}\right)\right)^{-1} \mathbf{1}_{B_{t^*}(\frac{1}{n})}$ for some $n \in \mathbb{N}$, where $B_{t^*}(\frac{1}{n})$ is a closed ball in $\mathcal{T}_j$ with center t^* and radius $\frac{1}{n}$, $\lambda(\cdot)$ denotes the Lebesgue measure and $\mathbf{1}$ is the indicator function. Clearly, $f^{(j)} \in L^2(\mathcal{T}_j)$ and $f \in \mathcal{H}$. Therefore

$$\begin{aligned}
0 &\leq \langle\langle \Gamma f, f \rangle\rangle - \langle\langle \Gamma_M f, f \rangle\rangle \\
&= \lambda\left(B_{t^*}\left(\tfrac{1}{n}\right)\right)^{-2} \int_{B_{t^*}(\frac{1}{n})} \int_{B_{t^*}(\frac{1}{n})} C_{jj}(s_j, t_j) - \sum_{m=1}^{M} \nu_m \psi_m^{(j)}(s_j) \psi_m^{(j)}(t_j) \mathrm{d}s_j\ \mathrm{d}t_j \\
&\to C_{jj}(t^*, t^*) - \sum_{m=1}^{M} \nu_m \psi_m^{(j)}(t^*) \psi_m^{(j)}(t^*) \quad \text{for } n \to \infty
\end{aligned}$$

by the Lebesgue Differentiation Theorem (Rudin, 1987, Thm. 7.10.). As t^* was arbitrary in $\mathcal{T}_j$, this implies that for all $M \in \mathbb{N}$

$$\sum_{m=1}^{M} \nu_m \psi_m^{(j)}(t)^2 \leq C_{jj}(t,t) \leq \|C_{jj}\|_\infty < \infty \quad \forall\ t \in \mathcal{T}_j,$$

since C_{jj} is continuous and $\mathcal{T}_j$ is compact, implying that $\|C_{jj}\|_\infty := \sup_{t \in \mathcal{T}_j} |C_{jj}(t,t)|$ is finite. Using Hölder's inequality

$$\sum_{m=1}^{\infty} \left|\nu_m \psi_m^{(j)}(s) \psi_m^{(j)}(t)\right| \leq C_{jj}(s,s)^{1/2} C_{jj}(t,t)^{1/2} < \infty,$$

i.e. the series $\tilde{C}_j(s,t) := \sum_{m=1}^{\infty} \nu_m \psi_m^{(j)}(s) \psi_m^{(j)}(t)$ is absolutely convergent for all $s, t \in \mathcal{T}_j$. In the following, assume $t \in \mathcal{T}_j$ to be fixed. For $\varepsilon > 0$ choose $M \in \mathbb{N}$ such that $\sum_{m=M+1}^{\infty} \nu_m \psi_m^{(j)}(t)^2 < \varepsilon^2$. Then, again by Hölder's inequality

$$\sum_{m=M+1}^{\infty} \left|\nu_m \psi_m^{(j)}(s) \psi_m^{(j)}(t)\right| \leq C_{jj}(s,s)^{1/2} \cdot \varepsilon \leq \|C_{jj}\|_\infty^{1/2} \cdot \varepsilon. \tag{A.2}$$

The upper bound in (A.2) does not depend on s, hence $\tilde{C}_j(s,t)$ converges uniformly for fixed t. As the eigenfunctions $\psi_m^{(j)}(s)$ are continuous in s for all $m \in \mathbb{N}$ (Lemma 1), $\tilde{C}_j(s,t)$ is also continuous in s (Uniform Limit Theorem, Munkres, 2000, Thm. 21.6.). Define

$$h_j(s) := C_{jj}(s,t) - \tilde{C}_j(s,t), \quad s \in \mathcal{T}_j.$$

Let now $g^{(j)} \in L^2(\mathcal{T}_j)$ and define $g := \left(0, \ldots, 0, g^{(j)}, 0, \ldots, 0\right)$, which is clearly in $\mathcal{H}$. Therefore

$$\begin{aligned}
\int_{\mathcal{T}_j} h_j(s) g^{(j)}(s) \mathrm{d}s &= \sum_{i=1}^{p} \int_{\mathcal{T}_i} C_{ij}(s_i, t) g^{(i)}(s_i) \mathrm{d}s_i - \sum_{m=1}^{\infty} \nu_m \sum_{i=1}^{p} \int_{\mathcal{T}_i} \psi_m^{(i)}(s_i) g^{(i)}(s_i) \mathrm{d}s_i\ \psi_m^{(j)}(t) \\
&= (\Gamma g)^{(j)}(t) - \sum_{m=1}^{\infty} \nu_m \langle\langle \psi_m, g \rangle\rangle \psi_m^{(j)}(t) \\
&= \sum_{m=1}^{\infty} \nu_m \langle\langle g, \psi_m \rangle\rangle \psi_m^{(j)}(t) - \sum_{m=1}^{\infty} \nu_m \langle\langle \psi_m, g \rangle\rangle \psi_m^{(j)}(t) = 0
\end{aligned}$$

according to the Spectral Theorem. Choosing $g^{(j)} = h_j$ implies $h_j(s) = 0$ for all $s \in \mathcal{T}_j$, as h_j is continuous in s. Therefore,

$$\tilde{C}_j(s,t) = \sum_{m=1}^{\infty} \nu_m \psi_m^{(j)}(s)\psi_m^{(j)}(t) = C_{jj}(s,t) \quad \forall\ s \in \mathcal{T}_j.$$

By Dini's Theorem (Werner, 2011, Thm. VI.4.6.) the series $C_{jj}(t,t) = \sum_{m=1}^{\infty} \nu_m \psi_m^{(j)}(t)^2$ converges uniformly. Hence, M can be chosen independent of t in (A.2). This implies that $\tilde{C}_j(s,t)$ converges absolutely and uniformly to $C_{jj}(s,t)$ for all $s,t \in \mathcal{T}_j$. □

Proof of Prop. 4. By the Hilbert-Schmidt Theorem (Reed and Simon, 1980, Thm. VI.16.), the (deterministic) eigenfunctions of Γ form an orthonormal basis of $\mathcal{H}$, i.e. X can be written in the form $X(\boldsymbol{t}) = \sum_{m=1}^{\infty} \rho_m \psi_m(\boldsymbol{t}),\ \boldsymbol{t} \in \mathcal{T}$ with random variables $\rho_m = \langle\langle X, \psi_m \rangle\rangle$. Hence for $m,n \in \mathbb{N}$

1. $\mathbb{E}(\rho_m) = \sum_{j=1}^{p} \int_{\mathcal{T}_j} \mathbb{E}\left(X^{(j)}(t_j)\right) \psi_m^{(j)}(t_j)\mathrm{d}t_j = 0$, since $\mathbb{E}\left(X^{(j)}(t_j)\right) = 0$ for all $t_j \in \mathcal{T}_j$, $j = 1,\dots,p$ by assumption.

2. $$\begin{aligned}\mathrm{Cov}(\rho_m,\rho_n) &= \mathbb{E}\left(\sum_{i=1}^{p}\int_{\mathcal{T}_i} X^{(i)}(s_i)\psi_m^{(i)}(s_i)\mathrm{d}s_i \cdot \sum_{j=1}^{p}\int_{\mathcal{T}_j} X^{(j)}(t_j)\psi_n^{(j)}(t_j)\mathrm{d}t_j\right)\\ &= \sum_{j=1}^{p}\int_{\mathcal{T}_j}\sum_{i=1}^{p}\int_{\mathcal{T}_i} C_{ij}(s_i,t_j)\psi_m^{(i)}(s_i)\mathrm{d}s_i\ \psi_n^{(j)}(t_j)\mathrm{d}t_j\\ &= \sum_{j=1}^{p}\int_{\mathcal{T}_j} \nu_m\psi_m^{(j)}(t_j)\psi_n^{(j)}(t_j)\mathrm{d}t_j = \nu_m\delta_{mn}.\end{aligned}$$

3. Let $X_{\lceil M\rceil}(\boldsymbol{t}) := \sum_{m=1}^{M}\rho_m\psi_m(\boldsymbol{t}) = \sum_{m=1}^{M}\left[\sum_{i=1}^{p}\int_{\mathcal{T}_i} X^{(i)}(s_i)\psi_m^{(i)}(s_i)\mathrm{d}s_i\right]\psi_m(\boldsymbol{t})$ for $\boldsymbol{t} \in \mathcal{T}$ be the truncated Karhunen-Loève representation of X. Then

$$\begin{aligned}\mathbb{E}\left(\left\|X(\boldsymbol{t}) - X_{\lceil M\rceil}(\boldsymbol{t})\right\|^2\right) &= \sum_{j=1}^{p}\left[\mathbb{E}\left(X^{(j)}(t_j)^2\right) - 2\mathbb{E}\left(X^{(j)}(t_j)X_{\lceil M\rceil}^{(j)}(t_j)\right) + \mathbb{E}\left(X_{\lceil M\rceil}^{(j)}(t_j)^2\right)\right]\\ &= \sum_{j=1}^{p}\Bigg[C_{jj}(t_j,t_j) - 2\sum_{m=1}^{M}\sum_{i=1}^{p}\int_{\mathcal{T}_i} C_{ij}(s_i,t_j)\psi_m^{(i)}(s_i)\mathrm{d}s_i\ \psi_m^{(j)}(t_j)\\ &\quad + \sum_{m=1}^{M}\sum_{n=1}^{M}\sum_{i=1}^{p}\int_{\mathcal{T}_i}\sum_{k=1}^{p}\int_{\mathcal{T}_k} C_{ki}(u_k,s_i)\psi_n^{(k)}(u_k)\mathrm{d}u_k\ \psi_m^{(i)}(s_i)\mathrm{d}s_i\ \psi_m^{(j)}(t_j)\psi_n^{(j)}(t_j)\Bigg]\\ &= \sum_{j=1}^{p}\Bigg[C_{jj}(t_j,t_j) - 2\sum_{m=1}^{M}\nu_m\psi_m^{(j)}(t_j)\psi_m^{(j)}(t_j)\\ &\quad + \sum_{m=1}^{M}\sum_{n=1}^{M}\sum_{i=1}^{p}\int_{\mathcal{T}_i}\nu_n\psi_n^{(i)}(s_i)\psi_m^{(i)}(s_i)\mathrm{d}s_i\psi_m^{(j)}(t_j)\psi_n^{(j)}(t_j)\Bigg]\\ &= \sum_{j=1}^{p}\left[C_{jj}(t_j,t_j) - \sum_{m=1}^{M}\nu_m\psi_m^{(j)}(t_j)^2\right] \to 0 \quad \text{for } M\to\infty\end{aligned}$$

uniformly for $\boldsymbol{t} \in \mathcal{T}$ by Prop. 3.

□

Proof of Prop. 5.

1. Let X have a finite Karhunen-Loève representation (3.7). Then, each element is given by $X^{(j)} = \sum_{m=1}^{M} \rho_m \psi_m^{(j)}$. For $t \in \mathcal{T}_j$

$$\begin{aligned}\left(\Gamma^{(j)}\tilde{\phi}^{(j)}\right)(t) &= \int_{\mathcal{T}_j} \mathrm{Cov}(X^{(j)}(s), X^{(j)}(t))\tilde{\phi}^{(j)}(s)\mathrm{d}s \\ &= \int_{\mathcal{T}_j} \sum_{m=1}^{M} \nu_m \psi_m^{(j)}(s)\psi_m^{(j)}(t)\tilde{\phi}^{(j)}(s)\mathrm{d}s \overset{!}{=} \lambda^{(j)}\tilde{\phi}^{(j)}(t), \qquad \text{(A.3)}\end{aligned}$$

which is a homogenous Fredholm integral equation of the second kind with separable kernel function $K(s,t) = \sum_{m=1}^{M} \nu_m \psi_m^{(j)}(s)\psi_m^{(j)}(t) = \sum_{m=1}^{M} a_m(s)b_m(t)$ with continuous functions $a_m(s) = \nu_m^{1/2}\psi_m^{(j)}(s)$, $b_m(t) = \nu_m^{1/2}\psi_m^{(j)}(t)$ (cf. Lemma 1). Following the argumentation in Zemyan (2012, Chapter 1.3.), (A.3) can be transformed into the matrix eigenequation

$$\boldsymbol{A^{(j)}}\boldsymbol{u} = \lambda^{(j)}\boldsymbol{u}$$

with a symmetric matrix $\boldsymbol{A^{(j)}} \in \mathbb{R}^{M\times M}$ given by $A_{mn}^{(j)} = \langle a_m, b_n\rangle_2$. Positivity of $\Gamma^{(j)}$ implies $\lambda^{(j)} \geq 0$ and therefore $\boldsymbol{A^{(j)}}$ is positive semidefinite. Hence it can have at most M strictly positive eigenvalues $\lambda_m^{(j)}$ associated with eigenvectors $\boldsymbol{u_m^{(j)}} \in \mathbb{R}^M$

$$\boldsymbol{A^{(j)}}\boldsymbol{u_m^{(j)}} = \lambda_m^{(j)}\boldsymbol{u_m^{(j)}}.$$

Let $\lambda_1^{(j)} \geq \ldots \geq \lambda_{M_j}^{(j)} > 0$, $M_j \leq M$ be the non-zero eigenvalues of $\boldsymbol{A^{(j)}}$. They are at the same time the only non-zero eigenvalues of $\Gamma^{(j)}$. The associated eigenfunctions $\tilde{\phi}_m^{(j)}$ of $\Gamma^{(j)}$ are parametrized by the eigenvectors $\boldsymbol{u_m^{(j)}}$ via (Zemyan, 2012, Chapter 1.3.)

$$\Gamma^{(j)}\tilde{\phi}_m^{(j)} = \sum_{n=1}^{M} \nu_n^{1/2}\psi_n^{(j)}[\boldsymbol{u_m^{(j)}}]_n = \lambda_m^{(j)}\tilde{\phi}_m^{(j)} \quad \Leftrightarrow \quad \tilde{\phi}_m^{(j)} = \left(\lambda_m^{(j)}\right)^{-1}\sum_{n=1}^{M} \nu_n^{1/2}\psi_n^{(j)}[\boldsymbol{u_m^{(j)}}]_n.$$

Since $\langle\tilde{\phi}_m^{(j)}, \tilde{\phi}_n^{(j)}\rangle_2 = \left(\lambda_m^{(j)}\lambda_n^{(j)}\right)^{-1}\boldsymbol{u_m^{(j)}}^{\top}\boldsymbol{A^{(j)}}\boldsymbol{u_n^{(j)}} = \left(\lambda_m^{(j)}\right)^{-1}\delta_{mn}$, orthonormal eigenfunctions $\phi_m^{(j)}$ are given by

$$\phi_m^{(j)} = \lambda_m^{(j)^{1/2}}\tilde{\phi}_m^{(j)} = \left(\lambda_m^{(j)}\right)^{-1/2}\sum_{n=1}^{M} \nu_n^{1/2}\psi_n^{(j)}[\boldsymbol{u_m^{(j)}}]_n.$$

Therefore, $X^{(j)}$ has a finite Karhunen-Loève representation $X^{(j)} = \sum_{m=1}^{M_j} \xi_m^{(j)}\phi_m^{(j)}$ with scores

$$\xi_m^{(j)} = \langle X^{(j)}, \phi_m^{(j)}\rangle_2 = \left(\lambda_m^{(j)}\right)^{-1/2}\sum_{n=1}^{M} \nu_n^{1/2}\left[\boldsymbol{u_m^{(j)}}\right]_n \sum_{k=1}^{M} \rho_k\langle\psi_n^{(j)}, \psi_k^{(j)}\rangle_2$$

and $\mathbb{E}(\xi_m^{(j)}) = 0$, $\mathrm{Cov}(\xi_m^{(j)}, \xi_n^{(j)}) = \lambda_m^{(j)}\delta_{mn}$.

2. Assume the functional covariates $X^{(1)}, \ldots, X^{(p)}$ do each have a finite Karhunen-Loève representation, i.e. for each $j = 1, \ldots, p$: $X^{(j)} = \sum_{m=1}^{M_j} \xi_m^{(j)}\phi_m^{(j)}$. Let $\Gamma\psi = \nu\psi$. Then for all $j = 1, \ldots, p$ and $t_j \in \mathcal{T}_j$

$$\begin{aligned}(\Gamma\psi)^{(j)}(t_j) &= \sum_{k=1}^{p}\int_{\mathcal{T}_k} \mathrm{Cov}\left(X^{(k)}(s_k), X^{(j)}(t_j)\right)\psi^{(k)}(s_k)\mathrm{d}s_k \\ &= \sum_{k=1}^{p}\sum_{l=1}^{M_j}\sum_{n=1}^{M_k} \underbrace{\mathrm{Cov}\left(\xi_l^{(j)}, \xi_n^{(k)}\right)}_{=:Z_{ln}^{(jk)}}\phi_l^{(j)}(t_j)\underbrace{\int_{\mathcal{T}_k}\phi_n^{(k)}(s_k)\psi^{(k)}(s_k)\mathrm{d}s_k}_{=:c_n^{(k)}} \overset{!}{=} \nu\psi^{(j)}(t_j).\end{aligned}$$

With a similar argumentation as in Zemyan (2012, Chapter 1.3.) it holds

$$\begin{aligned} &\sum_{k=1}^{p}\sum_{l=1}^{M_j}\sum_{n=1}^{M_k} Z_{ln}^{(jk)}\phi_l^{(j)}(t_j)c_n^{(k)} = \nu\psi^{(j)}(t_j) \qquad (A.4)\\ \Rightarrow &\int_{\mathcal{T}_j}\phi_m^{(j)}(t_j)\cdot\sum_{k=1}^{p}\sum_{l=1}^{M_j}\sum_{n=1}^{M_k} Z_{ln}^{(jk)}\phi_l^{(j)}(t_j)c_n^{(k)}\mathrm{d}t_j = \int_{\mathcal{T}_j}\phi_m^{(j)}(t_j)\cdot\nu\psi^{(j)}(t_j)\mathrm{d}t_j\\ \Leftrightarrow &\sum_{k=1}^{p}\sum_{n=1}^{M_k} Z_{mn}^{(jk)}c_n^{(k)} = \nu c_m^{(j)} \end{aligned}$$

for $m = 1,\dots,M_j$ due to orthonormality of $\phi_m^{(j)}$. Since m and j were arbitrarily chosen, this is equivalent to

$$\underbrace{\begin{pmatrix} \boldsymbol{Z^{(11)}} & \dots & \boldsymbol{Z^{(1p)}}\\ \vdots & \ddots & \vdots\\ \boldsymbol{Z^{(p1)}} & \dots & \boldsymbol{Z^{(pp)}}\end{pmatrix}}_{=:\boldsymbol{Z}}\underbrace{\begin{pmatrix}\boldsymbol{c^{(1)}}\\ \vdots\\ \boldsymbol{c^{(p)}}\end{pmatrix}}_{=:\boldsymbol{c}} = \nu\begin{pmatrix}\boldsymbol{c^{(1)}}\\ \vdots\\ \boldsymbol{c^{(p)}}\end{pmatrix}$$

with matrices $\boldsymbol{Z^{(jk)}} \in \mathbb{R}^{M_j\times M_k}$ and $\boldsymbol{c^{(j)}} \in \mathbb{R}^{M_j}$. The last equation is again an eigenequation for the symmetric (and since $\nu \geq 0$) positive semidefinite block matrix $\boldsymbol{Z} \in \mathbb{R}^{M_+\times M_+}$. Let $\nu_1 \geq \dots \geq \nu_M > 0,\ M \leq M_+$ be the non-zero eigenvalues of $\boldsymbol{Z}$. These are also the only non-zero eigenvalues of Γ and the elements $\psi_m^{(j)}$ of the associated eigenfunctions ψ_m are parametrized by the (orthonormal) eigenvectors $\boldsymbol{c_1},\dots,\boldsymbol{c_M}$ associated with $\nu_1,\dots,\nu_M$:

$$\psi_m^{(j)}(t_j) \overset{(A.4)}{=} \frac{1}{\nu_m}\sum_{k=1}^{p}\sum_{l=1}^{M_j}\sum_{n=1}^{M_k} Z_{ln}^{(jk)}[\boldsymbol{c_m}]_n^{(k)}\phi_l^{(j)}(t_j) = \sum_{l=1}^{M_j}[\boldsymbol{c_m}]_l^{(j)}\phi_l^{(j)}(t_j), \quad t_j \in \mathcal{T}_j$$

for $m = 1,\dots,M,\ j = 1,\dots,p$. The eigenfunctions form an orthonormal system with respect to $\langle\langle\cdot,\cdot\rangle\rangle$:

$$\langle\langle\psi_m,\psi_n\rangle\rangle = \sum_{j=1}^{p}\langle\psi_m^{(j)},\psi_n^{(j)}\rangle_2 = \sum_{j=1}^{p}\sum_{l=1}^{M_j}[\boldsymbol{c_m}]_l^{(j)}[\boldsymbol{c_n}]_l^{(j)} = \boldsymbol{c_n}^\top\boldsymbol{c_m} = \delta_{mn}.$$

The Karhunen-Loève decomposition of X is therefore given by $X = \sum_{m=1}^{M}\rho_m\psi_m$ with scores

$$\rho_m = \langle\langle X,\psi_m\rangle\rangle = \sum_{j=1}^{p}\sum_{n=1}^{M_j}[\boldsymbol{c_m}]_n^{(j)}\,\xi_n^{(j)}$$

and $\mathbb{E}(\rho_m) = 0,\ \mathrm{Cov}(\rho_m,\rho_n) = \nu_m\delta_{mn},\ m = 1,\dots,M \leq M_+$.

□

Proof of Prop. 6. For $f \in \mathcal{H},\ \boldsymbol{t} \in \mathcal{T}$ and $j = 1,\dots,p$, the covariance operator $\Gamma^{[M]}$ associated with $X^{[M]}$ is given by

$$(\Gamma^{[M]}f)^{(j)}(t_j) = \sum_{i=1}^{p}\int_{\mathcal{T}_i}\mathrm{Cov}(X^{[M](i)}(s_i),X^{[M](j)}(t_j))f^{(i)}(s_i)\mathrm{d}s_i.$$

In the following, use $C_{ij}^{[M]}(s_i, t_j) := \mathrm{Cov}(X^{[M](i)}(s_i), X^{[M](j)}(t_j))$ as short notation for the covariance functions (cf. the definition of C_{ij} in (3.1) in the paper). Next, recall some well-known results for univariate functional data: By Mercer's Theorem (Mercer, 1909)

$$C_{jj}^{[M]}(t_j, t_j) = \sum_{m=1}^{M_j} \lambda_m^{(j)} \phi_m^{(j)}(t_j)^2 \nearrow \sum_{m=1}^{\infty} \lambda_m^{(j)} \phi_m^{(j)}(t_j)^2 = C_{jj}(t_j, t_j) \text{ for } M_j \to \infty,\ t_j \in \mathcal{T}_j. \quad \text{(A.5)}$$

The univariate Karhunen-Loève Theorem (e.g. Bosq, 2000, Thm 1.5.) states that

$$\mathbb{E}\left[\left|X^{(j)}(t_j) - \textstyle\sum_{m=1}^{M_j} \xi_m^{(j)} \phi_m^{(j)}(t_j)\right|^2\right]$$

converges uniformly to 0 for $t_j \in \mathcal{T}_j$ and $M_j \to \infty$. As both $X^{(j)}$ and $X^{[M](j)}$ have zero mean ($X^{(j)}$ by assumption and $X^{[M](j)}$ since the scores $\xi_m^{(j)}$ have zero mean), this implies

$$\mathrm{Var}\left(X^{(j)}(t_j) - X^{[M](j)}(t_j)\right) \to 0 \quad \text{for } M_j \to \infty. \quad \text{(A.6)}$$

With the assumptions of Prop. 2 it further holds (cf. proof of Prop. 3) that

$$\mathrm{Var}(X^{(j)}(t_j)) = C_{jj}(t_j, t_j) \leq \|C_{jj}\|_\infty < \infty. \quad \text{(A.7)}$$

For fixed $s_i \in \mathcal{T}_i,\ t_j \in \mathcal{T}_j$ with $i, j = 1, \ldots, p$, these three properties give

$$\begin{aligned}
&\left|C_{ij}(s_i, t_j) - C_{ij}^{[M]}(s_i, t_j)\right| \\
&\quad \leq \left|\mathrm{Cov}(X^{(i)}(s_i) - X^{[M](i)}(s_i), X^{(j)}(t_j))\right| + \left|\mathrm{Cov}(X^{[M](i)}(s_i), X^{(j)}(t_j) - X^{[M](j)}(t_j))\right| \\
&\quad \overset{\text{(A.5)}}{\leq} \underbrace{\mathrm{Var}(X^{(i)}(s_i) - X^{[M](i)}(s_i))^{1/2}}_{\to 0 \text{ for } M_i \to \infty \quad \text{(A.6)}} \underbrace{C_{jj}(t_j, t_j)^{1/2}}_{< \infty \quad \text{(A.7)}} + \underbrace{C_{ii}(s_i, s_i))^{1/2}}_{< \infty \quad \text{(A.7)}} \underbrace{\mathrm{Var}(X^{(j)}(t_j) - X^{[M](j)}(t_j))^{1/2}}_{\to 0 \text{ for } M_j \to \infty \quad \text{(A.6)}}
\end{aligned}$$

Hence it holds that

$$C_{ij}^{[M]}(s_i, t_j) \to C_{ij}(s_i, t_j) \quad \text{for } M_i, M_j \to \infty. \quad \text{(A.8)}$$

The main proof is now in three steps:

1. $\Gamma^{[M]}$ converges in norm to Γ for $M_1, \ldots, M_p \to \infty$: Let $|||\cdot|||_{\mathrm{op}}$ be the operator norm induced by $|||\cdot|||$. Then

$$\begin{aligned}
\left|\left|\left|\Gamma - \Gamma^{[M]}\right|\right|\right|_{\mathrm{op}}^2 &= \sup_{|||f|||=1} \sum_{j=1}^{p} \int_{\mathcal{T}_j} \left[\sum_{i=1}^{p} \int_{\mathcal{T}_i} \left(C_{ij}(s_i, t_j) - C_{ij}^{[M]}(s_i, t_j)\right) f^{(i)}(s_i) \mathrm{d}s_i\right]^2 \mathrm{d}t_j \\
&\leq \sup_{|||f|||=1} \sum_{j=1}^{p} \int_{\mathcal{T}_j} \left[\sum_{i=1}^{p} \int_{\mathcal{T}_i} \left|(C_{ij}(s_i, t_j) - C_{ij}^{[M]}(s_i, t_j)) f^{(i)}(s_i)\right| \mathrm{d}s_i\right]^2 \mathrm{d}t_j \\
&\overset{\text{Hölder}}{\leq} \sup_{|||f|||=1} \sum_{j=1}^{p} \int_{\mathcal{T}_j} \left[\sum_{i=1}^{p} \left\|C_{ij}(\cdot, t_j) - C_{ij}^{[M]}(\cdot, t_j)\right\|_2 \left\|f^{(i)}\right\|_2\right]^2 \mathrm{d}t_j \\
&\leq \sum_{j=1}^{p} \int_{\mathcal{T}_j} \left[\sum_{i=1}^{p} \left\|C_{ij}(\cdot, t_j) - C_{ij}^{[M]}(\cdot, t_j)\right\|_2\right]^2 \mathrm{d}t_j, \quad \text{(A.9)}
\end{aligned}$$

where the last equality holds since $\left\|f^{(i)}\right\|_2 \leq 1$ for all $f \in \mathcal{H}$ with $|||f||| = 1$. The final bound for $\left|\left|\left|\Gamma - \Gamma^{[M]}\right|\right|\right|_{\text{op}}^2$ converges to zero for $M_1, \ldots, M_p \to \infty$ by (A.8), applying the Dominated Convergence Theorem twice: For the norm term in (A.9) consider

$$\left|C_{ij}(s_i, t_j) - C_{ij}^{[M]}(s_i, t_j)\right| \leq |C_{ij}(s_i, t_j)| + \left|C_{ij}^{[M]}(s_i, t_j)\right| \overset{(A.5)}{\leq} 2C_{ii}(s_i, s_i)^{1/2} C_{jj}(t_j, t_j)^{1/2},$$

thus $\left|C_{ij}(s_i, t_j) - C_{ij}^{[M]}(s_i, t_j)\right|^2 \overset{(A.7)}{\leq} 4 \left\|C_{ii}\right\|_\infty \left\|C_{jj}\right\|_\infty < \infty$. This upper bound is constant and therefore integrable over $\mathcal{T}_i$, which implies

$$\lim_{M_i \to \infty} \left\|C_{ij}(\cdot, t_j) - C_{ij}^{[M]}(\cdot, t_j)\right\|_2 = \left\|C_{ij}(\cdot, t_j) - \lim_{M_i \to \infty} C_{ij}^{[M]}(\cdot, t_j)\right\|_2.$$

For the outer integral in (A.9) the results of the norm term give

$$\left(\sum_{i=1}^p \left\|C_{ij}(\cdot, t_j) - C_{ij}^{[M]}(\cdot, t_j)\right\|_2\right)^2 \leq 4 \left\|C_{jj}\right\|_\infty \left(\sum_{i=1}^p \left(\left\|C_{ii}\right\|_\infty \lambda(\mathcal{T}_i)\right)^{1/2}\right)^2,$$

where $\lambda(\mathcal{T}_i)$ is the Lebesuge measure of $\mathcal{T}_i$ as in the Proof of Prop. 2. The term on the right hand side is constant and hence integrable over $\mathcal{T}_j$, which gives that for $M_1, \ldots, M_p \to \infty$, the limit $M_j \to \infty$ and the integral over $\mathcal{T}_j$ in (A.9) can be interchanged. In summary, these results give that $\Gamma^{[M]}$ converges to Γ in norm for $M_1, \ldots, M_p \to \infty$.

2. $\Gamma^{[M]}$ is bounded: Let $f \in \mathcal{H}$. Clearly, $\left\|f^{(i)}\right\|_2 \leq |||f|||$ for all $i = 1, \ldots, p$ and therefore

$$\begin{aligned}
\left|\left|\left|\Gamma^{[M]} f\right|\right|\right|^2 &= \sum_{j=1}^p \int_{\mathcal{T}_j} \left(\sum_{i=1}^p \int_{\mathcal{T}_i} C_{ij}^{[M]}(s_i, t_j) f^{(i)}(s_i) \mathrm{d}s_i\right)^2 \mathrm{d}t_j \\
&\overset{\text{Hölder}}{\leq} \sum_{j=1}^p \int_{\mathcal{T}_j} \left(\sum_{i=1}^p \left(\int_{\mathcal{T}_i} \left|C_{ij}^{[M]}(s_i, t_j)\right|^2 \mathrm{d}s_i\right)^{1/2} \left(\int_{\mathcal{T}_i} \left|f^{(i)}(s_i)\right|^2 \mathrm{d}s_i\right)^{1/2}\right)^2 \mathrm{d}t_j \\
&\overset{(A.5)\,(A.7)}{\leq} \sum_{j=1}^p \int_{\mathcal{T}_j} \left(\sum_{i=1}^p \left\|C_{ii}\right\|_\infty^{1/2} \left\|C_{jj}\right\|_\infty^{1/2} \lambda(\mathcal{T}_i)^{1/2} \left\|f^{(i)}\right\|_2\right)^2 \mathrm{d}t_j \\
&\leq \sum_{j=1}^p \left\|C_{jj}\right\|_\infty \lambda(\mathcal{T}_j) \left(\sum_{i=1}^p \left\|C_{ii}\right\|_\infty^{1/2} \lambda(\mathcal{T}_i)^{1/2}\right)^2 |||f|||^2 \leq p^3 C^2 T^2 |||f|||^2,
\end{aligned}$$

for $T = \max_{j=1,\ldots,p} \lambda(\mathcal{T}_j)$ and $C = \max_{j=1,\ldots,p} \left\|C_{jj}\right\|_\infty$. The value $p^{3/2} CT$ is constant and finite, hence $\Gamma^{[M]}$ is bounded.

3. Convergence results for $\nu_m^{[M]}, \psi_m^{[M]}$ and $\rho_m^{[M]}$: In Prop. 2, it was shown that Γ is compact, which implies that this operator is also bounded (Reed and Simon, 1980, Chapter VI.5.). As Γ and $\Gamma^{[M]}$ are both bounded, norm convergence is equivalent to convergence in the generalized sense (Kato, 1976, Chapter IV, §2.6., Thm. 2.23). This

implies that the eigenvalues $\nu_m^{[M]}$ of $\Gamma^{[M]}$ converge to the eigenvalues ν_m of Γ including multiplicity (if the multiplicity is finite, which holds for all non-zero eigenvalues, as Γ is compact (cf. Reed and Simon, 1980, Thm. VI.15.)) and the associated total projections converge in norm (Kato, 1976, Chapter IV, §3.5.). If the m-th eigenvalue has multiplicity 1, then the projections on the eigenspaces spanned by ψ_m and $\psi_m^{[M]}$, respectively, are given by

$$P_m f = \langle\langle \psi_m, f \rangle\rangle \psi_m, \quad P_m^{[M]} f = \langle\langle \psi_m^{[M]}, f \rangle\rangle \psi_m^{[M]}, \quad f \in \mathcal{H}.$$

Without loss of generality one may choose the orientation of ψ_m and $\psi_m^{[M]}$ such that $\langle\langle \psi_m, \psi_m^{[M]} \rangle\rangle \geq 0$. In this case, as $\psi_m, \psi_m^{[M]}$ both have norm 1,

$$\begin{aligned}
\left|\!\left|\!\left| P_m - P_m^{[M]} \right|\!\right|\!\right|_{\text{op}}^2 &\geq \left|\!\left|\!\left| \langle\langle \psi_m, \psi_m \rangle\rangle \psi_m - \langle\langle \psi_m^{[M]}, \psi_m \rangle\rangle \psi_m^{[M]} \right|\!\right|\!\right|^2 = \left|\!\left|\!\left| \psi_m - \langle\langle \psi_m^{[M]}, \psi_m \rangle\rangle \psi_m^{[M]} \right|\!\right|\!\right|^2 \\
&= |||\psi_m|||^2 - 2\langle\langle \psi_m, \langle\langle \psi_m^{[M]}, \psi_m \rangle\rangle \psi_m^{[M]} \rangle\rangle + \langle\langle \psi_m^{[M]}, \psi_m \rangle\rangle^2 \left|\!\left|\!\left| \psi_m^{[M]} \right|\!\right|\!\right|^2 \\
&= (1 - \langle\langle \psi_m^{[M]}, \psi_m \rangle\rangle)(1 + \langle\langle \psi_m^{[M]}, \psi_m \rangle\rangle) \geq (1 - \langle\langle \psi_m^{[M]}, \psi_m \rangle\rangle) \\
&= \frac{1}{2} \cdot \left|\!\left|\!\left| \psi_m - \psi_m^{[M]} \right|\!\right|\!\right|^2.
\end{aligned}$$

Norm convergence of the total projections hence implies $\left|\!\left|\!\left| \psi_m - \psi_m^{[M]} \right|\!\right|\!\right| \to 0$ for $M_1, \ldots, M_p \to \infty$.

To derive convergence of the scores $\rho_m^{[M]}$, note that for $\varepsilon > 0$ and $c := \left(\frac{2}{\varepsilon} \sum_{j=1}^p \|C_{jj}\|_\infty \lambda(\mathcal{T}_j)\right)^{1/2}$

$$\begin{aligned}
P\left(|||X||| > c\right) &\overset{\text{Markov}}{\leq} \frac{1}{c^2} \mathbb{E}\left[|||X|||^2\right] \overset{\text{Fubini}}{=} \frac{1}{c^2} \sum_{j=1}^p \int_{\mathcal{T}_j} \mathbb{E}\left[X^{(j)}(t_j)^2\right] \mathrm{d}t_j \\
&= \frac{1}{c^2} \sum_{j=1}^p \int_{\mathcal{T}_j} C_{jj}(t_j, t_j) \mathrm{d}t_j \leq \frac{1}{c^2} \sum_{j=1}^p \|C_{jj}\|_\infty \lambda(\mathcal{T}_j) = \frac{\varepsilon}{2} < \varepsilon, \qquad \text{(A.10)}
\end{aligned}$$

i.e. the norm of X is bounded in probability. Moreover,

$$\mathbb{E}\left[\left|\!\left|\!\left| X - X^{[M]} \right|\!\right|\!\right|^2\right] \overset{\text{Fubini}}{=} \sum_{j=1}^p \int_{\mathcal{T}_j} \mathbb{E}\left[\left| X^{(j)}(t_j) - X^{[M](j)}(t_j) \right|^2\right] \mathrm{d}t_j \to 0$$

for $M_1, \ldots, M_p \to \infty$, as the expectation in the integral converges uniformly to 0 and is thus bounded (by univariate Karhunen-Loève). As $\mathcal{T}_j$ has finite measure, the overall integral converges to 0. Hence $\left|\!\left|\!\left| X - X^{[M]} \right|\!\right|\!\right|$ converges in the second mean to 0, thus $\left|\!\left|\!\left| X - X^{[M]} \right|\!\right|\!\right| = o_p(1)$. Finally, this leads to

$$\begin{aligned}
\left|\rho_m - \rho_m^{[M]}\right| &= \left|\langle\langle X, \psi_m \rangle\rangle - \langle\langle X^{[M]}, \psi_m^{[M]} \rangle\rangle\right| \leq \left|\langle\langle X, \psi_m - \psi_m^{[M]} \rangle\rangle\right| + \left|\langle\langle X - X^{[M]}, \psi_m^{[M]} \rangle\rangle\right| \\
&\leq |||X||| \left|\!\left|\!\left| \psi_m - \psi_m^{[M]} \right|\!\right|\!\right| + \left|\!\left|\!\left| X - X^{[M]} \right|\!\right|\!\right| \left|\!\left|\!\left| \psi_m^{[M]} \right|\!\right|\!\right| = O_p(1) o(1) + o_p(1) = o_p(1).
\end{aligned}$$

i.e. $\rho_m^{[M]}$ converges in probability to ρ_m.

□

Lemma 2. *Under the assumptions of Prop. 7 it holds that*

$$\lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}}) \leq O_p(M_{\max} \max(N^{-1/2}, \Delta_M r_N^{\Gamma})).$$

with $\boldsymbol{Z}$ as defined in Prop. 5, $\hat{\boldsymbol{Z}} = (N-1)^{-1}\boldsymbol{\Xi}^\top\boldsymbol{\Xi}$ as in Section 3.3.2, $M_{\max} = \max_{j=1,\dots,p} M_j$ and $\Delta_M := \max_{j=1,\dots,p} \Delta_{M_j}^{(j)}$.

Proof of Lemma 2. For $j,k = 1,\dots,p$ and $f \in L^2(\mathcal{T}_j)$ define the bounded operator $\Gamma^{(jk)}\colon L^2(\mathcal{T}_j) \to L^2(\mathcal{T}_k)$ via

$$\left(\Gamma^{(jk)} f\right)(t) := \int_{\mathcal{T}_j} \operatorname{Cov}\left(X^{(j)}(s), X^{(k)}(t)\right) f(s)\mathrm{d}s = \int_{\mathcal{T}_j} C_{jk}(s,t) f(s)\mathrm{d}s$$

Analogously, define $\hat{\Gamma}^{(jk)}\colon L^2(\mathcal{T}_j) \to L^2(\mathcal{T}_k)$ by

$$\left(\hat{\Gamma}^{(jk)} f\right)(t) := \int_{\mathcal{T}_j} \widehat{\operatorname{Cov}}\left(X^{(j)}(s), X^{(k)}(t)\right) f(s)\mathrm{d}s = \int_{\mathcal{T}_j} \hat{C}_{jk}(s,t) f(s)\mathrm{d}s$$

with $\hat{C}_{jk}(s,t) := \widehat{\operatorname{Cov}}\left(X^{(j)}(s), X^{(k)}(t)\right) = \frac{1}{N}\sum_{i=1}^N X_i^{(j)}(s) X_i^{(k)}(t)$. If the X_i are independent copies of the process X, it holds

$$\begin{aligned}
&\mathbb{E}\left[\int_{\mathcal{T}_j}\int_{\mathcal{T}_k} \left(C_{jk}(s,t) - \hat{C}_{jk}(s,t)\right)^2 \mathrm{d}s\ \mathrm{d}t\right] \\
&\quad = \mathbb{E}\left[\frac{1}{N^2}\sum_{i=1}^N\sum_{l=1}^N \int_{\mathcal{T}_j}\int_{\mathcal{T}_k} \left(C_{jk}(s,t) - X_i^{(j)}(s)X_i^{(k)}(t)\right)\left(C_{jk}(s,t) - X_l^{(j)}(s)X_l^{(k)}(t)\right)\mathrm{d}s\ \mathrm{d}t\right] \\
&\quad = \frac{1}{N^2}\sum_{i=1}^N \int_{\mathcal{T}_j}\int_{\mathcal{T}_k} \mathbb{E}\left[\left(C_{jk}(s,t) - X_i^{(j)}(s)X_i^{(k)}(t)\right)^2\right]\mathrm{d}s\ \mathrm{d}t \\
&\quad = \frac{1}{N}\int_{\mathcal{T}_j}\int_{\mathcal{T}_k} \mathbb{E}\left[X^{(j)}(s)^2 X^{(k)}(t)^2\right] - C_{jk}(s,t)^2 \mathrm{d}s\ \mathrm{d}t = O(N^{-1}).
\end{aligned}$$

The last step follows from the fact that the integral term does not depend on N and is finite by assumption (A2) and the conditions in Prop. 2 for C_{jk}. This implies

$$\left\|\Gamma^{(jk)} - \hat{\Gamma}^{(jk)}\right\|_{\mathrm{op}} \leq \left(\int_{\mathcal{T}_j}\int_{\mathcal{T}_k} \left(C_{jk}(s,t) - \hat{C}_{jk}(s,t)\right)^2 \mathrm{d}s\ \mathrm{d}t\right)^{1/2} \overset{\text{Markov}}{=} O_p(N^{-1/2}).$$

Recall $Z_{ln}^{(jk)} = \operatorname{Cov}(\xi_l^{(j)}, \xi_n^{(k)})$ and $\hat{Z}_{ln}^{(jk)} = \frac{1}{N-1}\sum_{i=1}^N \hat{\xi}_{i,l}^{(j)}\hat{\xi}_{i,n}^{(k)}$ for $j,k = 1,\dots,p$, $l = 1,\dots,M_j$, $n = 1,\dots,M_k$. As $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$ are both symmetric matrices in $\mathbb{R}^{M_+\times M_+}$ it holds (cf. Horn and Johnson, 1991, Chapter 3.7)

$$\lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}}) \leq \max_{j=1,\dots,p}\max_{l=1,\dots,M_j} \sum_{k=1}^p\sum_{n=1}^{M_k} \left|Z_{ln}^{(jk)} - \hat{Z}_{ln}^{(jk)}\right|. \tag{A.11}$$

Let now $j,k=1,\dots,p,\ l=1,\dots,M_j,\ n=1,\dots,M_k$ be fixed. Assumption (A5) gives

$$
\begin{aligned}
\left|Z_{ln}^{(jk)}-\hat{Z}_{ln}^{(jk)}\right| &= \left|\mathrm{Cov}\left(\xi_l^{(j)},\xi_n^{(k)}\right)-\frac{1}{N}\sum_{i=1}^{N}\hat{\xi}_{i,l}^{(j)}\hat{\xi}_{i,n}^{(k)}-\frac{1}{N(N-1)}\sum_{i=1}^{N}\hat{\xi}_{i,l}^{(j)}\hat{\xi}_{i,n}^{(k)}\right| \\
&\overset{(A5)}{\leq} \left|\mathrm{Cov}\left(\langle X^{(j)},\phi_l^{(j)}\rangle_2,\langle X^{(k)},\phi_n^{(k)}\rangle_2\right)-\frac{1}{N}\sum_{i=1}^{N}\langle X_i^{(j)},\hat{\phi}_l^{(j)}\rangle_2\cdot\langle X_i^{(k)},\hat{\phi}_n^{(k)}\rangle_2\right| \\
&\quad+\frac{1}{N(N-1)}\sum_{i=1}^{N}\left|\langle X_i^{(j)},\hat{\phi}_l^{(j)}\rangle_2\right|\left|\langle X_i^{(k)},\hat{\phi}_n^{(k)}\rangle_2\right| \\
&\leq \left|\int_{\mathcal{T}_j}\int_{\mathcal{T}_k}\mathbb{E}\left(X^{(j)}(s)X^{(k)}(t)\right)\phi_l^{(j)}(s)\phi_n^{(k)}(t)\mathrm{d}s\ \mathrm{d}t-\right. \\
&\qquad\left.\int_{\mathcal{T}_j}\int_{\mathcal{T}_k}\frac{1}{N}\sum_{i=1}^{N}\left(X_i^{(j)}(s)X_i^{(k)}(t)\right)\hat{\phi}_l^{(j)}(s)\hat{\phi}_n^{(k)}(t)\mathrm{d}s\ \mathrm{d}t\right| \\
&\quad+\frac{1}{N(N-1)}\sum_{i=1}^{N}\left\|X_i^{(j)}\right\|_2\left\|\hat{\phi}_l^{(j)}\right\|_2\left\|X_i^{(k)}\right\|_2\left\|\hat{\phi}_n^{(k)}\right\|_2 \\
&\overset{(A.10)}{=} \left|\int_{\mathcal{T}_j}\int_{\mathcal{T}_k}\mathrm{Cov}\left(X^{(j)}(s),X^{(k)}(t)\right)\phi_l^{(j)}(s)\phi_n^{(k)}(t)-\widehat{\mathrm{Cov}}\left(X^{(j)}(s),X^{(k)}(t)\right)\hat{\phi}_l^{(j)}(s)\hat{\phi}_n^{(k)}(t)\mathrm{d}s\ \mathrm{d}t\right| \\
&\quad+\frac{1}{N(N-1)}\sum_{i=1}^{N}O_p(1)\cdot 1\cdot O_p(1)\cdot 1 \\
&= \left|\int_{\mathcal{T}_j}\int_{\mathcal{T}_k}C_{jk}(s,t)\left[\phi_l^{(j)}(s)\phi_n^{(k)}(t)-\hat{\phi}_l^{(j)}(s)\hat{\phi}_n^{(k)}(t)\right]\mathrm{d}s\ \mathrm{d}t\right| \\
&\quad+\left|\int_{\mathcal{T}_j}\int_{\mathcal{T}_k}\left[C_{jk}(s,t)-\hat{C}_{jk}(s,t)\right]\hat{\phi}_l^{(j)}(s)\hat{\phi}_n^{(k)}(t)\mathrm{d}s\ \mathrm{d}t\right|+O_p(N^{-1}) \\
&\leq \int_{\mathcal{T}_j}\int_{\mathcal{T}_k}C_{jj}(s,s)^{1/2}C_{kk}(t,t)^{1/2}\left|\phi_l^{(j)}(s)\phi_n^{(k)}(t)-\hat{\phi}_l^{(j)}(s)\hat{\phi}_n^{(k)}(t)\right|\mathrm{d}s\ \mathrm{d}t \\
&\quad+\int_{\mathcal{T}_k}\left|\left((\Gamma^{(jk)}-\hat{\Gamma}^{(jk)})\hat{\phi}_l^{(j)}\right)(t)\hat{\phi}_n^{(k)}(t)\right|\mathrm{d}t+O_p(N^{-1}) \\
&\leq \|C_{jj}\|_\infty^{1/2}\|C_{kk}\|_\infty^{1/2}\left(\int_{\mathcal{T}_j}\int_{\mathcal{T}_k}\left|\phi_l^{(j)}(s)\right|\left|\phi_n^{(k)}(t)-\hat{\phi}_n^{(k)}(t)\right|\mathrm{d}s\ \mathrm{d}t+\right. \\
&\qquad\left.\int_{\mathcal{T}_j}\int_{\mathcal{T}_k}\left|\phi_l^{(j)}(s)-\hat{\phi}_l^{(j)}(s)\right|\left|\hat{\phi}_n^{(k)}(t)\right|\mathrm{d}s\ \mathrm{d}t\right) \\
&\quad+\left\|(\Gamma^{(jk)}-\hat{\Gamma}^{(jk)})\hat{\phi}_l^{(j)}\right\|_2\left\|\hat{\phi}_n^{(k)}\right\|_2+O_p(N^{-1}) \\
&\leq \left(\|C_{jj}\|_\infty\|C_{kk}\|_\infty\lambda(\mathcal{T}_j)\lambda(\mathcal{T}_k)\right)^{1/2}\left(\left\|\phi_l^{(j)}\right\|_2\left\|\phi_n^{(k)}-\hat{\phi}_n^{(k)}\right\|_2+\left\|\phi_l^{(j)}-\hat{\phi}_l^{(j)}\right\|_2\left\|\hat{\phi}_n^{(k)}\right\|_2\right) \\
&\quad+\left\|\Gamma^{(jk)}-\hat{\Gamma}^{(jk)}\right\|_{\mathrm{op}}\left\|\hat{\phi}_l^{(j)}\right\|_2\left\|\hat{\phi}_n^{(k)}\right\|_2+O_p(N^{-1}) \\
&= \left(\|C_{jj}\|_\infty\|C_{kk}\|_\infty\lambda(\mathcal{T}_j)\lambda(\mathcal{T}_k)\right)^{1/2}\left(\left\|\phi_n^{(k)}-\hat{\phi}_n^{(k)}\right\|_2+\left\|\phi_l^{(j)}-\hat{\phi}_l^{(j)}\right\|_2\right)+\left\|\Gamma^{(jk)}-\hat{\Gamma}^{(jk)}\right\|_{\mathrm{op}}+O_p(N^{-1}) \\
&= O_p(\Delta_{M_j}^{(j)}r_N^\Gamma)+O_p(\Delta_{M_k}^{(k)}r_N^\Gamma)+O_p(N^{-1/2})+O_p(N^{-1})=O_p(\max(\Delta_{M_j}^{(j)}r_N^\Gamma,\Delta_{M_k}^{(k)}r_N^\Gamma,N^{-1/2})).
\end{aligned}
$$

The rate for $\phi_n^{(k)}$ and $\phi_l^{(j)}$ in the last steps is shown at the beginning of the proof of Prop. 7. In total, equation (A.11), $M_{\max} = \max_{j=1,\dots,p} M_j$ and $\Delta_M := \max_{j=1,\dots,p} \Delta_{M_j}^{(j)}$ give

$$\lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}}) \leq O_p(M_{\max} \max(N^{-1/2}, \Delta_M r_N^{\Gamma})).$$

□

Proof of Prop. 7. Under assumption (A4) and using the convention $\lambda_0^{(j)} := \infty$, Lemma 4.3. in Bosq (2000) gives for $m = 1, \dots, M_j$

$$\begin{aligned}\left\|\phi_m^{(j)} - \hat{\phi}_m^{(j)}\right\|_2 &\leq 8^{1/2} \left[\min\left(\lambda_m^{(j)} - \lambda_{m+1}^{(j)}, \lambda_{m-1}^{(j)} - \lambda_m^{(j)}\right)\right]^{-1} \left\|\Gamma^{(j)} - \hat{\Gamma}^{(j)}\right\|_{\text{op}} \\ &\leq 8^{1/2} \Delta_{M_j}^{(j)} \left\|\Gamma^{(j)} - \hat{\Gamma}^{(j)}\right\|_{\text{op}} = O_p(\Delta_{M_j}^{(j)} r_N^{\Gamma}).\end{aligned}$$

Based on this result, Lemma 2 states that $\lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}}) \leq O_p(M_{\max} \max(N^{-1/2}, \Delta_M r_N^{\Gamma}))$ with $\Delta_M := \max_{j=1,\dots,p} \Delta_{M_j}^{(j)}$ and $M_{\max} = \max_{j=1,\dots,p} M_j$.

1. Eigenvalues: Let $\boldsymbol{\xi} \in \mathbb{R}^{M_+}$ with entries $\xi_m^{(j)} = \langle X^{(j)}, \phi_m^{(j)} \rangle_2 = \langle X^{[M](j)}, \phi_m^{(j)} \rangle_2$, $m = 1, \dots, M_j$, $j = 1, \dots, p$. For fixed $m = 1, \dots, M_+$ it holds that

$$\begin{aligned}\left|\nu_m^{[M]} - \hat{\nu}_m\right| &= \left|\operatorname{Var}(\langle\langle X_i^{[M]}, \psi_m^{[M]} \rangle\rangle) - \hat{\boldsymbol{c}}_{\boldsymbol{m}}^{\top} \hat{\boldsymbol{Z}} \hat{\boldsymbol{c}}_{\boldsymbol{m}}\right| = \left|\boldsymbol{c}_{\boldsymbol{m}}^{\top} \operatorname{Var}(\boldsymbol{\xi}) \boldsymbol{c}_{\boldsymbol{m}} - \hat{\boldsymbol{c}}_{\boldsymbol{m}}^{\top} \hat{\boldsymbol{Z}} \hat{\boldsymbol{c}}_{\boldsymbol{m}}\right| \\ &= \left|(\boldsymbol{c}_{\boldsymbol{m}} - \hat{\boldsymbol{c}}_{\boldsymbol{m}})^{\top} \boldsymbol{Z} \boldsymbol{c}_{\boldsymbol{m}} + \hat{\boldsymbol{c}}_{\boldsymbol{m}}^{\top} \boldsymbol{Z} (\boldsymbol{c}_{\boldsymbol{m}} - \hat{\boldsymbol{c}}_{\boldsymbol{m}}) + \hat{\boldsymbol{c}}_{\boldsymbol{m}}^{\top} (\boldsymbol{Z} - \hat{\boldsymbol{Z}}) \hat{\boldsymbol{c}}_{\boldsymbol{m}}\right| \\ &\leq \|\boldsymbol{c}_{\boldsymbol{m}} - \hat{\boldsymbol{c}}_{\boldsymbol{m}}\| \cdot \nu_m^{[M]} \|\boldsymbol{c}_{\boldsymbol{m}}\| + \lambda_{\max}(\boldsymbol{Z}) \|\boldsymbol{c}_{\boldsymbol{m}} - \hat{\boldsymbol{c}}_{\boldsymbol{m}}\| + \lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}}) \|\hat{\boldsymbol{c}}_{\boldsymbol{m}}\| \\ &= \|\boldsymbol{c}_{\boldsymbol{m}} - \hat{\boldsymbol{c}}_{\boldsymbol{m}}\| (\nu_m^{[M]} + \nu_1^{[M]}) + \lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}}) \\ &\leq \frac{8^{1/2} \lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}})}{\min(\nu_{m-1}^{[M]} - \nu_m^{[M]}, \nu_m^{[M]} - \nu_{m+1}^{[M]})} 2\nu_1^{[M]} + \lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}}) \\ &= \left[\frac{2^{5/2} \nu_1^{[M]}}{\min(\nu_{m-1}^{[M]} - \nu_m^{[M]}, \nu_m^{[M]} - \nu_{m+1}^{[M]})} + 1\right] \lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}}) \\ &= O_p(M_{\max} \max(N^{-1/2}, \Delta_M r_N^{\Gamma})),\end{aligned}$$

   as the expression in square brackets converges to a constant $C < \infty$ (cf. Prop. 6 and the fact that ν_m is assumed to have multiplicity 1, see p. 41 in Chapter 3). Here $\lambda_{\max}(\boldsymbol{A})$ denotes the maximal eigenvalue of a symmetric matrix $\boldsymbol{A}$. The second inequality follows from Corollary 1 in Yu et al. (2015) and the fact that $\nu_m^{[M]} \leq \nu_1^{[M]}$.

2. Eigenfunctions: Consider the j-th element of the m-th eigenfunctions:

$$\begin{aligned}
\left\|\psi_m^{[M](j)} - \hat{\psi}_m^{(j)}\right\|_2 &\leq \sum_{n=1}^{M_j} \left|[\boldsymbol{c_m}]_n^{(j)} - [\hat{\boldsymbol{c}}_{\boldsymbol{m}}]_n^{(j)}\right| \left\|\phi_n^{(j)}\right\|_2 + \left|[\hat{\boldsymbol{c}}_{\boldsymbol{m}}]_n^{(j)}\right| \left\|\phi_n^{(j)} - \hat{\phi}_n^{(j)}\right\|_2 \\
&\leq M_j^{1/2} \|\boldsymbol{c_m} - \hat{\boldsymbol{c}}_{\boldsymbol{m}}\| + M_j^{1/2} \|\hat{\boldsymbol{c}}_{\boldsymbol{m}}\| O_p(\Delta_{M_j}^{(j)} r_N^{\Gamma}) \\
&\leq M_j^{1/2} \left(\frac{8^{1/2} \lambda_{\max}(\boldsymbol{Z} - \hat{\boldsymbol{Z}})}{\min(\nu_{m-1}^{[M]} - \nu_m^{[M]}, \nu_m^{[M]} - \nu_{m+1}^{[M]})} + O_p(\Delta_{M_j}^{(j)} r_N^{\Gamma}) \right) \\
&= M_j^{1/2} O_p\left(\max(M_{\max} N^{-1/2}, M_{\max} \Delta_M r_N^{\Gamma}, \Delta_{M_j}^{(j)} r_N^{\Gamma}) \right),
\end{aligned}$$

where the last inequality uses again Corollary 1 in Yu et al. (2015). By definition of the norm, the result for the single elements implies

$$\left|\left|\left|\psi_m^{[M]} - \hat{\psi}_m\right|\right|\right| = O_p\left(M_{\max}^{3/2} \max(N^{-1/2}, \Delta_M r_N^{\Gamma}) \right).$$

3. Scores and reconstructed $\hat{X}$: For $\hat{\rho}_{i,m} = \boldsymbol{\Xi_{i,.}} \hat{\boldsymbol{c}}_{\boldsymbol{m}} = \langle\langle \hat{X}_i^{[M]}, \hat{\psi}_m \rangle\rangle$ as in Section 3.3.2 with $\hat{X}_i^{[M](j)} = \sum_{m=1}^{M_j} \hat{\xi}_{i,m}^{(j)} \hat{\phi}_m^{(j)}$,

$$\begin{aligned}
\left|\rho_{i,m}^{[M]} - \hat{\rho}_{i,m}\right| &= \left|\langle\langle X_i^{[M]}, \psi_m^{[M]} - \hat{\psi}_m \rangle\rangle + \langle\langle X_i^{[M]} - \hat{X}_i^{[M]}, \hat{\psi}_m \rangle\rangle\right| \\
&\leq \left|\left|\left| X_i^{[M]} \right|\right|\right| \cdot \left|\left|\left| \psi_m^{[M]} - \hat{\psi}_m \right|\right|\right| + \left|\left|\left| X_i^{[M]} - \hat{X}_i^{[M]} \right|\right|\right| \cdot \left|\left|\left| \hat{\psi}_m \right|\right|\right|.
\end{aligned}$$

$\left|\left|\left| X_i^{[M]} \right|\right|\right|$ is bounded in probability using (A.5) and analogous arguments as for $|||X|||$ in the proof of Prop. 6 (convergence of $\rho_m^{[M]}$). For the second term, note that

$$\begin{aligned}
\left\|X_i^{[M](j)} - \hat{X}_i^{[M](j)}\right\|_2 &= \left\| \sum_{m=1}^{M_j} \xi_{i,m}^{(j)} \phi_m^{(j)} - \hat{\xi}_{i,m}^{(j)} \hat{\phi}_m^{(j)} \right\|_2 \\
&\leq \sum_{m=1}^{M_j} \left|\xi_{i,m}^{(j)}\right| \left\|\phi_m^{(j)} - \hat{\phi}_m^{(j)}\right\|_2 + \left|\xi_{i,m}^{(j)} - \hat{\xi}_{i,m}^{(j)}\right| \left\|\hat{\phi}_m^{(j)}\right\|_2 \\
&\leq \sum_{m=1}^{M_j} \left|\xi_{i,m}^{(j)}\right| \left\|\phi_m^{(j)} - \hat{\phi}_m^{(j)}\right\|_2 + \left|\langle X_i^{(j)}, \phi_m^{(j)} \rangle_2 - \langle X_i^{(j)}, \hat{\phi}_m^{(j)} \rangle_2\right| \\
&\leq \sum_{m=1}^{M_j} \left(\left|\xi_{i,m}^{(j)}\right| + \left\|X_i^{(j)}\right\|_2 \right) \left\|\phi_m^{(j)} - \hat{\phi}_m^{(j)}\right\|_2.
\end{aligned}$$

The univariate scores are uniformly bounded in probability: For $m = 1, \ldots, M_j$, let $\varepsilon > 0$ and $c := (\frac{2\lambda_1^{(j)}}{\varepsilon})^{1/2} < \infty$. Then

$$P(|\xi_{i,m}^{(j)}| > c) \overset{\text{Markov}}{\leq} \frac{1}{c^2} \mathbb{E}[|\xi_{i,m}^{(j)}|^2] = \frac{1}{c^2} \operatorname{Var}(\xi_{i,m}^{(j)}) = \frac{\lambda_m^{(j)}}{c^2} < \varepsilon.$$

Hence $\left\|X_i^{[M](j)} - \hat{X}_i^{[M](j)}\right\|_2 = M_j O_p(1) O_p(\Delta_{M_j}^{(j)} r_N^\Gamma)$ and

$$\left|\left|\left|X_i^{[M]} - \hat{X}_i^{[M]}\right|\right|\right| = O_p(M_{\max} \Delta_M r_N^\Gamma).$$

In total,

$$\begin{aligned}
\left|\rho_{i,m}^{[M]} - \hat{\rho}_{i,m}\right| &\leq \left|\left|\left|X_i^{[M]}\right|\right|\right| \left|\left|\left|\psi_m^{[M]} - \hat{\psi}_m\right|\right|\right| + \left|\left|\left|X_i^{[M]} - \hat{X}_i^{[M]}\right|\right|\right| \\
&= O_p(1) O_p(M_{\max}^{3/2} \max(N^{-1/2}, \Delta_M r_N^\Gamma)) + O_p(M_{\max} \Delta_M r_N^\Gamma) \\
&= O_p(M_{\max}^{3/2} \max(N^{-1/2}, \Delta_M r_N^\Gamma)).
\end{aligned}$$

□

A.2. Simulation – Additional Results

A.2.1. Construction of Eigenfunctions (Technical Details)

Setting 1 and 2: The first two settings of the simulation study consider multivariate functional data where each element has a one-dimensional domain (cf. Section 3.4.1). As a starting point for the construction of the multivariate eigenfunctions ψ_m with p elements, we use Fourier basis functions $f_1, \ldots, f_M$ on the interval $[0, 2]$. Next, choose split points $0 = T_1 < T_2 < \ldots < T_p < T_{p+1} = 2$ and shift values $\eta_1, \ldots, \eta_p \in \mathbb{R}$ such that $\mathcal{T}_j = [T_j + \eta_j, T_{j+1} + \eta_j]$. In the first setting with $p = 2$, one has $T_1 = 0, T_2 = 1, T_3 = 2$ and $\eta_1 = 0, \eta_2 = 1$, i.e. the functions are cut at $T_2 = 1$, and the second part is shifted to the left by 1 such that $\mathcal{T}_1 = \mathcal{T}_2 = [0, 1]$. Given random signs $\sigma_1, \ldots, \sigma_p \in \{-1, 1\}$, the multivariate eigenfunctions are given by their elements

$$\psi_m^{(j)}(t_j) = \sigma_j \cdot f_m|_{[T_j, T_{j+1}]}(t_j - \eta_j), \quad m = 1, \ldots, M.$$

The constuction process is illustrated in Fig. A.1. Clearly, $\{\psi_m,\ m = 1, \ldots, M\}$ is an orthonormal system in $\mathcal{H} = L^2(\mathcal{T}_1) \times \ldots \times L^2(\mathcal{T}_p)$. The observations x_i for the simulation are constructed as a truncated Karhunen-Loève expansion, cf. the introduction of Section 3.4. Exemplary data for the second simulation setting including sparse data and data with measurement error is given in Fig. A.2

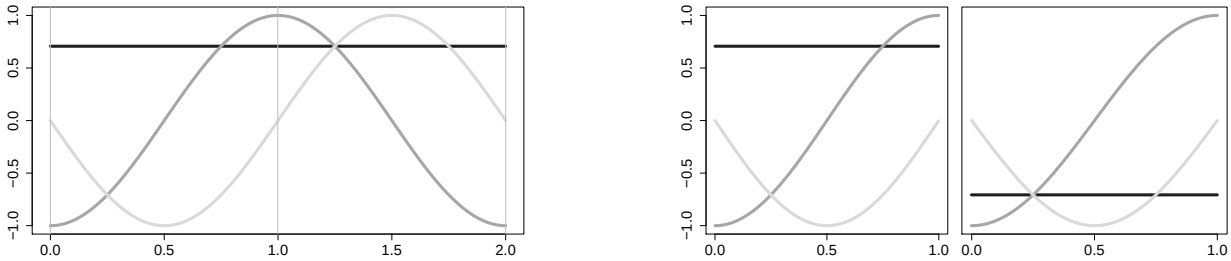


Figure A.1.: Illustration of the construction of the multivariate eigenfunctions ψ_m for the first setting. Left: The first $M = 3$ functions of the Fourier basis on $[0, 2]$ with one split point. Right: The shifted pieces multiplied with random signs form the first three bivariate eigenfunctions.

Setting 3: The data in the third setting consists of images and functions, hence multivariate functional data with elements having different dimensional domains (cf. Section 3.4.2). The basic idea here is to find orthonormal bases for each of the domains and to construct the eigenfunctions as weighted combinations of those bases. Specifically, we use five Fourier basis functions $f_{m_1}^{(1,1)},\ f_{m_2}^{(1,2)}$ on $[0, 1]$ or $[0, 0.5]$, respectively, to form $M = 25$ tensor product

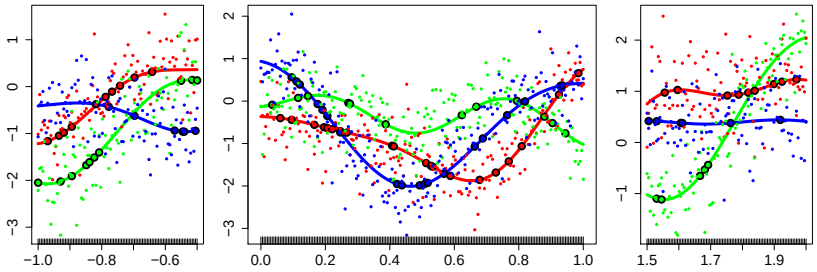


Figure A.2.: Three examples for simulated data in simulation setting 2 based on the leading $M = 8$ Fourier basis functions and exponential eigenvalue decay. Left: $x_i^{(1)}$, Middle: $x_i^{(2)}$, Right: $x_i^{(3)}$. Solid lines show the realizations x_i, small points are the corresponding data with measurement error, big points mark measurements of the artificially sparsified data (high sparsity level).

functions $f_m^{(1)}$ on $[0, 1] \times [0, 0.5]$ and $M = 25$ Legendre Polynomials $f_m^{(2)}$ on $[-1, 1]$. The eigenfunctions are defined via

$$\psi_m^{(1)}(s,t) = \sqrt{\alpha} f_{m_1}^{(1,1)}(s) \cdot f_{m_2}^{(1,2)}(t), \quad (s,t) \in \mathcal{T}_1 := [0,1] \times [0,0.5],$$
$$\psi_m^{(2)}(t) = \sqrt{1-\alpha} f_m^{(2)}(t), \quad t \in \mathcal{T}_2 := [-1,1]$$

with a random weight $\alpha \in (0, 1)$. This choice implies that ψ_m forms an orthonormal system in $\mathcal{H} = L^2(\mathcal{T}_1) \times L^2(\mathcal{T}_2)$. In order to avoid extreme weights, α is set to $u_1/(u_1 + u_2)$ with $u_1, u_2 \sim U(0.2, 0.8)$. This construction restricts $\alpha \in (0.2, 0.8)$ and can easily be generalized to the simulation of multivariate functional data with p elements. Example data based on this type of eigenfunctions is shown in Fig. A.3.

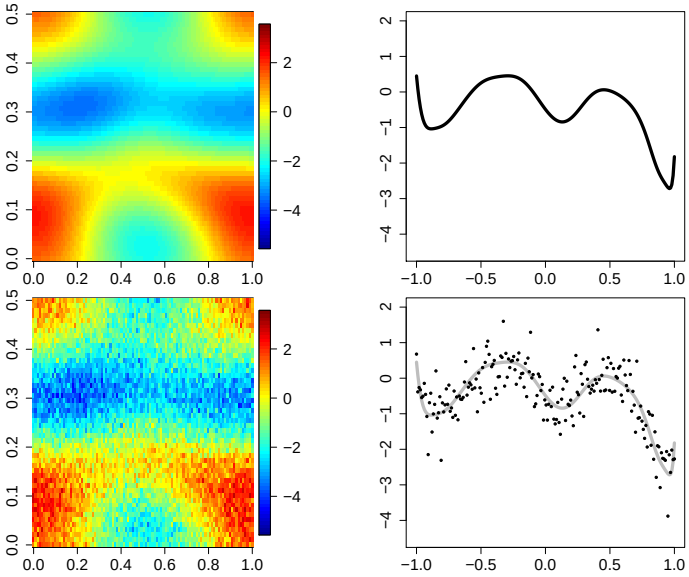


Figure A.3.: Examples for simulated data in the third simulation setting (cf. Section 3.4.2) consisting of images ($x_i^{(1)}$, left) and functions ($x_i^{(2)}$, right) without (1st row) and with measurement error (2nd row).

A.2.2. Example Fits

Table A.1.: True and estimated eigenvalues for the first simulation setting (exponential eigenvalue decay, eigenfunctions based on the first $M = 8$ Fourier basis functions) for one replication with $N = 250$ observations. The reconstruction errors are (in %) 0.007 (MFPCA, $\sigma^2 = 0$; simulation median: 0.008), 0.734 (MFPCA, $\sigma^2 = 0.25$; simulation median: 0.497), $< 10^{-3}$ (MFPCA$_{\text{RS}}$, $\sigma^2 = 0$; simulation median: $< 10^{-3}$) and 0.710 (MFPCA$_{\text{RS}}$, $\sigma^2 = 0.25$; simulation median: 0.480). The results for the corresponding eigenfunctions are given in Fig. A.4.

$m =$	1	2	3	4	5	6	7	8
True Eigenvalues	1.000	0.607	0.368	0.223	0.135	0.082	0.050	0.030
MFPCA ($\sigma^2 = 0$)	1.144	0.502	0.316	0.249	0.128	0.090	0.048	0.034
MFPCA ($\sigma^2 = 0.25$)	1.140	0.501	0.316	0.249	0.128	0.087	0.046	0.031
MFPCA$_{\text{RS}}$ ($\sigma^2 = 0$)	1.140	0.500	0.315	0.248	0.127	0.090	0.048	0.034
MFPCA$_{\text{RS}}$ ($\sigma^2 = 0.25$)	1.139	0.504	0.317	0.252	0.130	0.091	0.048	0.035

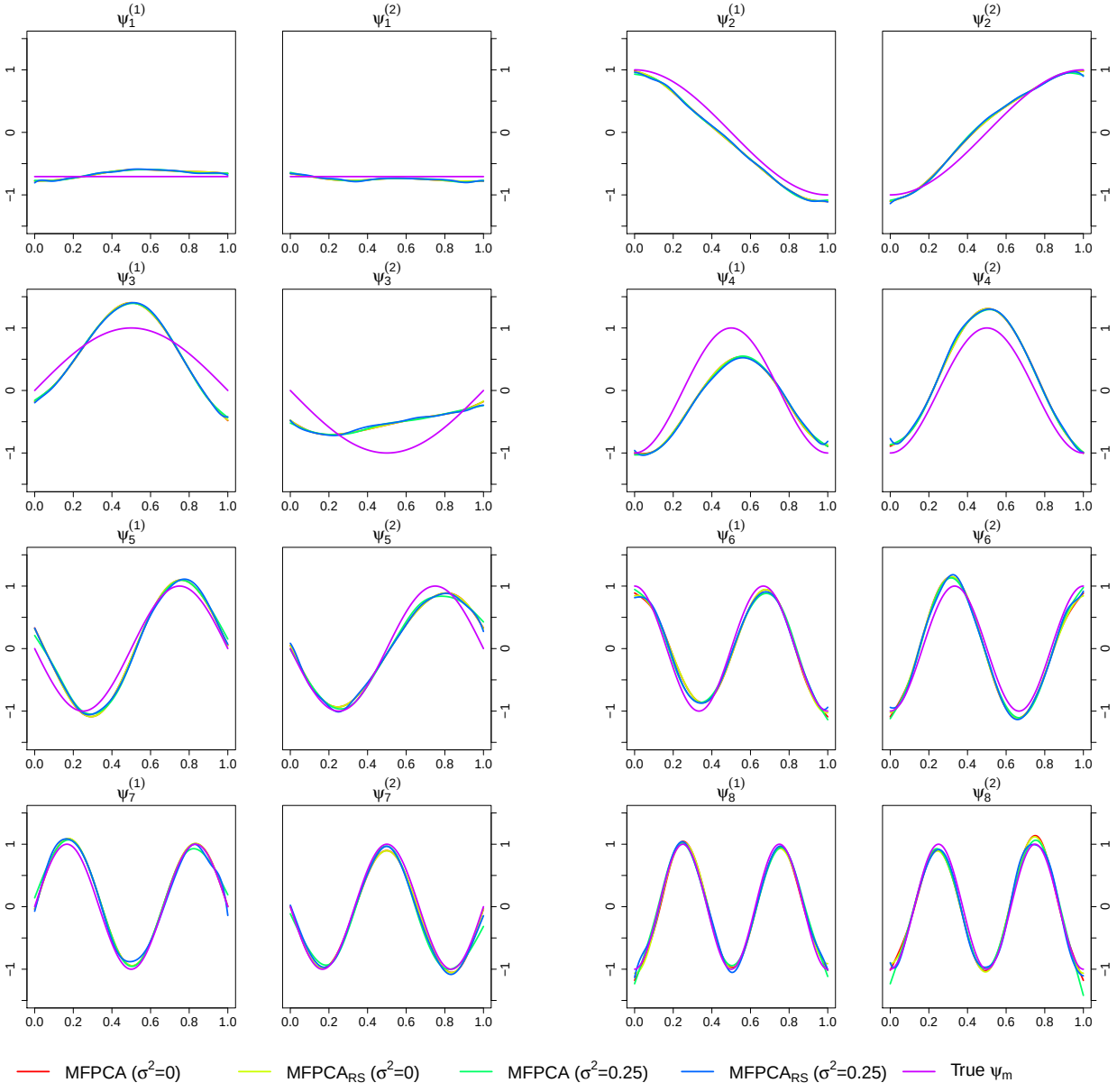


Figure A.4.: True and estimated eigenfunctions for the first setting based on one example replication with $N = 250$ observations. The results for the corresponding eigenfunctions are given in Table A.1.

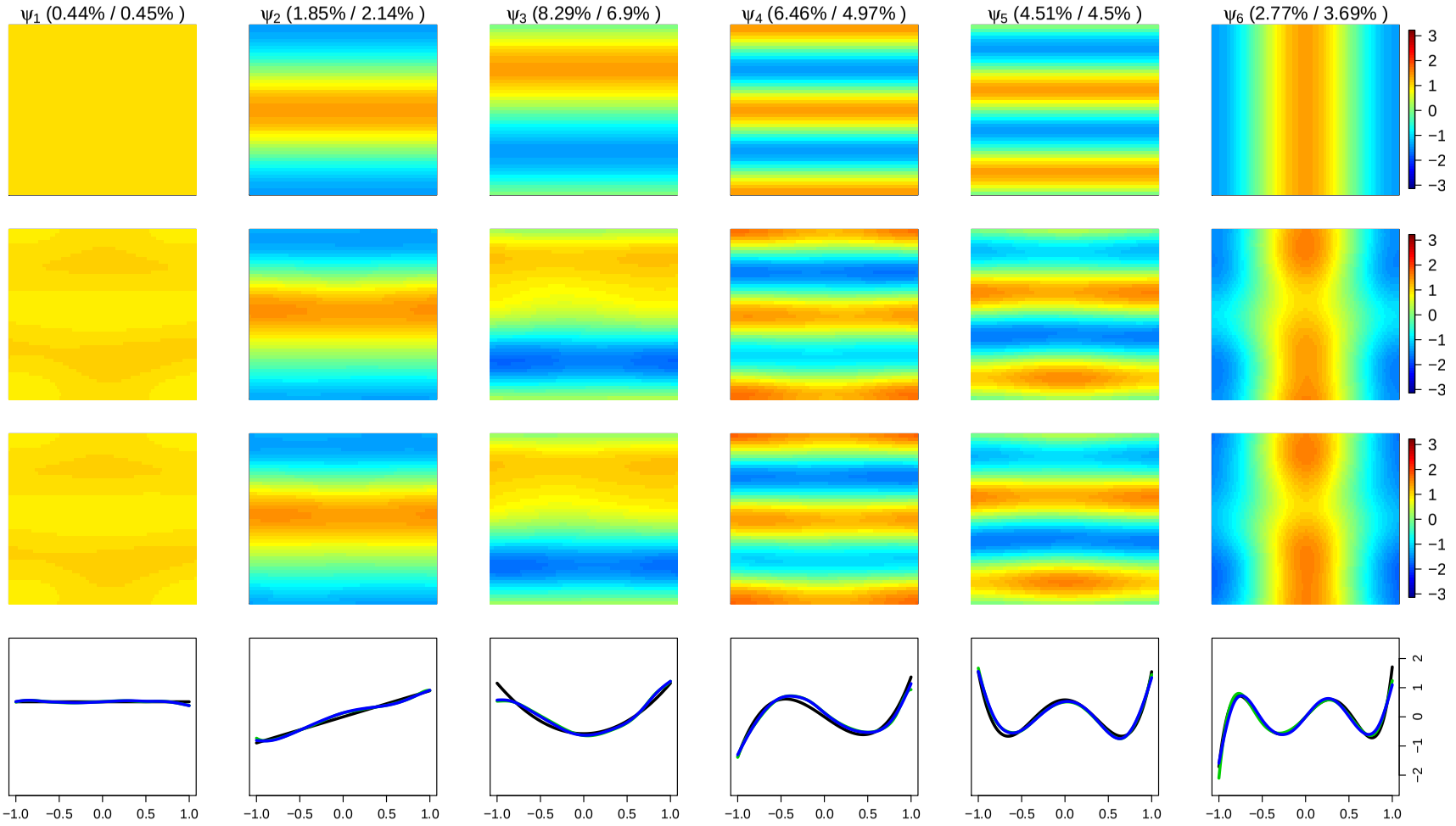


Figure A.5.: Exemplary result for one replication in simulation setting 3 and MFPCA based on univariate spline expansions (results for the eigenfunctions $\psi_1, \ldots, \psi_6$). The first row shows the true first elements $\psi_m^{(1)}$, the second/third row gives the results of $\hat{\psi}_m^{(1)}$ for data without/with measurement error. In the fourth row, the second elements $\psi_m^{(2)}$ of the true eigenfunctions are shown in black and the corresponding estimates for data without/with measurement error are shown in green/blue. Percentages in the titles give the relative errors $\text{Err}(\hat{\psi}_m)$ for the estimates based on data without/with measurement error. The reconstruction error MRSE is 0.40%/2.25% for data without/with measurement error (simulation median: 0.40%/2.05%). Results for the eigenfunctions $\psi_7, \ldots, \psi_{12}$ are shown in Fig. A.6.

Figure A.6.: Exemplary result for one replication in simulation setting 3 (results for the eigenfunctions $\psi_7, \ldots, \psi_{12}$). Please refer to Fig. A.5 for details.

A.2.3. Sensitivity Analysis

As discussed in Section 3.3.2, the number M_j of univariate eigenfunctions used for MFPCA clearly has an impact on the results, as they control how much of the information in the univariate elements is used for calculating the multivariate FPCA. A standard approach in functional data analysis for quantifying the amount of information contributed by single eigenfunctions $\phi_m^{(j)}$ is the percentage of variance explained (pve), which is the ratio of the associated eigenvalue $\lambda_m^{(j)}$ and the sum of all eigenvalues. The following simulation systematically examines the sensitivity of the MFPCA result based on the pve of the univariate eigenfunctions.

Simulation Setup: The simulation is based on 100 replications with $N = 250$ observations of bivariate data on the unit interval (cf. setting 1 in Section 3.4.1), with $M = 8$ Fourier basis functions and exponentially decreasing eigenvalues for simulating the data. The number of univariate eigenfunctions M_1, M_2 for MFPCA is chosen based on pve $\in \{0.75, 0.90, 0.95, 0.99\}$ for both elements and $M_1 = M_2 = M = 8$ for comparison. The number of multivariate principal component functions is then set to $\min\{M_1 + M_2, M\}$.

Results: The results of the sensitivity analysis are shown in Fig. A.7 and Table A.2. The number of estimated multivariate eigenvalues/eigenfunctions is for all 100 datasets $\hat{M} = 4$ for pve $= 0.75$, $\hat{M} = 6$ for pve $= 0.90$ and $\hat{M} = 8$ in all other cases. The results are as expected: Increasing the pve, and hence the information in the univariate FPCA, improves the estimation accuracy for both, multivariate eigenvalues and eigenfunctions. As a consequence, the reconstruction error reduces with increasing pve. Moreover, for a fixed m, the results show that there is a critical amount of information in univariate FPCA that is needed to describe the multivariate eigenvalues and eigenfunctions well. If this is reached (e.g. pve $= 0.95$ for $m = 5$, cf. Fig. A.7), the additional benefit of using more univariate eigenfunctions (pve > 0.95) becomes negligible. If, in contrast, the univariate FPCA does not contain enough information (pve < 0.95), the error rates for the MFPCA estimates are considerably increased. For fixed pve, the error rates rise abruptly for the last pair of eigenfunctions ($m \in \{\hat{M}_+ - 1, \hat{M}_+\}$). This is due to the fact that in this simulation, the multivariate functional principal components are derived from a Fourier basis. The last two eigenfunctions are hence sine and cosine functions with highest frequency and cannot be represented well by the univariate functions used, as they contain only functions with lower frequency, in other words, they do not contain enough information.

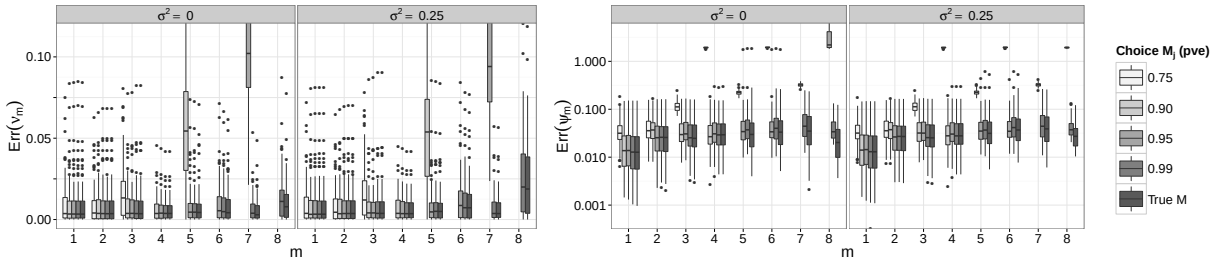


Figure A.7.: Relative errors for estimated eigenvalues (left) and eigenfunctions (right, log-scale) for the sensitivity analysis. Extreme values cut off for better comparability.

Table A.2.: Average MRSE (in %) in the sensitivity analysis.

	Choice of M_j (pve)				True M
	0.75	0.90	0.95	0.99	
$\sigma^2 = 0$	23.756	9.075	2.924	0.165	0.006
$\sigma^2 = 0.25$	24.099	9.583	3.593	0.842	0.740

A.2.4. Coverage Analysis of Pointwise Bootstrap Confidence Bands

In Section 3.5, pointwise bootstrap confidence bands were calculated for the multivariate functional principal components estimated from the ADNI data to quantify the variability in the estimates. The following simulation study examines the coverage properties of such confidence bands.

Simulation Setup: The data generating process is the same as in the simulation in Section 3.4.2, mimicking the ADNI data that consists of functions on a one-dimensional domain and images. In total, the simulation is based on 100 datasets, all having $N = 250$ observations. Each dataset is considered with and without measurement error. Both elements are represented in terms of B-spline basis functions with appropriate smoothness penalties in the presence of measurement error (cf. Section 3.4.2). For each dataset and each estimated eigenfunction, a pointwise 95% bootstrap confidence band is calculated based on 100 bootstrap samples on the level of subjects (cf. Section 3.5). The coefficients of the spline basis decompositions can efficiently be reused when bootstrapping, as the basis is fixed and does not depend on the bootstrap sample. In contrast, the univariate functional principal components for the ADAS-Cog trajectories in the ADNI application have to be re-estimated for each bootstrap sample. This computational aspect is taken into account in the bootstrap implementation in the `MFPCA` package (Happ, 2017b). Finally, the confidence bands are calculated separately for each element as pointwise percentile bootstrap confidence intervals.
For each eigenfunction and each observation point, the estimated coverage at one point $t_j \in \mathcal{T}_j$ is the percentage of datasets for which the true eigenfunction $\psi_m^{(j)}$ evaluated at t_j

is enclosed in the bootstrap confidence band (up to a sign change of the whole function). Fig. A.8 shows the estimated coverages of the elements of the eigenfunctions for data with and without measurement error aggregated over the observation points.

Results: If the data is observed without measurement error, the pointwise confidence bands enclose the true functions fairly precisely in 95% of all cases with very little variation between the observation points. For the leading eigenfunctions, the same holds true if the data is observed with measurement error. For higher order eigenfunctions, that explain hardly any variation in the data, the estimated coverage decreases, especially for the second element ($\psi_m^{(2)}$, one-dimensional domain) and shows a much higher variation between the observation points. On the one hand this may be caused by the fact that the true eigenfunctions $\psi_m^{(2)}$ have a stronger curvature for growing m (cf. Fig. A.6). Severe undercoverage for higher order eigenfunctions occurs mainly in regions of high curvature and slope of the eigenfunctions, where the low signal-to-noise level leads to oversmoothing (cf. Fig. A.9). On the other hand, the results of Section 3.4.2 show that the estimates for higher order eigenfunction elements become more inaccurate due to interchanging of eigenfunctions, hence the bootstrap confidence bands can be centered incorrectly. For the image elements $\psi_m^{(1)}$, the bootstrapped confidence bands give much better results, except for some outliers that form spatially smooth outlying regions (see e.g. Fig. A.9). This reflects that the pointwise coverages are not independent, as the true eigenfunctions as well as the confidence bands are smooth: If the function $\psi_m^{(j)}$ lies within the bootstrap confidence band at a point t_j, it is very likely that it will also be inside the confidence band at the neighbouring observation points (analogously for points outside the CI). This relation is highlighted in Fig. A.9, which illustrates the coverage rates for ψ_3 (having a good coverage) and ψ_9 (having a rather poor coverage) in the case of measurement error. In summary, the results of the simulation show that the bootstrapped confidence bands give reliable results, in particular for the leading eigenfunctions that explain most of the variation in the data. Moreover, smooth eigenfunctions will have a stabilizing effect for the coverage. However, when interpreting such pointwise confidence bands, one should keep in mind the dependence across neighbouring observation points due to the smoothness of the eigenfunctions.

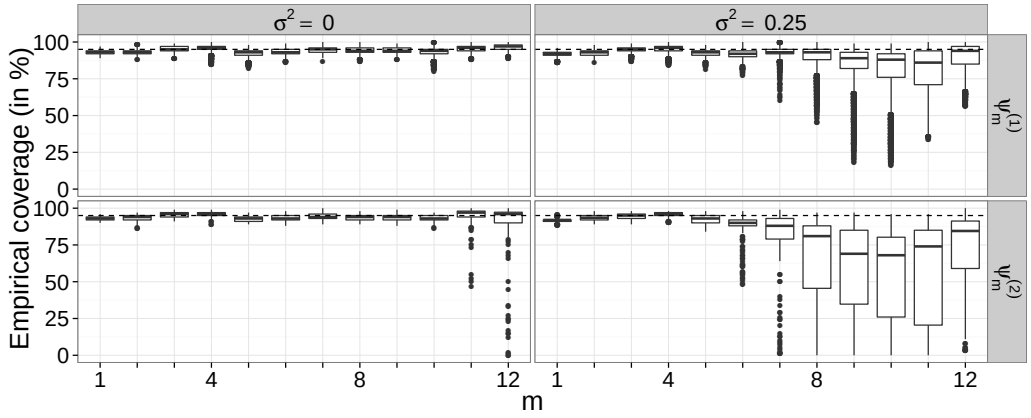


Figure A.8.: Empirical coverages from the bootstrap simulation study for data without ($\sigma^2 = 0$) and with ($\sigma^2 = 0.25$) measurement error. The boxplots show the pointwise coverage of the bootstrap confidence bands aggregated over the corresponding domains for both elements of the true eigenfunctions ψ_m, $m = 1, \ldots, 12$ (1st row: Image element $\psi_m^{(1)}$, 2nd row: Element $\psi_m^{(2)}$ with one-dimensional domain). The dashed line marks a coverage of 95%.

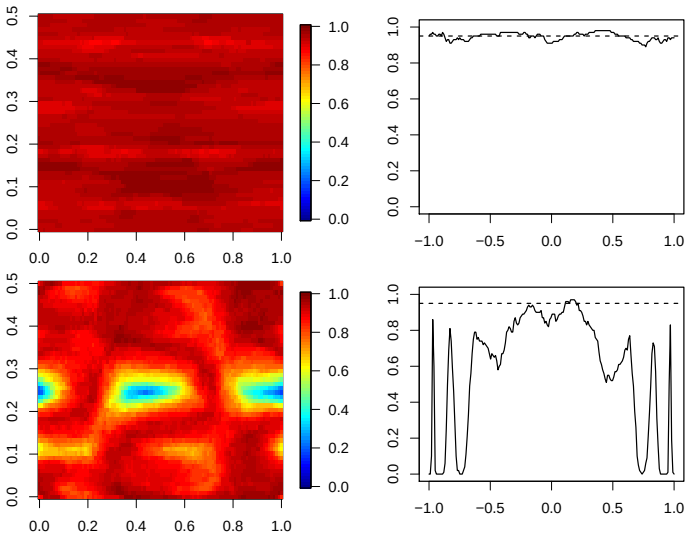


Figure A.9.: Exemplary results from the bootstrap simulation study for data observed with measurement error. The first row shows the estimated coverages for the third eigenfunction, the second row shows the estimated coverages for the eigenfunction of order 9 (see also Fig. A.8). The first column corresponds to the estimated elements $\hat{\psi}_m^{(1)}$ and the second column corresponds to the estimated elements $\hat{\psi}_m^{(2)}$. For the latter, the dashed lines correspond to the nominal level of 95%.

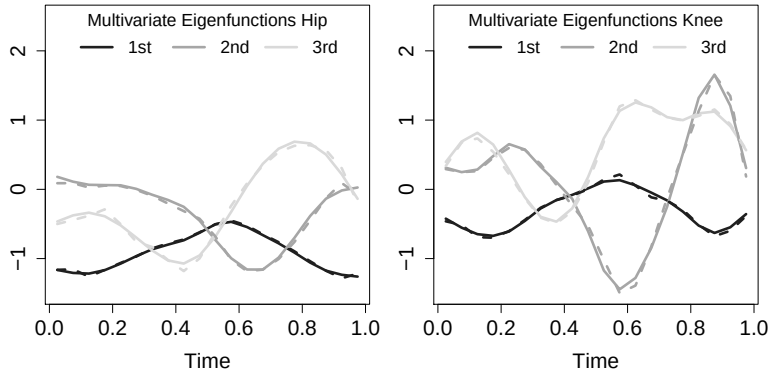


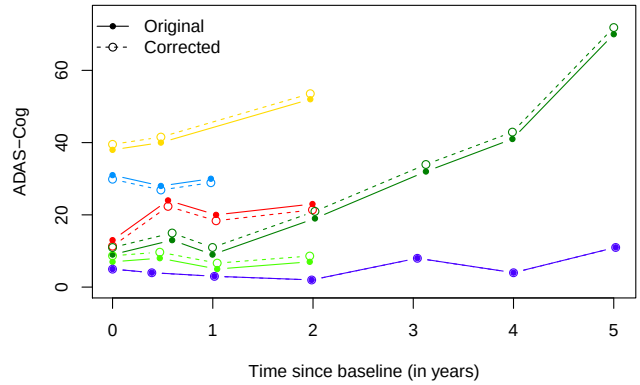
Figure A.10.: The first three estimated bivariate eigenfunctions for the gait data set. Solid lines show the results of the new MFPCA approach, dashed lines correspond to the approach of Ramsay and Silverman (2005). The functions have been reflected, if necessary, for comparison purposes.

A.3. Applications – Gait Cycle Data

For comparison to an existing method in the special case of densely sampled bivariate data on the same one-dimensional interval, the new MFPCA approach is applied to the gait cycle data (cf. Fig. 3.1 in the main document) and compared to the method of Ramsay and Silverman (2005) as implemented in the `R`-package `fda` (Ramsay, Wickham, et al., 2014). The results are shown in Fig. A.10. For the new approach, the multivariate principal components are calculated based on univariate FPCA with $M_1 = M_2 = 5$ principal components. For MFPCA_{RS}, the observed functions are presmoothed using $K = 15$ cubic spline basis functions as in the simulation study (cf. Section 3.4.1). As for synthetic data, the two methods give nearly identical results.

B. Appendix for Chapter 4

B.1. Supplementary Material



RID	Gender	Age (BI)	EDU
2045	Male	72.3	16
4728	Male	**81.9**	16
4015	**Female**	73.6	**20**
4194	Male	**62**	16
4696	**Female**	73	16
4192	Male	**82.2**	**12**

Figure B.1.: Illustration of the ADAS-Cog correction for confounding variables. Left: Original and corrected ADAS-Cog trajectories for six example subjects, selected to systematically differ from the reference values. Right: Subject roster identification (RID) and demographic values for all example subjects, coded by color. Bold entries mark systematic differences from the reference values.

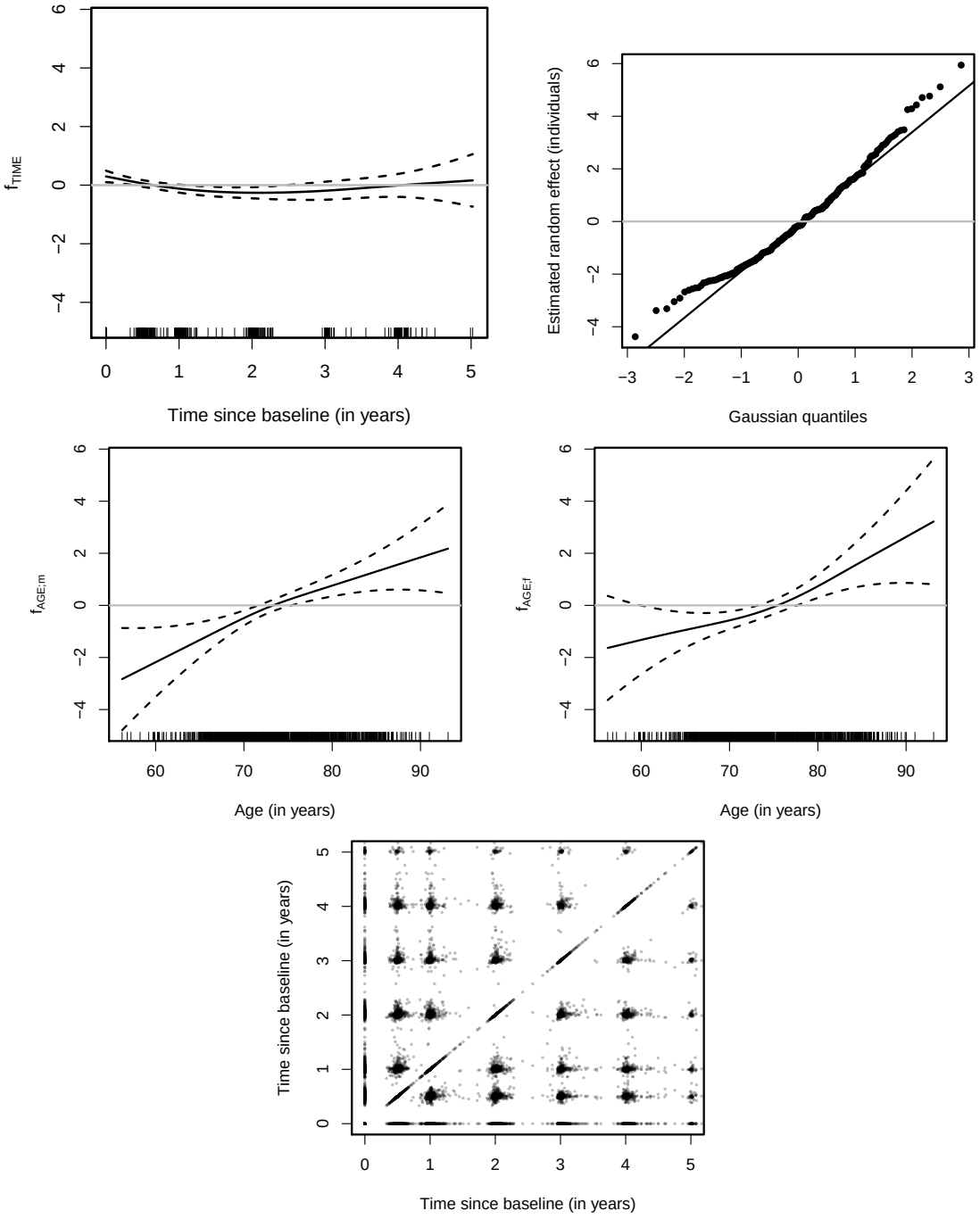


Figure B.2.: Smooth and random effects for the ADAS-Cog correction based on the exact exam dates. The gray horizontal lines indicate zero. 1st row: Estimated effect for time including 95% confidence bands (left) and QQ plot for the random effect of individuals (right). 2nd row: Estimated age effect in males (left) and estimated effect for age in females (right). 3rd row: Scatterplot of the individual times t_{ij}, t_{ik}, revealing the variability in the examination dates, which is advantageous to univariate functional principal component analysis.

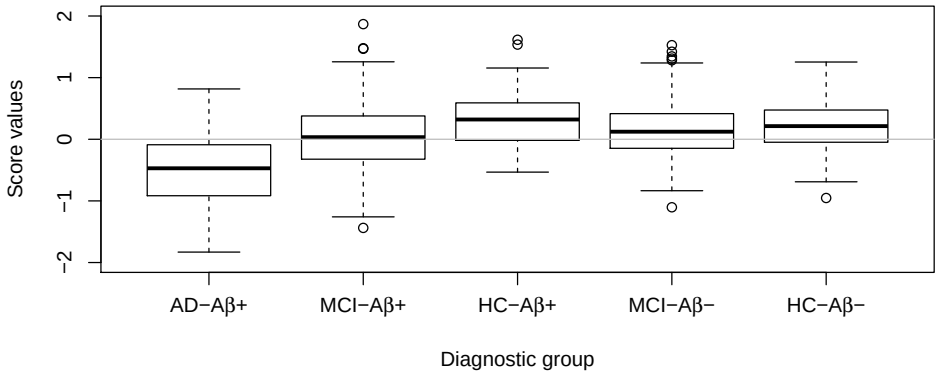


Figure B.3.: MFPCA score values for the second principal component in the whole sample MFPCA depending on diagnostic groups. The gray horizontal line marks zero.

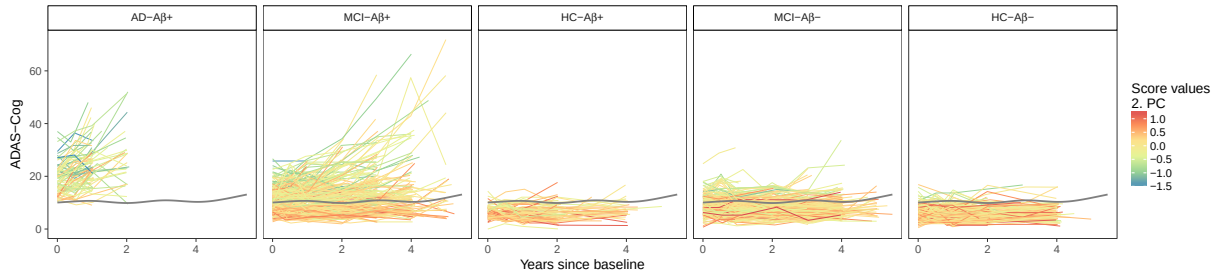


Figure B.4.: Corrected ADAS-Cog trajectories by diagnostic group. The colors of the lines correspond to the score values for the second principal component in the whole sample MFPCA. For reasons of clarity, the 1% subjects having the highest and lowest score values, each, have been removed from the plot. The gray curve marks the smooth overall mean.

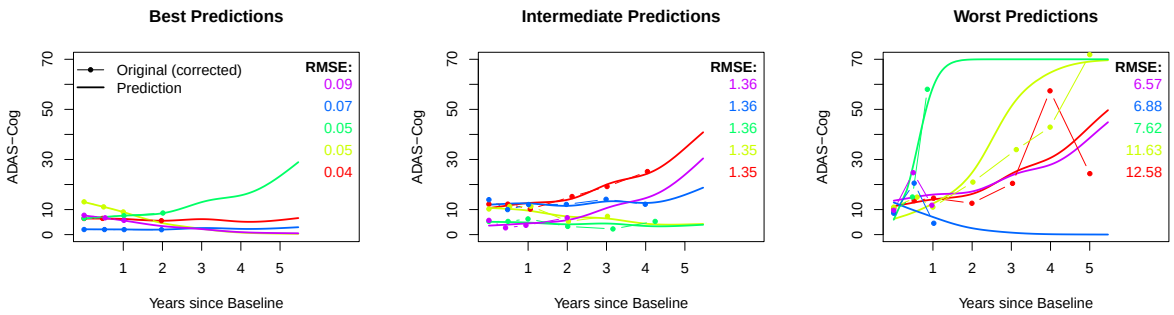


Figure B.5.: Example reconstructions of ADAS-Cog trajectories based on the whole sample MPFCA for five subjects, each, according to their (in-sample) prediction accuracy. The dots represent the original ADAS-Cog scores corrected for age, gender and education. The solid lines represent the reconstructed trajectories. Figures on the right of the plots give the RMSE for each subject, coded by color.

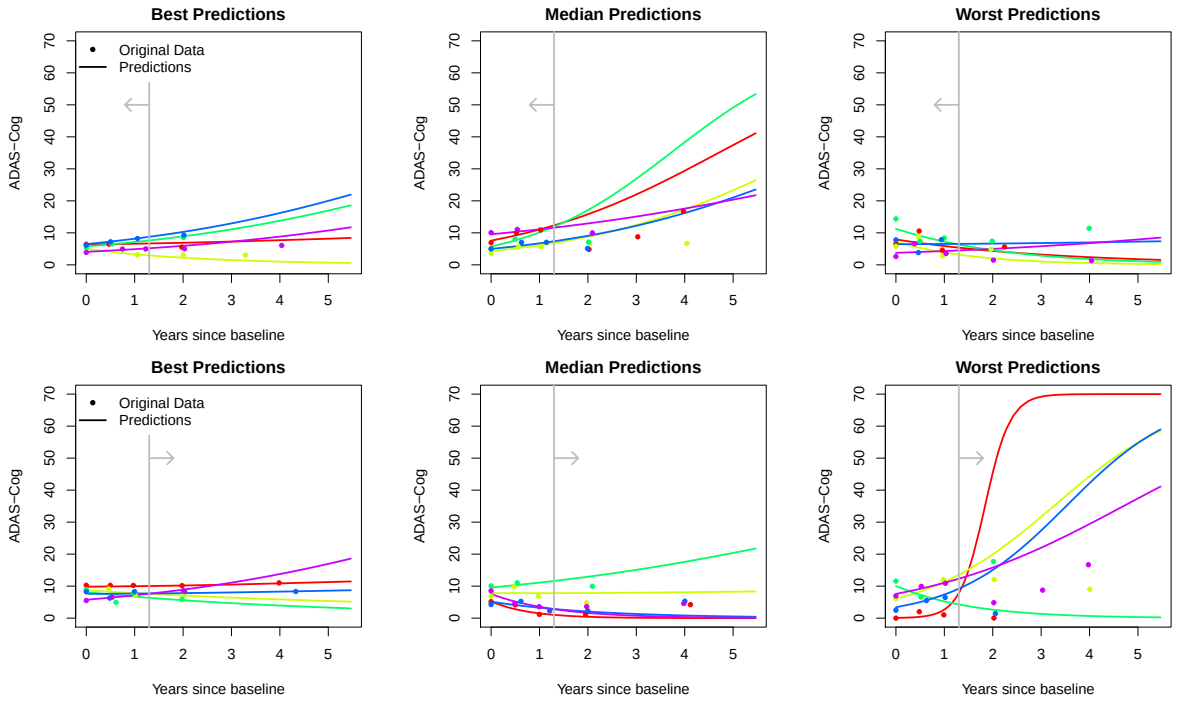


Figure B.6.: Example predictions of ADAS-Cog trajectories for HC-$A\beta$+ based on the MFPCA for MCI-$A\beta$+, according to their (out-of-sample) prediction accuracy within the first follow-up period (top row) and within the second follow-up period (bottom row). The gray vertical line marks the split point between the first and the second follow-up period. The gray arrows indicate which period was considered for calculating the RMSE.

C. Appendix for Chapter 5

Simulation Toolbox: Example Plots

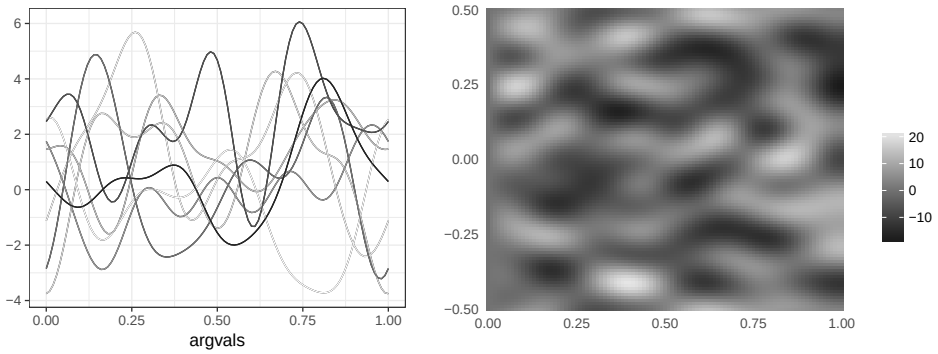


Figure C.1.: Left: $N = 8$ simulated curves on $[0, 1]$ based on the first $M = 10$ Fourier basis functions and eigenvalues with a linear decrease. Right: One simulated image on $[0, 1] \times [-0.5, 0.5]$ based on tensor products of $M_1 = 10$ eigenfunctions of the Wiener process on $[0, 1]$ and $M_2 = 12$ Fourier basis functions on $[-0.5, 0.5]$ and linearly decreasing eigenvalues.

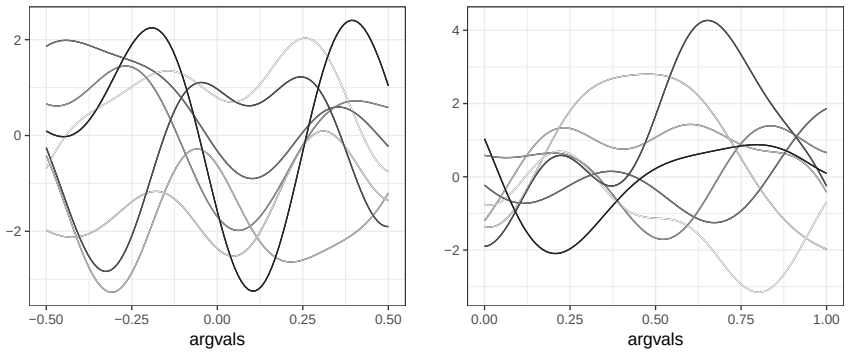


Figure C.2.: $N = 7$ simulated bivariate curves on $[-0.5, 0.5]$ and $[0, 1]$ with eigenfunctions obtained from the first $M = 10$ Fourier basis functions by the splitting algorithm (`type = "split"`) and linearly decreasing eigenvalues.

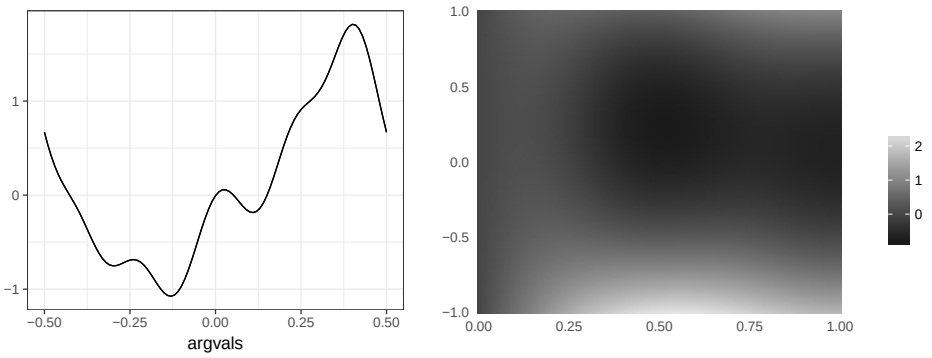


Figure C.3.: One observation of simulated bivariate data on $[-0.5, 0.5]$ and $[0, 1] \times [-1, 1]$ using weighted orthonormal elements (`type = "weighted"`). See text for details.

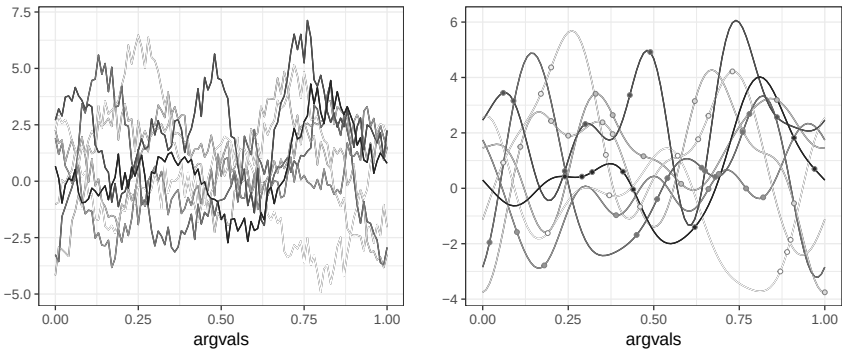


Figure C.4.: Transforming the simulated univariate functions in `simUniv1D` (see Fig. C.1). Left: Adding noise with a standard deviation of $\sigma = 0.5$. Right: The effect of sparsification, keeping five to ten observations per curve. Solid lines show the original data, filled dots correspond to the observed values of the sparsified version.

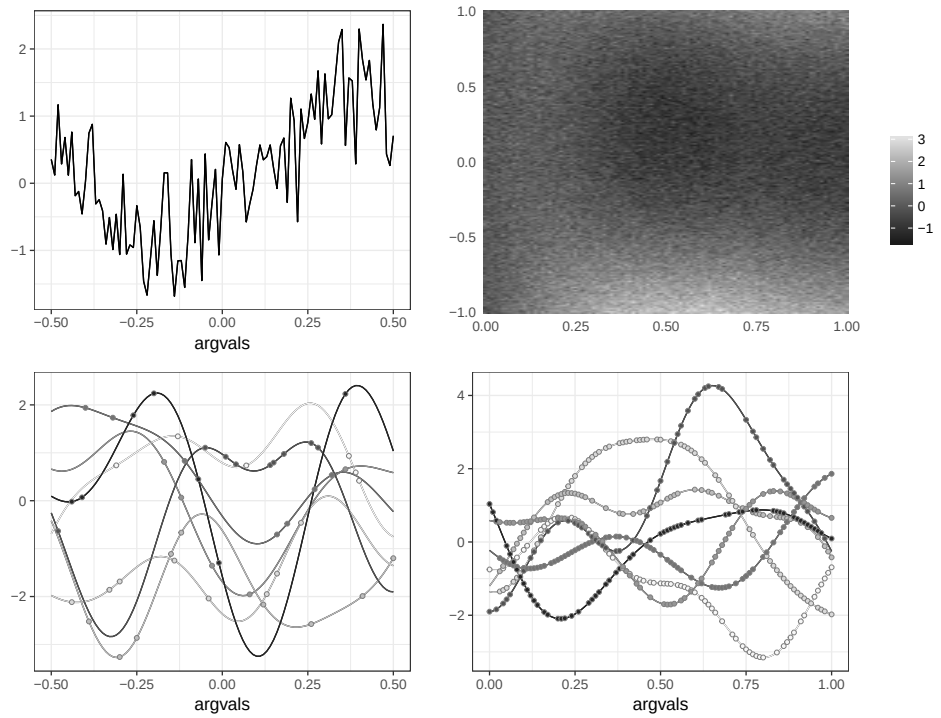


Figure C.5.: Transforming the simulated bivariate data. First row: A noisy version of the first observation of `simMultiWeight` (see Fig. C.3). Second row: All 7 observations of `simMultiSplit` after sparsification (see Fig. C.2). Solid lines show the original data, filled dots correspond to the observed values of the sparsified version. Note that the standard deviation in the noise as well as the degree of sparsification varies across elements.

D. Appendix for Chapter 6

D.1. Example Plots for Simulation

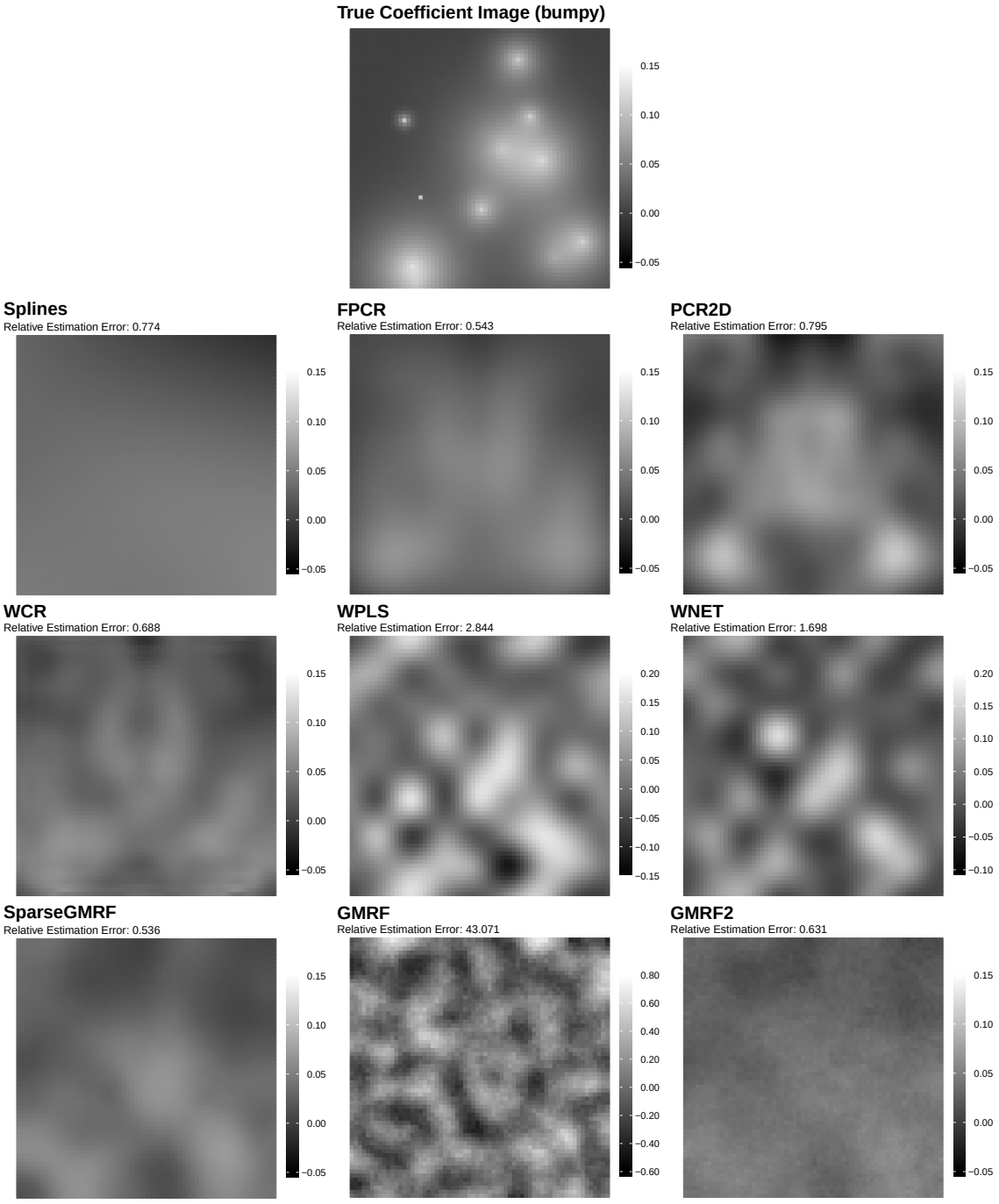


Figure D.1.: The *bumpy* coefficient image and corresponding estimates for all nine models used in the simulation study for one example iteration ($N = 250,\ \text{SNR} = 4$). Note the different scales for *WPLS*, *WNET* and *GMRF*.

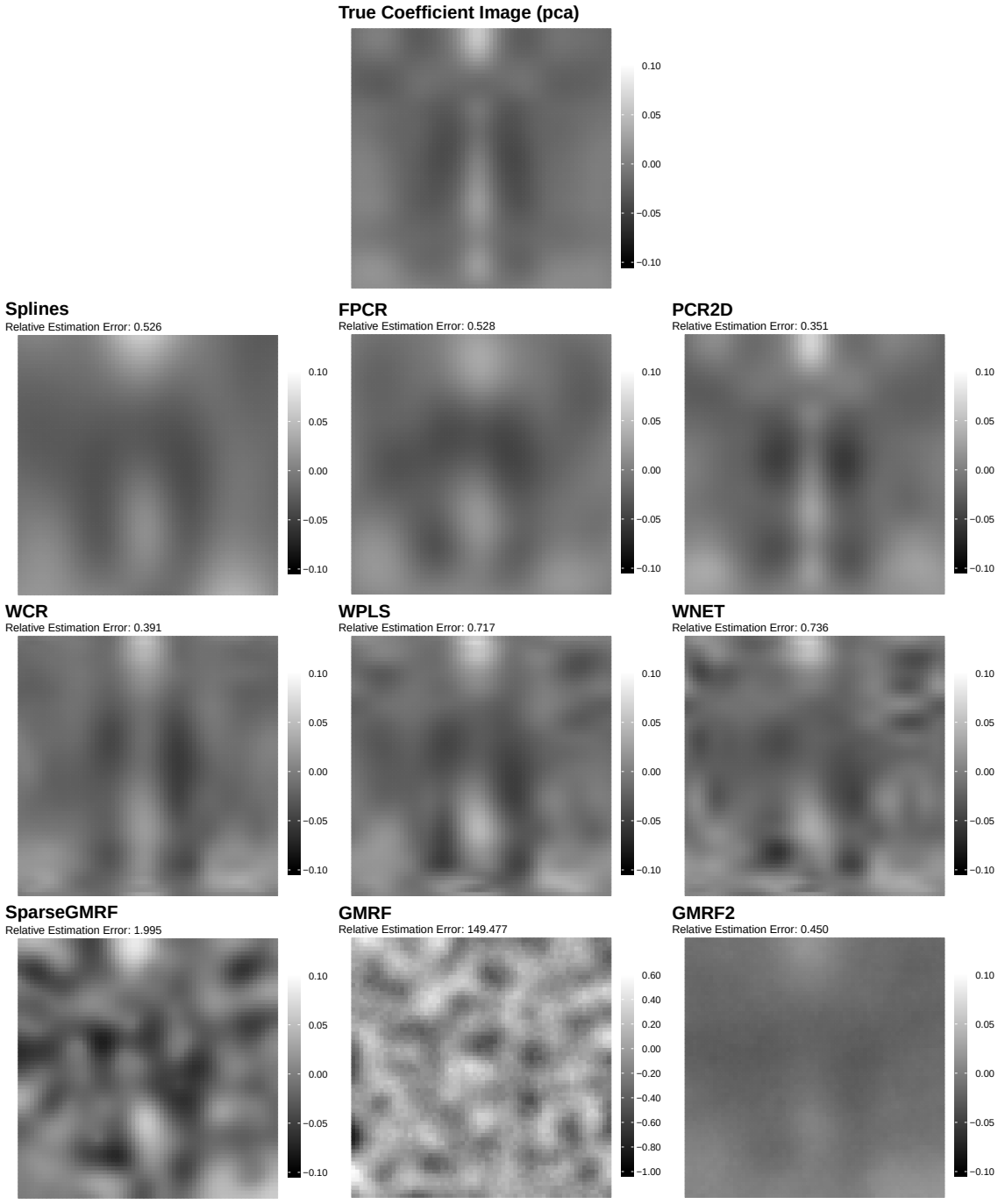


Figure D.2.: The *pca* coefficient image and corresponding estimates for all nine models used in the simulation study for one example iteration ($N = 250$, SNR $= 4$). Note the different scale for *GMRF*.

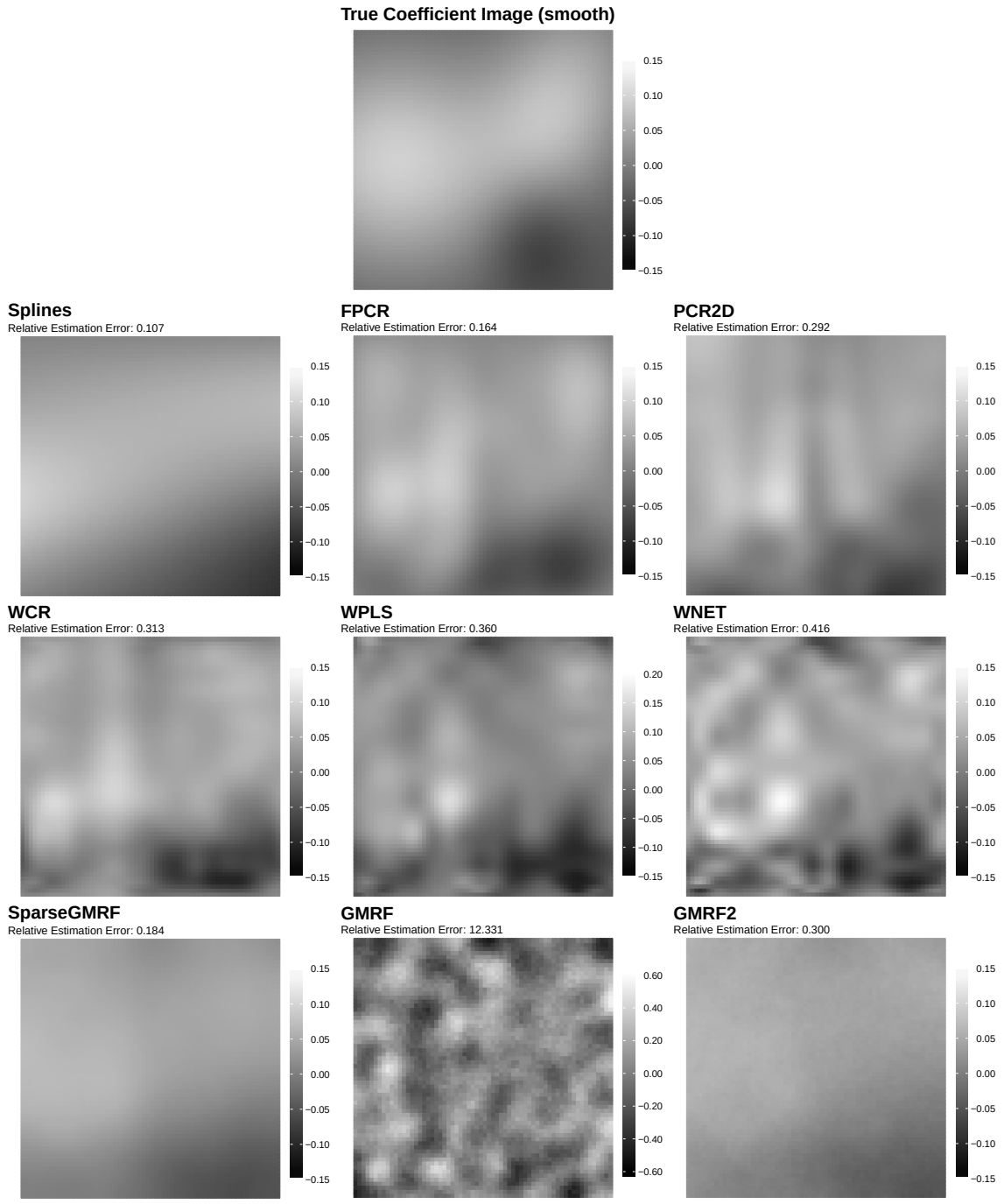


Figure D.3.: The *smooth* coefficient image and corresponding estimates for all nine models used in the simulation study for one example iteration ($N = 250,\ \text{SNR} = 4$). Note the different scales for *WPLS* and *GMRF*.

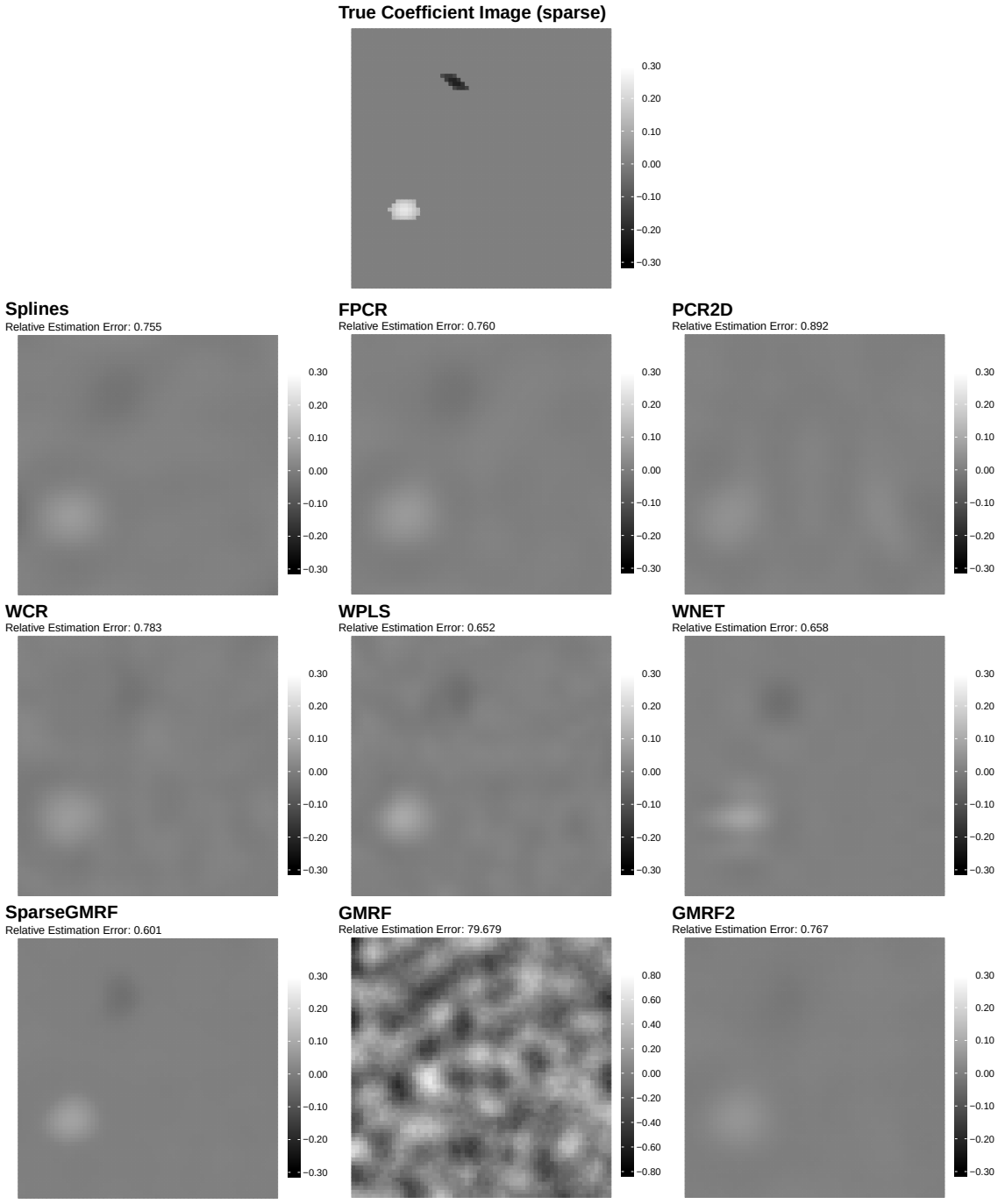


Figure D.4.: The *sparse* coefficient image and corresponding estimates for all nine models used in the simulation study for one example iteration ($N = 250,\ \text{SNR} = 4$). Note the different scale for *GMRF*.

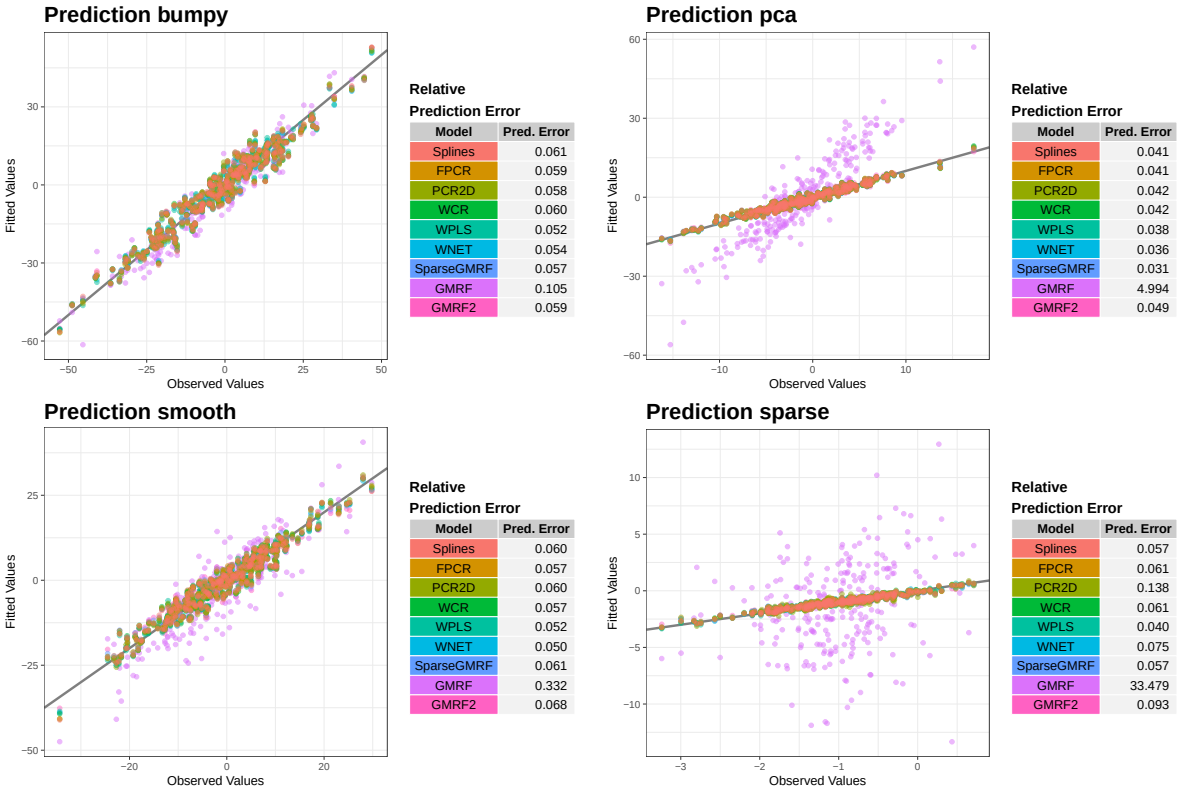


Figure D.5.: Predictions and relative prediction errors for one example iteration ($N = 250$, SNR $= 4$) in the simulation study. The plots show the observed response values y_i and the fitted values $\hat{y}_i$ for all nine models depending on the true coefficient image used. The diagonal line in each plot corresponds to a perfect fit.

D.2. Supplementary Results for the Application

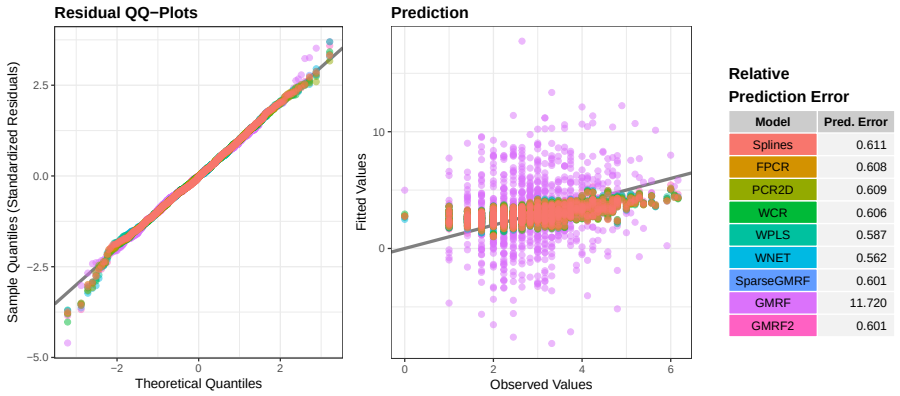


Figure D.6.: Assessing the goodness of fit for the application. Left: Normal QQ-Plots for the standardized residuals in each model, showing that they are approximately normal. Center: Observed response values y_i vs. fitted values $\hat{y}_i$ found by the nine different models. The diagonal line corresponds to a perfect fit. Right: Relative prediction errors for each model.

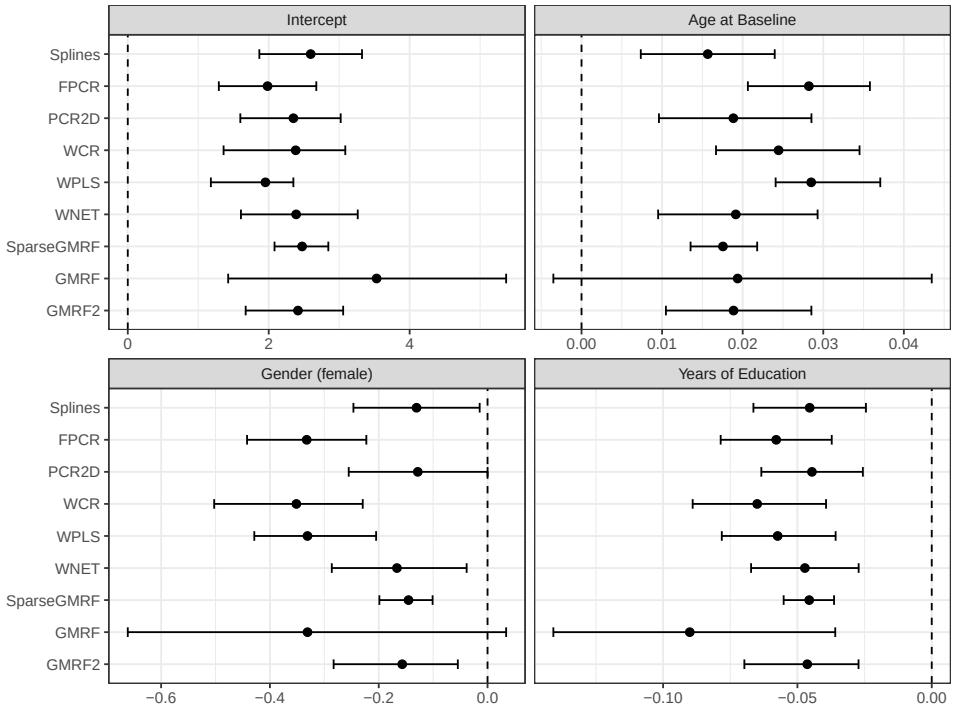


Figure D.7.: Coefficient estimates for the scalar variables in the application with empirical 95% confidence intervals. The solid point marks the coefficient estimate for each of the nine models and the horizontal lines correspond to the 95% confidence intervals. The dashed vertical line marks zero.

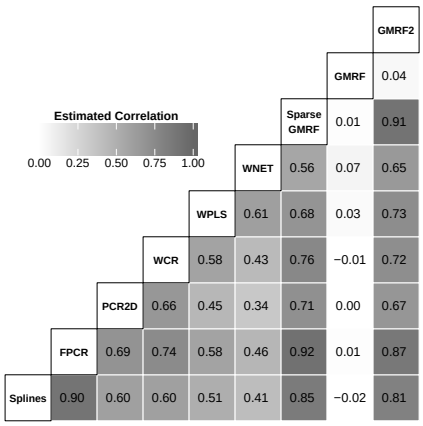


Figure D.8.: Correlation between the vectorized estimated coefficient images $\hat{\beta}$ depending on the model used.

Table D.1.: Measures for underlying and parametric model assumptions in the application.

	Underlying Assumptions			**Parametric Assumptions**						
	Smoothness	Sparsity		Smoothness		Sparsity				Prior
Model	Image	Image	Wavelets	Coef.	Pixels	Pixels	PCs	PLSCs	Wavelets	σ^2_β
Splines	0.002	-	-	0.001	-	-	-	-	-	-
FPCR	0.002	-	-	$< 10^{-3}$	-	-	0.667	-	-	-
PCR2D	-	-	-	-	-	-	0.800	-	-	-
WCR	-	-	0.146	-	-	-	0.200	-	0.244	-
WPLS	-	-	0.116	-	-	-	-	0.067	0.012	-
WNET	-	-	0.104	-	-	-	-	-	0.010	-
SparseGMRF	0.007	0.569	-	-	0.007	1.000	-	-	-	0.104
GMRF	0.043	-	-	-	0.043	-	-	-	-	0.300
GMRF2	0.010	-	-	-	0.010	-	-	-	-	0.998

D.3. Calculation of Confidence/Credible Intervals for the Application

For the application, Fig. 6.8 and Fig. D.7 show confidence or credible intervals for the estimated coefficient image $\hat{\beta}$ and the coefficient vector $\hat{\alpha}$ for the scalar covariates, which have been obtained as follows:

Splines: For $\hat{\beta}$, standard errors based on the Bayesian posterior covariance matrix of the model coefficients are calculated using the `predict.gam` function of the `mgcv` package (Wood, 2017), giving pointwise standard errors conditional on the estimated smoothing

parameters, while not including uncertainty of the intercept α (as this is considered separately). Using an approximate normality assumption, the pointwise confidence bands in a pixel $l = 1, \ldots, L$ are constructed as 95% Wald confidence intervals

$$[\hat{\beta}_l + \Phi^{-1}(0.025) \cdot \widehat{\text{se}}(\hat{\beta}_l), \hat{\beta}_l + \Phi^{-1}(0.975) \cdot \widehat{\text{se}}(\hat{\beta}_l)] \tag{D.1}$$

with Φ the distribution function of the standard normal distribution and $\widehat{\text{se}}(\hat{\beta}_l)$ the standard error for $\hat{\beta}$ in pixel l. For the $\hat{\alpha}$ coefficient, confidence intervals are constructed analogously, using the standard errors $\widehat{\text{se}}(\hat{\alpha}_j)$ produced by the `summary.gam` function from the `mgcv` package:

$$[\hat{\alpha}_j + \Phi^{-1}(0.025) \cdot \widehat{\text{se}}(\hat{\alpha}_j), \hat{\alpha}_j + \Phi^{-1}(0.975) \cdot \widehat{\text{se}}(\hat{\alpha}_j)]. \tag{D.2}$$

FPCR: Pointwise Bayesian standard errors for $\hat{\beta}$ are calculated using the `fpcr` function in `refund` (Goldsmith, Scheipl, et al., 2016). In a next step, pointwise 95% Wald confidence bands are obtained in full analogy to the *Splines* model (D.1). For $\hat{\alpha}$, we use again the `summary.gam` function from the `mgcv` package to obtain standard errors and calculate 95% Wald confidence bands based on them as in (D.2).

PCR2D: The confidence intervals for $\hat{\beta}$ and $\hat{\alpha}$ are found based on a nonparametric bootstrap approach. To this end, the data was resampled 200 times and the coefficients were re-estimated using the optimal number K of eigenimages found for the original fit due to computational reasons. Pointwise confidence bands for $\hat{\beta}$ and for the $\hat{\alpha}$ coefficients are obtained as 95% percentile bootstrap intervals.

WCR/ WPLS/ WNET: For all three wavelet-based methods, the confidence bands for $\hat{\beta}$ and $\hat{\alpha}$ are also based on a nonparametric bootstrap with 200 resampling iterations. For each bootstrap sample, the models are refit, using $M_0 = 3$ and the optimal parameters of the original fit (K^*, K_0 for *WCR* and *WPLS*; K^*, η, λ for *WNET*), similar to the case in *PCR2D*. The confidence bands are calculated as 95% percentile bootstrap intervals for both $\hat{\beta}$ and $\hat{\alpha}$ on a pointwise basis.

SparseGMRF/GMRF/GMRF2: For the Bayesian methods, we construct Bayesian 95% credible intervals for each pixel in the coefficient image $\hat{\beta}_l$ and for each coefficient $\hat{\alpha}_j$ based on the posterior drawings produced by the Gibbs sampling algorithm. The credible intervals are obtained as 2.5% and 97.5% empirical quantiles of the samples after burnin and potential thinning.

D.4. Sensitivity Study

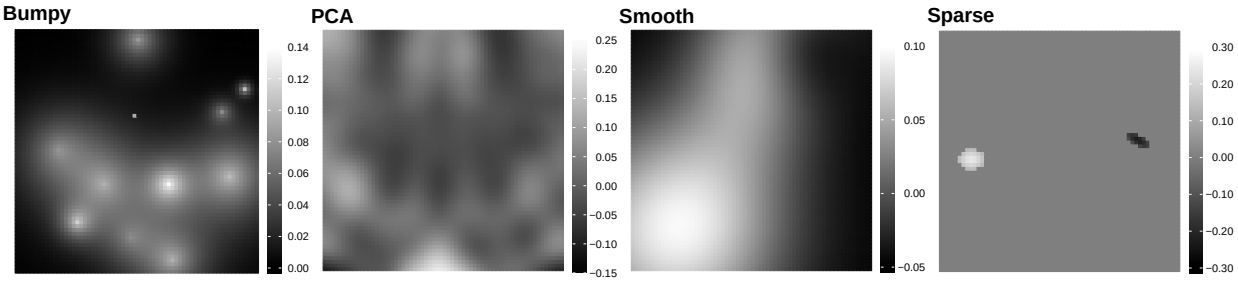


Figure D.9.: Random variations of the beta functions used for the sensitivity study. From left to right: *bumpy*, *pca*, *smooth* and *sparse*.

The results in Section 6.4.2 have been obtained for fixed coefficient images. As the covariate images x_i do not have a constant variation over all pixels, some features of β might be easier to find than others, notably if they are in areas with high variation and thus more information. In order to study the sensitivity of the results with respect to the spatial structure of β, a second study was conducted for $N = 250$ and SNR $= 4$ with spatially varying coefficient images. Therefore, a new coefficient image was generated in each iteration of the simulation, sampling the locations of the features randomly (for *bumpy, smooth* and *sparse*) or with a random number of principal components and randomly chosen coefficient b_k (for *pca*). Examples for one iteration are shown in Fig. D.9. In this study, we consider all models except for *GMRF* due to extreme error rates in the first simulation study and *SparseGMRF* due to long computations. The results are given in Fig. D.11 (error rates) and D.14 (correlations of the estimates with the true coefficient image and across models). Boxplots of the measures for underlying and parametric model assumptions are given in Figs. D.12, D.13 and D.10.

Overall, the results are very similar to the ones from the previous simulation study with fixed image covariates. This shows that variations in the features of the true coefficient images β have only marginal influence on the simulation results. Notable differences are found for the parametric model measures concerning principal components as well as in the results for the *pca* coefficient image. This is plausible, as for varying coefficient images β, different numbers of principal components might be optimal in the *FPCR*, *PCR2D* and *WCR* models. For *pca*, the higher variation can be explained by the fact that for this coefficient image, the number of eigenimages and their coefficients are resampled for generating new images β and hence may lead to a higher variation.

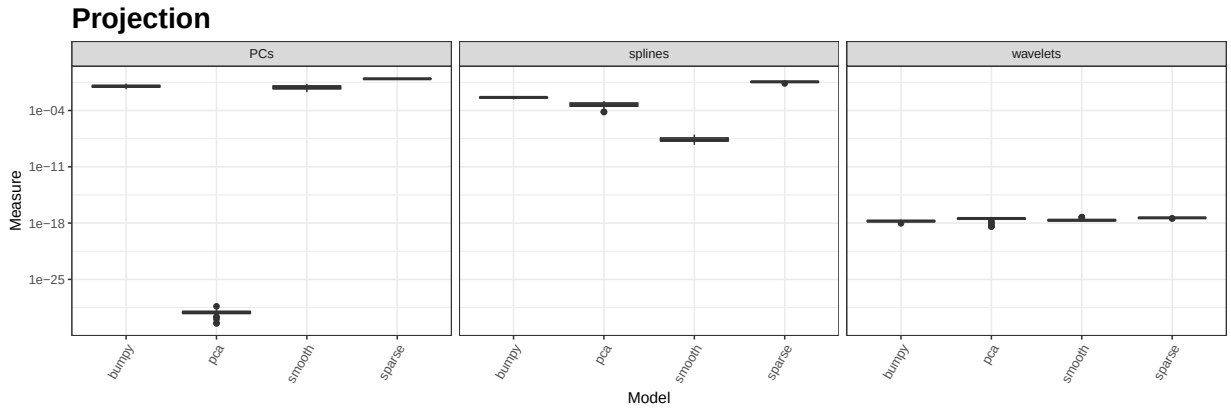


Figure D.10.: Values of $m_{\text{Projection}}$ in the sensitivity study for the different coefficient images depending on the basis functions used.

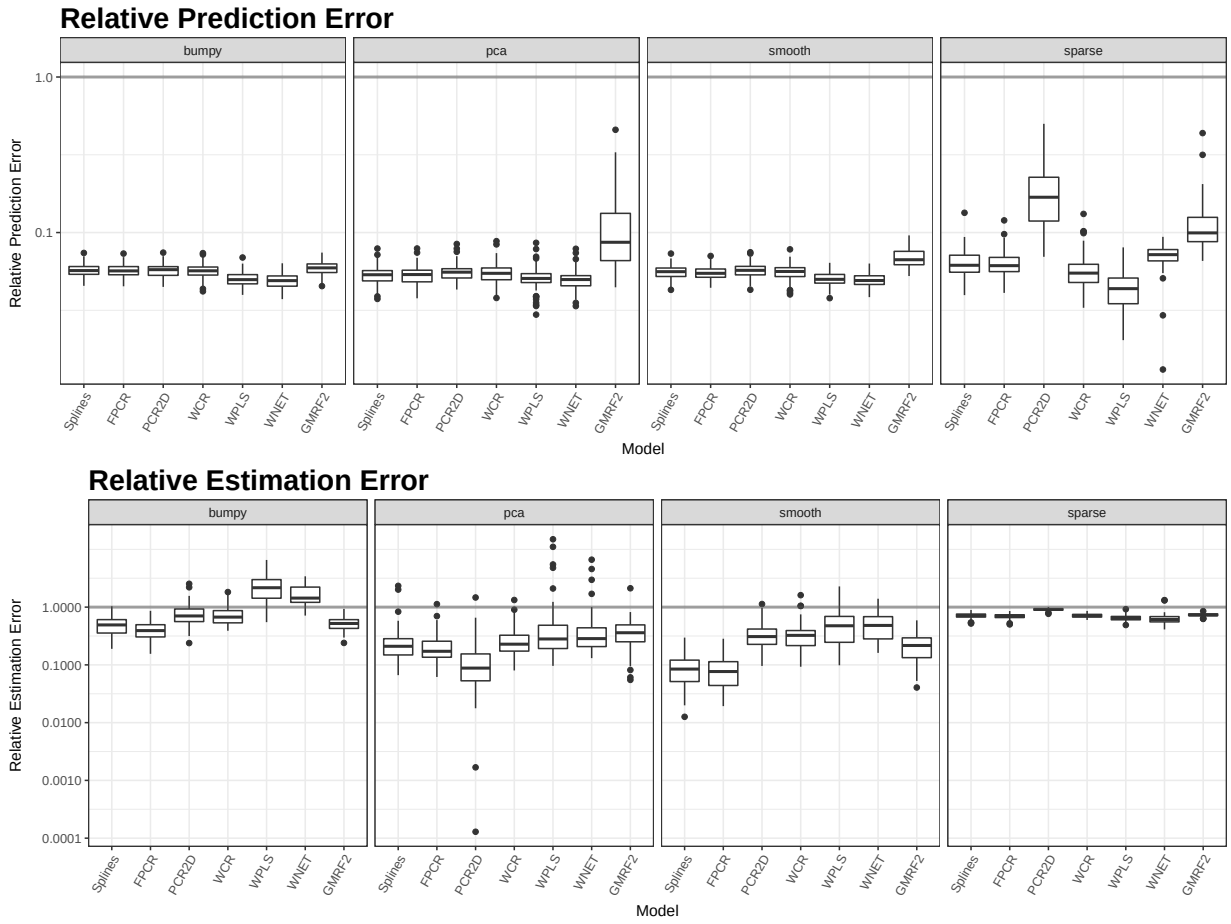


Figure D.11.: Results of the sensitivity study. Boxplots show the relative prediction and estimation error for all nine models depending on the coefficient image and the signal-to-noise ratio (SNR) over all 100 simulation runs. Gray horizontal lines mark 1, which corresponds to the simple intercept model (for prediction error) or to a constant coefficient image, having the average value of the true β image (estimation error).

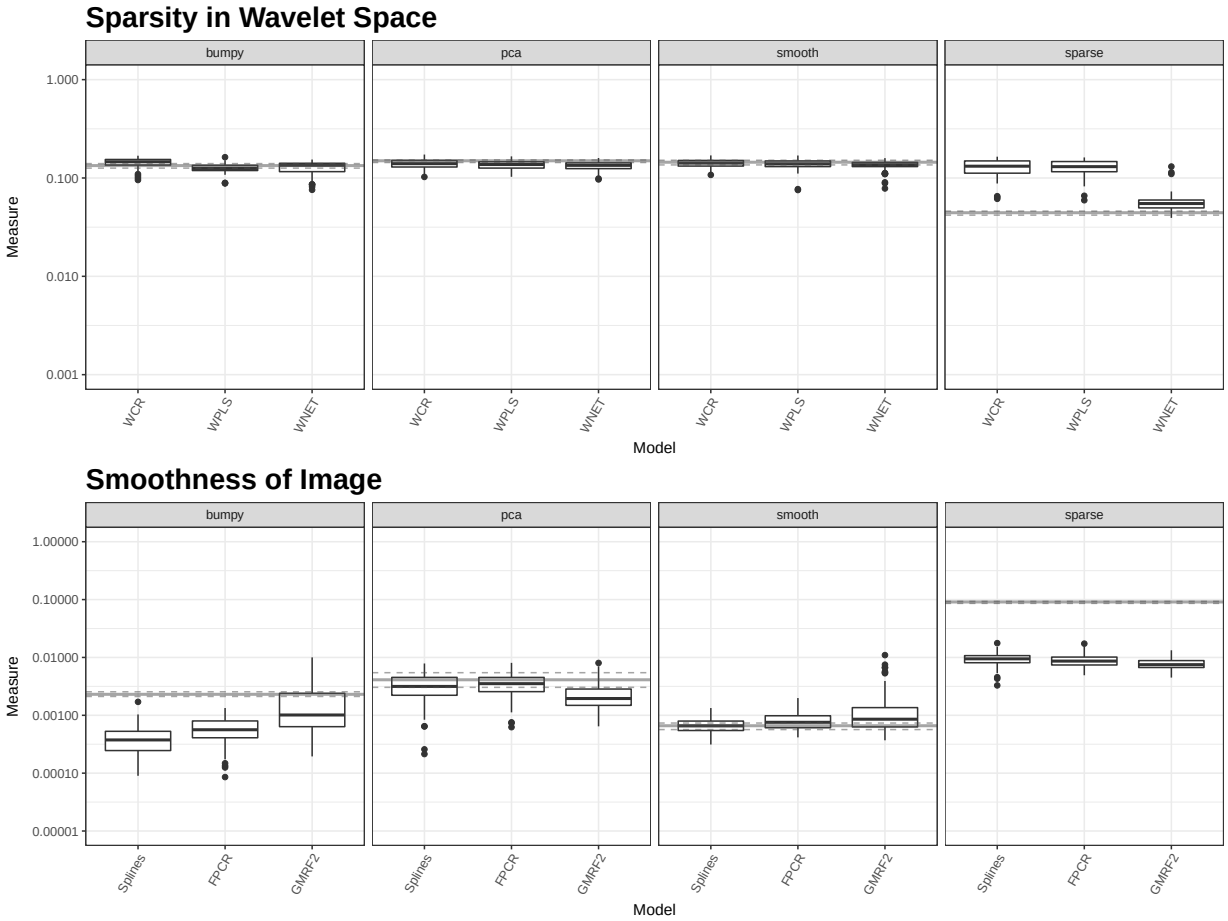


Figure D.12.: Measures for underlying model assumptions in the sensitivity study. Boxplots show the measures for the different models depending on the true coefficient image over all 100 simulation runs. All values on log-scale. Gray horizontal lines correspond to the median (solid line) and the 25% and 75% quantiles (dashed lines) for the true coefficient images.

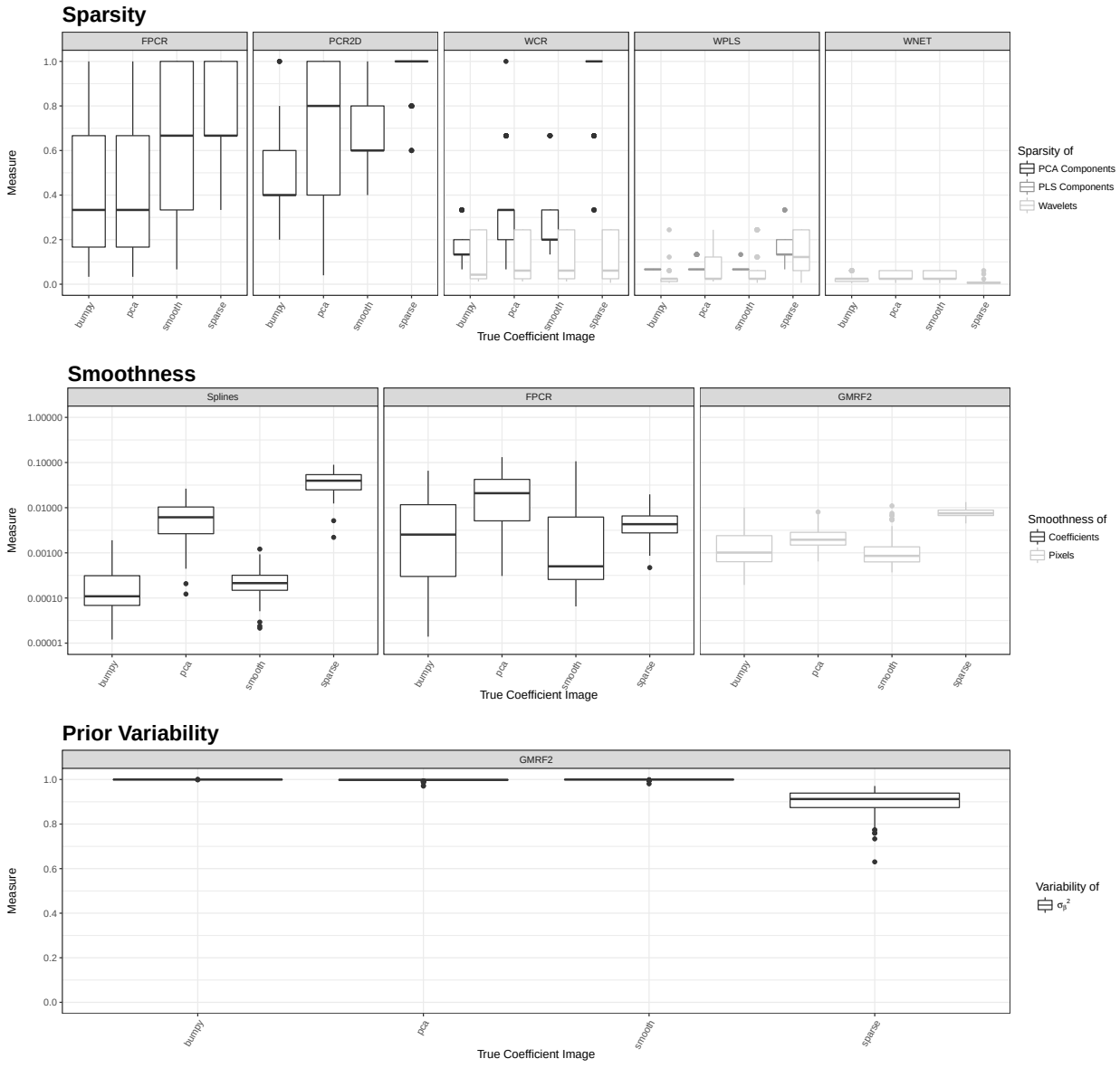


Figure D.13.: Measures for parametric model assumptions in the sensitivity study. Boxplots show the measures for the different coefficient images depending on the model used over all 100 simulation runs. Measures with extremely low values on log-scale.

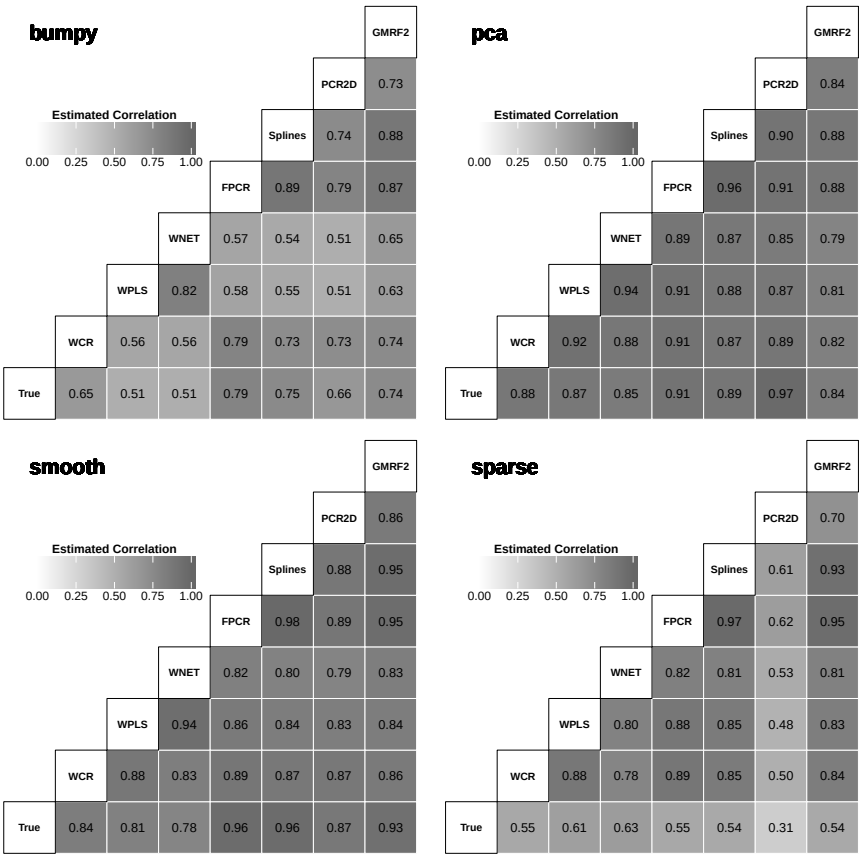


Figure D.14.: Correlation between the true coefficient images and the estimates found by the different models in the sensitivity study. The figures show the median correlation of the vectorized images over 100 simulation runs depending on the true images and the models used.

D.5. Simulation Results for 500 Individuals

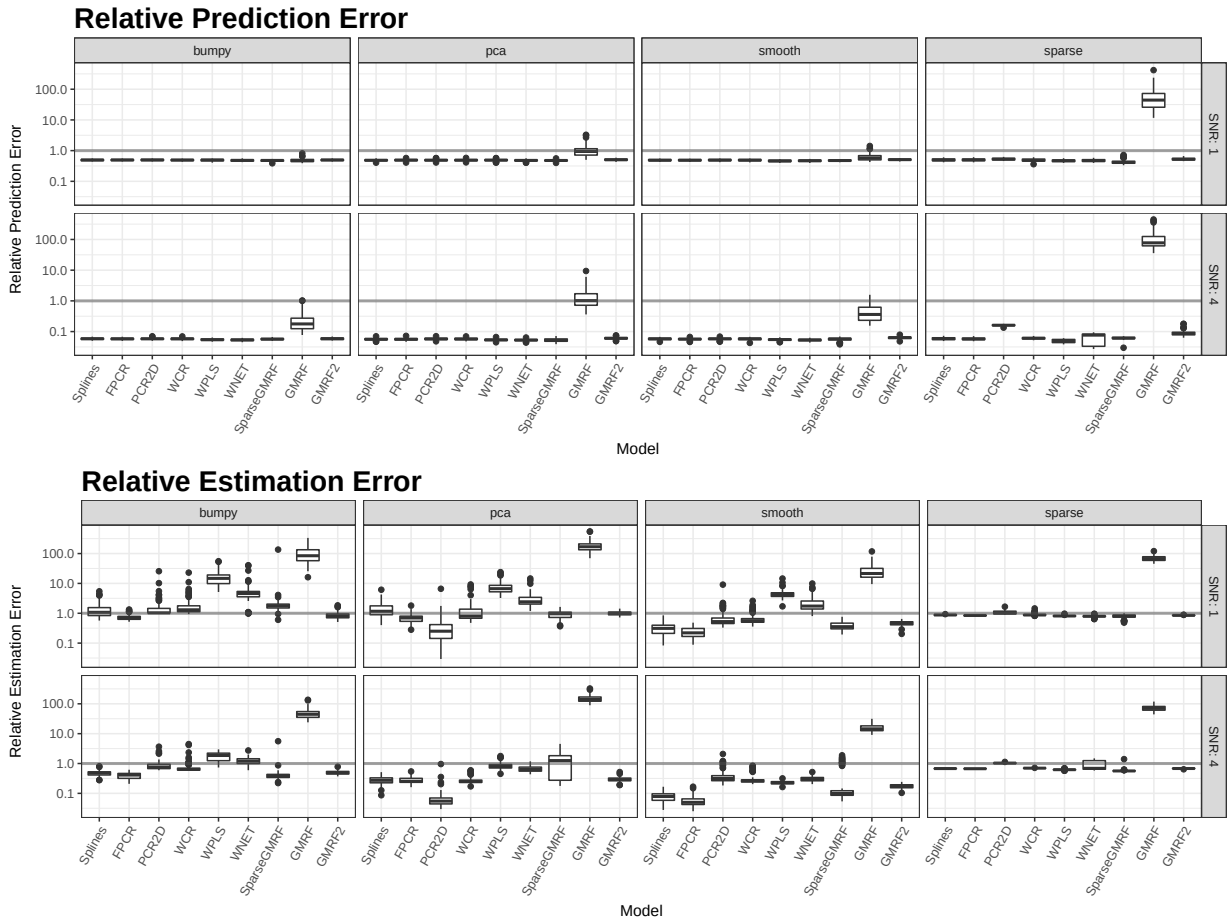


Figure D.15.: Simulation results for $N = 500$ observations. Boxplots show the relative prediction and estimation error for all nine models depending on the coefficient image and the signal-to-noise ratio (SNR) over all 100 simulation runs. Gray horizontal lines mark 1, which corresponds to the simple intercept model (for prediction error) or to a constant coefficient image, having the average value of the true β image (estimation error).

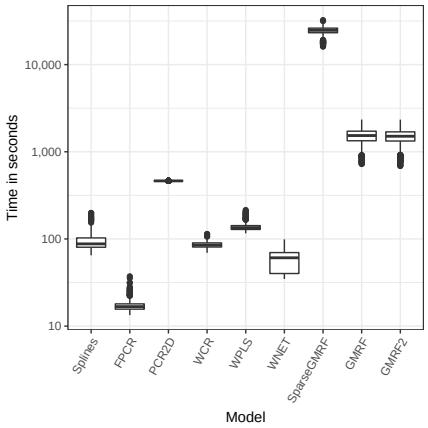


Figure D.16.: Computation times for all nine models and $N = 500$ observations over all 100 simulation runs. The boxplots contain the merged values for all coefficient images and signal-to-noise ratios.

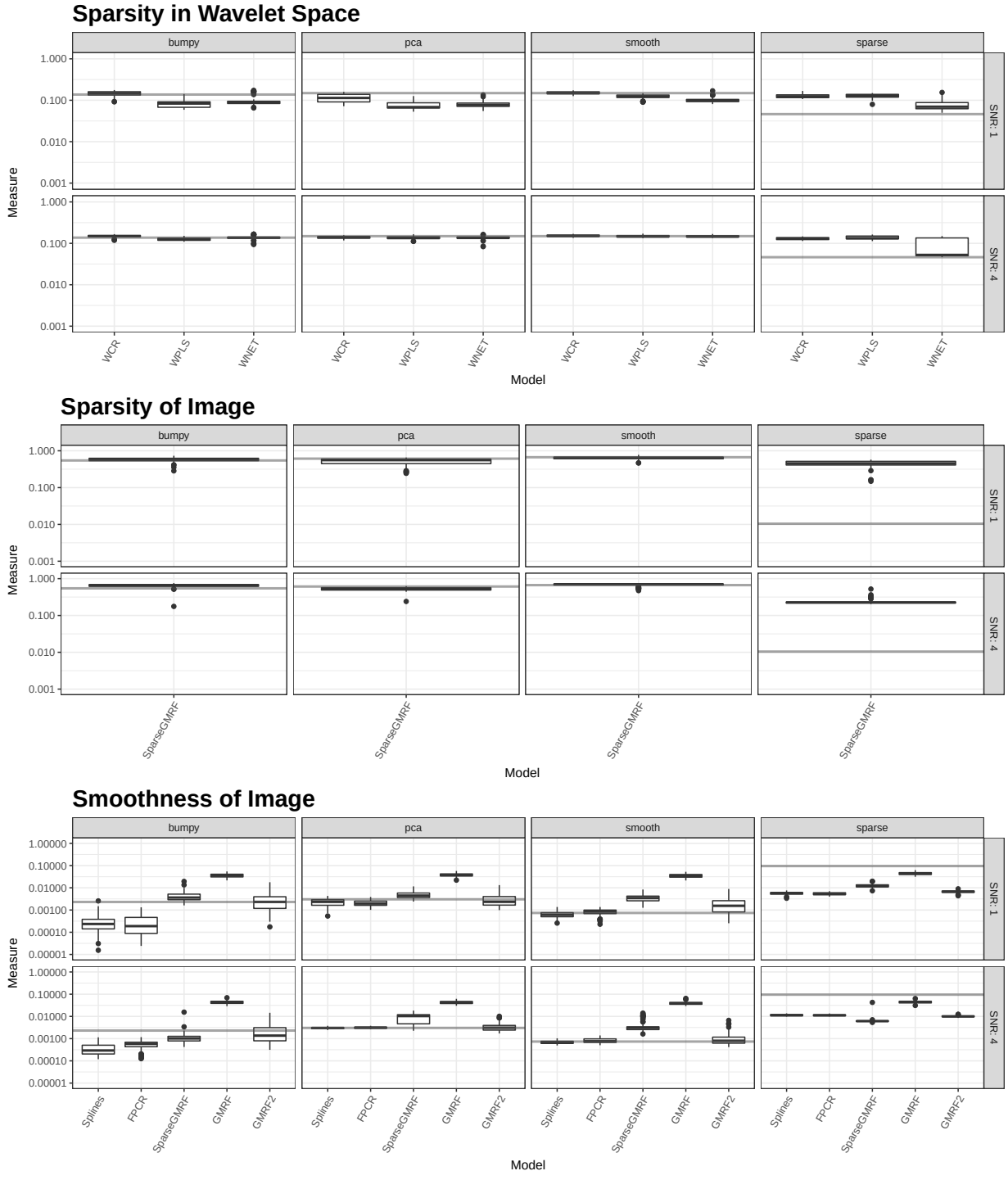


Figure D.17.: Measures for underlying model assumptions in the simulation for $N = 250$ observations. Boxplots show the measures for the different models depending on the true coefficient image and the signal-to-noise ratio (SNR) over all 100 simulation runs. All values on log-scale. Gray horizontal lines correspond to the values for the true coefficient images.

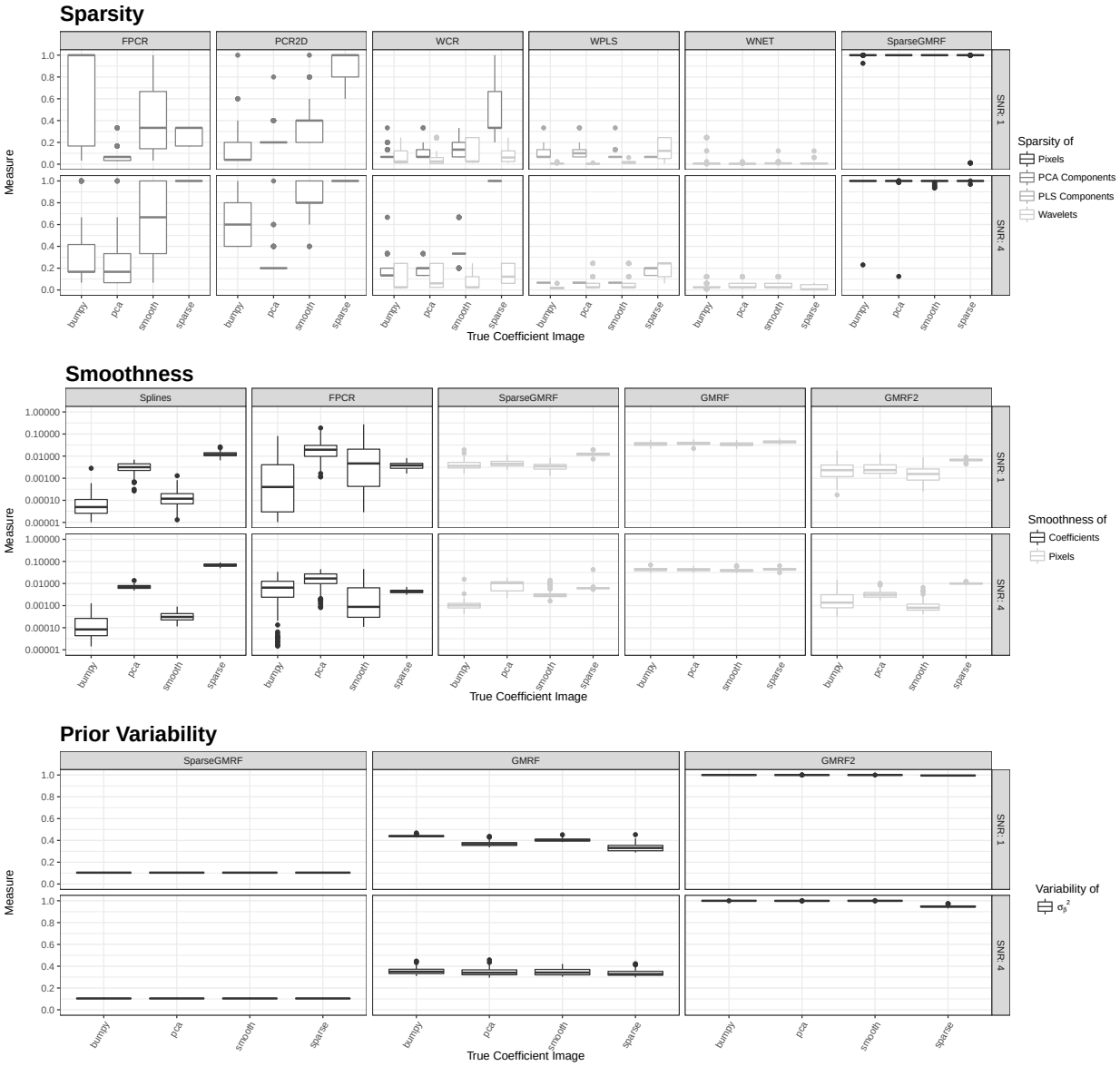


Figure D.18.: Measures for parametric model assumptions in the simulation for $N = 500$ observations. Boxplots show the measures for the different coefficient images depending on the model used and the signal-to-noise ratio (SNR) over all 100 simulation runs. Measures with extremely low values on log-scale.

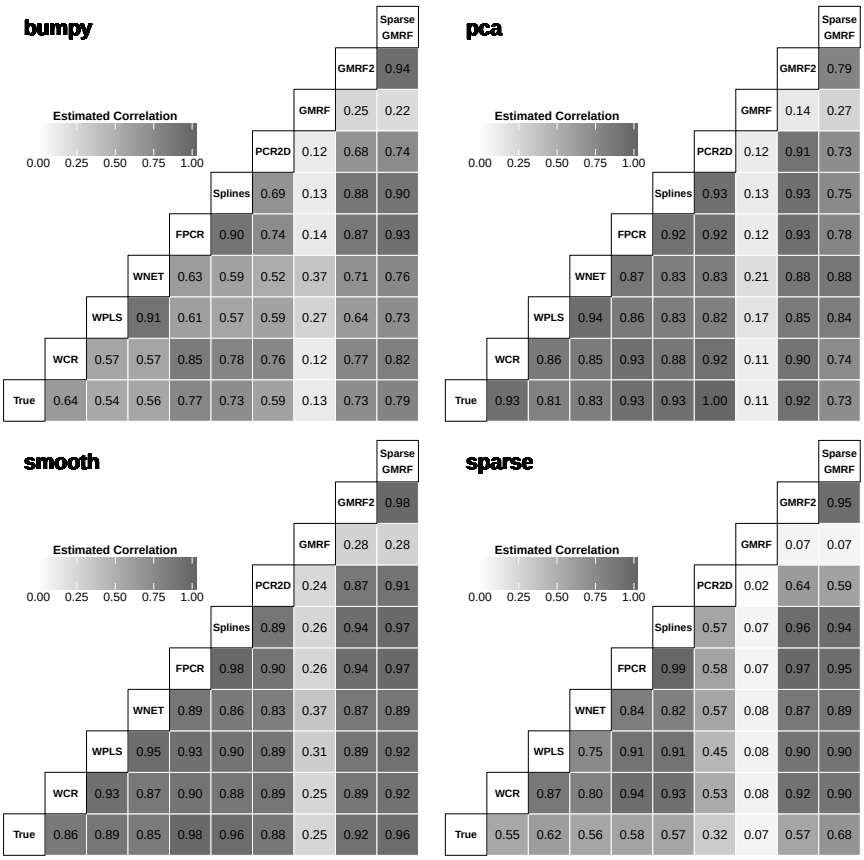


Figure D.19.: Correlation between the true coefficient images and the estimates found by the different models in the simulation study with $N = 500$ observations and SNR $= 4$. The figures show the median correlation of the vectorized images over 100 simulation runs depending on the true images and the models used.

Bibliography

Abraham, C., P. Cornillon, E. Matzner-Løber & N. Molinari (2003). "Unsupervised Curve Clustering using B–Splines". *Scandinavian Journal of Statistics* 30, pp. 581–595.

Allen, G. I. (2013). "Multi-way Functional Principal Components Analysis". *IEEE 5th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pp. 220–223.

Araque Caballero, M. Á., M. Brendel, A. Delker, J. Ren, A. Rominger, P. Bartenstein, M. Dichgans, M. W. Weiner & M. Ewers (2015). "Mapping 3-year changes in gray matter and metabolism in Aβ-positive nondemented subjects". *Neurobiology of Aging* 36.11, pp. 2913–2924.

Armstrong, D. J. (2006). "The Quarks of Object-oriented Development". *Commun. ACM* 49.2, pp. 123–128.

Ashburner, J. (2007). "A fast diffeomorphic image registration algorithm". *NeuroImage* 38.1, pp. 95–113.

Auguie, B. (2016). *gridExtra: Miscellaneous Functions for "Grid" Graphics*. R package version 2.2.1. URL: `https://CRAN.R-project.org/package=gridExtra`.

Baglama, J. & L. Reichel (2005). "Augmented Implicitly Restarted Lanczos Bidiagonalization Methods". *SIAM Journal on Scientific Computing* 27.1, pp. 19–42.

Baglama, J., L. Reichel & B. W. Lewis (2017). *irlba: Fast Truncated SVD, PCA and Symmetric Eigendecomposition for Large Dense and Sparse Matrices*. R package version 2.2.1. URL: `https://CRAN.R-project.org/package=irlba`.

Bates, D. & M. Maechler (2017). *Matrix: Sparse and Dense Matrix Classes and Methods*. R package version 1.2-10. URL: `https://CRAN.R-project.org/package=Matrix`.

Bayes, T. (1763). "An Essay towards Solving a Problem in the Doctrine of Chances." *Philosophical Transactions of the Royal Society of London* 53, pp. 370–418.

Berrendero, J., A. Justel & M. Svarc (2011). "Principal components for multivariate functional data". *Computational Statistics & Data Analysis* 55.9, pp. 2619–2634.

Besag, J. (1974). "Spatial interaction and the statistical analysis of lattice systems". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 36.2, pp. 192–236.

Besag, J., J. York & A. Mollié (1991). "Bayesian image restoration, with two applications in spatial statistics". *Annals of the Institute of Statistical Mathematics* 43.1, pp. 1–20.

Blennow, K., M. Leon & H. Zetterberg (2006). "Alzheimer's disease". *Lancet* 368, pp. 387–403.

Booch, G., R. A. Maksimchuk, M. W. Engle, B. J. Young & K. A. H. Jim Conallen (2007). *Object-Oriented Analysis and Design with Applications*. 3rd ed. Upper Saddle River, NJ: Addison-Wesley.

Bosq, D. (2000). *Linear Processes in Function Spaces*. New York: Springer.

Bouveyron, C. & J. Jacques (2014). *funHDDC: Model-based clustering in group-specific functional subspaces*. R package version 1.0. URL: https://CRAN.R-project.org/package=funHDDC.

Bouveyron, C. (2015). *funFEM: Clustering in the Discriminative Functional Subspace*. R package version 1.1. URL: https://CRAN.R-project.org/package=funFEM.

Brezger, A., L. Fahrmeir & A. Hennerfeind (2007). "Adaptive Gaussian Markov random fields with applications in human brain mapping". *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 56.3, pp. 327–345.

Brockhaus, S. & D. Ruegamer (2017). *FDboost: Boosting Functional Regression Models*. R package version 0.3-0. URL: https://CRAN.R-project.org/package=FDboost.

Cardot, H., F. Ferraty & P. Sarda (1999). "Functional linear model". *Statistics & Probability Letters* 45.1, pp. 11–22.

– (2003). "Spline estimators for the functional linear model". *Statistica Sinica* 13, pp. 571–591.

Carlin, B. P. & T. A. Louis (2009). *Bayesian methods for data analysis*. 3rd ed. Boca Raton: Chapman & Hall/CRC.

Carroll, J. D. & J. J. Chang (1970). "Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition". *Psychometrika* 35.3, pp. 283–319.

Cattell, R. (1966). "The Scree Test For The Number Of Factors". *Multivariate Behavioral Research* 1.2, pp. 245–276.

Cederbaum, J. (2017). *sparseFLMM: Functional Linear Mixed Models for Irregularly or Sparsely Sampled Data*. R package version 0.1.1. URL: https://CRAN.R-project.org/package=sparseFLMM.

Chambers, J. M. (2008). *Software for Data Analysis*. New York: Springer.

Chételat, G., R. Ossenkoppele, V. L. Villemagne, A. Perrotin, B. Landeau, F. Mézenge, W. J. Jagust, V. Dore, B. L. Miller, S. Egret, W. W. Seeley, W. M. van der Flier, R. La Joie, D. Ames, B. N. M. van Berckel, P. Scheltens, F. Barkhof, C. C. Rowe, C. L. Masters, V. de La Sayette, F. Bouwman & G. D. Rabinovici (2016). "Atrophy, hypometabolism and clinical trajectories in patients with amyloid-negative Alzheimer's disease". *Brain* 139.9, pp. 2528–2539.

Chiou, J.-M. & H.-G. Müller (2014). "Linear manifold modelling of multivariate functional data". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 76.3, pp. 605–626.

Chiou, J.-M., Y.-F. Yang & Y.-T. Chen (2014). "Multivariate functional principal component analysis: A normalization approach". *Statistica Sinica* 24, pp. 1571–1596.

Coombs, C. H. (1964). *A theory of data*. New York: Wiley.

Coppi, R. & S. Bolasco, eds. (1989). *Multiway data analysis*. Amsterdam: North-Holland.

Dai, X., P. Z. Hadjipantelis, H. Ji, H.-G. Mueller & J.-L. Wang (2017). *fdapace: Functional Data Analysis and Empirical Dynamics*. R package version 0.3.0. URL: https://CRAN.R-project.org/package=fdapace.

Dalton, H. (1920). "The Measurement of the Inequality of Incomes". *The Economic Journal* 30.119, pp. 348–361.

Daubechies, I. (1988). "Orthonormal bases of compactly supported bases". *Communications On Pure and Applied Mathematics* 41, pp. 909–996.

De Boor, C. (1972). "On calculating with B-splines". *Journal of Approximation Theory* 6.1, pp. 50–62.

Di, C.-Z., C. M. Crainiceanu, B. S. Caffo & N. M. Punjabi (2009). "Multilevel functional principal component analysis". *Annals of Applied Statistics* 3.1, pp. 458–488.

Donoho, D. L. & J. M. Johnstone (1994). "Ideal spatial adaptation by wavelet shrinkage". *Biometrika* 81.3, pp. 425–455.

Eilers, P. & B. Marx (1996). "Flexible Smoothing with B-splines and Penalties". *Statistical Science* 11.2, pp. 89–121.

Evans, M. & H. Moshonov (2006). "Checking for prior-data conflict". *Bayesian Analysis* 1.4, pp. 893–914.

Fan, J. & I. Gijbels (1997). *Local polynomial modelling and its applications*. Reprint. London: Chapman & Hall.

Febrero-Bande, M. & M. Oviedo de la Fuente (2012). "Statistical Computing in Functional Data Analysis: The R Package fda.usc". *Journal of Statistical Software* 51.4, pp. 1–28.

Ferraty, F. & P. Vieu (2006). *Nonparametric Functional Data Analysis*. New York: Springer.

Fisher, R. A. (1922). "On the Mathematical Foundations of Theoretical Statistics". *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 222, pp. 309–368.

Friedman, J., T. Hastie & R. Tibshirani (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent". *Journal of Statistical Software* 33.1, pp. 1–22.

Frigo, M. & S. Johnson (2005). "The Design and Implementation of FFTW3". *Proceedings of the IEEE* 93.2, pp. 216–231.

Friston, K. J., J. T. Ashburner, S. J. Kiebel, T. E. Nichols & W. D. Penny, eds. (2007). *Statistical parametric mapping: The analysis of functional brain images*. Academic press.

Gelfand, A. E. & A. F. M. Smith (1990). "Sampling-Based Approaches to Calculating Marginal Densities". *Journal of the American Statistical Association* 85.410, pp. 398–409.

Gelman, A. (2006). "Prior distributions for variance parameters in hierarchical models (Comment on Article by Browne and Draper)". *Bayesian Analysis* 1.3, pp. 515–534.

Gelman, A., J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari & D. B. Rubin (2014). *Bayesian Data Analysis*. 3rd ed. Boca Raton: Chapman & Hall/CRC.

Gelman, A. & K. Shirley (2011). "Inference from Simulations and Monitoring Convergence". *Handbook of Markov Chain Monte Carlo*. Ed. by S. Brooks, A. Gelman, G. L. Jones & X.-L. Meng. Boca Raton: Chapman and Hall/CRC. Chap. 6, pp. 163–174.

Geman, S. & D. Geman (1984). "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images". *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.6, pp. 721–741.

Geyer, C. J. (2011). "Introduction to Markov Chain Monte Carlo". *Handbook of Markov Chain Monte Carlo*. Ed. by S. Brooks, A. Gelman, G. L. Jones & X.-L. Meng. Boca Raton: Chapman and Hall/CRC. Chap. 1, pp. 3–48.

Gilks, W. R., S. Richardson & D. J. Spiegelhalter (1996). "Introducing Markov chain Monte Carlo". *Markov chain Monte Carlo in practice*. Ed. by W. R. Gilks, S. Richardson & D. J. Spiegelhalter. Boca Raton: Chapman and Hall/CRC. Chap. 1, pp. 1–19.

Goldsmith, J., S. Greven & C. Crainiceanu (2013). "Corrected Confidence Bands for Functional Data Using Principal Components". *Biometrics* 69.1, pp. 41–51.

Goldsmith, J., J. Bobb, C. M. Crainiceanu, B. Caffo & D. Reich (2011). "Penalized Functional Regression". *Journal of Computational and Graphical Statistics* 20.4, pp. 830–851.

Goldsmith, J., L. Huang & C. M. Crainiceanu (2014). "Smooth Scalar-on-Image Regression via Spatial Bayesian Variable Selection". *Journal of Computational and Graphical Statistics* 23.1, pp. 46–64.

Goldsmith, J. & F. Scheipl (2014). "Estimator selection and combination in scalar-on-function regression". *Computational Statistics & Data Analysis* 70, pp. 362–372.

Goldsmith, J., F. Scheipl, L. Huang, J. Wrobel, J. Gellar, J. Harezlak, M. W. McLean, B. Swihart, L. Xiao, C. Crainiceanu & P. T. Reiss (2016). *refund: Regression with Functional Data*. R package version 0.1-16. URL: `https://CRAN.R-project.org/package=refund`.

Gregorutti, B. (2016). *RFgroove: Importance Measure and Selection for Groups of Variables with Random Forests*. R package version 1.1. URL: `https://CRAN.R-project.org/package=RFgroove`.

Greven, S. & F. Scheipl (2017). "A general framework for functional regression modelling". *Statistical Modelling* 17.1-2, pp. 1–35.

Hall, P. & J. L. Horowitz (2007). "Methodology and convergence rates for functional linear regression". *Annals of Statistics* 35.1, pp. 70–91.

Hall, P. & M. Hosseini-Nasab (2006). "On properties of functional principal components analysis". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.1, pp. 109–126.

Happ, C. (2013). "Identifiability in Scalar-on-Functions Regression". MA thesis. LMU Munich.

– (2017a). *funData: An S4 Class for Functional Data*. R package version 1.1.

– (2017b). *MFPCA: Multivariate Functional Principal Component Analysis for Data Observed on Different Dimensional Domains*. R package version 1.1.

Happ, C. & S. Greven (2017+). "Multivariate Functional Principal Component Analysis for Data Observed on Different (Dimensional) Domains". *Journal of the American Statistical Association*. To appear. DOI: `10.1080/01621459.2016.1273115`.

Hastie, T. & R. Tibshirani (1986). "Generalized Additive Models". *Statistical Science* 1.3, pp. 297–310.

Hastings, W. K. (1970). "Monte Carlo Sampling Methods Using Markov Chains and Their Applications". *Biometrika* 57.1, pp. 97–109.

Horn, R. A. & C. R. Johnson (1991). *Topics in matrix analysis*. Cambridge: Cambridge University Press.

Horn, R. A. & C. R. Johnson (1985). *Matrix Analysis*. Cambridge: Cambridge University Press.

Horváth, L. & P. Kokoszka (2012). *Inference for Functional Data with Applications*. New York: Springer.

Hsing, T. & R. Eubank (2015). *Theoretical Foundations of Functional Data Analysis, with an Introduction to Linear Operators*. Chichester: Wiley.

Huang, J. Z., H. Shen & A. Buja (2009). "The Analysis of Two-Way Functional Data Using Two-Way Regularized Singular Value Decompositions". *Journal of the American Statistical Association* 104.488, pp. 1609–1620.

Huang, L., J. Goldsmith, P. T. Reiss, D. S. Reich & C. M. Crainiceanu (2013). "Bayesian scalar-on-image regression with application to association between intracranial DTI and cognitive outcomes." *NeuroImage* 83, pp. 210–23.

Huo, L., P. Reiss & Y. Zhao (2014). *refund.wave: Wavelet-Domain Regression with Functional Data*. R package version 0.1. URL: `https://CRAN.R-project.org/package=refund.wave`.

Hurley, N. & S. Rickard (2009). "Comparing Measures of Sparsity". *IEEE Transactions on Information Theory* 55.10, pp. 4723–4741.

Itti, L. & P. F. Baldi (2005). "A Principled Approach to Detecting Surprising Events in Video". *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. San Diego, CA, pp. 631–637.

Jacques, J. & C. Preda (2013). "Funclust: A curves clustering method using functional random variables density approximation". *Neurocomputing* 112, pp. 164–171.

– (2014). "Model-based clustering for multivariate functional data". *Computational Statistics & Data Analysis* 71, pp. 92–106.

Jagust, W. J., D. Bandy, K. Chen, N. L. Foster, S. M. Landau, C. A. Mathis, J. C. Price, E. M. Reiman, D. Skovronsky & R. A. Koeppe (2010). "The Alzheimer's Disease Neuroimaging Initiative positron emission tomography core". *Alzheimer's & Dementia : The Journal of the Alzheimer's Association* 6.3, pp. 221–229.

James, G. M. & B. W. Silverman (2005). "Functional Adaptive Model Estimation". *Journal of the American Statistical Association* 100.470, pp. 565–576.

James, G. M. & C. A. Sugar (2003). "Clustering for Sparsely Sampled Functional Data". *Journal of the American Statistical Association* 98.462, pp. 397–408.

Johnstone, I. M. & A. Y. Lu (2009). "On Consistency and Sparsity for Principal Components Analysis in High Dimensions". *Journal of the American Statistical Association* 104.486, pp. 682–693.

Kang, J., B. J. Reich & A.-M. Staicu (2016). "Scalar-on-Image Regression via the Soft-Thresholded Gaussian Process". arXiv: `1604.03192`.

Karhunen, K. (1947). "Über lineare Methoden in der Wahrscheinlichkeitsrechnung". *Annales Academiae Scientiarum Fennicae. Series A. 1, Mathematica-Physica* 37, pp. 1–79.

Kato, T. (1976). *Perturbation Theory for Linear Operators*. 2nd ed. Berlin: Springer.

La Joie, R., A. Perrotin, L. Barré, C. Hommet, F. Mézenge, M. Ibazizene, V. Camus, A. Abbas, B. Landeau, D. Guilloteau, V. de La Sayette, F. Eustache, B. Desgranges & G. Chételat (2012). "Region-Specific Hierarchy between Atrophy, Hypometabolism, and β-Amyloid ($A\beta$) Load in Alzheimer's Disease Dementia". *Journal of Neuroscience* 32.46, pp. 16265–16273.

Landau, S. M., C. Breault, A. D. Joshi, M. Pontecorvo, C. A. Mathis, W. J. Jagust & M. A. Mintun (2013). "Amyloid-β Imaging with Pittsburgh Compound B and Florbetapir: Comparing Radiotracers and Quantification Methods". *Journal of Nuclear Medicine* 54.1, pp. 70–77.

Landau, S. M., A. Horng, A. Fero & W. J. Jagust (2016). "Amyloid negativity in patients with clinically diagnosed Alzheimer disease and MCI". *Neurology* 86.15, pp. 1377–1385.

Li, F., T. Zhang, Q. Wang, M. Z. Gonzalez, E. L. Maresh & J. A. Coan (2015). "Spatial Bayesian variable selection and grouping for high-dimensional scalar-on-image regression". *Annals of Applied Statistics* 9.2, pp. 687–713.

Lila, E., L. M. Sangalli, J. Ramsay & L. Formaggia (2016). *fdaPDE: Functional Data Analysis and Partial Differential Equations; Statistical Analysis of Functional and Spatial Data, Based on Regression with Partial Differential Regularizations*. R package version 0.1-4. URL: `https://CRAN.R-project.org/package=fdaPDE`.

Loève, M. (1946). "Fonctions aléatoires du second ordre". *La Revue Scientifique* 84, pp. 195–206.

Lu, H. (2012). *Uncorrelated Multilinear Principal Component Analysis (UMPCA)*. MATLAB Central File Exchange. Version 1.0. URL: `https://de.mathworks.com/matlabcentral/fileexchange/35432`.

Lu, H., K. N. Plataniotis & A. Venetsanopoulos (2013). *Multilinear Subspace Learning: Dimensionality Reduction of Multidimensional Data*. Boca Raton: CRC press.

Lu, H., K. N. Plataniotis & A. N. Venetsanopoulos (2009). "Uncorrelated Multilinear Principal Component Analysis for Unsupervised Multilinear Subspace Learning". *IEEE Transactions on Neural Networks* 20.11, pp. 1820–1836.

Mallat, S. G. (1989). "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11.7, pp. 674–693.

Marx, B. D. & P. H. C. Eilers (1999). "Generalized Linear Regression on Sampled Signals and Curves: A P-Spline Approach". *Technometrics* 41.1, pp. 1–13.

– (2005). "Multidimensional Penalized Signal Regression". *Technometrics* 47.1, pp. 13–22.

Mercer, J. (1909). "Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations". *Philosophical Transactions A* 209.441-458, pp. 415–446.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller & E. Teller (1953). "Equation of State Calculations by Fast Computing Machines". *The Journal of Chemical Physics* 21.6, pp. 1087–1092.

Meyer, B. (1988). *Object oriented software construction*. New York: Prentice Hall.

Meyer-Baese, A. & V. J. Schmid (2014). *Pattern Recognition and Signal Analysis in Medical Imaging*. 2nd ed. Academic Press.

Møller, J., A. Pettitt, R. Reeves & K. Berthelsen (2006). "An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants". *Biometrika* 93.2, pp. 451–458.

Morris, J. S. (2015). "Functional Regression". *Annual Review of Statistics and Its Application* 2, pp. 321–359.

Mueller, S. G., M. W. Weiner, L. J. Thal, R. C. Petersen, C. R. Jack, W. Jagust, J. Q. Trojanowski, A. W. Toga & L. Beckett (2005). "Ways toward an early diagnosis in Alzheimer's disease: the Alzheimer's Disease Neuroimaging Initiative (ADNI)." *Alzheimer's & Dementia : The Journal of the Alzheimer's Association* 1.1, pp. 55–66.

Müller, H. G. & U. Stadtmüller (2005). "Generalized functional linear models". *Annals of Statistics* 33.2, pp. 774–805.

Müller, U. K. (2012). "Measuring prior sensitivity and prior informativeness in large Bayesian models". *Journal of Monetary Economics* 59.6, pp. 581–597.

Munkres, J. (2000). *Topology*. 2nd ed. Upper Saddle River, NJ: Prentice-Hall.

Nason, G. (2016). *wavethresh: Wavelets Statistics and Transforms*. R package version 4.6.8. URL: `https://CRAN.R-project.org/package=wavethresh`.

Nelder, J. A. & R. W. M. Wedderburn (1972). "Generalized Linear Models". *Journal of the Royal Statistical Society: Series A (General)* 135.3, pp. 370–384.

Palma, M. (2017). "Multivariate Functional Principal Component Regression for Image Data with application to Neuroimaging". MA thesis. Università di Bologna.

Peng, J. & D. Paul (2011). *fpca: Restricted MLE for Functional Principal Components Analysis*. R package version 0.2-1. URL: `https://CRAN.R-project.org/package=fpca`.

Petersen, R. C., P. S. Aisen, L. A. Beckett, M. C. Donohue, A. C. Gamst, D. J. Harvey, C. R. Jack, W. J. Jagust, L. M. Shaw, A. W. Toga, J. Q. Trojanowski & M. W. Weiner (2010). "Alzheimer's Disease Neuroimaging Initiative (ADNI): Clinical characterization". *Neurology* 74.3, pp. 201–209.

Propp, J. G. & D. B. Wilson (1996). "Exact sampling with coupled Markov chains and applications to statistical mechanics". *Random Structures & Algorithms* 9.1-2, pp. 223–252.

Quarteroni, A., R. Sacco & F. Saleri (2007). *Numerical Mathematics*. 2nd ed. New York: Springer.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. URL: `https://www.R-project.org/`.

Ramsay, J. O. (1982). "When the data are functions". *Psychometrika* 47.4, pp. 379–396.

Ramsay, J. O. & B. W. Silverman (2005). *Functional Data Analysis*. 2nd ed. New York: Springer.

Ramsay, J. O., H. Wickham, S. Graves & G. Hooker (2014). *fda: Functional Data Analysis*. R package version 2.4.4. URL: `https://CRAN.R-project.org/package=fda`.

Reed, M. & B. Simon (1980). *Methods of Modern Mathematical Physics. I: Functional Analysis*. Rev. and enl. ed. San Diego: Academic Press.

Reimherr, M., X.-L. Meng & D. L. Nicolae (2014). "Being an informed Bayesian: Assessing prior informativeness and prior – likelihood conflict". arXiv: `1406.5958`.

Reiss, P. T., J. Goldsmith, H. L. Shang & R. T. Ogden (2016+). "Methods for Scalar-on-Function Regression". *International Statistical Review*. To appear. DOI: `10.1111/insr.12163`.

Reiss, P. T., L. Huo, Y. Zhao, C. Kelly & R. T. Ogden (2015). "Wavelet-domain regression and predictive inference in psychiatric neuroimaging". *Annals of Applied Statistics* 9.2, pp. 1076–1101.

Reiss, P. T. & R. T. Ogden (2007). "Functional Principal Component Regression and Functional Partial Least Squares". *Journal of the American Statistical Association* 102.479, pp. 984–996.

– (2010). "Functional Generalized Linear Models with Images as Predictors". *Biometrics* 66, pp. 61–69.

Rosen, W. G., R. C. Mohs & K. L. Davis (1984). "A new rating scale for Alzheimer's disease". *American Journal of Psychiatry* 141.11, pp. 1356–1364.

Rudin, W. (1987). *Real and complex analysis*. 3rd ed. New York: McGraw-Hill.

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

Rue, H., S. Martino & N. Chopin (2009). "Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 71.2, pp. 319–392.

Ruigrok, A., G. Salimi-Khorshidi, M.-C. Lai, S. Baron-Cohen, M. Lombardo, R. Tait & J. Suckling (2014). "A meta-analysis of sex differences in human brain structure." *Neuroscience and biobehavioral reviews* 39, pp. 34–50.

Ruppert, D., M. P. Wand & R. J. Carroll (2003). *Semiparametric Regression*. Cambridge: Cambridge University Press.

Saporta, G. (1981). "Méthodes exploratoires d'analyse de données temporelles". PhD thesis. Université Pierre et Marie Curie, Paris.

Scheipl, F. & S. Greven (2016). "Identifiability in penalized function-on-function regression models". *Electronic Journal of Statistics* 10.1, pp. 495–526.

Schoenberg, I. J. (1946a). "Contributions to the problem of approximation of equidistant data by analytic functions. Part A. On the problem of smoothing or graduation. A first class of analytic approximation formulae". *Quarterly of Applied Mathematics* 4.1, pp. 45–99.

– (1946b). "Contributions to the problem of approximation of equidistant data by analytic functions. Part B. On the problem of osculatory interpolation. A second class of analytic approximation formulae". *Quarterly of Applied Mathematics* 4.2, pp. 112–141.

Shang, H. L. & R. J. Hyndman (2016). *rainbow: Rainbow Plots, Bagplots and Boxplots for Functional Data*. R package version 3.4. URL: `https://CRAN.R-project.org/package=rainbow`.

Shi, R. & J. Kang (2015). "Thresholded Multiscale Gaussian Processes with Application to Bayesian Feature Selection for Massive Neuroimaging Data". arXiv: `1504.06074`.

Smith, M. & L. Fahrmeir (2007). "Spatial Bayesian Variable Selection With Application to Functional Magnetic Resonance Imaging". *Journal of the American Statistical Association* 102.478, pp. 417–431.

Soueidatt, M. (2014). *Funclustering: A package for functional data clustering*. R package version 1.0.1. URL: `https://CRAN.R-project.org/package=Funclustering`.

Tarabelloni, N., A. Arribas-Gil, F. Ieva, A. M. Paganoni & J. Romo (2017). *roahd: Robust Analysis of High Dimensional Data*. R package version 1.3. URL: `https://CRAN.R-project.org/package=roahd`.

Tibshirani, R. (1996). "Regression Shrinkage and Selection via the Lasso". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 58.1, pp. 267–288.

Tierney, L. (1996). "Introduction to general state-space Markov chain theory". *Markov chain Monte Carlo in practice*. Ed. by W. R. Gilks, S. Richardson & D. J. Spiegelhalter. Boca Raton: Chapman and Hall/CRC. Chap. 4, pp. 59–74.

Tucker, J. D. (2017). *fdasrvf: Elastic Functional Data Analysis*. R package version 1.8.1. URL: `https://CRAN.R-project.org/package=fdasrvf`.

Tucker, L. R. (1966). "Some Mathematical Notes on Three-Mode Factor Analysis". *Psychometrika* 31.3, pp. 279–311.

Wand, M. & J. Ormerod (2011). "Penalized wavelets: Embedding wavelets into semiparametric regression". *Electronic Journal of Statistics* 5, pp. 1654–1717.

Wang, H. & J. S. Marron (2007). "Object oriented data analysis: Sets of trees". *Annals of Statistics* 35.5, pp. 1849–1873.

Wang, J.-L., J.-M. Chiou & H.-G. Müller (2016). "Functional Data Analysis". *Annual Review of Statistics and Its Application* 3.1, pp. 257–295.

Weiner, M. W., D. P. Veitch, P. S. Aisen, L. A. Beckett, N. J. Cairns, J. Cedarbaum, M. C. Donohue, R. C. Green, D. Harvey, C. R. Jack, W. Jagust, J. C. Morris, R. C. Petersen, A. J. Saykin, L. Shaw, P. M. Thompson, A. W. Toga & J. Q. Trojanowski (2015). "Impact of the Alzheimer's Disease Neuroimaging Initiative, 2004 to 2014". *Alzheimer's & Dementia : The Journal of the Alzheimer's Association* 11.7, pp. 865–884.

Werner, D. (2011). *Funktionalanalysis*. 7th ed. Berlin: Springer.

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. New York: Springer.

– (2011). "testthat: Get Started with Testing". *The R Journal* 3.1, pp. 5–10.

Wickham, H., W. Chang & RStudio (2016). *ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. R package version 2.2.1. URL: `https://CRAN.R-project.org/package=ggplot2`.

Winkler, G. (2003). *Image Analysis, Random Fields and Markov Chain Monte Carlo Methods*. Berlin: Springer.

Wood, S. (2003). "Thin plate regression splines". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 65.1, pp. 95–114.

Wood, S. N. (2006). *Generalized Additive Models: An Introduction with R*. London: Chapman & Hall.

Wood, S. N. (2011). "Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73.1, pp. 3–36.

– (2017). *mgcv: Mixed GAM Computation Vehicle with GCV/AIC/REML Smoothness Estimation*. R package version 1.8-17. URL: `https://CRAN.R-project.org/package=mgcv`.

Yao, F., H.-G. Müller & J.-L. Wang (2005). "Functional Data Analysis for Sparse Longitudinal Data". *Journal of the American Statistical Association* 100.470, pp. 577–590.

Yassouridis, C. (2017). *funcy: Functional Clustering Algorithms*. R package version 0.8.6. URL: `https://CRAN.R-project.org/package=funcy`.

Yu, Y., T. Wang & R. J. Samworth (2015). "A useful variant of the Davis-Kahan theorem for statisticians". *Biometrika* 102.2, pp. 315–323.

Zemyan, S. (2012). *The Classical Theory of Integral Equations*. Basel: Birkhäuser.

Zhao, Y., H. Chen & R. T. Ogden (2015). "Wavelet-Based Weighted LASSO and Screening Approaches in Functional Linear Regression". *Journal of Computational and Graphical Statistics* 24.3, pp. 655–675.

Zou, H. & T. Hastie (2005). "Regularization and variable selection via the elastic-net". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.2, pp. 301–320.

Eidesstattliche Versicherung

(Siehe Promotionsordnung vom 12. Juli 2011, § 8 Abs. 2 Pkt. 5)

Hiermit erkläre ich an Eides statt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

München, den 12.07.2017 Clara Maria Happ