POLITECNICO
MILANO 1863

Politecnico di Milano

MSc in Mathematical Engineering
Applied Statistics

Master Thesis

Geostatistics analysis on functional data decoupled in phase and amplitude variability

Advisor: Prof. Alessandra Menafoglio[a]

[a]MOX, Department of Mathematics, Politecnico di Milano

Student: Benedetta Maria Argenio
Matr. 913653

Academic Year 2019-2020

Abstract

In the context of the functional data analysis (FDA), there are some methods that allow to decompose the original functions into warping functions and alignment functions, the first accounting for the phase variability, instead the second taking into account the variability of amplitude.
The alignment step was however considered as a preprocessing step that allowed to focus on the amplitude variability. However, it was later understood that phase variability allows to obtain very important information.
The goal of our work is therefore to show that phase variability must actually be considered and in particular that if we make some geostatistics analysis, it is better to decouple the original functions into warping functions and aligned functions, study them separately and in a second moment to re-couple the results obtained.
First we explained what phase and amplitude variability are.
We then discuss how to carry out geostatistical analyses to functions belonging to the space L^2 and to those belonging to Hilbert spaces.
We then show how to apply these analyses to the warping and alignment functions.
In particular as regards the analyses on the warping functions, we study their derivatives which are embedded in the Bayes spaces.
Finally we investigate an application in the seismological field, thus showing that the error made in the analyses on the original functions is on average greater than the error made by doing the separate analyses of phase and amplitude variability.

Sintesi

Nell'ambito dell'analisi su dati funzionali (FDA) sono stati sviluppati metodi che permettono di decomporre le funzioni originali in funzioni di decomposizione, che tengono conto della variabilitá di fase, e in funzioni di allineamento, che invece tengono conto della variabilitá di ampiezza.
Lo step di allineamento peró veniva inizialmente considerato come uno step di preelaborazione che permetteva di concentrarsi sulla variabilitá di ampiezza. Successivamente si é capito che la variabilitá di fase permette di ottenere informazioni molto importanti.
L'obiettivo quindi del nostro lavoro é mostrare che effettivamente la variabilitá di fase deve essere considerata ed in particolare che, se si vogliono attuare analisi geostatistiche, é meglio disaccoppiare le funzioni originali in funzioni di deformazione e funzioni allineate, studiarle separatamente e in un secondo momento riaccoppiare i risultati ottenuti.
Come prima cosa quindi abbiamo spiegato cosa sono variabilitá di fase ed ampiezza. Abbiamo poi discusso come attuare analisi geostatistiche a funzioni che appartengono allo spazio L^2 e a quelle che appartengono agli spazi di Hilbert .
Poi abbiamo mostrato come poter applicare queste analisi alle funzioni di deformazione e a quelle di allineamento. In particolare per quanto riguarda le analisi sulle funzioni di deformazione, abbiamo studiato le loro derivate che sono incorporate negli spazi di Bayes.
Infine abbiamo indagato un'applicazione nell'ambito sismologico, mostrando cosí che l'errore commesso nelle analisi sulle funzioni originali é in media maggiore rispetto all'errore commesso facendo le analisi separate.

Contents

List of Figures

Chapter 1

Introduction

Functional data analysis is the branch of statistics which studies methods for infinite-dimensional data such as curve, images or surfaces. For this reason functional data analysis deals with data represented by functions $x_1, ..., x_N$.
Warping approaches (see, e.g., Marron, Ramsay, Sangalli, Srivastava, 2015, and references therein) consinsts in decomposing the observed function x_i into warping functions γ_i, that account for the phase variation in x_i, and aligned functions w_i, that account for the amplitude variation in the data. In particular we can obtain the observed function through the concatenation:

$$x_i(t) = (w_i \circ \gamma_i)(t) = w_i(\gamma_i(t)), \qquad t \in \tau = [a, b].$$

Thanks to the warping functions it is possible to compute the alignment, in fact the result is that the main features of w_i are aligned, this reduces the variation and makes it easier to find common structures.
Throughout this thesis, the warping and the aligned functions shall here assumed to be known, so we know the decomposition of x_i into w_i and γ_i. Therefore, to make these analyses, one should have to choose first an appropriate warping algorithm for the data. For a discussion on possible warping approaches, see, for example, Marron et al. (2015).
Usually, the warping functions are considered to transform the original functions in the aligned ones and this step is consider a part of preprocessing that allow to focus on the amplitude variability only. Instead, sometimes phase variation is an integral and foundamental part of the data that carries important information (Kneip Ramsay, 2008; Sangalli, Secchi, Vantini, Vitelli, 2010). It is thus worth to be incorporated into the analyses to obtain more information into the mechanisms generating the data. In this work we shall assume the latter viewpoint, motivated by a seismological application in which the seismic waves arrive not only with

different intensity but also with different time delays depending on their type and the geographical relation between the hypocentre and each seismometer.
In order to apply geostatistical methods to functional data, we have to consider separately the aligned and the warping functions.
In particular, for what concerning the aligned functions, since they are in L^2, we see the geostatistical methods that we can apply which have already been extensively studied.
Instead, for what concerning the warping functions, due to the complex geometric structure of their space, we shall transform them to the space of square-integrable functions, $L^2(\tau)$ through the centred log-ratio (clr) transformation, which relates to the Bayes space of densities (see, e.g., Egozcue, D-Barrero, Pawlowsky-Glahn, 2006; Hron, Menafoglio, Templ, Hruzová, Filzmoser, 2016).
Next, we compute separate geostatistical analyses for the transformed warping functions and the aligned functions and also for the original functions on data from a seismological computer experiment yielding spatially referenced high-resolution time series of ground velocity measurements. In particular we apply variogram and kriging methods to our data in order to make predictions on test locations. In this context, the joint analysis of amplitude and phase variation is of particular interest as both contain relevant information on the propagation of seismic waves.
This work proceeds as follows: in Section 2 we see how to model the amplitude and phase variability. In Section 3 we see geostatistical methods for the functions belonging to L^2 and to Hilbert spaces, while in Section 4 we see how to apply these geostatistical analyses to aligned and warping functions. In particular for what concerning the latter functions we decide to analyze their derivative γ' and to apply the Bayes approach, using the center log-ratio transformation to map the warping function from their complex space $\Gamma(\tau)$ to the known space L^2. At the end in the Section 6 we see an application in seismological framework.

Chapter 2

Modeling phase and amplitude in functional data analysis

Functional data analysis (FDA) is a part of statistics that studies infinite-dimensional data, such as curves, surfaces and images.
Recently there has been an important development of tools for functional data analysis (FDA), this is because the functional observations in the scientific framework are increasing more and more. However, it is not easy to manage this type of data for many reasons, for example because of their infinite dimensional nature or due to the presence of observation noise and for many other reasons.
The most interesting phenomenon in FDA is the presence of lateral deformations in curves, this deformation is different from variability in height or amplitude and is called phase variation.
Therefore, given a dataset of functional observations $\mathrm{X}_1, ..., \mathrm{X}_n$ defined on a domain $\tau \subset \mathbb{R}$, $\mathrm{X}_i : \tau \to R$, we can see two different types of variability: a phase variation and an amplitude variation (Menafoglio [2020]).
The phase variability represents the horizontal variability of the data, the possible misalignment of the functional observations and the amplitude variation represents the vertical variability, it affects the range of values $\mathrm{X}_i(t)$ taken by the data (Menafoglio [2020]).
The alignment problem consists on finding a set of warping function γ_i, $i = 1, ..., n$ that allow to transform the original functional data in a set of aligned functions

$$w_i = \mathrm{X}_i(\gamma_i(t)) \tag{2.1}$$

which are characterized by amplitude variability only.
The alignment phase is often considered as a preprocessing phase. This allows to eliminate the phase variability and therefore to focus only on the amplitude

variability. Later it was understood that phase variability, instead, contained significant information of the process and consequently that it was an important part of the study.
The discovery of the importance of the phase variability for the data analyses has led to an increase of the studies of the analyses of datasets forms by warping functions $\gamma_1, ..., \gamma_n$ and of the analysis of multivariate datasets $(\gamma_1, w_1), ..., (\gamma_n, w_n)$, constructed considering jointly the phase and amplitude components. (Menafoglio [2020])
There are a lot of methods that allow to decouple these two sources of variability, but in this thesis we consider as given the variability in phase and amplitude and we concentrate on the geostatistical analyses on the warping and aligned functions.
For what concerning the analysis on the variability of amplitude, geostatistical methods in L^2 are used. There are a lot of studies of these methods like Giraldo et al. (2011) or Delicado et al. (2009).
Instead, the studies on the warping functions have not been so thorough.
Therefore, in order to analyze these functions we have followed the approach of Happ et al. [2019], namely to analyze warping functions by considering their derivatives

$$\delta_i = \frac{d\gamma_i}{dt}.$$

Where δ_i is a positive function, bound to unite integration, if the codomain of γ is $I = [0, 1]$. However if δ_i is rescaled by a constant c, this just implies a rescaling of the interval I to $I' = [0, c]$ without any conceptual difference on the interpretation of the warping.
The functions $\delta_1, ..., \delta_n$ can not be considered as functions in L^2 because the geometry of this space is not appropriate. Happ et al. [2019] propose to model $\delta_1, ..., \delta_n$ as generalized distributions and to embed them in a Bayes space which instead of L^2 is designed to preserve and represent the features of these data.
The analyses of the variability, that takes into account the theory of Bayes spaces and the theory of functional alignment, have a lot of applications. For example we consider a dataset of misaligned seismic records (as in Happ et al. [2019]). In order to predict the reaction of buildings and infrastructures to shaking events (such as earthquakes), it is necessary to study the misalignment between the measurements, because this provides informations on the local effects of the propagation of seismic waves due to the geomorphology of the study area.

2.1 Amplitude-Phase Separation

As we have said we consider warping and aligned function as given.
However there are a lot of methods to implement the amplitude-phase separation.

In order to do this, Guo et al. [2020] give the following proposal following the framework of Srivastava et al. [2011], Srivastava Klassen [2016].
They consider the following space of functional data objects

$$\mathcal{F} = \{\chi : \tau \to \mathbb{R} | \chi \text{ is absolutey continuous}\}$$

The group of warping functions representing phase is $\Gamma(\tau) = \{\gamma : \tau \to \tau : \gamma$ is a diffeomorphism, $\gamma(a) = a, \gamma(b) = b\}$ with $\tau = [a, b]$.
For any $\chi \in \mathcal{F}$, $\gamma \in \Gamma$, the warping of f by γ is given by the following composition: $\chi \circ \gamma$.
The settings used by Guo et al. [2020] is the same of Menafoglio et al. [2013].
We assume a square-integrable functional random field $\{\chi_s : s \in D\}$ on a spatial domain $D \subset \mathbb{R}^2$. Associated with χ_s is its square-root slope transformed version $\{q_s : s \in D\}$ such that $s \mapsto q_s \in Q$.
The amplitude of a function χ is the equivalence class $[\chi] = \{\chi \circ \gamma | \gamma \in \Gamma\} \subset \mathcal{F}$, known as its orbit under the action of Γ. The amplitude space then is the quotient $\mathcal{F}/\Gamma = \{[\chi] | \chi \in \mathcal{F}\}$.
In order to separate amplitude and phase, we need a metric on the amplitude space $\mathcal{F}/\Gamma$. One can be defined through a metric d on $\mathcal{F}$ that is invariant to simultaneous warpings: $\forall \gamma \in \Gamma$, $d(\chi_1, \chi_2) = d(\chi_1 \circ \gamma, \chi_2 \circ \gamma)$.
To reduce the Fisher-Rao metric (Srivastava et al. [2011]) on $\mathcal{F}$ to the standard metric L^2 on the transformed space, the idea is to use the square-root slope transformation.
The transformation maps

$$f \to Q(f) = q = sgn(\chi')|\chi'|^{1/2} \tag{2.2}$$

(χ' is the time derivative of χ).
Under Q, the Fisher-Rao metric on $\mathcal{F}$ is mapped to the standard L^2 metric on Q and thus it is possible to do the analyses of the square-root slope transformed functional observations using standard Hilbert space machinery.
Warping of $\chi \in \mathcal{F}$ by γ induces the warping action $(q, \gamma) = (q \circ \gamma)\gamma'^{1/2}$ on Q with corresponding amplitude $[q] := \{(q, \gamma) | \gamma \in \Gamma\}$ and amplitude space $Q/\Gamma = \{[q] | q \in Q\}$. Amplitude and phase separation through registration or alignment of χ_2 to χ_1 is formulated as the determination of the relative phase obtained by solving

$$\gamma^* = \arg\min_{\gamma \in \Gamma} ||q_1 - (q_2, \gamma)||, \tag{2.3}$$

where q_1, q_2 are the square-root transformed χ_1, χ_2.
So observed functional data $\chi_{s_i} \in D$ $(i = 1, ..., n)$ are transformed through the square-root slope transform to obtain q_{s_i} and the model is explained using q_{s_i}.

Chapter 3

Geostatistics

In order to analyze the amplitude and phase variabilities, we have to study the aligned and warping functions.
In this section we recall the geostatistical analyses in L^2 and for the Hilbert spaces that are useful to be able to study the aligned and warping functions.
In particular first we recall the basic notions of geostatistics and then its extension to functional data.
The main purpose of geostatistics is making inference on the distribution of a random fields using a finite number of observations in some fixed locations of the domain. Whenever data are spatially distributed, there is the need to develop some analyses able to take advantage of the spatial dependence among data for modelling and prediction purposes.

3.1 Univariate approach to the geostatistical analysis

Univariate geostatistics focuses on the statistical characterization of real-valued random fields. These are collections of real random variables Z_s, which are commonly modelled as

$$Z_s = m_s + \delta_s, s \in D$$

where the mean function m_s is called drift and describes the large scale variability and the stochastic residual δ_s resumes the small scale variability and is characterized by a structure of spatial dependence.
For any given set of locations $s_1, ..., s_n$, the distributional properties of the ran-

dom vector $Z = (Z_{s_1}, ..., Z_{s_n})'$ are defined via its joint distribution function, called finite-dimensional law:

$$F_{s_1,...,s_n}(z_1, ..., z_n) = \mathbb{P}(Z_{s_1} \leq z_1, ..., Z_{s_n} \leq z_n), z_1, ..., z_n \in \mathbb{R}.$$

Definition 3.1.1. Process $\{Z(s), s \in D\}$ is said second-order stationary if the following condition hold:

- $\mathbb{E}(Z_{s_i}, Z_{s_j}) = m$, for all $s \in D$;
- $\text{Cov}(Z_{s_i}, Z_{s_j}) = \mathbb{E}[(Z_{s_i} - m)(Z_{s_j} - m)] = C(h)$, for all $s_i, s_j \in D, h = s_i - s_j$ Function C is said covariogram.

Definition 3.1.2. Process $\{Z(s), s \in D\}$ is said intrinsically stationary if

- $\mathbb{E}(z_{s_i}, Z_{s_j}) = m$, for all $s \in D$;
- $\text{Var}(Z_{s_i}, Z_{s_j}) = \mathbb{E}[(Z_{s_i} - Z_{s_j})^2] = 2\gamma(h)$, for all $s_i, s_j \in D, h = s_i - s_j$ Function γ is said semivariogram and 2γ variogram.

The semivariogram is related to the covariogram via the identity

$$\gamma(h) = C(0) - C(h), \qquad h \in \mathbb{R}^d.$$

The variogram modeling plays a key role in the geostatistical analysis of spatially dependent data.
A valid semivariogram $2\gamma(\cdot)$ is symmetric and null at the origin. However, it may present a discontinuity at the origin, associated to a non-zero limit as h approaches 0:

$$\lim_{h \to 0} \gamma(h) = \tau^2 \neq 0 = \gamma(0)$$

n this case, τ^2 is called nugget.
A further relevant property of a valid semivariogram is the sill. This is defined as:

$$\tau^2 + \sigma^2 = \lim_{h \to 0} \gamma(h)$$

where τ^2 is the nugget effect and σ^2 is said partial sill.
We also define the range R of a valid semivariogram as the value where it reaches the sill

$$\gamma(R) = \tau^2 + \sigma^2.$$

The semivariogram range quantifies the range of influence of the process: for distances greater than the range, two elements of the process are uncorrelated. The variogram range can be infinite if the sill does not exist (indication of non-stationarity) or if the sill is reached asymptotically.

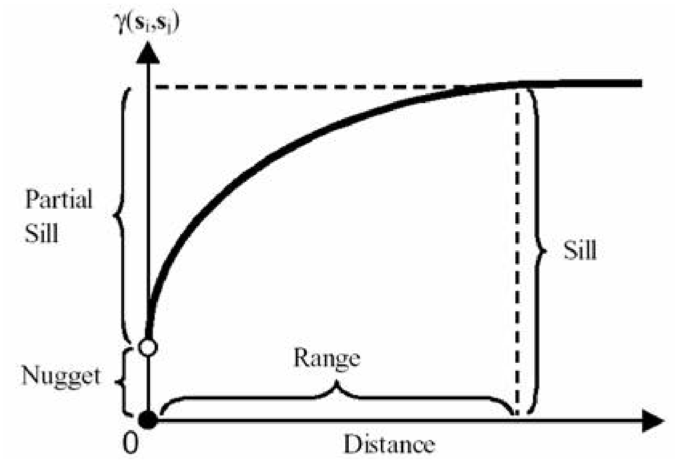


Figure 3.1: Variogram model.

To guarantee that the properties of a valid variogram are fulfilled, a number of parametric valid model are commonly employed, the most common are:

- Pure nugget (valid in $\mathbb{R}^d, d \geq 1$):

$$\gamma(h) = \begin{cases} \tau^2 & h > 0 \\ 0 & h = 0 \end{cases}$$

- Exponential model (valid in $\mathbb{R}^d, d \geq 1$):

$$\gamma(h) = \begin{cases} \sigma^2(1 - e^{-h/a}) & h > 0 \\ 0 & h = 0 \end{cases}$$

- Spherical model (valid in $\mathbb{R}^d, d = 1, 2, 3$):

$$\gamma(h) = \begin{cases} 0 & h = 0 \\ \sigma^2[\frac{3}{2}\frac{h}{a} - \frac{1}{2}(\frac{h}{a})^3] & 0 < h < a \\ \sigma^2 & h \geq a \end{cases}$$

  with $a, \sigma \in \mathbb{R}$. The model parameter are alrady interpretable: a is the range and σ^2 the sill.

- Gaussian model:

$$\gamma(h) = \sigma^2\Big(1 - exp(\frac{-\alpha h^2}{a^2})\Big) + \tau^2.$$

The variogram model which is chosen to describe the spatial variability should reflect the smoothness that one may expect from the field realization.
Given a dataset $Z_{s_1}, ..., Z_{s_n}$, under the stationarity assumption, the sample semi-variogram is computed as

$$\gamma(\hat{h}) = \frac{1}{2|N(h)|} \sum_{(i,j)\in|N(h)|} [Z_{s_i} - Z_{s_j}]^2$$

where $N(h) = \{(i, j) : ||s_i - s_j|| = h\}$ and $|N(h)|$ is its cardinality.
Then given a set of locations $s_1, ..., s_n$ in D and the observations of process $Z_s, s \in D$ at these locations, $Z_{s_1}, ..., Z_{s_n}$, one is often interested in predicting an unobserved element Z_{s_0} at s_0. A possible approach for the prediction problem is Kriging. The idea behind this method is to create a statistical model for the phenomenon starting from the available data, first by identifying the covariance structure and then, through a linear combination of the data, by making the spatial prediction. We can distinguish different kriging: Simple Kriging (SK), Ordinary Kriging (OK) and Universal Kriging (UK). Simple and Ordinary Kriging are used in the stationary setting, SK in case the mean is known and OK in case the mean is unknown over the domain D. Instead, in the non-stationary context, Universal Kriging can be applied.

3.2 Geostatistics for functional data analysis

In this period, because of the need to analyze infinite-dimensional data, such as curves, surfaces and images, functional data analysis (FDA) was deepened. Whenever functional data are spatially dependent, it is important to develop the techniques of regression and estimation, but other topics also need to be studied. In particular, an essential argument is the spatial prediction.
Given a distributional dataset $\chi_{s_1}, ..., \chi_{s_n}$ observed over n sampling locations $s_1, ..., s_n$ there are two key goals of geostatistics, namely estimate the dependence among observations at different locations and perform prediction.
To achieve these two goals, in compositional geostatistics, one have first to model a set of variograms and cross-variograms and then one have to define linear unbiased predictors from the data (i.e., compositional (co)kriging, see Tolosana-Delgado et al. [2019]).

3.2.1 Geostatistics for L^2

In this section we concentrate on functions in L^2 and we follow Giraldo et al. (2011).
Suppose to have a random process $\{\chi_s, s \in D \subseteq \mathbb{R}^d\}$ whose element χ_s is a function for any $s \in D$. These functional data belong to

$$L^2(\tau) = \{f : \tau \to \mathbb{R}, s.t. \int_\tau f(t)^2 dt < \infty\}.$$

Following Giraldo et al. (2011), we assume that the random process is weakly stationary and that the covariance functions and variograms are isotropic.
Our goal is to predict χ_{s_0} at a location s_0 that we have not observed. In particular, for the nature of our problem, we want to predict a function.
In multivariable geostatistics, the best linear unbiased predictor (BLUP) for χ_{s_0} is defined as

$$\hat{\chi}_{s_0} = \sum_{i=1}^{n} \lambda_i \chi_{s_i} \qquad \lambda_1, ..., \lambda_n \in \mathbb{R} \tag{3.2}$$

where the λs are the weights and give the influence of the curves around the position that we want to predict, the curves closer to this point will have greater influence than others more distant. The BLUP is find minimizing (Giraldo et al. [2011])

$$\sigma^2_{s_0} = \sum_{j=1}^{p} \text{Var}(\hat{Z}_{s_0}(j) - Z_{s_0}(j)) \tag{3.3}$$

where p is the variables on the location s_0 that is not observed and $Z(s) = \int_\tau \chi_s(t)dt$.
Minimize this quantity means minimize the trace of the mean-squared prediction error matrix.
Moreover, in order to find the BLUP, the weights in the kriging predictor of χ_{s_0} must be the solution of the following optimization problem:

$$\min_{\lambda_1,...,\lambda_n} \int_\tau \text{Var}(\hat{\chi}_{s_0}(t) - \chi_{s_0}(t))dt \quad s.t. \quad \sum_{i=1}^{n} \lambda_i = 1 \tag{3.4}$$

where the constraint is for the unbiasedness.
Remembering that $\gamma_t(h) = C_t(0) - C_t(h)$, the optimal weights can be formulated as the solution of the following linear system:

$$\begin{bmatrix} \int_\tau \gamma_t(\|s_1 - s_1\|)dt & \dots & \int_\tau \gamma_t(\|s_1 - s_n\|)dt & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \int_\tau \gamma_t(\|s_n - s_1\|)dt & \dots & \int_\tau \gamma_t(\|s_n - s_n\|)dt & 1 \\ 1 & \dots & 1 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \\ -\mu \end{bmatrix} = \begin{bmatrix} \int_\tau \gamma_t(\|s_0 - s_1\|)dt \\ \vdots \\ \int_\tau \gamma_t(\|s_0 - s_n\|)dt \\ 1 \end{bmatrix}$$

The function $\gamma(h) = \int_{\mathcal{T}} \gamma_t(h)dt$ is called trace-variogram and the prediction trace-variance of the functional ordinary kriging is

$$\sigma_{s_0}^2 = \int_{\mathcal{T}} \text{Var}(\hat{\chi}_{s_0}(t) - \chi_{s_0}(t))dt = \sum_{i=1}^{n} \lambda_i \int_{\mathcal{T}} \gamma_t(||s_i - s_0||)dt - \mu = \sum_{i=1}^{n} \lambda_i \gamma(||s_i - s_0||) - \mu. \tag{3.5}$$

We can also estimate the trace-variogram as

$$\hat{\gamma}(h) = \frac{1}{2|N(h)|} \sum_{(i,j)\in N(h)} \int (\chi_{s_i}(t) - \chi_{s_j}(t))^2 dt \tag{3.6}$$

where $N(h) = \{(s_i, s_j) : ||s_i - s_j|| = h\}$ and $|N(h)|$ is the number of distinct elements of $|N(h)|$.

3.2.2 Geostatistics for Hilbert spaces

This method that we have seen can be apply only to functions belonging to L^2 and not allow to treat functional data belonging to a general Hilbert spaces. Here, following Menafoglio et al. [2013,2014] we give a theoretical framework for universal kriging prediction for every separable Hilbert space, not just L^2. Suppose that the dataset $\chi_{s_1}, ..., \chi_{s_n}$ is the collection of n observations of a random field $\{\chi_s, s \in D \subseteq \mathbb{R}^d\}$ relative to n locations $s_1, ..., s_n$.
The main operative assumption is the square integrability (Menafoglio et al. [2013, 2014])

- Each element $\chi_s, s \in D$ of the random field $\{\chi_s, s \in D \subseteq \mathbb{R}^d\}$ belongs to L^2.

The first-order properties of the field are described by the mean of $\{\chi_s, s \in D\}$, which is defined, in the Fréchet sense,

$$m_s = \underset{x \in H}{\arg\min} ||\chi_s \ominus x||_H^2, \quad s \in D \tag{3.7}$$

where the operation $\ominus$ is defined as $f \ominus g = f \oplus [(-1) \odot g]$ for f,g in H.
We want to define a measure of spatial dependence for $\{\chi_s, s \in D\}$ so we introduce an extension of the classical variogram that is the trace-variogram

$$2\gamma(||s_1 - s_2||_d) = \mathbb{E}[||\chi_{s_1} \ominus \chi_{s_2}||_H^2], \tag{3.8}$$

for s_1, s_2 in D and where $||s_1 - s_2||_d$ is the distance between s_1 and s_2.
The trace-variogram that we have defined then gives a global notion of spatial

dependence, which is characterized by similar properties as its scalar counterpart (e.g., positivity, conditional negative semi-definiteness, see [Menafoglio et al., 2013]).
A stationary field is characterized by a trace-variogram that stabilizes around an asymptote for diverging distances, while the distance at which the trace-variogram reaches the sill is defined as the range of influence of the data. Alternatively, one may define a trace-covariogram as

$$C(||s_1 - s_2||_d) = \mathbb{E}[< \chi_{s_1} - m_{s_1}, \chi_{s_2} - m_{s_2} >_H], \tag{3.9}$$

that is similar to the classical covariogram. The trace-covariogram describes, in a global sense, the covariation between couples of objects of the field, and so it measures the second-order spatial dependence of the process. Intuitively, the more the distance between the positions increases, the more the spatial dependence between the associated objects decreases until it vanishes and therefore the absolute value of the corresponding trace covariogram decreases towards zero.
They are called 'trace-variogram' and 'trace-covariogram' because there is a relation between the semivariogram and the global covariance with their operatorial counterparts, they represent the trace of the corresponding operator (Menafoglio et al. [2013, 2014]).

Proposition 3.2.1. *For every couple of location s_i, s_j in D, $C(s_i, s_j)$ is the trace of the corresponding cross-covariance operator C_{s_i,s_j}:*

$$C(s_i, s_j) = \sum_{k=1}^{\infty} < C_{s_i,s_j} e_k, e_k >$$

where $\{e_k, k \in \mathbb{N}\}$ is any orthonormal basis of H. In particular:

$$|C(s_i, s_j)| \leq \sum_{k=1}^{\infty} |\lambda_k^{(s_i,s_j)}|,$$

being $\lambda_k^{(s_i,s_j)}$, $k = 1, 2, ...,$ the singular valued of the cross-variance of the operator C_{s_i,s_j}.

The expression of the trace-covariogram induces naturally the following definition:

Definition 3.2.1. The (global) variance of the process $\{\chi_s, s \in D \subseteq \mathbb{R}^d\}$ is the function $\sigma^2 : D \to [0, +\infty]$:

$$\sigma^2(s) = \mathbb{E}[||\chi_s - m_s||^2], \quad s \in D \tag{3.10}$$

These functions preserve the same properties as their finite-dimensional analogue. For what concerning the stationarity and isotropy we now see some definition:

Definition 3.2.2. A process $\{\chi_s, s \in D \subseteq \mathbb{R}^d\}$ is said to be (globally) second order stationary if the following conditions hold:

- $\mathbb{E}[\chi_s] = m, \forall s \in D \subseteq \mathbb{R}^d$,
- $\mathrm{Cov}(\chi_{s_i}, \chi_{s_j}) = \mathbb{E}[< \chi_{s_i} - m_{s_i}, \chi_{s_j} - m_{s_j} >] = C(h)$, for all $s_i, s_j \in D, h = s_i - s_j$

Definition 3.2.3. A process $\{\chi_s, s \in D \subseteq \mathbb{R}^d\}$ is said to be (globally) intrinsically stationary if the following hold:

- $\mathbb{E}(\chi_{s_i}) = m$, for all $s \in D \subseteq \mathbb{R}^d$;
- $\mathrm{Var}(\chi_{s_i}, \chi_{s_j}) = \mathbb{E}[||\chi_{s_i} - \chi_{s_j}||^2] = 2\gamma(h)$, for all $s_i, s_j \in D, h = s_i - s_j$

Definition 3.2.4. A second order stationary process $\{\chi_s, s \in D \subseteq \mathbb{R}^d\}$ is said to be isotropic if

$$\mathrm{Cov}(\chi_{s_i}, \chi_{s_j}) = C(||h||), \forall s_i, s_j \in D \subseteq R^d, h = s_i - s_j \tag{3.11}$$

where $|| \cdot ||$ is a norm on D.

We can also compute the sample semi-variogram as

$$2\hat{\gamma}(h) = \frac{1}{|N(h)|} \sum_{(i,j) \in N(h)} ||\chi_{s_i} \ominus \chi_{s_j}||_H^2 \tag{3.12}$$

where $N(h)$ is the set of pairs of data approximately separated by a distance h and $|N(h)|$ is its cardinality.
Once we have estimated the trace-variogram we want to make prediction. Suppose we have an unobservent element χ_{s_0} in the location s_0, here we use the kriging predictor in H.

3.2.2.1 Universal Kriging predictor for functional data

Kriging is a geostatistical technique that allows to perform linear spatial prediction from a set of spatially distributed data. In order to stimate a kriging predictor, we follow the proposal of Menafoglio et al. [2013,2014]. Suppose we have a non-stationary random process $\{\chi_s, s \in D \subseteq \mathbb{R}^d\}$ whose elements are representable as:

$$\chi_s = m_s + \delta_s$$

where m_s is called drift and describes the non-constant spatial mean variation and we suppose that the residual δ_s is zero-mean, second-order stationary and isotropic random field:

$$\begin{cases} \mathbb{E}[\chi_s] = m_s, & s \in D \subseteq R^d \\ \mathbb{E}[\delta_s] = 0, & s \in D \subseteq R^d \\ (\delta_{s_i}, \delta_{s_j}) = \mathbb{E}[< \delta_{s_i}, \delta_{s_j} >] = C(\|h\|), & \forall s_i, s_j \in D \subseteq R^d, h = s_i - s_j. \end{cases}$$

Assume then a linear model for m_s as in Menafoglio et al. [2013,2014]:

$$m_s(t) = \sum_{l=0}^{L} a_l(t) f_l(s), \quad s \in D, t \in \tau, \tag{3.13}$$

where $f_0(s) = 1$ for all $s \in D, f_l(\cdot), l = 1, ..., L$, are known functions of the spatial variable $s \in D$ and $a_l(\cdot) \in H, l = 0, ..., L$, are functional coefficients independent from the spatial location and H is the feature space the objects χ_s belongs to.
The coefficient a_l quantifies the effect of a unit variation of the spatial regressor $f_l(s)$, when the others are fixed, on the mean value process, $l = 1, ..., L$, instead the coefficient a_0 represents a functional intercept, in the sense that it corresponds to the (functional) mean value of the response when all the regressors $f_l(s), l = 1, ..., L$, are null.
Given n observation $\chi_{s_i}, ..., \chi_{s_n}$ sampled from $\{\chi_s, s \in D\}$, we want to formulate the Universal Kriging predictor of the variable χ_{s_0} located in $s_0 \in D$, which is the best linear unbiased predictor (BLUP):

$$\chi^*_{s_0} = \sum_{i=1}^{n} \lambda^*_i \chi_{s_i}, \tag{3.14}$$

where $\lambda^*_1, ..., \lambda^*_n \in \mathbb{R}$ are the weight that minimize the global variance of the prediction error under the unbiased constraint:

$$(\lambda^*_1, ..., \lambda^*_n) = \underset{\substack{\lambda_1, ..., \lambda_n \in \mathbb{R}, \\ \chi^\lambda_{s_0} = \sum_{i=1}^n \lambda_i \chi_{s_i}}}{\arg\min} \; \text{Var}(\chi^\lambda_{s_0} - \chi_{s_0}) \quad \text{s.t. } \mathbb{E}[\chi_{s_0}^\lambda] = m_{s_0}$$

Here, the variance to be minimized and the unbiased constrained are both well defined, this because the linear predictor $\chi^\lambda_{s_0}$ belongs to the same space H as the variables $\chi_{s_1}, ..., \chi_{s_n}$, because H is closed with respect to linear combinations of its elements. From the unbiasedness constraint, the following set of restrictions on the weights can be easily derived:

$$\sum_{i=1}^{n} \lambda_i f_l(s_i) = f_l(s_0), \quad \forall l = 0, ..., L. \tag{3.15}$$

Including this equation in the minimization problem and using the Lagrange multipliers $\mu_0, ..., \mu_L$, the problem becomes minimizing the following quantity:

$$\text{Var}(\chi^\lambda_{s_0} - (\chi_{s_0}) + 2 \sum_{l=0}^{L} \mu_l \Big(\sum_{i=1}^{n} \lambda_i f_l(s_i) - f_l(s_0) \Big)$$

and it can be reduced as:

$$\Phi = \sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j C(s_i,s_j) + C(0) - 2\sum_{i=1}^{n}\lambda_i C(s_i,s_0) + 2\sum_{l=0}^{L}\mu_l\Big(\sum_{i=1}^{n}\lambda_i f_l(s_i) - f_l(s_0)\Big)$$

$\Sigma = C(h_{i,j}) \in \mathbb{R}^{nxn}$ is the covariance matrix of the observations, namely it is a measure of dependence defined through the trace-covariogram. If this matrix is positive definite and $\mathbb{F}_s = (f_l(s_i)) \in \mathbb{R}^{n\times(L+1)}$, that is the design matrix of model, is full rank, the functional Φ admits a unique global minimum that can be found solving the following linear system:

$$\begin{bmatrix} C(0) & \dots & C(h_{1,n}) & 1 & f_1(s_1) & \dots & f_L(s_1) \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ C(h_{n,1}) & \dots & C(0) & 1 & f_1(s_n) & \dots & f_L(s_n) \\ 1 & \dots & 1 & 0 & 0 & \dots & 0 \\ f_1(s_1) & \dots & f_1(s_n) & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ f_L(s_1) & \dots & f_L(s_n) & 0 & 0 & \dots & 0 \end{bmatrix}\begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \\ \mu_0 \\ \mu_1 \\ \vdots \\ \mu_L \end{bmatrix} = \begin{bmatrix} C(h_{0,1}) \\ \vdots \\ C(h_{0,n}) \\ 1 \\ f_1(s_0) \\ \vdots \\ f_L(s_0) \end{bmatrix}$$

where $C(h_{i,j})$ denotes the trace-covariogram function of the residual process $\{\delta_s, s \in D\}$ evaluated in $h_{i,j} = \|s_i - s_j\|$.
We can also associate to the pointwise prediction $\chi^*_{s_0}$ in s_0 a measure of its global variability through the Universal Kriging variance:

$$\sigma^2_{UK}(s_0) = C(0) - \sum_{i=1}^{n}\lambda_i C(h_{i,0}) - \sum_{l=0}^{L}\mu_l f_l(s_0)$$

$$= \sum_{i=1}^{n}\lambda_i\gamma(h_{i,0}) + \sum_{l=0}^{L}\mu_l f_l(s_0), \quad s_0 \in D, f_0(s) = 1, \forall s \in D.$$

We have seen that to estimate C(h) one possible identity is

$$C(h) = C(0) - \gamma(h)$$

that relates the trace-covariogram with the trace-variogram.
If the mean function m_s is spatially constant, estimation of the trace-covariogram can be then obtained by fitting a parametric valid model (e.g., spherical, Matérn, exponential) to the empirical trace-semivariogram

$$2\hat{\gamma}(h) = \frac{1}{|N(h)|}\sum_{(i,j)\in N(h)} \|\chi_{s_i} \ominus \chi_{s_j}\|^2_H \tag{3.16}$$

where $N(h)$ is the set of pairs of data approximately separated by a distance h and $|N(h)|$ is its cardinality.

When the mean functions m_s is not spatially constant, this estimator cannot be directly used to provide a meaningful estimate of the trace-covariogram, but should be applied to the (estimated) residuals $\delta_s = \chi_s \sum_{l=0}^{L} f_l(s) \cdot \hat{a}_l$ instead.
In fact, also in the classical setting, a good estimate of the drift is then crucial for the statistical analysis.

Chapter 4

Geostatistical analysis of functional data with phase and amplitude variability

The problem to apply the geostatistical analyses on spatial functional data is that it uses methods with the variogram in L^2, this implies that the function must be assumed to be perfectly aligned otherwise the phase variation will be treated as noise. In reality, however, it is very difficult to have aligned functions, in fact the functions are often out of phase, namely there is a temporal misalignment of the geometric characteristics of the functions, such as maximum and minimum. The negative effects of disregarding phase variation are well described in (Marron et al., 2015; Srivastava et al., 2011).
So since there is often the phase variation in the observed functions, if one wants to get a better prediction of the functions in locations that have not been observed, the idea is to decompose the variability into phase and amplitude variabilities and then to do separated analyses.
In this thesis, the aligned and warping function are given. Therefore we concentrate on computing the analyses of the two types of functions and at the end propose the linear unbiased estimators for spatial prediction of amplitude and phase (and combine them to form the final prediction).
The aligned functions w_i and the warping functions γ_i have different nature and consequently have to be treated in a different way.
As we have already seen, the aligned functions are functions that belong to L^2, therefore to analyze them we apply the geostatistical analyses for L^2 functions that we have recalled in Section 3.
Instead for what concerning the warping functions we have to apply a different

approach.
We follow Happ et al. (2018) and we analyze the warping functions γ_i considering their derivatives γ_i'. These derivatives can be seen as probability density functions (PDFs) and cannot be simply considered as square-integrable functions, because the geometry of the L^2 space is not the right one to treat them.
Therefore, in order to analyze these types of data, we have decided to use the Bayes space geometry that is perfectly design to represent and to preserve the features of this data.
For this reason we now explain the Bayes approach to analyze the warping functions and then, since these spaces have the geometry of the Hilbert spaces, we will apply the geostatistical analyses for the Hilbert spaces that we have seen in chapter 3.

4.1 Bayes spaces

We have seen that the study of the warping function is not so easy.
Indeed, in order to analyze them, we have to map the functions into different space, this because the space of the warping functions

$$\Gamma(\tau) = \{\gamma : \tau \to \tau : \gamma \text{ is a diffeomorphism}, \gamma(a) = a, \gamma(b) = b\}$$

has a complex, non-Euclidean geometric structure (Lee Jung, 2016; Srivastava Klassen, 2016). In particular we want to transform the warping function to L^2.
The idea is to follow Happ et al. [2018] and to find a map $\Psi : \Gamma(\tau) \to L^2(\tau)$ which allow to transform functions to square-integrable functions, then we compute geostatistical analyses in the well-studied space $L^2(\tau)$ and at the end, through the inverse map $\Psi^{-1} : L^2(\tau) \to \Gamma(\tau)$, we transform the results back.
In particular: $\Psi : \Gamma(\tau) \to L^2(\tau)$ with $\Psi = \psi \circ D$ where D is the differential operator that maps a warping function γ to its density γ' and ψ depends on the type of the transformation.
The map ψ transports the functions from the space of density functions to L^2 and depends on the transformation that we choose.

4.1.1 Centred log-ratio transformation

Therefore, to do our analysis on the warping functions, we have decided to use the Bayes space following Happ et al.[2018] and Menafoglio [2020].
These spaces are functional spaces, whose elements are probability density functions (PDFs) over a compact support τ (as representative of σ-positive measures).

The Bayes spaces are constructed on the fact that the PDFs can also be interpreted as vector with infinitesimal parts, namely as functional compositional data. We decide to use these spaces because the warping functions can also be interpreted as generalized cumulative distribution functions of continuous random variables $X : \Omega \to \tau$ in the sense that $\gamma(a) = a, \gamma(b) = b$ and γ is monotonically increasing. Taking the first derivative γ' yields a unique (scaled) probability density function on τ.

Bayes spaces were originally introduced in Egozcue et al. [2006] and later extended in van den Boogaart et al. [2010, 2014]; they provide a mathematical and geometrical framework for performing continuous and discrete distributional data analysis.

We focus on a particular Bayes space, the Bayes Hilbert space

$$B^2(\tau) = \{f > 0 s.t. \int_\tau [lnf(t)]^2 dt < \infty\}, \tag{4.1}$$

this is the space of positive functions defined on τ with square-integrable logarithm.

The analysis of PDF data should follow some key principles (Menafoglio [2020]). The principle according to which PDFs should be scale invariants gives us the definition of the equivalence relation underlying Bayes spaces, namely given two functions $f, g \in B^2(\tau)$, they are equivalent in the Bayes space if they are proportional ($f = \alpha g$) through a positive constant α.

A natural representative for each class is given by the function integrating to $\eta = b - a$, which we interpret as the derivative of a warping function. For $f, g \in B^2(\tau)$ and $\alpha \in \mathbb{R}$, operation on $B^2(\tau)$ are defined as:

$$(f \oplus g)(t) = \eta \frac{f(t)g(t)}{\int_\tau f(s)g(s)ds} \qquad (\alpha \odot f)(t) = \eta \frac{f(t)^\alpha}{\int_\tau f(s)^\alpha ds}$$

$$< f, g >_B = \frac{1}{2\eta} \int_\tau \int_\tau log(\frac{f(x)}{f(y)}) log(\frac{g(x)}{g(y)}) dydx.$$

Usually the origin of a Bayes space is the reference measure set to define the space itself, in our case it coincides with the neutral element of the perturbation and in particular it is the uniform density $1/\eta$ (Menafoglio [2020]).

In the light of Bayes theorem, these operations give us interesting notions. In particular the first operation show us how the perturbation is represented as a sum of information, namely to the information in g are added the information in f and viceversa. Similarly, regarding the second operation, it can be seen that the information in f is inflated by α, which is a constant value (Menafoglio [2020]). Moreover the inner product induce the following norm:

$$||f||_B = [\frac{1}{2\eta}\int_\tau\int_\tau (log\frac{f(t)}{f(s)})^2 dtds]^{1/2}, f, g \in B^2. \tag{4.2}$$

Each element of $B^2(\tau)$ can be mapped to $L^2(\tau)$. To do that we use the log-ratio transformations, among these the one by far most used is the centred log-ratio transformation (clr), which is defined, for $f \in B^2(\tau)$, as

$$\Psi_B(f)(t) = log(f(t)) - \frac{1}{\eta}\int_\tau log(f(x))dx \quad \Psi_B^{-1}(f)(t) = \eta \cdot \frac{exp(f(t))}{\int_\tau exp(f(s))ds}$$

The centred log-ratio transformation is an isometric isomorphism with respect to the norm induced $< \cdot, \cdot >_B$ between the space B^2 and the space L^2, in particular Ψ_B allow to map into the subspace $U_B(\tau) = \{v \in L^2(\tau) : \int_\tau v(s)ds = 0\}$.
Therefore, the operations and inner products in B^2 are preserved under the clr-transformation:

$$\Psi_B(f \oplus g) = \Psi_B(f) + \Psi_B(g), \quad \Psi_B(\alpha \odot f) = \alpha \cdot \Psi_B(f)$$
$$< f, g >_B = < \Psi_B(f), \Psi_B(g) >$$

with the standard operations $(+, \cdot, < \cdot, \cdot >_2)$ on $L^2(\tau)$.
Therefore, the Hilbert space geometry of the space $(B^2, +, \cdot, < \cdot, \cdot >_2)$ with the clr-transformation properties allow to formulate the methods of geostatistical analyses equivalently in the Bayes space.
The advantage of this approach is that the geometry of these spaces takes into account the features of the data, which is not possible working in $L^2(\tau)$.

4.1.2 Other transformations

Always following Happ et al. (2018) we can see other transformations that one can used instead of the clr. In particular we can see the square-root velocity transformation, the log-Hazard transformation and log-quantile density transformation.

4.1.2.1 Square-root velocity transformation

Even this transformation is useful to transform the warping functions to $L^2(\tau)$. In particular to do that we need to fine the map Ψ, that consists into 2 different

steps.
In the first step, the warping functions are mapped to the positive orthant of the (scaled) unit sphere in $L^2(\tau)$: $S_+^\infty(\tau) = \{s \in L^2(\tau) : ||s||_2^2 = \eta, s \geq 0\}$, where η is the length of the interval τ.
This is realized by the square-root velocity function:

$$SRVF : \gamma \mapsto \sqrt{\gamma'} \tag{4.3}$$

where γ' denotes the first derivative of γ.
The second step is done via mapping

$$\hat{\psi_{S,\mu}} : S_+^\infty(\tau) \to T_\mu(\tau),\ \ q \mapsto \frac{\theta}{\eta^{1/2}\sin(\theta)}(q - \cos(\theta)\mu),\ \ \theta = \cos^{-1}\Big(\frac{< q, \mu >_2}{\eta}\Big) \tag{4.4}$$

where $\mu \in S_+^\infty(\tau)$ and $T_\mu(\tau) = \{v \in L^2(\tau) :< v, \mu >_2 = 0\}$.
The back transformation to $\Gamma(\tau)$ is again in two steps. First to map v to the sphere $S^\infty(\tau) = \{s \in L^2(\tau) : ||s||_2^2 = \eta\}$ apply

$$\hat{\psi_{S,\mu}} : T_\mu(\tau) \to S_+^\infty(\tau),\ \ v \mapsto \cos\big(||v||_2\big)\mu + \eta^{1/2}\sin\big(||v||_2\big)\frac{v}{||v||_2} \tag{4.5}$$

Then, the results are mapped back to the space of warping functions via

SRVF^{-1} : $S^\infty(\tau) \to \Gamma(\tau)$ with $SRVF^{-1}(s)(t) = a + \int_0^t s(u)^2 du$ The total mapping from $\Gamma(\tau)$ to $L^2(\tau)$ is given by:

$\Psi = \hat{\psi_{S,\mu}} \circ SRVF$ with the inverse $\Psi^{-1} = SRVF^{-1} \circ \hat{\psi_{S,\mu}}$ However the square-root velocity transformation has two defects.
First $\hat{\psi_{S,\mu}}$ and $\hat{\psi_{S,\mu}^{-1}}$ are undefined for the rather interesting points $\mu \in S_+^\infty(\tau)$ and $v_0 = 0 \in T_\mu(\tau)$, which implies $||v_0||_2 = 0$. Theoretically, thanks to L'Hopital's rule they can be easily be completed, but in practice, for functions close to μ and v_0, often occur computational instabilities (Happ et al.[2018]). At the same time, the projection from the positive orthant of the sphere $S_+^\infty(\tau)$ to the tangent space $T_\mu(\tau)$ is a local approximation that works best close to μ. This shows that the choice of μ is important and requires the data to be neither too close nor too far from μ (Happ et al.[2018]).
The second defects is that the mappings $\hat{\psi_{S,\mu}}$ and $\hat{\psi_{S,\mu}^{-1}}$ are not inverse to each other as

$$dom(\hat{\psi_{S,\mu}}) = S_+^\infty(\tau) \subset S^\infty(\tau) = im(\hat{\psi_{S,\mu}^{-1}}) \quad im(\hat{\psi_{S,\mu}}) \subset T_\mu(\tau) = dom(\hat{\psi_{S,\mu}^{-1}}). \tag{4.6}$$

Whenever one uses the projected SRVFs v_i for statistical analysis in the tangent space $T_\mu(\tau)$ whose results are not guaranteed to stay within $im(\hat{\psi_{S,\mu}})$, there is a risk of obtaining atypical results on the level of the SRVFs and of the warping functions (Happ et al. 2018).

4.1.2.2 Log-Hazard transformation and log-quantile density transformation

Petersen and Muller (2016) present other two alternative transformations that can be extended to warping functions by applying them to the derivatives of the latter. The log-hazard transformation is

$$\psi_H(f) = log(\frac{f}{1-F}) \tag{4.7}$$

where F is the warping function scaled to $[0,1]$ and f is the associated density. The transformation is implemented only on $[a, b - \delta\eta]$, namely on a subinterval with a threshold parameter δ because of the hazard functions are known to diverge at the right endpoint of τ. Instead, for what concerning the back transformation, uniform weight is assigned to $t \in (b - \delta\eta, b]$.
The second transformation is the log-quantile density transformation

$$\psi_Q(f) = -log(f(Q)) \tag{4.8}$$

where Q is the inverse (quantile) function associated with F.
Petersen and Muller advise to use the second one.
The log-hazard transformation ψ_H can be highly influenced by the threshold parameter δ, while the log-quantile density transformation ψ_Q requires numerical inversion of the warping functions γ, which may also lead to instabilities. Happ et al. [2018] try to use both the transformations and note that, in all the analyzes cases, the log-hazard transformation was the best contradicting the recommendation given by Petersen and Muller (2016).

4.2 Another approach

Cuncurrently with the writing of the thesis, Guo et al (2020) study a different approach.
Their idea is to do the same procedure that we have done but with two differences. The first one is that they don't consider the warping and aligned functions as given functions, they have separated them through the square-root slope transformation. The second difference is that they don't use the Bayesian approach to study the warping function but the SRVF transformation that we have introduced before.
In particular they find two random fields, one related to the amplitude phase $\{w_s, s \in D\}$ and one related to the warping $\{\psi_y = \gamma'^{1/2}, y \in D'\}$ and they find two trace-variogram related to them and consequently two predictions through the kriging technique.

Chapter 5

Application

Now we apply what we have seen in the previous section. In particular, we apply geostatistical analyses to a subset of data from a seismological in silico experiment based on the Mw 6.7 1994 Northridge earthquake, a blind thrust event that was felt over 200.000 km^2.
The induced ground shaking caused 60 victims, more than 7.000 wounded, 40.000 damaged buildings and $44 billion in economic losses.
On a fault dipping southward at about 35' below the San Fernando Valley in the Los Angeles metropolitan area (Hauksson, Jones, Hutton, 1995) was located the hypocentre at about 19 km depth.
Irregular distributions are shown from the patterns of damage that occurred during the Northridge event. Generally, the region closest to the earthquake (so closest to the hypocentre) was shaken most severely. However, there were also isolated pockets of damage at distant locations.
In computational seismology there has always been the challenge of estimating realistic ground movements for the topography of complex surfaces. However, this is not easy because the local site responses to seismicity are influenced by topography and this influence, despite having a major impact in the propagation of seismic waves and in the movement of the ground during earthquakes, is very complex to consider to predict the seismic risk. This seismic risk is based on equations that describe the forecasts of ground movement (Boore, 1973; Bouchon, 1973).
For the data analysed here, physics-based earthquake scenarios were modelled using three-dimensional unstructured meshes constructed from geological constraints such as high-resolution topography data and the Southern California Earthquake Center Community Fault Model combined with a one-dimensional subsurface structure (Happ et al. [2019]).
It is used SeisSol, an open-source package that simulate the earthquakes. It couples

three-dimensional seismic wave propagation to the simulation of dynamic rupture propagation across earthquake fault zones, (Heinecke et al., 2014; Pelties, Gabriel, Ampuero, 2014; Uphoff et al., 2017, Happ et al. [2019]). SeisSol has performed multiple simulations varying the initial fault stress and strength conditions.
Through a dense network of virtual seismometers distributed across Southern California were registered time series of absolute ground velocity.
Bauer, Scheipl, Kchenhoff, and Gabriel (2018) have described more in details this data and have done an analyses on the full dataset of original, unregistered data. On this dataset Happ et al. [2019] has already done the multivariate PCA, instead we want to use this dataset in order to make some predictions through geostatistics analysis.
For our application we decide to focus on locations that are at most $40km$ away from the hypocenter from two simulations and therefore on average they show more significant movements of ground velocity.
This constraint has reduced the sites to a total of 1,558 observation units, each of which is recorded at 2 Hz over 30 s for a total of 61 timepoints.
Happ et al. [2019] presmoothed the ground velocity curves using a Tweedie distribution with log-link for the response and 40 cubic regression splines with the penalized second derivative using the R-package mgcv (Wood, Pya, Sen, 2016) to model the evolution over time before registration.
These data are available as additional material of Happ et al. [2018].
The goal of our analysis is to show that from the kriging on the original data (not aligned) we obtained a worse prediction than the prediction obtained from the analysis of the functions decoupled in aligned and warping.
First we have uniformly sampled from these data 500 units and we have divided them into training (350 points) and test (150 points) sets.

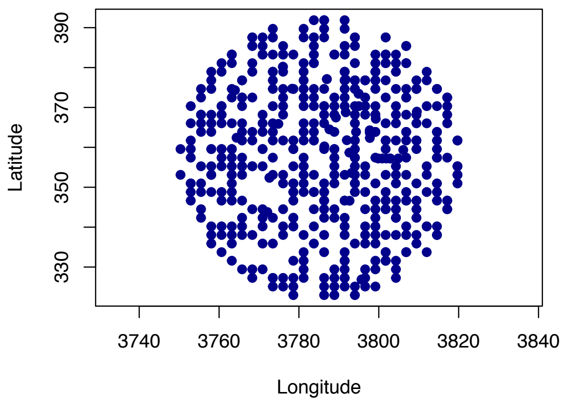


Figure 5.1: 500 points we have selcted.

The idea is to do our analyses on the training set and then to compute the prediction error we make on the test set. In particular we do the same procedure for the original, aligned and warping function. We will see that the prediction error obtained from the analyses on the original data is bigger than the error committed decoupling the functions in aligned and warping and doing separate analyses.
In 5.2 we can see the smoothed curves χ_s representing $log(1 + V_i(t))$ for ground velocity V at locations s_i and time t, the warping functions γ_s and the aligned functions χ_s^a after SRVF-based warping (Tucker, 2014). The results of the warping and aligned are the inputs of our analysis.

$$\chi_s^a = \chi_s(\gamma_s) \qquad \chi_s = \gamma_s^{-1}(\chi_s^a)$$

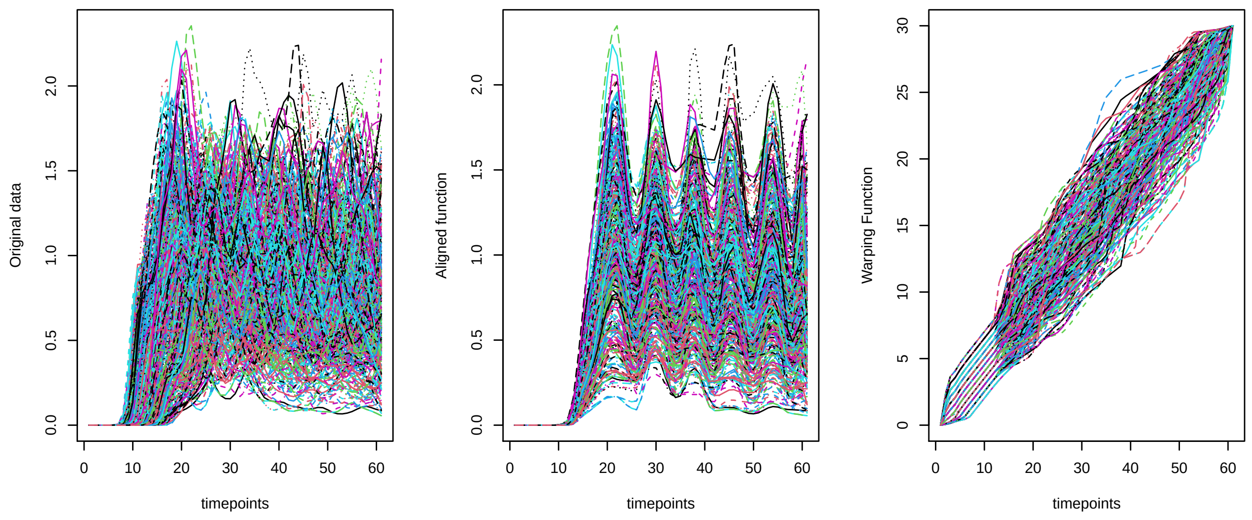


Figure 5.2: Earthquake dataset representing 500 observations of simulated ground velocity over time.

Therefore, our dataset is composed by the following columns: longitude, latitude, distance from the hypocentre (transformed in km), ground velocity, aligned function and warping function.
From the 5.2 it is evident that while the original data are characterized by both variabilities, the aligned function only has the amplitude variability without any horizontal component.
We start by performing exploratory data analysis.
From the following figure, we see that effectively the hypocentre and the ground velocity have a negative correlation, indeed the further a location is from the hypocentre, the slower the velocity and viceversa.

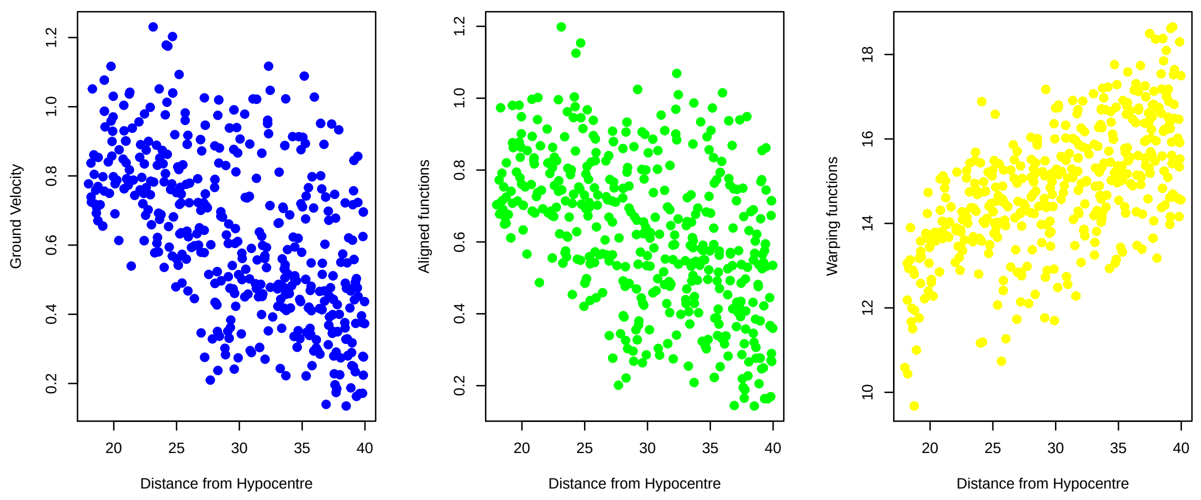
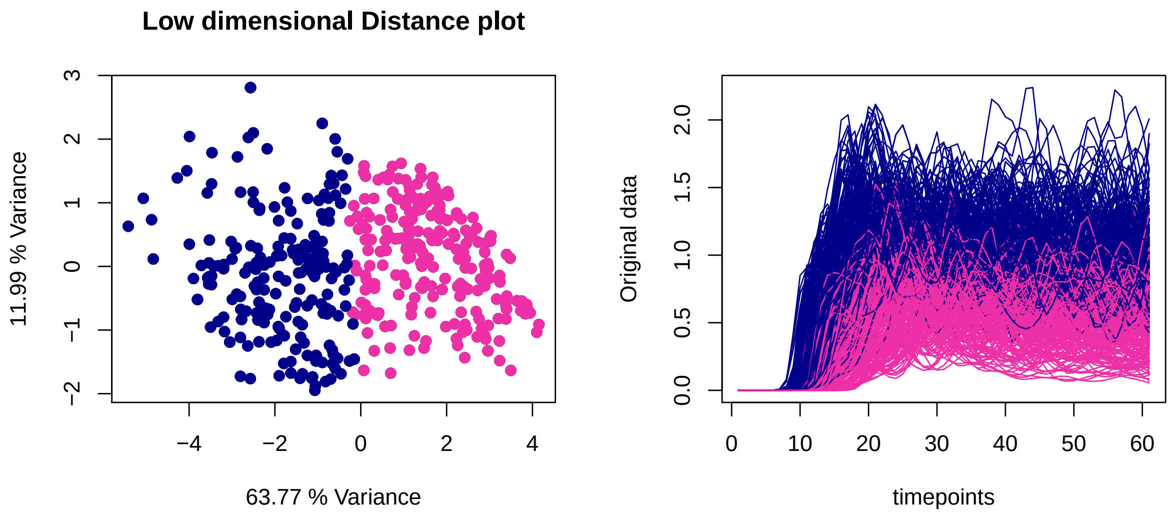


Figure 5.3: Correlation between the mean of the functions and the hypocentre distance.

In order to be able to visualize the figure (5.3) we have computed the mean of original, aligned and warping functions in time and then the correlation between these means and the distance from the hypocentre measure.
We have also clustered the three functions, for this purpose we have computed Euclidean distances between the production profiles and then we have applied simple kmeans clustering.



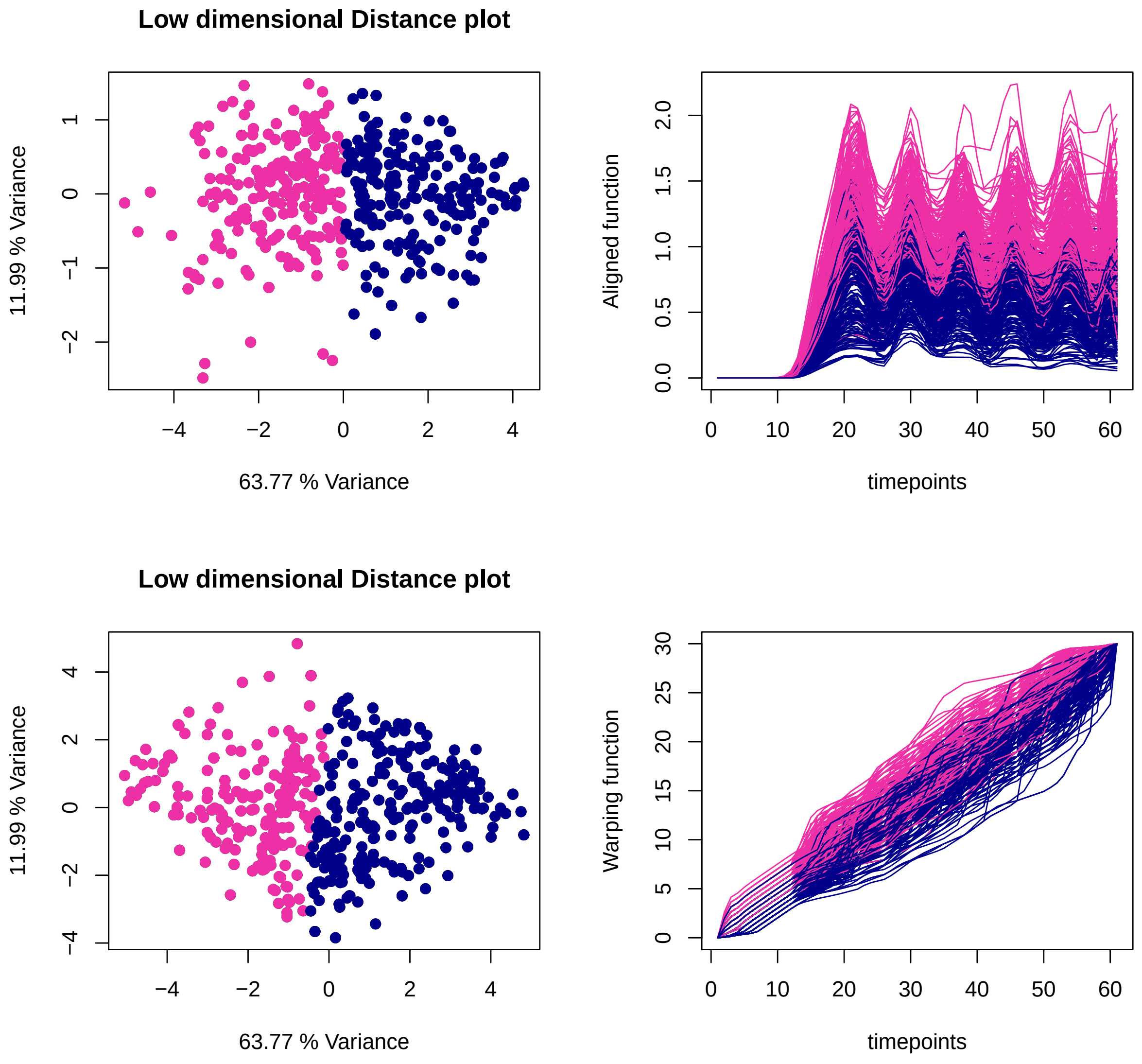


Figure 5.4: Clustering of original, aligned and warping functions.

We have also visualized the statistical data depth, that is a nonparametric tool applicable to multivariate datasets in an attempt to generalize quantiles to complex data. The depth measures how deep (or central) is a datum respect to a population. We have visualized this parameter not only for the functions but also for their derivatives.

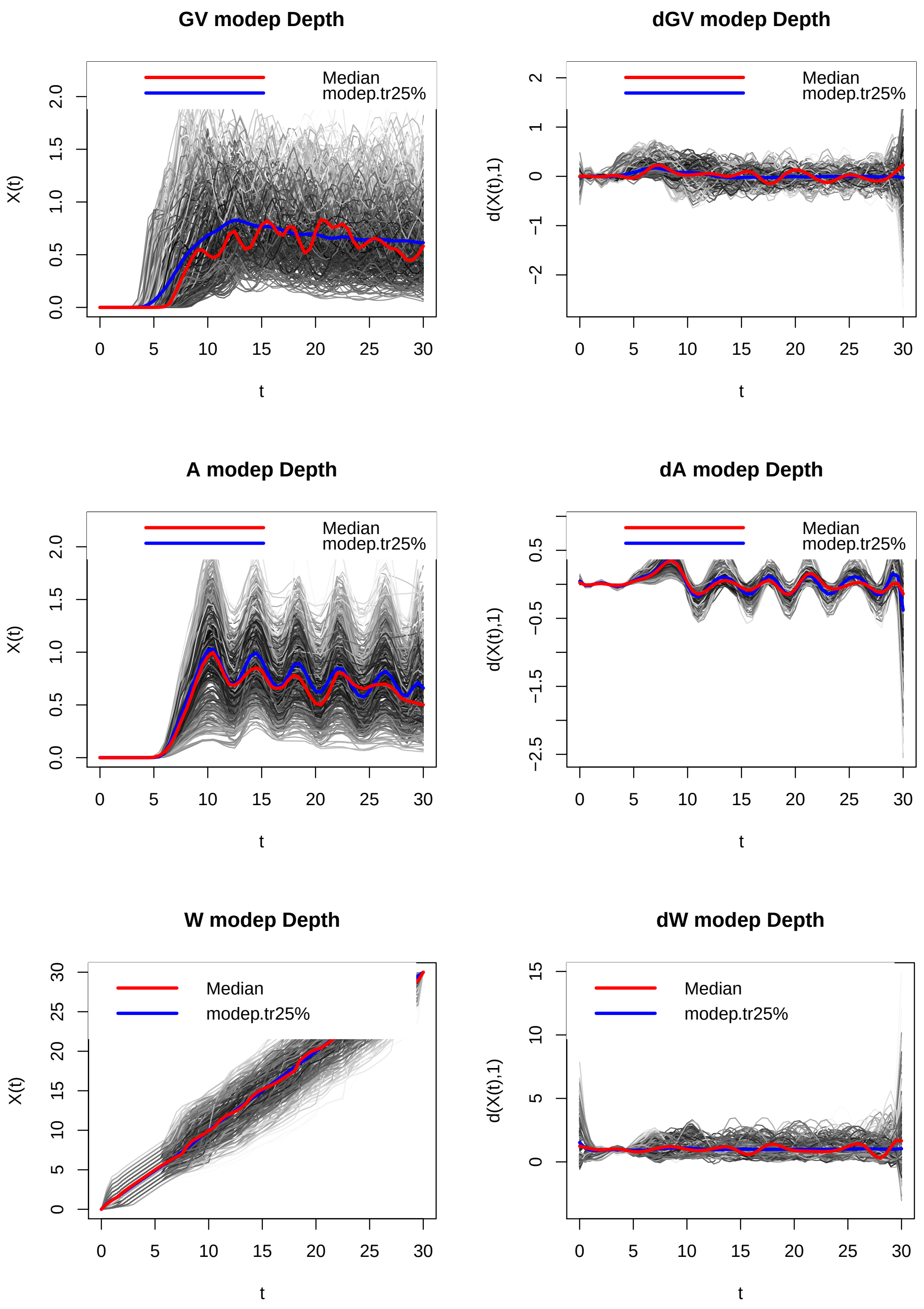


Figure 5.5: Depth measures for the original, aligned and warping functions and their derivatives.

This depth function also displays (see figure 5.5) the curves (in terms of its depth on a gray scale), the median curve (red line) and the mean of $(1-\alpha)\%$ deepest curves (blue line).
Following the analysis of Happ et al. (2018) we have computed the multivariate functional principal component analysis

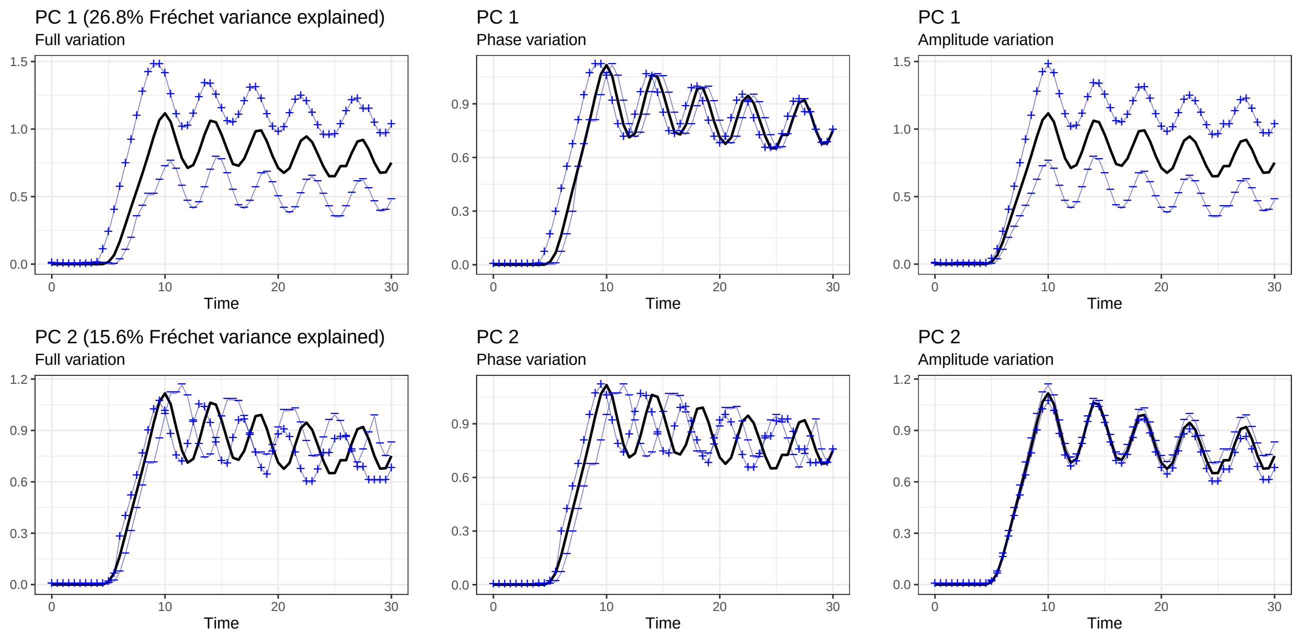


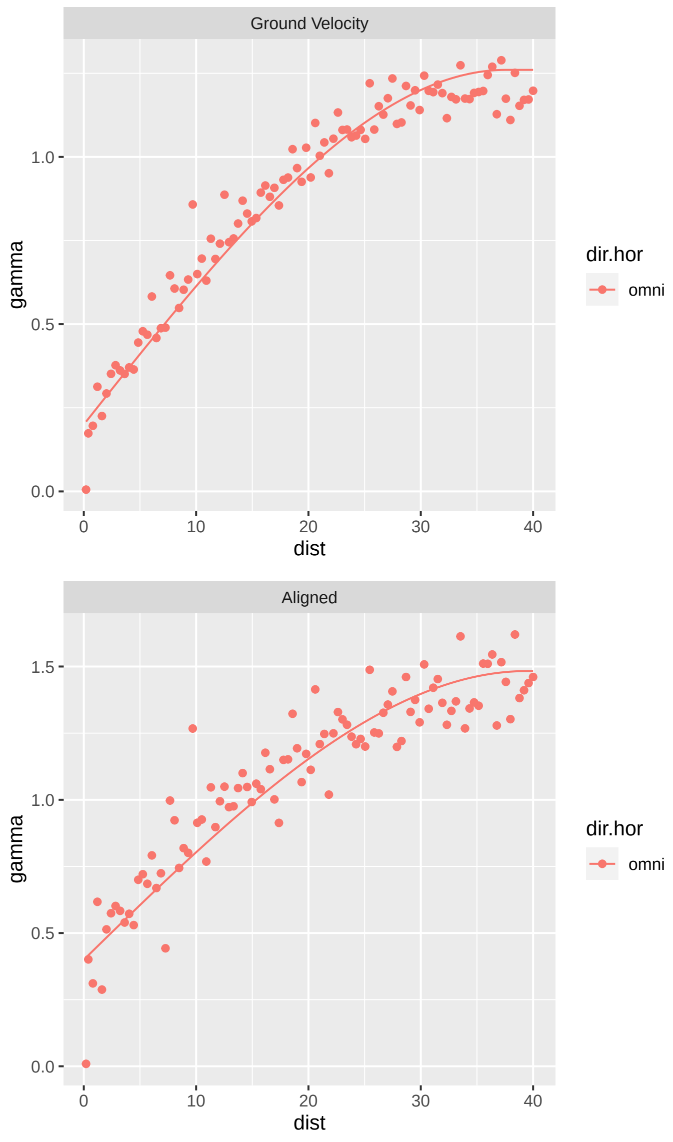
Figure 5.6: First two principal component of multivariate functional principal component analysis.

We have seen that, as Happ et al. (2018) say, for the first principal component the main component is the amplitude variability that means that for positive scores the ground movement is bigger than for the negative scores instead for the second principal component the main component is the phase variability that shows that for positive scores the peaks occur earlier than in the mean function instead for the negative scores they occur later than the mean function.
After this exploratory analyses, we started to do our geostatistical analyses.
In particular, first of all we have transformed the functions in the following way:

- The warping functions γ_i, through the center log-ratio transformation Ψ_B.
- The aligned functions and the original functions respectively in $log(\chi_s^a + 1)$ and $log(\chi_s + 1)$, this in order to keep the positivity constrain.

After that we have started to do our analysis using the package of R called 'fdagstat' ([12]). We have search the best trace-variogram model for all the functions

using the training set.
In order to do this, we have seen how the variance of the statistical variable (or, better, the semivariance) evolved as a function of the distance from the measurement points through a plot and thanks to this plot we have fixed the value of nugget, range and sill.
For the original ground velocity we have chosen a spherical model with nugget equal to 0.2, sill 35 and range 2, for what concerning the aligned function we have chosen a spherical model with nugget equal to 0.4, sill 30 and range 1.5 and for the warping function we have choosen an exponential model with nugget equal to 0.45, sill 30 and range 1.5.



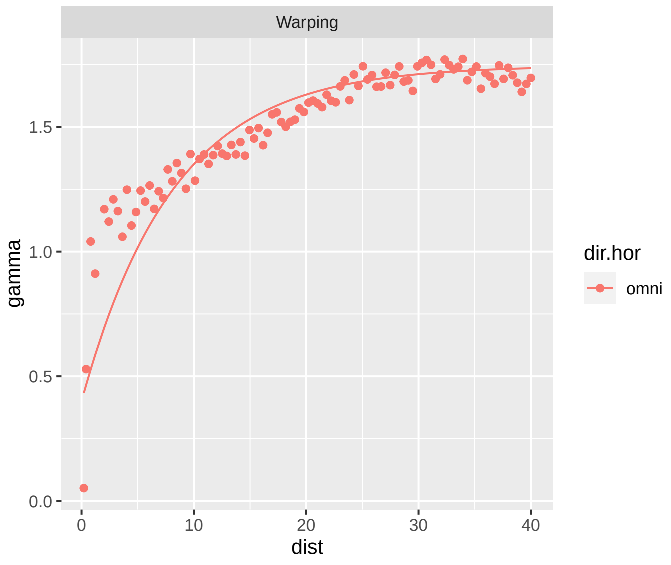


Figure 5.7: Variograms of original ground velocity, aligned functions and warping functions respectively.

After that our goal is to make prediction, in particular we have used the trace-variogram model found before and the Universal Kriging on the test set to compare the results with the real values of the functions for the original, aligned and warping functions.

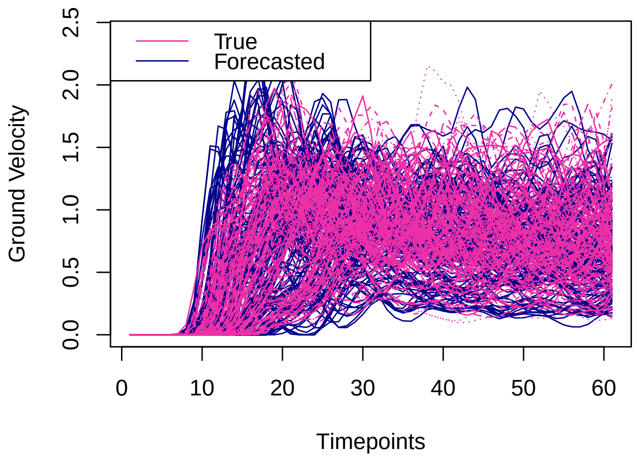
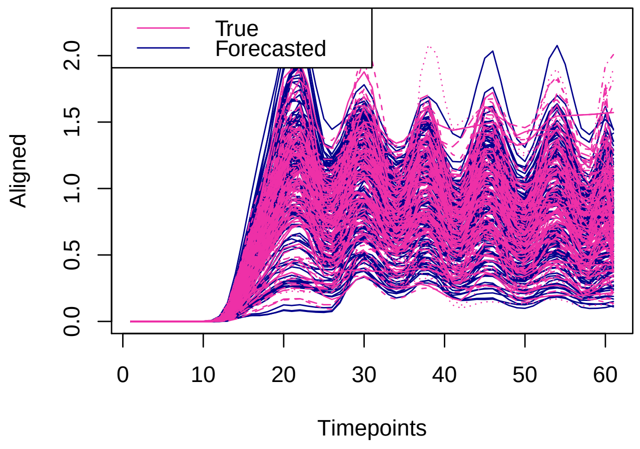
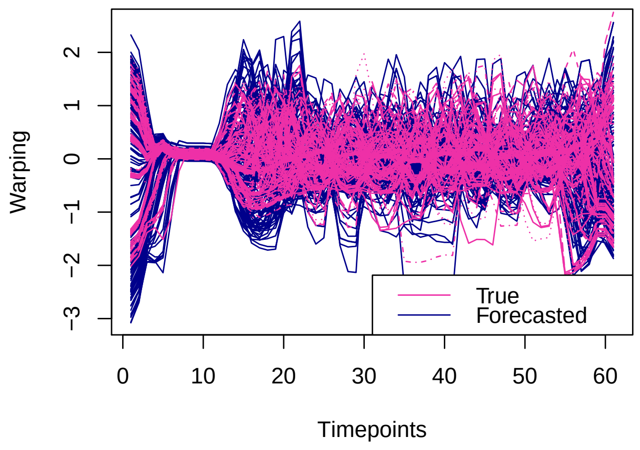


Figure 5.8: Kriging of original ground velocity.





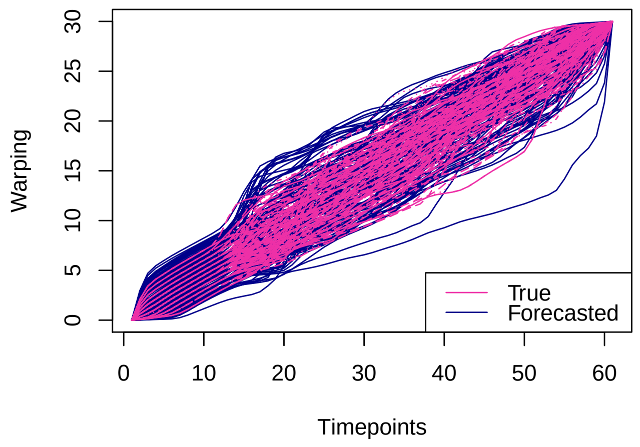


Figure 5.9: Kriging of aligned functions, of warping functions transformed through the clr transformation and of the warping function transformed back through the clr inverse transformation respectively.

So we did three separate analyses and we have obtained three different predictions $\chi^*_{s_0}, \chi^{a,*}_{s_0}, \gamma^*_{s_0}$.
We after have transformed back the result of the prediction of the warping function through the inverse map, namely the center log-ratio inverse transformation.
Then we want to couple what was obtained from warping functions and from aligned functions and subsequently compare this result with the analysis made on the original data. In particular we rewrite the predictions in the following way:

$$\chi^*_{s_0} = \chi^{a,*}_{s_0}(\gamma^*_{s_0})$$

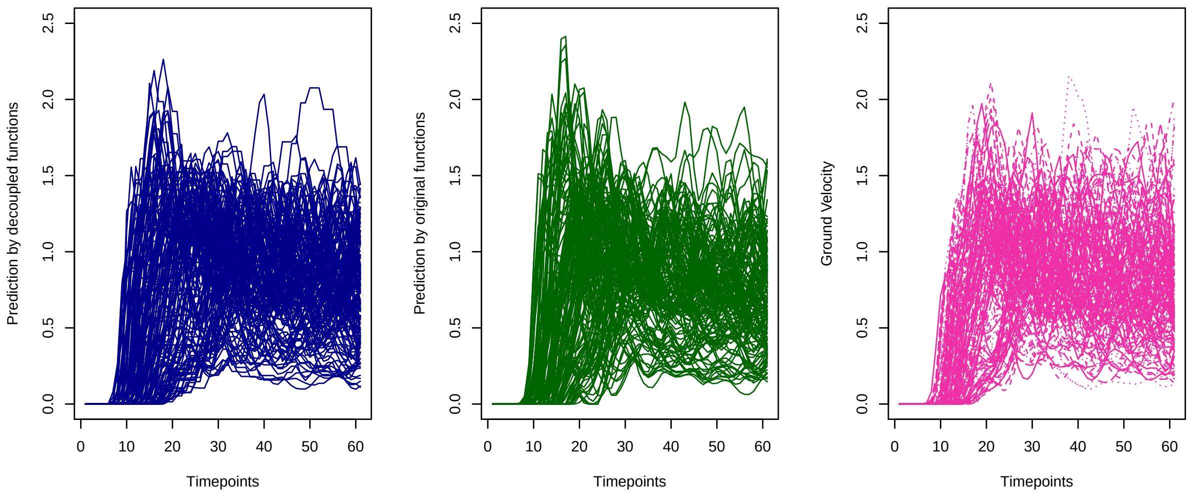


Figure 5.10: Left: prediction of ground velocity by coupling aligned and warping functions, Middle: prediction of ground velocity by original function, Right: ground velocity of the test set.

We compute the two errors in $L^2(\tau)$ and to do it we use the integral of the square of the difference

$$d^2(\chi^*_{s_0}, \chi_{s_0}) = \int_{\mathcal{T}} (\chi^*_{s_0} - \chi_{s_0})^2,$$
$$d^2(\chi^{a,*}_{s_0}, \chi_{s_0}(\gamma_s)) = \int_{\mathcal{T}} (\chi^{a,*}_{s_0} - \chi_{s_0}(\gamma_s))^2.$$

After that we compute the boxplot of these two errors to see if the analysis on the decouple data is the best one or not and we obtain the following figure

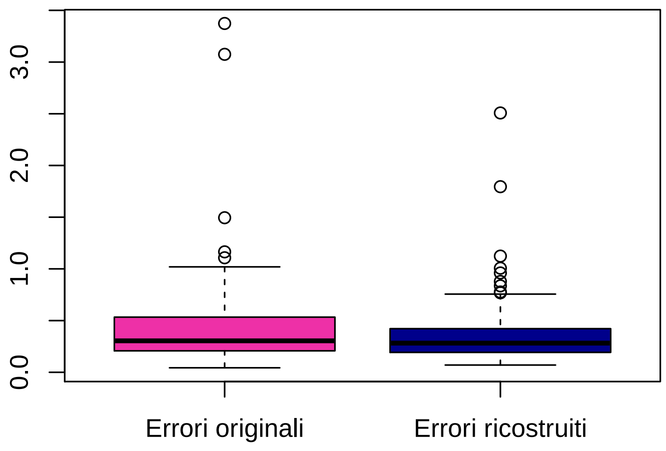


Figure 5.11: Prediction errors of the two different analysis.

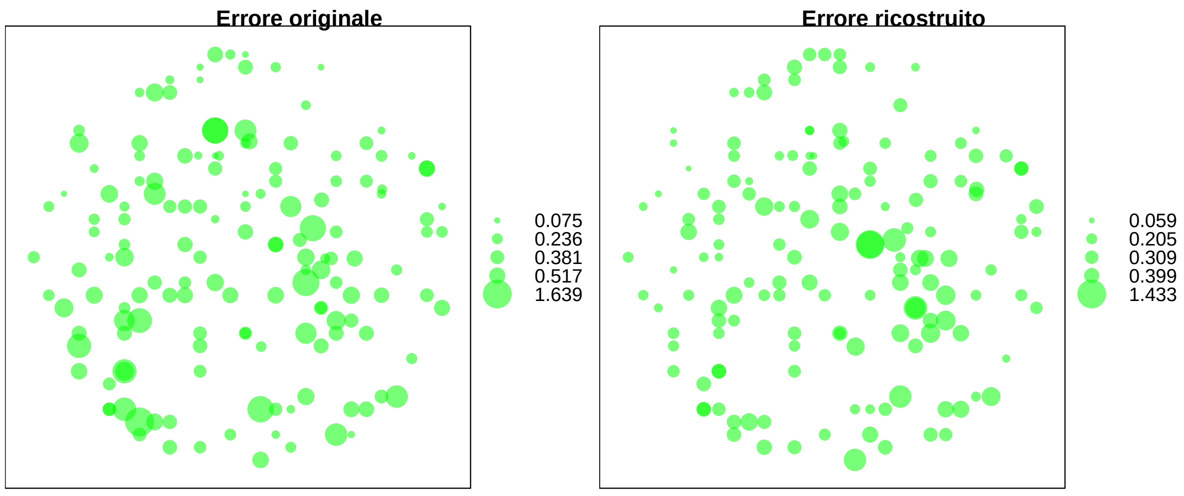


Figure 5.12: Map of the two types of errors.

Statistiche	Errore originale	Errore ricostruito
mean	0.423	0.361
median	0.304	0.282
minimum	0.043	0.070
maximum	3.373	2.501
sum	64.086	54.131
standard deviation	0.421	0.294

We can see from 5.11 that the boxplot of the coupled prediction errors is lower then the boxplot of the original one. This mean that we have proved that it is better the analysis on the data decoupled in warping and aligned function instead of a single analysis on the observed function.
Also from 5.12 we can see this result.
This result confirm that considering the data and making the geostatistical analyses on the original functions means not consider the phase variability. Instead this variability is significant, in fact contains important information about the process. Therefore, decoupling the functions in aligned and warping we take into account both variabilities and we make a better analyses.
But we did this analysis on a single division of the training and test sets, we want to control that on average for all the choices of the two sets the result is the same. Therefore we have done 100 different choices of the 2 sets and 100 different analysis, for all of them we have saved the two errors and at the end we have computed the boxplot once again

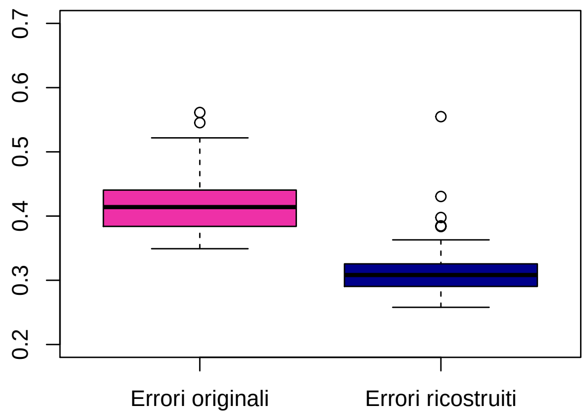


Figure 5.13: Prediction errors of the two analysis for 100 different choices of training and test sets.

This result therefore confirms what we have concluded before by making a single analysis, even on average the predicted error relating to the original functions is higher than the one relating to the functions decoupled in algned and warping functions and separately studied.

Chapter 6

Conclusion

The seismological experiment presented in section 5 is a perfect example of data where both amplitude and phase variabilities are of interest.
Let's summarize what we have done: to do our analysis on this dataset we have first decoupled the original functions into alignment and warping functions to take into account the overall variability, therefore both that of phase and amplitude. We have used the clr transformation shown in Section 3 for the warping functions, to preserve the nature of the data. We have made the geostatistical analyses on the functional data illustrated in Section 3 for the aligned functions and warping functions, using the analyses for functions in L^2 and for functions belonging to Bayes spaces, respectively. We have therefore obtained two prediction estimates using the kriging methods. Next we have coupled the results of the analyses on the warping and alignment functions. We have compared this result with the one obtained through geostatistical analysis made on the original data by comparing the forecast errors of both procedures.
We have performed these steps 100 times and we have seen that on average the error we made on the prediction using the original data is greater than the one we made through the decoupling process.
The conclusion is not only that phase variability is actually important, that it contains useful data information and therefore decoupling should not only be a preprocessing phase, but we have also concluded that to study and to do geostatistical analyses on functional data, the best thing to do is to decouple this data, study the warping and alignment functions separately, and only later couple the results. In this way, in fact, we are able to study the warping functions through the Bayes approach, preserving the intrinsic features of the data.

References

[1] Matilde Dalla Rosa Alessandra Menafoglio Piercesare Secchi. “A Universal Kriging predictor for spatially dependent functional data of a Hilbert Space”. In: *Electronic Journal of Statistics* (2013).

[2] Piercesare Secchi Alessandra Menafoglio. “O2S2: A new venue for computational geostatistics”. In: *Applied Computing and Geosciencesr* (2019).

[3] Piercesare Secchi Alessandra Menafoglio. “Statistical analysis of complex and spatially dependent data: A review of Object Oriented Spatial Statistics”. In: *European Journal of Operational Research* (2016).

[4] Piercesare Secchi Alessandra Menafoglio and Alberto Guadagnini. “Geostatistical analysis in Bayes spaces: probability densities and compositional data”. In: (2018).

[5] James O. Ramsay Alois Kneip. “Combining Registration and Fitting for Functional Models”. In: *Journal of the American Statistical Association* (2008).

[6] Scheipl Bauer A., F. Kchenhoff, and H. Gabriel. “An introduction to semi-parametric function-on-scalar regression.” In: *Statistical Modelling* (2018).

[7] Boore. “Functional Data Analysis of Amplitude and Phase Variation”. In: *Bulletin of the Seismological Society of America* (2015).

[8] Boore. “The effect of simple topography on seismic waves: Implications for the accelerations recorded at Pacoima Dam, San Fernando Valley, California.” In: *Bulletin of the Seismological Society of America* (1973).

[9] Bouchon. “Effect of topography on surface motion”. In: *Bulletin of the Seismological Society of America* (1973).

[10] Mateau Giraldo Delicado. “Geostatistical Analysis of functional data”. In: (2009).

[11] Mateau Giraldo Delicado. “Ordinary kriging for function-valued spatial data”. In: (2010).

[12] Ognjen Grujic and Alessandra Menafoglio. *fdagstat, an R package*. R package version 1.0. 2017. URL: https://github.com/ogru/fdagstat.

[13] P. Z. Hadjipantelis et al. “Unifying amplitude and phase analysis: A compositional data approach to functional multivariate mixed-effects modeling

of Mandarin Chinese." In: *Journal of the American Statistical Association* (2015).

[14] Clara Happ et al. "A general framework for multivariate functional principal component analysis of amplitude and phase variation". In: *Stat. 2019;8:e220.* (2018).

[15] Jones Hauksson E., L. M. Hutton, and K. "The 1994 Northridge earthquake sequence in California: Seismological and tectonic aspects." In: *Journal of Geophysical Research: Solid Earth* (1995).

[16] A. Heinecke et al. "Petascale high order dynamic rupture earthquake simulations on heterogeneous supercomputers." In: *International Conference for High Performance Computing, Networking, Storage and Analysis* (2014).

[17] Laura M. Sangalli J. S. Marron James O. Ramsay and Anuj Srivastava. "Functional Data Analysis of Amplitude and Phase Variation". In: *Statistical Science* (2015).

[18] V. Pawlowsky-Glahn J.J. Egozcue J.L. Diaz-Barrero. "Hilbert Space of Probability Density Functions Based on Aitchison Geometry". In: *Research Gate* (2006).

[19] A. Menafoglio K. Hron et al. "Simplicial principal component analysis for density functions in Bayes spaces". In: *Computational Statistics and Data Analysis* (2016).

[20] Pawlowsky-Glahn Karl Gerard van den Boogaart Juan Jose Egozcue. "Bayes linear spaces". In: *Research Gate* (2010).

[21] A. Ramsay Kneip and J. O. "Combining registration and fitting for functional models." In: *Journal of the American Statistical Association* (2008).

[22] Simone Vantini Laura M. Sangalli Piercesare Secchi and Valeria Vitelli. "Functional clustering and alignment methods with applications". In: *Communications in Applied and Industrial Mathematics* (2010).

[23] S. Jung Lee and S. "Combined analysis of amplitude and phase variations in functional data." In: *arXiv:1603.01775* (2016).

[24] Alessandra Menafoglio. "Spatial statistics for distributional data in Bayes spaces: from object-oriented kriging to the analysis of warping functions". In: (2020).

[25] Christopher K. Wikle Noel Cressie. *Statistics for Spatio-Temporal Data.* 2011.

[26] C. Pelties et al. "Verification of an ADER-DG method for complex dynamic rupture problems." In: *Geoscientific Model Development* (2014).

[27] Muller Petersen A. "Functional data analysis for density functions by transformation to a Hilbert space." In: *The Annals of Statistics* (2016).

[28] M. Khodadadzadeh. R. Tolosana-Delgado H. Talebi. "On machine learning algorithms and compositional data." In: *Research Gate* (2019).

[29] J. O. Silverman andB. W. Ramsay. “Functional data analysis.” In: *New York: Springer.* (2005).

[30] Jorge Mateu Ramiraldo. “Kriging for Functional Data”. In: *Research Gate* (2013).

[31] A. Klassen Srivastava and E. P. “Functional and shape data analysis.” In: *New York: Springer.* (2016).

[32] Wu Srivastava A. et al. “Registration of functional data using fisher-rao metric.” In: *arXiv preprint arXiv:1103.3817* (2011).

[33] J. D. Tucker. “Functional component analysis and regression using elastic methods.” In: *Florida State University, Tallahassee, USA.* (2014).

[34] C. Uphoff et al. “Extreme scale multi-physics simulations of the tsunamigenic 2004 Sumatra megathrust earthquake.” In: *In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (2017).

[35] K. G. Van den Boogaart et al. “Bayes Hilbert spaces.” In: *Australian and New Zealand Journal of Statistics.* (2014).

[36] S. N. Wood et al. “Smoothing parameter and model selection for general smooth models.” In: *ournal of the American Statistical Association* (2013).

[37] S. Kurtek X. Guo and K. Bharath. “Variograms for spatial functional data with phase variation”. In: (2020).

Appendices

For our study, we have used some functions from Happ et al. (2018). In particular app-smooth.R, app-warp-MFPCA.R and app-seismology.R.

```
############## app_smooth.R #############
Parallelize smoothing
plan(multicore(workers = 30))
seissol = readRDS("../data/data_northridge.rds")

# Preprocess
data = seissol$Bodenbewegung %>%
  mutate(id = seissol$Seismogram:seissol$Simulation) %>%
  gather(key = "time", value = "y", -id) %>%
  mutate(time = as.numeric(str_extract(time, "[0-9\\.]+$")),
         y = zapsmall(y)) %>%
  nest(-id, .key = "boden")

smooth_one = function(d, k = 40, bs = "cr", m = 2, family = tw(),
optimizer = c("outer","newton"), sp = NULL,
return = c("fitted.values", "sp"), offset=NULL) {
  m = gam(y ~ s(time, k = k, bs = bs, m = m), offset = offset,
          family = family, optimizer = optimizer, sp = sp, data = d)
  return(m[return])
}

# Smooth ground velocity using smooth_one function
# future map allow you to run the map in parallel
plan(sequential)
groundVel_tw_smooth =
  future_map(data$boden,
           ~ cbind(.x, fit = smooth_one(.x, sp = 2.7)[["fitted.values"]]))
```

```
# Add id
data_groundVel_tw_smooth = groundVel_tw_smooth %>%
  imap(~cbind(id = .y, .x[, c(1,3)])) %>% #drop original data, add id
  bind_rows %>%
  spread(key = time, value = fit)

# Add smoothed data to data frame
seissol$groundVel_tw_smooth = data_groundVel_tw_smooth[,-1]

# Save data frame
saveRDS(seissol, "../data/seissol_tweedie_smooth.rds")

############## app_warp_MFPCA.R  #############
### Calculate SRVF warping & MFPCA ###

### Log-transform smoothed velocities
groundVel_smooth <- funData(argvals = x,
                       X = as.matrix(log1p(seissol$groundVel_tw_smooth[ind, ])))

### Calculate SRVF warping
SRVFwarp <- fdasrvf::time_warping(f = t(groundVel_smooth@X), time =
                                  groundVel_smooth@argvals[[1]],
                                  lambda = 0,# controls elasticity
                                  method = "mean", # Karcher mean
                                  showplot = FALSE,
                                  smooth_data = FALSE, #no box filter
                                  MaxItr = 500) # maximum number of iterations

# extract warping results in form of funData objects
h <- funData(groundVel_smooth@argvals,
min(groundVel_smooth@argvals[[1]]) +
t(SRVFwarp$gam)  * diff(range(groundVel_smooth@argvals)))
gamma <- invert.warps(h) # invert, for our notation
aligned <- funData(groundVel_smooth@argvals, t(SRVFwarp$fn))

### Calculate MFPCA

# Create multiFunData object
m <- multiFunData(clr.warp(gamma), aligned)

# univariate FPCA (use as given basis for faster calculation)
```

```
pca <- list(MFPCA::PACE(m[[1]]),
            nonSmoothFPCA(m[[2]]))
uniEx <- list(list(type = "given", functions = pca[[1]]$functions,
scores = pca[[1]]$scores),
              list(type = "given", functions = pca[[2]]$functions,
               scores = pca[[2]]$scores))

# subtract univariate means
mu <- multiFunData(pca[[1]]$mu, pca[[2]]$mu)
m <- m - mu

# Optimize weight of warping element
findWeight <- function(C, M){
  if(options()$verbose)
    cat("C: ", C, "\n")

  PCAm <- MFPCA::MFPCA(m, M = M,
                       uniExpansions = uniEx,
                       weights = c(C,1), fit = TRUE)

  # reconstruction
  xHat <- warp.funData(mu[[2]] + PCAm$fit[[2]],
   clrInv.warp(mu[[1]] + sqrt(C)*PCAm$fit[[1]]),
smooth = FALSE) # really smooth?

  mean(norm(xHat - groundVel_smooth))
}

bestApprox <- optimize(findWeight, interval = c(0,10), M = 10)

# save all
save(seissol, groundVel_smooth, SRVFwarp, h,
aligned, gamma, mu, m, bestApprox,
file = "../data/SRVF_seis_tw_smooth_rawPCA.Rdata")

############## app_seismology.R.  #############
### Application to Seismology Data (Northridge Earthquake Simulations) ###

### load required packages / utility functions
library(tidyfun) # for preprocessing
library(tidyverse) # for preprocessing
library(furrr) # for preprocessing
library(mgcv) # for preprocessing
library(funData) # for funData representation
library(MFPCA) # for MFPCA calculation
library(ggplot2) # for plotting
source("utils.R") # for warping utilities


### Preprocessing

if(file.exists("../data/seissol_tweedie_smooth.rds"))
{
  seissol <- readRDS("../data/seissol_tweedie_smooth.rds")
}  else {
  source("app_smooth.R")
}

# select only those functions that are close to the epicenter (< 40 km)
ind <- which(seissol$hypo.dist < 40000)

# raw data
x <- seq(0, 30, by=.5)
raw <- funData(argvals = x, X = as.matrix(log1p(seissol$Bodenbewegung[ind, ])))

### Calculate SRVF warping & MFPCA

if(file.exists("../data/SRVF_seis_tw_smooth_rawPCA.Rdata"))
{
  load("../data/SRVF_seis_tw_smooth_rawPCA.Rdata")
} else {
  source("app_warp_MFPCA.R")
}
   ### Final analysis and plotting
source("app_plot.R")
```

After that we have implemented the following code:

```
#### MODIFICO DATASET DIVIDENDO IN FUNZIONI ORIGINALI, ALLINEATE E WARPING ####
seissol2 <- seissol[which(seissol$hypo.dist < 40000),]
seissol2[[8]] <- aligned@X
seissol2[[9]] <- h@X
```

```
seissol2 <- seissol2[,-6]
seissol2$groundVel_tw_smooth <- groundVel_smooth@X
colnames(seissol2) <- c("Simulation","Seismogram","lon","lat","hypo.dist",
                                   "groundVel_smooth","Aligned","Warping")

# campiono 500 punti
p <- rep(1/1558,1558)
v <- sample(seissol2$hypo.dist, 500, replace = FALSE, prob = p)
k <- 1
ind2 <- rep(0,100)

for (j in 1:500){
  ind2[k] <- which(seissol2$hypo.dist==v[j])[1]
  ind2[k+1] <- which(seissol2$hypo.dist==v[j])[2]
  k <- k+2
}

data_finale <- seissol2[ind2,]

# scrivo latitudine, longitudine e distanza dall'ipocentro in km anzichetri
data_finale[,3] <- data_finale[,3]/1000
data_finale[,4] <- data_finale[,4]/1000
data_finale[,5] <- data_finale[,5]/1000
data_finale <- data_finale[which(data_finale$Simulation==102),]
groundVel <- funData(argvals = x, X = as.matrix((data_finale$groundVel_smooth)))
aligned <- funData(argvals = x, X = as.matrix((data_finale$Aligned)))
warping <- funData(argvals = x, X = as.matrix((data_finale$Warping)))

m <- rep(0,500)
m_a <- rep(0,500)
m_w <- rep(0,500)
for ( i in 1:500){
  m[i] <- mean(data_finale[,6][i,])
  m_a[i] <- mean(data_finale[,7][i,])
  m_w[i] <- mean(data_finale[,8][i,])
}
par(mfrow=c(1,3))
plot(data_finale[,5],m,col="blue", pch=19, xlab="Distance from Hypocentre",
         ylab="Ground Velocity",lwd=2)
plot(data_finale[,5],m_a,col="green", pch=19, xlab="Distance from Hypocentre",
         ylab="Aligned functions",lwd=2)
plot(data_finale[,5],m_w,col="yellow", pch=19, xlab="Distance from Hypocentre",
         ylab="Warping functions",lwd=2)

par(mfrow=c(1,2))
GV <- fdata(data_finale[,6], argvals = x)
dGV <- fdata.deriv(GV)
depth.mode(GV,draw=TRUE)
depth.modep(list(GV=GV,dGV=dGV),draw=T)
A <- fdata(data_finale[,7], argvals = x)
dA <- fdata.deriv(A)
depth.mode(A,draw=TRUE)
depth.modep(list(A=A,dA=dA),draw=T)
W <- fdata(data_finale[,8], argvals = x)
dW <- fdata.deriv(W)
depth.mode(W,draw=TRUE)
depth.modep(list(W=W,dW=dW),draw=T)

# cluster funzioni originali
D <- dist((data_finale$groundVel_smooth), 'euclidean')
Clustering <- kmeans(D, 2) # two clusters
mds <- cmdscale(D, eig=TRUE)
prVar <- round(100*mds$eig/sum(mds$eig),2)
par(mfrow=c(1,2))
plot(mds$points, main="Low dimensional Distance plot",
        xlab=paste(prVar[1],"% Variance"),
        ylab=paste(prVar[2],"% Variance"))
points(mds$points[Clustering$cluster==1, ], col="maroon2", pch=19)
points(mds$points[Clustering$cluster==2, ], col="darkblue", pch=19)
colours = rep("maroon2", length(Clustering$cluster))
colours[Clustering$cluster==2] <- "darkblue"
matplot(t(data_finale$groundVel_smooth), type="l", lty=1, col=colours,
          xlab="timepoints", ylab="Original data")

# cluster funzioni allineate
D_aligned <- dist((data_finale$Aligned), 'euclidean')
Clustering_aligned <- kmeans(D_aligned, 2) # two clusters
mds_aligned <- cmdscale(D_aligned, eig=TRUE)
prVar_aligned <- round(100*mds_aligned$eig/sum(mds_aligned$eig),2)
par(mfrow=c(1,2))
```

```
plot(mds_aligned$points, main="Low dimensional Distance plot",
        xlab=paste(prVar[1],"% Variance"),
        ylab=paste(prVar[2],"% Variance"))
points(mds_aligned$points[Clustering_aligned$cluster==1, ],col="maroon2", pch=19)
points(mds_aligned$points[Clustering_aligned$cluster==2, ],col="darkblue", pch=19)
colours = rep("maroon2", length(Clustering_aligned$cluster))
colours[Clustering_aligned$cluster==2] <- "darkblue"
matplot(t(data_finale$Aligned), type="l", lty=1, col=colours,
        xlab="timepoints", ylab="Aligned function")

# cluster funzioni di deformazione
warping_transformed <- clr.warp(warping)
D_warping <- dist((warping_transformed@X), 'euclidean')
Clustering_warping <- kmeans(D_warping, 2) # two clusters
mds_warping <- cmdscale(D_warping, eig=TRUE)
prVar_warping <- round(100*mds_warping$eig/sum(mds_warping$eig),2)
par(mfrow=c(1,2))
plot(mds_warping$points, main="Low dimensional Distance plot",
         xlab=paste(prVar[1],"% Variance"),
         ylab=paste(prVar[2],"% Variance"))
points(mds_warping$points[Clustering_warping$cluster==1, ],col="maroon2", pch=19)
points(mds_warping$points[Clustering_warping$cluster==2, ],col="darkblue", pch=19)
colours = rep("maroon2", length(Clustering_warping$cluster))
colours[Clustering_warping$cluster==2] <- "darkblue"
matplot(t(data_finale$Warping), type="l", lty=1, col=colours,
          xlab="timepoints", ylab="Warping function")


#### ANALISI GEOSTATISTICHE SULLE FUNZIONI ORIGINALI
train <- sample(nrow(data_finale), floor(0.7*nrow(data_finale)), replace=FALSE)

ones <- matrix(nrow = 350, ncol = 61)
for (i in 1:350)
  ones[i,] <- rep(1,61)

fun <- data.frame(Ground_Vel= t((log(data_finale[train,6]+ones))))
r <- rnorm(350, mean=0, sd=1)
g <- fstat(NULL, vName = "Ground Velocity",
            Coordinates = data_finale[train,3:4]+r*0.01,
            Functions = fun, scalar = FALSE)
```

```
g <- estimateDrift("~.", g, Intercept = TRUE)
g <- fvariogram("~lon+lat", g, Nlags = 100, LagMax = 50, ArgStep = 0.6,
                         comments=FALSE)
g <- fitVariograms(g, vgm(2, "Sph", 40, 0.2),fitRanges = TRUE, forceNugget = TRUE)
g <- addCovariance(g, type = 'omni')
g$model$omni$`Ground Velocity`$psill <- g$model$omni$`Ground Velocity`$psill/
                                                                sum(g$model$omni$`G
g <- addCovariance(g)
g <- estimateDrift("~.", g, .type = "GLS", Intercept = TRUE)
g <- fvariogram("~lon+lat", g, Nlags=100, LagMax = 40, ArgStep = 0.6,
                          useResidual = TRUE, comments=FALSE)
g <- fitVariograms(g, vgm(2, "Sph", 35, 0.2),fitRanges = TRUE, forceNugget = TRUE)
plotVariogram(g)

########## PREDICTION ##########
forecasts <- predictFstat(g, .newCoordinates = data_finale[-train,3:4]+r,
                                       .what = "Ground Velocity", .type = "UK")

ones_2 <- matrix(nrow = 150,ncol = 61)
for (i in 1:150)
  ones_2[i,] <- rep(1,61)

nuove <- matrix(nrow = 61, ncol = 150)
for (i in 1:61) {
  for (j in 1:150){
    if(forecasts$Forecast[i,j] < 0)
      nuove[i,j] <-  0
    else
      nuove[i,j] <- exp(forecasts$Forecast[i,j]) - 1
  }
}

par(mfrow=c(1,1))
matplot(nuove, type = "l",lty=1, col="darkblue", xlab="Timepoints",
             ylab="Ground Velocity")
matplot(t((data_finale[-train,6])), type="l",col="maroon2",add = TRUE)
legend("topleft", c("True", "Forecasted"), col=c("maroon2","darkblue"),
             lty=1, lwd=1)
```

```
#### ANALISI GEOSTATISTICHE SULLE FUNZIONI ALLINEATE
fun_aligned <- data.frame(Aligned= t(log((data_finale[train,7]+ones))))
r <- rnorm(350, mean=0, sd=1)
g_aligned <- fstat(NULL, vName = "Aligned",
                            Coordinates = data_finale[train,3:4]+r*0.01,
                            Functions = fun_aligned, scalar = FALSE)
# Drift Estimation
g_aligned <- estimateDrift("~lon+lat", g_aligned, Intercept = TRUE)
# Trace Variogram Estimation
g_aligned <- fvariogram("~lon+lat", g_aligned, Nlags = 100,
                                          LagMax = 40, ArgStep = 1.3,
                                          comments=FALSE)
plotVariogram(g_aligned)
# non so se i Mat, Sph o Exp
g_aligned <- fitVariograms(g_aligned, vgm(2.5, "Sph", 40, 0.3),
                                          fitRanges = TRUE,  forceNugget = TRUE)
g_aligned <- addCovariance(g_aligned, type = 'omni')
g_aligned$model$omni$Aligned$psill <- g_aligned$model$omni$Aligned$psill/      su
g_aligned <- addCovariance(g_aligned)
g_aligned <- estimateDrift("~.", g_aligned, .type = "GLS", Intercept = TRUE)
g_aligned <- fvariogram("~lon+lat", g_aligned, Nlags=100, LagMax = 40, ArgStep = 1
useResidual = TRUE, comments=FALSE)
g_aligned <- fitVariograms(g_aligned,  vgm(1.5, "Sph", 30, 0.4), fitRanges = TRUE,
forceNugget = TRUE)
plotVariogram(g_aligned)

########## PREDICTION ##########
forecasts_aligned <- predictFstat(g_aligned,
.newCoordinates = data_finale[-train,3:4]+r,
.what = "Aligned", .type = "UK")

par(mfrow=c(1,1))
matplot(exp(forecasts_aligned$Forecast)-1, type = "l",lty=1, col="darkblue",
xlab="Timepoints", ylab="Aligned")
matplot(t(((data_finale[-train,7]))), type="l",col="maroon2",add=TRUE)
legend('topleft', c("True", "Forecasted"), col=c("maroon2","darkblue"),
lty=1, lwd=1)


#### ANALISI GEOSTATISTICHE SULLE FUNZIONI DI DEFORMAZIONE
```

```
# trasformo le funzioni con clr
warping_transformed <- clr.warp(warping)
fun_warping <- data.frame(Warping= t((warping_transformed@X[train,])))
r <- rnorm(350, mean=0, sd=1)
g_warping <- fstat(NULL, vName = "Warping",
Coordinates = data_finale[train,3:4]+r*0.01,
Functions = fun_warping, scalar = FALSE)
# Drift Estimation
g_warping <- estimateDrift("~lon+lat", g_warping, Intercept = TRUE)
# Trace Variogram Estimation
g_warping <- fvariogram("~lon+lat", g_warping, Nlags = 100, LagMax = 40, ArgStep =
comments=FALSE)
plotVariogram(g_warping)
# non so se i Mat, Sph o Exp
g_warping <- fitVariograms(g_warping, vgm(1.5, "Exp",30, 0.5),fitRanges = TRUE,
forceNugget = TRUE)
plotVariogram(g_warping)
g_warping <- addCovariance(g_warping, type = 'omni')
g_warping$model$omni$Warping$psill <- g_warping$model$omni$Warping$psill/
                                                                      sum(g_warping
g_warping <- addCovariance(g_warping)
g_warping <- estimateDrift("~.", g_warping, .type = "GLS", Intercept = TRUE)
g_warping <- fvariogram("~lon+lat", g_warping, Nlags = 100, LagMax = 40, ArgStep =
                                          useResidual = TRUE, comments=FALSE)
g_warping <- fitVariograms(g_warping, vgm(1.5, "Exp",30, 0.4), fitSills = TRUE,
                                            fitRanges = TRUE, forceNugget = TRUE)
plotVariogram(g_warping)
########## PREDICTION ##########
forecasts_warping <- predictFstat(g_warping,
                                   .newCoordinates = data_finale[-train,3:4]+r*0.0
                                    .what = "Warping", .type = "UK")

matplot(forecasts_warping$Forecast, type = "l",lty=1, col="darkblue",
             xlab="Timepoints", ylab="Warping")
matplot(t(warping_transformed@X[-train,]), type="l",col="maroon2",add=TRUE)
legend('bottomright', c("True", "Forecasted"), col=c("maroon2","darkblue"),
              lty=1, lwd=1)

l <- funData(argvals = x, X=t(forecasts_warping$Forecast))
l2 <- funData(argvals = x, X=(warping_transformed@X[-train,]))
```

```
matplot(t(clrInv.warp(l)@X), type = "l",lty=1, col="darkblue",
               xlab="Timepoints", ylab="Warping")
matplot(t(clrInv.warp(l2)@X), type="l",col="maroon2",add=TRUE)
legend('bottomright', c("True", "Forecasted"), col=c("maroon2","darkblue"),
               lty=1, lwd=1)


forecast_warping_fundata <- funData(argvals = x,
                                                                X=t(forecasts_warping$
forecasts_warping_trasformed <- clrInv.warp(forecast_warping_fundata)

par(mfrow=c(1,3))
matplot(exp(t(ricostruite@X))-1,type = "l",lty=1, col="darkblue",
               xlab="Timepoints", ylab="Prediction by decoupled functions",ylim=c(
matplot(nuove, type = "l",lty=1, col="darkgreen",
               xlab="Timepoints", ylab="Prediction by original functions",ylim=c(0
matplot(t((data_finale[-train,6])), type="l",col="maroon2",
               xlab="Timepoints", ylab="Ground Velocity",ylim=c(0,2.5))

or <- funData(argvals = x, X=t(forecasts$Forecast))
warp <- warp.funData(or, forecasts_warping_trasformed , smooth = FALSE)
se_ricostruito <- (t(warpate@X)-forecasts_aligned$Forecast)^2

par(mfrow=c(1,2))
matplot(se,type = "l",lty=1, col="maroon2",
               xlab="Timepoints", ylab="Errore originale",ylim=c(0,0.8))
matplot(se_ricostruito_2, type = "l",lty=1, col="darkblue",
               xlab="Timepoints", ylab="Errore ricostruito",ylim=c(0,0.8))

err_originale <- funData(argvals = x, X=t(se))
err_ricostruito <- funData(argvals = x, X=t(se_ricostruito_2))
e_or <- integrate(err_originale)
e_ric <- integrate(err_ricostruito)

par(mfrow=c(1,1))
boxplot(e_or, e_ric, names=c("Errori originali","Errori ricostruiti"), col=c("maro

d <- data.frame(e_or,e_ric)
b <- SpatialPointsDataFrame(data_finale[-train,c(3,4)],d)
par(mfrow=c(1,2))
```

```
bubble(b, "e_or", col=c("#00ff0088", "#00ff0088"), main = "Errore originale")
bubble(b, "e_ric", col=c("#00ff0088", "#00ff0088"), main = "Errore ricostruito")
```

Then we have applied this procedure 100 times, changing the test and training set. The functions for the clr transformation that we have used derives from another file from Happ et al.(2018):

```
######## utils.R ########
### SRVF transformation approach ###

#' SRVF transformation of warping functions
#'
#' Calculates the SRVF transformation of a set of warping functions
#' \eqn{\gamma}{gamma}: \deqn{\sqrt{\gamma'},}{sqrt(gamma'),} where
#' \eqn{\gamma'}{gamma'} denotes the first derivative of
#' \eqn{\gamma}{gamma}.
#'
#' @param gamma A \code{funData} object containing the warping functions.
#'
#' @return The SRVF transformation of all functions in \code{gamma}, again
#'   as a \code{funData} object.
srvf <- function(gamma)
{
  return(sqrt(diff.funData(gamma)))
}

#' Tangent space transformation of SRVFs
#'
#' This function calculates shooting vectors for SRVFs with respect to a
#' given mean function.
#'
#' @param srvf A \code{funData} object containing the SRVFs.
#' @param mu A \code{funData} object containing the mean function for the
#'   tangent space. Defaults to the constant function, which is associated
#'   with identity warping.
#'
#' @return A \code{funData} object containing the shooting vectors.
tangent <- function(srvf, mu = funData(argvals(srvf),
matrix(rep(1, nObsPoints(srvf)), nrow = 1)))
{
  if(nObs(mu) != 1)
```

```
    stop("The mean function mu must contain a single function.")

  eta <- norm(mu, squared = TRUE)

  tmp <- scalarProduct(srvf, mu) / eta

  # tmp is guaranteed to lie between 0 and 1, all deviations have numerical reasons
  tmp[tmp < 0] <- 0
  tmp[tmp >= 1] <- 1 - sqrt(.Machine$double.eps)

  # angles
  theta <- acos(tmp)

  v <- srvf * 0 # initialize return object
  for(i in 1:nObs(srvf))
    v@X[i,] <- theta[i]/(sqrt(eta) * sin(theta[i]))* srvf@X[i,] - cos(theta[i]) * m

  return(v)
}

#' Tangent space transformation for warping functions
#'
#' Combine SRVF transformation with projection on a tangent space for
#' warping functions.
#'
#' @param gamma A \code{funData} object containing the warping functions.
#' @param mu A \code{funData} object containing the mean function for the
#'   tangent space in the warping space. Defaults to identity warping.
#'
#' @return A \code{funData} object containing the shooting vectors.
tangent.warp <- function(gamma, mu = funData(argvals(gamma),
matrix(argvals(gamma)[[1]], nrow = 1)))
{
  return(tangent(srvf(gamma) , srvf(mu)))
}

#' Map tangent space vectors to SRVFs
#'
#' This function maps shooting vectors in a tangent space to their SRVFs.
#'
```

```
#' @param v A \code{funData} object containing the shooting vectors.
#' @param mu A \code{funData} object containing the mean function for the
#'   tangent space. Defaults to the constant function, which is associated
#'   with identity warping.
#'
#' @return A \code{funData} object containing the srvfs.
tangentInv <- function(v, mu = funData(argvals(v), matrix(rep(1, nObsPoints(v)), n
{
  if(nObs(mu) != 1)
    stop("The mean function mu must contain a single function.")

  eta <- norm(mu)

  srvf <- v*0 # initialize result object
  nv <- norm(v, squared = FALSE)

  for(i in 1:nObs(v))
    srvf@X[i,]   <- sqrt(eta) * sin(nv[i])/nv[i]*v@X[i,] + cos(nv[i]) * mu@X[1,]

  return(srvf)
}

#' Inverse SRVF transformation to warping functions
#'
#' Calculates the inverse SRVF transformation of a set of SRVFs.
#'
#' @param srvf A \code{funData} object containing the SRVFs.
#'
#' @return The inverse SRVF transformation of all functions in \code{srvf}, again
#'   as a \code{funData} object.
srvfInv <- function(srvf){
  return(cumInt.funData(srvf^2))
}

#' Inverse tangent space transformation for warping functions
#'
#' Combine projection from a tangent space to SRVF and inverse SRVF
#' transformation to warping functions.
#'
#' @param v A \code{funData} object containing the functions in tangent space.
#' @param mu A \code{funData} object containing the mean function for the
#'   tangent space in the warping space. Defaults to identity warping.
#'
#' @return A \code{funData} object containing the warping functions.
tangentInv.warp <- function(v, mu = funData(argvals(v), matrix(argvals(v)[[1]], nr
{
  return(srvfInv(tangentInv(v , srvf(mu))))
}


### clr transformation approach ###

#' Centred log-ratio transform for warping functions
#'
#' @section Warning: The function does not check if the argument
# is really a warping function!
#'
#' @param f A funData object that represents a warping function.
#'
#' @result A funData object, which is the cenred log-ratio transform (CLR)
#applied to the derivative of f.
clr.warp <- function(f)
{
  clr(diff.funData(f))
}


#' Centred log-ratio transform
#'
#' Centred-log ratio transform for functional data objects.
#The domain needs not necessarily be [0,1].
#'
#' @section Warning: it is not checked if g is positive and integrable!
#'
#' @param g A funData object
#'
#' @return The CLR of g
clr <- function(g)
{
  if(any(g@X < 0, na.rm = TRUE))
  {
    warning("Negative values found, set to NA")
    g@X[g@X < 0] <- NA
   intG <- integrate(as.irregFunData(log(g))) # do integration on irregFunData obj
  }
  else
  {
    intG <- integrate(log(g))
  }

  return(log(g) - 1/diff(range(g@argvals[[1]])) * intG)
}

#' Inverse centred log-ratio transform
#'
#' @param g A funData object
#'
#' @return A funData object, that corresponds to the inverse CLR of g.
clrInv <- function(g)
{
  # map from L^2 to B^2 (densities)
  return(exp(g)/integrate(exp(g)))
}

#' Inverse CLR for warping functions
#'
#' The function calculates an inverse CLR for a funData object g (which gives
#' a density) and integrates it to a warping function. In particular, if the
#' density maps from [a,b], the resulting warping function h has h(a) = a and
#' h(b) = b.
#'
#' @param g A funData object
#'
#' @return A funData object corresponding to the associated warping function.
clrInv.warp <- function(g)
{
  h <- clrInv(g)
  # integrate densities to obtain warping functions
  # warping functions h fulfill h(a) = a, h(b) = b
  return(min(h@argvals[[1]]) + diff(range(h@argvals[[1]]))  * cumInt.funData(h))
```

```
}


### Petersen and Mueller (2016) transformations ###

#' Log-hazard-transformation
#'
#' Log-hazard-transformation for density functions as defined in Petersen
#' & Mueller (2016), including differentiation of warping functions.
#'
#' @param g A \code{funData} object containing the warping functions.
#' @param delta Cutoff for calculating the hazard function.
#'   Defaults to \code{0.05}.
#'
#' @return A \code{funData} object containing the transformed functions.
LH.warp <- function(g, delta = 0.05)
{
  # extract argvals
  argvals <- g@argvals[[1]]
  # renormalize to cdf, mapping to [0,1]:
  gN <- (g - min(argvals)) / diff(range(argvals))

  # calculate density
  f <- diff.funData(gN)

  gN@X[, argvals > max(argvals) - delta * diff(range(argvals))] <- NA

  # trafo
  v <- log(f / (1 - gN))

  return(v)
}


#' Inverse Log-hazard-transformation
#'
#' Inverse log-hazard-transformation for density functions as defined in Petersen
#' & Mueller (2016), including differentiation of warping functions.
#'
#' @param v A \code{funData} object containing the functions in L2.
#' @param delta Cutoff for calculating the hazard function.
#'   Defaults to \code{0.05}.
#'
#' @return A \code{funData} object containing the warping functions.
LHInv.warp <- function(v, delta = 0.05)
{
  argvals <- v@argvals[[1]]
  thresh <- max(argvals) - delta * diff(range(argvals))
  v1 <- extractObs(v, argvals = argvals[argvals < thresh])

  # density level
  f1 <- exp(v1 - cumInt.funData(exp(v1)))
  f2 <- 1/delta * exp(-1*integrate(exp(v1)))

  f <- funData(argvals = argvals, cbind(f1@X, t(sapply(f2, rep,
   each = length(which(argvals >= thresh))))))

  # warping level
  g <- min(argvals) + cumInt.funData(f) * diff(range(argvals))

  return(g)
}

#' Log-quantile-transformation
#'
#' Log-quantile-transformation for density functions as defined in
#' Petersen & Mueller (2016), including differentiation of warping
#' functions.
#'
#' @param g A \code{funData} object containing the warping functions.
#'
#' @return A \code{funData} object containing the transformed functions.
LQ.warp <- function(g)
{
  # quantile function
  Q = invert.funData(g)

  # renormalize to cdf, mapping to [0,1]:
  gN <- (g - min(g@argvals[[1]])) / diff(range(g@argvals))
  # calculate density
  f <- diff.funData(gN)

  # trafo
  v <- -1 * log(warp.funData(f, Q, smooth = TRUE))

  return(v)
}


#' Inverse Log-quantile-transformation
#'
#' Inverse log-quantile-transformation for density functions as defined in
#' Petersen & Mueller (2016), including retransformation to warping
#' functions.
#'
#' @param v A \code{funData} object containing the functions in L2.
#'
#' @return A \code{funData} object containing the warping functions.
LQInv.warp <- function(v)
{
  # Quantile function
  Q <- cumInt.funData(exp(v)) / integrate(exp(v))

  # Warpign function is the inverse of Q, appropriately scaled
  g <- min(v@argvals[[1]]) + invert.funData(Q) * diff(range(v@argvals))

  return(g)
}



### Others ###

#' Discrete differentiation for funData objects
#'
#' For the inner points (2:(nObsPoints - 1)), central differentiation is
#' used, which borrows information from the preceding (i-1) as well as the
#' following (i+1) value for calculating the gradient. For the boundary
#' values (1, nObsPoints(f)) only one neighbouring value is used: Forward
#' differentiation for the left bound, backward differentiation for the right
#' bound.
#'
#' @param f A funData object. Must have a one-dimensional domain,
#otherwise an error is thrown.
#'
#' @return Another funData object, which corresponds to the derivative of f.
diff.funData <- function(f)
{
  if(dimSupp(f) > 1)
    stop("Implementation is only for one-dimensional domains.")

  nP <- nObsPoints(f)
  x <- argvals(f)[[1]]

  g <- array(NA, dim = dim(f@X)) # initialize g matrix for differentiation

  # left bound: forward differentiation
  g[,1] <- (f@X[,2] - f@X[,1]) / truncX(x[2] - x[1])

  # right bound: backward differentiation
  g[,nP] <- (f@X[,nP] - f@X[,nP - 1]) / truncX(x[nP] - x[nP - 1])

  # all other points: central differentiation
  g[,2:(nP-1)] <- (f@X[,3:nP] - f@X[,1:(nP-2)]) / truncX(x[3:nP] - x[1:(nP-2)])

  return(funData(x, g))
}

#' Cumulative integration of funData objects
#'
#' Calculate the integral over a funData object from zero to all points in
#' the observation grid. For numerical stability, the calculation is made
#' backward for lower values and forward for higher values. This avoids
#' integrating over only a few points.
#'
#' @param h A funData object. Must have a one-dimensional domain, otherwise
#'   an error is thrown.
#'
#' @return A funData object, containing the integrated values for each
#'   observation point.
cumInt.funData <- function(h)
{
  if(dimSupp(h) > 1)
    stop("Implementation is only for one-dimensional domains.")

  x <- argvals(h)[[1]]
  intH <- integrate(h)

  # split integration into two parts to guarantee stability at the boundaries
  n <- max(which(x < median(x)))

  cumInt <- lapply(1:nObsPoints(h), function(ind){
    if(ind <= n)
      intH - integrate(extractObs(h, argvals = x[ind:nObsPoints(h)]))
    else
      integrate(extractObs(h, argvals = x[1:ind]))
  })

  # cbind & list can handle all numbers of observations (incl. 1)
  return(funData(h@argvals, do.call("cbind", cumInt)))
}


#' Inverse of funData
#'
#' This function returns the inverse of a funData object
#' representing a set of functions on a common grid. The result is again a funData
#' object. The method is based on the smooth.spline function in R (stats)
#'
#' @param f The functions to be inverted, passed as a funData object
#' @param ... Options to be passed to smooth.spline
invert.funData <- function(f,...)
{
  argvals <- f@argvals[[1]]

  return(funData(argvals = argvals,
          X = t(apply(f@X, 1, function(z){mgcv::predict.gam(mgcv::gam(argvals ~ s(
           newdata = data.frame(z = argvals))}))))
}
```

```
#' Inverse of warping function
#'
#' This function returns the inverse of a funData object representing a
#' set of warping functions on a common grid. The result is again a
#' funData object and respects the monotonicity of the input functions.
#' The method is based on the splinefun function in R (stats)
#'
#' @param f The functions to be inverted, passed as a funData object
#' @param ... Options to be passed to splinefun
invert.warps <- function(f,...)
{
  argvals <- f@argvals[[1]]

  return(funData(argvals = argvals,
                 X = t(apply(f@X, 1, function(z){
                   splinefun(x = z, y = argvals, method = "monoH.FC")(argvals)
                 }))))
}


#' Discrete warping
#'
#' The calculate the warping of a funData object represented by a funData
#object containing warping functions.
#'
#' @section Warning: The function does not check if all elements of w
#are correct warping functions.
#'
#' @param f The funData object to be warped
#' @param w The funData object containing the warping functions.
#' @param smooth Logical, should the warped functions be smoothed (via gam?)
#'
#' @return A funData object containing the warped functions.
warp.funData <- function(f, w, smooth = FALSE)
{
  # check if functions have the same domain
  if(! all.equal(range(argvals(f)), range(argvals(w))))
    stop("f and w need to have the same domain!")
  res <- matrix(NA, nrow = nObs(f), ncol = length(w@argvals[[1]]))
  # for all functions in f
  for(i in 1:nObs(f))
  {
    allDiff <- outer(w@X[i,], f@argvals[[1]], function(x,y){abs(x-y)})
    p <- f@X[i,apply(allDiff, 1, which.min)] # choose nearest x- value

    if(smooth)
    {
      xi <- argvals(w)[[1]]
      GAM <- mgcv::gam(p ~ s(xi, bs = "ps"))
      p <- predict(GAM)
    }

    res[i,] <- p
  }

  return(funData(w@argvals,res))
}
```

Ringraziamenti

É giunto ora il momento dei ringraziamenti.
Anzitutto voglio ringraziare la mia relatrice Alessandra Menafoglio, per l'infinita pazienza, per la disponibilitá, il sorriso e la passione che mi ha trasmesso.
Voglio poi ringraziare i miei genitori per avermi sempre dato la possibilitá di studiare ció che mi piaceva, spronandomi sempre e credendo in me. Ringraziarli per avermi insegnato l'importanza e la bellezza dello studio.
Grazie mamma e papá per tutto questo, e soprattutto per essermi sempre stati accanto.
Un grazie speciale va anche a mia sorella e a suo marito, da sempre un esempio per me.
Un esempio nella vita scolastica e un esempio nella vita quotidiana.
Grazie Angelo e Allegra per essere stati cosí coraggiosi da aver spinto anche me ad esserlo.
Grazie ai miei nipoti, Elia Agostino e Margherita. Mi hanno insegnato che ogni tanto lo studio é giusto metterlo da parte per non perdersi momenti speciali.
Un grazie anche alla famiglia di mio marito e in particolare a Chiara, grazie perché so che é anche per le vostre preghiere che sono riuscita ad arrivare fin qui.
Grazie a tutti i miei amici, a quelli che mi sono stati accanto in questo percorso e che hanno subito le infinite buche perché "non posso mi spiace devo studiare", grazie di avermi sopportata.
Un grazie speciale ai miei compagni di avventura. Grazie per aver reso le lezioni piú divertenti e leggere, non sarebbe stato tanto bello questo percorso senza di voi.
Infine i ringraziamenti piú importanti.
Grazie alla mia roccia, al mio compagno di vita, al mio migliore amico, a mio marito.
Grazie Tommaso, per non aver mai avuto dubbi sul fatto che ce l'avrei fatta (anche quando io li avevo). Grazie per aver sopportato i miei isterismi e i miei sbalzi d'umore a causa degli esami e dello stress. Grazie per essermi sempre stato accanto, a volte in silenzio ad ascoltare i miei sfoghi e a volte trovando le parole giuste per farmi tornare serena.

Grazie per tutto amore mio, sei la mia casa.
Grazie a nostro figlio Sebastiano che mi ha fatto compagnia in pancia nelle ultime due sessioni, grazie per avermi motivavo ancora di piú. Grazie anche per essere un bambino incredibilmente bravo, non sarei mai riuscita a scrivere la tesi altrimenti.
Grazie poi al mio angelo custode d'onore, a Davide, il mio primo figlio. Grazie perché tu da lassú mi hai ascoltata piú di tutti nei momenti di difficoltá.
Il grazie piú grande di tutti peró va a Dio, per avermi aiutato incredibilmente durante questo percorso, per aver sopportato le mie infinite preghiere e per avermi donato queste persone speciali che giorno dopo giorno scelgono di starmi accanto.
Sono infinitamente grata per ogni cosa.