Functional Data Analysis for Earth Observation

Julian Austin

Department of Mathematics, Statistics and Physics.
Newcastle University

This dissertation is submitted for the degree of
Doctor of Philosophy

November 2022

Abstract

Earth observation data, that is data observed over the surface of the earth, is often characterised by its spatial and temporal dependency. Such datasets are being collected more frequently and over larger spatial domains as remote sensing and in-situ collection methodologies become more sophisticated. However, they often include large amounts of missing observations. There is a high demand for models which can help interpret and interpolate to aid in the use of these datasets for a vast array of disciplines. Often the most challenging aspect of such data is how to interpolate missing observations. In this work, we consider such datasets from a functional data perspective. In particular, we focus on methods which can help explain the datasets variation in a parsimonious way whilst maintaining predictive accuracy for missing observations.

We begin by discussing the current methods available to earth observation datasets from both a spatio-temporal and functional data perspective. Following this, we introduce an interim functional time series model, based on a functional data decomposition which considers the spatial dimension as our functional domain. We discuss the consequences of taking this approach from a practical perspective.

Finally, we develop a novel framework which treats the temporal dimension as the functional domain. We maintain parsimony by basing this model on the main modes of variation using a functional principal components analysis and incorporate spatial dependency between functional observation using a structured Gaussian process. We present the validity of this methodology under spatial correlation of the observed data and evidence the ability of this framework using various spatial dependency models on both a simulation and real world study. We show that such a model performs well on sparsely observed datasets and also highlight the approaches used to make the model applicable to large datasets.

Acknowledgements

I owe thanks to many people for helping me to this point. I would like to express my gratitude to my supervisors; Robin Henderson, Jian Qing Shi, and Zhenhong Li. All of your guidance, support, and knowledge throughout has helped me considerably. Zhenhong; thank you for introducing me to the world of earth observation. Jian; many thanks for your immeasurable support even through challenging time differences. Robin; special thanks for stepping in to supervise for the last couple of years, you made what could be a challenging situation a smooth process.

I would like to express my thanks to the Centre for Doctoral Training in Cloud Computing for Big Data for their support. I had the pleasure to work with many brilliant teaching and administrative colleagues, and fellow PhD students. Thanks for your unwavering commitment to research, interesting talks, and friendships. Thanks should also go to the Engineering and Physical Sciences Research Council for my funding, [grant number EP/L015358/1].

To my family. Thank you for your support through the years of this work; and for all the years that came before it. In particular, mum and dad, your kindness and love has made this possible.

To Hannah, thank you for being there with me; every step of the way. For putting up with the ups and downs of the whole process. At times when I struggled you were there to help me. I could not have undertaken this journey without you and I am deeply grateful for everything you have done.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Earth Observation

Many areas of science produce data on both a spatial and temporal scale. Take for example, the production of Earth Observation data. Earth Observation (EO) is the collection of information on the state of a physical, chemical or biological system of the planet. Typically EO data is acquired through some form of remote sensing in addition to perhaps some in-situ measurements. EO data is acquired to study a process either over a large area of land, a large time horizon, or both. For example, such EO studies include; land usage change in wetland environments in southern Spain, [58], crop production in the Netherlands, [44], and land deformation of the Tuscany region over a two year time period, [64]. In each case there is significant spatial and temporal dependency that is to be considered in the observed processes. For example, Raspini et al. use the temporal dependency in ground deformation signals to highlight areas of significant change in movement, [64]. They combine this with spatial maps to provide a monitoring bulletin for their area of interest. Of course to provide actionable insights from EO data requires an understanding of both the spatial and temporal dependency. As such, models that can handle both forms of dependency whilst maintaining parsimony are desired in the EO community.

An area where EO data is prominent is climatology. Climatology is the study of the atmosphere and weather patterns over time. In this case spatial and temporal dependency in the EO data used in various climatology studies is fairly evident. For example, consider the Community Earth System Model (CESM), [43], produced by the National Centre for Atmospheric Research (NCAR). Such a model provides simulations of various aspects of the Earth's climates for past, present and future time points. See Section 2.1 for a more detailed description of such a dataset. Figure 1.1 provides an example of a subset of the CESM data focusing on the temperature in Kelvin (K) over the globe. As expected we can see a clear temporal pattern emerging in various regions which relates to the seasons. For example, there is clear evidence for temporal correlation over the polar regions due to the gradual increase then reduction in temperature we see over these regions. Figure 1.1 also highlights clear spatial patterns with temperature varying as we move around the globe. One particularly noticeable pattern is the localised change in temperature over North Africa between the months of May to September. For another example, consider the transition from sea to land. The change in temperature as we move from sea to land is often more abrupt than the same change in temperature we may see when moving the same distance over land. This is possibly evidence for the existence of a complex spatial and temporal process driving such a variable. Understanding such a complex process motivates a model which must take into account both spatial and temporal correlations jointly.

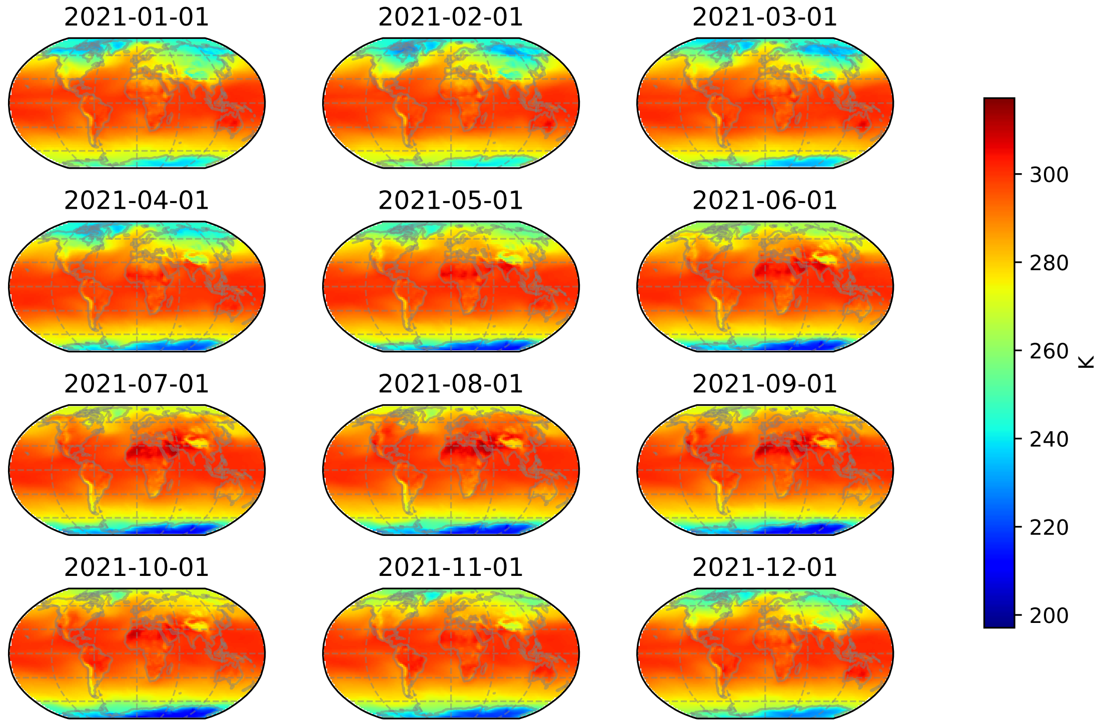


Fig. 1.1 Monthly average temperature over the globe in Kelvin (K) from a single simulation from CESM-LE. The figure illustrates both the temporal and spatial dependency that is observed in temperature across the globe. The figure is projected to the Robinson projection for illustration purposes.

The CESM data is generated through simulations of a complex model of the Earth (see Section 2.1 for more detail). However, EO data is also becoming more frequently generated through remote sensing. Remote sensing is the monitoring of a physical characteristic of an area through measuring its reflected and emitted radiation at a distance. Space borne remote sensing, typically achieved through the use of satellite based sensors, is becoming more prominent as a source of EO data and in particular as a source of climatology data. The three studies above, [58, 64, 44], use space borne remote sensing to observe their process of interest. This is largely due to the increase in satellites launched which have been designed to capture various processes of the earth. Figure 1.2 highlights the rise in availability of a single type of remote sensing satellite. One particularly prominent remote sensing system is the European Space Agency's Sentinel Constellation, [4]. The Sentinel constellation of satellites provides a wide range of remote sensing sensors which are easily accessible. The constellation provides capabilities to capture various physical characteristics through the many forms of sensors equipped to its satellites. These include Synthetic Aperture Radar (SAR), optical and multispectral sensors. For example, European Space Agency's Sentinel-4 from the Sentinel constellation, [4], provides observation dedicated to air quality monitoring. As such the Sentinel constellation has been widely used in EO studies; the three studies above, [58, 44, 64] all utilise the Sentinel 1 SAR sensors for their observation source.

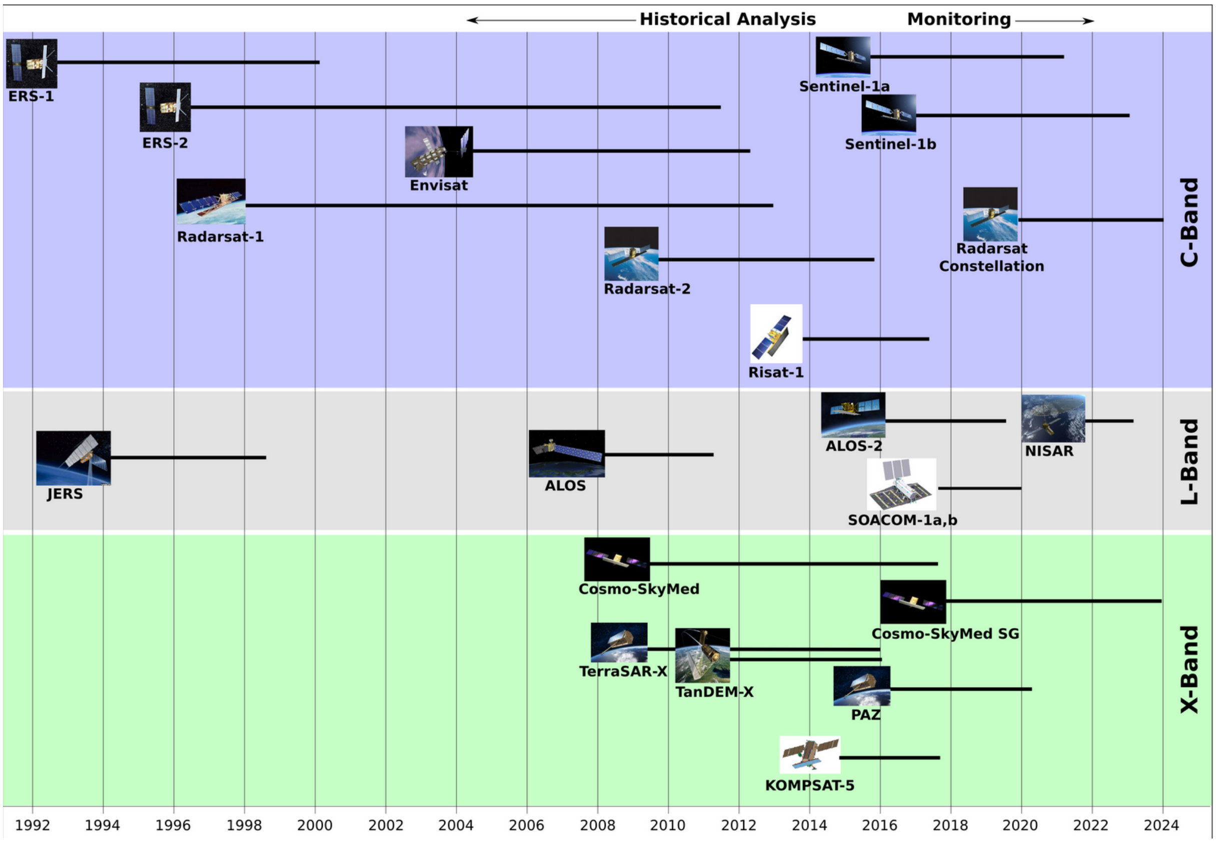


Fig. 1.2 A timeline of major satellite launches and operating periods for EO missions using SAR based sensors. This highlights the rapid rise in availability of such remote sensing capabilities, driven by the demand for observations to cover large spatial and temporal scales in areas such a climatology. The C, L, X band refers to the type of SAR sensor equipped to the satellite which are used in different applications. See [60] for details.

A prominent focus of the Sentinel satellite constellation is to provide repeated observations at relatively high frequency, [4]. This is in response to the rising demand for monitoring EO processes over time. High frequency revisit times have been made possible by the development of remote sensing technologies. For example, the Sentinel 1 satellite constellation can provide revisit times of approximately five days for areas of Europe. Such short revisit times are advantageous as they give higher temporal resolution and thus studies can incorporate this additional information. For example, Raspini et al. utilise this in their study of land deformation change to identify anomalous regions, [64]. The increasing availability of high temporal frequency EO data such as those provided by the CESM-LE dataset or the Sentinel satellite constellation thus drives a demand for statistical models which can handle both high resolution spatial and temporal dependency.

Earth observation data, such as those provided by the CESM model or through remote sensing, have an inherent spatial and temporal dependency. That is to say the underlying process driving both remotely sensed observations of the Earth and the CESM simulations will vary over the globe and also will be driven by the state of the system at prior time points. That is not to say the process is the same for both but rather that there is a commonality in that they could both be considered spatio-temporal processes. In addition there are more concrete similarities in the collection of such EO data. Typically, EO data are described on a lattice of points over space which is usually regular. This usually relates to the pixels of an image over the area of interest. Such a lattice is represented through a geodetic coordinate system which grounds a datum to a real world location. Finally, EO data typically have repeated observations through time over the same space. This usually relates to the repeated imaging of the same area of interest at multiple points in time.

The description of such spatio-temporal processes is well studied in statistics and a large amount of effort has been used to develop various models to suit them. A well known monograph which deals with such processes is that written by Cressie and Wikle, [12]. The monograph details various forms of spatio-temporal processes and typically focuses on the extension of spatial methods to incorporate the additional temporal dimension. We discuss these methods in more detail in Section 1.3. Of particular importance in these methods is that temporal and spatial dimensions are treated distinctly as they are inherently different in the physical process. For example, one could consider a spatial point influencing its neighbours in all directions however a temporal point reasonably shouldn't influence its past. As such there is often a distinction in the method used to model the temporal and spatial aspects of the physical process.

An area of statistics which is often used to model data with temporal dependency is Functional Data Analysis (FDA). FDA is typically applied to analyse data which vary over a continuum, [63]. Time is one such continuum. EO data with high frequency temporal observations are therefore suitable candidates for FDA models. FDA is a relatively new branch of statistics and as such few studies have been presented which use FDA techniques on EO data. Liu et al. consider FDA techniques on periodic EO data, [49]. Similarly Hooker et al. consider FDA techniques to model the Harvard modified vegetation index sourced from EO data. In both the above studies they consider EO data as a collection of functional observations indexed by space where each functional observation represents a trajectory over time.

The monograph of Ramsay and Silverman provides a comprehensive introduction to the themes of FDA, [63]. FDA are often intuitive since viewing responses as being trajectories from an unknown smooth random function in some contexts closely matches the actual data generating process compared to a multivariate analysis. The use of such techniques therefore could be helpful in modelling such high frequency EO data in conjunction with the multivariate methods discussed by Cressie and Wikle in [12]. However, focus in the FDA literature to date has primarily revolved around independently observed functional data. This is typically not the case in our motivating case of EO data where there is often obvious spatial dependency. Thus there is a need to describe functional data models which incorporate dependency among observations. In this work we consider developing such models for dependent functional data with a focus on application to EO data. We consider adapting well studied FDA methodologies and borrow techniques from spatio-temporal statistics to allow for spatially dependent observations. In Section 1.2 we make concrete our definition of functional data.

1.2 Functional Representation

As mentioned in Section 1.1 EO data can be viewed as a collection of functional data. However, there is a choice about how we interpret observations in this collection. We may consider the data as a collection of functional observations with time being our functional dimension and space our collection dimension. For example, we may consider each spatial location from Figure 1.1 giving rise to a trajectory over time of which we have only observed 12 time points. Or we may consider the functional observations having a spatial domain and the collection dimension being time. That is we may consider each image in Figure 1.1 being a surface with observations only at pixel locations and the collection consists of a time series of such surfaces. The canonical presentation of functional data in FDA is to use time as the functional dimension, [63], and thus we use the below definition of functional data from this point of view.

1.2.1 Functional data

Multivariate data analysis usually revolves around the study of observations which are finite dimensional and is well studied. Modern data collection techniques can now create data which are extremely numerous and thus can often be viewed as functions.

For example, Ferraty and Vieu consider the case where we can observe a random variable at several times between some minimum and maximum time, $(t_{\min}, t_{\max})$, [18]. A single observation can then be considered as the collection $\{X(t_j); j = 1, 2, \ldots, J\}$ where J is the total number of temporal sample points and $X(t)$ is the response variable at time t. Unlike multivariate data we consider the case that the separation between observations becomes minimal. That is we consider the data as an observation from the continuous random process $\mathcal{X} = \{X(t); t \in (t_{\min}, t_{\max})\}$. We therefore use, as in [18] and [70], the following definition of a *functional variable.*

Definition 1.1 (Functional Variable)**.** *A random variable $\mathcal{X}$ is called a functional variable if it takes values in an infinite dimensional space (or functional space). Observations χ of $\mathcal{X}$ are called a functional data.*

Further to this, suppose we observe a collection of functional data (realisations of $\mathcal{X}$). Then we will denote this collection by the term *functional dataset.*

Definition 1.2 (Functional Dataset)**.** *A functional dataset, $\chi_1, \chi_2, \cdots, \chi_N$ is the collection of N realisations of functional variables $\mathcal{X}_1, \cdots, \mathcal{X}_N$ identically distributed to $\mathcal{X}$.*

The canonical way to present functional data and the subsequent methods is to use time as the continuous variable, [63, 18, 70], as described above. However, there is no such restrictions in either Definition 1.1 or Definition 1.2. In fact, another case is to consider the functional domain of the variables to be space. In our proposed methodologies we present when possible with respect to time due to the simplification it brings in notation. We will make explicit reference when we change the domain of our functional data, for example if we consider space as our continuous domain.

We introduce the following notation for use in the remainder of this work. We consider our EO dataset to be observed in some spatial domain with dimension p, which we denote by $\mathcal{S} \subset \mathbb{R}^p$, and temporal domain donated by $\mathcal{T} \subset \mathbb{R}$. Typically p will be two or three since these represent the space of common spatial domains in EO data. Any observed dataset we can enumerate with one index over the spatial location and the other indexing the temporal locations.

We assume our dataset is comprised of N spatial locations and let $\boldsymbol{s}_i \in \mathcal{S}$ be the spatial location of the i^{th} observed functional variable. At each spatial location we suppose we observe J_i temporal observations and denote by $t_{ij} \in \mathcal{T}$ the j^{th} temporal observation of the i^{th} functional variable. Then our dataset can be summarised by Y,

$$Y = \{y_{ij}; i = 1, 2, \cdots, N, j = 1, 2, \cdots, J_i\}, \tag{1.1}$$

where y_{ij} is the response value of the i^{th} functional variable at time t_{ij} observed with error. That is we consider for each spatial location the discrete temporal observations being a sample from a realisation of a functional variable observed with error. That is:

$$y_{ij} = \chi_i\left(t_{ij}\right) + \varepsilon_{ij}, \tag{1.2}$$

where, as in Definition 1.2, χ_i is a realisation of functional variable $\mathcal{X}_i$ for $i = 1, 2, \cdots, N$. We consider each functional variable as being identically distributed as $\mathcal{X}$. As is common in most observation models, we assume that we observe data with error. Typically one assumes that the error process $\{\varepsilon_{ij}; i = 1, 2, \cdots, N, j = 1, 2, \cdots, J_i\}$ is a white noise process with variance σ_ε^2.

In this case one considers the modelling of the EO dataset by ensuring smoothness of some kind over the temporal domain via its functional data representation. We can then consider building in spatial dependency by assuming a sampling correlation in our N functional data. An area where such spatial dependency has been long studied is multivariate spatio-temporal methods.

1.3 Spatio-Temporal Methods

In the statistical literature spatial and spatio-temporal models have been extremely well studied, especially due to the prevalence of geo-statistical applications. In the following we briefly review some of the most commonly observed spatial and spatio-temporal statistical models in the multivariate analysis literature.

The monograph of [11] and references within provide a succinct summary of traditional methodologies in spatial statistics; many of which are applicable to EO data. Generally speaking, spatial data can be split into one of three categories; geo-statistical, area, and point process data, [11]. In this work the EO data described in Section 1.1 are most suitably modelled using geo-statistical models. The canonical model used in geo-statistical settings is the Kriging model. The Kriging model is well described in [73]. Such models treat spatial data as samples from a random spatial process and predictions for unknown values can be calculated from a weighted combination of known values in a neighbourhood of our unknown location utilising the correlation among neighbouring points. A prime example of the spatial Kriging model in use for EO data is given in [65]. Extensions to the basic Kriging technique have also been employed across a number of geo-statistical settings, including Co-Kriging involving extra covariate information for reconstruction, [91]. Kriging is well known in many fields through various names, in the FDA literature it is most often referred to as Gaussian processes regression. Shi and Choi describe in detail the concept of Gaussian processes in the context of functional regression, [70], and we discuss them more in Section 3.5.

As is detailed in [11] a key aspect to geo-statistical modelling is the specification of spatial dependency in the observed data. A common way for such specification is through parametric covariance or kernel functions. Traditional stationary parametric functions such as the Matérn covariance are discussed in detail in [11]. These commonly rely on the assumption of isotropy and stationarity in modelling which rarely holds in practice. Further literature has considered extensions of these and is in fact an active area of research. Schmidt and Guttorp compare a variety of methods for producing non-stationary and heterogeneous covariance structures for the goal of spatial interpolation, [68]. They group the various methods of creating such structures into four categories; deformation, convolution, covariate, and stochastic partial differential equations. The deformation approach proposed by Sampson and Guttorp extends the anisotropic stationary covariances, such as those described in [11], by allowing for a non linear transformation to the space which creates a latent space where isotropy holds, [67]. The convolution approach proposed by Higdon uses a specific form of the covariance kernel which can be represented as a convolution between a convolution kernel and a white noise process. We discuss such an approach more in Chapter 5. The covariate based approach to constructing non-stationary kernels tends to use an adaption to the convolution or deformation approaches with specific covariates, [68]. Finally the stochastic partial differential equation method proposed by Lindgren et al. construct non-stationary covariances through formulating the resulting Gaussian process as the solution of a stochastic partial differential equation which guarantees the construction of a well defined covariance function, [47].

A natural extension to purely spatial modelling of spatio-temporal data is to include the temporal domain. Such models are known as spatio-temporal models. Spatio-temporal models are well discussed in the monograph [12] by Cressie and Wikle. Spatio-temporal

Kriging models are well suited to EO data; however these models are relatively scarce in the literature. Militino et al. considers the application of such modelling in the satellite remote sensing literature and reasons the lack of them is primarily due to the added complexity these models produce in specifying valid space-time covariance functions, [56]. As such, one particular direction spatio-temporal modelling has considered is the creation of spatio-temporal covariance functions. Spatio-temporal covariance functions are discussed in [12]. Separability between spatial and temporal correlations is often a key assumption in some methods due to the reduction in computational complexity they provide. The separability assumption proposes that a space-time covariance function $k(\boldsymbol{s}, t, \boldsymbol{s}', t')$ can be factored into two separate covariance functions $k_s(\boldsymbol{s}, \boldsymbol{s}')$ and $k_t(t, t')$, one for each of the spatial and temporal dimensions. In particular for EO data, [22] consider the selection of separable covariances and [15] consider such models for air pollution data. However, the separability assumption may be too restrictive for capturing complex spatio-temporal processes, [12]. As such [57, 20, 5] considers tests for when the separability assumptions hold. The work in [10, 24, 38] consider the construction of non separable covariance functions for when separability does not hold for use in spatio-temporal models.

1.4 Summary of Research

The motivation of this work is to provide a model designed for EO data which provides an explanation of both the spatial and temporal process in a parsimonious way. We present a novel method named Correlated Principal Analysis through Conditional Expectation (CPACE), that is designed for modelling EO data. The model builds upon existing FDA techniques to extend modelling from independently observed functional data to functional data which exhibits spatial correlation. The emphasis in this work is to utilise the FDA paradigm over the temporal domain to aid in the decomposition of the data; with the understanding that our data generating process is smooth across the temporal domain. Such a decomposition gives a parsimonious description of the data with respect to its temporal domain by describing its principal modes of variation. We then estimate a spatial correlation structure for each component using well known spatial statistical methods. The combination of the resulting estimated spatial covariance structures with the principal directions aims to capture the majority of temporal and spatial dependency observed in the data. We can then utilise the CPACE model to help predict responses at unseen spatial and temporal locations, which is an area of keen interest in EO studies. We assess our model using various simulated data both with known correct data generating distribution and to simulations drawn from an incorrect data generating procedure. We apply our CPACE model to selected atmospheric variables from the CESM dataset as an example application of the model to EO data.

The work is structured as follows. In Chapter 2 we describe our example datasets which we use to illustrate the performance of the model. In Chapter 3 we present the methodologies underpinning the CPACE models, these are typically well known FDA and spatio-temporal statistical methods. We also present the smoothing methodologies used to estimate the mean and covariance surfaces of our random functional variables. In Chapter 4 we present an interim model built on the combination of two well known existing methodologies in the FDA literature with a focus on application to an EO dataset. Such a model proposes a novel approach to modelling EO data and helps to highlight the need to include both spatial and temporal effects in modelling such data. In addition, the proposed model in Chapter 4 gives an opportunity to explore EO data where the functional domain is space rather than time. We present the benefits and limitations of such an approach in practice in this chapter. In Chapter 5, we introduce the main contribution of this work; which is the CPACE model for correlated functional data. We describe the model in detail as well as providing asymptotic results. In Chapter 6, we apply the CPACE model to simulated data and in Chapter 7 to real world datasets. Simulation results are presented with comparisons to various existing models with a focus on comparative ability to recover known data generating parameters. Applied results to real world datasets are included with comparisons to existing techniques; with a focus on interpolation and forecasting abilities of the model. In Chapter 8, we highlight the practical difficulties in implementing the model, with discussion on various techniques which are used to overcome the high dimensionality which is typical in the EO data. Finally, in Chapter 9 we draw the conclusions of the work and present areas of possible further work.

Chapter 2

Datasets

In the following chapter we describe in detail our data which we will use as a source for assessing the performance of the models described within. We use a publicly available set of climate model simulations known as the CESM Large Ensemble (CESM-LE) dataset, [43]. The CESM-LE dataset provides a good example of EO data that is discussed in Section 1.1. The dataset is publicly available from https://www.cesm.ucar.edu/projects/community-projects/LENS/data-sets.html.

2.1 Community Earth System Model - Large Ensemble

The CESM-LE dataset is an extremely popular and significant dataset in the climate research community. It was developed to enable the assessment of recent past and near future climate change in the presence of internal climate variability, [43]. It does so by providing 40 simulations of a complex climate model where each simulation is subject to the same radiative forcing scenario but initialised from a slightly perturbed atmospheric state. As such the forty resultant simulations present the various trajectories the model might take due to internal climate variability of the model.

The model used to run the forty member ensemble is the Community Earth System Model version 1, [33], with the Community Atmosphere model version 5, [33], as the atmospheric component. The model is a fully coupled climate model which consists of a model for each of land, ocean, atmosphere and sea ice components of the climate. These are brought together with a coupler model. Figure 2.1 provides a simple overview as to how the CESM model couples the various components. This model is capable of simulating various Land, Ocean, Atmosphere and Sea Ice variables of the climate, such as the wind speed, temperature or pressure. The CESM-LE produces simulations of variables on the nominal 1 deg horizontal separation across the globe which induces our spatial resolution of the data. The ensemble produces variables at three different levels of temporal resolution between the years 1920 and 2100 for non-control simulations. The ensemble is able to produce variables at 6-hourly, daily, and monthly intervals.

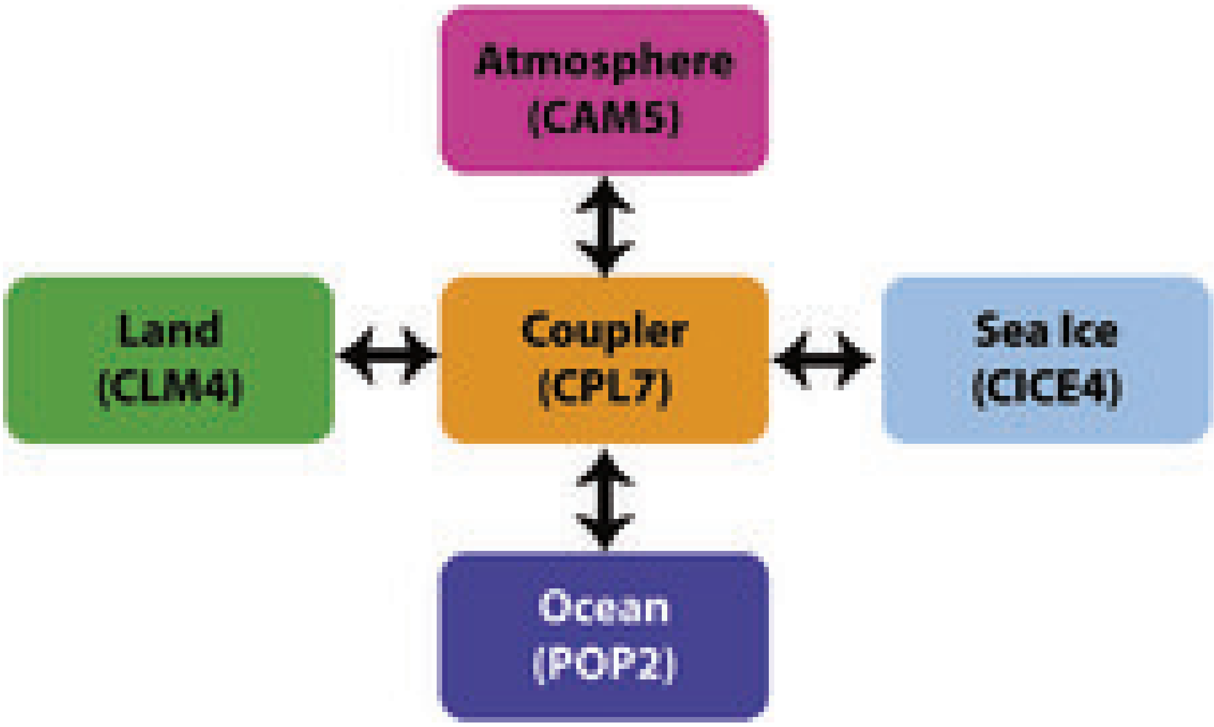


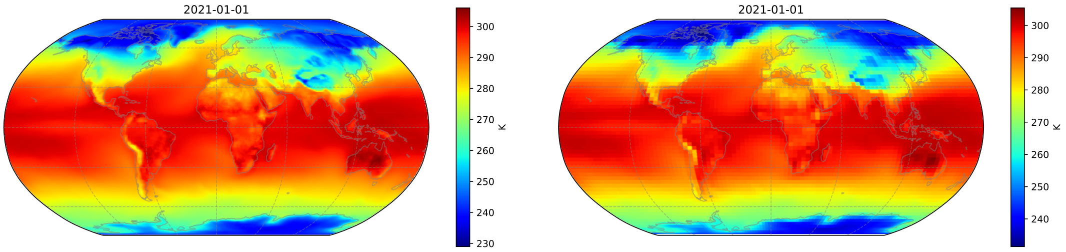
Fig. 2.1 The component models for the full CESM model, Figure from [43]. The individual component models are atmospheric (CAM5), ocean (POP2), land (CLM4), Sea Ice (CICE4), and coupler (CPL7). Details of which can be found in [43] and [33].

For this body of work we use the CESM-LE data by considering the forty members as separate simulations. Each simulation represents a single realisation of the various climate variables generated by the model described in [43] where the variation between realisations is coming from the internal climate variability. We apply a set of preprocessing to the raw data provided by the CESM-LE model as described below.

2.1.1 Preprocessing

The full CESM-LE data consists of a large number of climate variables on a relatively large spatial grid consisting of 192 × 276 locations and as such is a rather large dataset. The main goal in our preprocessing is to reduce the size of the data through a series of variable selection, spatial resampling, and temporal sectioning. We reduce the data size by considering only a subset of the full dataset by selecting four variables to study from the full model which are; temperature, pressure, wind speed, and precipitation. The next preprocessing step we use is a temporal cut. We consider only the output of the CESM-LE which occurs between December 2020 and January 2026. These time points were chosen such that the length of time gave reasonable ability to capture periodic elements but that the size of the data did not become too large. By using monthly frequency observations and this five year time horizon we have sixty temporal observations for each spatial grid point and for each of our four variables considered.

The final step in our preprocessing pipeline is to reduce the spatial dimension. To do this we resample the model simulations to a smaller spatial grid for each variable of interest. Resampling is achieved by averaging values of neighbouring locations until our desired resolution is achieved. In this case we resample until the spatial size of the dataset is 64 × 96 which corresponds to a reduction factor of 3 from the original CESM-LE data. Figure 2.2 shows the comparison between the full and resampled spatial observation grid over the globe due to our preprocessing step. The figure uses the temperature variable as at January 2021 as an illustrative example. Obviously using such an approach reduces the spatial resolution and thus our ability to see small scale spatial patterns. However, this is traded off against agility in terms of modelling due to the reduced data size. The reduction factor of 3 was chosen based on this trade off.



(a) Full resolution.

(b) Reduced resolution.

Fig. 2.2 The resampled spatial grid of observation measurements across the globe. Notice the reduced spatial resolution in Figure 2.2b compared to that in Figure 2.2a due to the resampling causing some loss of fine spatial detail. Example used is the average monthly temperature in Kelvin (K) on January 2021.

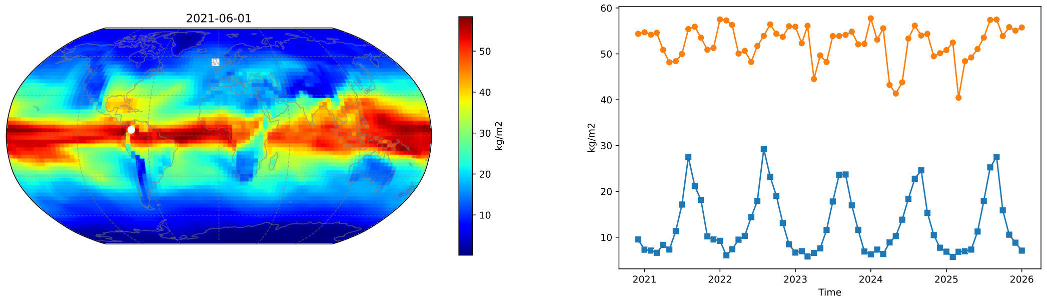
2.1.2 Variables

In the following section we focus our description to the four atmospheric model variables from the CESM-LE simulations that we will use in this body of work. These are; pressure, temperature, precipitation, and wind speed. We describe each variable in detail in their respective section and throughout this work we consider each as a separate EO dataset. We aim to show, through the use of an example time point and spatial locations, the various spatial and temporal processes that exist in each of these variables. These make such a dataset a credible EO dataset to test our proposed CPACE model on.

Precipitation

The total (vertically integrated) precipitable water component, abbreviated as TMQ, is an atmospheric component output of the CESM-LE, [43]. The variable is given units of $\mathrm{kg\,m^{-2}}$ and is available monthly on the full spatial grid. The monthly precipitation is calculated as the average over time from the CESM-LE model six hourly output.

We can see clearly the spatial variability of the precipitation over the globe by considering the heat map of June 2021 monthly precipitation for a single simulation; which is shown in Figure 2.3a. As one would expect, there is clear spatial correlation. For example, the tropics observe large amounts of precipitation whereas desert regions observe little. We can also see some subtler differences in the spatial correlation structure. Figure 2.3a shows that bands of precipitation are evident over the globe. This indicates that precipitation is much more correlated over lines of latitude than lines of longitude. This may be an indicator of spatial anisotropy in the generating process. There is also a case of observing more complex, possibly non stationary, spatial processes as the correlation structure seems different between say North America and Indonesia. We can similarly observe clear temporal correlations in the precipitation variable of the CESM model. Figure 2.3b shows the time series of two locations; the United Kingdom (UK) and Colombia. Each exhibit clear periodic signals as wet seasons and dry seasons repeat each year. However, we can see a clear level difference between the UK and Colombia precipitation as well as differences in the range of precipitation. This highlights the fact that not only do we have temporal correlation but this correlation is dependent on location.



(a) TMQ as at June 2021.

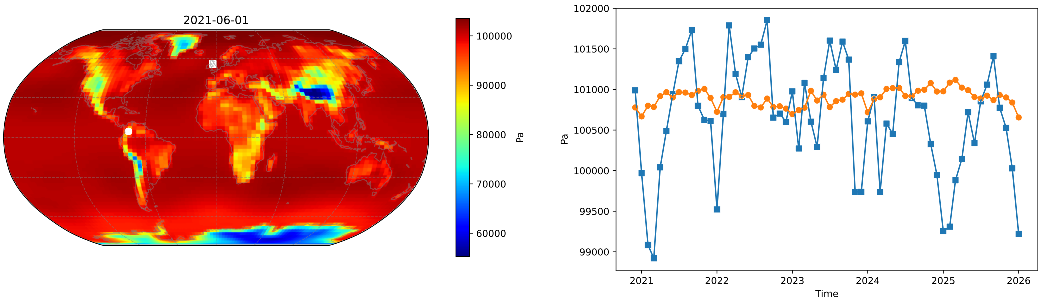
(b) TMQ over time.

Fig. 2.3 Overview of the monthly average precipitation variable (TMQ) from CESM-LE ensemble member 1. Figure 2.3a highlights the spatial correlation present while Figure 2.3b highlights the temporal correlation at two distinct locations; namely Colombia and the United Kingdom. In Figure 2.3a we have marked the location of Colombia with a white circle and the location of the UK with a white square. The orange circle and blue square markers in Figure 2.3b represent these countries respectively. Note the level difference in temporal correlation structure between the two locations, indicative of a spatio-temporal correlation process occurring.

Pressure

The surface pressure variable, abbreviated as PS, is another atmospheric component output of the CESM-LE, [43]. The component is given in Pascals (Pa) and represents the surface pressure at a height of 2m. It is available monthly on the full spatial grid and the monthly average is calculated as the average over time from the CESM-LE model six hourly output.

Figure 2.4 gives a brief insight to this variable. We can see the spatial correlation structure of pressure over the globe in Figure 2.4a. One can clearly see areas of high and low pressure. For example, there is a significant difference between the low pressure zone over Antarctica and high pressure zone over Australia. It is interesting to note that we see a clear difference in the smooth structure over sea and a rougher structure over land variables. This again might motivate that a non stationary spatial process is driving such a variable. Considering the temporal variation displayed for Colombia and the UK in Figure 2.4b; we can see definite structure over time, albeit different for each location. The UK exhibits much more variation in pressure than Colombia, however both do exhibit temporal correlation. Again, similar to the precipitation variable discussed in Section 2.1.2, this might suggest that modelling such a variable will need to consider both spatial and temporal correlations in conjunction with each other.



(a) PS as at June 2021.

(b) PS over time.

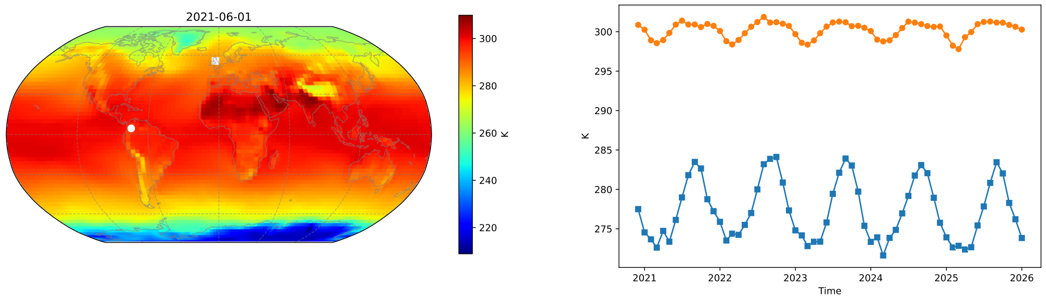
Fig. 2.4 Overview of the monthly average pressure variable from CESM-LE ensemble member 1. Figure 2.4a highlights the spatial correlation present while Figure 2.4b highlights the temporal correlation at two distinct locations; namely Colombia and the United Kingdom. In Figure 2.4a we have marked the location of Colombia with a white circle and the location of the UK with a white square. The orange circle and blue square markers in Figure 2.4b represent these countries respectively. Notice the stark difference between the time series variance in the UK and Colombia.

Temperature

The temperature variable, abbreviated to TREFHT, is an atmospheric component output of the CESM-LE, [43]. The variable refers to the average temperature in Kelvin (K) at the model reference height which is 2m above sea level. The average is available monthly with said average being calculated from the model six hourly output over the month.

Quite clearly the temperature exhibits spatial correlation across the globe and periodic signals through time as the seasons unfold. Figures 2.5a, 2.5b highlight this for the spatial and temporal correlation respectively. Clearly the temperature in June increases as we move closer to the equator and decreases at the poles. Similarly to the precipitation variable, we observe that the spatial correlation structure is clearly anisotropic. Correlation is much more pronounced over longitude than latitude. Looking more deeply at Figure 2.5 we can observe more complex correlation structures, as mentioned in Section 1.1. For example, Asia exhibits a localised area of low temperature right next to an area of relatively high temperature. This is very different to the extremely smooth variation that we see over the larger oceans such as the Atlantic. Again similar to the previous variables, this is an indication that there may be a non-stationary spatial process helping to drive this variable.

Figure 2.5b gives an insight into the temporal correlation structure in the variable for two locations; namely Colombia and the UK. There is strong evidence of a periodic signal driving both but clearly there is a level shift and change in amplitude for the two locations. This is again similar to the precipitation variable discussed in Section 2.1.2 and motivates the idea that there is a clear spatio-temporal process driving the variable rather than just either temporal or spatial process.



(a) TREFHT as at June 2021.

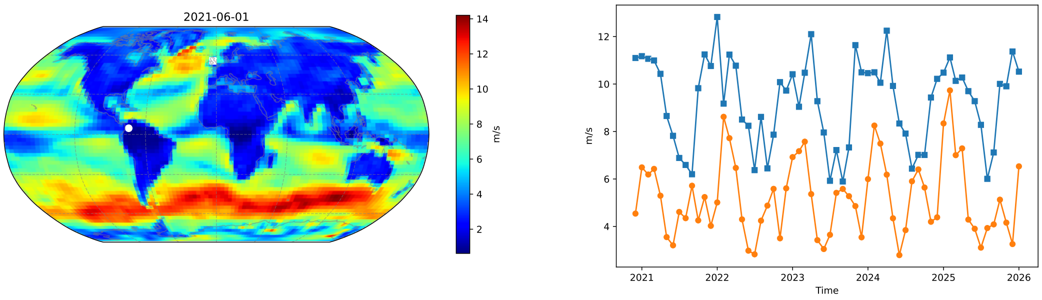
(b) TREFHT over time.

Fig. 2.5 Overview of the monthly average temperature variable from CESM-LE ensemble member 1. Figure 2.5a highlights the spatial correlation present while Figure 2.5b highlights the temporal correlation at two distinct locations; namely Colombia and the UK. In Figure 2.5a we have marked the location of Colombia with a white circle and the location of the UK with a white square. The orange circle and blue square markers in Figure 2.5b represent these countries respectively.

Wind speed

The wind speed variable, abbreviated to U10, is another atmospheric component output of the CESM-LE model. The variable refers to the average wind speed in $\mathrm{m\,s^{-1}}$ at a height of 10m above sea level. Again the variable is available on the full spatial grid and is available as a monthly average over time.

We visualise the spatial correlation of the variable in Figure 2.6a by considering a snap shot of the monthly average wind speed in June 2021. We can see, in contrast to the previous three variables, that this has a much rougher spatial correlation structure over the sea. In fact it is interesting to observe the distinct difference in variability over the sea compared to that over the land. This may suggest that we have two types of correlation structure existing for this variable, one for the land and one for the sea. In this case the model for the whole variable will clearly need to include a non-stationary spatial component to capture such a phenomena. Wind speed, like the other studied model variables, also exhibits temporal correlation. This is illustrated for the usual two locations, Colombia and the UK, in Figure 2.6b. The temporal correlation is much less pronounced for this variable than compared to the others. Visually there is perhaps evidence for a periodic signal for the UK location. However, yet again we do see a clear level shift between the two location. This again suggests that the spatial coordinate clearly impacts the observed function of wind speed over time. Similar to the other variables this suggests a model which includes both space and time as drivers for the process.



(a) U10 as at June 2021.

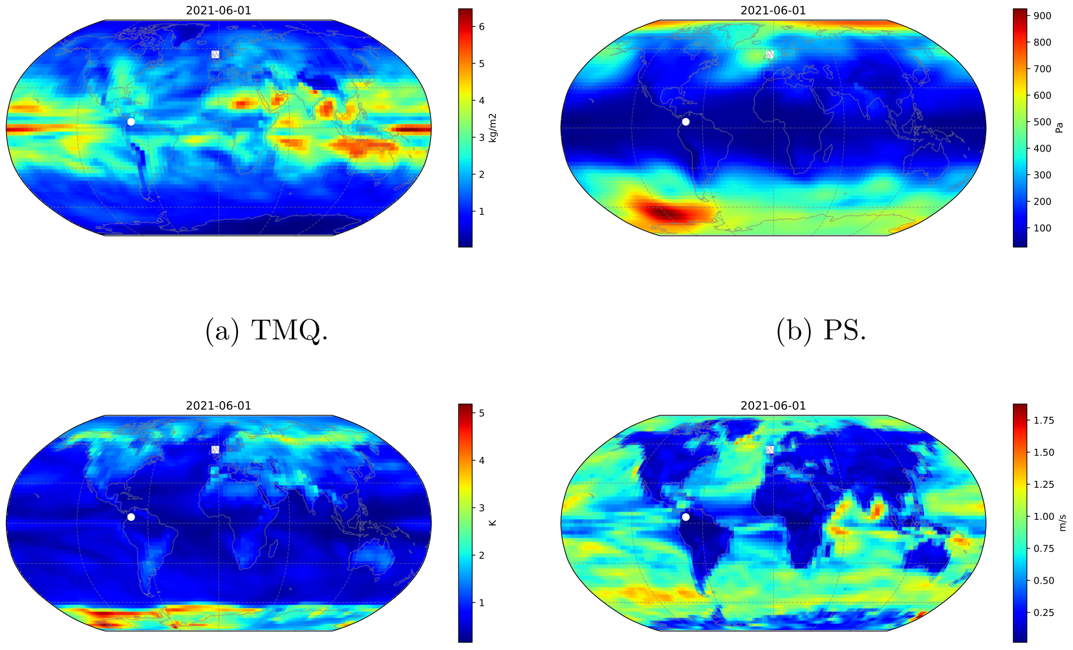
(b) U10 over time.

Fig. 2.6 Overview of the monthly average wind variable from CESM-LE ensemble member 1. Figure 2.6a highlights the spatial correlation present while Figure 2.6b highlights the temporal correlation at two distinct locations; namely Colombia and the UK. In Figure 2.6a we have marked the location of Colombia with a white circle and the location of the UK with a white square. The orange circle and blue square markers in Figure 2.6b represent these countries respectively.

2.1.3 Replications

For each variable discussed in Section 2.1.2 the CESM-LE data provides 40 replications; one from each ensemble member. We have illustrated the spatial and temporal correlations in the four variables in the Figures 2.3, 2.4, 2.5, and 2.6 for a single simulation. However we also have variability between replications and it is useful to illustrate this as it may provide insight into where we may expect difficulty in modelling. It is important that any model developed for such data should be able to account for this variability in the data generating process. Figure 2.7 displays a snap shot of the standard deviation of the respective variables at all sites in June 2021. Here the standard deviation is with respect to the 40 replications of the CESM-LE data.

Figure 2.7 gives an indication of how drastically each simulation differs for each variable. We would like to propose a model that can accommodate all the different scenarios presented in the various replications. Thus our model must be able to adapt to the regions of high standard deviation. From Figure 2.7a we can see that each simulation varies significantly in the tropics, but mostly around Indonesia. As such, we might be particularly interested in our model performance in this area for the precipitation variable. For the pressure variable there is an increase in standard deviation for the poles and particularly to the south west of Antarctica, see Figure2.7b. There is a similar result for the temperature variable in Figure 2.7c. Therefore we will be interested in model performance in this area for these variables. The wind speed variable has many more localised areas with high variance from replications. For example, the localised high variance of the coast of India and eastern Africa. These may pose a significant challenge to accommodate in a model for this variable.



(a) TMQ.

(b) PS.

(c) TREFHT.

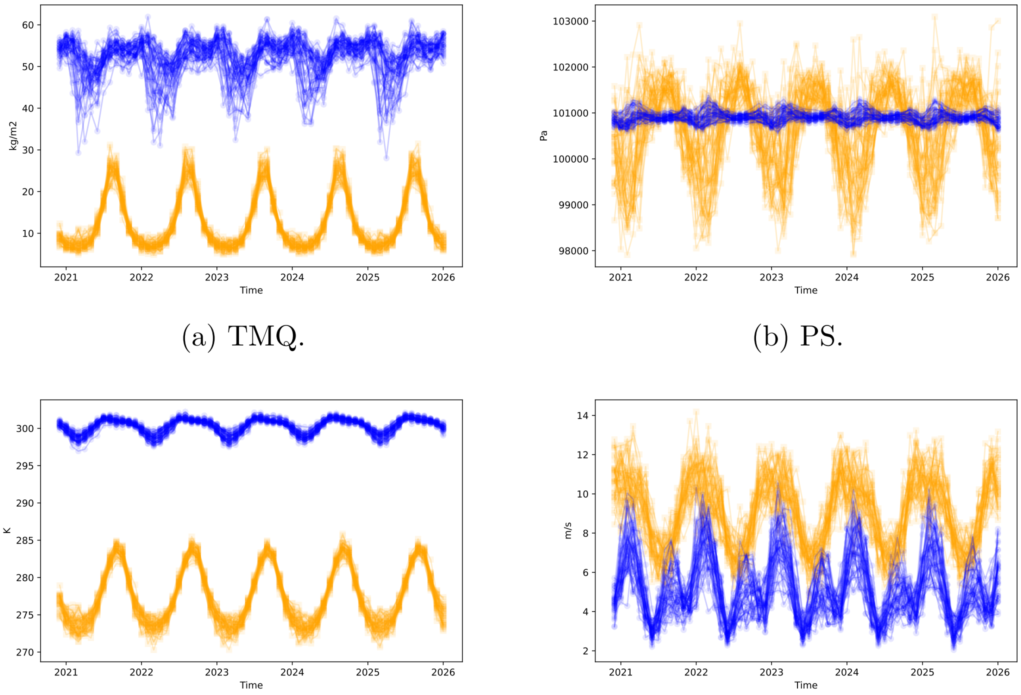
(d) U10.

Fig. 2.7 Standard deviation of the four variables considered at June 2021 for the 40 replications present in the CESM-LE dataset. The locations of the UK and Colombia are shown by the white square and circle markers respectively. These points are used as examples in Figure 2.8.

Similar to the spatial variability in replications we can consider the variability over time. Figure 2.8 highlights this variation over the 40 replications from the CESM-LE dataset.

We can see clearly from Figure 2.8c that the temperature variable function over time shows little variation between replications. However, variability does tend to increase in the troughs of these functions. This phenomena is more pronounced for the Colombia location. It may be interesting to assess performance of proposed models of the temperature variable with regard to this. The precipitation functions vary differently for the Colombia and UK locations. From the UK time series we see that although variability is high we can observe a periodic signal. It may be interesting to assess whether any model for this variable is able to pick up such a periodic signal, given that a single simulation may not show great periodicity. Pressure similarly exhibits very different function variability between locations, with the Colombia functions showing large changes in pressure compared to the changes observed between replications at the UK location. Again this indicates that there is a clear spatial component to the process driving such replications.

The above illustration of variability in replications gives an indication about the difficulties that any model must overcome to describe such variables and gives testament to the use of such a dataset to test our proposed model for EO data.



(a) TMQ.

(b) PS.

(c) TREFHT.

(d) U10.

Fig. 2.8 Four variables over observation period December 2020 to January 2026 for the 40 replications present in the CESM-LE dataset at two locations; namely Colombia and UK. These are represented by the blue and orange colours respectively.

Chapter 3

Background Methodologies

.

In the following chapter we consider the various statistical methodologies upon which we build our CPACE model. This chapter is structured as follows. First we focus on common FDA techniques applicable to EO data. We follow this by discussing smoothing methodologies which are of use in the FDA techniques. Finally we discuss Gaussian processes that are used in the CPACE framework to model correlation between functions.

3.1 Functional Principal Components Analysis

A commonly used technique in multivariate statistics is Principal Components Analysis (PCA), [81]. The technique finds dominant directions of variation and helps to achieve dimensionality reduction. This offers a parsimonious way to view data which is driven by the data themselves. The equivalent technique when the data are functional in nature is known as Functional Principal Components Analysis (FPCA), [63]. The basic concepts were studied in the mid twentieth century. The work of Karhunen and independently Loève paved the basic foundations of the technique in the FDA literature, [42, 51]. The FPCA technique essentially stems from representing the random function $\mathcal{X}(t)$ as an infinite linear combination of orthogonal functions. Such a representation is now known as the Karhunen-Loève theorem after its discoverers.

3.1.1 Formulation

The formulation of FPCA begins by assuming that $\mathcal{X}(t)$, $t \in \mathcal{T}$ is a square integrable stochastic process over some domain $\mathcal{T}$. By square integrable we formally mean that:

$$\mathcal{X} \text{ is square integrable } \iff \mathbb{E}\left(\int_{\mathcal{T}} |\mathcal{X}(t)|^2 dt\right) < \infty.$$

Let the mean and the covariance of the stochastic process $\mathcal{X}$ be denoted by $\mu(t)$ and $G\left(s,t\right)$ respectively, where:

$$\begin{aligned}\mu(t) &= \mathbb{E}\left(\mathcal{X}(t)\right),\\ G\left(s,t\right) &= \operatorname{Cov}\left(\mathcal{X}(s),\mathcal{X}(t)\right),\end{aligned}$$

Associated with the covariance surface $G\left(s,t\right)$ we have the linear operator T_G defined by:

$$\begin{aligned}T_G &: L^2\left(\mathcal{T}\right) \to L^2\left(\mathcal{T}\right),\\ T_G &: f \mapsto T_G f = \int_{\mathcal{T}} G\left(s,\cdot\right) f(s) ds,\end{aligned}$$

where $L^2\left(\mathcal{T}\right)$ is the set of all square integrable functions over our domain $\mathcal{T}$.

As T_G is a linear operator we can consider its eigenvalues and eigenfunctions which we will denote by λ_k and ϕ_k respectively (following convention set out in [90]) for $k = 1, 2, \cdots$. These are defined as the solutions to the Fredholm integral equations of the second kind, [90]:

$$\langle G(\cdot,t), \phi_k\rangle = \lambda_k \phi_k(t),$$

where $\langle f, g\rangle = \int_{\mathcal{T}} f(s)g(s)ds$ is the inner product in the space $L^2(\mathcal{T})$. Then by the Karhunen-Loève theorem one can express the centred process through the eigenvalues and eigenfunctions of the linear operator associated to the covariance surface, [42, 51]. That is:

$$\mathcal{X}(t) - \mu(t) = \sum_{k=1}^{\infty} \xi_k \phi_k(t), \tag{3.1}$$

where ξ_k is the k^{th} principal component associated to the eigenfunction ϕ_k. The Karhunen-Loève theorem assures us this L^2 convergence is uniform in t. The principal components are given by the following:

$$\xi_k = \langle \mathcal{X} - \mu, \phi_k\rangle. \tag{3.2}$$

Further to this decomposition the Karhunen-Loève theorem means that the principal components are independent from each other, centred, and have variance equal to their associated eigenvalue, [42, 51]. That is:

$$\begin{aligned}\mathbb{E}\left(\xi_k\right) &= 0,\\ \operatorname{Var}\left(\xi_k\right) &= \lambda_k,\\ \mathbb{E}\left(\xi_k\xi_l\right) &= 0, \text{ for } k \neq l.\end{aligned} \tag{3.3}$$

3.1.2 Interpretation

As with the multivariate principal components analysis the interpretation of the eigenvectors is often useful in exploratory analysis of data. The functional principal components analysis is of a similar form to the multivariate case and as such the same interpretation of the eigenfunctions is often employed. The first eigenfunction $\phi_1(t)$ encapsulates the dominant mode of variation in $\mathcal{X}(t)$ by construction since:

$$\phi_1 = \underset{\|\phi\|=1}{\arg\max} \operatorname{Var}\left(\langle \mathcal{X} - \mu, \phi \rangle\right).$$

Similarly, the k^{th} eigenfunction is the dominant mode of variation which is orthogonal to the preceding $k-1$ components. Therefore exploring the first few eigenfunctions often gives a parsimonious way to view the variation in the data. Alike multivariate PCA, it is often that the structure of the eigenfunctions replicates some observed physical process. As such, the FPCA decomposition is often used widely as a tool for data exploration.

In addition to this, we can use the fact that subsequent eigenfunctions capture less and less variation of the data as a form of dimensionality reduction, like PCA, [81]. In this sense we can consider truncating the full representation given in Equation (3.1) to the K leading eigenfunctions which gives an approximation to the full process which we will denote by $\mathcal{X}^K(t)$ where:

$$\mathcal{X}^K(t) = \mu(t) + \sum_{k=1}^{K} \xi_k \phi_k(t).$$

The approximation of $\mathcal{X}$ by $\mathcal{X}^K$ converges as:

$$\mathbb{E}\left(\langle \mathcal{X} - \mathcal{X}^K, \mathcal{X} - \mathcal{X}^K \rangle\right) = \sum_{k>K}^{\infty} \lambda_k \to 0 \text{ as } K \to \infty.$$

Using the leading principal components for reconstruction has the effect of capturing the main modes of variation of the data and ignoring smaller modes of variation. Choosing the number of principal components is then up to the practitioner; as in multivariate PCA, [81]. Ramsay and Silverman discuss in length the comparison of PCA to FPCA including commentary on the optimal choice of the number of principal components, [63, Chapter 8]. The practical implementation of FPCA then involves estimating various components. In particular estimation of; the mean function $\mu(t)$, the covariance surface $G(s,t)$, the K eigenfunctions and eigenvalues $\phi_k(t)$, λ_k respectively, and the principal components ξ_k for each realisation of the process $\mathcal{X}$ we observe.

3.2 Principal Analysis Through Conditional Expectation

We will assume for now that we have a sufficient method for estimating the mean and covariance surfaces which we will denote by $\hat{\mu}(t)$ and $\hat{G}(s,t)$ respectively. We discuss in more detail the estimation of these components in Section 3.3. Prior to the introduction of the Principal Analysis through Conditional Expectation (PACE) methodology in [90] FPCA decomposition was restricted due to the need for approximating the integrals in

Equation (3.2). As such, it was often a requirement that the functional data were observed on a dense regular grid which meant that the principal components could be reliably estimated though some numerical integration scheme, [63, Chapter 8]. This very much restricted the application of the FPCA technique. However Yao et al. introduced the PACE method for overcoming such an obstacle using conditional expectations for sparsely observed functional data, [90]. At the same time the technique of [90] accommodates for observation error.

Traditionally Equation (3.2), used for estimating the principal component scores for the i^{th} realisation, is approximated through sums. Substituting y_{ij} for $\mathcal{X}(t_{ij})$, $\hat{\mu}(t_{ij})$ for $\mu(t_{ij})$, and $\hat{\phi}_k(t_{ij})$ for $\phi_k(t_{ij})$ we obtain the estimate $\xi_i^S = \sum_1^{J_i} \left(y_{ij} - \hat{\mu}(t_{ij})\right) \hat{\phi}_k(t_{ij}) \left(t_{ij} - t_{i(j-1)}\right)$, [90], where y_{ij} is as described in Equation (1.2) and setting $t_{i0} = 0$. However, such an estimate breaks for the case that observations are sparse. Similarly this approximation will be biased when the error processes from Equation (1.2), ε_{ij}, has non-zero mean. Yao et al. overcome this by first assuming that the model is as follows, [90]:

$$\begin{aligned} y_{ij} &= \chi_i(t_{ij}) + \varepsilon_{ij}, \\ &= \mu(t_{ij}) + \sum_{k=1}^{\infty} \xi_{ik}\phi_k(t_{ij}) + \varepsilon_{ij}, \end{aligned} \tag{3.4}$$

with ε_{ij} being jointly Gaussian with ξ_{ik}. We also require the noise process satisfies:

$$\begin{aligned} \mathbb{E}\left(\varepsilon_{ij}\right) &= 0, \\ \operatorname{Var}\left(\varepsilon_{ij}\right) &= \sigma_\varepsilon^2. \end{aligned}$$

The number of measurements of the i^{th} subject is considered random which reflect sparse functional data. Such a description follows naturally from our dataset description, given in Equation (1.2), by using the FPCA decomposition structure of $\mathcal{X}$ as discussed in Section 3.1. Following [90] we define the subsequent vector notations:

$$\begin{aligned} \boldsymbol{Y}_i &= \left(y_{i1}, y_{i2}, \cdots, y_{iJ_i}\right)^\top, \\ \boldsymbol{\phi}_{ik} &= \left(\phi_k(t_{i1}), \phi_k(t_{i2}), \cdots, \phi_k(t_{iJ_i})\right)^\top, \\ \boldsymbol{\mu}_i &= \left(\mu(t_{i1}), \mu(t_{i2}), \cdots, \mu(t_{iJ_i})\right)^\top, \\ \boldsymbol{t}_i &= \left(t_{i1}, t_{i2}, \cdots, t_{iJ_i}\right)^\top. \end{aligned}$$

With such a model and assumptions, as stated in [90], the best prediction of the principal component scores for the i^{th} subject is given by:

$$\tilde{\xi}_{ik} = \mathbb{E}\left(\xi_{ik} | \boldsymbol{Y}_i, \boldsymbol{t}_i\right) = \lambda_k \boldsymbol{\phi}_{ik}^\top \boldsymbol{\Sigma}_{\boldsymbol{Y}_i}^{-1} \left(\boldsymbol{Y}_i - \boldsymbol{\mu}_i\right), \tag{3.5}$$

where $\mathbf{\Sigma}_{\boldsymbol{Y}_i} = \operatorname{Cov}(\boldsymbol{Y}_i, \boldsymbol{Y}_i)$. The estimate for the principal component score can then be found by substituting in estimates for the various components in Equation (3.5). That is:

$$\hat{\xi}_{ik} = \hat{\mathbb{E}}\left(\xi_{ik}|\boldsymbol{Y}_i, \boldsymbol{t}_i\right) = \hat{\lambda}_k \hat{\boldsymbol{\phi}}_{ik}^{\top} \hat{\mathbf{\Sigma}}_{\boldsymbol{Y}_i}^{-1} \left(\boldsymbol{Y}_i - \hat{\boldsymbol{\mu}}_i\right). \tag{3.6}$$

The covariance matrix $\hat{\mathbf{\Sigma}}_{\boldsymbol{Y}_i}$ is formed with $(l, m)^{\text{th}}$ element:

$$\left[\hat{\mathbf{\Sigma}}_{\boldsymbol{Y}_i}\right]_{lm} = \hat{G}(t_{il}, t_{im}) + \hat{\sigma}_{\varepsilon}^2 \delta_{lm},$$

where $\hat{\sigma}_{\varepsilon}^2$ is the estimated variance of the noise process. The estimation method for this is discussed in Section 3.3. Yao et al. also provide asymptotic properties of such an estimator along with asymptotic confidence bands where the mean and covariance surfaces are estimated with local linear smoothers, [17].

The conditional expectation technique described above from [90] alleviates the issue of poor integral approximation from sparsely observed data when the estimated covariance surface is a relatively good fit to the true covariance surface. This is a somewhat better condition as it allows one to pool data from different observed subjects to estimate such a surface and thus the requirement of dense data per subject is relaxed to having dense data from the collection over all subjects. We discuss a particular method for estimating such surfaces in Section 3.3.

3.3 Penalised Regression Splines

Smoothing models underpin much of FDA. FDA uses the smoothness of observations over a continuous domain to help inform and model observed data, [63]. Typically, as described in Section 1.2, data is only observed discretely. Therefore with most FDA methodology there must be a conversion from discretely observed data and the continuous functional variable that generates it. This is particularly the case for our EO data since we have discrete observations specified by our data model given in Equation (1.1) which we assume is generated by observations of continuous functions given by our models in Equation (1.2). Many models for obtaining a smooth of the data have been studied, such as kernel smoothing, polynomial regression, and local linear smoothing, [63, Chapter 4]. In this section we consider the well studied technique of obtaining smooths of discrete data through penalised regression splines, [66]. We will use such a method to estimate the mean and covariance surfaces present in the PACE methodology as described in Section 3.2. We first describe the components that form the foundations of a regression spline; the spline basis.

3.3.1 Basis splines

One of the components of a penalised regression splines is the basis functions used in the regression. As the name suggests regression splines uses spline functions as the regression basis. A spline function of order $d \in \mathbb{Z}^+$, which is well documented in the monograph of De Boor, is a piecewise polynomial function of degree $d-1$, [14]. In the case of a spline function of order d over a univariate domain $\mathcal{T} = [a,b] \subset \mathbb{R}$, which we denote by $S : \mathcal{T} \to \mathbb{R}$, we have:

$$S : t \mapsto S(t) = \begin{cases} P_0(t) \text{ if } \tau_0 < t \leq \tau_1, \\ P_1(t) \text{ if } \tau_1 < t \leq \tau_2, \\ \vdots \\ P_{m-1}(t) \text{ if } \tau_{m-1} < t \leq \tau_m, \end{cases},$$

where $P_i : [\tau_i, \tau_{i+1}] \to \mathbb{R}$ are polynomial functions of degree $d-1$. The vector of points $\boldsymbol{\tau} = (\tau_0, \tau_1, \cdots, \tau_m)$ is known as the knot vector for the spline and must satisfy $a = \tau_0 < \tau_1 < \cdots < \tau_m = b$. By specifying that the piecewise polynomials must share the same derivative order up to a degree we can ensure continuity of relative smoothness over the knot points and the whole spline function. We specify the continuity at each point in our knot vector by the continuity vector $\boldsymbol{r} = (r_0, \cdots, r_{m-1})^\top$ where r_i specifies that P_i and P_{i+1} share common derivative values at point τ_i for derivatives up to order r_i. The spline type can be specified completely by specifying the knot locations and the continuity vector, [14]. In fact, one can extend our definition of the knot vector to incorporate both the knot and continuity vector into one. This is known as the extended knot vector, which will completely specify the spline type, [14]. We define the extended knot vector as the vector of knot points which repeats the i^{th} knot vector exactly $n - r_i$ times. That is:

$$(\tau_0, \cdots, \tau_0, \tau_1, \cdots, \tau_1, \cdots, \tau_{m-1}, \cdots, \tau_{m-1}, \tau_m \cdots, \tau_m)$$

We denote the spline functions of order d with extended knot vector by $S_{d,\boldsymbol{\tau}}$.

The Basis splines are more commonly referred to as B-splines, [45]. B-splines are basis functions for splines of the same order defined over the same knots. They are typically defined recursively, [45, 14]. The classic algorithm for the recursive construction is known as the Cox-de Boor recursion formula, [14], and is given as follows. Given a knot vector $(\tau_0, \cdots, \tau_0, \tau_1, \cdots, \tau_1, \cdots, \tau_{m-1}, \cdots, \tau_{m-1}, \tau_m, \cdots, \tau_m)^\top$ the B-spline of order 1 is given by:

$$B_{i,1}(t) = \begin{cases} 1, & \text{for } \tau_i \leq t < \tau_{i+1} \\ 0, & \text{otherwise.} \end{cases}.$$

The higher order B-splines are defined by recursion as:

$$B_{i,q+1}(t) = w_{i,p}(t) B_{i,q}(t) + [1 - w_{i+1,q}(t)] B_{i+1,q}(t), \tag{3.7}$$

where $w_{i,q}$ is a weighting for the i^{th} B-spline of order d given by:

$$w_{i,q}(t) = \begin{cases} \frac{x-\tau_i}{\tau_{i+q}-\tau_i}, & \text{for } \tau_{i+q} \neq \tau_i \\ 0, & \text{otherwise.} \end{cases}.$$

A B-spline basis system of size Q can then be considered by choosing the extended knot vector $\boldsymbol{\tau}$ and specifying the order, d, of the B-spline functions, and is given by the collection:

$$\{B^{\boldsymbol{\tau}}_{d,q}(t)\}_{q=1}^{Q},$$

where Q is the number of basis functions to use in the system, $\boldsymbol{\tau}$ is the extended knot vector, and $B^{\boldsymbol{\tau}}_{d,q}$ is the q^{th} B-spline of order d defined by Equation (3.7) for our knot vector $\boldsymbol{\tau}$.

3.3.2 Regression splines

As discussed in Section 3.2 the PACE methodology requires estimation of both the mean function, $\mu(t)$, and covariance surface, $G\left(s,t\right)$. Estimating such functions is a problem due to their infinite dimensional nature. A well studied and effective method for representing such functions is the use of a basis function expansion, [63]. That is representing the target surface using a linear combination of known basis functions. In this work we will utilise the B-spline basis function; as discussed in Section 3.3.1. The B-spline system is exceptionally popular due to its ease of computation and ability to reconstruct many surfaces, [14]. Such ease of computation makes it feasible to not only create large basis systems but also alleviates many fitting procedures as we can re-evaluate the basis system at various points with ease. These properties are very useful when using such a basis for regression models. Other common basis systems include the Fourier, Monomial, and Polynomial basis systems. See [63] for details of these basis systems in the functional framework. There are other methods of fitting smooth functions to observed data; such as the locally linear smoother. These are fairly common in functional data analysis. For example, [90] uses such smoothers in their work on the PACE methodology. We have chosen to consider spline smoothing,due to the above properties. In addition we feel there is an extra benefit of these by the ability to specify smoothness constraints through a penalty term which can be useful in capturing known properties of the underlying functional data. We discuss such regression penalties later in this section.

In the following we present the approach for estimating an arbitrary realisation of our functional random variable $\chi_i(t)$ over domain $\mathcal{T}$ and discuss how we extend the same concept to a two dimensional surface over $\mathcal{T} \times \mathcal{T}$.

We assume that our function can be represented using an order d B-spline basis system with knot vector $\boldsymbol{\tau}$:

$$\begin{aligned}\chi_i(t) &= \sum_{q=1}^{Q} c_q B^{\boldsymbol{\tau}}_{d,q}(t),\\ &= \boldsymbol{c}^{\mathsf{T}} \boldsymbol{B}^{\boldsymbol{\tau}}_{d}(t),\end{aligned} \tag{3.8}$$

where $\boldsymbol{c} = (c_1, \cdots, c_Q)^\intercal \in \mathbb{R}^Q$, $\boldsymbol{B}_d^{\boldsymbol{\tau}}(t) = \left(B_{d,1}^{\boldsymbol{\tau}}(t), B_{d,2}^{\boldsymbol{\tau}}(t), \cdots, B_{d,Q}^{\boldsymbol{\tau}}(t)\right)$, and Q is the dimension of the expansion. If such basis functions have nice properties like being easy to compute then such a representation for χ_i given by Equation (3.8) can be extremely useful since most problems can be reduced to involving only the finite dimensional vector $\boldsymbol{c} \in \mathbb{R}^Q$.

Our representation of χ_i using a basis system then becomes the problem of choosing the coefficients $\boldsymbol{c}$ using only our set of observations of $\boldsymbol{Y}_i$ which are observed with error. The most common method for fitting a basis system to discretely observed data is by choosing the coefficients of the expansion, c_q, given in Equation (3.8) by minimising the criterion, [8]:

$$\mathrm{SSE}_{\boldsymbol{Y}_i}(\boldsymbol{c}) = \|\boldsymbol{Y}_i - \boldsymbol{B}\boldsymbol{c}\|^2, \tag{3.9}$$

where $\boldsymbol{B} = (\boldsymbol{B}_d^{\boldsymbol{\tau}}(t_{i1}), \boldsymbol{B}_d^{\boldsymbol{\tau}}(t_{i2}), \cdots, \boldsymbol{B}_d^{\boldsymbol{\tau}}(t_{iJ_i}))^\intercal$ is the $J_i \times Q$ matrix of the basis system evaluated at observed time points corresponding to the J_i length observation vector $\boldsymbol{Y}_i$. Minimising such a criterion is given by, [8]:

$$\hat{\boldsymbol{c}} = \left(\boldsymbol{B}^\intercal \boldsymbol{B}\right)^{-1} \boldsymbol{B}^\intercal \boldsymbol{Y}_i.$$

The simple least squares approximation is a well studied and standard approach. See [8] for a thorough introduction to the concept. Such a methodology is often suitable for situations where our error process $\varepsilon(t)$ is a white noise process. This process for the noise is often unrealistic; as such a simple adjustment to the least squares criterion in Equation (3.9) can be used to allow for correlation among the observation errors:

$$\mathrm{SSE}_{\boldsymbol{Y}_i, \boldsymbol{W}}(\boldsymbol{c}) = \|\boldsymbol{W}^{\frac{1}{2}}(\boldsymbol{Y}_i - \boldsymbol{B}\boldsymbol{c})\|^2,$$

where $\boldsymbol{W}$ is a weighting matrix for the observations. Ideally the matrix will be the inverse of the variance-covariance matrix of the observations. Minimising the adjusted criterion is given by, [8]:

$$\hat{\boldsymbol{c}} = \left(\boldsymbol{B}^\intercal \boldsymbol{W} \boldsymbol{B}\right)^{-1} \boldsymbol{B}^\intercal \boldsymbol{W} \boldsymbol{Y}_i. \tag{3.10}$$

The estimate with least squares fitting can then be found by substituting $\hat{\boldsymbol{c}}$ for the $\boldsymbol{c}$ in Equation (3.8), [8]. That is:

$$\hat{\chi}_i(t) = \hat{\boldsymbol{c}}^\intercal \boldsymbol{B}_d^{\boldsymbol{\tau}}(t).$$

The selection of the knot vector is well studied and the classical choice is to choose a knot vector where knots are located at the sampling points, [14].

An issue with the classical least squares fitting using a basis system expansion is the choice of number of basis functions, [63]. We are constrained to choose Q to be less than or equal to the number of observations, J_i. This is because more than J_i basis functions would results in Equation (3.10) being ill defined since the matrix $\boldsymbol{B}$ would have linearly dependent columns. However, we still have the choice to choose Q between 1 and J_i. Exactly which value for Q to choose is unknown and results in bias - variance trade off in the estimator, [63]. A large number of basis functions reduce bias in the estimator

$\hat{\chi}_i(t)$, but the variance of this estimator may be unacceptably high. Conversely, a lower number of basis functions will result in high bias of the estimator but low variance. The bias-variance trade off is well studied and there is a vast literature on the methodology of choosing the number of basis functions. However, there is no gold standard and often the choice is made in an ad hoc fashion, [63]. Such an issue motivates modifying the fitting criterion which determines $\hat{\boldsymbol{c}}$ in Equation (3.10).

Penalties

Ideally, we want to penalise estimators which have high variance, that occur naturally when we have a large number of basis functions, but keep bias low. The naive choice of just reducing the number of basis functions, known as regression splines, fails in this respect, [66]. One such approach to do this is to reduce the number of basis functions in conjunction with a penalty, known as penalised regression splines, [66]. Such an approach was first used in [61] who applied such a technique to ill posed inverse problems. [66] discusses various other spline smoothing techniques as well as the penalised regression splines.

Penalised regression spline models adjust the fitting criterion in Equation (3.10) to, [66]:

$$\mathrm{PSSE}_{\boldsymbol{Y}_i,\boldsymbol{W},\lambda}(\boldsymbol{c}) = \|\boldsymbol{W}^{\frac{1}{2}}\left(\boldsymbol{Y}_i - \boldsymbol{B}\boldsymbol{c}\right)\|^2 + \omega\boldsymbol{c}^{\intercal}\boldsymbol{P}\boldsymbol{c}, \tag{3.11}$$

where $\boldsymbol{P}$ is formed with $(l,m)^{\text{th}}$ element $[\boldsymbol{P}]_{lm} = \langle L\left(\boldsymbol{B}_l\right), L\left(\boldsymbol{B}_m\right)\rangle$ and $\omega \in \mathbb{R}^+$ is a parameter which controls the regularisation trade off. L is some linear differential operator. Typically, one chooses L to be the required smoothness of the target function and examples include simple first or second derivatives, [66].

Analytically minimising the PSSE criterion in Equation (3.11) can be found via, [66]:

$$\hat{\boldsymbol{c}} = \left(\boldsymbol{B}^{\intercal}\boldsymbol{W}\boldsymbol{B} + \omega\boldsymbol{P}\right)^{-1}\boldsymbol{B}^{\intercal}\boldsymbol{W}\boldsymbol{Y}_i. \tag{3.12}$$

Essentially, such a penalisation term determines that there should be a trade off between the bias which corresponds to the first term in Equation (3.11) and the variance which is the second term. This trade off is controlled by the regularisation parameter ω. The advantage of this method is that we can now let Q, our number of basis functions, be large without worrying of over fitting as the penalty term in Equation (3.11) will penalise functions with high variability in terms of the differential operator L.

The choice of differential operator is a well studied problem also. A common choice is the first or second order differential, denoted by D^1 and D^2 respectively, as this specifies a reasonable level of smoothness in the target function, [66]. However, often more complex terms are used to facilitate known properties of the target functions, such as letting L be the harmonic acceleration operator which forces a periodic form of the target functions. More care must be taken when extending the linear differential operator to higher dimensions which is discussed below. Additionally, in the case of B-spline basis system these penalty matrices are typically evaluated using a form of numerical integration, [63].

Penalised regression splines moves our problem of selecting Q to choosing our regularisation parameter, ω. Such a parameter influences the strictness with which we expect our target function to be smooth as defined by the operator L. Choosing this parameter is a problem that is present not only in spline smoothing but other penalised regression approaches, [53]. A popular method for choosing such a parameter is Generalised Cross Validation (GCV). GCV, introduced by Wahba in [75], is a well studied method which has good asymptotic properties as the number of observations tends to infinity, [77, 76]. GCV chooses ω as the minimiser of the GCV criterion $V(\omega)$ which is given by, [77]:

$$V(\omega) = \frac{J_i^{-1} \|(\boldsymbol{I} - \boldsymbol{A}) \boldsymbol{Y}_i\|^2}{\left[J_i^{-1} \mathrm{tr}\,(\boldsymbol{I} - \boldsymbol{A})\right]^2}, \tag{3.13}$$

where $\boldsymbol{A}$ is the influence matrix defined by:

$$\boldsymbol{A} = \boldsymbol{B} \left(\boldsymbol{B}^{\intercal} \boldsymbol{W} \boldsymbol{B} + \omega \boldsymbol{P}\right)^{-1} \boldsymbol{B}^{\intercal} \boldsymbol{W}.$$

The GCV method can then be minimised for ω using a numerical minimisation routine. For large J_i it is known that the GCV criterion performs well in recovering a regularisation parameter which minimises variance while maintaining low bias in the reconstruction of the target function, [76]. For the case of low J_i the GCV method may not be reliable. As such, methods to extend the GCV criterion have been considered. The modified GCV criterion adds a further modifier to the denominator in Equation (3.13) by multiplying the trace of the influence matrix by a factor, [13]. The modified GCV approach effectively increases the cost associated with each effective parameter in the curve which reduces the chance of choosing an ω which under smooths the data, [13]. A similar but separate approach to adjusting the GCV is robust GCV, introduced by Lukas. Lukas uses a weighted sum of the GCV function with a term which penalises ω values that are close to zero, [53]. The performance of such methods are discussed in [54].

Choosing a basis system, a criterion to choose the regularisation parameter, and a differential operator then fully specifies the penalised regression spline approach. In the case of one dimensional functions the procedure applies as above. As such, we can estimate our mean function $\mu(t)$ through the use of a penalised regression splines where our observation points for the mean function are the pooled mean across subjects of the union of observed time points for all curves. However for multiple dimensions, particularly the case when we wish to smooth the covariance surface, we must make some adjustments to the approach described above.

Extension to higher dimensions

There are two issues when extending the penalised regression spline to higher dimensions; extending the basis system and extending the penalty specification. To alleviate the first we must specify a basis system which can cover multiple dimensions. In fact there are many such systems, [77]. One popular approach when we have regular data for FDA is using a tensor product B-spline system, [89]. We describe the extension to two dimensional surfaces, but the same extension will work for higher dimensional surfaces. Consider a two dimensional surface, $\sigma(s,t)$, which we represent by the tensor product spline given by, [89]:

$$\sigma(s,t) = \sum_{1 \leq q_1, q_2 \leq \bar{Q}} c_{q_1,q_2} \boldsymbol{B}_{d_1}^{\boldsymbol{\tau}_1}(s) \boldsymbol{B}_{d_2}^{\boldsymbol{\tau}_2}(t), \tag{3.14}$$

where $\boldsymbol{B}_{d_i}^{\boldsymbol{\tau}_i}$ is the B-spline basis system for the i^{th} dimension for $i = 1, 2$. For notational simplicity we assume the dimension of each marginal basis system is the same, $\bar{Q}$. However, in general this need not be the case. $\boldsymbol{C} \in \mathbb{R}^{\bar{Q} \times \bar{Q}}$ is a coefficient matrix to be determined. Equation (3.14) can be written more succinctly using a Kronecker product as, [89]:

$$\sigma(s,t) = \bar{\boldsymbol{B}}^{\mathsf{T}}(s,t) \operatorname{Vec}(\boldsymbol{C}), \tag{3.15}$$

where $\bar{\boldsymbol{B}}(s,t) = \boldsymbol{B}_{d_2}^{\boldsymbol{\tau}_2}(t) \otimes \boldsymbol{B}_{d_1}^{\boldsymbol{\tau}_1}(s)$ and $\operatorname{Vec}(\cdot)$ is an operator which stacks the columns of a matrix into a vector. We use the $\bar{\cdot}$ notation to make explicit that this basis is over multiple dimensions.

The same methods now follow as in the non penalised univariate case with this Kronecker basis system, [89]. However, we must still adjust the penalty matrix in Equation (3.12) to account for smoothness across multiple dimensions.

Using the tensor product basis system as described above; one might consider specifying the smoothness of the function in each dimension. Indeed, one such approach to extending the penalty specification which was introduced by Wood is to consider setting penalties on the marginal basis separately and to combine them by a weighted sum, [83]. Such an approach known as tensor product penalties is well studied in the linear generalised additive model setting, [82]. A two dimensional penalty matrix $\bar{\boldsymbol{P}}$ may be described as follows:

$$\bar{\boldsymbol{P}} = \omega_1 \boldsymbol{P}_1 \otimes \boldsymbol{I}_2 + \omega_2 \boldsymbol{I}_1 \otimes \boldsymbol{P}_2, \tag{3.16}$$

where $\boldsymbol{P}_i$ is marginal penalty over a single basis dimension as described in Equation (3.11), I_i is the identity matrix of dimension of the i^{th} dimension basis, and ω_i is the marginal regularisation parameter for $i = 1, 2$. The properties of such a smoothness penalty are discussed in detail in [83] with the main points being such a penalty is both scale invariant and low rank. In addition, [86] studies the use of such a penalty for the case of unevenly distributed data. The additional complication is that we now have multiple smoothness parameters $\omega_i \in \mathbb{R}^+$, one for each dimension of the surface to be smoothed. In this case the GCV methodology described above, can still be applied but now minimisation occurs with respect to the vector $\boldsymbol{\omega} = (\omega_1, \omega_2)^{\mathsf{T}}$. Implementation details of such can be found in [82].

We can now use the above approach to estimate our covariance surface, denoted by $\hat{G}(s,t)$, for use in PACE methodology, [90]. The discrete observations for the covariance surface to be smoothed are gathered by pooling individual observed covariances from across subjects, which is discussed in detail in both [90, 89]. Xiao provides asymptotic properties of such an approach to the covariance surface of independent functional data which are on par to the asymptotic results of other smoothers used in [90] for the PACE methodology, [89].

3.4 Functional Time Series

As discussed in Section 1.1; EO data is often both spatially and temporally correlated. The two types of correlation are often considered separately. An area in FDA which has considered a similar case where functional data is observed and observations are correlated is functional time series, [3]. Typically, functional observations are naturally indexed by some time of observation and correlation may occur between observations. Hence we may build up a time series of functional observations. Functional time series models are some of the first in the FDA literature to start to consider correlated functional observations. Although they limit themselves to temporal correlation many of the ideas can be considered for extensions to higher dimension correlation and so we discuss a few of the more popular methodologies in this section.

We focus on a technique introduced by Hyndman and Shang in [37] to forecast functional time series. Such a method is of interest as it expands methodology on how to use existing forecasting techniques in a functional setting. In particular [37] uses the FPCA decomposition described in Section 3.1 to decompose functional observations and then uses independent forecasting of each principal component score using standard multivariate techniques.

Hyndman and Shahid Ullah in [36] suggest to assume the principal component scores, ξ_{ik}, follow independent univariate time series models. Then, conditioning on the observed data $\boldsymbol{Y}$ given in Equation (1.1) and the set of principal components $\boldsymbol{\phi}(t) = (\phi_1(t), \phi_2(t), \cdots, \phi_K(t))$ they obtain the h-step ahead forecast of $\chi_{i+h|i}(t)$ as, [36]:

$$\hat{\chi}_{i+h|i} = \mathbb{E}\left(\chi_{i+h}(t)|\boldsymbol{Y}, \boldsymbol{\phi}\right) = \hat{\mu}(t) + \sum_{k=1}^{K} \hat{\xi}_{i+h|i,k}\phi_k(t), \tag{3.17}$$

where $\hat{\xi}_{i+h|i,k}$ denotes the h-step ahead forecast of the k^{th} principal component score. The method for which $\hat{\xi}_{i+h|i,k}$ is obtained can be any univariate time series method. Such methods are extremely well studied and discussed in the monograph of Hyndman and Athanasopoulos in [34]. Hyndman and Booth highlight that the forecast, given by Equation (3.17), is relatively insensitive to the choice of number of components in the principal decomposition provided it is sufficiently large. The variance of such a method can also easily be obtained through the sum of the component variances, [35]. The component variance of the forecast principal component scores are generally readily available from many time series models, [34]. The above forecasting methodology initially described in [36] used normal FPCA procedure with outliers weighted to zero, however this was reconsidered in [37] to include a geometric weighting to the principal components to allow for changes in the function over time.

Such a methodology motivates the construction of the CPACE model described in Chapter 5 for correlated functional data by considering the case where the principal components scores obey univariate correlated models not just time series. To describe some of these such models we use the concept of a Gaussian Process. We give background to this in the following section.

3.5 Gaussian Process Regression

The above section on functional time series shows there is scope for placing a model on the principal component scores to allow for correlation among functional observations. The natural progression to such work is to consider what options are available when we have more complex correlation structure or a higher dimensional domain. For example, in the case of EO data discussed in Section 1.1, we have functional observations over a spatial domain indexed by some coordinate, $\boldsymbol{s} \in \mathcal{S}$. For this the univariate time series methods discussed in [37] are not suitable and we look to Gaussian processes as one possible solution to model principal component scores which are indexed by space. As such, we discuss the basic concept of a Gaussian process in the following.

To describe a Gaussian process we first discuss the concept of a stochastic process. A real valued stochastic process is a collection of real random variables defined on the same probability space $(\Omega, \mathcal{F}, \mathcal{P})$ where Ω is a sample space, $\mathcal{F}$ is a σ-algebra, $\mathcal{P}$ is a probability measure, and the random variables (indexed by some set $\mathcal{S}$) are all real valued. More details of such constructions can be found in [7]. A stochastic process can then be written as the collection:

$$\{\xi(\boldsymbol{s}, w) | \boldsymbol{s} \in \mathcal{S}\},$$

where $w \in \Omega$. A sample function of the stochastic process is the mapping, for a point $w \in \Omega$:

$$\xi(\cdot, w) : \mathcal{S} \to \mathbb{R}.$$

A Gaussian process is a stochastic process which is parametrised by a mean function $m : \mathcal{S} \to \mathbb{R}$ where $m(\boldsymbol{s}) = \mathbb{E}\left(\xi(\boldsymbol{s})\right)$ and its covariance function:

$$\begin{aligned} &k : \mathcal{S}^2 \to \mathbb{R}, \\ &k : (\boldsymbol{s}, \boldsymbol{s}') \mapsto \operatorname{Cov}\left(\xi(\boldsymbol{s}), \xi(\boldsymbol{s}')\right), \end{aligned}$$

where for any finite collection of points, $\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_n \in \mathcal{S}$, the joint distribution of $\boldsymbol{\xi}_n = (\xi(\boldsymbol{s}_1), \xi(\boldsymbol{s}_2), \cdots, \xi(\boldsymbol{s}_n))^\intercal$ is a multivariate normal distribution with mean vector $\boldsymbol{m}_n = (m(\boldsymbol{s}_1), m(\boldsymbol{s}_2), \cdots, m(\boldsymbol{s}_n))^\intercal$ and covariance matrix $\boldsymbol{K}_n$ whose $(l, m)^{\text{th}}$ entry is given by $k(\boldsymbol{s}_l, \boldsymbol{s}_m)$, [70]. As such, Gaussian processes are a natural way of defining a prior distribution over spaces of functions, which are the parameter spaces for Bayesian non linear regression models. In this work, we will denote such a Gaussian process by $\mathcal{GP}$ and write

$$\xi(\cdot) \sim \mathcal{GP}\left(m(\cdot), k(\cdot, \cdot)\right).$$

One aspect of Gaussian process regression models is that under Gaussian assumptions they have a nice closed form for prediction. Without loss of generality, we assume for the below that we have a zero mean function. That is we assume $m(\boldsymbol{s}) = 0$ for $\boldsymbol{s} \in \mathcal{S}$. Let $S = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_n\}$ denote the design matrix of the regression, and $\boldsymbol{\xi}$ denote the corresponding target vector. Then conditioning on the joint Gaussian prior distribution on the observations gives the posterior at prediction points S_*, [80]:

$$\boldsymbol{\xi}_*|S_*, S, \boldsymbol{\xi} \sim \mathcal{N}\left(K(S_*, S)K(S,S)^{-1}\boldsymbol{\xi}, K(S_*, S_*) - K(S_*, S)K(S,S)^{-1}K(S, S_*)\right), \quad (3.18)$$

where $\boldsymbol{\xi}_*$ is our posterior process evaluated at the collection of prediction points, and $K(\cdot,\cdot)$ is the covariance matrix formed by evaluating the covariance function $K(\cdot,\cdot)$ at all pairs of inputs. This can be extended easily to noisy observations by adjusting the observed covariance $K(S,S)$ to include a diagonal component which is the effect of model observation error. See [80] for details.

One key aspect of the Gaussian process is the covariance function $k(\cdot,\cdot)$. The covariance function characterises various smoothness properties such as the sample path continuity and its differentiability, [80]. As such the choice of $k(\cdot,\cdot)$ heavily influences the prediction mean and covariance as described in Equation (3.18). There are various common forms of the covariance function but all must have the intrinsic property of being non-negative definite. The covariance function is of such importance in Gaussian process modelling and spatial statistics that it has been widely studied. See [80, Chapter 4] for a detailed introduction to various covariance functions. We expand on Section 1.3 to briefly introduce the form of covariance function which we will consider throughout this work.

Of the many different covariance functions employed in Gaussian processes, stationary covariance functions are most commonly employed due to their simplicity and ease of construction, [11]. One such commonly used covariance function is the Matérn covariance function which is given by, [2]:

$$C_\nu(d) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} d\right)^\nu K_\nu\left(\sqrt{2\nu} d\right), \quad (3.19)$$

where Γ is the gamma function, K_ν is the modified Bessel function of the second kind, $\nu \in \mathbb{R}^+$ is a shape parameter of the kernel, and d is the possibly anisotropic separation between two vectors $\boldsymbol{s}, \boldsymbol{s}'$, [2]. The covariance kernel $k(\boldsymbol{s}, \boldsymbol{s}')$ is then simply $C_\nu(d(\boldsymbol{s}, \boldsymbol{s}'))$. Figure 3.1 gives an illustrative example of samples from a Gaussian process with the Matérn covariance with varying ν values and fixed isotropic distance function d. For illustration, we consider realisations where the domain is one dimensional. We can see that ν controls the smoothness of the samples, with lower ν values resulting in less smooth realisations. One can see from this illustration how a simple change to the kernel can mimic a wide arrange of function, this is one of the reasons why Gaussian process are a powerful tool for representing functions. This concept extends to higher dimensions.

The issue with stationary covariance forms is that they are often quite restrictive in the sense that the correlation structure cannot vary across the domain. For example,

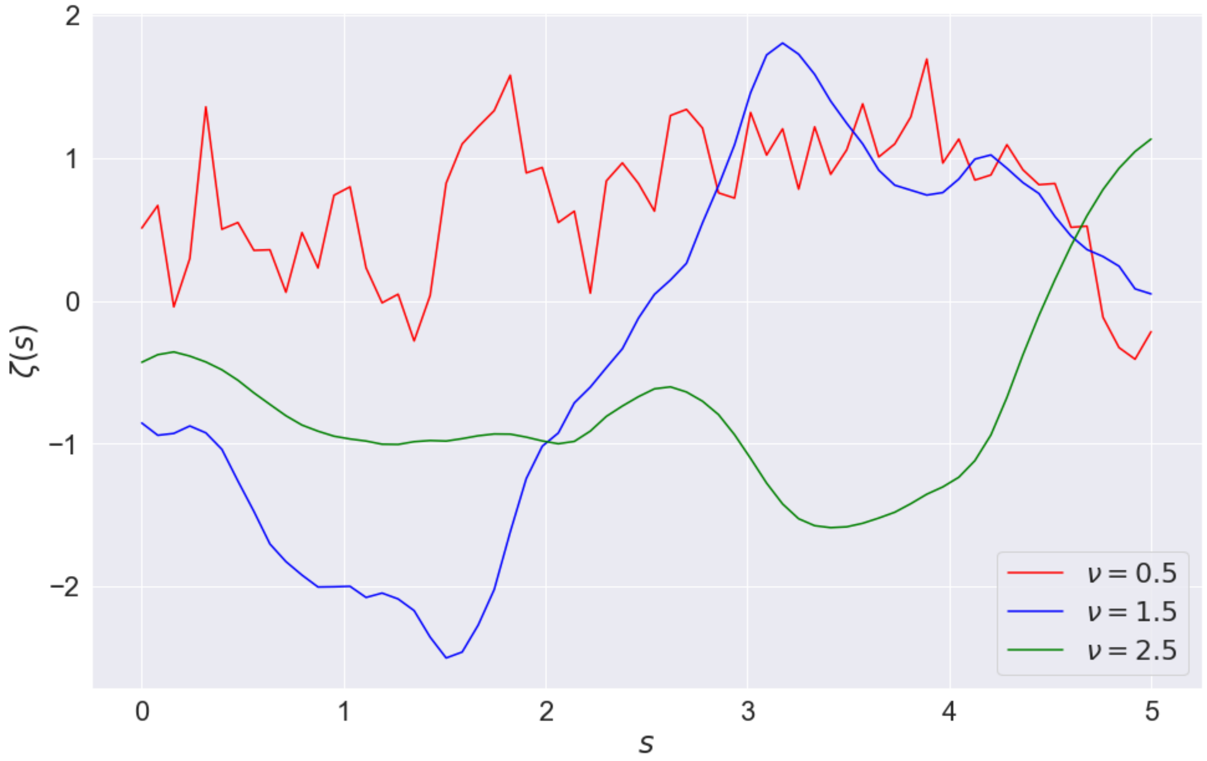


Fig. 3.1 Example realisations from Gaussian processes with the Matérn covariance function with differing ν parameters. We plot a single realisation from each of three processes with $m(s) = 0$ and $k(s, s') = C_\nu(d(s, s'))$ for $\nu = 0.5, 1.5, 2.5$ over the domain of $\mathcal{S} = [0, 5] \subset \mathbb{R}$.

this might be a too restrictive assumption in the case of climate data where correlation structure might be quite different in different parts of the globe. One particular way to extend the stationary Matérn kernel to be non-stationary is proposed in [62]. Paciorek and Schervish propose a method to knit together multiple stationary correlation functions such that the resultant function is non-stationary.

They provide a form of non-stationary covariance function $k^{NS}(\cdot,\cdot)$ from a stationary covariance function $k^S(\cdot,\cdot)$ as follows, [62]:

$$k^{NS}(\boldsymbol{s}, \boldsymbol{s}') = |\Sigma_{\boldsymbol{s}}|^{\frac{1}{4}} |\Sigma_{\boldsymbol{s}'}|^{\frac{1}{4}} |\frac{\Sigma_{\boldsymbol{s}} + \Sigma_{\boldsymbol{s}'}}{2}|^{-\frac{1}{2}} k^S(Q(\boldsymbol{s}, \boldsymbol{s}')),$$

where $Q(\boldsymbol{s}, \boldsymbol{s}') = (\boldsymbol{s} - \boldsymbol{s}')^\intercal \left(\frac{\Sigma_{\boldsymbol{s}} + \Sigma_{\boldsymbol{s}'}}{2}\right)^{-1} (\boldsymbol{s} - \boldsymbol{s}')$ and $\Sigma_{\boldsymbol{s}} = \Sigma(\boldsymbol{s})$ is the covariance matrix of the Gaussian kernel centred at $\boldsymbol{s}$. How $\Sigma_{\boldsymbol{s}}$ varies across the domain specifies how non-stationary the full covariance kernel is.

There have been other proposed approaches for introducing non-stationarity into the kernels of Gaussian processes. Sampson and Guttorp considers a non-parametric estimation procedure to model non-stationary kernels, [67]. Whilst non-stationary kernels have been considered by combining locally stationary kernels, [21]. Both of the above methods are interesting in their own right, however in this work we focus on the class of covariances formed through Paciorek and Schervish methods; as described above.

With almost all covariance functions and especially non-stationary covariances there are typically hyperparameters which must be estimated from the data. For example in the Matérn covariance we have the shape parameter ν and any length scale parameters defined in the distance function $d(\cdot, \cdot)$. These are typically estimated though maximum likelihood estimation, [80], however fully Bayesian estimation can also be achieved through some Markov Chain Monte Carlo (MCMC) scheme, [62].

Chapter 4

Functional Time Series Modelling For EO Data

EO datasets, as alluded to in Chapter 1, are often the primary source of information relating to a large spatial range. They may often be used when in situ measurements are not physically possible, or that in person collection is too dangerous. EO data is then often used to provide assistance to some monitoring or response effort. For example, Singha et al. uses such remotely sensed data to detect oil spills in oceans where in-situ measurements are not feasible, [71]. In such a scenario, a major drawback of EO data is often the limited acquisition times. There are two main problems with having limited acquisition times for EO data. Firstly, we may be observing a process where we are interested in the values of our data in-between acquisitions. Here, ideally, we would be able to capture another acquisition and thus increase our temporal resolution of the data. However, this may not be possible. So the problem at hand is how can we artificially increase the temporal resolution of the dataset. This is commonly referred to as interpolation, and our goal would be to interpolate the image that would be acquired, say at a time between two observed images for the whole spatial domain. Secondly, we may be observing a process where we are interested in future values of our data. That is, we are interested in the forecasted image based on our observations to date, and again we suppose we are interested in forecasting the imagery for the whole spatial domain. In this chapter, we discuss one such approach to both the interpolation and forecasting problems mentioned. In particular, we consider treating our functional dimension as space and using a combination of functional decomposition and functional time series modelling to aid in the interpolation and forecasting. To the authors knowledge this application of functional techniques, as described below, to EO data with the focus on space as the functional domain is a novel contribution.

4.1 Change of Representation

To describe our proposed model in this chapter we make a change to our representation of EO data. We will focus on viewing the EO data as a collection of images over space where we will index the images over time. To make this concrete we propose the following representation of our data which is adjusted from the discussion in Section 1.2. That is:

$$\bar{\boldsymbol{Y}} = \{\bar{y}_{ij}; i = 1, 2, \cdots, N, j = 1, 2, ..., J\},$$

where $\bar{y}_{ij}$ is the i^{th} spatial observation of the j^{th} acquisition in time. Here we represent $\bar{y}_{ij}$ as follows:

$$\bar{y}_{ij} = \bar{\chi}_j(\boldsymbol{s}_{ij}) + \bar{\varepsilon}_{ij}, \tag{4.1}$$

where $\bar{\chi}_j$ corresponds to the j^{th} functional variable over now the spatial domain, $\mathcal{S}$ and $\boldsymbol{s}_{ij}$ is the i^{th} spatial observation for the j^{th} acquisition. We use the $\bar{\cdot}$ notation to make explicit the change of representation from that discussed in Section 1.2 where time was our functional domain. We have also assumed here that we observe the same spatial observations for every acquisition by fixing $J_i = J$ for all $i = 1, 2, \cdots, N$. As we will see this is an assumption for notational simplicity only and the model set out in this chapter will work in the setting of sparsely observed data as well.

The change of representation facilitates the model described below and helps to emphasise that in this chapter we are interested in interpolation and forecasting the whole spatial domain through time. We note that the methodology discussed in Chapter 3 all equally applies in the setting of space being the functional domain, with some extensions needed for penalised regression splines which have been discussed.

Therefore, our goals of this chapter is then to estimate $\bar{\chi}$ for some unobserved acquisition time t_{j^*}. If $t_{j^*} \in \mathcal{T}$ this corresponds to interpolation, and likewise if $t_{j^*} \notin \mathcal{T}$ this corresponds to forecasting.

4.2 Modelling

The following model aims to combine a functional representation of the spatial surfaces using regularised spline smoothing with functional time series modelling as the technique to interpolate and forecast. We will use the term Functional Time Series Model (FTSM) to denote the model described below. The motivation of such an approach is that we wish to use a functional technique to reduce the dimensionality of the problem in the spatial domain, and then use relatively standard time series forecasting techniques for the interpolation and forecasting in the reduced domain. We describe this in two steps. We first describe the approach taken to reduce the dimensionality of the imagery then we describe our approach to forecasting and interpolation in the reduced domain.

4.2.1 Decomposition

As discussed in Chapter 3, a common form of functional decomposition is the functional principal components analysis (FPCA) (See Section 3.1). Re-framing this decomposition in the case of a spatial functional variable is simple, and extends in the natural way. That is we have the following representation of the centred functional process which is the equivalent to Equation (3.1) but for our functional random variable $\bar{\mathcal{X}}\left(\boldsymbol{s}\right)$ which is a surface over the spatial domain $\mathcal{S}$;

$$\bar{\mathcal{X}}\left(\boldsymbol{s}\right) - \bar{\mu}\left(\boldsymbol{s}\right) = \sum_{k=1}^{\infty} \bar{\zeta}_k \bar{\phi}_k\left(\boldsymbol{s}\right), \tag{4.2}$$

where $\bar{\mu}, \bar{\zeta}_k, \bar{\phi}_k\left(\boldsymbol{s}\right)$ are the natural extensions to the mean function, principal component score, and principal components respectively.

The determination of such components from our observed data $\bar{\boldsymbol{Y}}$ can be achieved in principal through the PACE framework as discussed in Section 3.2. However, one runs into difficulty in this setting as we would have to form a variety of spatial covariance matrices to estimate the covariance function $G : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ and perform matrix inversion of these matrices to estimate the principal component scores, as can be seen through Equation (3.6) in the one dimensional case. This can quickly become prohibitive in the spatial setting where even a small observed grid can lead to relatively large covariance matrices.

One method for overcoming such an obstacle is to estimate the functional variables $\bar{\chi}_j\left(\boldsymbol{s}\right), j = 1, 2, ..., J$ directly though a basis expansion. That is, we assume the following form for $\bar{\chi}_j\left(\boldsymbol{s}\right)$:

$$\bar{\chi}_j\left(\boldsymbol{s}\right) = \bar{\boldsymbol{B}}^{\intercal}(\boldsymbol{s})\bar{\boldsymbol{c}}_j, \tag{4.3}$$

where $\bar{\boldsymbol{B}}$ is the known basis system over two dimensions and $\bar{\boldsymbol{c}}_j$ corresponds to the coefficient matrix that is to be estimated directly.

We have discussed an approach to estimating the coefficients using penalised regression splines in Section 3.3.2 where the basis system formed of a Kronecker product of B-spline bases over each dimension. The form of Equation (4.3) is comparable to that given in Equation (3.15) but we now include an explicit form $\bar{c}$ of the vectorised coefficient matrix of Equation (3.15).

We can then use an appropriate method for estimating $\bar{\boldsymbol{c}}_j$ from our observed data $\bar{\boldsymbol{Y}}$ for each $j = 1, 2, \cdots, J$ such as penalised regression splines as discussed in Section 3.3.2. The approach described above; discussed in detail in [63], is a single step in estimating the FPCA decomposition. Therefore, we only use the observations corresponding to the j^{th} functional variable, $\bar{\boldsymbol{y}}_j = \{\bar{y}_{ij}; i = 1, 2, \cdots, N\}$ to estimate the basis expansion coefficients $\bar{\boldsymbol{c}}_j$. Here we can also see that, assuming we observe each functional variable densely is only a convenience; since sparsely observed functional variables will impact the estimation of the coefficients of the expansion but it will still admit such a representation.

Given a basis expansion form of our functional variable, the formation of the functional principal components can be applied in coefficient space, as discussed in [63, Chapter 8]. Expressing the simultaneous expansion of all J surfaces by:

$$\boldsymbol{\chi}(\boldsymbol{s}) = \boldsymbol{C}\bar{\boldsymbol{B}}(\boldsymbol{s}), \tag{4.4}$$

where $\boldsymbol{C}$ is the stacked matrix of J coefficient vectors of each basis expansion. The covariance function G is then given by:

$$G(\boldsymbol{s}, \boldsymbol{s}') = J^{-1}\bar{\boldsymbol{B}}^{\intercal}(\boldsymbol{s})\boldsymbol{C}^{\intercal}\boldsymbol{C}\bar{\boldsymbol{B}}(\boldsymbol{s}').$$

As discussed in Section 3.1 we are interested in the eigenfunctions of G. Again, following [63, Chapter 8], we suppose the eigenfunctions of G have a basis expansion.

$$\bar{\phi}(\boldsymbol{s}) = \bar{\boldsymbol{B}}^{\intercal}(\boldsymbol{s})\bar{\boldsymbol{b}}.$$

Now, following the discussion on FPCA in Section 3.1, we can find such eigenfunction by solving the Fredholm integral equations of the second kind, [90]. The form of these are simplified by the basis expansion as:

$$\langle G(\cdot, \boldsymbol{s}'), \bar{\phi}\rangle = J^{-1}\bar{\boldsymbol{B}}^{\intercal}(s)\boldsymbol{C}^{\intercal}\boldsymbol{C}\boldsymbol{W}\boldsymbol{b} = \lambda\bar{\boldsymbol{B}}^{\intercal}(\boldsymbol{s})\bar{\boldsymbol{b}}, \tag{4.5}$$

where $\boldsymbol{W} = \int_{\mathcal{S}} \bar{\boldsymbol{B}}(\boldsymbol{s}')\bar{\boldsymbol{B}}^{\intercal}(\boldsymbol{s})d\boldsymbol{s}$ is the symmetric matrix of pairwise inner products of the basis functions in our basis system. Since Equation (4.5) must hold for all $\boldsymbol{s} \in \mathcal{S}$ it implies a purely matrix equation of:

$$J^{-1}\boldsymbol{C}^{\intercal}\boldsymbol{C}\boldsymbol{W}\boldsymbol{b} = \lambda\bar{\boldsymbol{b}}.$$

The solutions of which can be obtained using standard procedures. In particular, this gives a methodology for obtaining the eigenfunctions of G utilising a matrix equation which is of the dimension of the basis system rather than that of the observed data. The associated score to the k^{th} principal components $\bar{\phi}_k(\boldsymbol{s})$ can be found similarly using matrix equation only as:

$$\bar{\zeta}_{jk} = (\boldsymbol{c}_j - \boldsymbol{c}_\mu)\,\boldsymbol{W}\boldsymbol{b}_k,$$

where $\boldsymbol{c}_\mu$ is the coefficient vector of the mean function in its basis expansion, or simply the mean of the coefficient matrix $\boldsymbol{C}$ in Equation (4.4).

The above FPCA using basis expansion, as proposed in [63, Chapter 8], gives a reduced dimension representation of our observed functional variables where we have overcome the issue of high spatial resolution making the PACE analysis unfeasible. As discussed in Section 3.1 these principal components will describe a maximum amount of variation in the dataset, however the aim of our modelling in this chapter is to achieve good interpolation and forecasting to unobserved time points. We note that the score process in our above representation encodes the temporal evolution of the observed imagery. Hence we would ideally like to produce a decomposition of our observed data that makes the score process as easy to interpolate and forecast as possible. There is no reason to believe therefore that the FPCA decomposition is the best for achieving this aim. To this end we will also consider a rotation to these principal components, known as Maximum Autocorrelation Factor Rotations (MAFR).

Maximal autocorrelation factor rotations

Rotations to multivariate principal component analysis have long been studied. Most often rotations are designed to emphasise a particular quality of the principal components. For example the Varimax rotation, [41], was established in 1958 by Kaiser. It places an emphasis on producing components which focus on particular ranges of the domain which often aids interpretability of the resulting components. Similar approaches to helping the interpretability of functional principal components have been studied. Ramsay and Silverman consider the extension of the Varimax rotation for FPCA, [63].

MAFR was proposed by Hooker et al. and developed in relation to functional observation with time as the functional domain in [29]. Here, they build on top of the multivariate rotation known as Maximum Autocorrelation Factors (MAF), [74]. MAF focused on finding a rotation that selects components which have minimum autocorrelation. Hooker and Roberts show that this can be extended to the functional domain by considering searching for components that have smallest integrated first derivative, [29]. They then highlight that this can be extended to any notion of smoothness given by some linear differential operator, such as those discussed in Section 3.3.2.

We detail the calculation of such MAFR rotations following the methodology proposed in [29]. For more details on the derivation of the rotations see [29].

Assume that we have a set of principal components $\{\bar{\phi}_k; k = 1, 2, \cdots, K\}$ obtained from the data. We collect this set into a vector notation as before, giving $\bar{\boldsymbol{\phi}}(\boldsymbol{s}) = \left(\bar{\phi}_1(\boldsymbol{s}), \bar{\phi}_2(\boldsymbol{s}), \cdots, \bar{\phi}_K(\boldsymbol{s})\right)^\intercal$. The MAFR rotation corresponds to, [29]:

$$\arg\min_{\boldsymbol{u}} \boldsymbol{u}^\intercal \langle L\bar{\boldsymbol{\phi}}, L\bar{\boldsymbol{\phi}} \rangle \boldsymbol{u}, \tag{4.6}$$

subject to $\boldsymbol{u}^\intercal\boldsymbol{u} = 1$ where L is some linear differential operator. Defining successive rotations in the standard way by minimising Equation (4.6) whilst being orthogonal to proceeding rotations. These can be found by the succeeding columns of U in the Eigen-decomposition of $P = \langle L\bar{\boldsymbol{\phi}}, L\bar{\boldsymbol{\phi}} \rangle$. That is:

$$P = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^\intercal.$$

The new rotated principal components are given by:

$$\boldsymbol{U}^\intercal \bar{\boldsymbol{\phi}}(\boldsymbol{s}).$$

As noted in [29] if the diagonal matrix D is ordered from largest to smallest eigenvalues, the final components of $\bar{\boldsymbol{\phi}}_{\text{MAFR}}$ will be the smoothest with respect to the operator L. A similar rotation to the scores gives the MAFR scores.

Both FPCA decomposition and the MAFR decomposition give principal components which are orthonormal and can be used interchangeably in the following. The idea is that the MAFR rotation will result in score processes that is easier to interpolate and forecast as it has been encouraged to be smooth by the operator L.

4.2.2 Interpolation and forecasting

Following the decomposition of our observed data into either FPCA or the MAFR components we have a series of principal components. These capture the various spatial structures present. The corresponding scores capture the temporal process of each component. Modelling such scores then paves the way for interpolating and forecasting the full imagery through this decomposition. This is exactly the same setup as formulated for functional time series methodology, [37]. We have previously discussed the approach to forecasting using functional time series methodology in Section 3.4 and we use the same methodology as previously discussed for interpolation and forecasting in this scenario. That is this methodology treats each principal component score as a univariate time series independently of the others. However, the MAFR scores will in fact be correlated by the rotation matrix U which would lend itself to possibly introducing a more complex modelling of the multivariate score processes. We choose not to and apply the same independent forecasting methodology for both FPCA and MAFR decompositions.

As mentioned in [37]; any univariate time series model could be used for forecasting and interpolation. For our model we choose to model each score process $\zeta_k(t)$ by a Gaussian process for $k = 1, 2, \cdots, K$. We have discussed Gaussian process regression in the general sense in Section 3.5 where our domain of interest was space. In this case we have a univariate temporal domain. That is, our score process $\zeta(t)$ (we drop the component indexing notation as we have the same structure on all components) is represented as:

$$\zeta(t) \sim \mathcal{GP}\left(m(t), k(t, t')\right),$$

where $m(t)$ is our mean function and $k(t, t')$ is the covariance function of the process. Here we choose the mean and covariance function which are tailored for forecasting.

In particular, we choose a linear mean function. That is:

$$m(t) = at + b,$$

for all $t \in \mathcal{T}$, where $a, b \in \mathbb{R}$ are unknown hyperparameters to be estimated for the mean function. The values of a and b are chosen through maximum likelihood estimation of the Gaussian process on the observed score process. We choose a non-zero mean to act as an aid in forecasting so that the Gaussian process will not be reverting back to zero at large forecast steps but will revert to the value of the mean function, [80].

As we chose a simple mean function for our model of the score process, we encode the possible complexity in the time series with a covariance kernel which is designed for pattern discovery. We choose to use a two component kernel function with an additive structure. That is our covariance function has the following form:

$$k(t, t') = k_{\text{trend}}(t, t') + k_{\text{med}}(t, t'),$$

where we choose k_{trend} to be a Gaussian kernel function that is designed to capture long term smooth trends of the functions. We choose k_{med} to be a Rational Quadratic kernel which is designed to capture medium and short term variations in the function. Both these kernels are standard in Gaussian process regression and are discussed in detail in [80]. Each of k_{trend} and k_{med} has a collection of hyperparameters which control their behaviour. These hyperparameters are chosen again through maximum likelihood estimation of the observed process. Using such a covariance function should provide an expressive tool for both interpolating and forecasting our score processes. A final remark is that we have an independent Gaussian process for each score process. They all have a common structure of the same mean and covariance function but the hyperparameters for each component score process will be separately estimated. This is in line with the methodology described in [37].

4.3 Simulation Experiment

To demonstrate the effectiveness of our model proposed in Section 4.2 we consider its application to a series of simulated datasets. We specify the data generating process for our simulations below.

4.3.1 Data generating process

We propose simulating data on a spatio-temporal grid. We do so as this is typically how EO data are observed (See Section 1.1). We define the grid by specifying the spatial domain as $\mathcal{S} = [0, 1] \times [0, 1]$ and assume we have 64×96 spatial locations arranged in a grid as our observation locations. The temporal domain we define simply as $\mathcal{T} = [0, 1]$ with possible 60 possible temporal observations evenly spaced within $\mathcal{T}$. This gives a full simulated data dimension of $64 \times 96 \times 60$.

To generate data on such a grid we assume our functional variables are generated as Equation (4.2) with only 3 principal components. These correspond to three different modes of variation which captures the spatial variation in our observed process. The corresponding score processes will capture the temporal variation of each principal component in the observed process. Finally, the sum-product of the score process with the principal components will then result in a fully spatio-temporal process.

We therefore need to simulate the 3 principal components which are two dimensional surfaces over the grid of our domain $\mathcal{S}$ and the 3 score processes which are one dimensional curves over our temporal grid $\mathcal{T}$. To do so we utilise a Gaussian process simulation for both. For simplicity, we restrict our simulations to coming from a zero mean Gaussian process. We simulate the principal components using an isotropic stationary Matérn covariance function and we generate the scores from a Gaussian covariance function. Both of which are standard covariance functions used in Gaussian process regression, more details of which can be found in [80] and the references within. The form of the Matérn covariance function is given in Equation (3.19).

To simulate our 3 principal components, we specify separate length scale hyperparameters, $\rho \in \mathbb{R}^+$, for each component. We keep the shape parameter for the Matérn covariance function fixed for all three components at 2.5. We do so because we wish to simulate data which is smooth across space, and we can adequately adjust the amount of spatial variation by changing the length scale of the process, whilst maintaining this smoothness by keeping the shape parameter fixed. We also note that the principal component variances parameters are fixed to 1.0, this is because the scale is set in the score process as described in Equation (4.2). The length scale parameters used in the simulation study are given in Table 4.1.

Table 4.1 The varying length scale parameter in the Matérn covariance function which is used to simulate the 3 functional principal components in the data generating process.

	Parameter
Component	ρ
1	0.5
2	0.4
3	0.2

An example of the three functional principal components simulated is given in Figure 4.1. As can be noted from the decreasing length scales, the succeeding components are increasingly spatially variable.

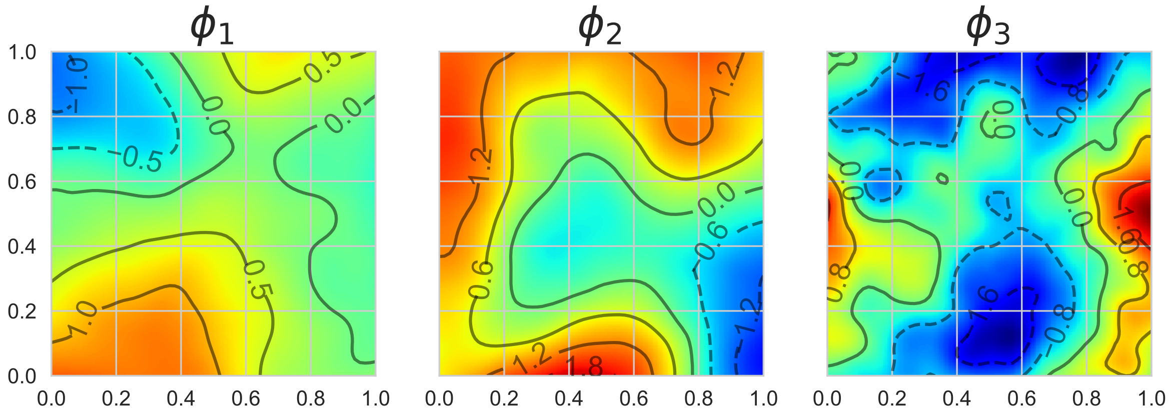


Fig. 4.1 Example of the 3 simulated principal component surfaces from the isotropic stationary Matérn covariance function. Notice the reduced spatial correlation in succeeding components. The simulation study is designed this way to provide difficulty for recovering the true functional principal components.

To simulate the three corresponding score processes we let the variance and length scale parameters vary with each component of our decomposition. These hyperparameters are given in Table 4.2. Again we choose these parameters to emphasise the smoothness and contribution of the leading components to the whole processes. That is, the succeeding score processes are more variable, becoming more difficult to distinguish and forecast.

Table 4.2 The varying length scale and noise parameters in the Gaussian covariance function which is used to simulate the 3 functional principal component scores in the data generating process.

	Parameter	
Component	σ	ρ
1	1.0	0.5
2	0.8	0.3
3	0.5	0.1

An example of the 3 functional principal component score processes is given in Figure 4.2. This highlights the decreasing correlation and the impact of the variance parameter on the score processes.

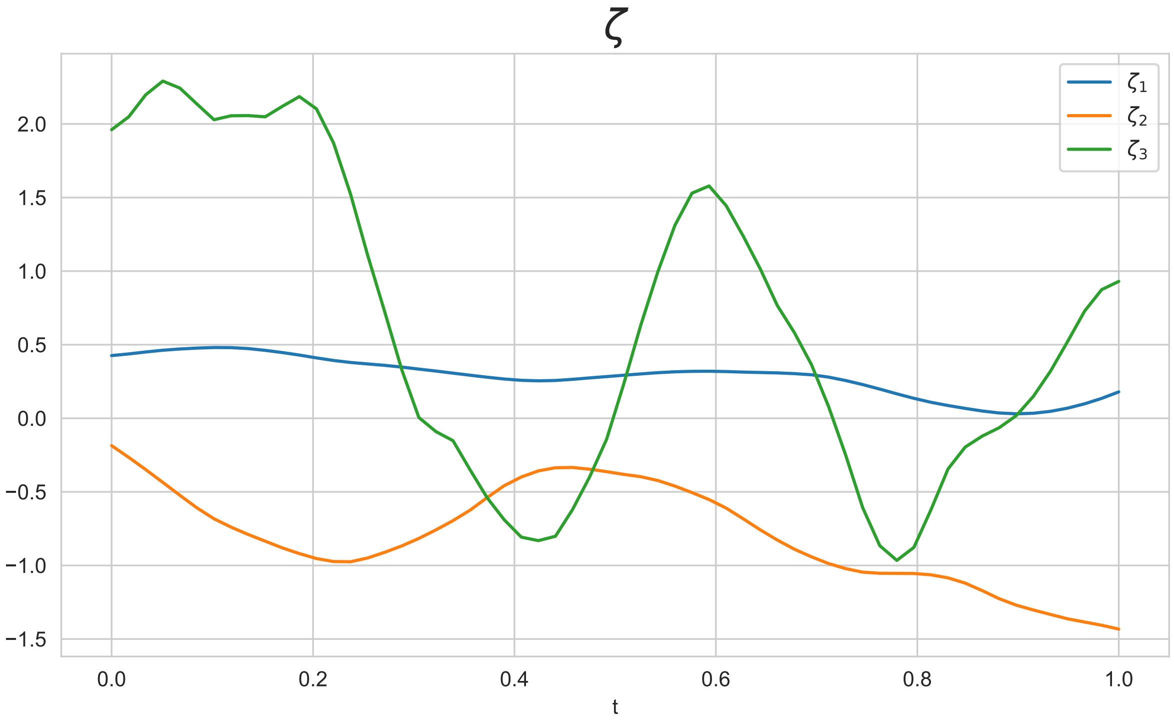


Fig. 4.2 Example of the 3 simulated principal component score functions from the stationary Matérn covariance function. Notice the reduced temporal correlation in succeeding components. The simulation study is designed this way to provide difficulty for interpolation and forecasting such components.

Combining the simulated principal components with their appropriate weightings given by the simulated principal component scores, then gives us a simulation from a truncated version of the model given in Equation (4.2). Figure 4.3 displays a selection of time points of the corresponding functional variable simulations from the components and scores displayed in Figure 4.1 and Figure 4.2 respectively.

As given in Equation (4.1), we do not observe the simulated functional variable directly but rather with an additive noise process $\bar{\varepsilon}_{ij}$ for $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, J$. For our simulation experiments we will consider 4 different types of observational noise; low variance independent noise (LN), high variance independent noise (HN), low variance isotropic spatially correlated noise with short range (LSN), and low variance isotropic spatially correlated noise with long range (HSN). In all cases we assume the noise process is independent over time. We consider the first two (LN and HN) as corresponding to experiments to discuss how the model deals with typical measurement error. The second two noise models (LSN and HSN) correspond to an additional challenge to our models by testing their ability to recover the imagery, even when there is spatially correlated noise which may look similar to that of a single functional variable. This is of special importance in some EO data, such as satellite imagery, where often noise corresponds to atmospheric interference which is spatially correlated, [60]. We consider two cases where the range of the spatial correlation changes, between short range in the LSN noise and long range dependency in the HSN noise. In these two noise processes they correspond to the noise processes being spatially similar to the last and first functional principal component respectively. In both cases we simulate the noise process using a Gaussian process with zero mean and Matérn covariance function. We modulate the smoothness of the spatially dependent noise processes by modifying the length scale parameter of the generating covariance function. Table 4.3 displays the noise process variance and shape parameters where applicable.

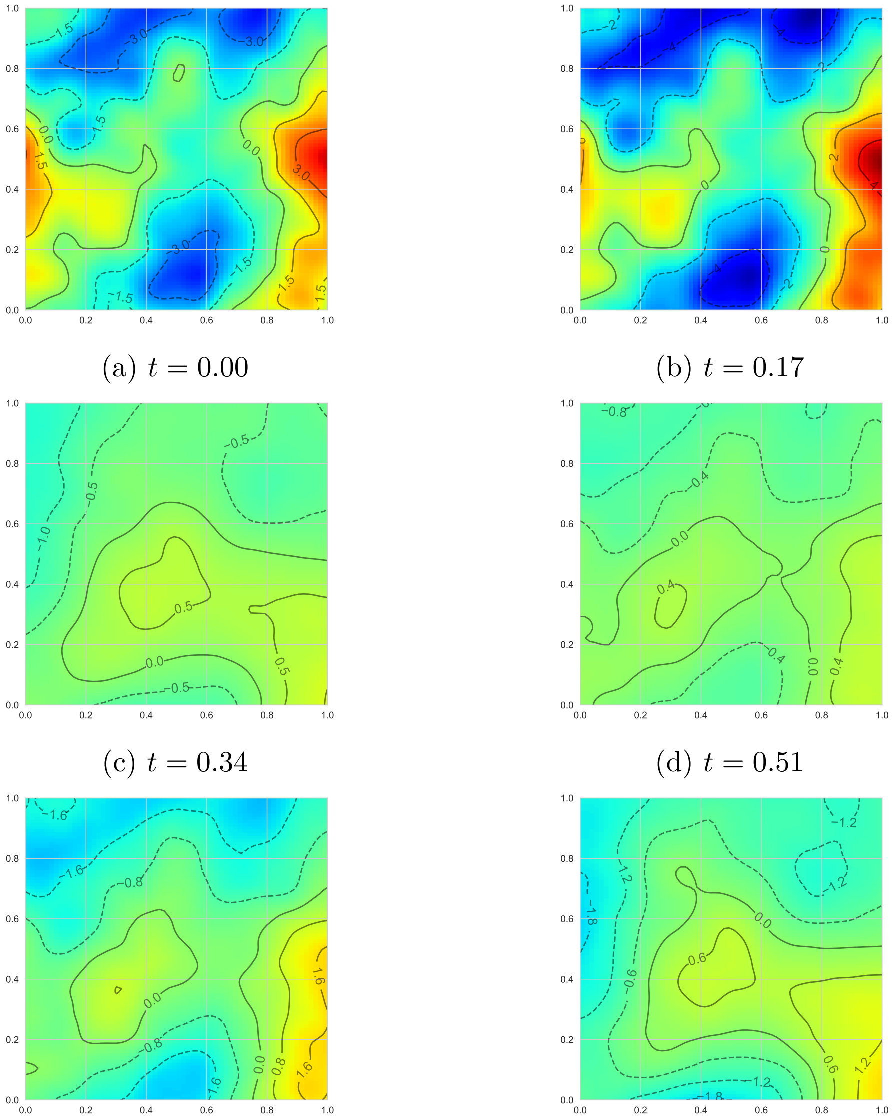
Table 4.3 Variance, length scale and structure parameters for the four different simulated noise processes. Independent noise over space corresponds to a blank ν and ρ parameters.

	Parameter		
Noise Type	σ	ρ	ν
LN	0.2	-	-
HN	1.0	-	-
LSN	0.2	0.2	2.5
HSN	0.2	0.5	2.5

The impact of such noise can be see in Figure 4.4 which displays the imagery observed after adding the various noise types to the unobserved functional variable displayed in Figure 4.3a.

4.3.2 Model parameters

The model described in Section 4.2 has various hyperparameters which control exactly how the model acts. We specify these below, along with justification for such choices where needed.



(a) $t = 0.00$

(b) $t = 0.17$

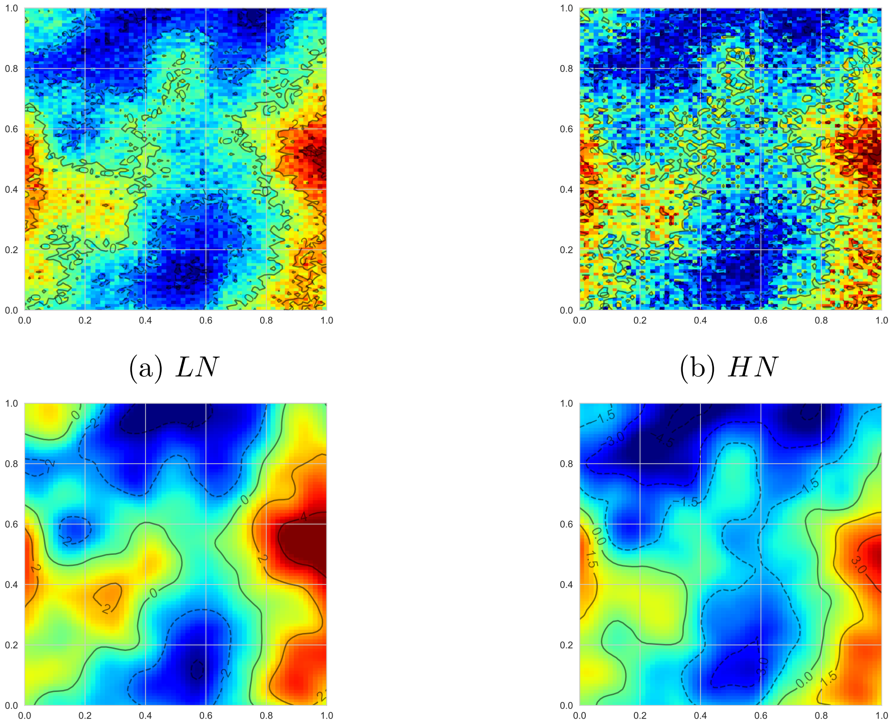
(c) $t = 0.34$

(d) $t = 0.51$

(e) $t = 0.68$

(f) $t = 0.85$

Fig. 4.3 An example simulated functional variables, $\bar{\chi}_j$. for various time points, t_j. These correspond to the simulated functional components and scores in Figure 4.1 and Figure 4.2 respectively. Notice how we see the third principal component prominently at the beginning then fade and reappear in line with its associated score, whereas the first two components are more consistent over time.



(a) *LN*

(b) *HN*

(c) *LSN*

(d) *HSN*

Fig. 4.4 An example of the impact of the various noise structures to the observed simulated data. Each figure highlights adding a noise process to the unobserved functional variable displayed in Figure 4.3a.

The first set of hyperparameters correspond to those of the basis system used for the basis expansion of the functional variables. In our case we limit ourselves to B-spline basis systems of order 4, as described in Section 3.3.1. We choose order 4 B-spline functions as cubic functions are the standard in many applications, [14]. We choose 16 basis functions across each dimension of our surfaces, giving a total number of 256 basis functions for the tensor product basis system, as described in Section 3.3.2. This is chosen as a trade off between flexibility to fit the surface and computational constraints. Obviously the higher number of basis functions, the closer we can recreate the observed data. However the additional computational cost in estimating these coefficients quickly grows as the number of basis functions in each dimension is increased. To penalise the fitting, as described in Section 3.3.2, we use the GCV fitting procedure given in [77] with the tensor product penalty given by [86]. We choose the penalty order to be 1 which essentially places first derivative smoothness over the marginal basis, again such a choice is standard in spline fitting.

The next set of hyperparameters relates to the functional decomposition. We choose to examine a maximum of 4 principal components in our decomposition. That is we set $K = 4$. This is chosen as to be a fairly substantive dimensionality reduction whilst maintaining a degree of representativeness, as we know through our simulation that the first 3 components should capture the majority of the variation. Next we set our MAFR operator L to be the first order derivative. This will again set out preference for smooth surfaces. Again, the first order derivative is chosen as it is often a standard in functional data analysis, [63].

We have already stated we choose the hyperparameters to these processes by maximum likelihood estimation of the Gaussian process. This estimation process is discussed in detail in [80].

The final set of simulation hyperparameters refers to how we split between training and testing data. The training and test data split depends on our objective of either interpolation or forecasting. For interpolation, we randomly select 30 points of our time domain $\mathcal{T}$ as training points and observe the noisy simulated data at these points. The remaining time points then represent the test data which we will evaluate our model performance against. For forecasting, we split the time domain at the 54th time point. Any observation before this point becomes our training data with any point after being the test data which we will forecast for. This gives a possibility of testing long range forecasting whilst maintaining enough training data to possibly infer patterns using the score process models.

4.3.3 Results

We present the results for interpolation and forecasting the simulated data in this section separately, as they correspond to two separate objectives. We repeat the simulation 100 times for both interpolation and forecasting. The simulation results are presented separately for each of our proposed models named FPCA and MAFR. The details of such models are described in Section 4.2 and the references within.

As a comparison to these methods we use a traditional PACE model using time as our functional variable. The details of the PACE approach are set out in Section 3.2.

This is a typical model for such data. For example, [29] uses such an approach on a similar styled dataset. However, this approach completely ignores any spatial dependency by instead treating each pixel as independent. Interpolation and forecasting using this methodology is performed by interpolating and extrapolating the spline basis functions in the model in standard ways, [14]. This is performed by keeping the estimated spline coefficients fixed and evaluating the spline basis system used to model the curve at the unobserved time point. For extrapolation, this essentially is performed by evaluating the last polynomial piece of the spline basis system at the unobserved time point. Therefore it is mainly tailored towards interpolation, as spline regression is well known to interpolate well but extrapolate poorly, [14]. We will denote this model by the term PACE for the following results.

To compare these models we use four metrics. We use two standard measures of mean square error (RMSE) and mean absolute error (MAE). These two are chosen to contrast the influence of any particularly large discrepancies between reconstruction and actual functional variables. The next metric we use is the structure similarity index (SSIM), [79].

The SSIM metric introduced by Wang et al. in 2004 and enhanced in 2009 considers the case that standard metrics such as RMSE and MAE are not indicative of perceived similarity. For example taking a grey-scale image and adding a constant value to the whole image will increase its RMSE and MAE, however to the observer the image will only look brighter. SSIM tries to incorporate structural similarity when comparing imagery to highlight perceived similarity, [78]. The SSIM metric is calculated over various windows of an image. The SSIM between two windows, x and y, of common size is given by:

$$\mathrm{SSIM}\,(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)},$$

where μ_x, μ_y are the respective means of the windows, σ_x, σ_y are the respective standard deviation of the windows, σ_{xy} is the covariance of x and y, and c_1, c_2 are two constants which are calculated based on the dynamic range of the two images under comparison. For further details we refer the reader to [78]. For the SSIM metric a value of one represents perfect similarity, the value of minus one represents perfect negative similarity, hence the closer to one the better for this metric.

The final metric we employ is the Peak Signal to Noise Ratio (PSNR). PSNR is commonly used to quantify image reconstruction quality. It has been used extensively in medical imaging applications. The metric is commonly defined using the MSE between two images, x and y say, then the value of the metric is given by:

$$\mathrm{PSNR}(x, y) = 10\log_{10}\left(\frac{\mathrm{MAX}_x^2}{MSE(x, y)}\right),$$

where MAX_x is the maximum possible value in image x, and $MSE(x, y)$ is the mean square error between x and y. For the PSNR metric a higher value is better.

Interpolation Results

Here we present the metric results for interpolation across our test dataset as described in Section 4.3.2. The results given are the metric values across all unobserved functional variables in the test set which we have interpolated. We present both the average across simulations as well as the standard deviation of the metric values across simulations.

Tables 4.4 displays the reconstruction results for interpolation from the various observations with the different noise processes structures. Discussion of these results can be found in Section 4.3.4.

Forecasting Results

Here we present the metric results for the forecasting ability of our models for our test dataset, as described in Section 4.3.2. The results given are the metric value at h-step ahead forecasts for the unobserved functional variables in the test set where h is one of $1, 3, 6$. A single step ahead corresponds to observing the next image on our temporal grid $\mathcal{T}$ constructed in our data generating process. These correspond loosely to short, medium, and long range forecasts. This provides an overview of the abilities of the model to forecast over various ranges. These h step ahead comparisons are standard in time

Table 4.4 Simulation results for interpolation by noise scenario for each model; PACE, FPCA, and MAFR. Bracketed values correspond to the standard deviation. Bold values illustrate best in class.

		Metric			
Noise	Model	RMSE	MAE	SSIM	PSNR
LN	PACE	0.25 (0.05)	0.20 (0.04)	0.80 (0.07)	28.15 (3.37)
	FPCA	**0.10 (0.05)**	**0.08 (0.04)**	**0.98 (0.02)**	**35.27 (4.59)**
	MAFR	0.12 (0.10)	0.10 (0.08)	0.97 (0.02)	34.66 (5.08)
HN	PACE	0.40 (0.06)	0.32 (0.05)	0.58 (0.09)	24.26 (2.93)
	FPCA	**0.16 (0.05)**	**0.13 (0.04)**	**0.95 (0.03)**	**31.78 (4.23)**
	MAFR	0.19 (0.07)	0.15 (0.06)	0.94 (0.03)	31.06 (4.30)
LSN	PACE	0.50 (0.08)	0.41 (0.07)	0.88 (0.04)	23.49 (2.59)
	FPCA	**0.46 (0.07)**	**0.38 (0.06)**	**0.88 (0.05)**	**23.56 (2.77)**
	MAFR	0.48 (0.08)	0.39 (0.06)	**0.88 (0.05)**	23.45 (2.56)
HSN	PACE	0.50 (0.10)	0.42 (0.09)	**0.93 (0.04)**	**23.65 (3.12)**
	FPCA	0.51 (0.10)	0.42 (0.09)	0.90 (0.05)	22.61 (2.72)
	MAFR	**0.48 (0.10)**	**0.40 (0.09)**	0.92 (0.04)	23.61 (2.71)

series forecasting, [34]. We present both the average across simulations as well as the standard deviation of the metric values across simulations.

Table 4.5 displays the reconstruction results for each model in our simulation studies under our forecasting scenario. Discussion of these results can be found in Section 4.3.4.

Table 4.5 Simulation results for the models ability to forecast unseen functional variables under the FTSM model at $1, 3, 6$ time steps ahead. Bracketed values correspond to the standard deviation. Bold values illustrate best in class.

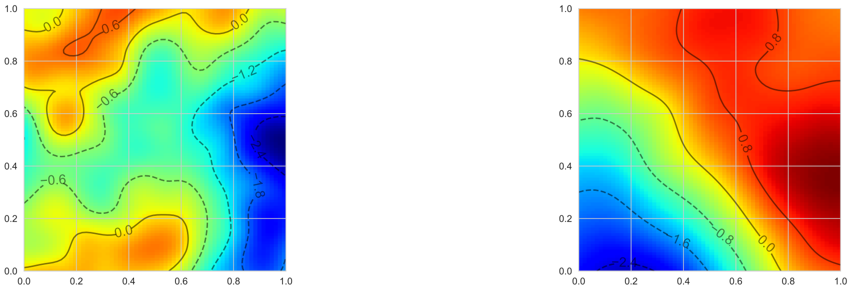
		Metric											
			RMSE			MAE			SSIM			PSNR	
Noise	Model	$h = 1$	$h = 3$	$h = 6$	$h = 1$	$h = 3$	$h = 6$	$h = 1$	$h = 3$	$h = 6$	$h = 1$	$h = 3$	$h = 6$
LN	PACE	0.35 (0.12)	0.68 (0.27)	1.14 (0.49))	0.28 (0.09)	0.54 (0.22)	0.92 (0.40)	0.60 (0.14)	0.41 (0.17)	0.29 (0.19)	20.21 (3.60)	19.18 (2.98)	17.28 (4.03)
	FPCA	0.10 (0.05)	0.28 (0.17)	**0.59 (0.33)**	**0.08 (0.04)**	0.23 (0.14)	**0.48 (0.27)**	**0.94 (0.06)**	**0.90 (0.14)**	**0.74 (0.27)**	**28.42 (7.01)**	**25.72 (5.97)**	**20.38 (6.07)**
	MAFR	**0.10 (0.05)**	**0.28 (0.16)**	0.60 (0.36)	**0.08 (0.04)**	**0.23 (0.13)**	0.49 (0.29)	**0.94 (0.06)**	0.88 (0.15)	0.73 (0.27)	28.04 (6.46)	25.39 (5.60)	20.21 (5.74)
HN	PACE	0.53 (0.18)	0.90 (0.35)	1.36 (0.54)	0.43 (0.14)	0.72 (0.28)	1.09 (0.43)	0.44 (0.11)	0.29 (0.12)	0.19 (0.13)	18.37 (2.76)	17.62 (2.36)	16.17 (3.28)
	FPCA	**0.17 (0.08)**	**0.34 (0.19)**	**0.63 (0.32)**	**0.14 (0.06)**	**0.28 (0.16)**	**0.52 (0.27)**	**0.90 (0.11)**	**0.86 (0.15)**	**0.72 (0.27)**	**25.62 (5.60)**	**23.81 (5.34)**	**19.47 (5.58)**
	MAFR	**0.17 (0.08)**	0.36 (0.19)	0.69 (0.37)	0.14 (0.07)	0.29 (0.15)	0.57 (0.31)	0.88 (0.12)	0.83 (0.17)	0.70 (0.25)	24.72 (5.61)	22.64 (4.88)	18.68 (5.42)
LSN	PACE	0.74 (0.26)	1.21 (0.49)	1.72 (0.64)	0.61 (0.22)	1.00 (0.42)	1.42 (0.54)	0.76 (0.13)	0.63 (0.19)	0.47 (0.26)	18.37 (3.28)	16.54 (3.21)	14.12 (3.57)
	FPCA	**0.54 (0.24)**	**0.70 (0.35)**	**0.90 (0.45)**	**0.44 (0.19)**	**0.58 (0.28)**	**0.74 (0.38)**	0.77 (0.20)	0.70 (0.25)	0.59 (0.31)	**19.86 (4.72)**	**18.52 (4.56)**	**16.53 (4.96)**
	MAFR	0.56 (0.20)	0.74 (0.30)	0.97 (0.44)	0.47 (0.16)	0.61 (0.25)	0.80 (0.37)	**0.78 (0.17)**	**0.72 (0.21)**	**0.61 (0.28)**	19.54 (4.32)	18.29 (4.21)	16.22 (4.42)
HSN	PACE	0.74 (0.29)	1.21 (0.47)	1.71 (0.68)	0.62 (0.25)	1.01 (0.42)	1.43 (0.60)	**0.82 (0.13)**	0.70 (0.21)	0.54 (0.32)	18.45 (3.34)	17.01 (3.37)	14.86 (4.17)
	FPCA	0.60 (0.23)	0.75 (0.34)	**0.94 (0.46)**	0.50 (0.19)	**0.62 (0.29)**	**0.78 (0.39)**	0.79 (0.20)	**0.74 (0.22)**	0.62 (0.29)	19.87 (4.97)	18.83 (4.89)	16.43 (4.75)
	MAFR	**0.59 (0.24)**	**0.75 (0.33)**	0.97 (0.46)	**0.49 (0.21)**	0.63 (0.28)	0.81 (0.40)	0.82 (0.17)	0.76 (0.21)	**0.64 (0.30)**	**20.22 (4.83)**	**18.98 (4.66)**	**16.65 (4.85)**

4.3.4 Discussion

We discuss the preceding results for the FPCA and MAFR models in the following section. We discuss the results relative to the PACE model as described in Section 4.3.3 under both the interpolation and forecasting objectives.

Table 4.4 states the mean and standard deviations of the estimation error under the various metrics discussed in Section 4.3.3 for the interpolation metric. We can see clearly an advantage of using the FPCA model over the PACE model under most metrics in nearly all noise process scenarios. The only exception to this being under the highly structured spatial noise process, denoted by HSN, where the FPCA model is inferior to the MAFR and PACE models; though only slightly. This advantage is most prominent under the independent noise scenarios, where it seems that the additional spatial smoothness constraints that the model enforces nullifies the impact of the spatially independent noise. Similarly, the relative deterioration in the FPCA model under structured noise agrees with this effect. In fact, it is possible that due to the noise process being highly similar to the leading functional principal components in the HSN scenario that the FPCA model may conflate the two, hence giving reduced performance.

The MAFR model has similar performance to that of the FPCA model; as expected, due to it being a rotated version of the FPCA model. It is narrowly but consistently beaten on most noise scenarios, except for the HSN. Here, the fact that the MAFR model prioritises smoothness in its components, [29], has meant that we have overcome some of the trouble that the FPCA model faced in conflating the noise and signal processes. We can see this effect by examining the second functional principal component for both the FPCA and MAFR model for a single simulation under the HSN noise scenario as an example. Figure 4.5 shows exactly this. The actual functional principal components for this simulation are given in Figure 4.1.



(a) FPCA

(b) MAFR

Fig. 4.5 Comparison of the second functional principal component recovered under the FPCA model and the MAFR model for an example simulation under the HSN noise scenario. Notice how the MAFR component is much smoother over the domain.

In addition, this rotation has caused the score processes to be correlated to the point that leading components will have smoother scores, aiding in removing the noise process which will have random walk like behaviour in the scores process, due to it being independent over time. This is illustrated in Figure 4.6 where we can see that although both score processes aren't particularly smooth, the MAFR process exhibits less variation.

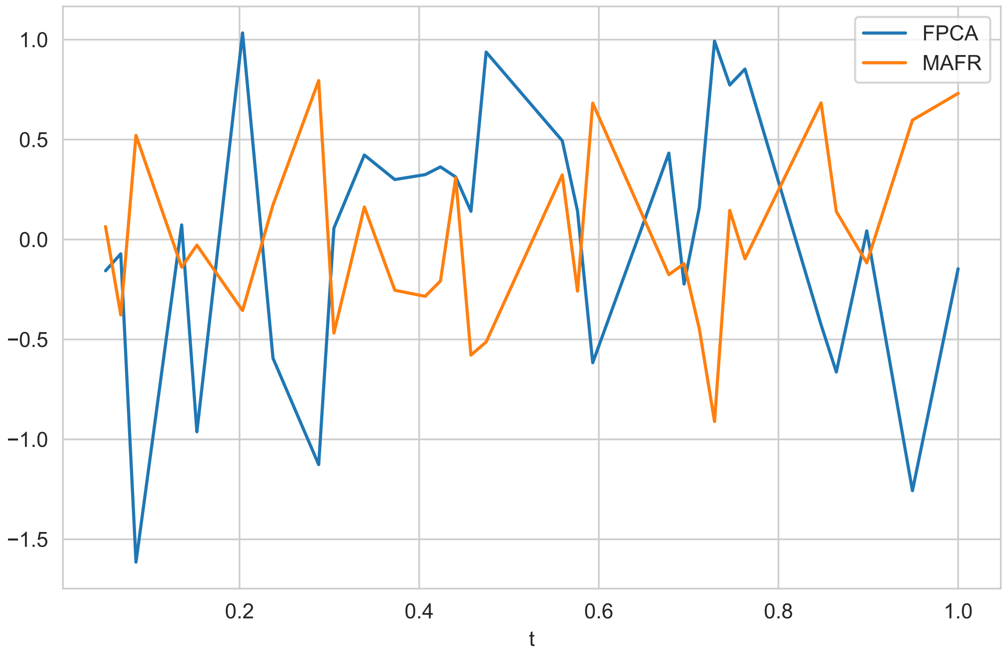
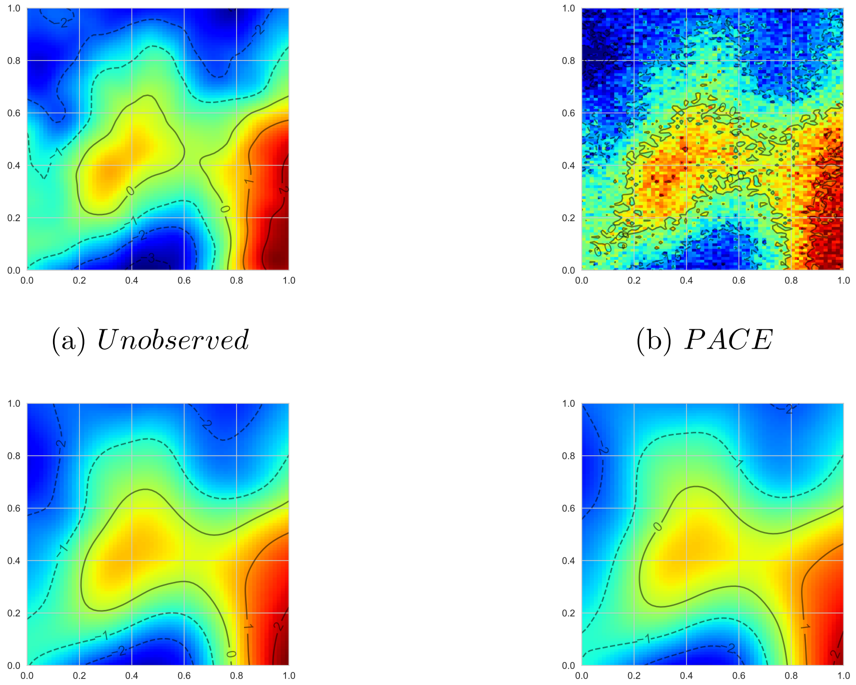


Fig. 4.6 Comparison of the second score process corresponding to the second functional principal component given in Figure 4.5. Notice how the MAFR score process exhibits less variation over time compared to the FPCA process.

The results for the forecasting objective are given in Table 4.5. Here, we see similar results to that which were observed in the interpolation objective. We see both the FPCA and MAFR models outperform the PACE model under most noise scenarios. Similarly to the interpolation results we see the most improvement under spatially independent noise processes. Again, this is due to the added ability of the FPCA and MAFR models to filter out any process which is not particularly smooth over space. Interestingly, we also see a greater improvement in reconstruction for the long term forecast rather than the one step ahead short term forecast relative to the PACE model. This is good as it suggests that the FPCA and MAFR approaches capture the spatio-temporal process in such a way that is easier to forecast. Figure 4.7 highlights this ability by illustrating the unobserved surface and estimated surfaces for each model under the LN noise scenario at three time steps ahead.

Again the FPCA and MAFR approaches lead to similar results, with the FPCA model edging the results for the majority of the noise processes. Similarly to the interpolation results we see that the MAFR model tends to perform better under the HSN noise scenario. This is due to the same reasons as the interpolation results described above.

These results offer a good indication that including spatial information into the model for such datasets can have a material impact on both interpolating and forecasting objectives. The next test for such models is then to see how this improvement translates to



(a) *Unobserved*

(b) *PACE*

(c) *FPCA*

(d) *MAFR*

Fig. 4.7 Example of the impact of the various models to reconstruct the unobserved surface at three steps ahead. The observed process was corrupted by the LN noise process. We note how the PACE model is easily over fitting to the independent noise process whereas the FPCA and MAFR models do not suffer this effect.

datasets not necessarily coming from the assumed data generating procedure. We consider this by applying these model to our EO dataset as discussed in Chapter 2 in the next section.

4.4 EO Application

We apply the same models as used in the simulation model to our CESM-LE dataset as described in Chapter 2. This dataset acts as an example to highlight the performance of the various models described above on a real world dataset. We perform the exact same analysis as in the simulation study but this time apply it to the 40 replications of the CESM-LE data for the various atmospheric variable.

We use the exact same model parameter setup as described in Section 4.3.2 to setup the models for the EO application study. We detail the results of the study in Section 4.4.1.

4.4.1 Results

We present the reconstruction results for the objectives of interpolation and forecasting separately. We repeat the model fitting and reconstruction procedure independently for the 40 realisations of the CESM-LE dataset and provide both the mean metric measures as well as their standard deviations across these realisations. We use the same 4 metrics as used in the simulation study for consistency.

Interpolation Results

Here we present the metric results for interpolation for our test dataset from our CESM-LE EO observations, as described in Chapter 2. See Section 4.3.2 for the construction methodology of the test and training datasets. The results given are the metric values across all unobserved functional variables in the test set which we have interpolated. We present both the average across simulations as well as the standard deviation of metric values across simulations.

Table 4.6 displays the reconstruction results for interpolation from the various atmospheric variables. Discussion of these results can be found in Section 4.4.2.

Table 4.6 Results for interpolation by atmospheric components with various FTSM models for the CESM-LE dataset. Bracketed values correspond to the standard deviation.

		Metric			
Component	Model	RMSE	MAE	SSIM	PSNR
TMQ	PACE	**5.60 (0.52)**	**4.31 (0.47)**	**0.73 (0.05)**	**18.61 (1.49)**
	FPCA	6.08 (0.30)	4.56 (0.25)	0.70 (0.01)	18.19 (0.52)
	MAFR	6.03 (0.30)	4.52 (0.25)	0.70 (0.01)	18.39 (0.75)
PS	PACE	**511.40 (30.69)**	**374.56 (22.04)**	**0.99 (0.00)**	**33.83 (3.35)**
	FPCA	2976.61 (4.81)	1814.34 (6.92)	0.70 (0.00)	21.50 (0.49)
	MAFR	2977.02 (5.68)	1815.04 (7.44)	0.70 (0.00)	21.51 (0.52)
TREFHT	PACE	**6.80 (1.05)**	**5.05 (0.84)**	**0.90 (0.02)**	**23.69 (2.58)**
	FPCA	7.45 (0.35)	5.33 (0.25)	0.84 (0.00)	20.43 (0.44)
	MAFR	7.48 (0.47)	5.38 (0.37)	0.83 (0.00)	20.39 (0.36)
U10	PACE	**1.25 (0.12)**	**0.92 (0.08)**	**0.76 (0.03)**	**19.27 (1.70)**
	FPCA	1.55 (0.03)	1.20 (0.02)	0.55 (0.00)	16.33 (0.77)
	MAFR	1.55 (0.03)	1.19 (0.02)	0.55 (0.00)	16.32 (0.81)

Forecasting Results

Here we present the metric results for the forecasting ability of our models for our test dataset from our CESM-LE EO observations, as described in Chapter 2. See Section 4.3.2 for the construction methodology of the test and training datasets. The results given are the metric value at h-step ahead forecasts for the unobserved functional variables in the test set where h is one of $1, 3, 6$. These correspond to short, medium, and long range forecasts. In the CESM-LE dataset each step corresponds to a month interval. This provides an overview of the abilities of the model to forecast over various ranges. We present both the average across simulations as well as the standard deviation of the metric values across simulations.

Table 4.7 displays the reconstruction results for each model in our simulation studies under our forecasting scenario. Discussion of these results can be found in Section 4.4.2.

Table 4.7 Results for the models ability to forecast unseen functional variables under the FTSM models at $1, 3, 6$ time steps ahead in the CESM-LE dataset. Bracketed values correspond to the standard deviation. Bold represents best in class.

		Metric											
		RMSE			MAE			SSIM			PSNR		
Noise	Model	$h = 1$	$h = 3$	$h = 6$	$h = 1$	$h = 3$	$h = 6$	$h = 1$	$h = 3$	$h = 6$	$h = 1$	$h = 3$	$h = 6$
TMQ	PACE	**3.50 (0.67)**	**5.92 (1.34)**	5.81 (2.20)	**2.41 (0.54)**	**4.71 (1.04)**	4.17 (1.72)	0.45 (0.02)	0.51 (0.03)	0.39 (0.03)	12.22 (0.50)	13.30 (0.48)	11.78 (0.46)
	FPCA	4.11 (0.46)	9.62 (1.17)	5.16 (0.86)	2.71 (0.36)	7.48 (0.99)	3.65 (0.67)	**0.77 (0.01)**	**0.60 (0.05)**	**0.72 (0.03)**	20.30 (0.89)	**16.27 (1.10)**	18.96 (1.03)
	MAFR	4.12 (0.52)	9.61 (1.20)	**5.12 (0.98)**	2.75 (0.42)	7.46 (1.02)	**3.62 (0.77)**	0.77 (0.02)	**0.60 (0.05)**	**0.72 (0.03)**	**20.33 (0.84)**	16.24 (1.11)	**19.00 (0.86)**
PS	PACE	**572.92 (254.02)**	**1416.61 (3179.94)**	4167.61 (19882.51)	**440.70 (243.79)**	**1176.94 (3209.63)**	3919.64 (19920.73)	**0.98 (0.06)**	**0.98 (0.06)**	**0.97 (0.07)**	**32.54 (5.30)**	**33.17 (5.42)**	**33.09 (5.88)**
	FPCA	2963.55 (16.73)	2962.08 (13.00)	2967.76 (11.98)	1801.39 (26.40)	1800.56 (20.45)	1810.16 (24.96)	0.70 (0.00)	0.70 (0.00)	0.69 (0.00)	20.96 (0.46)	21.03 (0.46)	21.16 (0.44)
	MAFR	2962.67 (17.11)	2962.55 (15.21)	**2965.04 (9.17)**	1800.63 (27.52)	1802.48 (25.31)	**1804.88 (20.88)**	0.70 (0.00)	0.70 (0.00)	0.69 (0.00)	20.96 (0.36)	21.03 (0.38)	21.17 (0.35)
TREFHT	PACE	10.13 (0.59)	27.36 (1.16)	30.60 (1.13)	6.90 (0.36)	18.52 (0.80)	20.17 (0.79)	0.81 (0.01)	**0.81 (0.01)**	0.59 (0.01)	17.48 (0.54)	18.04 (0.68)	17.03 (0.49)
	FPCA	**5.87 (1.31)**	**10.75 (1.66)**	**4.42 (1.05)**	**4.03 (0.89)**	**7.66 (1.28)**	**3.34 (0.83)**	**0.85 (0.02)**	0.77 (0.03)	**0.85 (0.01)**	**21.59 (1.39)**	**18.80 (1.59)**	**21.56 (0.77)**
	MAFR	6.29 (1.05)	11.30 (1.36)	4.65 (1.18)	4.36 (0.73)	8.12 (1.15)	3.56 (1.01)	0.84 (0.01)	0.76 (0.02)	0.84 (0.02)	21.19 (1.00)	18.26 (1.15)	21.40 (0.85)
U10	PACE	**1.21 (0.07)**	**1.51 (0.12)**	**1.27 (0.11)**	**0.91 (0.06)**	**1.11 (0.10)**	**0.92 (0.08)**	**0.78 (0.02)**	**0.71 (0.03)**	**0.74 (0.03)**	**21.47 (0.74)**	**19.84 (0.88)**	**21.31 (0.82)**
	FPCA	1.43 (0.06)	1.75 (0.11)	1.41 (0.06)	1.12 (0.05)	1.34 (0.09)	1.10 (0.05)	0.56 (0.02)	0.50 (0.03)	0.56 (0.02)	16.50 (0.83)	16.05 (0.85)	17.03 (0.74)
	MAFR	1.43 (0.05)	1.75 (0.09)	1.41 (0.04)	1.12 (0.05)	1.34 (0.08)	1.10 (0.04)	0.56 (0.02)	0.49 (0.02)	0.56 (0.02)	16.37 (0.61)	15.93 (0.59)	16.91 (0.53)

4.4.2 Discussion

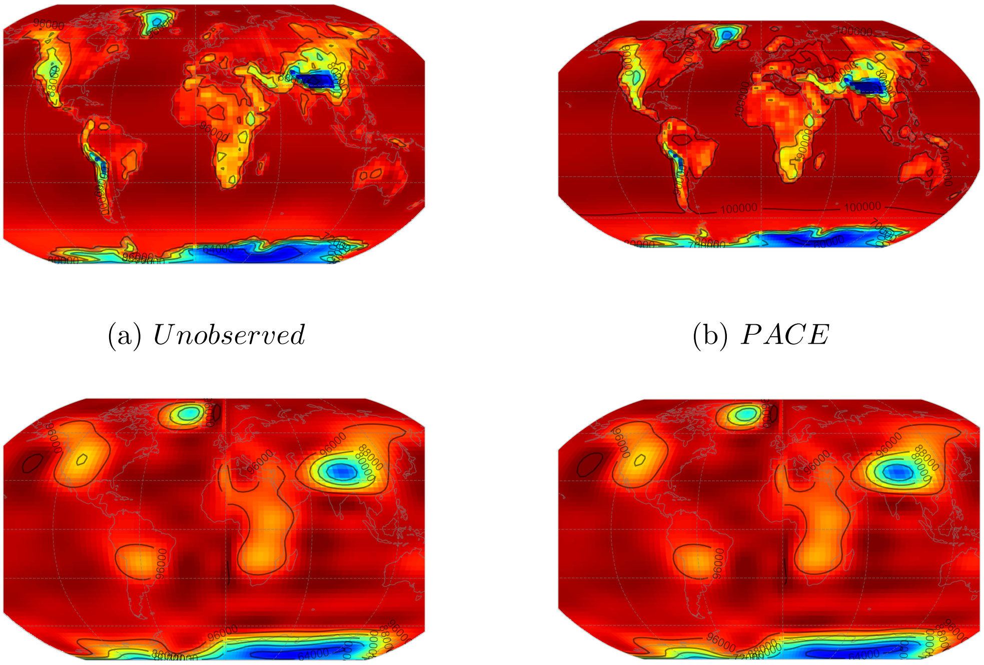
We discuss the preceding results for the FPCA and MAFR models with application on the CESM-LE dataset in the following section. We discuss the results relative to the PACE model as described in Section 4.3.3 under both the interpolation and forecasting objectives.

Evidently, from Table 4.6, the PACE model outperforms both the FPCA and MAFR models for all atmospheric variables studied across all metrics. This is particularly the case for the PS and U10 variables. The reason for this divergence from the PACE model can be seen by considering an example from the PS variable. Figure 4.8 gives the unobserved surface and the estimated surfaces from the PACE, FPCA, and MAFR models. We can clearly see from this that the FPCA and MAFR models, although capturing large scale spatial patterns, they fail to capture the small scale spatial variation which is present in the datasets. In this case this causes very divergent results as seen by the metrics in Table 4.6. The reason for this lack of flexibility of the FPCA and MAFR models is that the number of basis functions to represent the surface is not high enough to capture such spatial variability. Therefore, an obvious way to possibly alleviate this problem is to simply increase the dimension of the spline representation of these surfaces. However, this comes with additional computation cost. Another possibility is to just consider the FPCA and MAFR models on smaller sections of the domain which may vary less. It is reassuring however to see that the FPCA and MAFR models do capture the large scale spatial variation well. In fact, as the FPCA and MAFR metric values tend to vary less than those of the PACE model, if one is interested in large scale variation these models may well still be preferred.

The forecasting results, given in Table 4.7, show similar results. However, here we see the increasing impact of the FPCA and MAFR models. Especially in the atmospheric variables of TMQ and TREFHT we see an advantage in using the FPCA and MAFR models. Here we see, slightly inverse to the interpolation results, that although the PACE model can deal with small scale spatial variation it struggles to model coherence through time using the spline extrapolation. Whereas the FPCA and MAFR models produce score processes which can evidently be more easily forecast using the Gaussian process methodology outlined in Section 4.2.2. Additionally, we see that the RMSE and MAE metrics get worse for the PACE methodology as h increases, whereas the FPCA and MAFR methodologies are more consistent for larger forecasting steps. Again, we see little difference between the FPCA and MAFR models in both the interpolation and forecasting objectives.

4.5 Summary

In this chapter we have considered a model for EO datasets based on the FTSM technique discussed in Section 3.4. We have considered, relatively uniquely, the idea that we consider our dataset as a time series of surfaces over our spatial domain and use the functional time



(a) *Unobserved*

(b) *PACE*

(c) *FPCA*

(d) *MAFR*

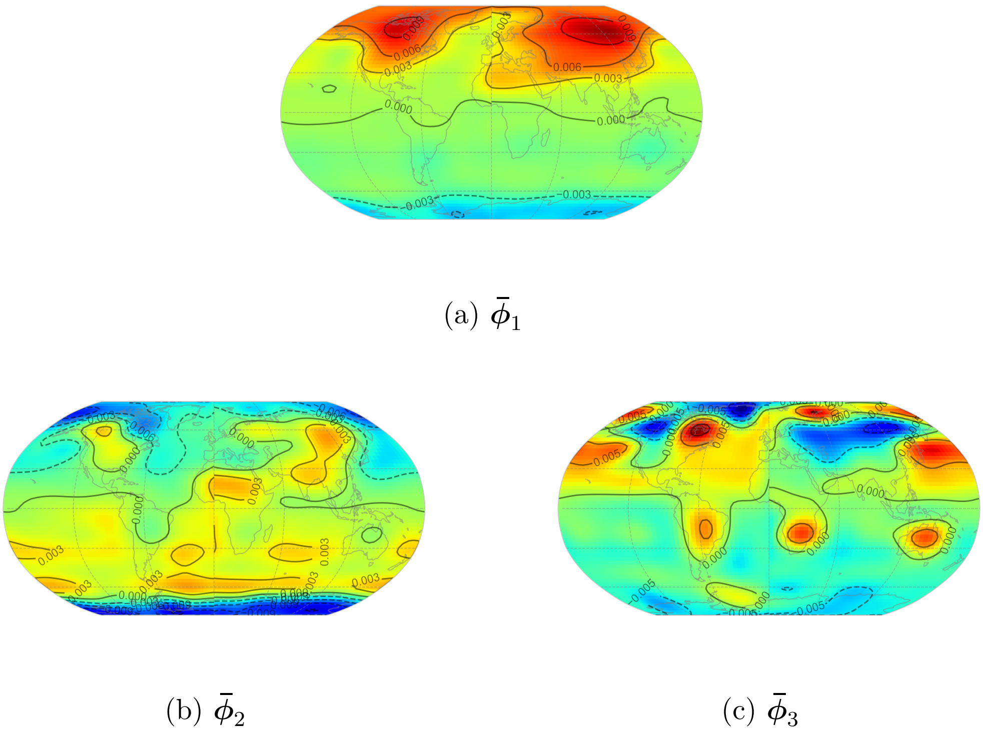
Fig. 4.8 Example of the reconstruction ability of the various models for the PS atmospheric variable component of the CESM-LE dataset. Notice how the FPCA and MAFR models miss the small scale spatial variation present in the unobserved surface. They have particular issue in recreating the abrupt changes at the sea-land boundary.

series methodology discussed by [37] to provide an elegant way to forecast such datasets. We have also considered a rotation to such models which, as discussed in Section 4.2.1, may perhaps promote better forecasting ability. This is compared against standard techniques that treat the data as a collection of independent functions over time, with each function corresponding to a spatial location.

We have seen in Section 4.3.3 that on simulated datasets this technique works well. We have compared these techniques under a variety of noise processes, including spatially structured noise which is often more realistic than independent noise processes in EO data. We find, on the whole, they work at least equally well as the standard methodology which ignores spatial dependency between observations for interpolation, and outperforms this technique when forecasting.

However, on the CESM-LE data this technique falls down due to its difficulty in recreating small spatial scale variation. As discussed in Section 4.4.2, this is due to our smoothing methodology to represent each observed surface using B-spline basis functions. Given ample computation time this can be relieved by extending the dimension of this basis system. However this is not always possible, and represents a real limitation of such methodologies.

A final advantage of such FTSM techniques for EO data is their ability to reduce the dataset dimension by introducing functional principal components which have a spatial domain. These components can help to inform about modes of variation that are occurring



(a) $\bar{\phi}_1$

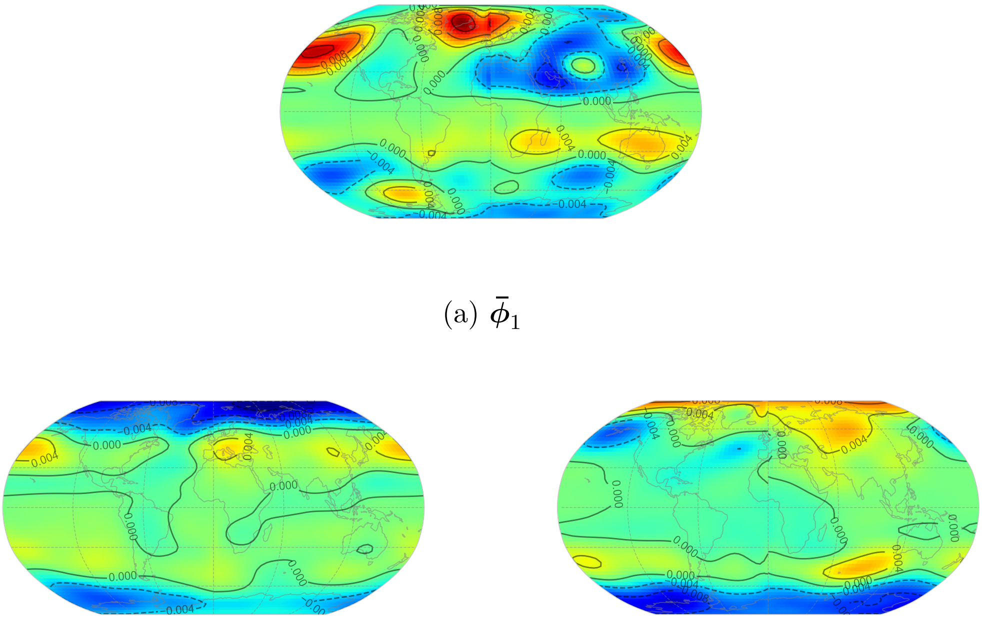
(b) $\bar{\phi}_2$

(c) $\bar{\phi}_3$

Fig. 4.9 Example of the functional principal components generated by the FPCA model with the TREFHT variable of the CESM-LE dataset. Notice how each component shows a different mode of spatial variation present in the process. Such decompositions like these can be useful to understanding the process as a whole.

over space. Such dissections of the data can be useful in understanding the processes under examination. For example, Figure 4.9 gives the first three components for the TREFHT atmospheric variable of the CESM-LE dataset. We can see clearly how the first corresponds to large scale variation between regions in the northern hemisphere whilst the second contains more localised areas of variation. We have further seen how the MAFR rotation can help to promote smooth functional principal components which may aide interpretability. Another example,which shows similar results for the pressure variable is given in Figure 4.10. Here, one can see clearly how the first component focuses on various regions across the globe excluding the poles, whereas the second and third components focus mainly on the polar regions.

The standard PACE methodology can be seen to do well on the real world EO dataset due to its ability to capture small scale variations. This occurs essentially because we treat each spatial location independently. However, we have seen from both the FPCA and MAFR models that incorporating spatial information can be useful in recovering unobserved surfaces. Utilising spatial dependency also has the added benefit of helping to ignore unstructured noise processes. A combination of both methodologies may then be desirable; that is, to incorporate spatial information into the PACE model. The challenge is then to do so in such a way that keeps the model performing well where there exists small scale spatial variation. In the following chapters we consider such a model.



(a) $\bar{\boldsymbol{\phi}}_1$

(b) $\bar{\boldsymbol{\phi}}_2$

(c) $\bar{\boldsymbol{\phi}}_3$

Fig. 4.10 Example of the functional principal components generated by the FPCA model with the PS variable of the CESM-LE dataset. Notice how each component shows a different mode of spatial variation present in the process. Such decompositions like these can be useful to understanding the process as a whole.

Chapter 5

Correlated Principal Analysis through Conditional Expectation

The foray into modelling using the spatial dimensions as the functional dimension in Chapter 4 gave us two clear findings. Firstly, that spatial information is often useful in reconstruction and ignoring the spatial correlation observed between functional observations is throwing away information. Secondly, smoothing across the spatial dimension can be problematic since it is incredibly easy to over smooth and lose important spatial details in the reconstructions. We note from Chapter 4 that the standard model, using time as the functional domain, managed to capture high levels of spatial detail since it treats each function as independent. The downside to this model is that unobserved spatial locations cannot be reconstructed as we have no information about how to interpolate between functional observations. One natural way to consider modelling such functional data is then to extend this model but to explicitly include the correlation between the observations in the model. We discuss these models in this chapter, we start with a discussion on models where the correlation between functional observations is assumed to be only spatial.

5.1 Spatially Correlated Functional Data

The CESM-LE dataset, as described in Chapter 2, provides a perfect example of spatially correlated functional data. Take Figure 5.1 for example, which shows the functional observations of the reference height temperature at 6 locations; the United Kingdom (UK), Ireland (IE), France (FR), Colombia (CO), Venezuela (VE), and Ecuador (EC).

Clearly, we can see locations in a similar vicinity having similar temperature patterns. The European countries have a much larger range and overall lower mean temperature than those of the South American countries. However, there are some similarities between all countries, such as the periodic nature of the functions, which indicate that there is some share of information across even large spatial distances.

Incorporating spatial information into the models such as PACE, described in Section 3.2, is therefore a natural extension. One such approach to this has been considered

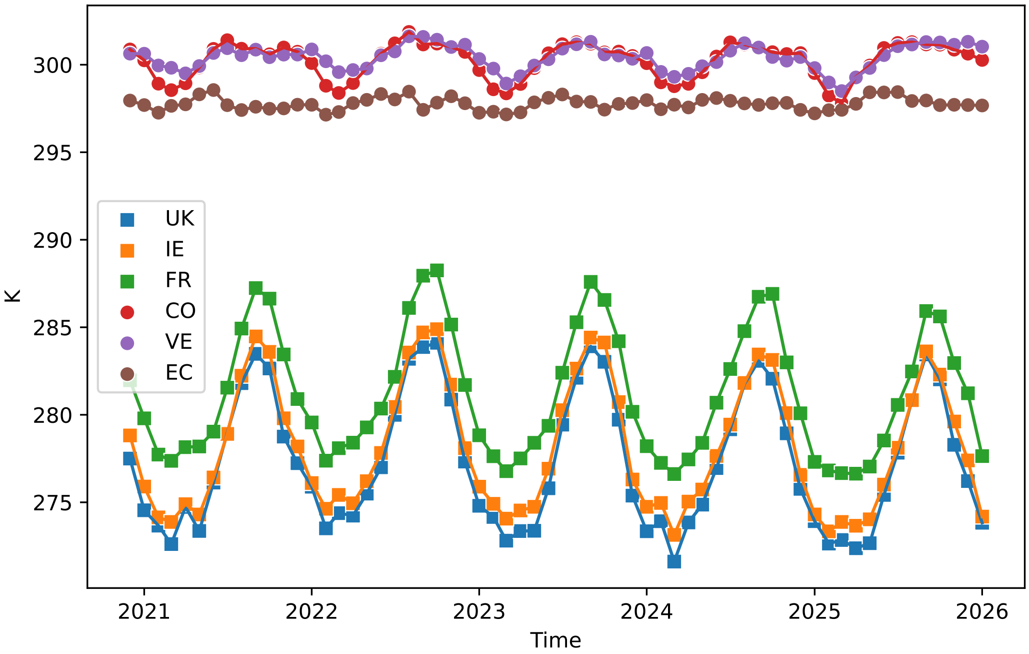


Fig. 5.1 Example of functional observations reference height temperature across the globe from the CESM-LE dataset. Notice that locations close to each other in the globe tend to have more similar structure.

by Liu et al. in [48]. We will describe their model below and use this as our basis for modification and expansion in our proposed model for correlated functional data.

5.1.1 Spatial Principal Analysis through Conditional Expectation

A natural way to incorporate spatial correlation in our functional observation from the PACE model is to adjust how we define the score processes in Equation (3.4). The PACE model, [90], implicitly uses the fact that $\mathbb{E}\left(\xi_{ik}\xi_{jk}\right) = 0$ for $i \neq j$ and $k = 1, 2, \cdots, K$. The Spatial PACE model (SPACE), [48] explicitly incorporates a model for this covariance. In particular they consider the following form:

$$\operatorname{Cov}\left(\xi_{ip}, \xi_{jq}\right) = \begin{cases} \lambda_k \rho_{ijk} \text{ for } p = q = k. \\ 0 \text{ otherwise.} \end{cases}, \tag{5.1}$$

where $0 \leq i, j \leq N$ index the functional realisation and $0 \leq p, q \leq K$ index the component. The correlation is induced by specifying the form of ρ_{ijk} which acts as a spatial correlation factor in [48]. As such, it is useful to explicitly describe the association by writing $\rho_{ijk} = \rho_k\left(\boldsymbol{s}_i, \boldsymbol{s}_j\right)$ where $\boldsymbol{s}$ is an element in the spatial domain $\mathcal{S}$ as discussed in Chapter 1. Often the correlation structure may have an assumed parametric form, in which case we write $\rho_k\left(\boldsymbol{s}, \boldsymbol{s}'\right) = \rho_k\left(\boldsymbol{s}, \boldsymbol{s}'; \boldsymbol{\theta}\right)$ where we collect any hyperparameters for the parametric form into $\boldsymbol{\theta}$. The covariance between $\mathcal{X}_i$ and $\mathcal{X}_j$ can be found as:

$$\operatorname{Cov}\left(\mathcal{X}_i(t), \mathcal{X}_j(t')\right) = \boldsymbol{\phi}^{\intercal}(t)\Sigma\left(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j\right)\boldsymbol{\phi}(t'), \tag{5.2}$$

where $\Sigma\left(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j\right) = \operatorname{Diag}\left(\lambda_1\rho_1(\boldsymbol{s}_i, \boldsymbol{s}_j), \lambda_2\rho_2(\boldsymbol{s}_i, \boldsymbol{s}_j), \cdots, \lambda_K\rho_K(\boldsymbol{s}_i, \boldsymbol{s}_j)\right)$ is the covariance of the score processes. We note that if $\Sigma\left(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j\right) = \operatorname{Diag}\left(\lambda_1, \lambda_2, \cdots, \lambda_K\right)\delta_{ij}$ in Equation (5.2) then the SPACE model reduces to the PACE model and corresponds to independent realisations of $\mathcal{X}$. Liu et al. then obtain the equivalent of Equation (3.5) for spatially correlated functional data as:

$$\check{\tilde{\boldsymbol{\xi}}} = \mathbb{E}\left(\tilde{\boldsymbol{\xi}}|\tilde{\boldsymbol{Y}}\right) = \boldsymbol{\Sigma}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{Y}}\right)\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{Y}}, \tilde{\boldsymbol{Y}}\right)^{-1}\left(\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{\mu}}\right), \tag{5.3}$$

where $\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{Y}}\right)$ represents the covariance between the vector of scores at the appropriate locations with the observed functional data. Similarly , $\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{Y}}, \tilde{\boldsymbol{Y}}\right)$ represents the covariance between the vector of observed functional data with itself. We have used the $\tilde{\cdot}$ notation as in [48] to denote these vectors. The breakdown of the various terms in Equation (5.3) is given below and follows [48]:

$$\begin{aligned}
\boldsymbol{y}_i &= \left(y_i(t_{i1}), y_i(t_{i2}), \cdots, y_i(t_{iJ_i})\right)^{\intercal}, \\
\tilde{\boldsymbol{Y}} &= \left(\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_N\right)^{\intercal}, \\
\boldsymbol{\mu}_i &= \left(\mu_i(t_{i1}), \mu_i(t_{i2}), \cdots, \mu_i(t_{iJ_i})\right)^{\intercal}, \\
\tilde{\boldsymbol{\mu}} &= \left(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \cdots, \boldsymbol{\mu}_N\right)^{\intercal}, \\
\boldsymbol{\xi}_i &= \left(\xi_{i1}, \xi_{i2}, \cdots, \xi_{iK}\right)^{\intercal}, \\
\tilde{\boldsymbol{\xi}} &= \left(\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \cdots, \boldsymbol{\xi}_N\right)^{\intercal},
\end{aligned}$$

Here we see the extension from PACE model to SPACE where we use all data from the various functional observations at separate locations to give the best linear unbiased predictors. This captures the correlation between functional observations induced by the spatial correlation in the K score processes. We can see the impact of the score processes more clearly if we rewrite Equation (5.3) as in [48]. Equation (5.3) can be rewritten in a form similar to Equation (3.5) using the structure of $\mathcal{X}$.

$$\check{\tilde{\boldsymbol{\xi}}} = \boldsymbol{\Sigma}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\xi}}\right)\tilde{\boldsymbol{\phi}}^{\intercal}\left(\tilde{\boldsymbol{\phi}}\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\xi}}\right)\tilde{\boldsymbol{\phi}}^{\intercal} + \sigma_{\varepsilon}^2\mathbf{1}\right)^{-1}\left(\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{\mu}}\right),$$

where $\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\xi}}\right)$ is the covariance between the vector of score values at the appropriate locations. Due to the construction in the SPACE model this has a relatively nice form as:

$$\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\xi}}\right) = \tilde{\boldsymbol{\rho}} \bullet \left(\mathbf{1}_{N\times N} \otimes \boldsymbol{\Lambda}\right), \tag{5.4}$$

where $\bullet$ represents the element wise multiplication of the two matrices, $\boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \lambda_2, \cdots, \lambda_K)$ is the diagonal matrix of eigenvalues, and we construct $\tilde{\boldsymbol{\rho}}$ as follows:

$$\begin{aligned}\boldsymbol{\rho}_{ij} &= \operatorname{Diag}\left(\rho_1(\boldsymbol{s}_i, \boldsymbol{s}_j), \rho_2(\boldsymbol{s}_i, \boldsymbol{s}_j), \cdots, \rho_K(\boldsymbol{s}_i, \boldsymbol{s}_j)\right), \\ \tilde{\boldsymbol{\rho}} &= \left[\boldsymbol{\rho}_{ij}\right],\end{aligned}$$

where $[\cdot_{ij}]$ represents a matrix with ij^{th} entry being $\cdot_{ij}$. As discussed in [48] the covariance of the score process $\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\xi}}\right)$ can take a simpler form if we assume that the correlation structure across all components are the same. That is if $\rho_{ijk} = \rho_{ij} = \rho(\boldsymbol{s}_i, \boldsymbol{s}_j)$ for all $k = 1, 2, \cdots, K$ we have the following form:

$$\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\xi}}\right) = \boldsymbol{\rho} \otimes \boldsymbol{\Lambda},$$

where $\boldsymbol{\rho} = [\rho_{ij}]$. This is a particularly strong assumption which is unlikely to be observed in practice.

By substituting estimates for the various terms in Equation (5.4) the estimate of $\tilde{\boldsymbol{\xi}}$ is derived, namely:

$$\hat{\tilde{\boldsymbol{\xi}}} = \hat{\boldsymbol{\Sigma}}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\xi}}\right) \hat{\tilde{\boldsymbol{\phi}}}^{\intercal} \left(\hat{\tilde{\boldsymbol{\phi}}} \hat{\boldsymbol{\Sigma}}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\xi}}\right) \hat{\tilde{\boldsymbol{\phi}}}^{\intercal} + \hat{\sigma}_\varepsilon^2 \mathbf{1}\right)^{-1} \left(\tilde{\boldsymbol{Y}} - \hat{\tilde{\boldsymbol{\mu}}}\right),$$

where $\hat{\cdot}$ represents the estimate of $\cdot$. In particular, $\hat{\boldsymbol{\Sigma}}\left(\tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\xi}}\right) = \hat{\tilde{\boldsymbol{\rho}}} \bullet \left(\mathbf{1}_{N \times N} \otimes \hat{\boldsymbol{\Lambda}}\right)$ is our estimated score covariance. This highlights the first significant change from the PACE methodology. As discussed in [48] the same estimation methodology as in PACE can be used to estimate the eigenfunctions, eigenvalues and noise variance. See Chapter 3 for details to these estimations under the PACE model. There is some need to show that such estimators are consistent under spatially correlated data, which [48] show for locally linear smoothers and we shall show for spline smoothers in the following work. Ignoring this for the time being, the additional work under the SPACE model is the estimation of the correlation values that construct Equation (5.4). In Liu et al. SPACE model these are estimated using the following approach. For a more in depth discussion, we refer the reader to [48].

Firstly, the construction of cross-covariances are required. The cross-covariance, $G_{ij} = \operatorname{Cov}(\mathcal{X}_i, \mathcal{X}_j)$, is the covariance between the i^{th} and j^{th} functional variable. This has the form given in Equation (5.2). If we further assume $\rho_1(\boldsymbol{s}_i, \boldsymbol{s}_j) > \rho_2(\boldsymbol{s}_i, \boldsymbol{s}_j) > \cdots > \rho_K(\boldsymbol{s}_i, \boldsymbol{s}_j)$ then the sequence $\{\boldsymbol{\rho}_k(\boldsymbol{s}_i, \boldsymbol{s}_j)\}_{k=1}^K$ are eigenvalues of G_{ij}. Therefore ρ_{ijk} can be estimated as the ratio of the k^{th} eigenvalue of cross-covariance G_{ij} to the k^{th} eigenvalue of the covariance G. That is:

$$\hat{\rho}_{ijk} = \frac{\hat{\lambda}_k(\boldsymbol{s}_i, \boldsymbol{s}_j)}{\hat{\lambda}_k},$$

where $\hat{\lambda}_k(\boldsymbol{s}_i, \boldsymbol{s}_j)$ is the k^{th} eigenvalue estimate from the decomposition of cross covariance G_{ij}. A series of empirical correlation factors $\hat{\rho}_{ijk}$ can then be used to estimate the hyperparameters in any parametric form of the correlation structure as needed. In [48] they use a quasi-Newton method (BFGS, [19]) to estimate hyperparameters by minimising the sum of squared differences between empirical and fitted correlations.

The SPACE model is shown to be effective, identifiable, and outperforms the PACE model in [48]. In particular, it performs more accurate gap filling for unobserved trajectories on real world datasets. These properties suggest it is a great framework to model spatially correlated functional data. However we note some possible limitations of the model as proposed by [48]. Firstly, the estimation procedure of the SPACE model for the score hyperparameters requires the forming of possibly numerous cross covariance surfaces. These may become either computationally tiresome to compute, as each needs smoothing and decomposing into its eigenvalues, or there may be insufficient data to obtain an accurate representation of the true cross-covariance. Secondly, the SPACE and PACE models both require the need of estimating the score process values at the prediction location, before then using this to reconstruct the unobserved trajectory. This is a slightly convoluted approach, as typically the score process is not what the end user needs but rather the full reconstruction is the useful quantity to estimate. Finally, [48] uses local linear smoothers for representing smooths for both the mean and covariance surfaces, and proves asymptotic properties under this kind of linear smoother for estimation of these surfaces under spatially correlated functional data. As discussed in Chapter 3 we prefer the properties of spline smoothing as an approach. To the authors knowledge there are no such asymptotic results for the mean and covariance smoothing under spatially correlated data using a regularised spline smoother.

In the following section we provide a simple framework based on the SPACE model which provides a more flexible way to view and estimate such functional data. We aim to overcome the limitation of estimating multiple cross-covariances and remove the need for the intermediate estimation of the score processes by considering the model in the context of a Gaussian process. We also show the asymptotic properties of using a spline smoother on correlated functional data in the process.

5.2 Correlated Principal Analysis through Conditional Expectation

To begin the extension to the SPACE model, discussed in Section 5.1.1, we restrict ourselves to the notation that $\boldsymbol{s} \in \mathcal{S} \subset \mathbb{R}^2$ for simplicity. In this case we are now considering, as in the SPACE model to incorporate spatial correlation between observations. To do this we employ a similar model as in SPACE, however we view this using Gaussian processes. In particular we will specify that each score process $\xi_k(\boldsymbol{s})$ be a Gaussian process. In doing so we will generate a framework which we will argue constitutes a more efficient, flexible, and robust framework than is discussed in SPACE.

Without further ado, we let ξ_k be a zero mean Gaussian process, which we denote as:

$$\xi_k : \mathcal{S} \to \mathbb{R},$$
$$\xi_k \sim \mathcal{GP}\left(0, a_k\right),$$

where $a_k : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is the kernel function of the k^{th} Gaussian process. This kernel function is responsible for determining the spatial correlation of our functional random variables. In fact, we can assume without loss of generality that the variance of this kernel function is λ_k. Therefore, we actually only need to specify a correlation function which will determine the correlation of our functional random variable. We also follow the convention of [48] and specify that the cross covariance between the K Gaussian process is zero. That is:

$$\text{Cov}\left(\xi_p(\boldsymbol{s}), \xi_q(\boldsymbol{s}')\right) = \begin{cases} a_k(\boldsymbol{s}, \boldsymbol{s}') \text{ if p=q=k} \\ 0 \text{ otherwise.} \end{cases}, \tag{5.5}$$

We note that this is equivalent to the SPACE model if $a_k(\boldsymbol{s}, \boldsymbol{s}') = \lambda_k \rho_k(\boldsymbol{s}, \boldsymbol{s}')$ which follows from equating Equation (5.5) and Equation (5.1). With such a framework for the score processes we can now look back at the whole process of generating the functional random variable $\mathcal{X}$ with the Gaussian process in mind and including the spatial coordinate as a parameter rather than an index.

$$\mathcal{X}\left(\boldsymbol{s}, t\right) = \mu(t) + \sum_{k=1}^{K} \xi_k(\boldsymbol{s})\phi_k(t), \tag{5.6}$$

where we now consider ξ_k to be a Gaussian process over the spatial domain $\mathcal{S}$. Using this structure for $\mathcal{X}$ we can view $\mathcal{X}$ as being drawn from a larger Gaussian process.

$$\mathcal{X} : \mathcal{S} \times \mathcal{T} \to \mathbb{R},$$
$$\mathcal{X} \sim \mathcal{GP}\left(\mu, a_{\mathcal{X}}\right),$$

where we can construct $a_{\mathcal{X}} : \mathcal{S}^2 \times \mathcal{T}^2 \to \mathbb{R}$,the kernel function, as follows:

$$a_{\mathcal{X}}\left(\boldsymbol{s}, t, \boldsymbol{s}', t'\right) = \boldsymbol{\phi}^{\intercal}(t)\text{Diag}\left(a_1(\boldsymbol{s}, \boldsymbol{s}'), a_2(\boldsymbol{s}, \boldsymbol{s}'), \cdots, a_K(\boldsymbol{s}, \boldsymbol{s}')\right)\boldsymbol{\phi}(t'), \tag{5.7}$$

and $\boldsymbol{\phi}(t) = \left(\phi_1(t), \phi_2(t), \cdots, \phi_K(t)\right)^{\intercal}$ is the K length vector of eigenfunctions evaluated at t. The mean function μ remains the same mean function as discussed in the SPACE and PACE models. There is possible scope to extend this mean function to be over spatial domain as well as the temporal domain, but we will focus on a mean function which is constant over the space. The kernel function $a_{\mathcal{X}}$ is a structured kernel comprising of the K eigenfunctions of $\mathcal{X}$ and the K score parametric kernel functions. As such we can view the kernel as having essentially K lots of hyperparameters, $\{\boldsymbol{\theta}_k\}_{k=1}^{K}$ that will need estimation to determine the parametric score kernel functions. In addition to this we need to estimate the K eigenfunctions and the mean function. In the following section we discuss the estimation of the mean function.

The discussion of the estimation of the eigenfunctions and hyperparameters are found in Section 5.4 and Section 5.5 respectively.

5.3 Mean Function Estimation

We consider the estimation of the mean function μ under correlated sparse functional data. The data model is as given by Equation (1.2). As described in Chapter 3 for independently observed functional data [90] have shown the asymptotic properties for an estimator of the mean function using a local linear smoother. For spatially correlated functional data these results were extended in [48] using the same locally linear smoothing techniques. However, as mentioned in Chapter 3, in our work we will focus on the use of regularised spline smoothing as our smoothing technique for estimation. In the following we consider an estimation technique for the mean function using regularised spline smoothing under correlated functional observations.

We approximate the mean function $\mu(t)$ by the spline function $\boldsymbol{c}_{\mu}^{\intercal}\boldsymbol{B}_{d}^{\boldsymbol{\tau}}(t)$ where, as discussed in Section 3.3, $\boldsymbol{B}_{\tau}^{d}(t) = \left(B_{d,1}^{\boldsymbol{\tau}}(t), B_{d,2}^{\boldsymbol{\tau}}(t), \cdots, B_{d,K_{\mu}}^{\boldsymbol{\tau}}(t)\right)$ is the K_{μ} length collection of B-splines of order d with knot vector $\boldsymbol{\tau}$. The estimate for the coefficient vector $\boldsymbol{c}_{\mu}$ is found as:

$$\hat{\boldsymbol{c}}_{\mu} = \arg\min_{\boldsymbol{c}} \left[\sum_{i=1}^{N} \sum_{j=1}^{J_i} w_i \{ y_i(t_{ij}) - \boldsymbol{c}^{\intercal}\boldsymbol{B}_{d}^{\boldsymbol{\tau}}(t_{ij}) \}^2 + \omega \boldsymbol{c}^{\intercal}\boldsymbol{P}\boldsymbol{c} \right],$$

where w_i are fixed weights to be specified and satisfy $\sum_{i=1}^{N} J_i w_i = 1$, the q^{th} order penalty matrix $\boldsymbol{P} \in \mathbb{R}^{K_{\mu} \times K_{\mu}}$ is positive semi-definite and to be specified, and $\omega \in \mathbb{R}^{+}$ is a smoothing parameter which balances the data fit and smoothness of the fitted mean function. The form of this penalised spline regression with the various components is discussed in general in Section 3.3.

We follow [89] and introduce some more notation corresponding to the above which will be advantageous for further results. Let $\boldsymbol{B}_i = [\boldsymbol{B}_{d}^{\boldsymbol{\tau}}(t_{i1}), \boldsymbol{B}_{d}^{\boldsymbol{\tau}}(t_{i2}), \cdots, \boldsymbol{B}_{d}^{\boldsymbol{\tau}}(t_{iJ_i})]^{\intercal} \in \mathbb{R}^{J_i \times K_{\mu}}$ where we drop the notation for the order and knot vector and treat these as to be specified and fixed. Let $\boldsymbol{Y} = [\boldsymbol{y}_1^{\intercal}, \boldsymbol{y}_2^{\intercal}, \cdots, \boldsymbol{y}_N^{\intercal}]^{\intercal}$ and $\boldsymbol{B} = \left[\boldsymbol{B}_1^{\intercal}, \boldsymbol{B}_2^{\intercal}, \cdots, \boldsymbol{B}_N^{\intercal}\right]^{\intercal}$. Similarly, let $\boldsymbol{W}_i = w_i \boldsymbol{I}_{J_i \times J_i}$ and $\boldsymbol{W} = \text{BlockDiag}\left(\boldsymbol{W}_1, \boldsymbol{W}_2, \cdots, \boldsymbol{W}_N\right)$. Then $\hat{\boldsymbol{c}}_{\mu} = \boldsymbol{H}_N^{-1}\left(\boldsymbol{B}^{\intercal}\boldsymbol{W}\boldsymbol{Y}\right)$ where $\boldsymbol{H}_N = \boldsymbol{G}_N + \omega\boldsymbol{P}$ and $\boldsymbol{G}_N = \boldsymbol{B}^{\intercal}\boldsymbol{W}\boldsymbol{B}$. The mean function estimator is then given by:

$$\hat{\mu}(t) = \hat{\boldsymbol{c}}_{\mu}^{\intercal}\boldsymbol{B}_{d}^{\boldsymbol{\tau}}(t).$$

Using the above notation we now establish the asymptotic properties of the mean function estimator, $\hat{\mu}(t)$. The majority of this theorem and proof follows a similar theorem proposed by Xiao in [89]. They propose the asymptotic properties of a penalised spline smoother for independently observed functional data, [89]. We extend these results to the case where we have dependently observed functional data. To facilitate this section we introduce some notation on norms, this notation is consistent with [89]. As usual let $\|\cdot\|_2$ denote the Euclidean norm, $\|\cdot\|_F$ denote the Frobenius norm, and $\|\cdot\|_{\text{op}}$ denote the operator norm. For a matrix $\boldsymbol{A} = [a_{ij}]$, let $\|\boldsymbol{A}\|_{\max} = \max_{i,j}|a_{ij}|$ and $\|\boldsymbol{A}\|_{\infty} = \max_i \sum_j |a_{ij}|$. For a univariate continuous function g over $\mathcal{T}$ we denote the supreme norm as $\|g\|$. The L_2 norm of g is denoted by $\|g\|_{L_2}$. Finally, for every positive integer p, denote the class of functions with continuous p^{th} derivative over $\mathcal{T}$ by $\mathcal{C}^p(\mathcal{T})$.

First we make the following assumptions required for Theorem 5.1. We proceed with assumptions which are similar to the those used for the asymptotic properties of the mean function estimator under independent functional data in [89].

Assumption 5.1. *(a) The random functions $\mathcal{X}_i$ are identically distributed according to $\mathcal{X}$. (b) The random errors ε_{ij} are independent of the random functions $\mathcal{X}_i$ and are independent and identically distributed with mean zero and variance $\sigma_\varepsilon < \infty$. (c) We have a finite cross covariance function, $\|a_{\mathcal{X}}\| < \infty$.*

Assumption 5.2. *(a) The number of basis functions K_μ satisfies $K_\mu \geq N^{\delta_1}$ for some constant $\delta_1 > 0$ and $K_\mu = o(N)$. (b) The smoothing parameter ω satisfies $\omega = o(N^{-\delta_2})$ for some constant δ_2.*

We then assume we have a fixed common design, as suggested by [89]. In this case we suppose each functional data is observed at the same fixed set of time points. This observational design is made explicit using Assumption 5.3.

Assumption 5.3. *(a) $J_i = J$ for all i and $t_{ij} = \frac{j-\frac{1}{2}}{J}$. (b) $J \geq N^{\delta_3}$ for some constant $\delta_3 > 0$. (c) There exists a sufficiently small constant $c_0 > 0$ such that $K_\mu \leq c_0 J$.*

Under Assumptions 5.1- 5.3 we then state our result for the L_2 convergence of the mean function estimator from correlated functional data under fixed common design conditions.

Theorem 5.1 (Mean function: L_2 convergence under fixed common design)**.** *Suppose that Assumptions 5.1- 5.3 hold. Let $h = K_\mu^{-1}$, $h_e = \max\{h, \omega^{\frac{1}{2d}}\}$, $\tau_1 = \sum_{i=1}^N J w_i^2$, $\tau_2 = \sum_{i=1}^N J(J-1)w_i^2$, and $\tau_3 = \sum_{i=1}^N \sum_{\substack{j=1 \\ i\neq j}}^N w_i w_j J^2 a^*_{ij}$ where $a^*_{ij} = \max_{j1,j2}|a_{\mathcal{X}}\left(\boldsymbol{s}_i, t_{ij_1}, \boldsymbol{s}_j, t_{jj_2}\right)|$.*

If $\mu \in \mathcal{C}^p\left(\mathcal{T}\right)$ with $q \leq \min(p, d+1)$, then:

$$\mathbb{E}\left[\|\hat{\mu} - \mu\|_{L_2}^2\right] = O\left(h^{2(d+1)}\right) + o(h^{2p}) + O\left(\omega^2 h_e^{-2q}\right) + O\left(\tau_1 h_e^{-1} + \tau_2 + \tau_3\right).$$

As mentioned in [89] the term $O\left(h^{2(d+1)}\right) + o(h^{2p})$ is the order of the integrated and squared approximation bias of the spline function, the term $O\left(\omega^2 h_e^{-2q}\right)$ is the order of the integrated and squared shrinkage bias from the smoothness penalty, and the final term $O\left(\tau_1 h_e^{-1} + \tau_2 + \tau_3\right)$ is the integrated variability of the penalised splines. In particular, the $O(\tau_2)$ term corresponds to within subject correlation and the $O(\tau_3)$ term to the between subject correlation. The above theorem holds for general weights w_i, however a popular choice for such weights is equally weighted observations and setting $w_i = (NJ)^{-1}$. In this case $\tau_1 = (NJ)^{-1}$, $\tau_2 = N^{-1} - \tau_1$, and $\tau_3 = \frac{1}{N^2}\sum_{i=1}^N \sum_{j=1}^N a^*_{ij}$. Thus with additional constraints on the correlation $a_{\mathcal{X}}$ we can ensure that the variance of the penalised splines can remain, like in the independent case, $O(N^{-1})$. One such simple requirement would be, like in the SPACE model, [48], that:

$$\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}|a_{\mathcal{X}}(\boldsymbol{s}_i,t,\boldsymbol{s}_j,t')| \to 0 \text{ as } N \to \infty,$$

for any $t, t' \in \mathcal{T}$. This essentially requires the correlation between observations to decay sufficiently fast so that the average correlation across all observation pairs becomes negligible as the number of observations tends to infinity.

5.3.1 Proof of Theorem 5.1

We first state some technical lemmas which will aid in the proof of Theorem 5.1. These lemmas are presented and discussed with proofs in more detail in [89] and the references within. We state these here without proof for completeness.

Lemma 5.1. *Suppose that Assumption 5.2 (a) holds. If $\mu \in \mathcal{C}^P(\mathcal{T})$, then there exists a spline function $\nu_\mu(t) = \boldsymbol{\beta}^\intercal \boldsymbol{B}(t)$ for some $\boldsymbol{\beta} \in \mathbb{R}^{K_\mu}$ such that:*

$$\|\mu^{(i)} - \nu_\mu^{(i)}\| = O\left(h^{d+1-i}\right) + o(h^{p-i}).$$

As in [89] we use notation for describing the design points. Let $Q_N(t) = \sum_{i=1}^{N} w_i \sum_{j=1}^{J_i} 1_{t_{ij}<t}$ where $1.$ is an indicator function. The function $Q_N(t)$ is an empirical cumulative distribution function under the fixed common design. Let $Q(t) = t$ be the cumulative distribution function under fixed common design with density $\rho(t) = 1$. It will be shown that such an empirical cumulative distribution function Q_N converges to Q in the following lemmas, we refer to [89] for proofs of these lemmas.

Lemma 5.2. *Suppose that Assumption 5.2 (a) holds and $p \geq 1$. Let $\nu_m u$ be the spline function described in Lemma 5.1 and $F(\cdot)$ be a cumulative distribution function in $\mathcal{T}$. Then for $i = 0$ or $i = 1$:*

$$\max_k \left|\int B_k(t)\{\mu^{(i)} - \nu_\mu^{(i)}\}dF(t)\right| = o\left(h^{p+1-i}\right) + o\left(h^{p-i}\|F - Q\|\right).$$

Lemma 5.2 is given as Lemma A.2 in [89] which shows the same lemma in a more general sense. The following lemmas apply under the fixed common design setting so we assume Assumption 5.3 holds for each. Let $\boldsymbol{G} = \int \boldsymbol{B}(s)\,\boldsymbol{B}^\intercal(s)\,ds$.

Lemma 5.3. *Suppose that Assumption 5.2 and Assumption 5.3 hold. Then:*

$$\boldsymbol{G}_N \simeq \boldsymbol{G} \simeq h\boldsymbol{I}.$$

Lemma 5.4. *Suppose that Assumption 5.2 and Assumption 5.3 hold. Let α_{ij} represent the $(i,j)^{th}$ element of $\boldsymbol{G}_N^{-1}$. Then there exists constants $c \geq 0$ and $0 < \gamma < 1$ such that, for large N:*

$$|\alpha_{ij}| \leq ch^{-1}\gamma^{|i-j|}.$$

In addition,

$$\|\boldsymbol{G}_N^{-1}\|_\infty = O\left(h^{-1}\right).$$

Lemma 5.5. *Suppose that Assumption 5.2 and Assumption 5.3 hold. Then, the following hold:*

$$\begin{aligned}
\|\boldsymbol{G}_N - \boldsymbol{G}\|_{max} &= O\left(\|Q_N - Q\|_{max}\right), \\
\|\boldsymbol{G}_N^{-1} - \boldsymbol{G}^{-1}\|_{max} &= O\left(h^{-2}\|Q_N - Q\|_{max}\right), \\
\|\boldsymbol{G}_N^{-1} - \boldsymbol{G}^{-1}\|_{\infty} &= O\left(h^{-2}\|Q_N - Q\|_{max}\right).
\end{aligned}$$

Lemmas 5.3, 5.4, 5.5 cause the following to hold under fixed common design, as shown in [89]:

Lemma 5.6. *Suppose that Assumption 5.2 and Assumption 5.3 hold. Define $\boldsymbol{\gamma} = \boldsymbol{G}_N^{-1}\left(\boldsymbol{B}^\intercal \boldsymbol{W}\boldsymbol{\mu}\right)$ where $\boldsymbol{\mu} = [\boldsymbol{\mu}_1^\intercal, \boldsymbol{\mu}_2^\intercal, \cdots, \boldsymbol{\mu}_N^\intercal]^\intercal$ and $\boldsymbol{\mu}_i = (\mu(t_{i1}), \mu(t_{i2}), \cdots, \mu(t_{iJ_i}))^\intercal$. Then the following equalities hold:*

$$\begin{aligned}
\|\boldsymbol{H}_N^{-1}\|_{max} &= O\left(h_e^{-1}\right), \\
\|\boldsymbol{H}_N^{-1}\|_{\infty} &= O\left(h^{-1}\right), \\
\|\boldsymbol{H}_N^{-1}\boldsymbol{P}\boldsymbol{\gamma}\|_{max} &= O\left(h_e^{-q}\right).
\end{aligned}$$

Using the above Lemma 5.1- 5.6 we can now prove the asymptotic properties described in Theorem 5.1 under fixed design conditions.

Proof of Theorem 5.1. The proof of the asymptotic properties of the mean estimator, using a penalised spline estimator, under dependently observed functional data follows closely the proof of the mean estimator, using a penalised spline, under independent observed functional data given in [89]. As a matter of completeness we will present the full proof, although much is the same as presented in [89] due to the similar nature of the theorems. Deviations from the proof in [89] are due to the fact we can no longer assume independent observations as is done in [89]. These will be clearly signposted.

First, without loss of generality, let $\mathcal{T} = [0,1]$. Then:

$$\mathbb{E}\left(\|\hat{\mu} - \mu\|_{L_2}^2\right) \leq \|\mathbb{E}\left((\hat{\mu} - \mu)^2\right)\| \leq \|\mathbb{E}\left(\hat{\mu}\right) - \mu\| + \|\text{var}\left(\hat{\mu}\right)\|, \tag{5.14}$$

where the first term in the right hand side of Equation (5.14) corresponds to a bias term and the second being the variance term. We bound the bias and variance terms separately. We start with the bias term.

The bound on the bias term is derived exactly as in [89]. First, let ν_μ be the spline function as defined in Lemma 5.1 such that:

$$\|\mu - \nu_\mu\| = O\left(h^{d+1}\right) + o(h^p). \tag{5.15}$$

Define $\boldsymbol{\nu}_{i,\mu} = (\nu_\mu(t_{i1}), \nu_\mu(t_{i2}), \cdots, \nu_\mu(t_{iJ_i}))^\intercal$ and $\boldsymbol{\nu}_\mu = \left[\boldsymbol{\nu}_{1,\mu}^\intercal, \boldsymbol{\nu}_{2,\mu}^\intercal, \cdots, \boldsymbol{\nu}_{N,\mu}^\intercal\right]^\intercal$. Then we can write:

$$\begin{aligned}\mathbb{E}\left(\hat{\mu}\right) &= \boldsymbol{B}^\intercal(t)\boldsymbol{H}_N^{-1}\left(\boldsymbol{B}^\intercal\boldsymbol{W}\boldsymbol{\mu}\right)\\ &= \boldsymbol{B}^\intercal(t)\boldsymbol{G}_N^{-1}\left(\boldsymbol{B}^\intercal\boldsymbol{W}\boldsymbol{\mu}\right) - \boldsymbol{B}^\intercal(t)\boldsymbol{H}_N^{-1}\left(\omega\boldsymbol{P}\right)\boldsymbol{G}_N^{-1}\left(\boldsymbol{B}^\intercal\boldsymbol{W}\boldsymbol{\mu}\right)\end{aligned}$$

Splitting the first term in Equation (5.17) using our spline function $\boldsymbol{\nu}_\mu$ we can write:

$$\begin{aligned}\mathbb{E}\left(\hat{\mu}\right) = \boldsymbol{B}^\intercal(t)\boldsymbol{G}_N^{-1}\left(\boldsymbol{B}^\intercal\boldsymbol{W}\boldsymbol{\nu}_\mu\right) + \boldsymbol{B}^\intercal(t)\boldsymbol{G}_N^{-1}\left(\boldsymbol{B}^\intercal\boldsymbol{W}\left(\boldsymbol{\mu} - \boldsymbol{\nu}_\mu\right)\right)\\ - \boldsymbol{B}^\intercal(t)\boldsymbol{H}_N^{-1}\left(\omega\boldsymbol{P}\right)\boldsymbol{G}_N^{-1}\left(\boldsymbol{B}^\intercal\boldsymbol{W}\boldsymbol{\mu}\right)\end{aligned}. \tag{5.18}$$

As ν_μ is a spline function we have $\nu_\mu(t) = \boldsymbol{B}^\intercal(t)\boldsymbol{\beta}$ for some $\boldsymbol{\beta} \in \mathbb{R}^{K_\mu}$. Thus $\boldsymbol{\nu}_\mu = \boldsymbol{B}\boldsymbol{\beta}$ and:

$$\boldsymbol{B}^\intercal(t)\boldsymbol{G}_N^{-1}\left(\boldsymbol{B}^\intercal\boldsymbol{W}\boldsymbol{\nu}_\mu\right) = \boldsymbol{B}^\intercal(t)\boldsymbol{\beta} = \nu_\mu(t). \tag{5.19}$$

Now define $\boldsymbol{\alpha} = \boldsymbol{B}^\intercal\boldsymbol{W}\left(\boldsymbol{\mu} - \boldsymbol{\nu}_\mu\right)$ and let $\boldsymbol{\gamma}$ be as defined in Lemma 5.6. Then using Equation (5.19) and Equation (5.18) we have:

$$\mathbb{E}\left(\hat{\mu}(t)\right) - \mu(t) = \left(\nu_\mu - \mu\right)(t) + \boldsymbol{B}^\intercal(t)\boldsymbol{G}_N^{-1}\boldsymbol{\alpha} - \boldsymbol{B}^\intercal(t)\boldsymbol{H}_N^{-1}\left(\omega\boldsymbol{P}\right)\boldsymbol{\gamma}. \tag{5.20}$$

Bounding this then follows as:

$$\|\mathbb{E}\left(\hat{\mu}(t)\right) - \mu(t)\| \leq \|\nu_\mu - \mu\| + \|\boldsymbol{G}_N^{-1}\boldsymbol{\alpha}\|_{\max} + \|\boldsymbol{H}_N^{-1}\left(\omega\boldsymbol{P}\right)\boldsymbol{\gamma}\|_{\max}, \tag{5.21}$$

where we use the non-negativity and unity of the B-spline functions present in Equation (5.20) through the terms of $\boldsymbol{B}^\intercal(t)$.

We formalise a bound on each of the terms in Equation (5.21) separately. Firstly by Lemma 5.1 we have a bound for $\|\mu - \nu_\mu\|$ by setting i to be zero in the lemma. This bound is given in Equation (5.15). Next we consider the bound for $\|\boldsymbol{G}_N^{-1}\boldsymbol{\alpha}\|_{\max}$. As proposed in [89] we bound $\boldsymbol{\alpha}$ and achieve a bound on the whole term. Let α_k be the k^{th} element of $\boldsymbol{\alpha}$. Then:

$$a_k = \sum_{i=1}^{N} w_i \sum_{j=1}^{J_i} B_{d,k}^{\tau}(t_{ij})\left(\mu(t_{ij}) - \nu_\mu(t_{ij})\right) = \int B_{d,k}^{\tau}(s)\left(\mu(t_{ij}) - \nu_\mu(t_{ij})\right) dQ_N(s),$$

where $Q_N(s) = \sum_{i=1}^{N} w_i \sum_{j=1}^{J_i} 1_{t_{ij}<s}$ is an empirical cumulative distribution function. Under Lemma 5.2 and replacing Q_N for F we have that:

$$\|\boldsymbol{a}\|_{\max} = o(h^{p+1}) + o(h^p \|Q_N - Q\|).$$

Noting that $\|Q_N - Q\| = O(J^{-1})$ under Assumption 5.3 we have that $\|\boldsymbol{\alpha}\|_{\max} = o(h^{p+1})$. Now, since $|\boldsymbol{G}_N^{-1}\boldsymbol{\alpha}\|_{\max} \leq \|\boldsymbol{G}_N^{-1}\|_\infty \|\boldsymbol{\alpha}\|_{\max}$ and by Lemma 5.4 we have that:

$$\|\boldsymbol{G}_N^{-1}\boldsymbol{a}\|_{\max} = o(h^p). \tag{5.22}$$

The final term in Equation (5.21) is bounded by Lemma 5.6. This is:

$$\|\boldsymbol{H}_N^{-1}(\omega \boldsymbol{P})\,\boldsymbol{\gamma}\|_{\max} = O(\omega h_e^{-(d+1)}). \tag{5.23}$$

Combining the bounds given in Equations (5.15), (5.22), (5.23) we obtain:

$$\|\mathbb{E}\left(\hat{\mu}(t)\right) - \mu(t)\|^2 = O(h^{2(d+1)}) + o(h^{2p}) + O(\omega^2 h_e^{-2q}). \tag{5.24}$$

We now move to consider the variance term of Equation (5.14). It is at this point where we deviate from the proof of the independent version of the corresponding theorem given in [89]. First we can decompose the variance term as:

$$\operatorname{var}\left(\hat{\mu}(t)\right) = \boldsymbol{B}^\intercal(t)\boldsymbol{H}_N^{-1}\boldsymbol{B}^\intercal\boldsymbol{W}\operatorname{var}\left(\boldsymbol{Y}\right)\boldsymbol{W}\boldsymbol{B}\boldsymbol{H}_N^{-1}\boldsymbol{B}(t). \tag{5.25}$$

Considering the innermost expression of the above and define:

$$\tilde{\boldsymbol{\Gamma}} = \boldsymbol{B}^\intercal\boldsymbol{W}\operatorname{var}\left(\boldsymbol{Y}\right)\boldsymbol{W}\boldsymbol{B} = \sum_{i=1}^{N}\sum_{j=1}^{N} \boldsymbol{B}_i^\intercal \boldsymbol{W}_i \operatorname{cov}\left(\boldsymbol{y}_i, \boldsymbol{y}_j\right)\boldsymbol{W}_j\boldsymbol{B}_j.$$

We note that $\operatorname{cov}\left(\boldsymbol{y}_i, \boldsymbol{y}_j\right) \in \mathbb{R}^{J_i \times J_j}$ is the matrix with $(l,m)^{\text{th}}$ element given by:

$$a_{\mathcal{X}}(\boldsymbol{s}_i, t_{il}, \boldsymbol{s}_j, t_{jm}) + \sigma^2 1_{i=j,\ l=m}.$$

Now let $\tilde{\gamma}_{lm}$ denote the $(l,m)^{\text{th}}$ element of $\tilde{\Gamma}$.

$$\tilde{\gamma}_{lm} = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sum_{j_1=1}^{J_i}\sum_{j_2=1}^{J_j} B_{d,l}^{\tau}(t_{ij_1}) B_{d,m}^{\tau}(t_{jj_2}) \left(a_{\mathcal{X}}(\boldsymbol{s}_i, t_{il}, \boldsymbol{s}_j, t_{jm}) + \sigma^2 1_{i=j,\ l=m}\right).$$

Now let $\gamma_{1lm}, \gamma_{2lm}$ be defined as:

$$\begin{aligned}
\gamma_{1lm} &= \sum_{i=1}^{N} w_i^2 \sum_{j=1}^{J_i} B_{d,l}^{\tau}(t_{ij}) B_{d,m}^{\tau}(t_{ij}), \\
\gamma_{2lm} &= \sum_{i=1}^{N} w_i^2 \sum_{j_1=1}^{J_i} \sum_{j_2=1}^{J_i} B_{d,l}^{\tau}(t_{ij_1}) B_{d,m}^{\tau}(t_{ij_2}), \\
\gamma_{3lm} &= \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} w_i w_j \sum_{j_1=1}^{J_i} \sum_{\substack{j_2=1 \\ j_1 \neq j_2}}^{J_j} B_{d,l}^{\tau}(t_{ij_1}) B_{d,m}^{\tau}(t_{jj_2}) |a_{\mathcal{X}}(\boldsymbol{s}_i, t_{ij_1}, \boldsymbol{s}_j, t_{jj_2})|,
\end{aligned}$$

Then $|\tilde{\gamma}_{lm}| \leq \sigma^2 \gamma_{1lm} + \|a_{\mathcal{X}}\| \gamma_{2lm} + \gamma_{3lm}$. Define $\boldsymbol{\Gamma}_i = [\gamma_{ilm}]$ for $i = 1, 2, 3$. Let $\boldsymbol{\Gamma} = [\gamma_{lm}] = \sigma^2 \boldsymbol{\Gamma}_1 + \|a_{\mathcal{X}}\| \boldsymbol{\Gamma}_2 + \boldsymbol{\Gamma}_3$. Now, similarly to [89], by the linearity of terms in Equation (5.25) we have:

$$\begin{aligned}
\boldsymbol{B}^{\intercal}(t) \boldsymbol{H}_N^{-1} \tilde{\boldsymbol{\Gamma}} \boldsymbol{H}_N^{-1} \boldsymbol{B}(t) &\leq \boldsymbol{B}^{\intercal}(t) \left(\boldsymbol{H}_N^{-1}\right)_+ \left(\tilde{\boldsymbol{\Gamma}}\right)_+ \left(\boldsymbol{H}_N^{-1}\right)_+ \boldsymbol{B}(t) \\
&\leq \boldsymbol{B}^{\intercal}(t) \left(\boldsymbol{H}_N^{-1}\right)_+ \boldsymbol{\Gamma} \left(\boldsymbol{H}_N^{-1}\right)_+ \boldsymbol{B}(t)
\end{aligned},$$

where $(\cdot)_+ = [|\cdot_{lm}|]$ is the matrix formed of absolute values of elements. Again by the unity and non-negativity of B-splines we have:

$$\|\text{var}\,(\hat{\mu})\| \leq \|\left(\boldsymbol{H}_N^{-1}\right)_+ \boldsymbol{\Gamma} \left(\boldsymbol{H}_N^{-1}\right)_+ \|_{\max}. \tag{5.29}$$

$$\begin{aligned}
\|\left(\boldsymbol{H}_N^{-1}\right)_+ \boldsymbol{\Gamma} \left(\boldsymbol{H}_N^{-1}\right)_+ \|_{\max} \leq \sigma^2 \|\boldsymbol{H}_N^{-1}\|_{\infty} \|\boldsymbol{H}_N^{-1}\|_{\max} \|\boldsymbol{\Gamma}_1\|_{\text{op}} + \|a_{\mathcal{X}}\| \|\boldsymbol{H}_N^{-1}\|_{\infty}^2 \|\boldsymbol{\Gamma}_2\|_{\max} \\
+ \|\boldsymbol{H}_N^{-1}\|_{\infty}^2 \|\boldsymbol{\Gamma}_3\|_{\max}
\end{aligned} \tag{5.30}$$

Under fixed common design of Assumption 5.3, as in [89], we have that $B_k(t_{ij}) \neq 0$ for $O(Jh)$ j's with the big-O notation being uniform in k. Hence, $\|\boldsymbol{\Gamma}_1\|_{\text{op}} = O(\tau_1 h)$. Similarly $\|\boldsymbol{\Gamma}_2\|_{\infty} = O((\tau_1 + \tau_2) h^2)$ and $\|\boldsymbol{\Gamma}_3\|_{\infty} = O(\tau_3 h^2)$. By Lemma 5.5 we have bounds for both the maximum and infinity norm of $\boldsymbol{H}_N^{-1}$. That is $\|\boldsymbol{H}_N^{-1}\|_{\max} = O(h_e^{-1})$ and $\|\boldsymbol{H}_N^{-1}\|_{\infty} = O(h^{-1})$. Combining these bounds with Equations (5.30), (5.29) we obtain:

$$\|\text{var}\,(\hat{\mu})\| = O(\tau_1 h_e^{-1}) + O(\tau_1 + \tau_2) + O(\tau_3) = O(\tau_1 h_e^{-1} + \tau_2 + \tau_3). \tag{5.31}$$

Combining Equations (5.24), (5.31) we are done. □

5.4 Covariance Function Estimation

We consider the estimation of the covariance function of $\mathcal{X}$ given our observed functional data is not observed independently. Typically, the methodology for estimating the covariance function G is a two step process. Firstly, an empirical covariance is constructed from the observed data and an estimate of the mean function. Lastly, a bivariate smoother is used to smooth the empirical covariance to obtain the estimated covariance surface. As mentioned in Chapter 3 the PACE methodology given by [90] details the method for estimating the covariance function using a local linear smoother. Under assumptions of independence they then go on to show such an estimator is consistent. Similarly, [48] considers the use of a local linear smoother to smooth the empirical covariance matrix when observations are spatially dependent. As mentioned before, in this work we will focus on the penalised spline smoother. The work of [89] considers this smoother for estimating the covariance function when observations are independent. In the following we consider the same penalised spline smoother as the estimator for the covariance function under functional data which are dependently observed.

Let $\hat{\mu}$ be an estimate of the mean function μ such as that described in Section 5.3. Let the residual for the i^{th} observation at time t_{ij} be denoted $\tilde{e}_{ij} = y_i(t_{ij}) - \hat{\mu}(t_{ij})$. Denote the auxiliary variables $\tilde{g}_{ij_1j_2} = \tilde{e}_{ij_1}\tilde{e}_{ij_2}$. The collection $\{\tilde{g}_{ij_1j_2}; 1 \leq j_1 \neq j_2 \leq J_i, i = 1, 2, \cdots, N\}$ is a collection of empirical estimates of the covariance function.

As G is a function over $\mathcal{T}\times\mathcal{T}$ we need a penalised spline smoothing over two dimensions. This is discussed in Section 3.3. In particular the function G is modelled as a tensor-product spline:

$$\mathcal{H}\left(s,t\right) = \bar{\boldsymbol{B}}^{\intercal}\left(s,t\right)\text{Vec}\left(\boldsymbol{C}_G\right),$$

where $\bar{\boldsymbol{B}}\left(s,t\right) = \boldsymbol{B}_{d_2}^{\boldsymbol{\tau}_2}(t) \otimes \boldsymbol{B}_{d_1}^{\boldsymbol{\tau}_1}(s)$ and $\text{Vec}\left(\cdot\right)$ is an operator which stacks the columns of a matrix into a vector. The notation $\boldsymbol{B}_{d_i}^{\boldsymbol{\tau}_i}(t) \in \mathbb{R}^{K_{iG}}$ is as previously discussed and K_{iG} is the basis size of the i^{th} dimension of the covariance surface. As in Section 3.3 we use the $\bar{\cdot}$ notation to make explicit that this basis is over multiple dimensions. The estimate of the coefficient matrix $\boldsymbol{C}_G$ is given by:

$$\hat{\boldsymbol{C}}_G = \underset{\boldsymbol{C}_G}{\arg\min}\left[\sum_{i=1}^{N} v_i \sum_{j_1=1}^{J_i}\sum_{\substack{j_2=1\\ j_1\neq j_2}}^{J_i}\left(\tilde{g}_{ij_1j_2} - \mathcal{H}(t_{ij_1}, t_{ij_2})\right)^2 + \text{Vec}\left(\boldsymbol{C}_G\right)^{\intercal}\bar{\boldsymbol{P}}_G\text{Vec}\left(\boldsymbol{C}_G\right)\right],$$

where $\bar{\boldsymbol{P}}_G$ is a penalty matrix such as the tensor penalty matrix specified in Equation (3.16) and $v_i > 0$ are weights to be specified such that $\sum_{i=1}^{N} v_i J_i(J_i-1) = 1$. Following Xiao in [89], we can analytically find the coefficient estimator as follows. Let $\boldsymbol{G}_{G,N} = \sum_{i=1}^{N}\boldsymbol{A}_i^{\intercal}\boldsymbol{V}_i\boldsymbol{A}_i$ and $\boldsymbol{H}_{G,N} = \boldsymbol{G}_{G,N} + \bar{\boldsymbol{P}}_N$. Here $\boldsymbol{A}_i$ is the same as in [89]. That is it is the sub-matrix $\bar{\boldsymbol{B}}_i \otimes \bar{\boldsymbol{B}}_i$ that excludes the rows corresponding to the same t_{ij}. Finally we let $\boldsymbol{V}_i = v_i\boldsymbol{I}_{J_i(J_i-1)}$. Then:

$$\hat{\boldsymbol{C}}_G = \boldsymbol{H}_{N,G}^{-1}\left(\sum_{i=1}^{N}\boldsymbol{A}_i^{\intercal}\boldsymbol{V}_i\tilde{\boldsymbol{g}}_i^*\right),$$

where $\tilde{\boldsymbol{g}}_i^* = \text{Vec}^*\left([\tilde{g}_{ij_1j_2}]\right)$ and Vec^* is the operator same as Vec except that it excludes diagonal element of the square matrix. We then simply have:

$$\hat{G}(s,t) = \bar{\boldsymbol{B}}^{\intercal}\left(s,t\right)\text{Vec}\left(\hat{\boldsymbol{C}}_G\right).$$

The above estimator of the covariance function G is simple to describe analytically, however its consistency as an estimator must be verified. We do so by providing a theorem to such an effect under certain assumptions. These assumptions build on those given for the mean function estimator in Section 5.3. We state these further assumptions below. Again these are the same assumptions for the covariance estimator under independent observed data discussed in [89].

Assumption 5.4. *We have the following holding:*

$$\begin{aligned} \sup \mathbb{E}\left(\mathcal{X}_i^4(t)\right) &< \infty, \\ \mathbb{E}\left(\varepsilon_{ij}^4\right) &< \infty, \end{aligned}$$

Assumption 5.5. *(a) The number of basis functions K_G satisfies $K_G \geq N^{\delta_4}$ for some constant $\delta_4 > 0$ and $K_G = o(N)$. (b) The smoothing parameter ω_G satisfies $\omega_G = o(N^{-\delta_5})$ for some constant δ_5.*

As the covariance estimator makes use of a mean function estimator, $\hat{\mu}$, we introduce an assumption on the convergence of this estimator.

Assumption 5.6. *The mean function estimator $\hat{\mu}$ satisfies:*

$$\sup_{t \in \mathcal{T}} \mathbb{E}\left((\hat{\mu}(t) - \mu(t))^4\right) = O(U_1),$$

as $N \to \infty$, where $U_1 = o(1)$ is a non-random number.

With the assumptions above and in Section 5.3 we now state the theorem for L_2 convergence of the covariance estimator under dependently observed functional data.

Theorem 5.2 (Covariance function: L_2 convergence under fixed common design)**.** *Suppose that Assumptions 5.1, 5.3, 5.4, 5.5, 5.6 hold. Let $h_G = K_G^{-1}$ and $h_{G,e} = \max\{h_G, \omega_G^{\frac{1}{2d}}\}$. Define $\tilde{\tau}_1 = J^4 \sum_{i=1}^N v_i^2$ and $\tilde{\tau}_2 = J^4 h_G^4 \sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N v_i v_j a_{ij}^*$ where $a_{ij}^* = \sup_{t,t' \in \mathcal{T}} |a_{\mathcal{X}}(\boldsymbol{s}_i, t, \boldsymbol{s}_j, t')|^2$. If $G \in \mathcal{C}^p\left(\mathcal{T}^2\right)$ with $q \leq \min(p, d+1)$ then:*

$$\mathbb{E}\left(\|G - \hat{G}\|_{L_2}^2\right) = O(U_1^2) + O(h_G^{2(d+1)}) + o(h_G^{2p}) + O(\omega_G^2 h_{G,e}^{-2q}) + O(\tilde{\tau}_1 + \tilde{\tau}_2).$$

We note the similarities between this theorem on the bound of the covariance estimator and that of the mean function estimator in Theorem 5.1. In particular, as in Theorem 5.1, each term in the above can be thought of separately. The first term $O(U_1)$ corresponds to any bias introduced via the mean function estimator. Assumption 5.6 essentially bounds this to be negligible. That is, we assume we have a consistent estimator for the mean function such as that given in Theorem 5.1. The final term corresponds to convergence introduced due to the variance between observations. It is notable that like the mean function estimator if we further assume that $\tilde{\tau}_2 = O(1)$ and we use equally weighted observations we have the convergence of the estimator being $O(N^{-1})$. Such an assumption can typically be enforced by specifying that the kernel function behaves in the sense that the squared kernel decays to zero sufficiently fast as the number of functional observations tends to infinity. This is similar to the additional assumption of spatial dependence that is used by Liu et al. in their SPACE model, [48]. If we impose such an assumption we obtain the same convergence as the independent results given in [89].

5.4.1 Proof of Theorem 5.2

To aid in the proof of Theorem 5.2 we state some technical lemmas which aid in this. These are given in more detail in [89] and the references within. We state these lemmas without proof for completeness and refer the reader to [89] for detailed discussions.

The first lemma is similar to Lemma 5.1 for the mean function estimator. We essentially need a result showing that the covariance surface G can be approximated by a spline function. That is:

Lemma 5.7. *Suppose that Assumption 5.6 holds. If $G \in \mathcal{C}^p(\mathcal{T}^2)$, then there exists a spline function $\nu_G(s,t) = \bar{\boldsymbol{B}}^\intercal(s,t)\,\boldsymbol{\beta}_G$ such that:*

$$\|G^{(i,j)} - \nu_G^{(i,j)}\| = O(h_G^{d+1-i} + h_G^{d+1-j}) + o(h_G^{p-i} + h_G^{p-j}),$$

for $i + j \leq \min(p,2)$.

The second lemma similarly mirrors Lemma 5.2 for the mean function estimator.

Lemma 5.8. *Suppose that Assumption 5.6 holds. Let $F(\cdot,\cdot)$ be any cumulative distribution function in $\mathcal{T}^2$ and let $Z(s,t) = F(s,t) - Q(s)Q(t)$. Then:*

$$\max_{l,m}\left|\int\int \bar{B}_l(s)\bar{B}m(t)\,(G(s,t) - \nu_G(s,t))\,dF(s,t)\right| = o(h_G^{p+2}) + o(h_G^p\|Z\|).$$

The next three lemmas again mirror the lemmas for the mean function. They are present to present bounds for a few key matrices, namely $\boldsymbol{G}_N = \boldsymbol{B}^\intercal \boldsymbol{W}\boldsymbol{B}$ and $\boldsymbol{G}_{G,N} = \sum_{i=1}^N \boldsymbol{A}_i^\intercal \boldsymbol{V}_i \boldsymbol{A}_i$.

Lemma 5.9. *Suppose that both Assumption 5.3 and Assumption 5.6 hold. Let $\bar{\boldsymbol{G}} = \int \bar{\boldsymbol{B}}(s)\bar{\boldsymbol{B}}^\intercal(s)ds$ and $\boldsymbol{G}_G = \bar{\boldsymbol{G}} \otimes \bar{\boldsymbol{G}}$. Then:*

$$\begin{aligned}\boldsymbol{G}_N &\simeq \bar{\boldsymbol{G}} \simeq h_G\boldsymbol{I},\\ \boldsymbol{G}_{G,N} &\simeq \boldsymbol{G}_G \simeq h_G^2\boldsymbol{I}.\end{aligned}$$

The proof of this lemma is not given in [89] however it is stated that the proof is similar to proofs in [88]. We refer the reader to [88] for proof of this lemma.

Lemma 5.10. *Suppose that both Assumption 5.3 and Assumption 5.6 hold. Denote the $(i,j)^{th}$ element of $\boldsymbol{G}_{G,N}^{-1}$ by $\alpha_{G,ij}$. Then there exists constants c_G and $0 < \gamma_G < 1$ such that, for large N:*

$$|\alpha_{G,ij}| \leq c_G h_G^{-2} \gamma_G^{|i-j|}.$$

In addition,

$$\|\boldsymbol{G}_{G,N}^{-1}\|_\infty = O(h_G^{-2}).$$

Lemma 5.11. *Suppose that both Assumption 5.3 and Assumption 5.6 hold. Let $R_N(s,t) = \sum_{i=1}^N v_i \sum_{j_1=1}^J \sum_{\substack{j_2=1 \\ j_1 \neq j_2}}^J 1_{t_{ij_1}<s} 1_{t_{ij_2}<t}$ be an empirical cumulative distribution function. Let $\tilde{Z}(s,t) = R_N(s,t) - Q(s)Q(t)$. Then:*

$$\begin{aligned}
\|\boldsymbol{G}_{G,N} - \boldsymbol{G}_G\|_{max} &= O(\|\tilde{Z}\|), \\
\|\boldsymbol{G}_{G,N}^{-1} - \boldsymbol{G}_G^{-1}\|_{max} &= O(h_G^{-4}\|\tilde{Z}\|), \\
\|\boldsymbol{G}_{G,N}^{-1} - \boldsymbol{G}_G^{-1}\|_\infty &= O(h_G^{-4}\|\tilde{Z}\|).
\end{aligned}$$

Similarly as given in [89] using Lemmas 5.9, 5.10, 5.11 we have the following lemma.

Lemma 5.12. *Suppose that both Assumption 5.3 and Assumption 5.6 hold. Then:*

$$\begin{aligned}
\|\boldsymbol{H}_{G,N}^{-1}\|_{max} &= O(h_{G,e}^{-2}), \\
\|\boldsymbol{H}_{G,N}^{-1}\|_{max} &= O(h_G^{-2}).
\end{aligned}$$

Finally we detail two more technical lemmas, which are given in [89], that propose bounds on constructs of $\boldsymbol{G}_N$ and the penalty matrix $\bar{\boldsymbol{P}}$. The proof of Lemmas 5.13, 5.14 are given in [88] and [89] respectively.

Lemma 5.13. *Suppose that both Assumption 5.3 and Assumption 5.6 hold. Let $\bar{\boldsymbol{\Lambda}}_N = \bar{\boldsymbol{G}}_N + \omega_G \bar{\boldsymbol{P}}$ and $\boldsymbol{\Delta}_q$ be the q^{th} order difference operator, then:*

$$\begin{aligned}
\|\bar{\boldsymbol{\Lambda}}_N^{-1}\|_{max} &= O(h_{G,e}^{-1}), \\
\|\bar{\boldsymbol{\Lambda}}_N^{-1}\|_\infty &= O(h_G^{-1}), \\
\|\bar{\boldsymbol{\Lambda}}_N^{-1}\boldsymbol{\Delta}_q^\intercal\|_\infty &= O(h_{G,e}^{-q}).
\end{aligned}$$

Lemma 5.14. *Suppose that both Assumption 5.3 and Assumption 5.6 hold. Let $\bar{\boldsymbol{\Lambda}}_n = \bar{\boldsymbol{G}}_N + \omega_G \bar{\boldsymbol{P}}$ and let $\bar{\boldsymbol{\Lambda}} = \bar{\boldsymbol{G}} + \omega_G \bar{\boldsymbol{P}}$. Define $\bar{\boldsymbol{H}}_{G,N} = \bar{\boldsymbol{\Lambda}}_n \otimes \bar{\boldsymbol{\Lambda}}_n$ and similarly $\bar{\boldsymbol{H}}_G = \bar{\boldsymbol{\Lambda}} \otimes \bar{\boldsymbol{\Lambda}}$. Then;*

$$\bar{\boldsymbol{H}}_{G,N}^{-1} = (\boldsymbol{I} + \boldsymbol{D})\,\bar{\boldsymbol{H}}_G^{-1},$$

where $\boldsymbol{D}$ satisfies $\|\boldsymbol{D}\|_\infty = o(1)$.

With the above technical lemmas for the covariance estimator we can now prove Theorem 5.2. Similar to the proof of Theorem 5.1 we closely follow the proof for the related theorem in [89] which focused on independently observed functional data. We present the full proof below but note that much is coming fully from [89] as no adjustment is needed due to the observed dependence. We make it clear when this proof deviates from that in [89].

Proof of Theorem 5.2. To prove Theorem 5.2, like [89], we set up some notation. Denote by $b(t)$ the difference between the estimated mean function and the true mean function, that is $b(t) = \hat{\mu}(t) - \mu(t)$. Let $b_{ij} = b(t_{ij})$ and similarly $e_{ij} = y_i(t_{ij}) - \mu(t_{ij})$. Then:

$$\operatorname{Cov}\left(e_{ij_1}, e_{ij_2}\right) = G(t_{ij_1}, t_{ij_2}) + 1_{j_1=j_2}.\sigma_\varepsilon^2$$

Further, let $\tilde{e}_{ij} = y_i(t_{ij}) - \hat{\mu}(t_{ij}) = e_{ij} - b_{ij}$. We denote by $\tilde{g}_{ij_1j_2}$ the product:

$$\tilde{g}_{ij_1j_2} = \tilde{e}_{ij_1}\tilde{e}_{ij_2} = e_{ij_1}e_{ij_2} - e_{ij_1}b_{ij_2} - b_{ij_1}e_{ij_2} + b_{ij_1}b_{ij_2}.$$

To simplify the above we let $\breve{g}_{ij_1j_2} = e_{ij_1}e_{ij_2}$ and $\tilde{d}_{ij_1j_2} = -e_{ij_1}b_{ij_2} - b_{ij_1}e_{ij_2} + b_{ij_1}b_{ij_2}$ such that:

$$\tilde{g}_{ij_1j_2} = \breve{g}_{ij_1j_2} + \tilde{d}_{ij_1j_2}.$$

Then $\mathbb{E}\left(\tilde{g}_{ij_1j_2}\right) = G(t_{ij_1}, t_{ij_2}) + d_{ij_1j_2}$ where $d_{ij_1j_2} = \mathbb{E}\left(\tilde{d}_{ij_1j_2}\right)$. As in [89] we have:

$$\max_{i,j_2,j_2} |d_{ij_1j_2}| = O(U_1), \tag{5.44}$$

where U_1 is defined in Assumption 5.6. Let $\boldsymbol{g}_i^* = \operatorname{Vec}^*\left(\{g_{ij_1j_2}; 1 \leq j_1, j_2 \leq J_i\}\right)$ and define $\boldsymbol{g}^* = (\boldsymbol{g}_1^{*\intercal}, \boldsymbol{g}_2^{*\intercal}, \cdots, \boldsymbol{g}_N^{*\intercal})^\intercal$. Similarly we can define $\tilde{\boldsymbol{g}}^*$, $\breve{\boldsymbol{g}}^*$, $\tilde{\boldsymbol{d}}^*$, and $\boldsymbol{d}^*$. Then:

$$\begin{aligned}
\hat{G}(s,t) &= \bar{\boldsymbol{B}}^\intercal(s,t)\,\boldsymbol{H}_{G,N}^{-1}\left(\boldsymbol{A}^\intercal\boldsymbol{V}\tilde{\boldsymbol{g}}^*\right),\\
\breve{G}(s,t) &= \bar{\boldsymbol{B}}^\intercal(s,t)\,\boldsymbol{H}_{G,N}^{-1}\left(\boldsymbol{A}^\intercal\boldsymbol{V}\breve{\boldsymbol{g}}^*\right),\\
f(s,t) &= \bar{\boldsymbol{B}}^\intercal(s,t)\,\boldsymbol{H}_{G,N}^{-1}\left(\boldsymbol{A}^\intercal\boldsymbol{V}\tilde{\boldsymbol{d}}^*\right),\\
\hat{G}(s,t) &= \breve{G}(s,t) + f(s,t),
\end{aligned}$$

where $\boldsymbol{A} = \left[\boldsymbol{A}_1^\intercal, \boldsymbol{A}_2^\intercal, \cdots, \boldsymbol{A}_N^\intercal\right]^\intercal$ and $\boldsymbol{V} = \operatorname{BlockDiag}(\boldsymbol{V}_1, \boldsymbol{V}_2, \cdots, \boldsymbol{V}_N)$. By using Equation (5.48) we have:

$$\begin{aligned}
\frac{1}{2}\mathbb{E}\left(\left(\hat{G}(s,t) - G(s,t)\right)^2\right) \leq {}& \left(\mathbb{E}\left(\breve{G}(s,t)\right) - G(s,t)\right)^2 + \mathbb{E}\left(f(s,t)\right)^2 \\
& + \operatorname{var}\left(f(s,t)\right) + \operatorname{var}\left(\breve{G}(s,t)\right)
\end{aligned}. \tag{5.49}$$

As in [89] we consider each term of Equation (5.49) separately. First we consider $\|\mathbb{E}\left(f\right)\|$. We note that:

$$\|\mathbb{E}\left(f\right)\| = \|\bar{\boldsymbol{B}}^\intercal(\cdot,\cdot)\,\boldsymbol{H}_{G,N}^{-1}\left(\boldsymbol{A}^\intercal\boldsymbol{V}\tilde{\boldsymbol{d}}^*\right)\| \leq \|\boldsymbol{H}_{G,N}^{-1}\|_\infty\|\boldsymbol{A}^\intercal\boldsymbol{V}\tilde{\boldsymbol{d}}^*\|_{\max}.$$

Let u_{lm} be the $(l,m)^{\text{th}}$ element of $\boldsymbol{A}^{\mathsf{T}}\boldsymbol{V}\tilde{\boldsymbol{d}}^{*}$. Then:

$$u_{lm} = \sum_{i=1}^{N} v_i \sum_{j_1=1}^{J} \sum_{\substack{j_2=1 \\ j_1 \neq j_2}}^{J} \bar{B}^{\tau}_{d,l}(t_{ij_1})\bar{B}^{\tau}_{d,m}(t_{ij_2}) d_{ij_1j_2}.$$

Then:

$$\begin{aligned} |u_{lm}| &\leq \max_{ij_1j_2} |d_{ij_1j_2}| \sum_{j_1=1}^{J} \sum_{\substack{j_2=1 \\ j_1 \neq j_2}}^{J} \bar{B}^{\tau}_{d,l}(t_{ij_1})\bar{B}^{\tau}_{d,m}(t_{ij_2}) \\ &\leq \max_{ij_1j_2} |d_{ij_1j_2}| \int\int \bar{B}^{\tau}_{d,l}(s)\bar{B}^{\tau}_{d,l}(d) d_s d_t R_N(s,t) \\ &= O(U_1 h_G^2), \end{aligned}$$

where $R_N(s,t) = \sum_{i=1}^{N} v_i \sum_{j_1=1}^{J} \sum_{\substack{j_2=1 \\ j_1\neq j_2}}^{J} 1_{t_{ij_1}<s} 1_{t_{ij_2}<t}$ and the final line follows from Assumption 5.6 and Equation (5.44). By combining Equation (5.52) and Lemma 5.12 we have:

$$\|\mathbb{E}\left(f\right)\| = O(U_1). \tag{5.53}$$

As in [89], we next consider the bound for the variance of f.

$$\begin{aligned} \operatorname{Var}\left(f(s,t)\right) &= \bar{\boldsymbol{B}}^{\mathsf{T}}(s,t)\boldsymbol{H}^{-1}_{G,N}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{V}\mathbb{E}\left(\tilde{\boldsymbol{d}}^{*}\tilde{\boldsymbol{d}}^{*\mathsf{T}}\right)\boldsymbol{V}\boldsymbol{A}\boldsymbol{H}^{-1}_{G,N}\bar{\boldsymbol{B}}(s,t) \\ &\leq \|\boldsymbol{H}^{-1}_{G,N}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{V}\mathbb{E}\left(\tilde{\boldsymbol{d}}^{*}\tilde{\boldsymbol{d}}^{*\mathsf{T}}\right)\boldsymbol{V}\boldsymbol{A}\boldsymbol{H}^{-1}_{G,N}\|_{\max} \\ &\leq \|\boldsymbol{H}^{-1}_{G,N}\|^2_{\infty}\|\boldsymbol{A}^{\mathsf{T}}\boldsymbol{V}\|^2_{\infty}\|\mathbb{E}\left(\tilde{\boldsymbol{d}}^{*}\tilde{\boldsymbol{d}}^{*\mathsf{T}}\right)\|_{\max}, \end{aligned}$$

By Assumption 5.6 and the form of $\tilde{d}_{ij_1j_2}$ we have that $\|\mathbb{E}\left(\tilde{\boldsymbol{d}}^{*}\tilde{\boldsymbol{d}}^{*\mathsf{T}}\right)\|_{\max} = O(U_1^2)$. We also have, as stated in [89], that $\|\boldsymbol{A}^{\mathsf{T}}\boldsymbol{V}\|_{\infty} = O(h_G^2)$ and $\|\boldsymbol{H}^{-1}_{G,N}\|_{\infty} = O(h_G^{-2})$. Combining the above we have:

$$\operatorname{Var}\left(f(s,t)\right) = O(U_1^2). \tag{5.57}$$

As in [89] we consider $\|\mathbb{E}\left(\check{G}\right) - G|$. First we note that:

$$\begin{aligned} \mathbb{E}\left(\check{G}(s,t)\right) &= \bar{\boldsymbol{B}}^{\mathsf{T}}(s,t)\boldsymbol{H}^{-1}_{G,N}\left(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{V}\boldsymbol{g}^{*}\right) \\ &= \bar{\boldsymbol{B}}^{\mathsf{T}}(s,t)\boldsymbol{G}^{-1}_{G,N}\left(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{V}\boldsymbol{g}^{*}\right) - \bar{\boldsymbol{B}}^{\mathsf{T}}(s,t)\boldsymbol{H}^{-1}_{G,N}\boldsymbol{P}_G\boldsymbol{G}^{-1}_{G,N}\left(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{V}\boldsymbol{g}^{*}\right). \end{aligned}$$

Letting ν_G be defined as in Lemma 5.7 such that:

$$\|G - \nu_G\| = O(h_G^m) + o(h_G^p),$$

and noting that $\nu_G(s,t) = \bar{\boldsymbol{B}}^{\mathsf{T}}(s,t)\boldsymbol{G}^{-1}_{G,N}\left(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{V}\boldsymbol{\nu}^{*}_G\right)$ where $\boldsymbol{\nu}^{*}_G$ is defined similarly to $\boldsymbol{g}^{*}$. We have:

$$\mathbb{E}\left(\check{G}(s,t)\right) = \nu_G(s,t) + \bar{\boldsymbol{B}}^{\mathsf{T}}(s,t)\boldsymbol{G}^{-1}_{G,N}\boldsymbol{\alpha}_G - \bar{\boldsymbol{B}}^{\mathsf{T}}(s,t)\boldsymbol{H}^{-1}_{G,N}\boldsymbol{P}_G\boldsymbol{\gamma}_G,$$

where $\boldsymbol{\alpha}_G = \boldsymbol{A}^{\intercal}\boldsymbol{V}\left(\boldsymbol{g}^* - \boldsymbol{\nu}_G^*\right)$ and $\boldsymbol{\gamma}_G = \boldsymbol{G}_{G,N}^{-1}\left(\boldsymbol{A}^{\intercal}\boldsymbol{V}\boldsymbol{g}^*\right)$. It follows that:

$$\mathbb{E}\left(\check{G}(s,t)\right) - G(s,t) = \nu_G(s,t) - G(s,t) + \bar{\boldsymbol{B}}^{\intercal}(s,t)\boldsymbol{G}_{G,N}^{-1}\boldsymbol{\alpha}_G - \bar{\boldsymbol{B}}^{\intercal}(s,t)\boldsymbol{H}_{G,N}^{-1}\boldsymbol{P}_G\boldsymbol{\gamma}_G.$$

Then:

$$\|\mathbb{E}\left(\check{G}\right) - G\| \leq \|G - \nu_G\| + \|\boldsymbol{G}_{G,N}^{-1}\boldsymbol{\alpha}_G\|_{\max} + \|\boldsymbol{H}_{G,N}^{-1}\boldsymbol{P}_G\boldsymbol{\gamma}_G\|_{\max}. \tag{5.60}$$

Following [89] by using Lemma 5.7 and Assumption 5.5 one can show that $\|\boldsymbol{\alpha}_g\| = O(h_G^{p+2})$. By Lemma 5.10 we have $\|\boldsymbol{G}_{G,N}^{-1}\|_\infty = O(h_G^{-2})$ which leads to a bound on the middle term as:

$$\|\boldsymbol{G}_{G,N}^{-1}\boldsymbol{\alpha}_G\|_{\max} = o(h_G^p). \tag{5.61}$$

It remains to bound $\|\boldsymbol{H}_{G,N}^{-1}\boldsymbol{P}_G\boldsymbol{\gamma}_G\|_{\max}$. Following the steps taken in [89] and using Lemmas 5.13, 5.14 we obtain the bound:

$$\|\boldsymbol{H}_{G,N}^{-1}\boldsymbol{P}_G\boldsymbol{\gamma}_G\|_{\max} = O(\omega_G h_{G,e}^{-q}). \tag{5.62}$$

We omit the details for this bound and refer the reader to [89] for the full derivation. It is exactly the same up to a change notation and does not add much to the intuition of the proof hence its omission. Combining Equations (5.60), (5.61), (5.62) and using Lemma 5.7 we obtain:

$$\|\mathbb{E}\left(\check{G}\right) - G\| = O(h_G^m) + o(h_G^p) + O(\omega_G h_{G,e}^{-q}). \tag{5.63}$$

Finally we need to bound the var $\left(\check{G}(s,t)\right)$ term. In what follows we deviate from [89] due to the dependence in observations complicating the variance term. We note that:

$$\text{var}\left(\check{G}(s,t)\right) = \bar{\boldsymbol{B}}^{\intercal}(s,t)\boldsymbol{H}_{G,N}^{-1}\boldsymbol{A}^{\intercal}\boldsymbol{V}\mathbb{E}\left(\check{\boldsymbol{g}}^*\check{\boldsymbol{g}}^{*\intercal}\right)\boldsymbol{V}\boldsymbol{A}\boldsymbol{H}_{G,N}^{-1}\bar{\boldsymbol{B}}(s,t).$$

Then letting:

$$\begin{aligned}\tilde{\Pi} &= \left[\tilde{\pi}_{l_1m_1l_2m_2}\right] \\ &= \boldsymbol{A}^{\intercal}\boldsymbol{V}\mathbb{E}\left(\check{\boldsymbol{g}}^*\check{\boldsymbol{g}}^{*\intercal}\right)\boldsymbol{V}\boldsymbol{A} \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N}\boldsymbol{A}_i^{\intercal}\boldsymbol{V}_i\mathbb{E}\left(\check{\boldsymbol{g}}_i^*\check{\boldsymbol{g}}_j^{*\intercal}\right)\boldsymbol{V}_j\boldsymbol{A}_j.\end{aligned}$$

Let $R_{ij} = \mathbb{E}\left(\check{\boldsymbol{g}}_i^*\check{\boldsymbol{g}}_j^{*\intercal}\right)$. Then the elements of R_{ij} can be written as $r_{ijj_1j_2j_3j_4}$ with the rows indexed by (j_1, j_2) and the columns by (j_3, j_4). Then $r_{ijj_1j_2j_3j_4} = \mathbb{E}\left(e_{ij_1}e_{ij_2}e_{jj_3}e_{jj_4}\right)$ and we can write:

$$\tilde{\pi}_{l_1m_1l_2m_2} = \sum_{i=1}^{N}\sum_{j=1}^{N}v_iv_j\sum_{j_1\neq j_2, j_3\neq j_4}^{J}\bar{B}_{d,l_1}^{\tau}(t_{ij_1})\bar{B}_{d,m_1}^{\tau}(t_{ij_2})\bar{B}_{d,l_2}^{\tau}(t_{jj_3})\bar{B}_{d,m_2}^{\tau}(t_{jj_4})r_{ijj_1j_2j_3j_4},$$

which we can break into two parts corresponding to the variance and cross-variance components respectively. That is $\tilde{\Pi} = \tilde{\Pi}_1 + \tilde{\Pi}_2$ where $\tilde{\Pi}_1 = \left[\tilde{\pi}_{l_1m_1l_2m_2}^1\right]$ and $\tilde{\Pi}_2 = \left[\tilde{\pi}_{l_1m_1l_2m_2}^2\right]$

:

$$
\begin{aligned}
\tilde{\pi}^1_{l_1 m_1 l_2 m_2} &= \sum_{i=1}^N v_i^2 \sum_{j_1 \neq j_2, j_3 \neq j_4}^J \bar{B}^{\tau}_{d,l_1}(t_{ij_1}) \bar{B}^{\tau}_{d,m_1}(t_{ij_2}) \bar{B}^{\tau}_{d,l_2}(t_{ij_3}) \bar{B}^{\tau}_{d,m_2}(t_{ij_4}) r_{iij_1j_2j_3j_4}, \\
\tilde{\pi}^2_{l_1 m_1 l_2 m_2} &= \sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N v_i v_j \sum_{j_1 \neq j_2, j_3 \neq j_4}^J \bar{B}^{\tau}_{d,l_1}(t_{ij_1}) \bar{B}^{\tau}_{d,m_1}(t_{ij_2}) \bar{B}^{\tau}_{d,l_2}(t_{jj_3}) \bar{B}^{\tau}_{d,m_2}(t_{jj_4}) r_{ijj_1j_2j_3j_4}.
\end{aligned}
$$

Now to bound the above we consider defining the following:

$$
\begin{aligned}
\pi^1_{l_1 m_1 l_2 m_2} &= \sum_{i=1}^N v_i^2 \sum_{j_1 \neq j_2, j_3 \neq j_4}^J \bar{B}^{\tau}_{d,l_1}(t_{ij_1}) \bar{B}^{\tau}_{d,m_1}(t_{ij_2}) \bar{B}^{\tau}_{d,l_2}(t_{ij_3}) \bar{B}^{\tau}_{d,m_2}(t_{ij_4}), \\
\pi^2_{l_1 m_1 l_2 m_2} &= \sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N v_i v_j \sum_{j_1 \neq j_2, j_3 \neq j_4}^J \bar{B}^{\tau}_{d,l_1}(t_{ij_1}) \bar{B}^{\tau}_{d,m_1}(t_{ij_2}) \bar{B}^{\tau}_{d,l_2}(t_{jj_3}) \bar{B}^{\tau}_{d,m_2}(t_{jj_4}) |r_{ijj_1j_2j_3j_4}|.
\end{aligned}
$$

Then,

$$
|\tilde{\pi}_{l_1 m_1 l_2 m_2}| \leq \max_{i,j_1,j_2,j_3,j_4} |r_{iij_1j_2j_3j_4}| \pi^1_{l_1 m_1 l_2 m_2} + \pi^2_{l_1 m_1 l_2 m_2}.
$$

This holds uniformly for l_1, m_1, l_2, m_2. Additionally we have due to Assumption 5.5 $|r_{iij_1j_2j_3j_4}| \leq 2^4 \left(\mathbb{E}\left(\|\chi_i\|\right)^4 + \mathbb{E}\left(\varepsilon_{ii}^4\right) \right)$ so $\max_{i,j_1,j_2,j_3,j_4} |r_{iij_1j_2j_3j_4}| = O(1)$. Thus as in the mean function estimator we have:

$$
\begin{aligned}
\bar{\boldsymbol{B}}^{\intercal}(s,t) \boldsymbol{H}^{-1}_{G,N} \tilde{\boldsymbol{\Pi}} \boldsymbol{H}^{-1}_{G,N} \bar{\boldsymbol{B}}(s,t) &\leq \bar{\boldsymbol{B}}^{\intercal}(s,t) \left(\boldsymbol{H}^{-1}_{G,N} \right)_+ \left(\tilde{\boldsymbol{\Pi}} \right)_+ \left(\boldsymbol{H}^{-1}_{G,N} \right)_+ \bar{\boldsymbol{B}}(s,t) \\
&\leq \bar{\boldsymbol{B}}^{\intercal}(s,t) \left(\boldsymbol{H}^{-1}_{G,N} \right)_+ \boldsymbol{\Pi} \left(\boldsymbol{H}^{-1}_{G,N} \right)_+ \bar{\boldsymbol{B}}(s,t)
\end{aligned},
$$

where $\boldsymbol{\Pi} = \boldsymbol{\Pi}_1 + \boldsymbol{\Pi}_2$. Again by the unity and non-negativity of B-splines we have:

$$
\operatorname{var}\left(\check{G}(s,t) \right) \leq \| \left(\boldsymbol{H}^{-1}_{G,N} \right)_+ \boldsymbol{\Pi} \left(\boldsymbol{H}^{-1}_{G,N} \right)_+ \|_{\max} \leq \| \left(\boldsymbol{H}^{-1}_{G,N} \right)_+ \|^2_{\infty} \|\boldsymbol{\Pi}\|_{\max}.
$$

By Lemma 5.12 we have $\| \left(\boldsymbol{H}^{-1}_{G,N} \right)_+ \|_{\infty} = O(h_G^{-2})$. It remains to bound $\|\boldsymbol{\Pi}\|_{\max}$. We have by construction that $\|\boldsymbol{\Pi}\|_{\max} \leq \|\boldsymbol{\Pi}_1\|_{\max} + \|\boldsymbol{\Pi}_2\|_{\max}$. For $\|\boldsymbol{\Pi}_1\|_{\max}$ we can follow [89] using Lemma 5.13, 5.13, that is:

$$
\|\boldsymbol{\Pi}_1\|_{\max} = O(J^4 h_G^4 \sum_{i=1}^N v_i^2),
$$

as $\max_l |\sum_j \bar{B}^{\tau}_{d,l}(t_j)| = O(J h_G)$. Finally we bound $\boldsymbol{\Pi}_2$ by considering its elements (similar to how we bound $\boldsymbol{\Pi}_1$). We note initially that we can bound $|r_{ijj_1j_2j_3j_4}|$ as:

$$
\begin{aligned}
|r_{ijj_1j_2j_3j_4}| &= |\mathbb{E}\left(e_{ij_1} e_{ij_2} e_{jj_3} e_{jj_4} \right)| \\
&\leq \sup_{s,t \in \mathcal{T}} |a_{\mathcal{X}}(\boldsymbol{s}_i, t, \boldsymbol{s}_j, t')|^2.
\end{aligned}
$$

Let $a^*_{ij} = \sup_{t,t' \in \mathcal{T}} |a_{\mathcal{X}}(\boldsymbol{s}_i, t, \boldsymbol{s}_j, t')|^2$. Then:

$$|\pi^2_{l_1 m_1 l_2 m_2}| \leq J^4 h_G^4 \sum_{i=1}^{N} \sum_{\substack{j=1 \\ i \neq j}}^{N} v_i v_j a^*_{ij},$$

which occurs uniformly in m_1, l_1, m_2, l_2 and thus through construction:

$$\|\mathbf{\Pi}_2\|_{\max} = O(h_G^4 \tilde{\tau}_2).$$

Thus $\|\mathbf{\Pi}\|_{\max} = O(h_G^4(\tilde{\tau}_1 + \tilde{\tau}_2))$. Therefore:

$$\operatorname{var}\left(\check{G}(s,t)\right) = O(\tilde{\tau}_1 + \tilde{\tau}_2). \tag{5.73}$$

Combining Equations (5.53), (5.57), (5.63), (5.73) gives the result of the theorem. □

5.5 Score Covariance Estimation

Following our consistency results for both the mean function estimator and covariance surface estimator, given in Theorems 5.1, 5.2 respectively, we can now consider the practical problems of using such estimators. The main problem left to resolve is to estimate the kernel function $a_{\mathcal{X}}$ hyperparameters. To do so we rely heavily on the theory of Gaussian processes. The basic premise of a Gaussian process has been discussed in Section 3.5.

For simplicity of notation we begin by defining $\boldsymbol{\theta}$ to be the vector of all hyperparameters from the collection $\{\boldsymbol{\theta}_k\}_{k=1}^K$. Our goal is then to estimate $\boldsymbol{\theta}$ from our observations. The standard way to do so is to consider choosing $\boldsymbol{\theta}$ to optimise the maximum marginal log-likelihood. A detailed discussion of such approaches in the general Gaussian process setting is available in [80].

We begin by considering, what is often termed as level two inference, that is inference for the hyperparameters $\boldsymbol{\theta}$. Level one inference being inference for the parameters. To do so we denote $\boldsymbol{X}$ be the design matrix consisting of elements of $\mathcal{S} \times \mathcal{T}$ which correspond to the functional observations $\boldsymbol{Y}$. Then the posterior over the hyperparameters can be expressed as:

$$p\left(\boldsymbol{\theta}|\boldsymbol{Y},\boldsymbol{X}\right) = \frac{p\left(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\theta}\right)p\left(\boldsymbol{\theta}\right)}{\int p\left(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\theta}\right)p\left(\boldsymbol{\theta}\right)d\boldsymbol{\theta}},$$

where $p(\boldsymbol{\theta})$ is the prior over the hyperparameters and $p(\boldsymbol{Y}|\boldsymbol{X}\boldsymbol{\theta})$ is the marginal likelihood or evidence, [80].

As is often the case in Bayesian inference, the normalising constant requires the evaluation of a possibly high dimensional integral which is often hard to analytically evaluate. In general one may resort to approximate regimes or Markov Chain Monte Carlo (MCMC) methods to evaluate this. A common method to overcome this in practice is to ignore the hyperparameter posterior, but choose hyperparameters based on maximising the marginal log-likelihood with respect to the hyperparameters. Under Gaussian error assumptions this has a nice form for our CPACE model, [80]. Let $\tilde{\boldsymbol{Y}}$ be $\boldsymbol{Y} - \boldsymbol{\mu}$, where $\boldsymbol{\mu}$

is our mean function evaluated at observation points corresponding to $\boldsymbol{Y}$. Then:

$$\log p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\theta}) = -\frac{1}{2}\tilde{\boldsymbol{Y}^\intercal}\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{Y}},\tilde{\boldsymbol{Y}}\right)^{-1}\tilde{\boldsymbol{Y}} - \frac{1}{2}\log|\boldsymbol{\Sigma}\left(\tilde{\boldsymbol{Y}},\tilde{\boldsymbol{Y}}\right)| - \frac{\sum_{i=1}^{N} J_i}{2}\log 2\pi, \tag{5.74}$$

where $\boldsymbol{\Sigma}(\boldsymbol{Y},\boldsymbol{Y})$ is the variance matrix observed which is constructed from evaluating the kernel, $a_{\mathcal{X}}$ at the observed design matrix $\boldsymbol{X}$ and adding on the error variance. In order to evaluate this kernel we must obtain the K eigenfunctions of the covariance surface. As discussed in Section 5.4 we can use the covariance function estimator to obtain this and the resulting eigenfunction decomposition of this can the be used as an estimator for the eigenfunctions present in $a_{\mathcal{X}}$. Maximising the marginal log-likelihood given in Equation (5.74) then gives a point estimate to use as our estimated hyperparameters. That is:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\theta}).$$

There are numerous minimisation routines that can be possibly used for such high dimensional minimisation. A good reference for these routines is [19]. As can be seen, such optimisation requires the evaluation of possibly large matrices which can be computationally intensive. Fortunately, the structure of our model alleviates this issue somewhat, but other methods can be utilised to speed up computation. Details on the practical implementation of hyperparameter optimisation and these methods can be found in Chapter 8. We also discuss the use of approximation for the fully Bayesian implementation in this chapter too.

Once we obtain the estimate for the hyperparameters of the kernel function we can combine these with the estimates for the variance, eigenfunctions and noise variance obtained as described above. This combination of estimators specifies our entire CPACE model.

5.6 Reconstruction

Using the above estimation techniques, we have all components of our model to produce reconstructions. Reconstruction can be on any point in the domain $\mathcal{S} \times \mathcal{T}$ and follows nicely using the Gaussian process framework. As discussed in Section 3.5 under Gaussian error assumptions the framework has a closed form for prediction at a new point in the domain. Let $\boldsymbol{\chi}^*$ denote the vector of functional observations at new design points $\boldsymbol{X}^*$ then, [80]:

$$\mathbb{E}\left(\boldsymbol{\chi}^*|\boldsymbol{Y},\boldsymbol{X},\boldsymbol{X}^*\right) = \boldsymbol{\mu}^* + \boldsymbol{\Sigma}\left(\boldsymbol{\chi}^*,\boldsymbol{\chi}\right)\boldsymbol{\Sigma}\left(\boldsymbol{Y},\boldsymbol{Y}\right)^{-1}\left(\boldsymbol{Y}-\boldsymbol{\mu}\right),$$

where $\boldsymbol{\mu}^*$ is the vector of the mean function evaluated at points $\boldsymbol{X}^*$ and $\boldsymbol{\Sigma}\left(\boldsymbol{\chi},\boldsymbol{\chi}^*\right)$ is the matrix formed of the covariance kernel $a_{\mathcal{X}}$ evaluated at all pairs of locations in $\boldsymbol{X}$ and $\boldsymbol{X}^*$. In most practical cases the evaluation of the kernel $a_{\mathcal{X}}$ is done using estimated hyperparameters, mean function, and eigenfunctions as laid out in the Sections 5.5, 5.3, 5.4 respectively.

Similarly we can obtain the variance of such predicted points by the following:

$$\operatorname{var}\left(\boldsymbol{\chi}^{*}|\boldsymbol{Y},\boldsymbol{X},\boldsymbol{X}^{*}\right)=\boldsymbol{\Sigma}\left(\boldsymbol{\chi}^{*},\boldsymbol{\chi}^{*}\right)-\boldsymbol{\Sigma}\left(\boldsymbol{\chi}^{*},\boldsymbol{\chi}\right)\boldsymbol{\Sigma}\left(\boldsymbol{Y},\boldsymbol{Y}\right)^{-1}\boldsymbol{\Sigma}\left(\boldsymbol{\chi}^{*},\boldsymbol{\chi}\right)^{\intercal},$$

where $\boldsymbol{\Sigma}\left(\boldsymbol{\chi}^{*},\boldsymbol{\chi}^{*}\right)$ is the matrix formed of evaluating the covariance kernel $a_{\mathcal{X}}$ at design points $\boldsymbol{X}^{*}$. Again, in a practical sense as with the mean prediction, we would replace by estimators the kernel hyperparameters, mean function and eigenfunctions needed in the CPACE model.

Under Gaussian error assumptions the distribution of the reconstructed points is normal with mean and variance given above. For non-Gaussian independent error extensions to the Gaussian process formula above exist and we refer the reader to [80, Ch. 9] and the references within.

This chapter has given a specification to the CPACE model and provides estimators for all key components of the model. We have shown under certain assumptions that both the mean function and covariance surfaces can be estimated consistently with penalised spline regression of the appropriate dimension. These can then be utilised to allow for estimation of the functional variables at unobserved points through the use of the Gaussian process framework. In this sense the CPACE model combines the attributes of the traditional FPCA model by maintaining eigenfunction for the temporal dimension but allows for more complicated spatial component by using the Gaussian process framework. One advantage of this is that we have abstracted the notion of dependency between observed functional variables to that of a Gaussian process covariance kernel for each eigenfunction. This then allows one to suitably choose the correct process for each eigenfunction and the choice of kernel can allow for many interesting phenomena. There are a wide variety of known covariance kernels which mean this methodology has particular flexibility. In the following chapter, Chapter 6, we present an application of the CPACE methodology to the CESM-LE dataset as well as simulation studies investigating the proposed estimators. We highlight the flexibility of this framework by considering a variety of parametric kernel functions to help capture the various intricate dependency observed between these functional data.

Chapter 6

Simulation Study for CPACE Model

The CPACE model as described in Chapter 5 is designed to allow for functional data which are observed dependently. Most often this may be a spatial dependence between neighbouring trajectories although the framework discussed is not reliant on this. We have shown theoretically how we would reconstruct unobserved trajectories using our estimators. However, we are yet to highlight in practice the ability of such estimators and their reconstruction ability. In the following chapter we present a series of simulated results designed to test the estimator capabilities, namely the ability to recover model hyperparameters and its reconstruction ability for unobserved locations. We contrast the CPACE models results against the traditional PACE model which does not utilise the spatial information of the functional data.

6.1 Simulation Study

In the following section we present a simulation study based on the simulation study of [90]. The simulation study was designed in [90] to showcase the implementation of sparse functional principal components analysis which are observed independently. We will take this study as a base and then develop their generating procedure to allow for spatial dependency between functional observations. We refer, in the following, to spatial dependency between functional observations because our main application of the CPACE model is the EO data, which naturally has spatial dependency. However, this simulation study, and indeed the CPACE model can naturally be applied with different dependency between functional observations.

The simulation study consists of four scenarios; namely A, B, C, and D. In each scenario the functional data are generated from the same mean and principal components. We vary the spatial dependence between each scenario to show how the CPACE model with different spatial kernels can accommodate differing spatial structures. We detail the exact specification for each scenario in its respective subsection. First we describe the common underlying data generating process for the functional data.

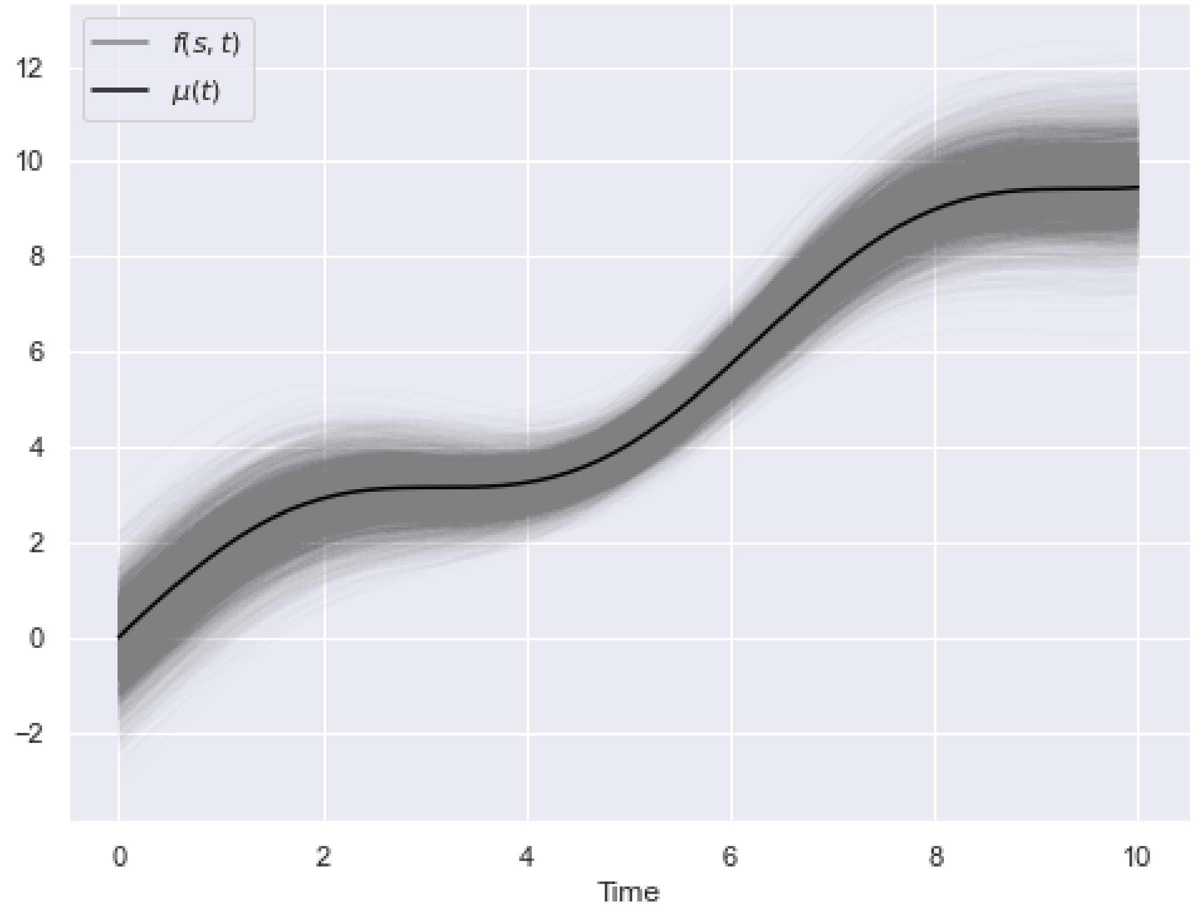


Fig. 6.1 The mean function chosen for the simulation study over the temporal domain. We have illustrated example functional data simulated using this mean function in grey.

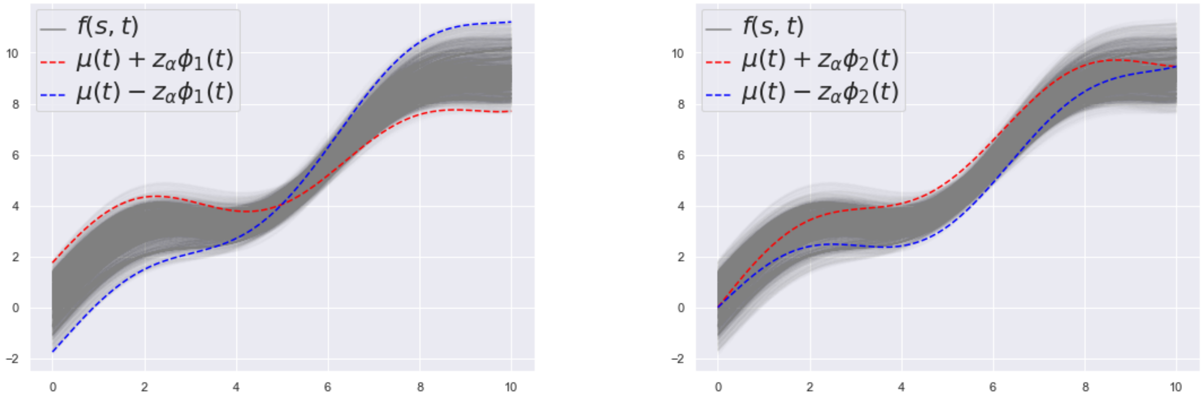
6.1.1 Data Generating Process

We propose the simulations to be generated over temporal domain $\mathcal{T} \in [0, 10] \subset \mathbb{R}$. The processes have mean, $\mu(t) = t + \sin(t)$, and generated as in Equation (5.6) with $K = 2$. The eigenfunctions are given by:

$$\begin{aligned} \phi_1(t) &= -\frac{1}{\sqrt{5}} \cos\left(\frac{\pi t}{10}\right), \\ \phi_2(t) &= -\frac{1}{\sqrt{5}} \sin\left(\frac{\pi t}{10}\right). \end{aligned}$$

Figure 6.1 displays the mean function we use for the simulation, and Figure 6.2 highlights the variation from the mean function that each eigenfunction contributes. As can be seen, the mean function used for the simulation has a clear upward trend and a periodic component. The first eigenfunction represent periodic variation at the start and end of the temporal domain, with the second eigenfunction giving variation in the middle of the temporal domain.

We propose to simulate observations on the spatial domain of $\mathcal{S} = [-5, 5] \times [-5, 5] \subset \mathbb{R}^2$. We choose this spatial domain without great consideration as we can always rescale our



(a) Variation from the mean function caused by the first eigenfunction.

(b) Variation from the mean function caused by the second eigenfunction.

Fig. 6.2 An example of the variation due to the two eigenfunctions from the mean function. The kernel variance chosen for this study is $\lambda_1 = 4$ and $\lambda_2 = 1$ (Discussed in detail below). We illustrate the impact of this variation with a z-score corresponding to $\alpha = 0.05$. That is that there is a 95% probability that the mean plus the respective eigenfunction lies within the upper and lower bounds. Again we highlight example functional data generated from this setup in grey.

input to this domain. It so happens that this scale for the domain allows for us to flexibly choose scenarios with varying degrees of spatial dependency. Each location $\boldsymbol{s} \in \mathcal{S}$ can give rise to a functional data generated by the above mean and principal components by Equation (5.6). In order to generate such data in practice we discretise the temporal domain to a fine grid by segmenting $\mathcal{T}$ into 128 equal segments, and use the midpoints for generation of the functional data. For each functional observation we suppose, as in [90] and our setup in Chapter 1, that we do not observe the full true function $\mathcal{X}(\boldsymbol{s}, t)$ but we sparsely observe a noisy version of it. For all the simulations we suppose our noise variance is $\sigma_\epsilon^2 = 0.25$. We set the sparsity of observations to be between 5% and 10% of our full 128 temporal grid. This is in line with the simulation study of Yao et al.. That is, our number of observation points are chosen uniformly from $[6, 7, \cdots, 11, 12]$. The observation points are sampled randomly from the full discretised grid over $\mathcal{T}$. This is a slight deviation from the simulation study in [90] but lies closer to reality for EO data which is often observed at regular grid intervals. Figure 6.3 highlights an example of the sparsity and observation error described above.

For the spatial domain, $\mathcal{S} \subset \mathbb{R}^2$ we take a similar approach. Again in practice we discretise this into 64 equally spaced segments across the first domain, and 48 equally spaced segments across the second domain. We take the midpoints as our discretised sampling locations. This gives in total 64×48 possible sampling locations for our simulation study. We take half of our sampling locations as observed data, that is used to estimate the parameters of the CPACE model. Namely, the spatial kernel hyperparameters and eigen decomposition parameters. As we only observe each functional datum sparsely this corresponds to a training dataset of approximately 3.75% of the total discretised points of simulation.

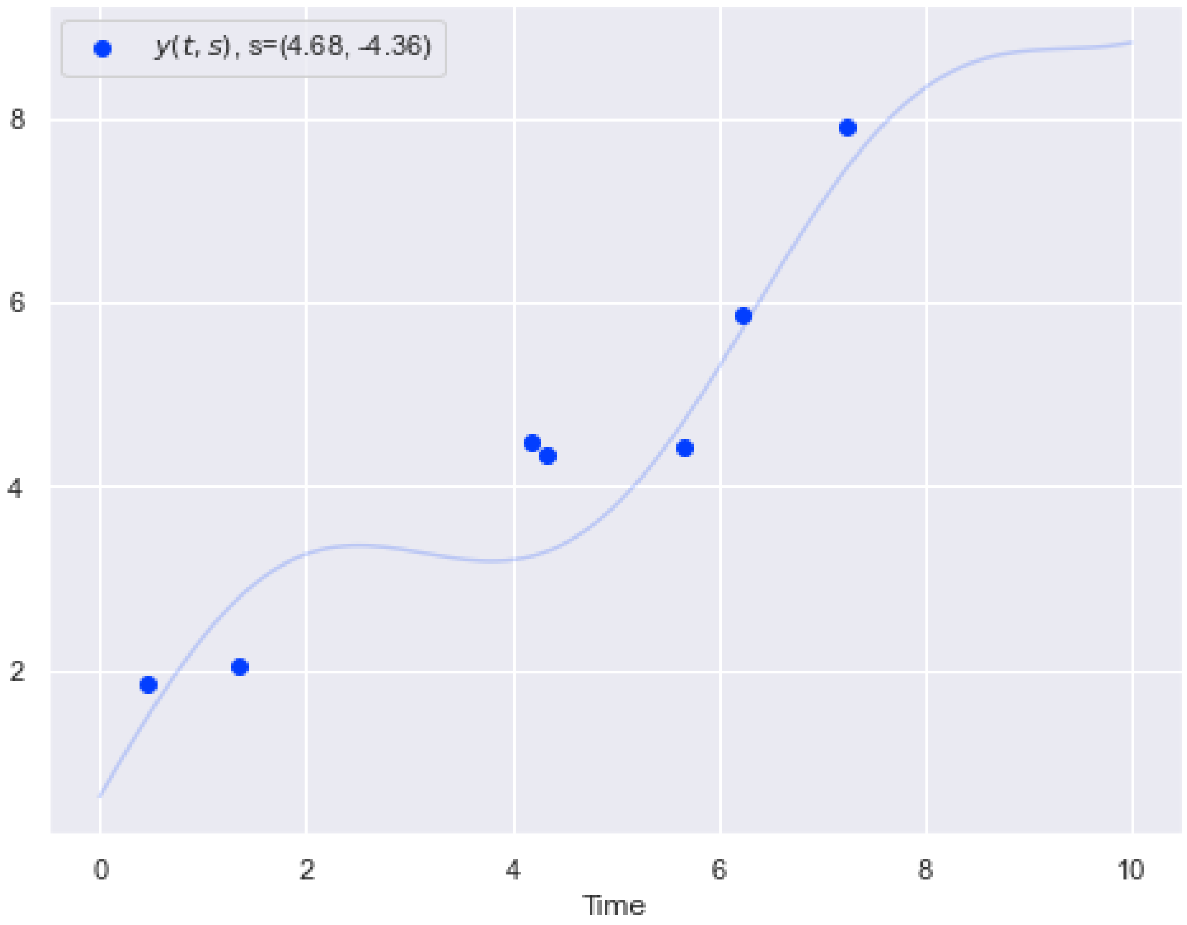


Fig. 6.3 An example functional data with observation points. This example is taken from Scenario A and highlights the sparsity of observation as well as the impact of the noise variance on observations.

As stated above, to simulate spatially dependent functional data we must simulate a realisation of our spatial covariance process $\xi_i(\boldsymbol{s})$ for $i = 1, 2$. The particular form of this process will change in each scenario but we keep the variance of these processes consistent to align closely with the study in [90]. That is, we set the variance of ξ_1 to be 4 and ξ_2 to be 1, which corresponds to $\lambda_1 = 4$ and $\lambda_2 = 1$ from Chapter 5.

Finally, we simulate 50 replications using the above data generating procedure to produce our scenario dataset. For each simulation of each scenario we will evaluate various versions of the CPACE model against the standard FPCA model which does not take into account the spatial correlation between functional observations. We discuss the evaluation of our models against these simulated data in the following section.

6.1.2 Evaluation Metrics

To evaluate the performance of our model we consider two standard metrics; the mean square error, and the mean absolute error. These are standard evaluation metrics for a regression based model but for clarity the mean square error and mean absolute error for prediction $\hat{\mathcal{X}}$ against unobserved functional data $\mathcal{X}$ is given by:

$$\begin{aligned} MSE &= \frac{1}{50}\sum_{i=1}^{50}\left(\hat{\mathcal{X}}(\boldsymbol{s},t) - \mathcal{X}(\boldsymbol{s},t)\right)^2, \\ MAE &= \frac{1}{50}\sum_{i=1}^{50}\left|\hat{\mathcal{X}}(\boldsymbol{s},t) - \mathcal{X}(\boldsymbol{s},t)\right|. \end{aligned}$$

In practice, we evaluate these on the discrete simulation grid and use numerical integration to approximate these metrics.

For our simulation study we have three distinct sets of data to evaluate performance against. The training dataset, which comprises solely of the location and time points for which we observe, noisily, the functional data. The validation dataset which comprises of the locations of where we observe our noisy data, but including unobserved time points. Finally, the test dataset which comprises of completely unseen locations and time points.

Comparing the performance of the CPACE model against the standard FPCA model for each dataset should highlight how well our CPACE model performs under the three separate conditions. Greater performance on the test dataset is most preferable as it provides insight into functions at unobserved locations. To place back in the context of EO data, this could be useful to interpolate data where it is not possible to get physical observations. The validation dataset highlights the ability of the model to recover observations from a possibly malfunctioning data source which leads to partial observations at a particular location. The training dataset is typically the easiest to achieve good performance on and is most suitable for evaluation of the model's ability to overcome observation error.

In this study we will examine the performance of the CPACE model with four different spatial kernels. Three of these correspond to stationary kernels, and one is designed to model non-stationary spatial dependence. We detail them below:

6.1.3 Spatial Kernels

We choose four kernels to examine. The first of which corresponds to the kernel with no spatial dependence, and is chosen to highlight the ability of the CPACE model to recreate the PACE model. The next two correspond to common spatial kernels using the Matérn covariance form, one of which is used in the SPACE model, [48]. These highlight the CPACE model's ability to capture quite simplistic spatial correlation structures. The final, is a less commonly used kernel which capture non-stationarity in the spatial dependence. This kernel is chosen to highlight that the CPACE model can effectively be used in cases of highly complex spatial dependence. As in Chapter 3 we refer the reader to [12] for a more in depth discussion of various covariance structures in spatial statistics.

White Kernel

The White, or Independent kernel, is the simplifying assumption which reverts the CPACE model to the standard PACE model. That is, it defines the covariance between $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ as follows:

$$a_k\left(\boldsymbol{s}_i, \boldsymbol{s}_j\right) = \lambda_k \delta_{ij},$$

where

$$\delta_{ij} = \begin{cases} 1 & \text{if } \boldsymbol{s}_i = \boldsymbol{s}_j, \\ 0 & \text{otherwise} \end{cases}.$$

This kernel places zero correlation between functional data at differing spatial locations, and a variance of $\lambda_k \in \mathbb{R}^+$ for functional observations at the same locations. Note that this corresponds to the full cross-correlation of $\mathcal{X}\left(\boldsymbol{s}_i, t\right)$ and $\mathcal{X}\left(\boldsymbol{s}_j, t\right)$ under our initial example given by Equation (3.3) in Chapter 3. The added distinction that we are now using is the use of parameter $\boldsymbol{s}$ to indicate spatial location over the domain $\mathcal{S}$ rather than an simple discrete index. As such the use of this kernel in the CPACE model will result in essentially a differing view of the PACE model under which there is assumed no spatial correlation.

Matérn One Half Kernel

The Matérn kernel is a standard in the spatial statistics literature, [12]. The general form in our setting is given by, [12]:

$$a_k\left(\boldsymbol{s}_i, \boldsymbol{s}_j\right) = \lambda_k \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}\frac{d}{\rho}\right)^{\nu} K_\nu \left(\sqrt{2\nu}\frac{d}{\rho}\right), \tag{6.2}$$

where $d = \|\boldsymbol{s}_i - \boldsymbol{s}_j\|$, K_ν is the Modified Bessel function of the second kind, Γ is the gamma function, $\rho \in \mathbb{R}^+$ is a length scale parameter, and $\nu \in \mathbb{R}^+$ is a parameter controlling the shape of the covariance. The Matérn One Half kernel is a specific example of the general form by setting $\nu = 0.5$ in Equation (6.2). When setting this shape parameter it leads to a simple closed form, given below:

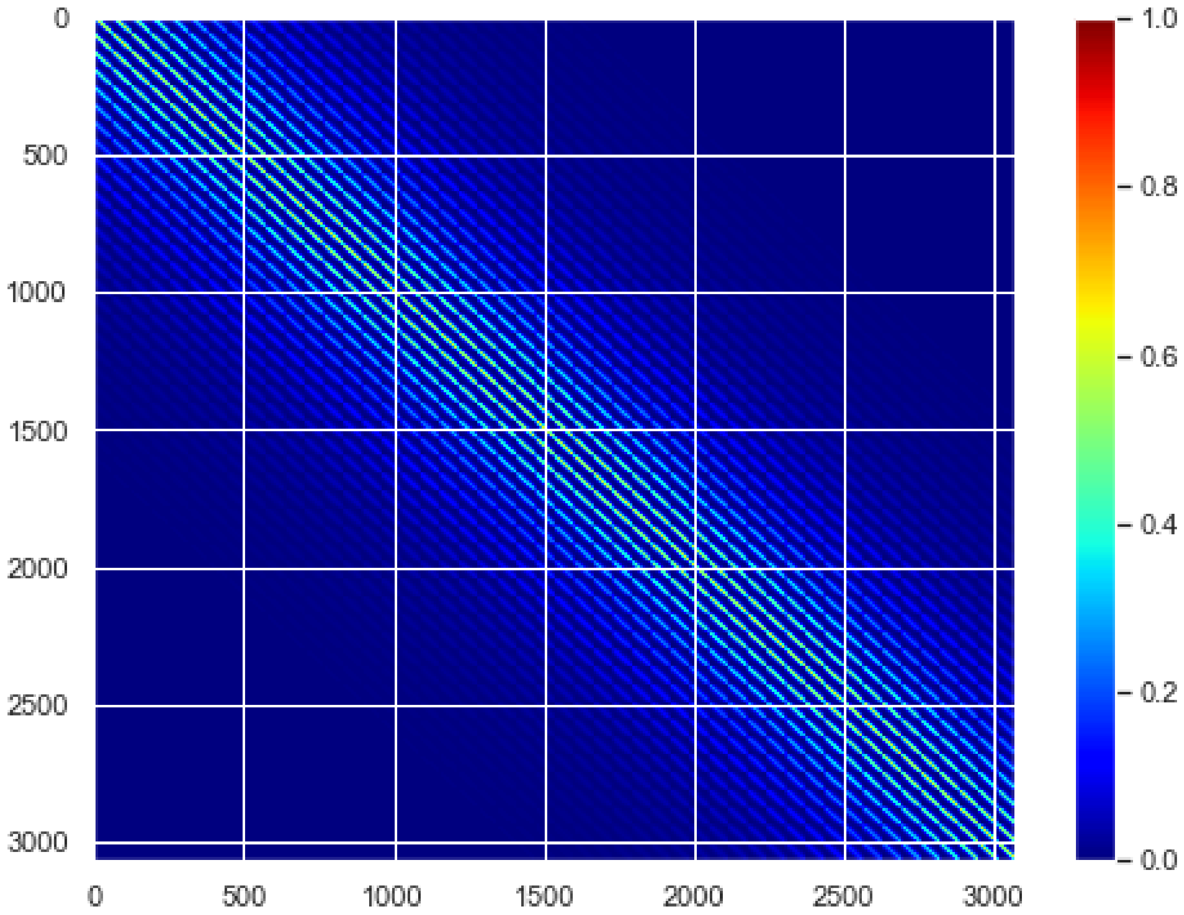


Fig. 6.4 The covariance between all points in our spatial domain $\mathcal{S}$ with the Matérn One Half covariance function with $\rho = 1.5$ and $\lambda = 1$. Each index on the axis corresponds to a point $\boldsymbol{s}_i$ in $\mathcal{S}$ in our discretised grid.

$$a_k\left(\boldsymbol{s}_i, \boldsymbol{s}_j\right) = \lambda_k \exp\left(-\frac{d}{\rho}\right).$$

This kernel is often used due to its simplicity to compute while it maintains good ability to capture stationary covariances. We use the Euclidean distance for the calculation of d which makes this kernel isotropic. The length scale parameter, $\rho \in \mathbb{R}^+$, is used to capture the correlation over differing spatial scales. In the CPACE model we treat ρ as a hyperparameter to estimate. This estimation procedure is described in Chapter 5 with more implementation details in Chapter 8. Figure 6.4 shows the covariance on our spatial domain S with length scale, $\rho = 1.5$, and variance λ_k equal to 1 .

The use of this model is akin to the SPACE model, [48]. However in the CPACE model the estimation procedure for hyperparameters is different.

Matérn Three Halves Kernel

Similar to the Matérn One Half kernel this is another specific example of the general Matérn covariance kernel. This time we set the shape parameter ν to 1.5 in Equation (6.2). This leads to the specific closed form of the covariance kernel given below:

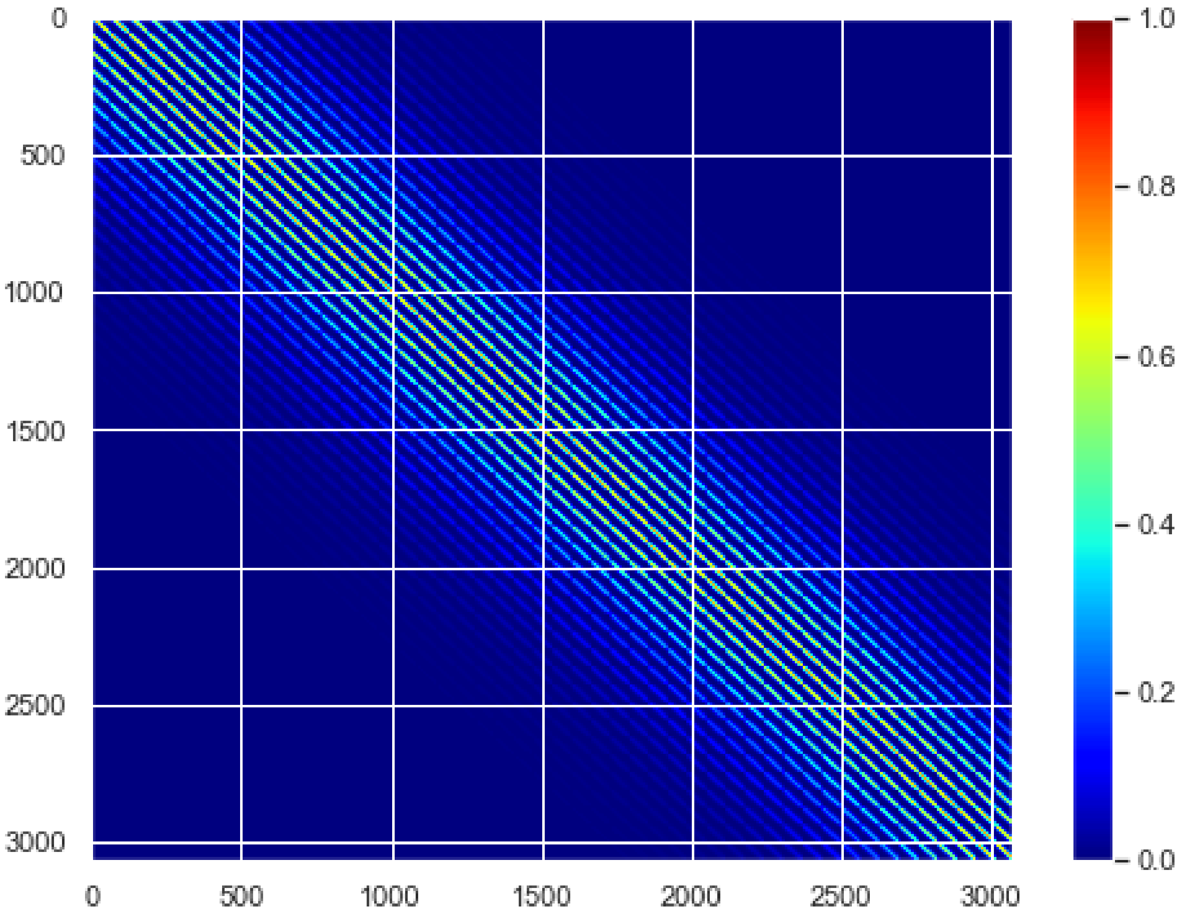


Fig. 6.5 The covariance between all points in our spatial domain $\mathcal{S}$ with the Matérn Three Halves covariance function with $\rho = 1.5$ and $\lambda = 1$. Each index on the axis corresponds to a point $\boldsymbol{s}_i$ in $\mathcal{S}$ in our discretised grid.

$$a_k\left(\boldsymbol{s}_i, \boldsymbol{s}_j\right) = \lambda_k \left(1 + \frac{\sqrt{3}d}{\rho}\right) \exp\left(\frac{-\sqrt{3}d}{\rho}\right),$$

where again d is the Euclidean distance between $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ given by $d = \|\boldsymbol{s}_i - \boldsymbol{s}_j\|$. Again, this is a stationary kernel, that is the covariance between two points depends only on the spatial distance between points and not their location. The length scale parameter, $\rho \in \mathbb{R}^+$, is used to capture the correlation over differing spatial scales. In the CPACE model we treat it as a hyperparameter to estimate. Figure 6.5 shows an example covariance structure over our spatial domain $\mathcal{S}$ for the simulation study with $\rho = 1.5$ and $\lambda = 1$. We note that comparing this to Figure 6.4, the covariance model is naturally smoother than the Matérn One Half kernel and, in some sense, is designed to model smoother variation over the domain.

Again, the use of this model is akin to the SPACE model, [48]. However in the CPACE model the estimation procedure for hyperparameters is different.

Gibbs Kernel

All the above kernels are stationary. Stationary kernels have been well studied due to the relative ease of estimation of kernel hyperparameters. However, they are often too simplistic, as many real world datasets exhibit some form of non-stationarity. We consider the Gibbs kernel, named after the Gibbs who authored the thesis which first described this kernel, [23], in our simulation study as a non-stationary kernel which can account for more complex spatial dependence. The form of the kernel is given succinctly by Paciorek and Schervish, [62], for one dimensional input.

$$a_k\left(s_i, s_j\right) = \lambda_k \sum_{q=1}^{Q} \sqrt{\frac{2l_q(s_i)l_q(s_j)}{l_q(s_i)^2 + l_q(s_j)^2}} \exp\left(-\frac{\left(s_i - s_j\right)^2}{l_q(s_i)^2 + l_q(s_j)^2}\right).$$

This can be simply extended to two, as in our simulation case, or more dimensions by having a Gibbs kernel on each dimension of the input and combining by simply multiplying them together. This remains a valid positive definite kernel as the product of two or more positive definite kernels remains positive definite. In this Gibbs kernel we have multiple components, referenced by Q total components. Each component has its separate length scale model $l_q(s)$ which controls the length scale parameters at each point in the domain. In principal these length scale models can be arbitrary positive functions, and the Gibbs kernel remains valid.

Figure 6.6 gives an example covariance function from the Gibbs kernel. As can be seen, the covariance structure changes over the domain, highlighting the non-stationarity of the Gibbs model. This can allow for much more complex spatial structures. Here we have restricted the kernel to two components, and use as in Scenario D, the lengthscale model given by:

$$\begin{aligned} l_1(s) &= \frac{1}{1+\exp(-s)}, \\ l_2(s) &= \frac{1}{1+\exp(s)}. \end{aligned}$$

For example, locations in the middle of the domain $\mathcal{S}$, are less correlated to immediate neighbours than the locations on the fringes of the domain. This has been induced by the choice of the length scale models above. In our simulation study, we treat these as hyperparameters to be estimated from the observed data. The way in which we do so will be discussed in greater detail in Section 6.5 and in Chapter 8.

Now we discuss the specific results of using the various kernels in our simulation studies.

6.2 Scenario A - Independent Functional Data

Our first simulation scenario considers the basic case when there is no spatial dependence between functional data. This corresponds to the case that for each component we simulate

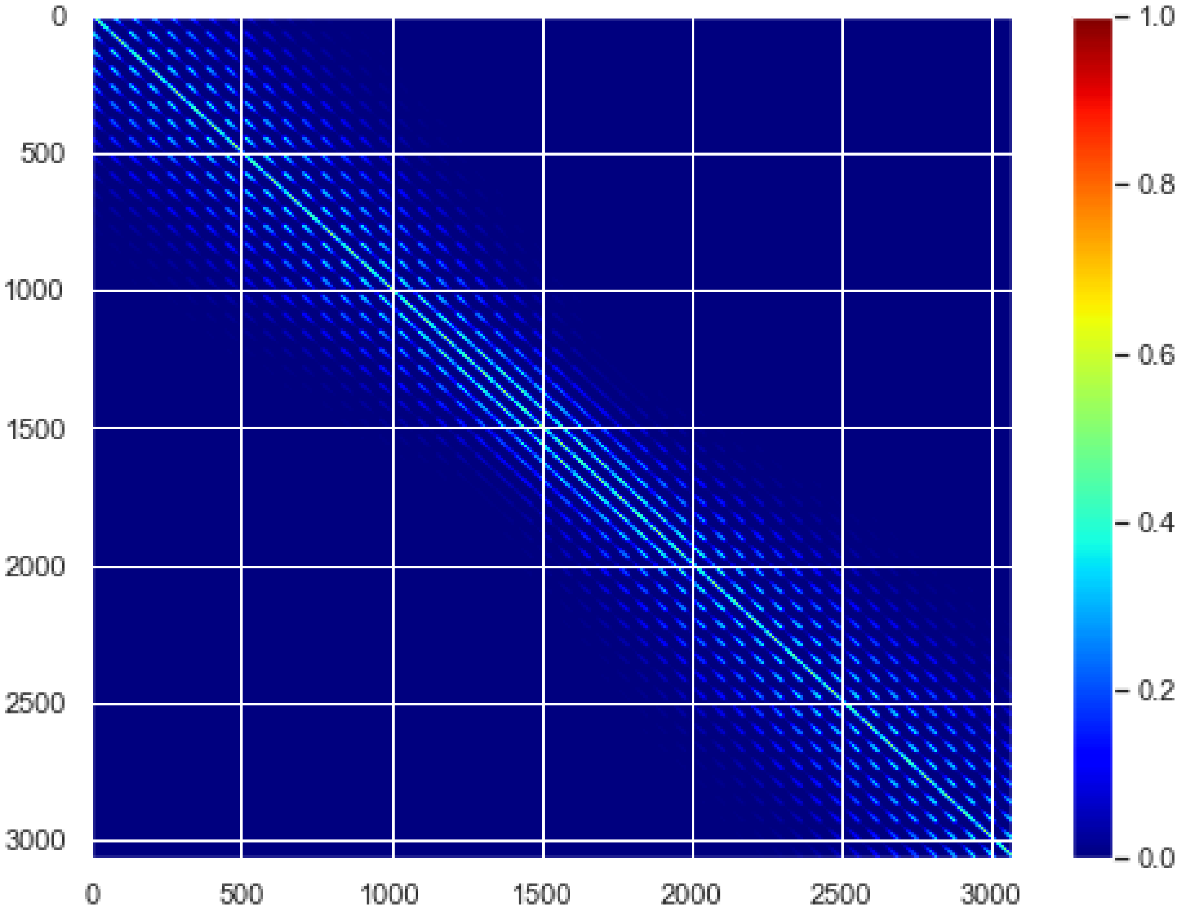


Fig. 6.6 The covariance between all points in our spatial domain $\mathcal{S}$ with the above Gibbs covariance function. Each index on the axis corresponds to a point $\boldsymbol{s}_i$ in $\mathcal{S}$ in our discretised grid.

data using a the White kernel, as described above. As stated in Section 6.1.1, we consider 50 replications using the White kernel with the aforementioned eigenfunctions with respective variance of 4 and 1. We consider 5 separate models to compare in our simulation study. The first is the standard PACE model as implemented in [90]. This is our base model, and is referred to as `fpca` in the following tables and graphs. The second model we consider is our CPACE model, using 5 components, each with an independent White kernel. This model is estimated under our framework of the CPACE model, and as such the kernel variances are initially estimated using the PACE methodology but additional refinement of the estimates is done through the CPACE framework. This will be referred to as the `fpca_gp` model in the following metrics and graphs. The third and fourth models are similar. They correspond to the use of the Matérn One Half and Matérn Three Halves kernels as spatial kernels in the CPACE model respectively. Again, we use the CPACE framework to refine the noise and variance estimation, but in addition use the CPACE framework to estimate each kernel‘s hyperparameter, ρ_k. We refer to these models as `matern_one` and `matern_three` respectively. Finally, the last model corresponds to using the Gibbs kernel for each component‘s spatial kernel in the CPACE model. We use $Q = 5$ components for each Gibbs kernel. The Gibbs kernel has length scale models. For our simulation study we assume we can approximate the true length scale functions using a neural network. This allows the model to be flexible to capture many different functions without having to be too specific in the setup of the model. We use a multi-layer perceptron network with two layers, [27]. Each length scale model has the same architecture of two fully connected hidden dense layers with 32 neurons in each layer. We use an rectified linear unit activation for the hidden layers, and a final activation which maps the length scale produced between 0.0001 and 1. This is to ensure the length scales remain positive and bounded within a sensible value. Under the CPACE model framework the parameters to each length scale model are estimated, along with the standard variance and noise variance estimation. We refer to this model as `gibbs` in the following.

Running 50 simulations, we first display, in Table 6.1 the training metrics of the models, both the MSE and the MAE, with their respective variance for the training observations from each simulation. We note here that, expectedly, the models which assume no spatial dependency do best on the training data. Quite simply, this is because the model is closest to the data generating procedure. However, it is interesting to note that on observed training data all the models results with the CPACE framework are on a similar order of magnitude as that of the PACE model; albeit that the `gibbs` model performs slightly worse. This is encouraging as it suggests that we don’t lose anything by considering a more complex model, because the CPACE framework can adapt to independently observed data, as the kernel parameters are estimated such that the kernel becomes close to the White kernel.

Next, we consider the metrics for reconstruction of the functional data across all observed locations. This is akin to the combination of the validation and the training datasets as described above. Table 6.2 shows the metrics on these data points.

Table 6.1 Simulation results for spatially independent data (Scenario A) for the model's ability to estimate the functional values at points of observation. Bold indicates best in class.

Model	**MSE**	**MAE**
`pace`	0.0502 (0.0015)	0.1771 (0.0026)
`fpca_gp`	**0.0499 (0.0014)**	**0.1766 (0.0025)**
`matern_one`	0.0508 (0.0014)	0.1782 (0.0025)
`matern_three`	0.0523 (0.0042)	0.1805 (0.0069)
`gibbs`	0.0685 (0.0042)	0.2052 (0.0115)

Table 6.2 Simulation results for spatially independent data (Scenario A) for the model's ability to estimate the functional data at locations of observation across the whole temporal domain. Bold indicates best in class.

Model	**MSE**	**MAE**
`pace`	0.5641 (0.0168)	1.8707 (0.0279)
`fpca_gp`	**0.5609 (0.0158)**	**1.8659 (0.0269)**
`matern_one`	0.5722 (0.0163)	1.8845 (0.0264)
`matern_three`	0.5901 (0.0482)	1.9109 (0.0744)
`gibbs`	0.7782 (0.1067)	2.1510 (0.1224)

We can see clearly the same pattern as from the training metrics. Again, the `fpca_gp` model is our best in terms of reconstruction ability. Figure 6.7 highlights a single example of such reconstruction. It can be seen that not only does the `fpca_gp` model capture the full unobserved functional data well, it does so with confidence using the variance that follows from the CPACE framework. It is easy to see that the `gibbs` model fails to capture as completely the functional data, possibly due to error in estimation of the length scale models. This may be because we have assumed a complex form of the length scale models in the `gibbs` model and the estimation procedure for them has failed to converge appropriately to the simple form they take in this scenario.

Finally, another advantage of the CPACE model is the ability to refine estimation of the noise variance and the component kernel variances, σ_ε^2 and λ_k respectively. While this, as shown in Table 6.2, may not gain much in reconstruction power, it does indicate the CPACE framework‘s ability to capture the true parameters of the model better. Figure 6.8 shows the distribution of the estimated noise variance over all our models. This distribution was estimated by smoothing the empirical distribution of the noise estimate from our 50 replications of the simulation study. Notice that the CPACE model's tend to produce more consistent estimates than the standard PACE model. This is because the CPACE framework allows for refinement of these estimates using the PACE estimate as an initial value through the Gaussian process likelihood of observed data. Similar results are seen in the eigenvalue estimates (kernel variances). Figure 6.9a and Figure 6.9b show this for the first and second eigenvalues respectively. This is wanted and useful behaviour for modelling, as it provides assurance that the CPACE model can improve on the initial

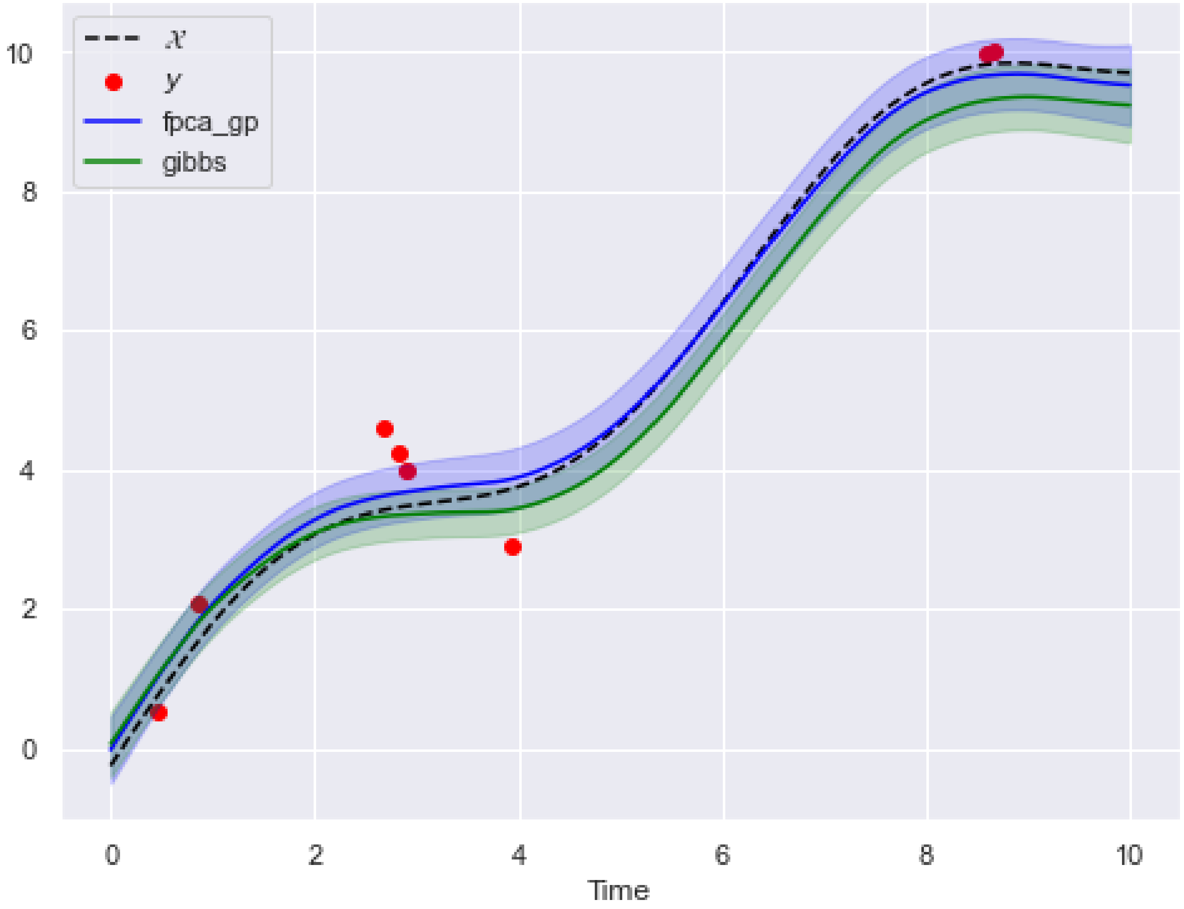


Fig. 6.7 An indicative example of the CPACE model performance on reconstruction of functional data where noisy observations have been made for Scenario A. The shaded regions represent our confidence interval of prediction and corresponds to two standard deviations from our mean predictions.

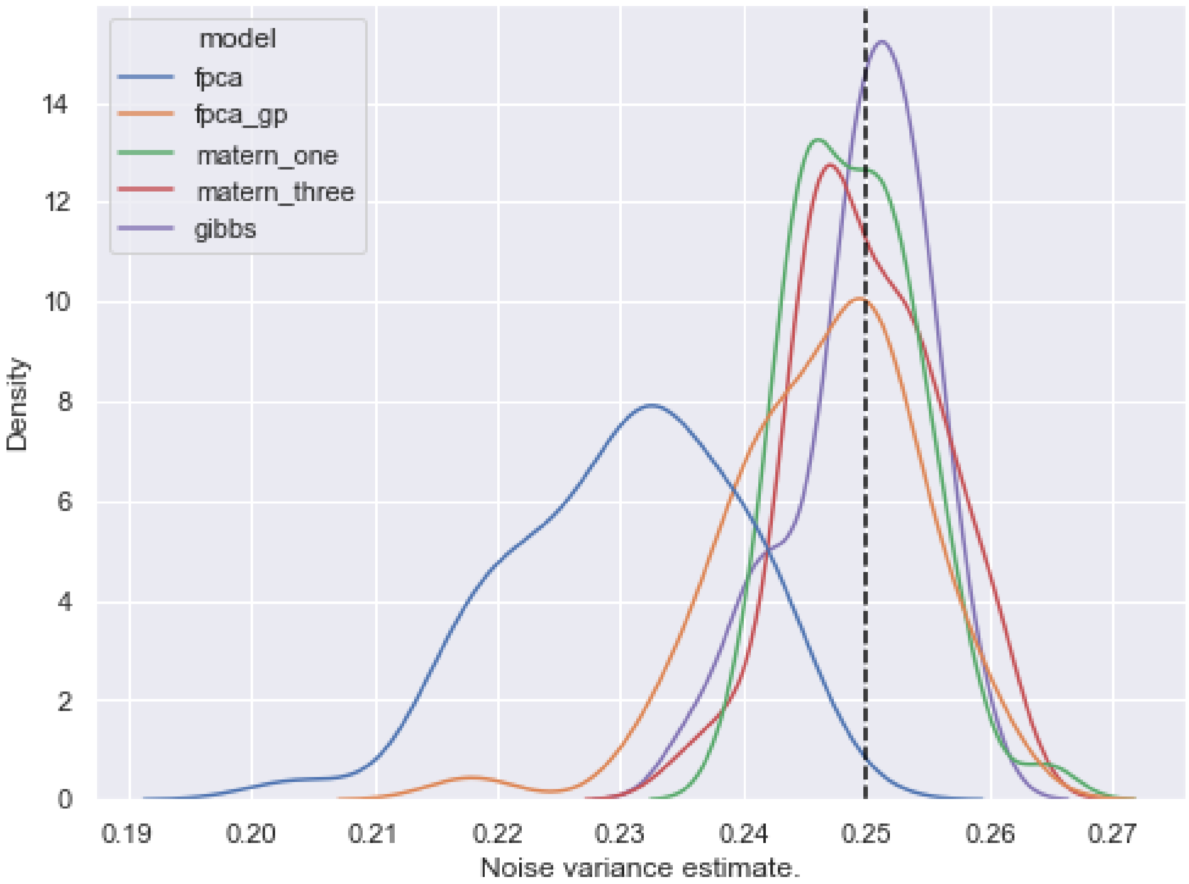
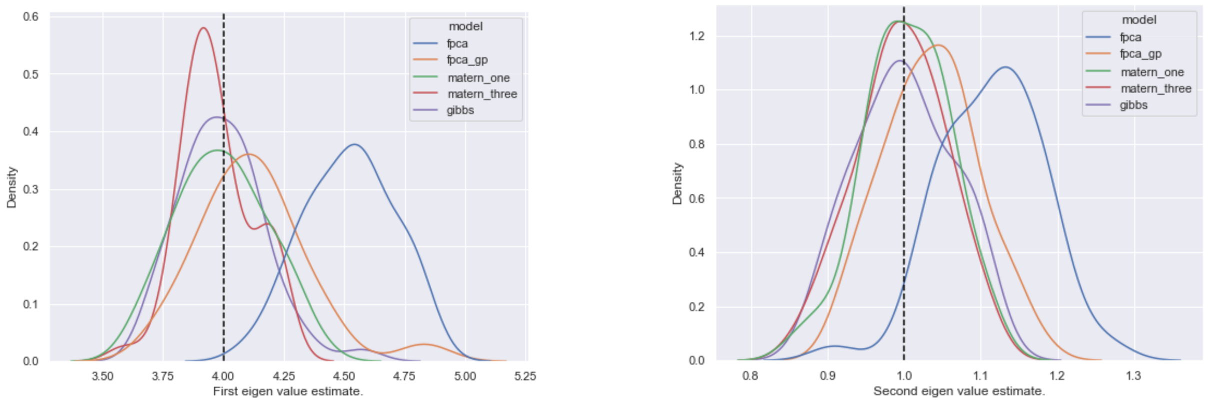


Fig. 6.8 Estimated noise variance distribution for each of our models over the 50 simulations for Scenario A. The true noise variance is given by the vertical black line for indication.

PACE estimates for both the noise parameter and variance parameters. This should lead to improved understanding of the possible data generation procedure when using these models on real world datasets.

Next we consider the test results for this scenario study. As described in Section 6.1.2 the test results are metrics for the reconstruction ability on the test dataset. This is the set of locations in the domain $\mathcal{S}$ in which functional data are simulated but we assume we have no observations at these locations as all. Table 6.3 displays the MSE and MAE results for all our models. Firstly, it is worth noting that the metric results across all models for the test dataset are worse than that of the training and validation datasets, see Table 6.1 and Table 6.2 respectively. This is understandable, due to the nature of this scenario, since we are considering spatially independent functional data. As expected both the `fpca` and `fpca_gp` model produce similar results which are the best possible under our simulation study models. They are similar as both models will predict the mean function at unobserved locations, due to the structure of the White kernel upon which they depend. It is expected that these models will produce the best in class results as they most closely represent the simulated data generation. Again, it is however worth noting that the other models have generalised well by estimation of their appropriate length scales. This bodes well, as it supports the use of these models in the case where the model may have small spatial structure and it is unknown as to whether to use a spatial



(a) Estimated first eigenvalue (first kernel variance) for each of our models over the 50 simulations for Scenario A. The true parameter is given by the vertical black line for indication.

(b) Estimated second eigenvalue (second kernel variance) for each of our models over the 50 simulations for Scenario A. The true parameter is given by the vertical black line for indication.

Fig. 6.9 Distributions of the estimated first and second eigenvalues for each model in Scenario A.

Table 6.3 Simulation results for Scenario A on the model's ability to estimate the functional data at locations with no observation across the whole temporal domain. Bold indicates best in class.

Model	MSE	MAE
pace	**5.0105 (0.1234)**	**5.5143 (0.0645)**
fpca_gp	**5.0105 (0.1234)**	**5.5143 (0.0645)**
matern_one	5.3783 (0.2622)	5.7094 (0.1298)
matern_three	5.5190 (0.4540)	5.7824 (0.2262)
gibbs	6.6118 (0.4765)	6.2981 (0.1987)

model or not. These simulation results suggest that you are not hindered by choosing a more complex kernel which may take into account spatial dependency; since the CPACE model will approximate spatial independence by adjusting the kernel hyperparameters appropriately. However, there are limitations. The comparatively poor performance of the gibbs model may be because the complexity of this model has lead to non-convergence of the length scale models, leading to spurious spatial dependence. This would then lead to less accurate forecasts and give higher test metrics. We discuss the implementation we have used to minimise this in Chapter 8. This issue wasn't present CPACE models using the Matérn based kernels.

6.3 Scenario B - Stationary Functional Data

The next scenario we consider under our simulation study is the case of simulated data which exhibits spatial dependence. We assume for Scenario B that we have stationary dependence. We induce this by considering that each component spatial covariance is given by the Matérn One Half covariance model. For both components we assume the length

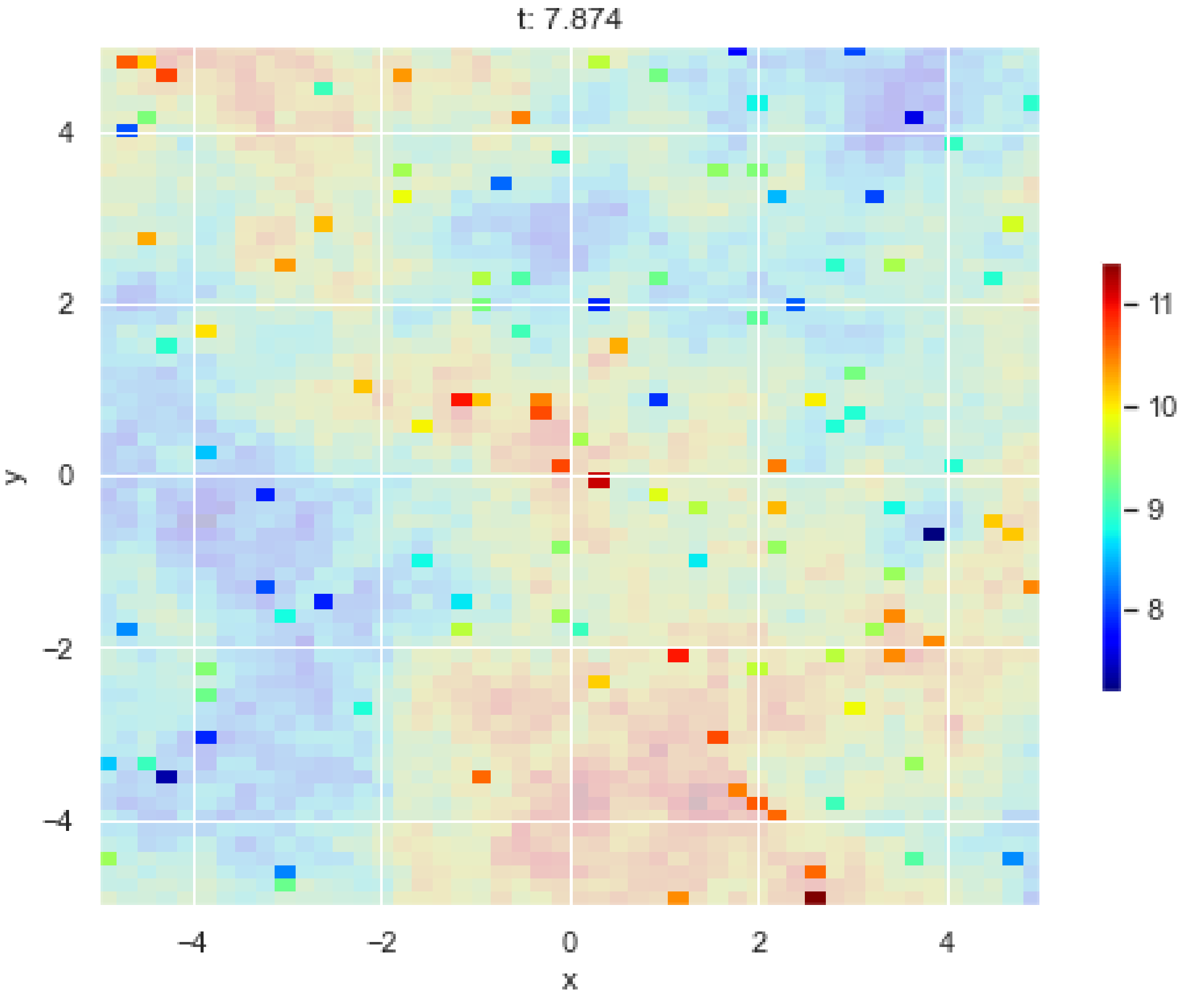


Fig. 6.10 Example realisation of the spatial dependence under Scenario B at a particular time point in domain $\mathcal{T}$. Here we show the full unobserved data across $\mathcal{S}$ with observed data being given by the less transparent pixels in the image.

scale for the Matérn One Half covariance is 2. In essence, we are considering here a simple case of spatial dependence, as both components have the same spatial covariance structure. Figure 6.10 gives an example of what the full simulated data looks like over space. This is for illustration only, to give the reader an idea of the scale of the spatial dependence induced. This corresponds to only one realisation of the data generating procedure for Scenario B at a particular time point.

Again, we compare the same models as in Scenario A, see Section 6.2 for a full description. Table 6.4 reports the metric results for these models on Scenario B observed data. As expected our Matérn One Half model performs best on the training data. Again, this is most probably because it closely resembles the data generating procedure for this scenario. It is also important to note the significant improvement any spatial dependence in the model kernel makes for reconstruction on the training data. Both the Matérn One Half and Matérn Three Halves models show considerable improvements, but so does the Gibbs model, although improvements are noticeably smaller.

Table 6.4 Simulation results for Scenario B for the model‘s ability to estimate the functional data at points of observation. Bold indicates best in class.

Model	MSE	MAE
`pace`	0.0483 (0.0016)	0.1738 (0.0029)
`fpca_gp`	0.0482 (0.0016)	0.1736 (0.0029)
`matern_one`	**0.0275 (0.0013)**	**0.1310 (0.0030)**
`matern_three`	0.0352 (0.0019)	0.1477 (0.0040)
`gibbs`	0.0425 (0.0038)	0.1613 (0.0071)

We note as in Scenario A we have refined estimates for the error variance and eigenvalues (kernel variances) than from the PACE model under the CPACE framework. For Scenario B these estimates are displayed in Figures 6.11, 6.12a, and 6.12b respectively. We see that the noise estimate is more precise for the CPACE models compared to the PACE model. However, for the eigenvalue estimate we see the PACE model actually delivers more consistent results. It is important to note that under spatial dependence it is possible for these to be biased away from the true values due to the correlation induced among functional observations.

In Scenario B we can also compare the length scale estimate from the Matérn based models to that of the scenario for each component. As we can see in Figure 6.13a and Figure 6.13b the Matérn One Half model manages to capture the true value in its distribution for each component, although it resides in the tail of the distribution. This may indicate that it hasn’t quite captured the data generating procedure entirely. However, coupled with the underestimate of the kernel variance, an underestimate of the kernel length scale does make sense. The Matérn Three Halves model fails to capture the true length scale parameter in its model. Again, as this model is inherently smoother over larger distances it makes sense that the length scale would be smaller when estimated on this scenario to account for this.

We now consider the test metrics for these models. Table 6.5 showcases these results. As expected these results highlight that the Matérn One Half model succeeds in being the most accurate of the models for reconstructing completely unobserved functional data. As can also be seen, the Matérn Three Halves model comes surprisingly close to recreating the simulated data. This is again showcasing the ability of the CPACE framework to tailor the hyperparameters of the chosen kernel to suitably capture the data. In this case we have seen, from Figure 6.13a, that this corresponds to estimating the shorter length scale hyperparameter to compensate for the smoother kernel function. However, it does not quite compensate for having the correct kernel for the data generating procedure, which is why the Matérn One Half kernel obtains the best metrics. As the standard deviation of the Matérn One Half and Matérn Three Halves metrics indicate that they may well in fact be equally as powerful, we use the distribution of metric results to highlight the Matérn One Half model to be best in class. Figure 6.14 highlights the distribution of metrics over the full 50 simulations between these two models. In this type of plot the smoothed empirical distribution of the metric is represented by the shape with the mean, interquartile range,

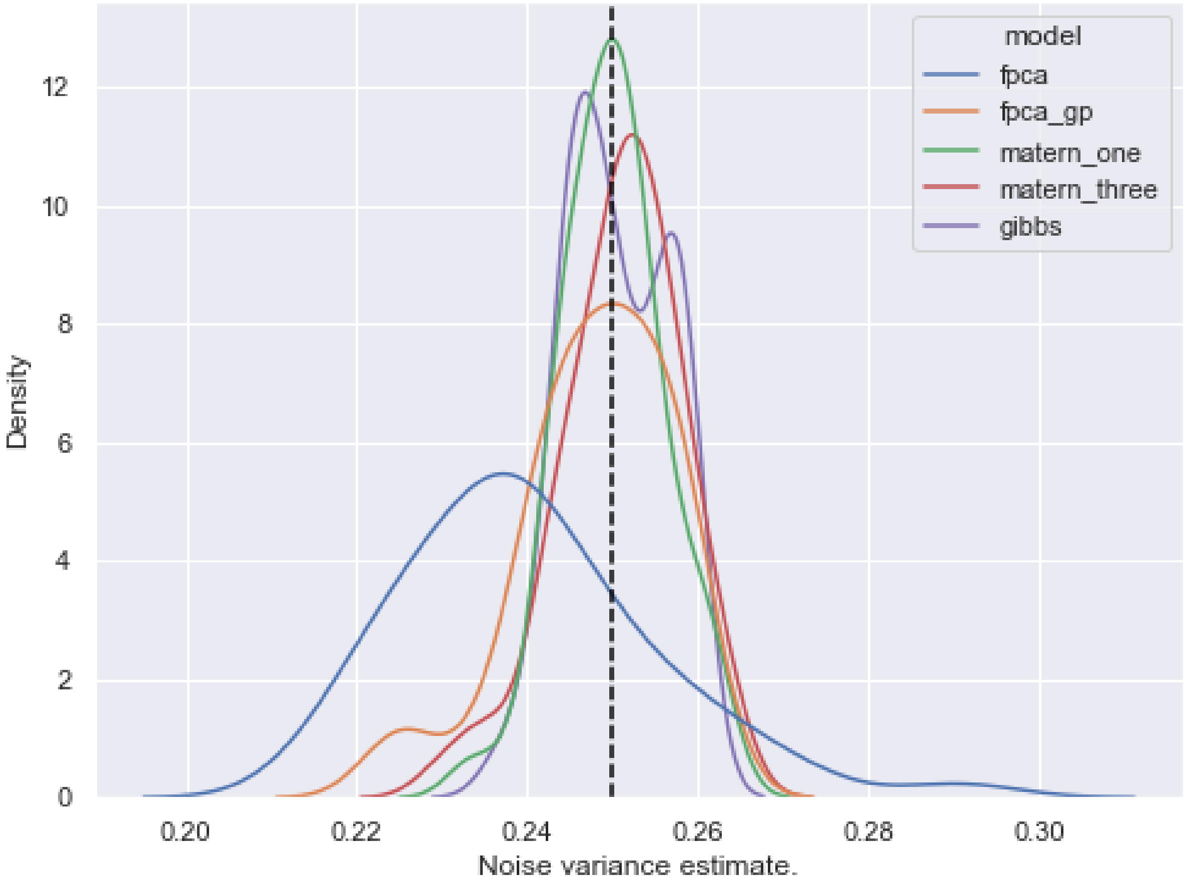
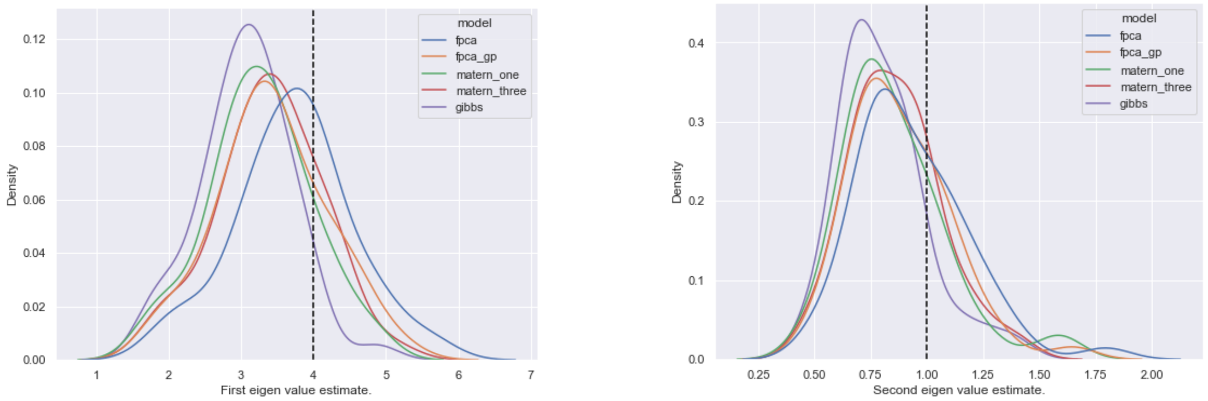


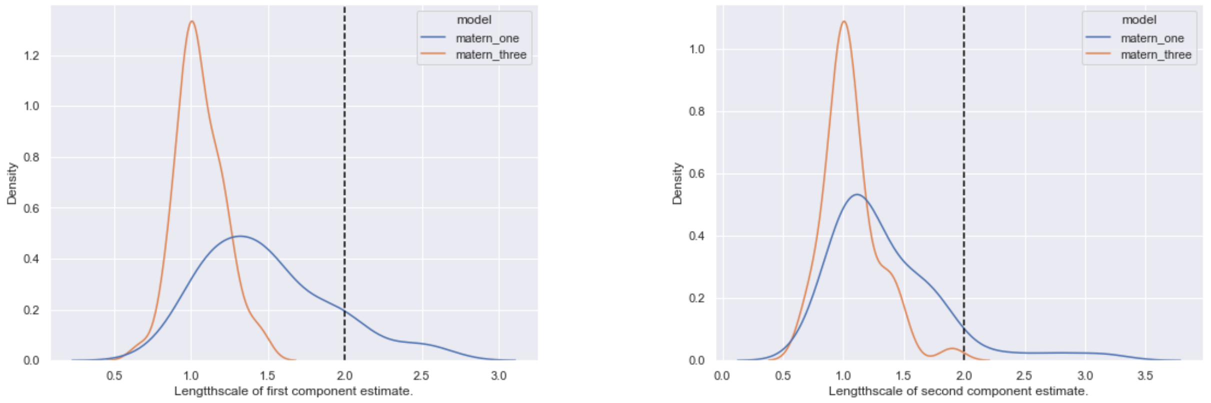
Fig. 6.11 Estimated noise variance distribution for each of our models over the 50 simulations for Scenario B. The true noise variance is given by the vertical black line for indication.



(a) Estimated first eigenvalue (first kernel variance) for each of our models over the 50 simulations for Scenario B. The true parameter is given by the vertical black line for indication.

(b) Estimated second eigenvalue (second kernel variance) for each of our models over the 50 simulations for Scenario B. The true parameter is given by the vertical black line for indication.

Fig. 6.12 Distributions of the estimated first and second eigenvalues for each model in Scenario B.



(a) Estimated first kernel length scale parameter for the Matérn models over the 50 simulations for Scenario B. The true parameter is given by the vertical black line for indication.

(b) Estimated second kernel length scale parameter for the Matérn models over the 50 simulations for Scenario B. The true parameter is given by the vertical black line for indication.

Fig. 6.13 Distributions of the estimated first and second length scale parameters for each model in Scenario B.

Table 6.5 Simulation results for Scenario B for the model‘s ability to estimate the functional data at locations with no observation across the whole temporal domain. Bold indicates best in class.

Model	**MSE**	**MAE**
`pace`	4.1851 (0.7638)	5.0613 (0.4810)
`fpca_gp`	4.1851 (0.7638)	5.0613 (0.4810)
`matern_one`	**0.5848 (0.0215)**	**1.8926 (0.0333)**
`matern_three`	0.5967 (0.0212)	1.9100 (0.0323)
`gibbs`	0.7413 (0.0505)	2.1187 (0.0666)

and range being given by the bar inside the shape. We can see a slightly more positive skew for the distribution of both MAE and MSE for the Matérn One Half model, again indicating that this model is more consistent in reconstruction that the Matérn Three Halves model. Hence we conclude that the Matérn One Half model constitutes our best in class on the test dataset for Scenario B.

We can see quite clearly that the PACE and CPACE framework with the White kernel are not flexible enough to help on unobserved functional data, from Table 6.5. This is due to the White kernel, as it does not take into account any spatial dependency, meaning they only propose the mean for all unobserved data. While this was advantageous in Scenario A, under spatial correlation between functional data this becomes a hindrance.

Finally, to highlight the abilities of the CPACE models, we display an example prediction of a test data location in Figure 6.15. Here we clearly see the reconstruction ability of the CPACE framework and the higher confidence in reconstruction that the CPACE framework can bring. This is applicable to both the Matérn One Half and Matérn Three Halves models. We can also see the advantage of the Matérn One Half model. As although the confidence band is much narrower than the PACE model under the CPACE

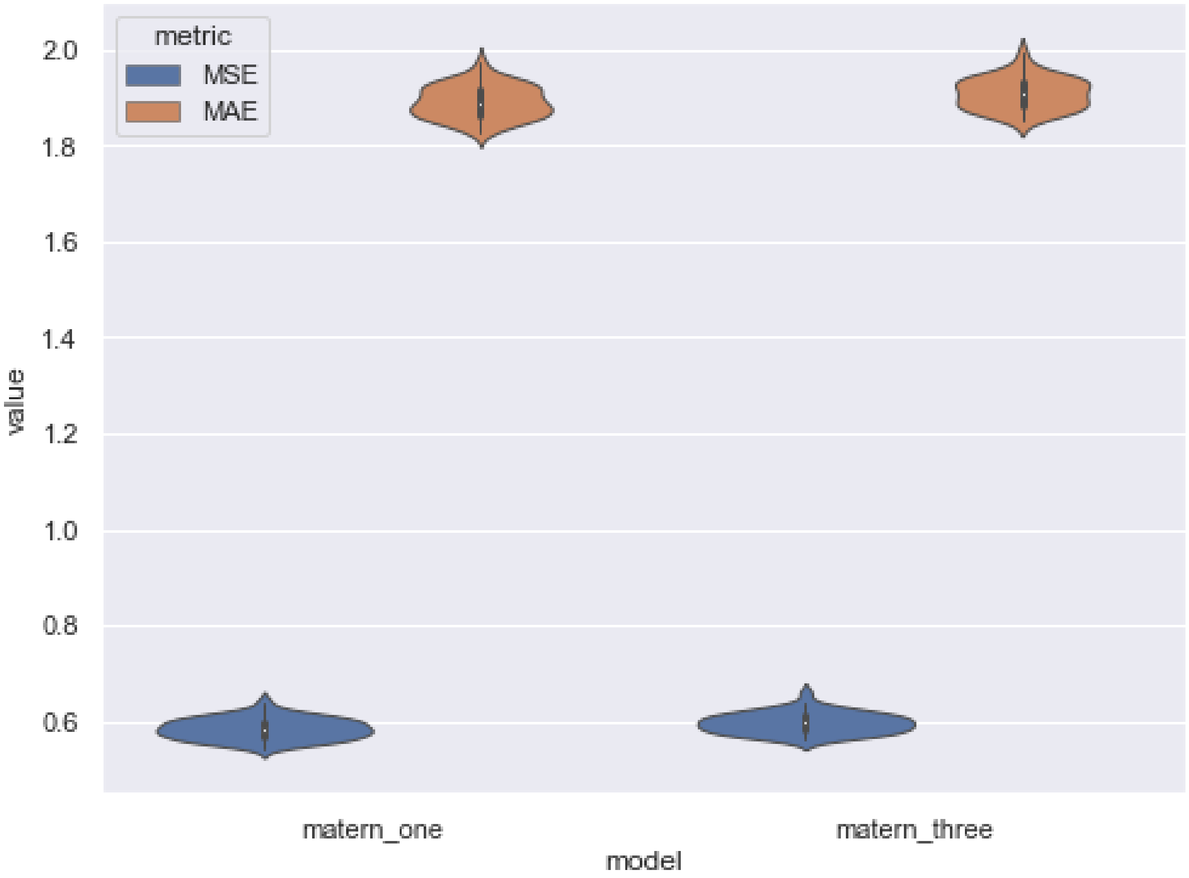


Fig. 6.14 Distribution of the MSE and MAE metrics over the 50 simulations for the test dataset of Scenario B. Here the shape represents the smoothed empirical distribution of the metric while the summary statistics of the mean, interquartile range, and range are given from the bars inside the shape.

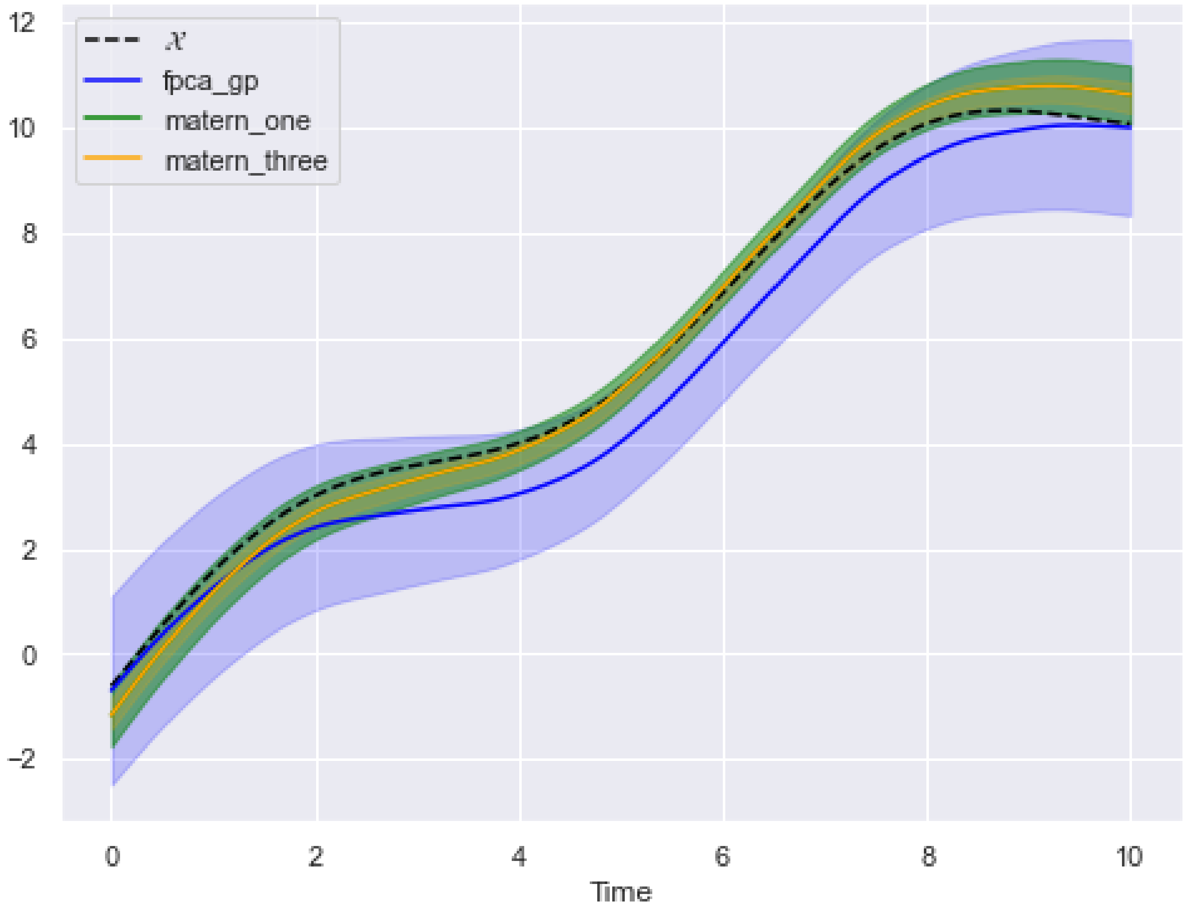


Fig. 6.15 An indicative example of the CPACE model performance on reconstruction of functional data for Scenario B. Example is taken from a test data point, so no observations were observed at this spatial location. The confidence bands correspond to a 95% simultaneous confidence interval for the predicted location.

framework, it is not as narrow as the Matérn Three Halves model. This is advantageous as can be seen in Figure 6.15. The actual functional data sometimes falls outside the Matérn Three Halves confidence band but still resides in the Matérn One Half model. This can be seen as an indicative example only, but may suggest the Matérn Three Halves model may be over confident in its predictions on the test data. This stems from the fact that the kernel itself is too smooth for the data generating procedure. The Matérn One Half model has no such issues.

From the above we can see an distinct advantage for using the CPACE framework when there is indeed simple spatial dependence, however this is as expected, and has been discussed in work by Liu et al., [48]. In the following sections we consider the more challenging case when the spatial dependence is more complex.

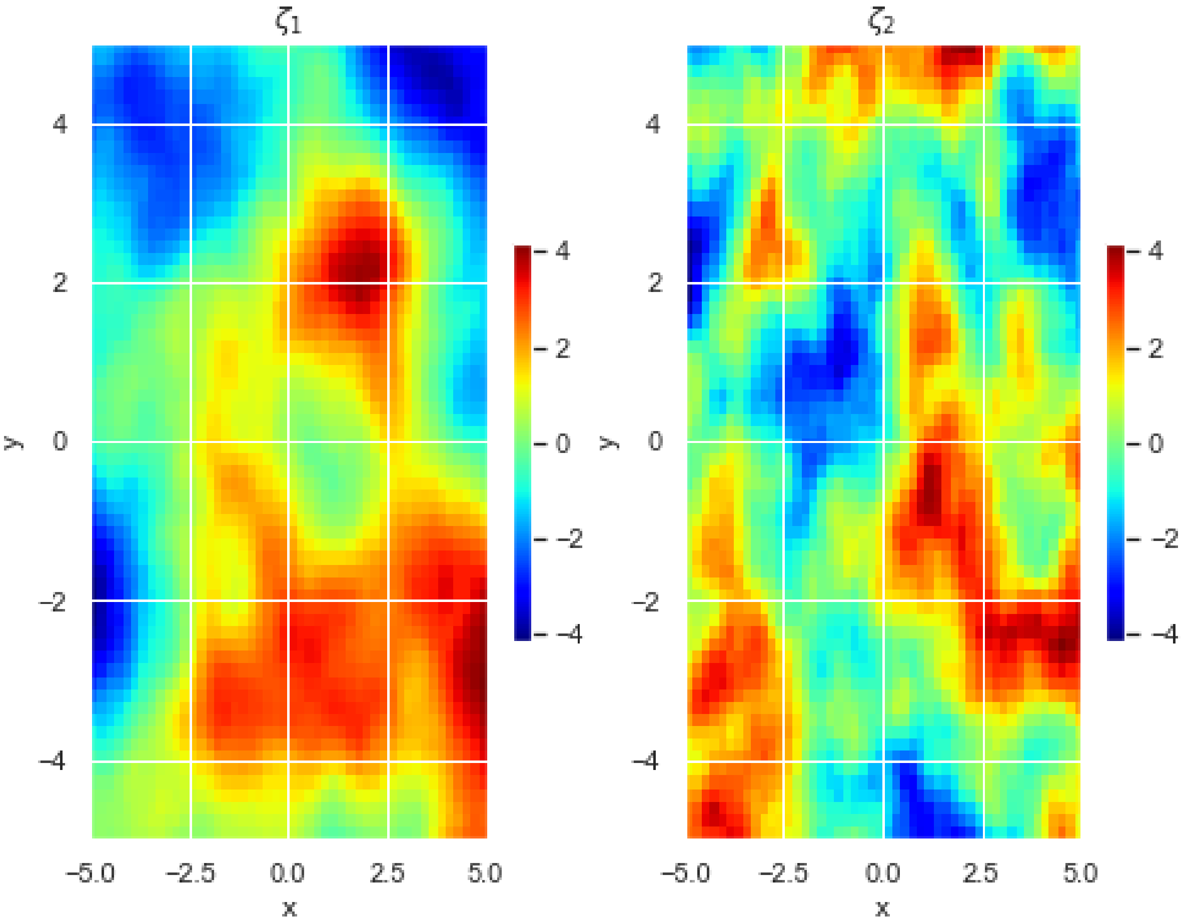


Fig. 6.16 An indicative example of a realisation from the component spatial kernels in Scenario C.

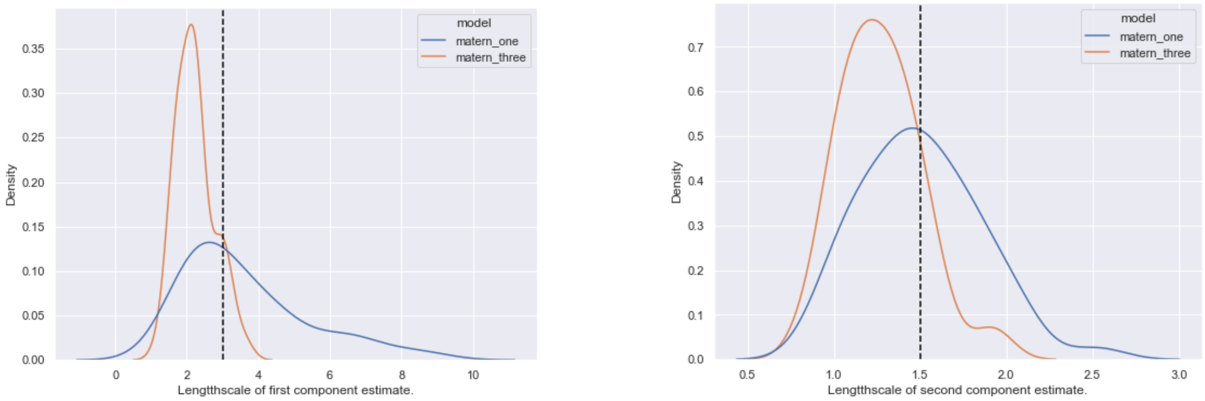
6.4 Scenario C - Complex Stationary Functional Data

In this scenario we consider modelling simulations which are generated with a more challenging spatial dependence. In particular, in this scenario we generate functional data as mentioned in Section 6.1.1 but this time we place different spatial kernels on each component of the model. We use the Matérn Three Halves kernel for both components but choose a length scale of 3 for the first component and a length scale parameter of 1.5 for the second component. In essence, we are now considering the case where there are two distinct covariance structures between the spatial components of the models. Previous scenarios had the same spatial covariance for both components. This scenario is designed as a test for the CPACE framework, to see if it can accommodate this added complexity. Figure 6.16 highlights an example of a realisation of each of these covariance components. This is intended to highlight the spatial dependency present in such a simulation from each of the components.

As in previous scenarios we first consider the resultant metrics from the training data. The results are displayed in Table 6.6. As with the other scenarios, all models provided reasonable reconstruction on observed data. This includes the PACE framework.

Table 6.6 Simulation results for Scenario C for the models ability to estimate the functional data at points of observation. Bold indicates best in class.

Model	MSE	MAE
`pace`	0.0479 (0.0021)	0.1733 (0.0038)
`fpca_gp`	0.0479 (0.0022)	0.1731 (0.0039)
`matern_one`	0.0132 (0.0013)	0.0909 (0.0044)
`matern_three`	**0.0085 (0.0009)**	**0.0730 (0.0038)**
`gibbs`	0.0094 (0.0009)	0.0767 (0.0036)



(a) Estimated first kernel length scale parameter for the Matérn models over the 50 simulations for Scenario C. The true parameter is given by the vertical black line for indication.

(b) Estimated second kernel length scale parameter for the Matérn models over the 50 simulations for Scenario C. The true parameter is given by the vertical black line for indication.

Fig. 6.17 Distributions of the estimated first and second length scale parameters for each model in Scenario C.

However, we can see a clear improvement in the metrics for the CPACE framework on the training data. This is indicative that the CPACE models, through using the learned spatial dependence, can use this extra information to gain an advantage in reconstructing the functional data from points of observation. Again, as expected the model which performs best is the `matern_three` model as it closely relates to the data generating process. We can see this from Figure 6.17a and Figure 6.17b. These highlight that the `matern_three` model captures, although not perfectly, the true length scale parameters of the data generating process. Similarly, the `matern_one` also captures the true length scale parameter, albeit with less certainty. This is another indication that the CPACE framework for learning hyperparameters of the spatial kernels works well. This will be discussed further in Chapter 8.

We can see an indicative example for reconstruction of the data using the best in class `matern_three` model and the `fpca_gp` model in Figure 6.18. This highlights the difference, not only in accuracy in reconstruction, but also the improved confidence of estimation of the best in class model. We note that in areas far away (in the temporal domain) from observation, the `fpca_gp` model tends to revert to the mean function, and its confidence interval expands. On the other hand the Matérn Three Halves model can

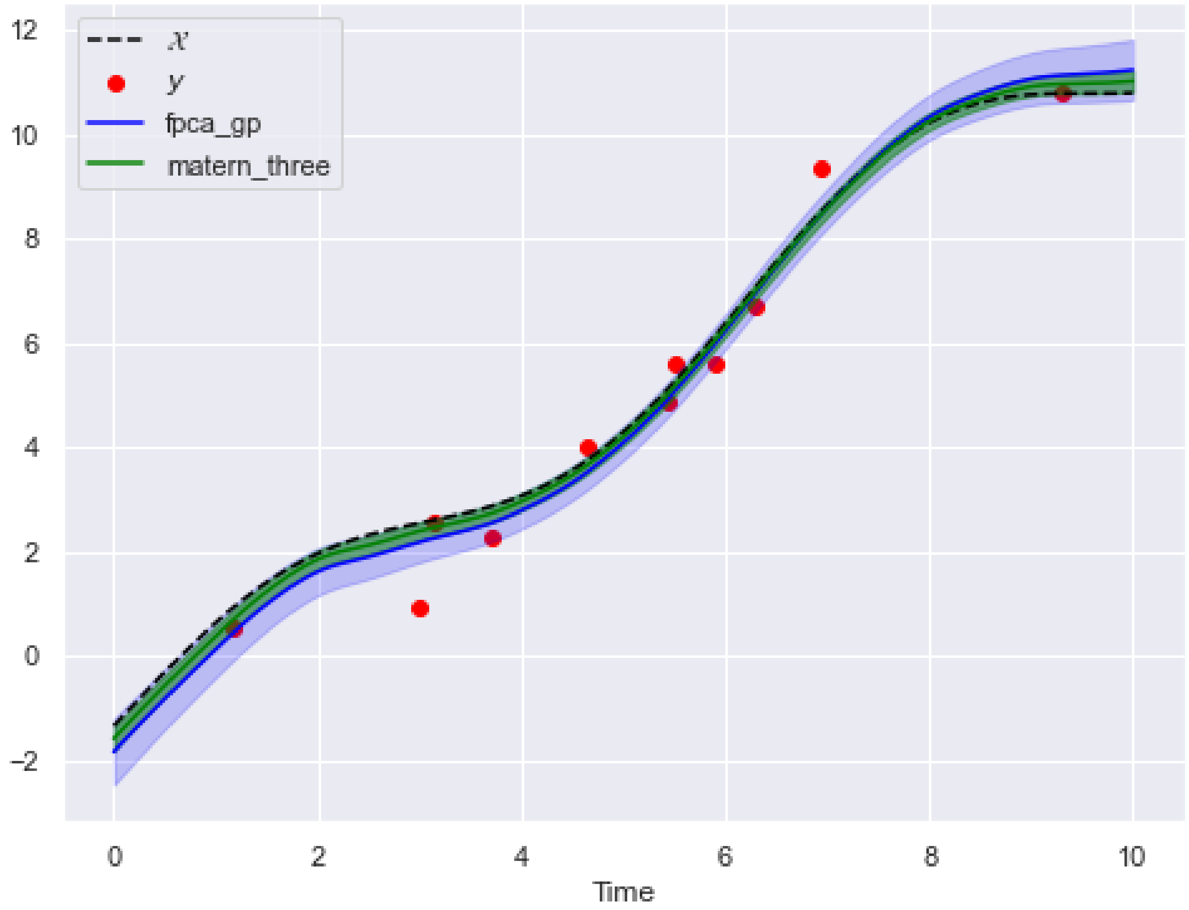


Fig. 6.18 An indicative example of model reconstructions for Scenario C for functional data with observed values. The shaded region is the prediction confidence band and corresponds to a 95% simultaneous confidence interval.

use information about observations from other spatial locations to improve its estimation for these areas.

Next, we consider the models comparative ability for reconstruction of completely unobserved functional data. Table 6.7 displays our models results on the test dataset for Scenario C. Here, we see similar results to the training metrics, where the CPACE framework outperforms the PACE models. It is interesting to note here that the `gibbs` model outperforms the `matern_one`, although very slightly. It again suggests that the CPACE framework is capable of flexibly estimating kernel hyperparameters, even when the parameter space is relatively large, as in the case of the Gibbs kernel. We note that we have highlighted the `matern_three` as best in class due to slightly larger positive skew in distribution of metrics over the test dataset versus the `gibbs` model. This is evidenced in Figure 6.19. In Figure 6.19 we compare only the `fpca_gp` and `matern_three`, this is to make it easier to see the comparative difference between best in class and the standard PACE model. We note that the `matern_one` and `gibbs` models produce similar reconstructions to that of `matern_three`.

Finally we give an indicative example of reconstruction using the best in class model and the `fpca_gp` model to highlight the improvement in reconstruction. This is given

Table 6.7 Simulation results for the models ability to estimate the functional data at locations with no observations across the whole temporal domain. Bold indicates best in class.

Model	MSE	MAE
pace	3.3175 (0.9034)	4.5480 (0.6234)
fpca_gp	3.3175 (0.9034)	4.5480 (0.6234)
matern_one	0.1109 (0.0112)	0.8343 (0.0410)
matern_three	**0.0957 (0.0104)**	**0.7745 (0.0412)**
gibbs	0.1053 (0.0106)	0.8117 (0.0400)

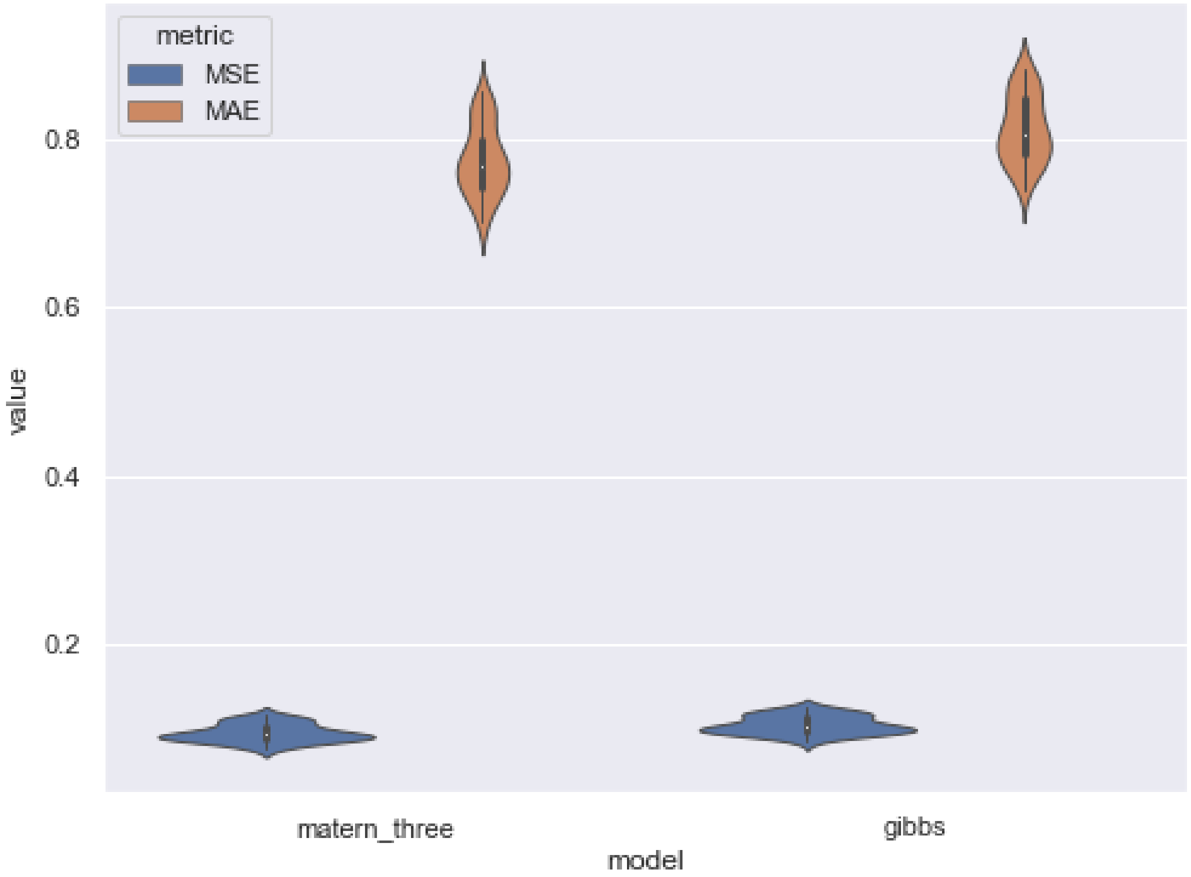


Fig. 6.19 Test metric distribution for Scenario C over 50 simulations for the Matérn Three Halves and Gibbs model. Here the shape represents the smoothed empirical distribution of the metric while the summary statistics of the mean, interquartile range, and range are given from the bars inside the shape.

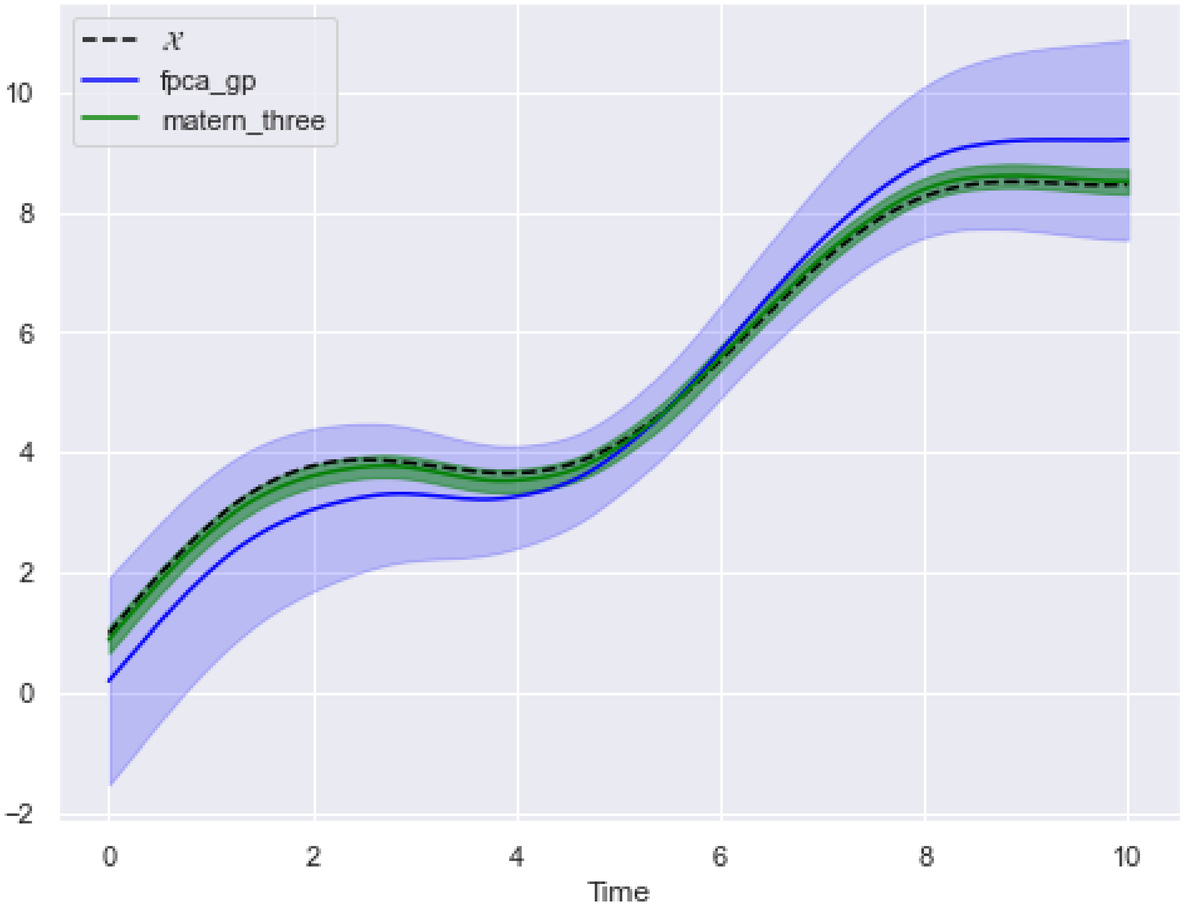


Fig. 6.20 An indicative example of model reconstructions for Scenario C for functional data with no observations. The shaded region is the prediction confidence band and corresponds to a 95% simultaneous confidence interval.

in Figure 6.20. We can clearly see the ability of the CPACE framework here to utilise the spatial dependency between observations, not only in reconstruction accuracy but in confidence of prediction. As can be seen, the CPACE framework has allowed for models which can handle quite complex spatial models. This has a real advantage in terms of both prediction accuracy and confidence in prediction on both unobserved locations and locations with observation points.

In the final scenario we consider one further step of complexity. That is, we examine the CPACE framework's ability to handle data which is simulated with a non-stationary spatial dependency.

6.5 Scenario D - Non-Stationary Functional Data

In the final scenario of the simulation study, we consider the case of data being generated which has non-stationary spatial dependency. We do so by simulating data for this scenario, following the data generation procedure given in Section 6.1.1, and use a Gibbs kernel for each component. We choose our length scale models for the scenario so that we create

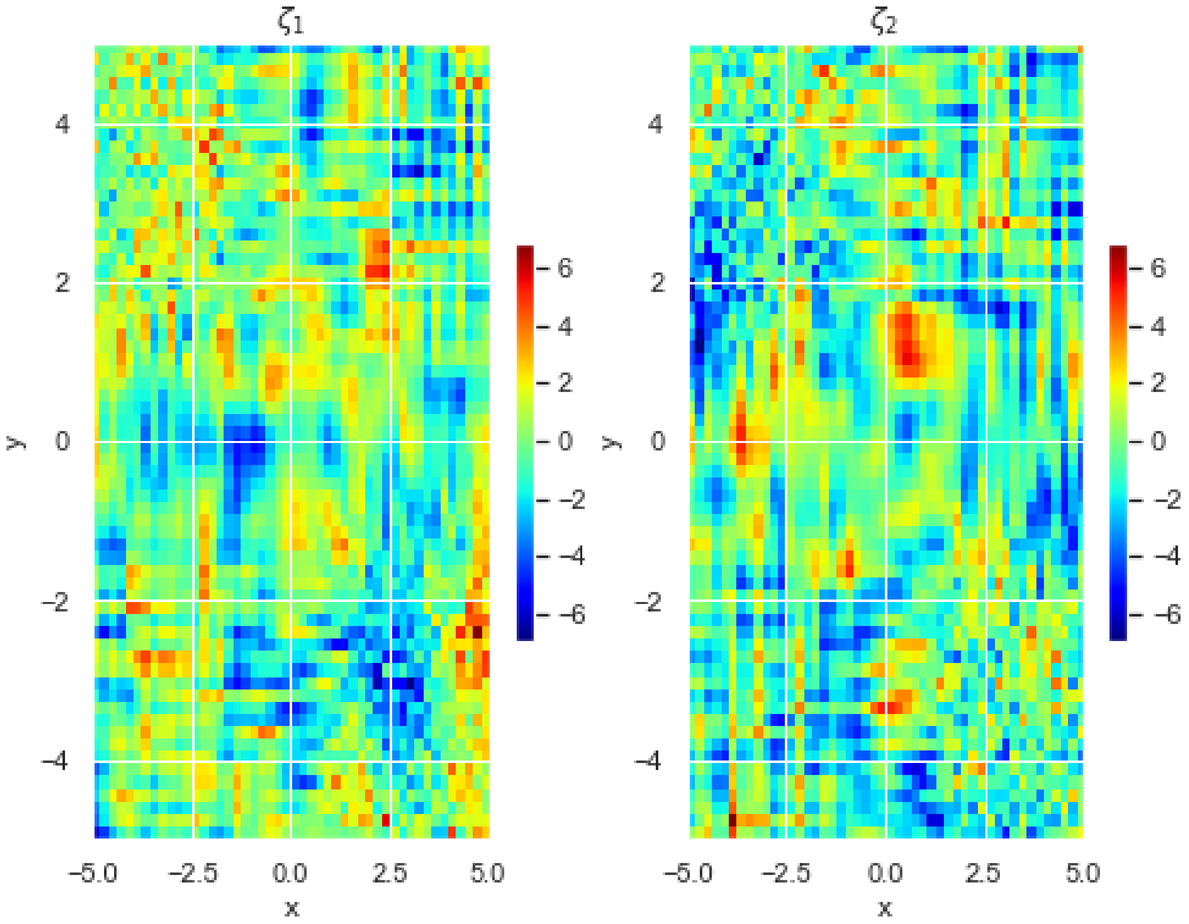


Fig. 6.21 Realisation from score processes corresponding to Scenario D.

a non stationary spatial dependence. In particular, we choose a two component Gibbs kernel to simulate from. The length scale model for each components is given by:

$$\begin{aligned} l_1(s) &= \frac{1}{1+\exp(-s)} \\ l_2(s) &= \frac{1}{1+\exp(s)} \end{aligned}$$

where, as mentioned before, the Gibbs kernel is extended to multiple dimensions by applying the Gibbs kernel over each separate dimension of the domain $\mathcal{S}$. An indicative example of the scores that are generated from these kernels is given in Figure 6.21. As can be seen, the realised scores for each kernel have a similar structure, but are clearly non-stationary, with the corners of the domain having least structure while the middle of the domain has fairly strong structure. A corresponding example realisation from this scenario is displayed in Figure 6.22. This highlights how the non-stationary score processes manifest in the scenario simulations.

For the study itself, we first present the training metrics of our various prospective models. This is displayed in Table 6.8. As we can see the best in class model on the training

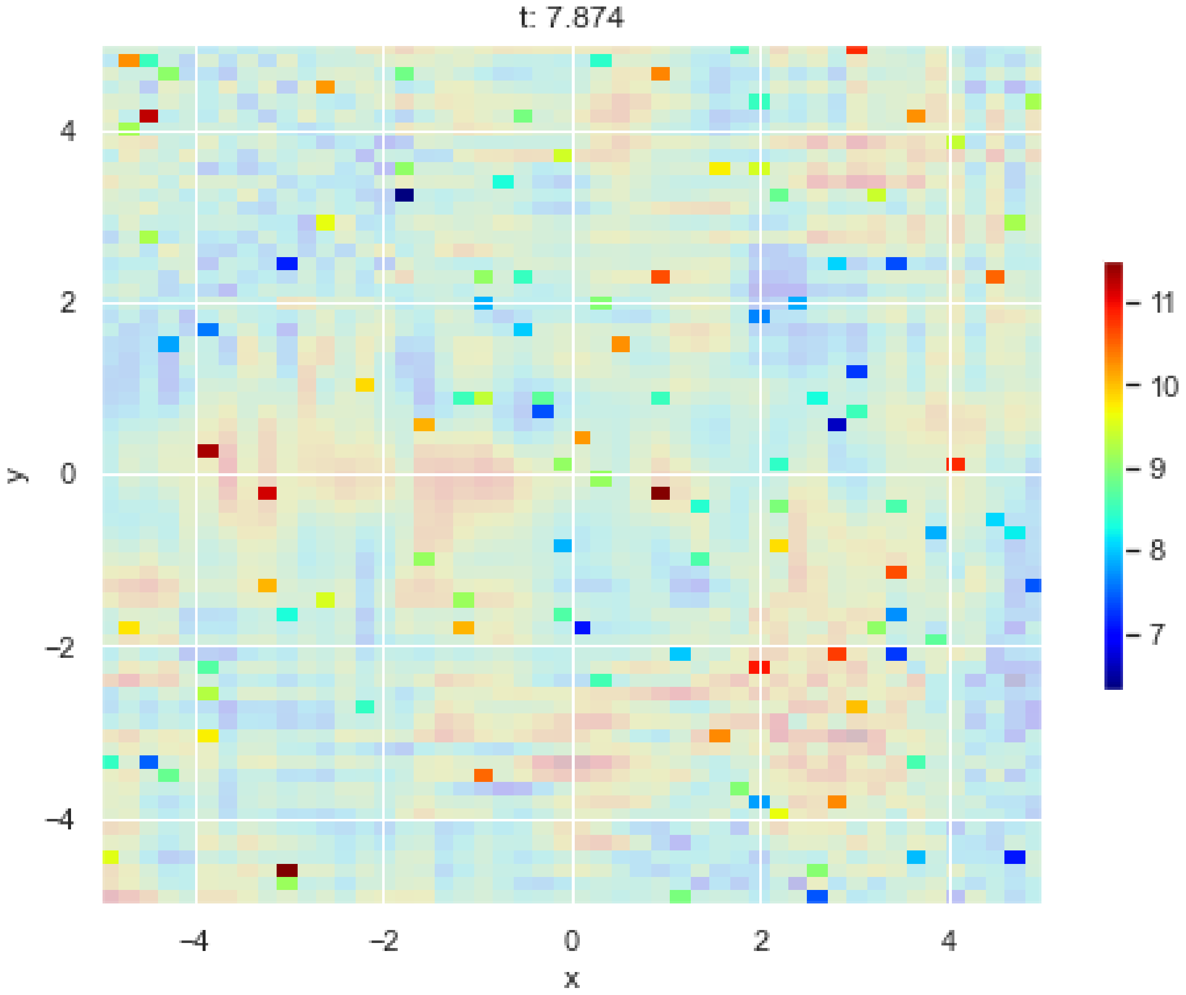


Fig. 6.22 Spatial view of a particular time point in $\mathcal{T}$ from a realisation from Scenario D. This realisation corresponds to the example score realisation in Figure 6.21. The darker pixels correspond to the noisy observations, with the lighter pixels representing unobserved locations at this time point.

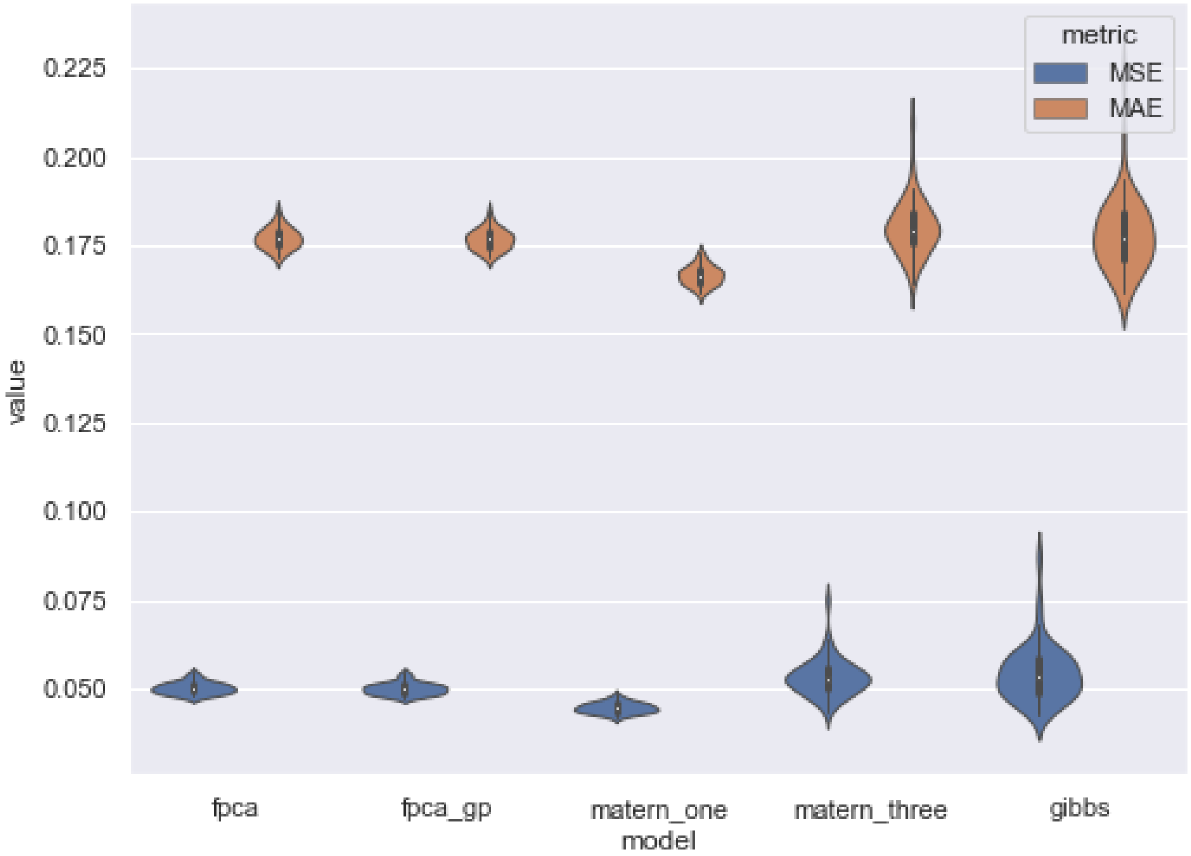


Fig. 6.23 Distribution of metrics, MAE and MSE over the training data for each simulation in Scenario D.

metric is the Matérn One Half model. However, all the models perform comparatively as well as each other. The Matérn One Half model is chosen as best in class based on its overall performance. If we look at the distribution of the metrics over the 50 simulations, Figure 6.23, we see that it is on average the best performer. However we can see the Gibbs model does have the optimal performance, but also the largest range in performance over the simulations of both metrics. We can see that the PACE and White kernel models also perform well on the training metrics. We account for this, as although the simulations have spatial dependency, some of this spatial dependency is quite weak. For example, at the corners of the domain $\mathcal{S}$; and so a reasonable approximation to the data generating procedure at these locations would be the White kernel CPACE models.

Table 6.8 Simulation results for the models ability to estimate the functional data at points of observation. Bold indicates best in class.

Model	MSE	MAE
pace	0.0503 (0.0017)	0.1772 (0.0030)
fpca_gp	0.0501 (0.0017)	0.1768 (0.0029)
matern_one	**0.0447 (0.0015)**	**0.1667 (0.0027)**
matern_three	0.0532 (0.0052)	0.1802 (0.0077)
gibbs	0.0542 (0.0079)	0.1785 (0.0109)

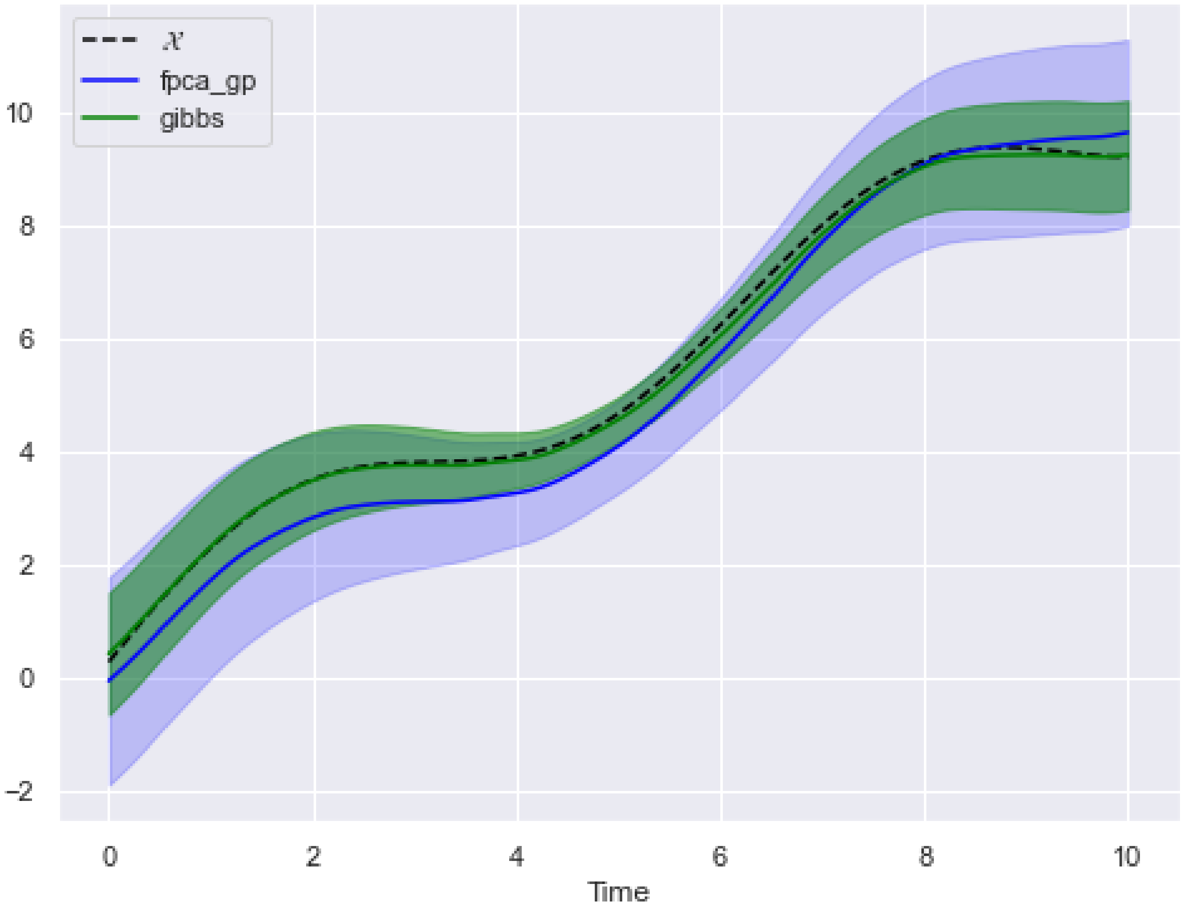


Fig. 6.24 An indicative example of model reconstructions for Scenario D for functional data with no observations. The shaded region is the prediction confidence band and corresponds to a 95% simultaneous confidence interval.

We now consider the resulting metrics on the test dataset for Scenario D. Table 6.9 displays these. Here we can see that on the test data, the Gibbs model becomes the best in class. We can see that the models with spatial dependency have an advantage over those which do not, which is as expected. The Gibbs model seems to have captured that added complexity in this scenario, by allowing for a non-stationary structure. This has given the model the edge in the ability to reconstruct unobserved data. However, it is worth noting that the Gibbs model has the largest variance in metric results of the models with a non-White kernel. This probably indicates that there is a possibility that using the more complex kernel in the CPACE framework means the estimation of hyperparameters may not always converge at the optimum values. This is a common situation for models where the parameter space is large and optimising, for the optimal parameters, may sometimes land in a local optimum rather than a global optimum. This issue, and how we attempted to minimise this issue, is discussed in Chapter 8. Finally, for illustration, we provide an example reconstruction of an unobserved location for the `fpca_gp` and `gibbs` models in Figure 6.24. Here we can clearly see the improvement in prediction from using the CPACE framework, both for mean prediction and confidence of prediction.

6.6 Summary

In this chapter we have considered a simulation study to assess the ability of the CPACE model in various conditions. Our primary aim was to compare the CPACE model, which introduces explicitly modelling spatial dependence between functional observations, with the standard PACE modelling. We consider four different scenarios with various levels of spatial dependence, to assess how the CPACE framework behaves. In these scenarios we considered using 4 different models under the CPACE framework, each utilising a different assumed form of the score processes. We have seen, from Scenario A results, that regardless of the assumed form of the score process, the CPACE framework can effectively mimic the PACE framework. This is achieved by estimating hyperparameters for the score processes that cause these to mimic the White kernel. Further, we have seen through Scenarios B and C that the CPACE model can accommodate stationary spatial dependence between functional observations. Here we saw that choosing the assumed score process as close to the data generating procedure is obviously the preferred choice, but using a score process which is flexible enough to accommodate stationary spatial dependency will give similar results. Finally, we tested the CPACE model on data which has a complex non-stationary spatial dependency between functional observations. Here, on training data the assumed score process chosen seemed not to mater too much. However, when looking at metrics on the test data, we saw clearly that the models which assumed only a stationary score process were outperformed by the Gibbs model. This underlines a common theme. That is, assuming a slightly more complex form of the score process under the CPACE framework tends to work better for test data reconstruction. This is because the CPACE framework allows nicely for tuning of hyperparameters so that complex forms can estimate simpler forms. However the reverse is not true. For example, one cannot choose hyperparameters of the Matérn kernel such that it becomes non-stationary. Whereas we can make the Gibbs kernel stationary by assuming the length scale model is constant over the spatial domain.

We have highlighted the pros and cons of the CPACE model, and contrasted this with the ability of the PACE model on simulated data. We proceed to discuss the CPACE model on our real world data, the CESM-LE dataset (see Chapter 2 for an in-depth description), in Chapter 7. We discuss the implementation of the CPACE model, which mainly revolves around the hyperparameter estimation of the assumed scores processes in Chapter 8.

Table 6.9 Simulation results for the models ability to estimate the functional data at points of observation. Bold indicates best in class.

Model	MSE	MAE
`pace`	4.9249 (0.4543)	5.4494 (0.2338)
`fpca_gp`	4.9249 (0.4543)	5.4494 (0.2338)
`matern_one`	2.4335 (0.1310)	3.7206 (0.0901)
`matern_three`	2.6123 (0.1446)	3.8147 (0.0952)
`gibbs`	**2.0135 (0.2170)**	**3.3181 (0.1806)**

Chapter 7

Application of CPACE model to CESM-LE

We have described the CPACE model in Chapter 5, in which we essentially model our observations as a noisy realisation from a Gaussian process with both a spatial and temporal domain. The kernel of the CPACE model is designed to have a temporal component which is constructed through the data's principal components and a spatial component which is chosen to accurately reflect observed spatial dependency between functional observations. In Chapter 6 we studied this model on simulated data where the simulated data arose from the same data generating procedure which the model is designed on. In this chapter we consider the model's ability to reconstruct unobserved data from the CESM-LE dataset, as described in Chapter 2, and compare to the FPCA model which considers no spatial dependency. This data has not come from our model's data generating procedure and this analysis will give us an understanding of the CPACE models' ability to capture phenomenon in a real world setting.

We begin by setting out two studies for this data; one which considers the globe at a reduced resolution, and one which considers only the continent of Europe. Following this we describe the spatial kernels we will consider in this study. Finally, we present the results of these studies for each of the four variables considered; Pressure (PS), Temperature (TREFHT), Wind speed (U10), and Precipitation (TMQ). More details on these variables and the CESM-LE dataset in general is given in Chapter 2.

7.1 Studies

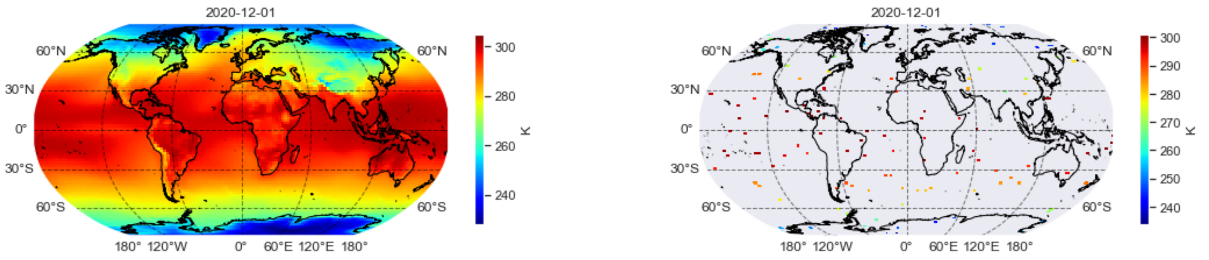
In the following we set out our studies on the CESM-LE data used to examine the CPACE models' reconstruction ability. This dataset is widely used, and often as a case study for an application of a new predictive methodology. For example, see [32]. We use two separate study areas, which are the same across our variables of interest. These correspond to a whole globe study, designed to see the CPACE ability at large spatial distances, and a Europe specific study which is designed to see if the framework can pick out more intricate details in reconstruction. In both cases our main priority is to reconstruct unobserved functional data, hoping to utilise observed data in neighbouring regions to help inform our reconstructions.

In both studies we assume we only observe sparse functional data, which we have enforced like our simulation study in Section 6.1, by only observing a selection of locations and a selection of time points from the full dataset. We could consider this scenario occurring in a real world study from perhaps a defective monitoring station which either fails to capture any readings at all, or only partially captures the variable of interest over our study period. Another situation where sparsely observed data across space may occur is when it is practically difficult or impossible to place monitoring stations at locations of interest and thus you can only collect observations at few points in your domain. Shen et al. gives an overview of the areas in satellite image reconstruction where this occurs, [69]. We specify the details of this sparsity and the study area for each study in Section 7.1.1 and Section 7.1.2.

7.1.1 Global Study

We consider the data from the CESM-LE dataset for the whole globe. That is, our spatial domain consists of the surface of the earth. In the latitude, longitude coordinate system this would correspond to setting $\mathcal{S} = [-90, 90] \times [-180, 180]$. As mentioned in Chapter 2 we downsample the resolution of this dataset to have a spatial grid size of 64×96. We have designed this study to observe large scale trends across space, and so this reduction in resolution is perfectly acceptable for this study. Figure 2.2b highlights the impact in resolution reduction for this case on the TREFHT variable.

As mentioned above, we withhold some observations to construct a sparse functional dataset. In particular, we assume that we only observe 10% of our spatial locations across the globe. For these we further induce sparsity by assuming we only observe between 5% and 15% of the temporal points for each of these observations. In this study, we uniformly choose the number of temporal observations for each of these locations between these percentages. This is done independently for each of the 40 replications which make up the study; so we have a random selection of temporal points each time. This construction is similar to that of the sparsity used in the simulation study in Section 6.1. These observed data we take to be our training dataset. We do not add any noise to our training data, but let our CPACE framework decide on the noise variance under its observational model. The partially observed functions we take to be our validation dataset. The remaining unseen observations we take to be our test dataset. We do this process independently for each of the 40 simulations present in the CESM-LE dataset. Figure 7.1 gives an indication of the training observations compared to the full data at a particular time point in the dataset for the TREFHT variable. As we can see this is an extreme case of sparsity, which should provide a good challenge to the CPACE framework.



(a) Observed and unobserved data.

(b) Observed data.

Fig. 7.1 An example of the sparsity induced from the generating procedure in Section 7.1.1 for the TREFHT variable of the CESM-LE dataset. Sparsity observed from the spatial view point.

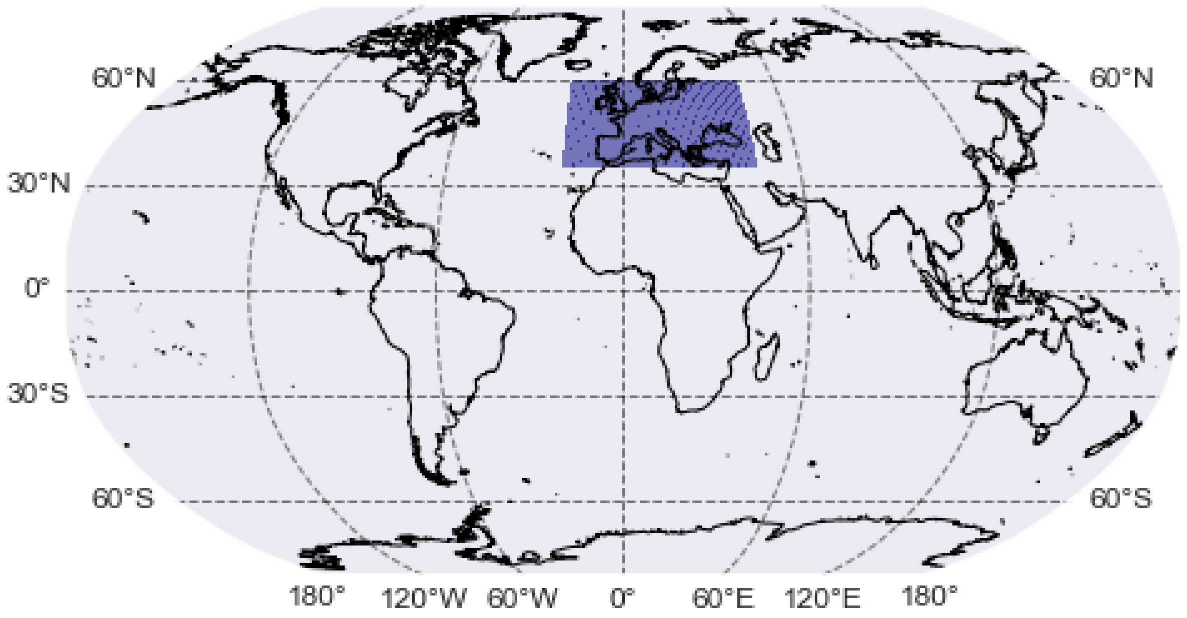
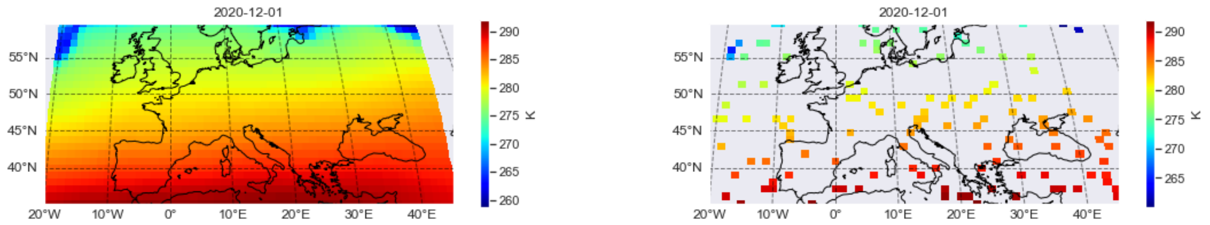


Fig. 7.2 Area of interest in the European study in blue. This is fixed and the same for each variable of interest in the CESM-LE dataset.

7.1.2 European Study

In contrast to the Global study presented in Section 7.1.1, this study will consider a much narrower spatial domain. We choose to observe in this study the domain bounded by the following latitude, longitude coordinates; $(-20, 35), (-20, 60), (45, 60), (-20, 60)$. This corresponds to setting our spatial domain to; $\mathcal{S} = [-20, 45] \times [35, 60]$ in a latitude, longitude coordinate system. This corresponds roughly to European continent, and is indicated in Figure 7.2.

The proposal of this study area is to consider a smaller spatial domain, where there is less likely to be large spatial variation. In this case we no longer downsample the resolution and keep the full resolution data from the CESM-LE datasets. The study is then set up that we observe much more densely the functional data over this domain. We use this setup to consider how well the CPACE framework deals with the intricacies of smaller scale variation as compared to the Global study.



(a) Observed and unobserved data.

(b) Observed data.

Fig. 7.3 An example of the sparsity induced from the generating procedure in Section 7.1.2 for the TREFHT variable of the CESM-LE dataset. Sparsity observed from the spatial view point.

Another difference we consider in this study is the frequency of observation. In this study we choose to observe the functional data more regularly and at more locations across the domain. Here, this may be viewed as relating to the scenario of having many monitoring locations, but often readings are missing of your variable of interest due to technical difficulties. For example, cloud obstructing imagery taken from satellites, [55]. This sparsity is induced in our dataset for this study by using the following parameters to generate our training data.

We assume we observe 25% of our possible spatial locations across our domain, $\mathcal{S}$. As with our previous studies we further induce sparsity by assuming we only observe between 20% and 40% of the temporal points for each of these observations. We do this independently for the 40 simulations present in the CESM-LE dataset. This construction is similar to that of the sparsity used in the simulation study in Section 6.1. These observed data we take to be our training dataset. The functional data which have partial observations on them we will take to be our validation dataset. The remaining unseen observations we take to be our test dataset for this study. We do this process independently for each of the 40 simulations present in the CESM-LE dataset. Figure 7.3 gives an indication of the training observations compared to the full data at a particular time point in the dataset for the TREFHT variable. As can be seen this sparsity is less severe than that of the globe study, but still present.

7.2 Spatial Kernels

As in the simulation study from Section 6.1, we consider a variety of spatial kernels in the CPACE framework to compare to the FPCA approach. In the following we describe the kernels which we use for the Global and European study described in Section 7.1. For most spatial kernels we must specify the metric between points of observation which captures the concept of distance in $\mathcal{S}$. In all of our studies on the CESM-LE data we must consider how we define this for our domain $\mathcal{S}$ which as given is the surface of the globe. We do so by projecting our domain from the surface of the globe to $\mathbb{R}^3$.

7.2.1 Domain Projection

The coordinate system for the CESM-LE data is given in latitude and longitude. The latitude of a point on the Earth's surface is the angle between the equatorial plane and the line which passes through that point and the centre of the Earth, [52]. Similarly the longitude of a point on the Earth's surface is the angle between the prime meridian and another meridian which passes through that point, [52]. As such, it describes a coordinate reference system (CRS) over the surface of the globe. Therefore, when considering distance between two points in $\mathcal{S}$ we should consider the distance travelled on the globe between the two. If the Earth was a true sphere this would be relatively simple as the distance between two points can be found using the great-circle distance. This is given by the formula below for two points $(\mathrm{lat}_1, \mathrm{lon}_1)$ and $(\mathrm{lat}_2, \mathrm{lon}_2)$:

$$r \arccos\left(\sin(\mathrm{lat}_1)\sin(\mathrm{lat}_2) + \cos(\mathrm{lat}_1)\cos(\mathrm{lat}_2)\left(\mathrm{lon}_1 - \mathrm{lon}_2\right)\right).$$

However, the world is not a true sphere, thus the above great circle distance would be an approximation. In fact a standard coordinate system, which the CESM-LE data uses, is the World Geodetic System (WGS) 84. This is a specific coordinate system which describes the surface of the globe using an ellipsoidal model, [52].

Another issue with such an approach is how we define hyperparameters, such as length scale parameters in the Matérn Three kernel (See Section 7.2.2), when using the WGS84 coordinate system. Since the WGS84 defines the coordinate system using latitude and longitude it revolves around angles. We would therefore need to consider sensible measures of distance between these angles. For example; the distance between two points on the globe with the same latitude but longitudes of 0 and 360 respectively should be zero. Guinness and Fuentes discuss a variety in [25], they have computation complexities in implementing.

As suggested in [25], we decide to project our domain $\mathcal{S}$ into $\mathbb{R}^3$. In this space we have alleviated the above issues as the standard concept of Euclidean distance will suffice in the kernels we study. That is, we project our surface of the globe into three dimensions, and calculate the distance between two points as the Euclidean distance in this space rather than using the great circle distance or a custom metric on the WGS84 coordinate system. Figure 7.4 highlights the difference between these approaches for the distance between two points `A` and `B` on a circle and the Euclidean distance. Here we have reduced the dimension for ease of illustration but the same concept extends to three dimensions and the surface of the unit sphere.

Although it is a simplification Guinness and Fuentes highlight it is not a limiting factor, [25]. Intuitively, this is because, although our measure of distance between two points on the globe is not accurate in the sense of true distance travelled our kernel hyperparameters will be estimated from the data and should allow for this from the observed correlation. In particular the approach to projecting to $\mathbb{R}^3$ will overcome the issue of periodicity in using latitude and longitude systems.

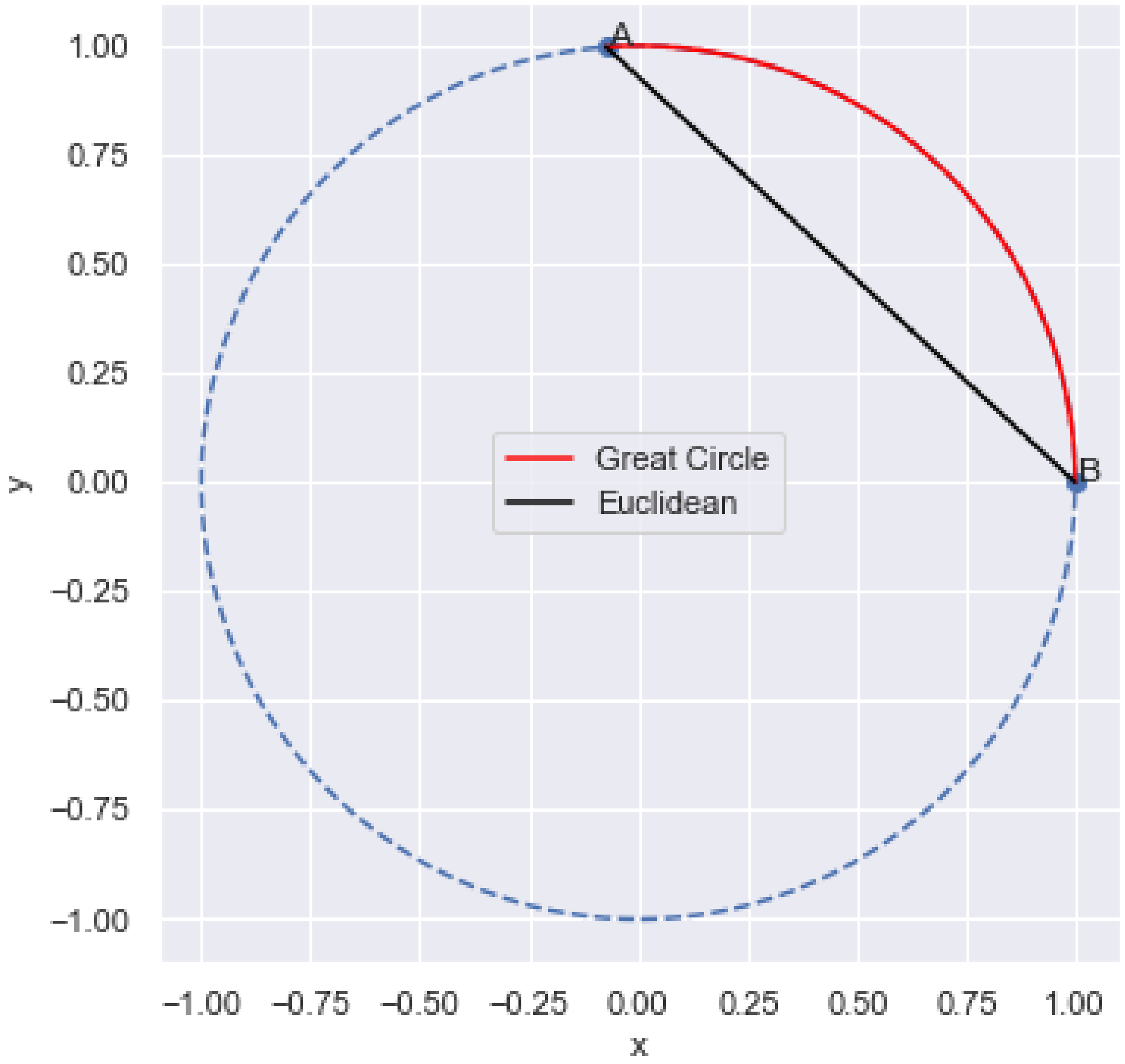


Fig. 7.4 Difference in great circle distance and the Euclidean distance between point A and point B on the unit circle.

Thus for each of the kernels specified below we first project our points in the spatial domain to $\mathbb{R}^3$ and then calculate the kernel value between them. We use a projection from the WGS84 geodetic view of latitude, longitude to the geocentric view of element in $\mathbb{R}^3$, [52], which corresponds to the projection that takes $\mathcal{S}$ from the surface of the globe to $\mathbb{R}^3$.

We do this for the full CPACE framework, for both likelihood calculation of our kernel hyperparameters and prediction.

7.2.2 Kernels

Our first kernel is the White kernel, see Section 6.1.3 for a complete description. We will denote the CPACE model with this kernel as `fpca_gp` in our studies as it corresponds to FPCA model which does not take into account the spatial dependency between functional observations but is estimated under our Gaussian process framework.

The second kernel we consider is the Anisotropic Matérn Three kernel. Alike in the simulation study, Section 6.1.3, this kernel follows a Matérn form that is formed from specifying that $\nu = \frac{3}{2}$. However, in this kernel we let the length scale hyperparameter be different for each dimension of the domain. That is, the general form of the Anisotropic Matérn Three kernel is given by:

$$\zeta_k\left(\boldsymbol{s}_i, \boldsymbol{s}_j\right) = \lambda_k\left(1 + \sqrt{3}d\right)\exp\left(-\sqrt{3}d\right),$$

where $d = \sqrt{\left(\boldsymbol{s}_i - \boldsymbol{s}_j\right)\boldsymbol{R}^{-1}\left(\boldsymbol{s}_i - \boldsymbol{s}_j\right)^{\intercal}}$, $R = \operatorname{diag}\left(\boldsymbol{\rho}\right)$ and $\boldsymbol{\rho} = \left(\rho_1, \rho_2, \rho_3\right)^{\intercal}$ is our vector of length scale parameters per dimension. Here we have restricted the anisotropy to a length scale over each dimension of the domain which are independent of each other by specifying the diagonal form of R. In this case $R \in \mathbb{R}^{3\times3}$ as we will project our spatial domain to three dimensions as discussed in Section 7.2.1. We will denote this kernel by `matern_three` for this study. Figure 7.5 highlights an example covariance structure using this anisotropic kernel with $\boldsymbol{\rho} = (1.0, 4.0, 4.0)$ on our spatial domain $\mathcal{S}$ projected. Note the off diagonal structure which comes from the shorter length scale in the first dimension. We choose to use an anisotropic stationary kernel such as this as it is well used in the literature for geospatial applications, [11]. The anisotropy should allow the spatial kernel to capture difference in correlation between points of latitude and longitude which are often present in EO datasets.

The final kernel we consider is the Gibbs kernel, [23]. We use `gibbs` to denote this kernel in our studies. This is similar to the one we utilised in the simulation study in Section 6.1.3. The only difference in the kernel we use for our study on the CESM-LE data is that we limit ourselves to two components in the kernels. We do this after initial ad-hoc tries of differing number of components. We found that two components was more than sufficient to capture interesting components of the data whilst maintaining simple computation. We speak more of this in Chapter 8. Again, this is a non-stationary kernel which means it has the capability to capture non-stationary covariance structure in the CESM-LE dataset. Similarly, the construction of the kernel should also allow it to capture stationary covariance structures as well. Alike the Matérn Three kernel above, it should be able to capture any anisotropy in the data. This is important for EO data, especially when observed at large spatial scales.

To compare the effectiveness of the CPACE models with different kernels and the FPCA framework we use a collection of metrics. We describe the metrics used in the study of the CESM-LE dataset in the following section.

7.3 Metrics

Alike our simulation study, Section 6.1, we use a variety of metrics to compare performance of our models. As with the simulation study we consider the RMSE and MAE metrics on the training, validation, and test datasets separately. Here we consider the training dataset to be the spatio-temporal points which we observe. The validation dataset consists

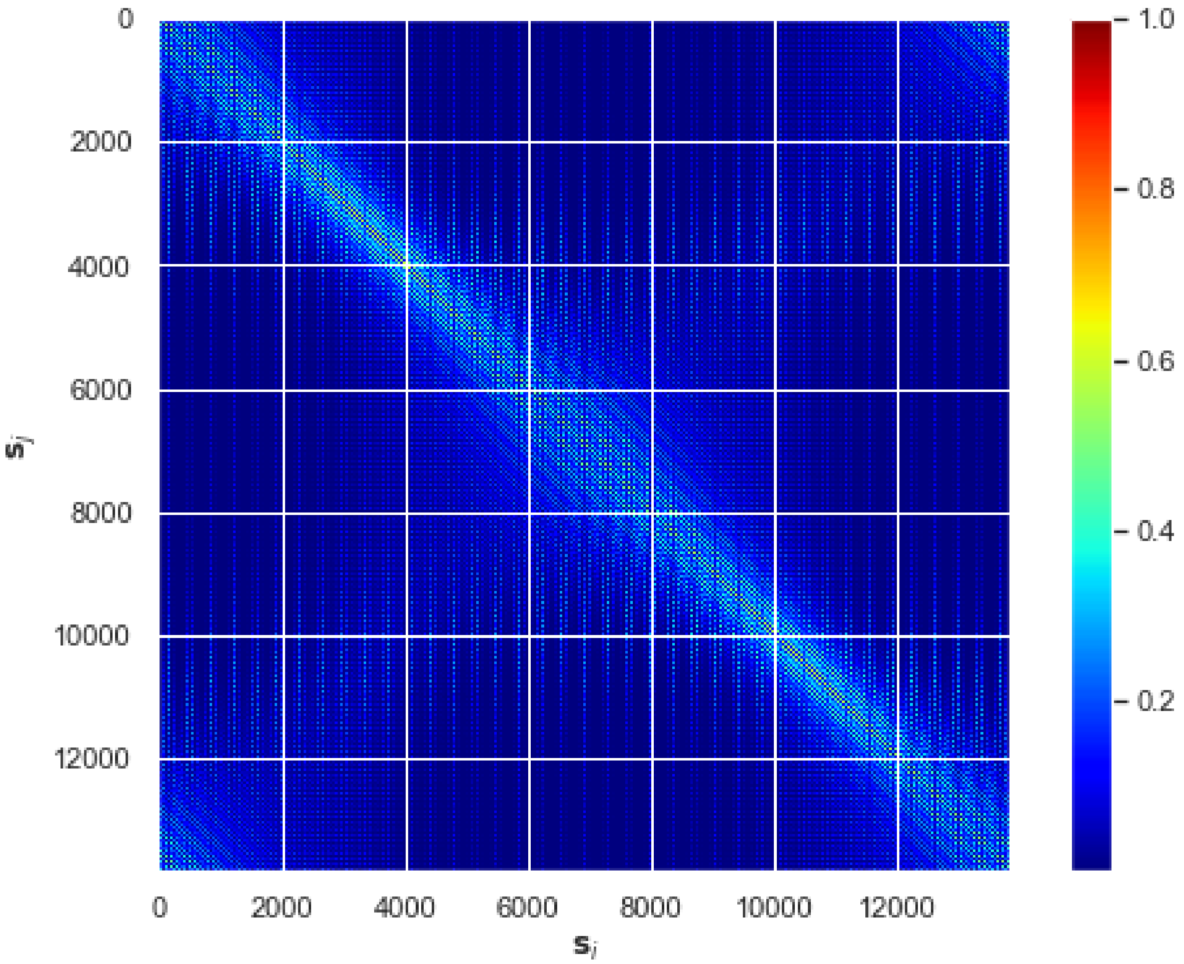


Fig. 7.5 An example of the covariance structure with the anisotropic Matérn Three kernel.

of all temporal points at each spatial location where we have at least one observation. Finally, the test dataset consists of all time points for which we have no observations for the spatial location. The metrics we consider are a measure of the ability of the models to reconstruct the individual pixel functions over the temporal domain. The same reasoning as given in Section 6.1 is applicable for the choice of these metrics for evaluating model performance.

In addition to the RMSE and MAE metrics we consider some metrics more tailored to image reconstruction. We utilise the PSNR and SSIM metrics, as described in Section 4.3.3. Both of these metrics offer a comparison of perceived similarity between images. This differs from the RMSE and MAE metrics which only consider difference between pixels in the images, and make no attempt to allow for the structure of the reconstruction. We use these metrics as a further gauge on the comparative performance of the CPACE framework with an eye to provide good reconstruction across the whole spatial domain. As such, these metrics will be evaluated using the full dataset which is both the validation and test data combined. Both these metrics are also discussed in Section 4.3.3 and their use in image reconstruction models is wide, [31].

We note as the CESM-LE data consists of 40 replications of each variable of interest across our spatial-temporal domain, we can evaluate these metrics on each simulation and present the distribution of these metrics across the simulations. We now proceed to discuss the results for both our Global and Europe study for each of our variables of interest from the CESM-LE dataset.

7.4 Global Study Results

Here we present the results of the study across the whole globe as described in Section 7.1.1. We present the results separately for each variable of interest; Pressure (PS), Temperature (TREFHT), Precipitation (TMQ), and Wind Speed (U10).

We have considered four models for this study, the same setup for each variable of interest. The `pace` model corresponds to the FPCA model or PACE framework, as describe in Chapter 3, which does not take into account spatial dependency. The second, `fpca_gp`, is our CPACE model with the White kernel. Again this does not take into account spatial dependency between functional observations, but is computed under our CPACE framework which allows for hyperparameters of the kernel; namely the spatial kernel variance for each component to be estimated using the Gaussian process framework. Thirdly, we use the Matérn Three kernel with anisotropic length scales as the spatial kernel in another CPACE model. This we denote by `matern_three`. Finally, we denote by `gibbs` the CPACE model using the Gibbs kernel. For each of these models we use 5 components in our decomposition of the observed data which are applicable in both the PACE and CPACE framework.

7.4.1 Pressure (PS)

Here we present and describe the results of using our CPACE framework to model the pressure variable from the CESM-LE dataset across the globe.

We begin by presenting the metric results for our validation dataset. These are reconstructions of partial observed functional data. These are displayed in Table 7.1. As can be seen there is a discrepancy between the CPACE models and the PACE model. This can be seen as an impact of the large sparsity we have used in the study. Since we assume such sparse observations it is obviously difficult for the FPCA model to truly capture the variations in the functional data. This can be alleviated in the models which use spatial dependency as they can utilise spatial dependency to inform the curve at the prediction location to be similar to those close by. It is alleviated in the `fpca_gp` model since the CPACE framework allows for further estimation of the White kernel variances in accordance with the Gaussian process structure, something which the `pace` model does not allow for.

We see this pattern again for the same metrics on the test dataset. These are presented in Table 7.2. Again we can see the improvement resulting from the use of spatial information to help predict the curves. We note the extremely poor performance of the `pace` model comes from a few simulations in the dataset having catastrophic errors. The performance of this model on the other simulations is in line with the `fpca_gp` model. We can see an

Table 7.1 Results for reconstruction of the validation data for the PS variable in the Global study from the CESM-LE dataset. Bold indicates best in class. We give the mean and standard deviation of the metrics over the 40 realisations.

Model	RMSE	MAE
`pace`	3634.1855 (2187.4974)	1379.6860 (606.6431)
`fpca_gp`	726.3168 (129.7112)	500.9342 (72.5722)
`matern_three`	**629.5518 (83.7596)**	**454.7355 (45.9620)**
`gibbs`	704.5824 (100.7908)	491.9828 (42.4173)

Table 7.2 Results for reconstruction of the test data for the PS variable in the Global study from the CESM-LE dataset. Bold indicates best in class. Scale of $e + 03$.

Model	RMSE	MAE
`pace`	4229.290 (298074.2)	69.2113 (4.63648)
`fpca_gp`	12.4485 (2.9705)	6.3486 (0.1641)
`matern_three`	**2.5648 (0.0138)**	**1.3305 (0.0387)**
`gibbs`	3.8448 (0.6388)	1.9998 (0.1129)

illustration of this occurring in Figure 7.7. Here, we can see how the true curve is very variable, and although all our models get the relative shape of the curve, they completely miss the peaks and troughs. This is because our component functions for these models haven't captured this aspect of the data. This results in fairly large prediction errors. However relative to the PACE framework our CPACE framework performs well. We do not include the `pace` model in the illustrative figures, this is due to it's similarity with the `fpca_gp` model. They tend to only differ at the points of catastrophic error that we mention above.

In fact the eigen decomposition is mostly dominated by the leading principal component. This can be seen in Figure 7.6. This figure displays the effect of the first eigenfunction and the lighter grey curves show an example of the true underlying functional data. As can be seen the models miss any periodic elements due to the fact that the major variation in this data comes from a shift in the curve level, which is captured in this eigenfunction.

Finally we consider the full dataset and present our image reconstruction metrics for this variable. These are shown in Table 7.3. Here we can clearly see the advantage of the Matérn Three model over say the Gibbs model. We can see a distinct advantage in both the PSNR and SSIM for this model. We can also see a distinct advantage in using the CPACE methodology with the White kernel over the standard PACE methodology. This is likely explainable due to the ability to fine tune the variance of the white kernels from the eigen decomposition. We illustrate further the comparative performances of these models by plotting the reconstruction for the dataset at a particular time point. Figure 7.8 displays this. We can see clearly the impact of utilising the spatial dependency but note we do fail to capture intricate details like the variance in pressure over South America. Similarly we can see the Gibbs model captures some more intricate spatial details but fails

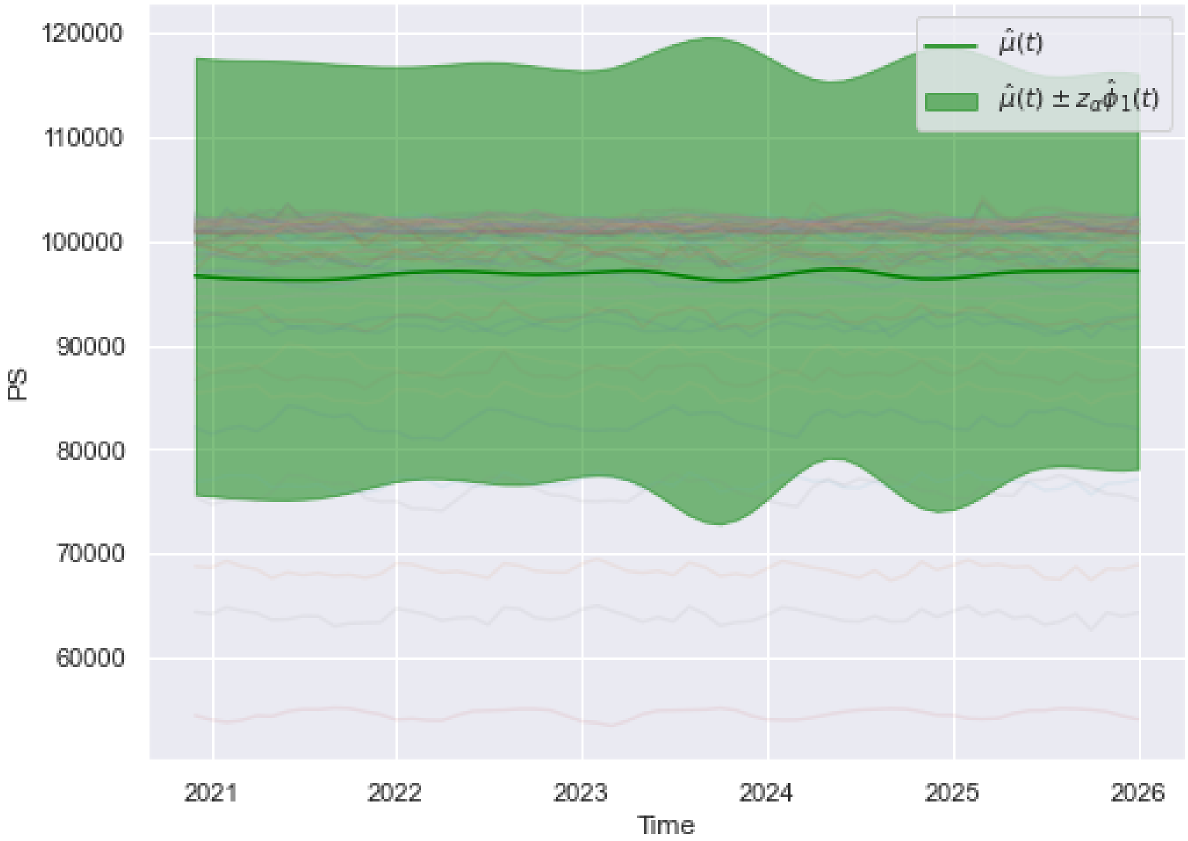


Fig. 7.6 The first eigenfunction from the FPCA decomposition used in all models for the PS variable in the Global study. Example true curves in light grey given for context.

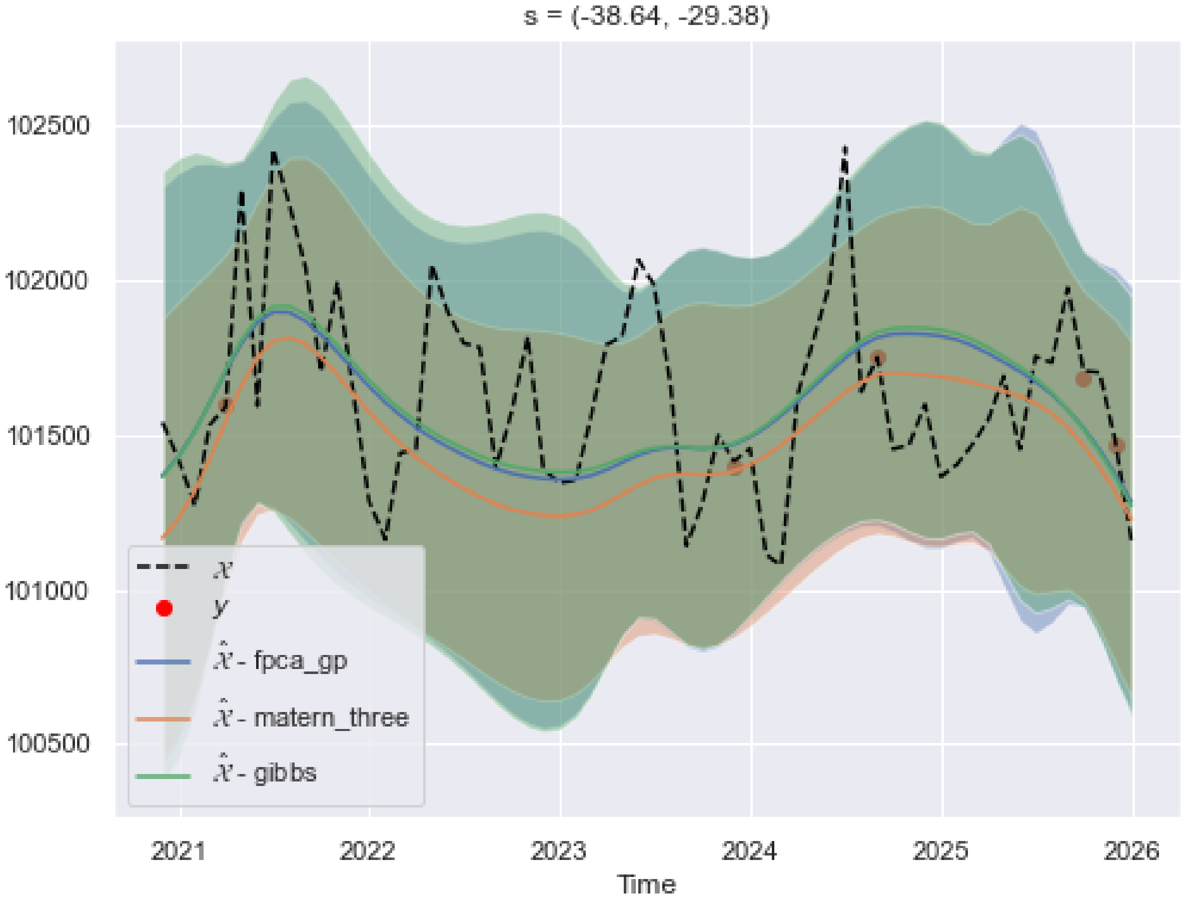


Fig. 7.7 An indicative example of the CPACE model performance on reconstruction of the CESM-LE pressure variable from the validation dataset from the Global study.

Table 7.3 Results for reconstruction of the full data for the PS variable in the Global study from the CESM-LE dataset. Bold indicates best in class.

Model	PSNR	SSIM
pace	-39.7596 (4.5310)	0.4016 (0.0234)
fpca_gp	12.6672 (1.8675)	0.4239 (0.0097)
matern_three	**26.1533 (0.4736)**	**0.8364 (0.0068)**
gibbs	22.7488 (1.3351)	0.7329 (0.0389)

to match the Matérn kernel in overall performance as it does not capture the scale of the changes as well.

Overall, we can see that the CPACE framework adapts well to the pressure variable from the CESM-LE dataset. We can see from Figure 7.6 that we find a reasonable and interpretable eigenfunction, which corresponds to level shifts in the pressure. However we find that this dominates the eigen decomposition, and thus the results can have relatively large errors due to failing to capture intricate local variations in the data. We view this as stemming from the sparsity of the study dataset, as many of these variations are not observed in the training data. For example, Figure 7.7 shows we only observe 4 observations on that whole example function. The CPACE models with spatial dependency perform substantially better for reconstruction from both the test metric perspective and the image reconstruction perspective. These promising results are a good indication of the usefulness of the CPACE framework with regards to EO data across the globe.

7.4.2 Temperature (TREFHT)

In this section we present and describe the results of using our CPACE framework to model the temperature variable from the CESM-LE dataset across the globe. We present these in a similar way to Section 7.4.1.

In Table 7.4 we show the resulting metrics from the study on the validation dataset. Here, we can see again the distinct advantage of the CPACE framework and especially when utilising a kernel which uses the spatial dependency. Similar to the pressure variable results in Section 7.4.1 we see that the Matérn model performs best in class followed closely by the Gibbs model. It is worth noting, that we do not observe the catastrophic error for the `pace` model unlike in the PS study. This causes the `pace` and `fpca_gp` model to be comparable. This is expected as they essentially are equivalent models, albeit the `fpca_gp` model uses our CPACE framework so has the ability to refine the eigenvalue estimates.

Next we consider the test reconstructions. That is, reconstruction of completely unobserved functions. An example of such reconstructions for our models considered is given in Figure 7.9.

Here we can clearly see the impact of using the spatial dependency which has the main advantage of being able to pick out the correct level shift of the mean function to match that of the unobserved functional data. We can also see that the Matérn kernel has a tighter confidence band than that of the Gibbs model, which may be a bit of an over

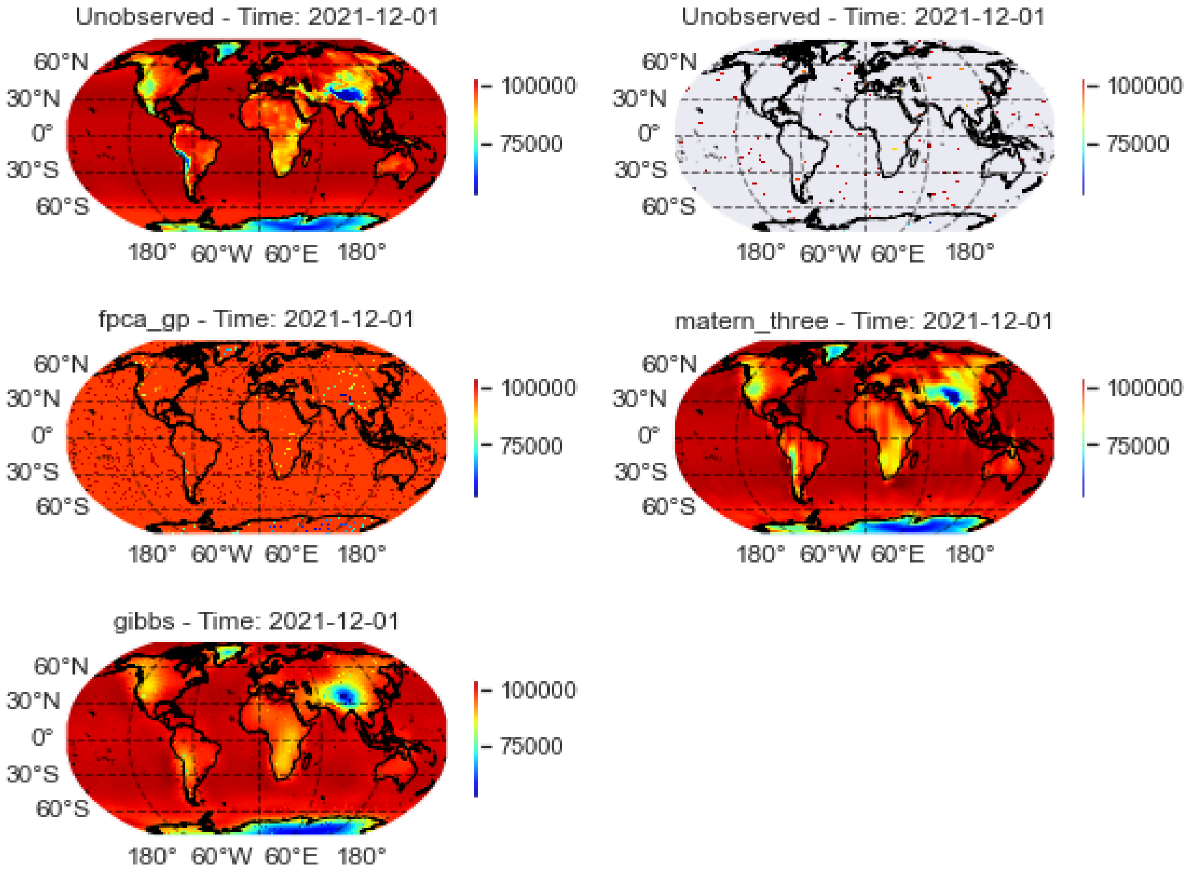

Unobserved - Time: 2021-12-01
Unobserved - Time: 2021-12-01
fpca_gp - Time: 2021-12-01
matern_three - Time: 2021-12-01
gibbs - Time: 2021-12-01
60°N
30°N
0°
30°S
60°S
180° 60°W 60°E 180°
100000
75000

Fig. 7.8 An indicative example of the CPACE model performance on reconstruction for the full globe of the CESM-LE PS variable using the various models in the Global study. Clockwise from the top left we have the true data, the observed data at a particular time point, the Matérn Three model reconstruction, the Gibbs model reconstruction, and the White model reconstruction under the CPACE framework.

Table 7.4 Results for reconstruction of the validation data for the TREFHT variable in the Global study from the CESM-LE dataset. Bold indicates best in class.

Model	RMSE	MAE
`pace`	8.2170 (0.2680)	5.5581 (0.1505)
`fpca_gp`	7.9646 (0.1809)	5.4940 (0.1154)
`matern_three`	**7.3703 (0.1805)**	**5.2337 (0.1198)**
`gibbs`	7.7548 (0.7118)	5.4087 (0.1170)

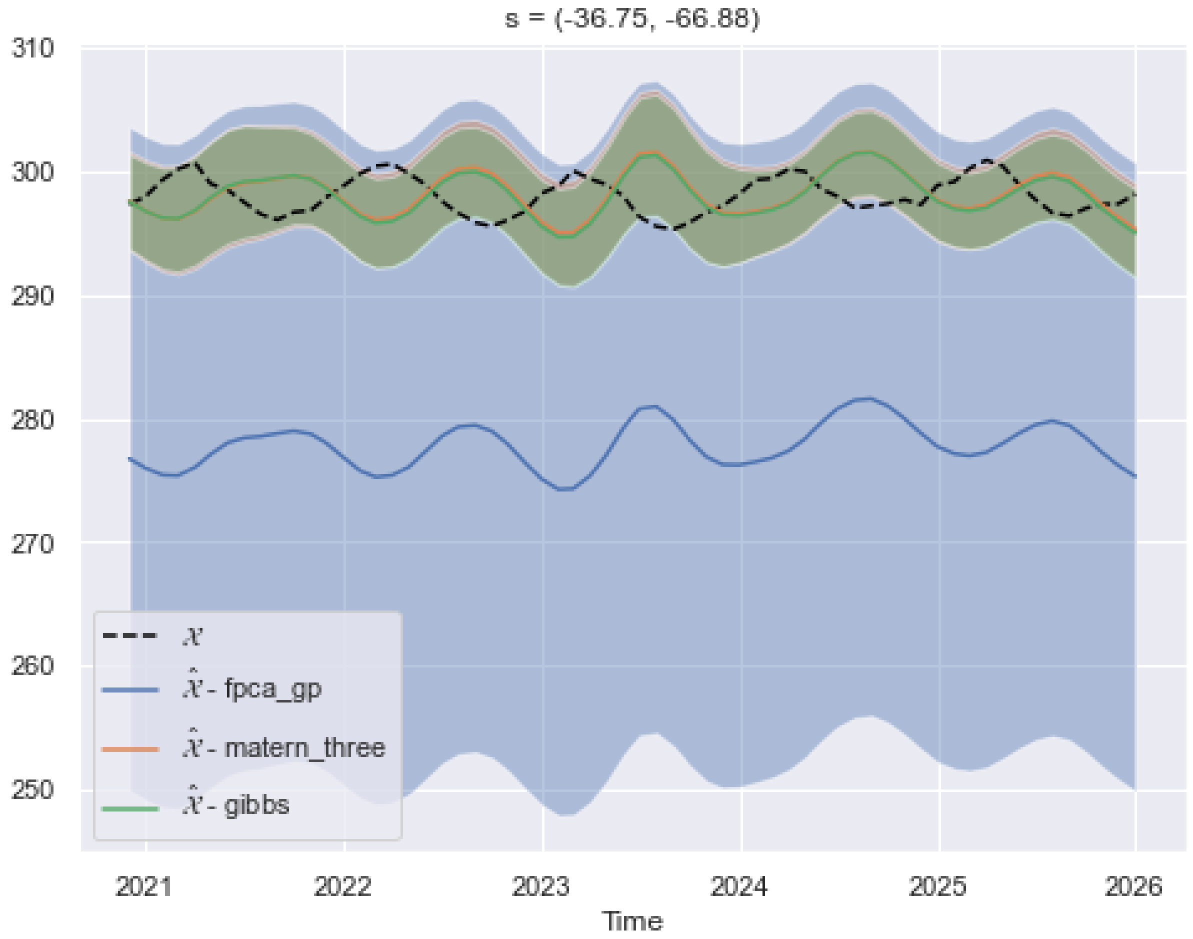


Fig. 7.9 An indicative example of the CPACE model performance on reconstruction of the CESM-LE TREFHT variable in the Global study from the test dataset. Note that the estimate curve and the true curve show alternate periodic patterns. We reason this is due to the fact that in this example the training data set consisted of more observations from the northern hemisphere which meant the model favoured eigenfunctions where the peaks occur in the months around May to August and the troughs in the months between November to February. The true curve in this case is from the southern hemisphere so shows an periodicity which is shifted by half a year; hence why we see such discrepancies. We consider a more compact study scenario which alleviates this issue for the TREFHT variable in the European Study in Section 7.5.2.

Table 7.5 Results for reconstruction of the test data for the TREFHT variable in the Global study from the CESM-LE dataset. Bold indicates best in class.

Model	**RMSE**	**MAE**
`pace`	23.3642 (1.7618)	17.9309 (0.2215)
`fpca_gp`	22.2936 (0.1841)	17.8549 (0.1656)
`matern_three`	**7.5008 (0.1373)**	**5.2875 (0.1387)**
`gibbs`	7.8361 (0.0916)	5.7924 (0.0855)

Table 7.6 Results for reconstruction of the full data for the TREFHT variable in the Global study from the CESM-LE dataset. Bold indicates best in class.

Model	**PSNR**	**SSIM**
`pace`	12.2512 (0.6205)	0.2634 (0.0043)
`fpca_gp`	12.6409 (0.0749)	0.2668 (0.0036)
`matern_three`	**22.5449 (0.2070)**	**0.8216 (0.0037)**
`gibbs`	21.9926 (0.2059)	0.7965 (0.0092)

estimate coming from model mis-specification, since the true curve sometimes falls outside this band, whereas with the Gibbs kernel it does not. The full metric results for the test data are given in Table 7.5. This indicates that the Matérn model is the best in class for the test data albeit within the range of the Gibbs model's performance. Noting this, one may well choose to prefer the Gibbs model. However, yet again we see that both of these models are vastly superior to the others.

This is confirmed if we consider the image reconstruction metrics for this dataset. In Table 7.6 we can see clearly the difference in terms of image similarity between reconstructions under the various models. For illustration, we display an example reconstruction using the various models in Figure 7.10.

Alike Section 7.4.1, we find similar patterns in our study. The CPACE framework outperforms the PACE methodology but we don't manage to truly capture all underlying eigenfunctions from our dataset. Again, as our study is using extremely sparse data this is understandable, as we simply haven't seen this variation in the training data. For the temperature variable the Matérn model tends to perform best however the Gibbs model is in close competition and often may perform better with a more useful confidence level. In fact Figure 7.11 highlight this by showing that the SSIM metric for the Gibbs kernel can outperform the Matérn model; just not on average. This may be because, in certain scenarios, the Gibbs kernel with its extra hyperparameters fails to converge to the best solution. Occasionally it will, and in those scenarios will capture more of the complex nature of the spatial dependency.

7.4.3 Precipitation (TMQ)

In this section we present the results of the Global study for the precipitation variable, TMQ. We start by considering the decomposition of the data.

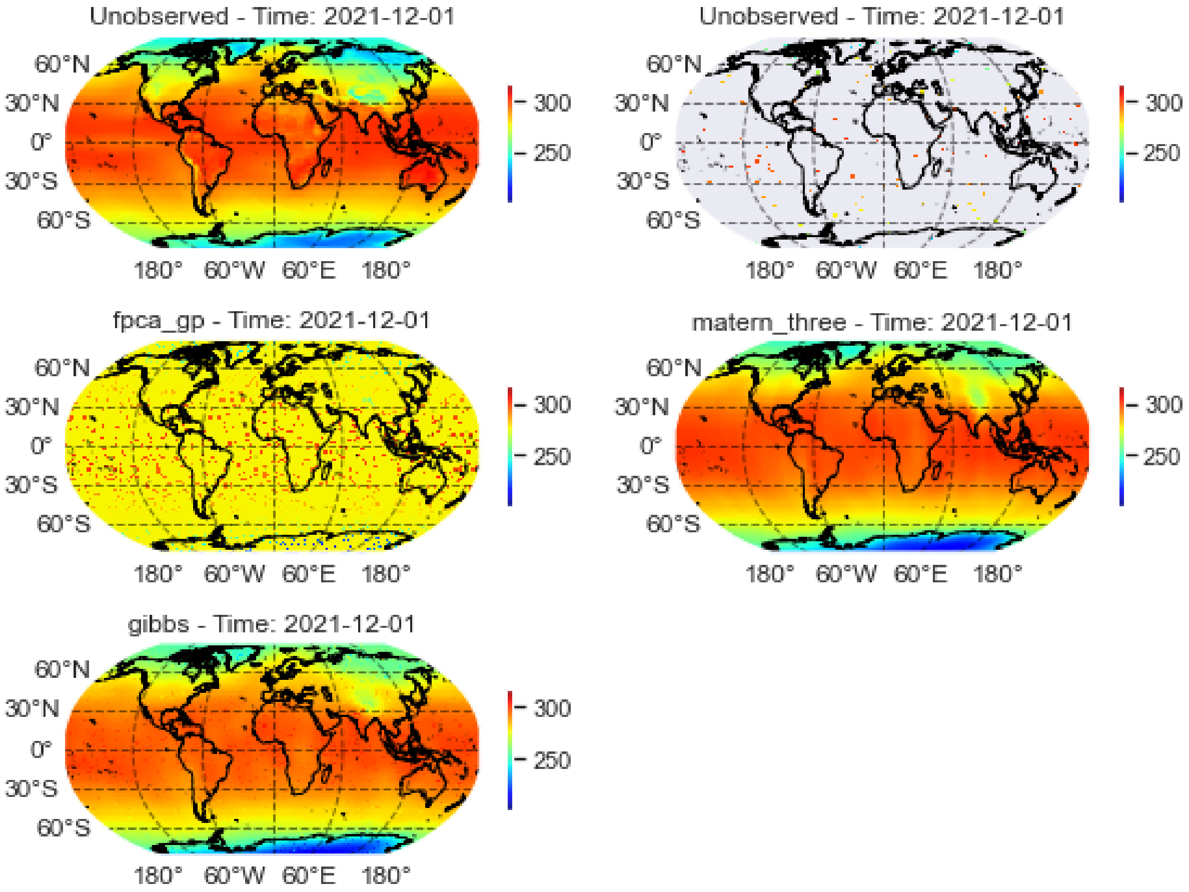

Unobserved - Time: 2021-12-01
Unobserved - Time: 2021-12-01
fpca_gp - Time: 2021-12-01
matern_three - Time: 2021-12-01
gibbs - Time: 2021-12-01
60°N
30°N
0°
30°S
60°S
180°
60°W
60°E
180°
300
250

Fig. 7.10 An indicative example of the CPACE model performance on reconstruction for the full globe of the CESM-LE TREFHT variable using the various models in the Global study. Clockwise from the top left we have the true data, the observed data at a particular time point, the Matérn Three model reconstruction, the Gibbs model reconstruction, and the White model reconstruction under the CPACE framework.

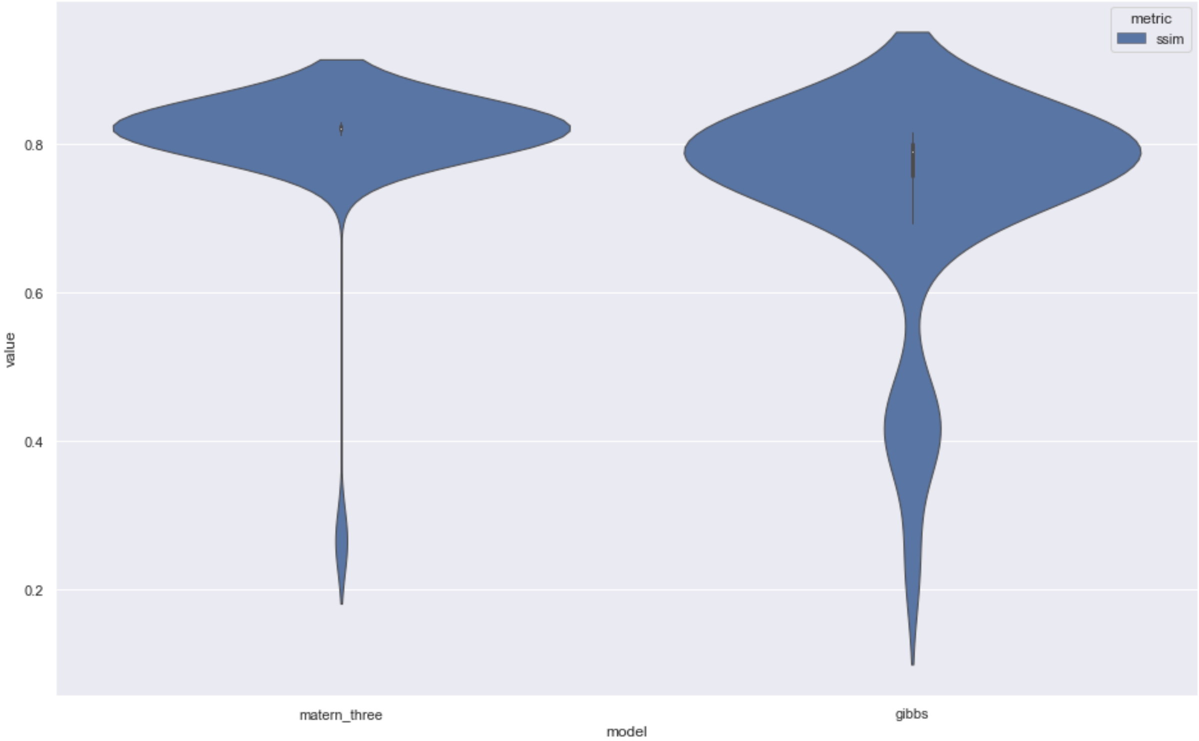
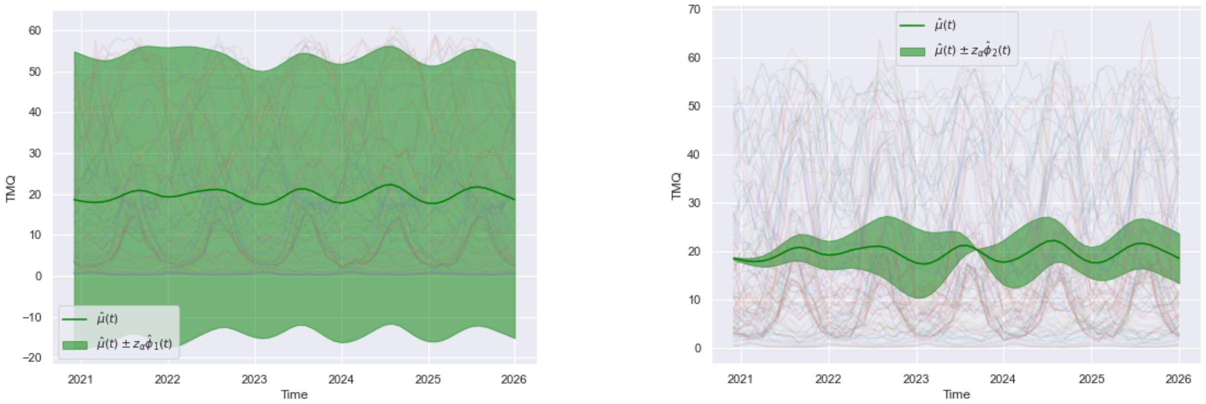


Fig. 7.11 Distribution plot of the SSIM metric for the Gibbs and Matérn kernel models for the TREFHT variable in the Global study.

One important aspect of both the PACE and CPACE methodologies is the eigen decomposition of the data common to both. Figure 7.12 shows the impact of the first two eigenfunctions in the FPCA decomposition. We can see from the first eigenfunction, Figure 7.12a, that this clearly represents a level shift in the functions. This is similar to the first eigenfunction from the study on the PS variable. This intuitively makes sense as we would expect different base precipitation levels for different areas of the globe. The second eigenfunction, Figure 7.12b, is perhaps a bit more complicated. This can be considered as an eigenfunction which will stretch the peaks and troughs of the mean function. These eigenfunctions are encouraging as they suggest that both the PACE and CPACE frameworks are appropriate for this dataset as they are outputting eigenfunctions with relatable interpretations. We note, as with the study on the PS variable, that the first eigenfunction is particularly dominating. This again is understandable as we can see that the main mode of variation comes from a level shift. However, it is disappointing that the second eigenfunction does not pick up the drastic changes in peaks that are present in some locations. This is most likely due to the sparsity of observations used for training, meaning it is not obviously present from the training data only.

Next we consider the ability for our models to reconstruct observed and unobserved data. Table 7.7 displays the metrics for the partially observed validation data reconstruction and Table 7.8 displays the metrics for the unobserved test data reconstruction. It is interesting to note that the Gibbs model provides best in class performance on the validation data. We can also see that the Matérn model performs best in class for the test data, while the Gibbs model performs close to best in class. The Gibbs model's test metrics have a much



(a) First eigenfunction.

(b) Second eigenfunction.

Fig. 7.12 The first two eigenfunctions impact on the mean function from the PACE and CPACE framework of the TMQ variable in the Global study. Here the shaded region shows the impact of two standard deviations from the mean function using the corresponding eigenfunction. Sample curves from the dataset are plotted for context.

Table 7.7 Results for reconstruction of the training data for the TMQ variable in the Global study from the CESM-LE dataset. Bold indicates best in class.

Model	**RMSE**	**MAE**
`pace`	6.3678 (0.1285)	4.5960 (0.0765)
`fpca_gp`	6.2901 (0.0902)	4.5786 (0.0779)
`matern_three`	5.5912 (0.2185)	4.1445 (0.1806)
`gibbs`	**5.1159 (0.2066)**	**3.7297 (0.1773)**

larger variance, and suggest that while it performs on average worse than the Matérn model it occasionally may produce better results. This is probably an effect of over fitting the complex non-stationary model which underpins the Gibb's model, whereas the Matérn model has a relatively simpler kernel, thus estimating its hyperparameters is less prone to error.

An illustration of the reconstruction ability from the temporal point of view for test data is given in Figure 7.13. It highlights the improvement that the spatial models can make, but also points out that we are not truly capturing the underlying functions. It is evident that we clearly miss the change in periodicity of the curves. Again, this may be difficult to pick up based on the sparsity of our observed data. The sparsity of observations

Table 7.8 Results for reconstruction of the test data for the TMQ variable in the Global study from the CESM-LE dataset. Bold indicates best in class.

Model	**RMSE**	**MAE**
`pace`	15.8224 (0.0925)	13.0445 (0.0824)
`fpca_gp`	15.7821 (0.0643)	13.0326 (0.0760)
`matern_three`	**6.1061 (0.0778)**	**4.4798 (0.0522)**
`gibbs`	6.3401 (0.0934)	4.7312 (0.0715)

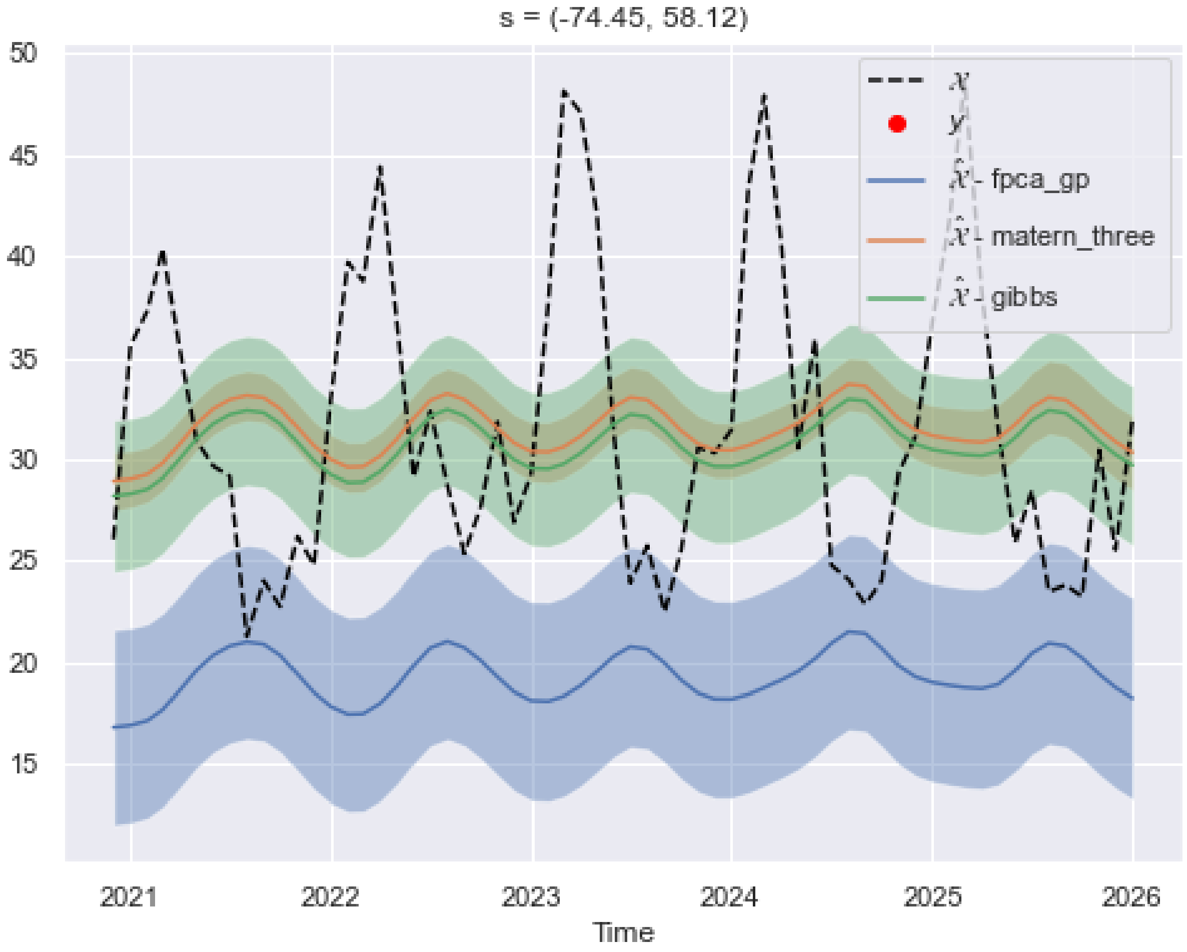


Fig. 7.13 An indicative example of the CPACE model performance on reconstruction of the CESM-LE TMQ variable in the Global study from the test dataset.

can be seen from Figure 7.14 which gives an illustration of the reconstruction ability spatially. This figure also highlights the discrepancy between the Gibbs and the Matérn model. We can see that although the Gibbs model has some areas of correct structure it fails to capture fully the data across the whole domain, whereas the Matérn model achieves a much better global structure.

7.4.4 Wind Speed (U10)

We present the results of the Global study for the wind speed variable, U10. Similar to the other global studies we focus on the metrics for the validation and test datasets.

First, the validation metrics. These are displayed in Table 7.9. We can see that on the validation data the models perform similarly well, with the Matérn and Gibbs models performing best in class. However, all models seem to have similar reconstruction ability. We can see an illustration of the relative performances of the models on validation data in Figure 7.15. This also identifies that we have again missed the periodicity in the model. Similar to the previous variables this is down to the models focusing on the level shift as their main mode of variation.

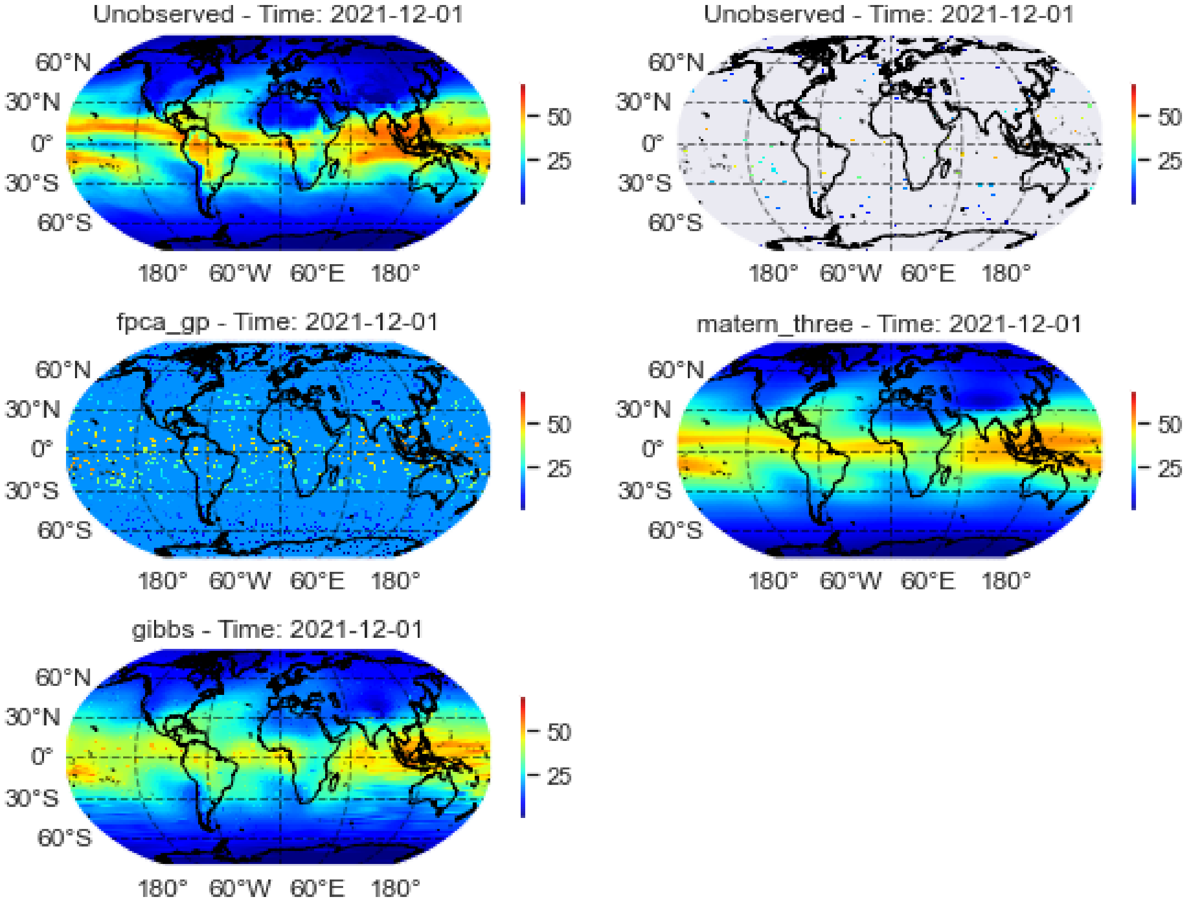

Unobserved - Time: 2021-12-01
Unobserved - Time: 2021-12-01
fpca_gp - Time: 2021-12-01
matern_three - Time: 2021-12-01
gibbs - Time: 2021-12-01
60°N
30°N
0°
30°S
60°S
180°
60°W
60°E
180°
50
25

Fig. 7.14 An indicative example of the CPACE model performance on reconstruction for the full globe of the CESM-LE TMQ variable using the various models in the Global study. Clockwise from the top left we have the true data, the observed data at a particular time point, the Matérn Three model reconstruction, the Gibbs model reconstruction, and the White model reconstruction under the CPACE framework.

Table 7.9 Results for reconstruction of the validation data for the U10 variable in the Global study from the CESM-LE dataset. Bold indicates best in class.

Model	RMSE	MAE
pace	1.2952 (0.0153)	0.9343 (0.0163)
fpca_gp	1.2879 (0.0121)	0.9306 (0.0120)
matern_three	**1.2659 (0.0106)**	0.9343 (0.0107)
gibbs	1.2761 (0.0135)	**0.9215 (0.0095)**

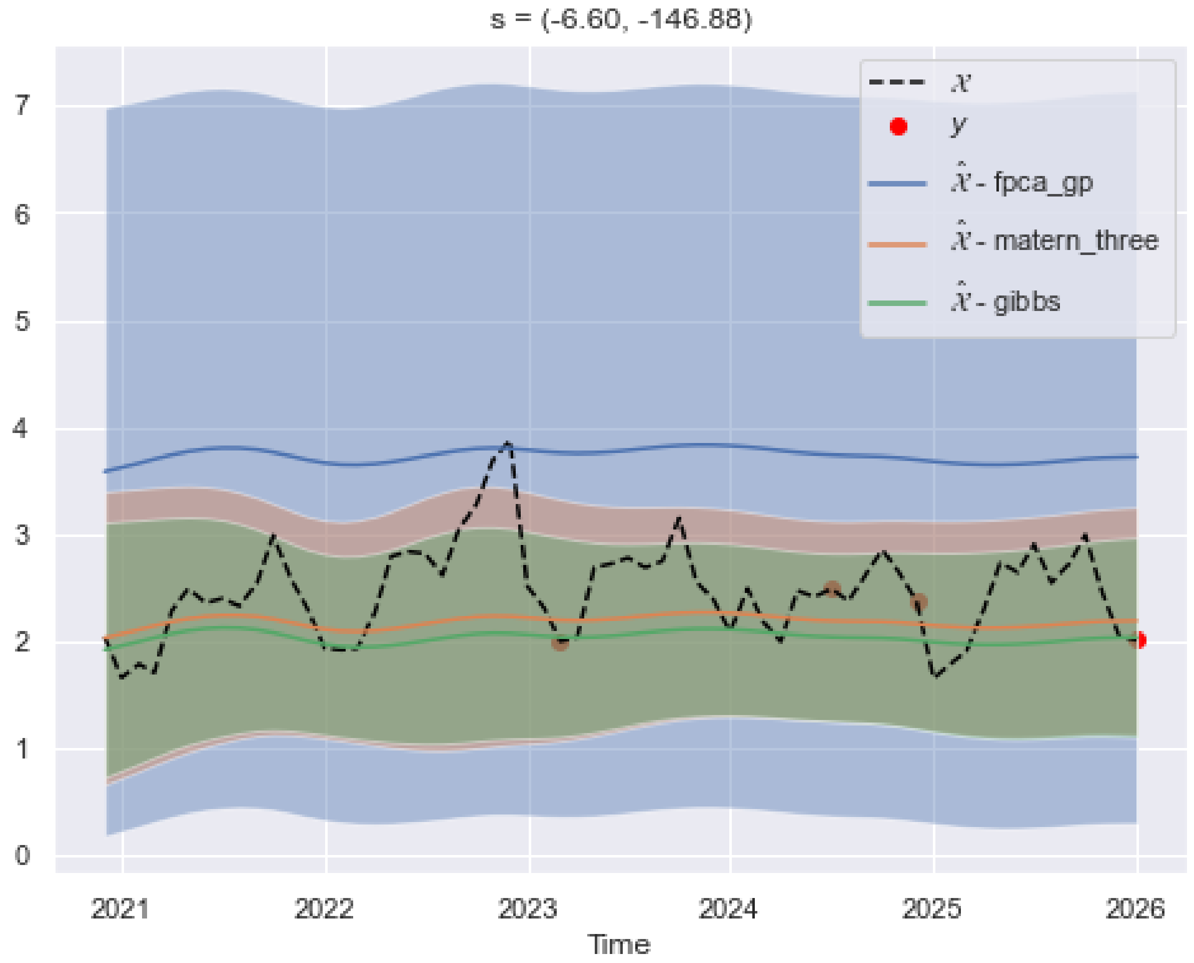


Fig. 7.15 An indicative example of the CPACE model performance on reconstruction of the CESM-LE U10 variable in the Global study from the validation dataset.

Table 7.10 Results for reconstruction of the training data for the U10 variable in the Global study from the CESM-LE dataset. Bold indicates best in class.

Model	RMSE	MAE
`pace`	3.0124 (0.0107)	2.5004 (0.0103)
`fpca_gp`	3.0098 (0.0116)	2.4991 (0.0103)
`matern_three`	**1.4713 (0.0129)**	**1.0965 (0.0128)**
`gibbs`	1.5642 (0.0318)	1.2051 (0.0125)

Table 7.11 Results for reconstruction of the test data for the U10 variable in the Global study from the CESM-LE dataset. Bold indicates best in class.

Model	PSNR	SSIM
`pace`	13.7495 (0.0743)	0.1699 (0.0021)
`fpca_gp`	13.7673 (0.0669)	0.1703 (0.0021)
`matern_three`	**19.8017 (0.0731)**	**0.5988 (0.0092)**
`gibbs`	19.2793 (0.1095)	0.5230 (0.0089)

We next consider the test data performance, and display the metrics for this in Table 7.10. We can see a clear distinction in the test metrics in preference to the CPACE models with the spatial kernels. It seems that the simpler Matérn model is performing best overall. However, the distinction between that and the Gibbs model is small from these metrics. Considering our image reconstruction metrics, we can see a clearer preference to the Matérn model, see Table 7.11. It is interesting to note that the Gibbs model can outperform that of the Matérn model in certain situations. We can see this by looking at the distribution of the SSIM metric over the simulations present in the CESM-LE dataset for the Global study. We show this in Figure 7.16, where the distribution of the `gibbs` model metrics has a higher peak SSIM. However, over all simulations, we also see a larger range which is on average lower than the `matern_three` model. Hence, the poorer performance on average.

Finally, we provide an indicative reconstruction over both a spatial view, see Figure 7.18, and a temporal view, see Figure 7.17. We can see from these that we have some limitation in our ability to reconstruct the periodic components of the functional observations. As mentioned before, this is likely because we just don't have enough evidence from our training data to suggest such periodic components in our FPCA decomposition which all models use.

7.4.5 Discussion

In Sections 7.4.1 - 7.4.4 we have considered an application of the CPACE model to the Global study of the CESM-LE data. We have presented results which show clearly the advantage of using the CPACE framework with spatial kernels. This is intuitive as the geographic nature of this dataset clearly should have spatial dependency for all of these variables.

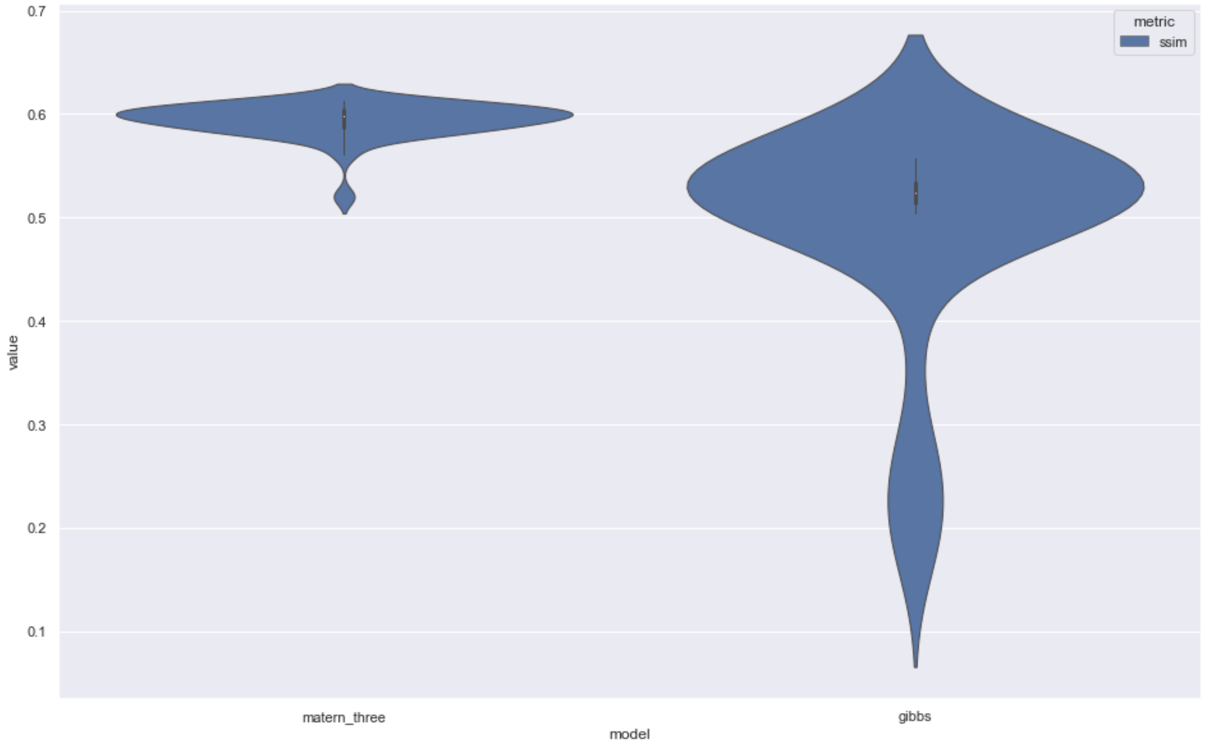


Fig. 7.16 Distribution plot of the SSIM metric for the Gibbs and Matérn kernel models for the U10 variable from the Global study.

Our results show that typically the Matérn model tends to perform the best of our models considered. This is intriguing since we may expect the non-stationary component of the Gibbs model to pick out more complex spatial dependencies. We suppose the reason for the lack of this is twofold. Firstly; on the global scale, with our reduced resolution we may indeed mask a lot of these complexities, since the variation across large spatial scales dwarfs the subtleties of local variation. Secondly; our challenge of using sparse training data for this study may have meant that we often exclude a lot of intricate variation by simply not having enough evidence in the training datasets. This theory is backed up by the fact that in most cases the Gibbs kernel tends itself towards behaving like the Matérn kernel. This is comforting as it suggest that rather than the Gibbs model failing to identify these trends, it is actually that our training data only suggests large scale variation.

We also note that in the study across the variables our eigen decomposition, which makes the base of all these models, tended to be dominated by a single large component. This represented a level shift of the functional data. The CPACE models with a spatial kernel tend to outperform the other as they use spatial information to inform on where to apply this level shift. This leads to a good increase in performance. Again, this intuitively makes sense which gives us comfort in the feasibility of the CPACE framework being applied to EO data as it produces interpretable results.

We note that even in the non-spatial dependent kernels the CPACE framework tends to outperform the PACE framework. Again, attributable to the additional refinement of the model's hyperparameters compared to the PACE model.

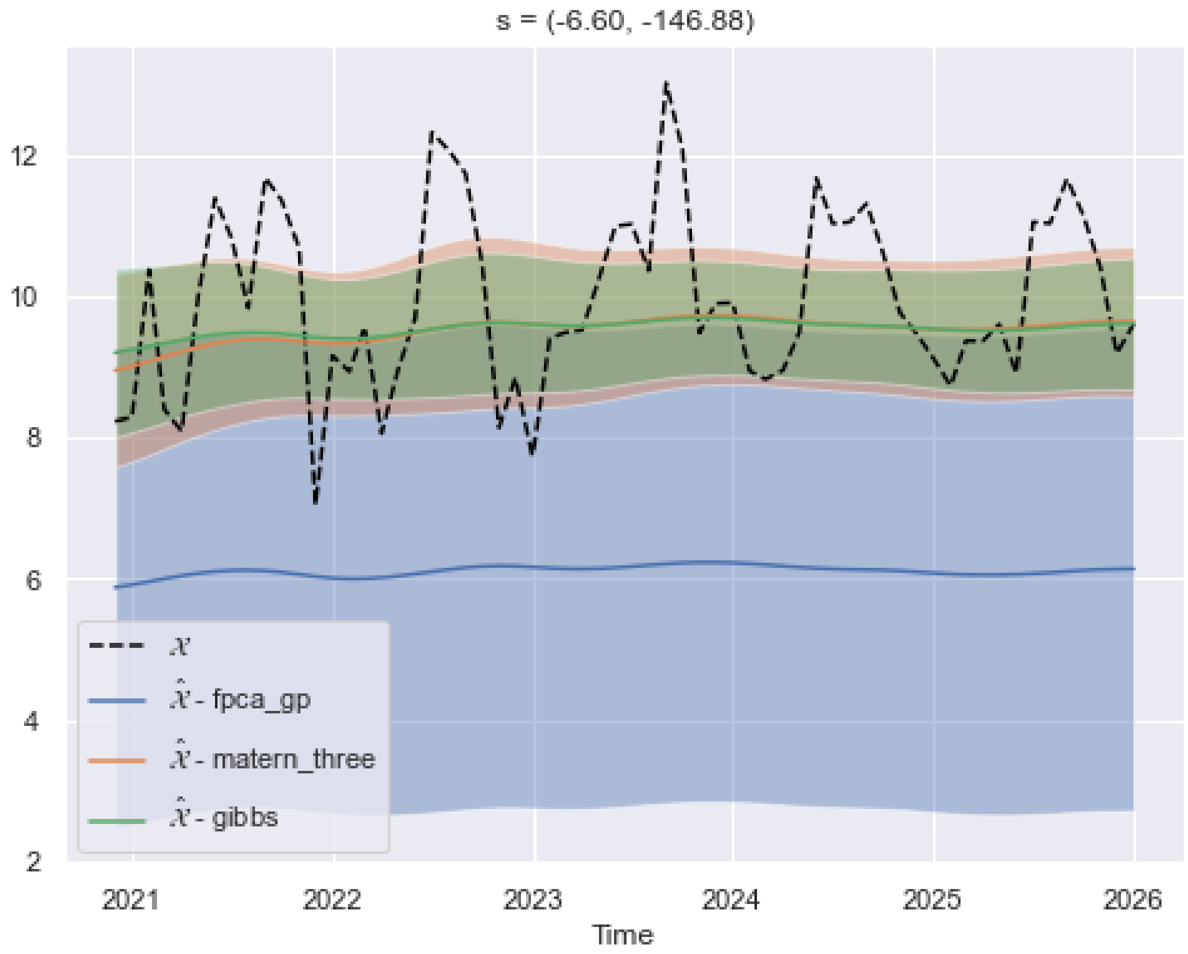


Fig. 7.17 An indicative example of the CPACE model performance on reconstruction of the CESM-LE U10 variable in the Global study from the test dataset.

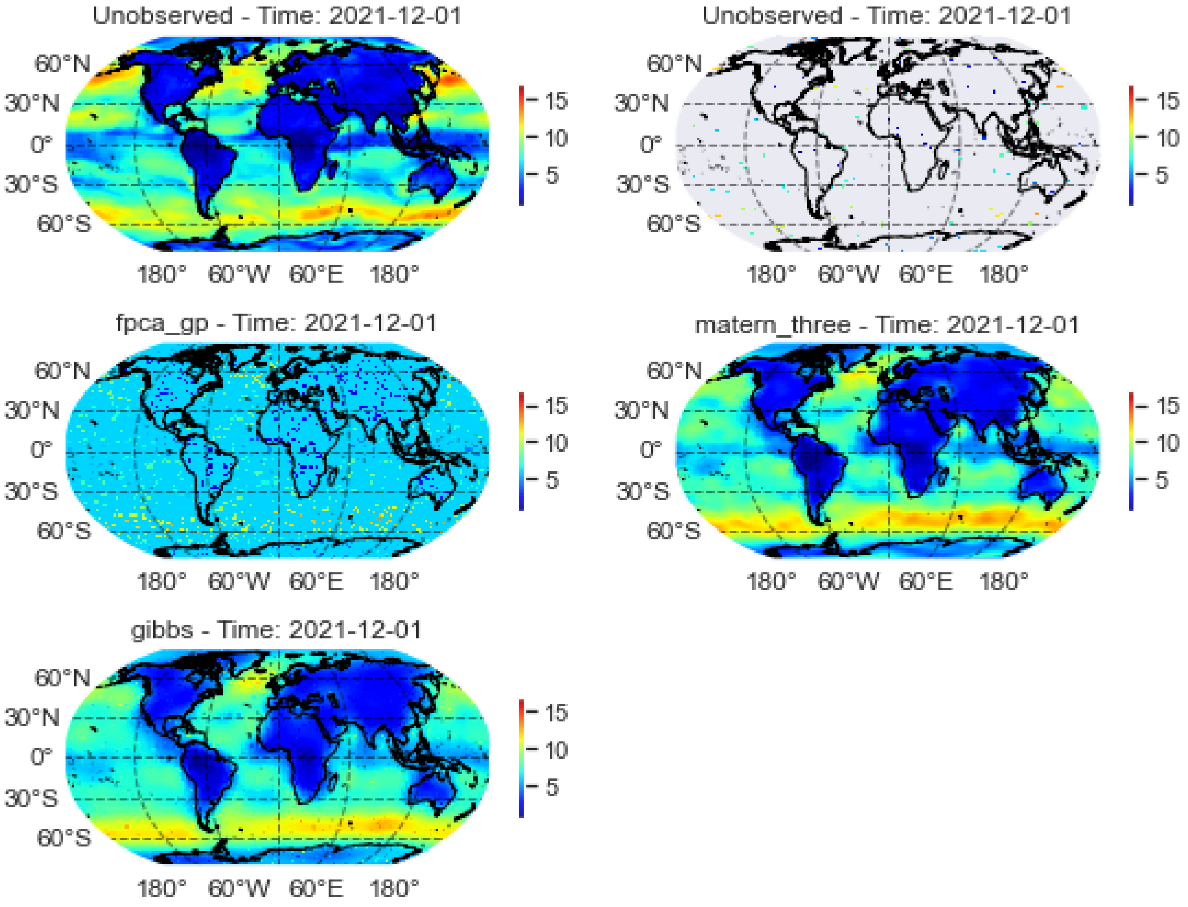

Unobserved - Time: 2021-12-01
Unobserved - Time: 2021-12-01
fpca_gp - Time: 2021-12-01
matern_three - Time: 2021-12-01
gibbs - Time: 2021-12-01
60°N
30°N
0°
30°S
60°S
180°
60°W
60°E
180°
15
10
5

Fig. 7.18 An indicative example of the CPACE model performance on reconstruction for the full globe of the CESM-LE U10 variable using the various models in the Global study. Clockwise from the top left we have the true data, the observed data at a particular time point, the Matérn Three model reconstruction, the Gibbs model reconstruction, and the White model reconstruction under the CPACE framework.

In the following section we move on to consider our European study. This more localised European study, presented in Section 7.1.2, will consider a slightly different challenge to the CPACE framework with less sparse data and less global variation.

7.5 European Study Results

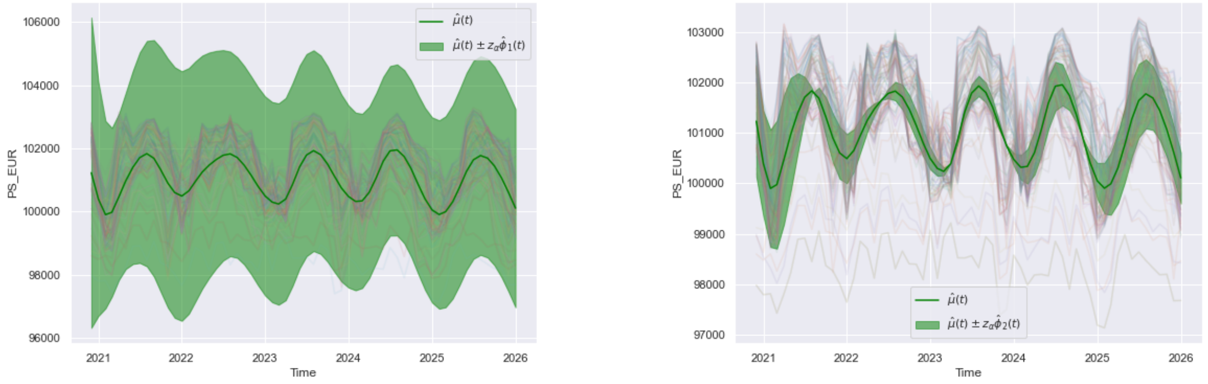
Here we present the results of the study across the spatial domain as described in Section 7.1.2. This roughly corresponds to the European continent. This study has two components that differ from the Global study; namely the spatial scale and the density of observations. This gives us a chance to see how well the CPACE framework compares to the PACE framework under these scenarios.

We have considered four models for this study, the same setup for each variable of interest. The `fpca` model corresponds to the FPCA model or PACE framework, which does not take into account spatial dependency. The second, `fpca_gp`, is our CPACE model with the White kernel. Again this does not take into account spatial dependency between functional observations, but is computed under our CPACE framework which allows for hyperparameters of the kernel; namely the spatial kernel variance for each component, to be estimated using the Gaussian process framework. Thirdly, we use the Matérn Three kernel with anisotropic length scales as the spatial kernel in another CPACE model. This we denote by `matern_three`. Finally, we denote by `gibbs` the CPACE model using the Gibbs kernel. For each of these models we use 5 components in our decomposition of the observed data and in the CPACE framework.

We present the results separately for each variable of interest; Pressure (PS), Temperature (TREFHT), Precipitation (TMQ), and Wind Speed (U10).

7.5.1 Pressure (PS)

Here we present the results from our models applied to the pressure variable in the European study. We start by looking at the eigen decomposition common to all models. Figure 7.19 displays the impact of the first two eigenfunctions under this study for a single simulation of the PS variable from the CESM-LE dataset. It is of note that again the first eigenfunction, Figure 7.19a, represents a level shift; as was seen in the Global study of the pressure variable. However, the second eigenfunction, Figure 7.19b, shows this time a stretching of the peaks and troughs from the mean function. It is perhaps slightly more complicated than that, showing more variation at the start and end of the domain, but in general that is a fair interpretation of the second eigenfunction. This is encouraging as intuitively this is an important part of the dataset and highlights the ability of the PACE methodology to help in understanding the data, something which the CPACE methodology builds on. It is also interesting to note that we obtain a much more representative mean function than that of the Global study. This is a result of the denser observations and the smaller spatial scale in this study.



(a) First eigenfunction.

(b) Second eigenfunction.

Fig. 7.19 The first two eigenfunctions impact on the mean function from the PACE and CPACE framework for the PS variable in the European study. Here the shaded region shows the impact of two standard deviations from the mean function using the corresponding eigenfunction. Sample curves from the dataset are plotted for context.

Table 7.12 Results for reconstruction of the validation data for the PS variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	**RMSE**	**MAE**
`pace`	579.4300 (47.2497)	438.0766 (28.8890)
`fpca_gp`	562.3507 (38.1133)	431.3178 (27.2087)
`matern_three`	560.6262 (41.2007)	**423.1480 (27.5427)**
`gibbs`	**559.1836 (36.0552)**	424.4744 (25.5373)

Next, we present the metrics of the predictions from our variety of models. We start by showing the resultant metrics on our validation dataset in Table 7.12. Again, similar to the Global study we see a close comparative performance between all CPACE models. We do not see the differentiation between the `pace` model and the CPACE models that was seen in the Global study.

All models perform roughly equally on the validation dataset. Perhaps we can argue that the spatial models perform slightly better than the non-spatial models, especially evident on the MAE metric. An indicative example of model reconstruction is given in Figure 7.20. It highlights how similar the models predict partially observed functions.

Next we consider the performance on the test dataset, that is completely unobserved functional data. We present the metric results for this in Table 7.13. As can be seen, we see a distinct improvement in using the CPACE framework with a spatial kernel. The Matérn kernel performs best in class. The Gibbs kernel unfortunately suffers from occasional bad prediction accuracy on some simulations leading to higher metric results with larger variance. This is probably due to the added complexity in estimating the kernel hyperparameters. We discuss our approach to minimising this effect in Chapter 8.

An example reconstruction from these models for functional data from the test set is displayed in Figure 7.21. This shows how the spatial models can adapt using the location

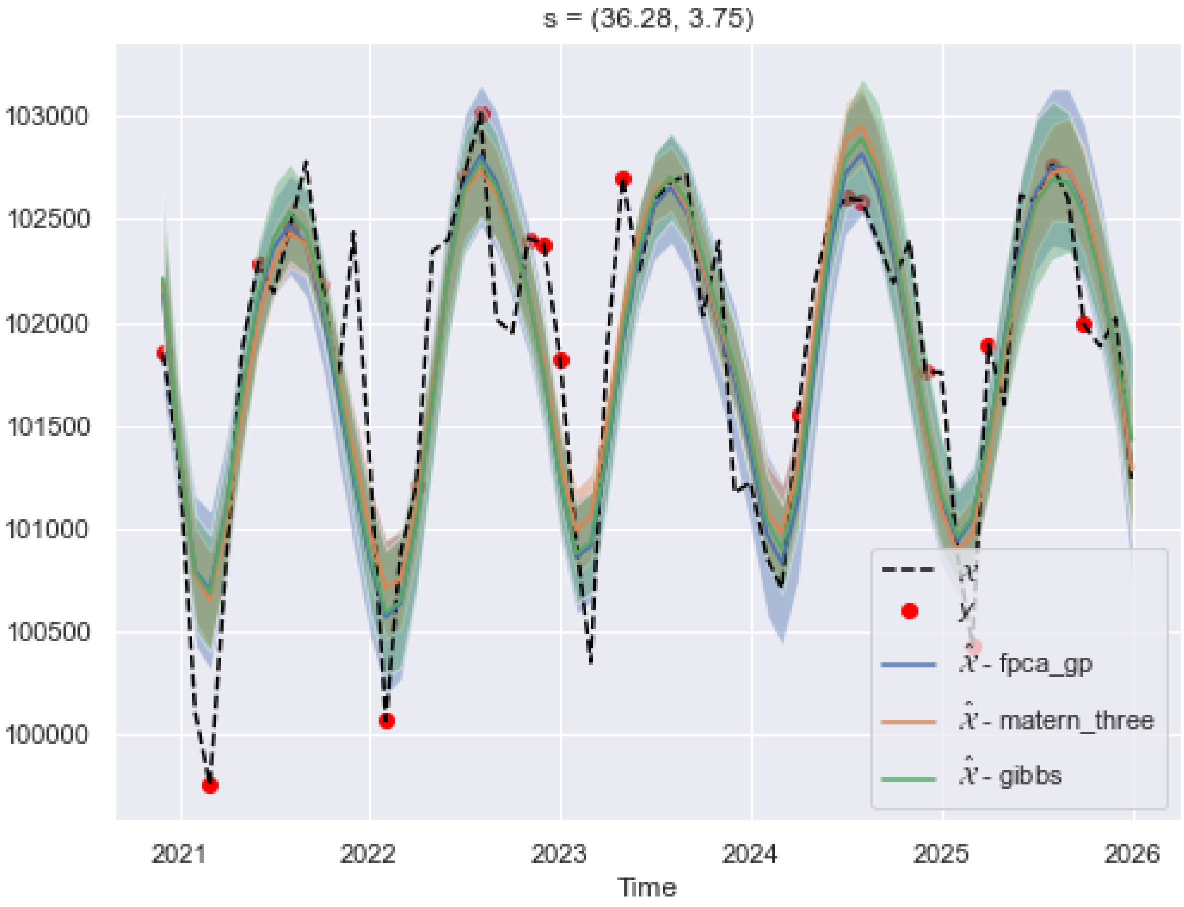


Fig. 7.20 An indicative example of the CPACE model performance on reconstruction of the CESM-LE PS variable in the European study from the validation dataset.

Table 7.13 Results for reconstruction of the test data for the PS variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	**RMSE**	**MAE**
`pace`	1297.1985 (29.4383)	760.5791 (12.6591)
`fpca_gp`	1297.1985 (29.4383)	760.5791 (12.6591)
`matern_three`	**667.1006 (64.2558)**	**465.4720 (24.7951)**
`gibbs`	942.3990 (125.7554)	532.5312 (25.7204)

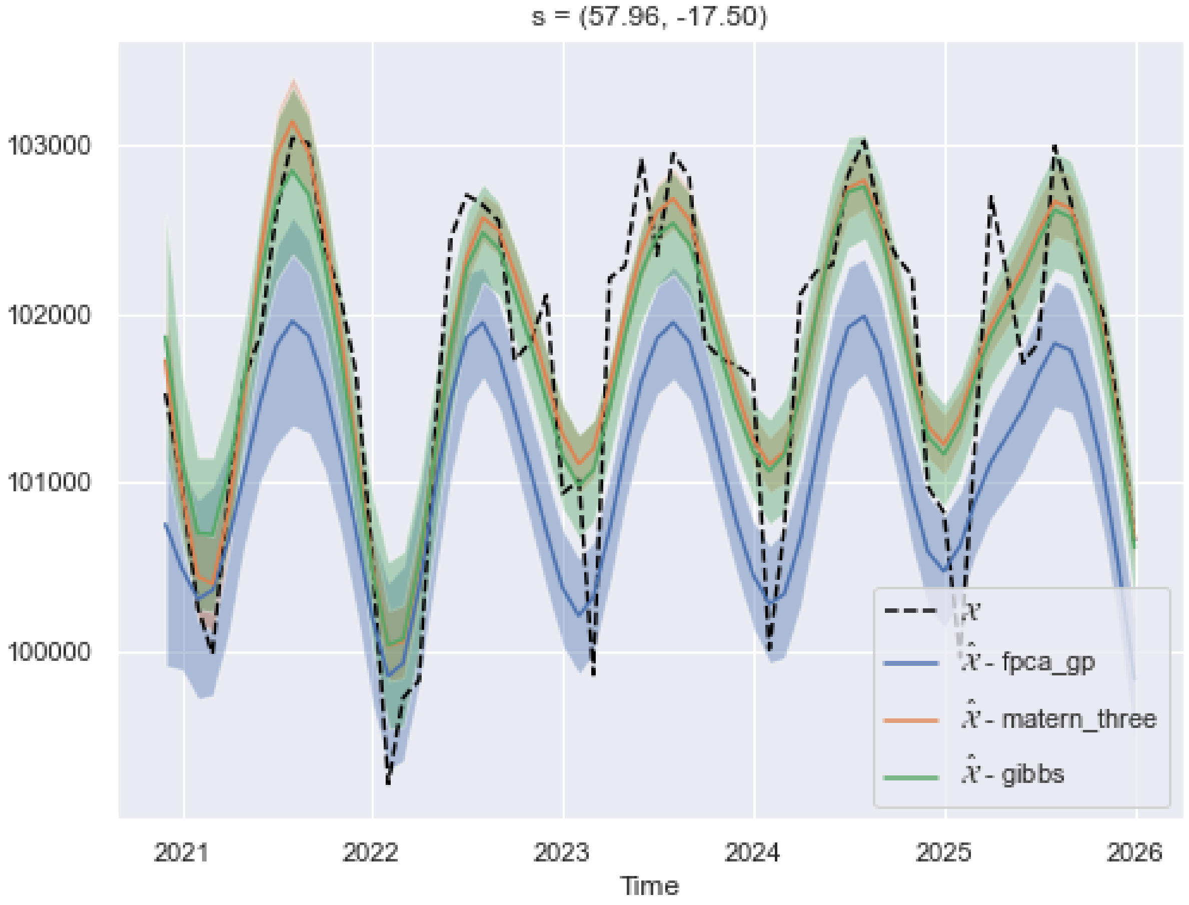


Fig. 7.21 An indicative example of the CPACE model performance on reconstruction of the CESM-LE PS variable in the European study from the test dataset.

of the spatial domain, whereas the non-spatial dependent kernel will just predict the mean function.

Finally, we consider the image reconstruction metrics. These consider how well the models perform to recreate the imagery as a whole. This is presented in Table 7.14. Again, we can see the better performance of the spatial kernels, with the Matérn kernel quite expectedly being best in class. Interestingly, the performance increase over the non-spatial kernel models is less pronounced than in the Global study. This is a function of the less varied data and the increase in observations, which gives the non-spatial models more points at which the functions have some observations, which is where they perform best.

Table 7.14 Results for reconstruction of the full data for the PS variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	PSNR	SSIM
pace	21.6920 (0.1913)	0.7137 (0.0121)
fpca_gp	21.6992 (0.1913)	0.7246 (0.0086)
matern_three	**27.2527 (0.8316)**	**0.8947 (0.0130)**
gibbs	23.8517 (1.4678)	0.8655 (0.0168)

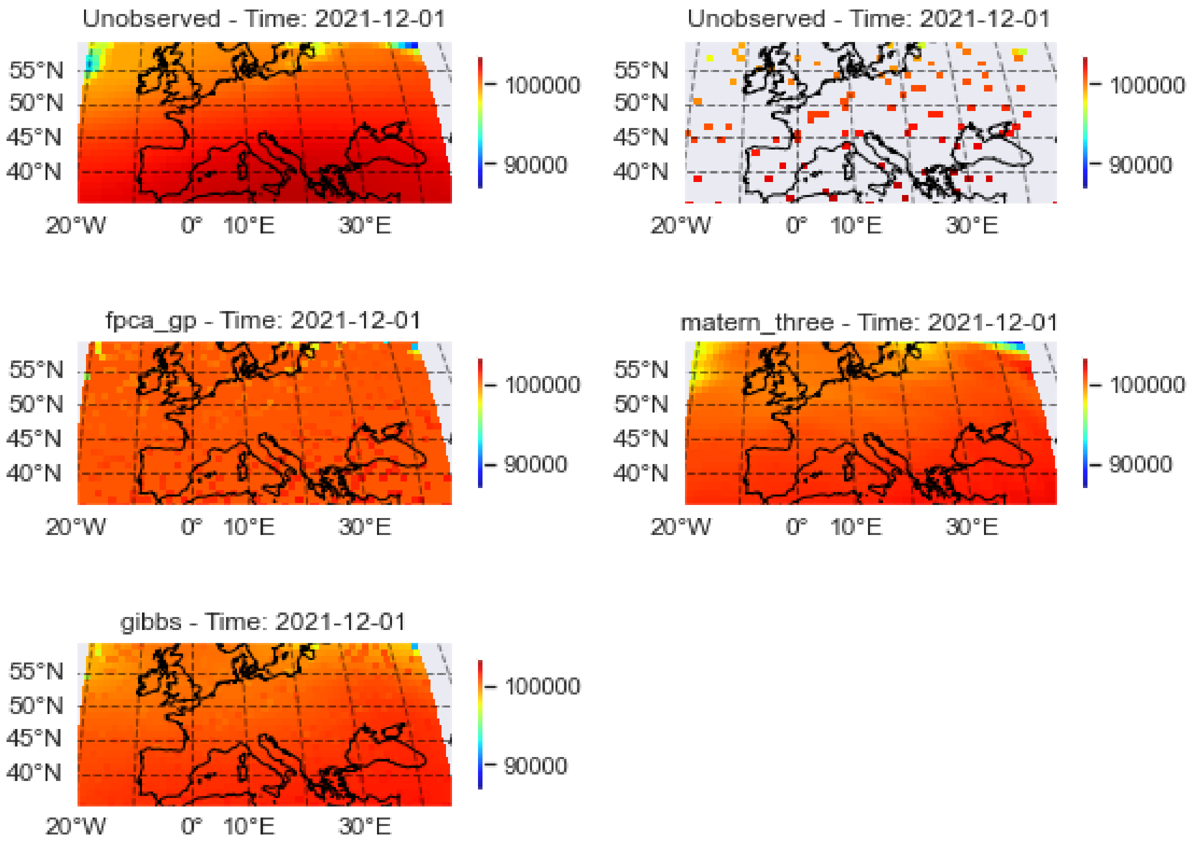

Unobserved - Time: 2021-12-01
Unobserved - Time: 2021-12-01
fpca_gp - Time: 2021-12-01
matern_three - Time: 2021-12-01
gibbs - Time: 2021-12-01
55°N
50°N
45°N
40°N
20°W
0°
10°E
30°E
100000
90000

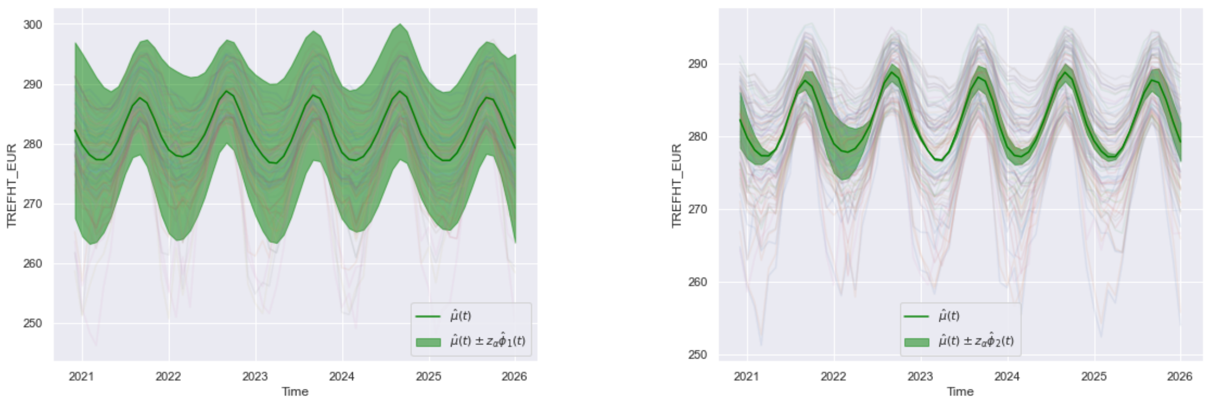
Fig. 7.22 An indicative example of the CPACE model performance on reconstruction for the European study of the CESM-LE PS variable using the various models in the European study. Clockwise from the top left we have the true data, the observed data at a particular time point, the Matérn Three model reconstruction, the Gibbs model reconstruction, and the White model reconstruction under the CPACE framework.

From the above, we can see clear evidence that the CPACE framework is effective for the pressure variable. We can see that it replicates the PACE framework as the `fpca_gp` model performs almost identically to that of the `pace` model. We can see, alike in the corresponding Global study, that the Matérn model is generally best in class, with the Gibbs model being a close second. It is encouraging to see that the performance of the CPACE framework is not particularly affected by the more dense observations, which suggests it is applicable in this setting. Figure 7.22 gives a example illustration of the model reconstructions at a particular time point in our dataset. It confirms, in an illustrative way, the advantage of the CPACE framework.

7.5.2 Temperature (TREFHT)

In this section we present the results of the European study for the temperature variable. Alike Section 7.5.1, we begin by considering the eigen decompostion of the dataset which is common to all models under the PACE and CPACE frameworks.

Figure 7.23 shows the impact of the leading two eigenfunction in the decomposition for an example simulation. Similar to the pressure variable, the first eigenfunction is clearly a level shift of the mean function, with the second being a stretching of the peaks and



(a) First eigenfunction.

(b) Second eigenfunction.

Fig. 7.23 The first two eigenfunctions impact on the mean function from the PACE and CPACE framework for the TREFHT variable in the European study. Here the shaded region shows the impact of two standard deviations from the mean function using the corresponding eigenfunction. Sample curves from the dataset are plotted for context.

Table 7.15 Results for reconstruction of the validation data for the TREFHT variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	**RMSE**	**MAE**
pace	1.7037 (0.0817)	1.0890 (0.0586)
fpca_gp	1.6692 (0.0653)	1.0855 (0.0464)
matern_three	**1.6115 (0.0767)**	**1.0348 (0.0593)**
gibbs	1.6522 (0.0865)	1.0662 (0.0552)

troughs. Again, this is encouraging, as they are intuitively reasonable leading modes of variation. It is reasonable to assume that as we move south the contribution of the first will be to shift the mean function higher to represent the effect of lower latitude on the overall temperature across time.

We now move on to consider how the models perform on the validation data. Table 7.15 presents these results. We see similar results for all models, and there is only slight noticeable improvements from the spatial kernels. This is expected as the PACE and CPACE methodology will utilise the observed values to inform the predicted function. It so happens in this study that the CPACE models don't gain much additional information from neighbouring curves.

We now consider the comparative performance on the test set. Here we may expect that utilising spatial information will give a good performance boost to the CPACE framework with spatial kernels. Table 7.16 displays these results.

In fact we do see such an improvement, which is an encouraging sign that the CPACE framework is applicable to this smaller spatial scale of the European study. As an illustrative example we display a reconstruction for a test curve from this dataset in Figure 7.24. It is interesting to note here, that the confidence interval for the Gibbs kernel is much more appropriate than the Matérn model. This might suggest that while the

Table 7.16 Results for reconstruction of the test data for the TREFHT variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	RMSE	MAE
`pace`	5.0376 (0.1367)	3.9641 (0.1143)
`fpca_gp`	5.0376 (0.1367)	3.9641 (0.1143)
`matern_three`	**1.7362 (0.0671)**	**1.0726 (0.0228)**
`gibbs`	1.9959 (0.0535)	1.2047 (0.0456)

Table 7.17 Results for reconstruction of the full data for the TREFHT variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	PSNR	SSIM
`pace`	15.3612 (0.1563)	0.3402 (0.0046)
`fpca_gp`	15.3692 (0.1543)	0.3426 (0.0072)
`matern_three`	**24.1329 (0.1560)**	**0.9188 (0.0035)**
`gibbs`	23.0612 (0.1784)	0.8957 (0.0047)

Matérn model performs best in mean, the Gibbs model actually predicts with a more realistic confidence interval.

Next we compare the results for full reconstruction of the dataset using our image reconstruction metrics. These are displayed in Table 7.17. Again we see the Matérn model performing best in class, but the Gibbs and Matérn model perform similarly. We note, as expected, a good improvement over the PACE framework using spatial dependent kernels. An illustration of the effect is given in Figure 7.25. We see good reconstruction overall in this example, but unfortunately all models fail to capture the more extreme areas of the study, such as the most north-westerly locations. The spatial kernels tend to over smooth this area and therefore under predict the values of the data in this location. We might expect the Gibbs kernel to be able to predict such spatial variation, however it seems in this study that the Gibbs model prefers to closely align with the Matérn model. We reason this will be because there is insufficient evidence in the training data to suggest such non-stationary dependency.

7.5.3 Precipitation (TMQ)

In this section we present the results of the European study for the Precipitation variable. Alike Section 7.5.2, we begin by considering the eigen decompostion of the dataset which is common to all models under the PACE and CPACE frameworks.

As with the study of other variables we display the first two eigenfunctions impact on the estimate mean function for the TMQ variable. This can be seen in Figure 7.26. Two things are displayed in this.

First, compared to the Global study of the same variable in Section 7.4.3, we see a stronger estimation of the mean function. That is for two reasons; the density of observations over space, and the frequency of observations across time is higher in the

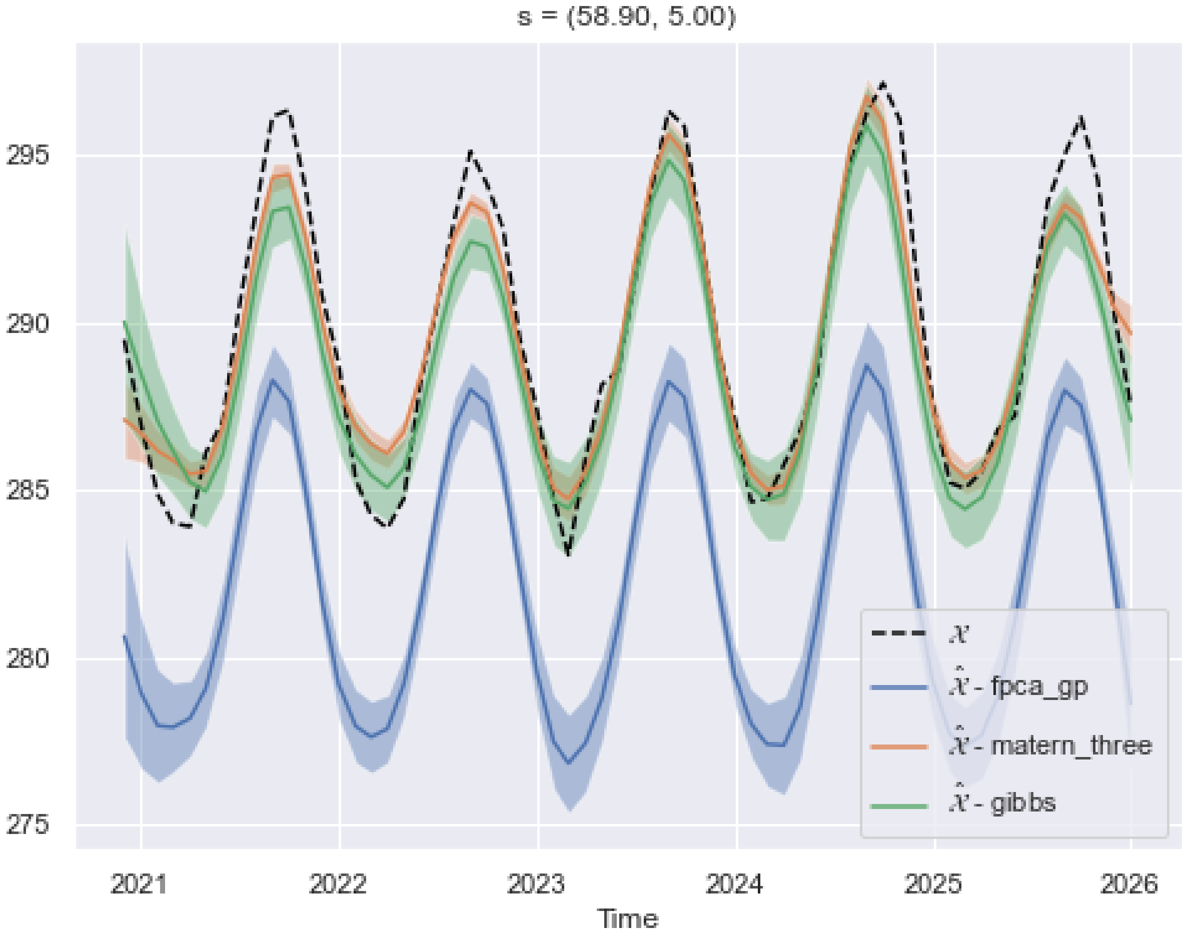


Fig. 7.24 An indicative example of the CPACE model performance on reconstruction of the CESM-LE TREFHT variable in the European study from the test dataset.

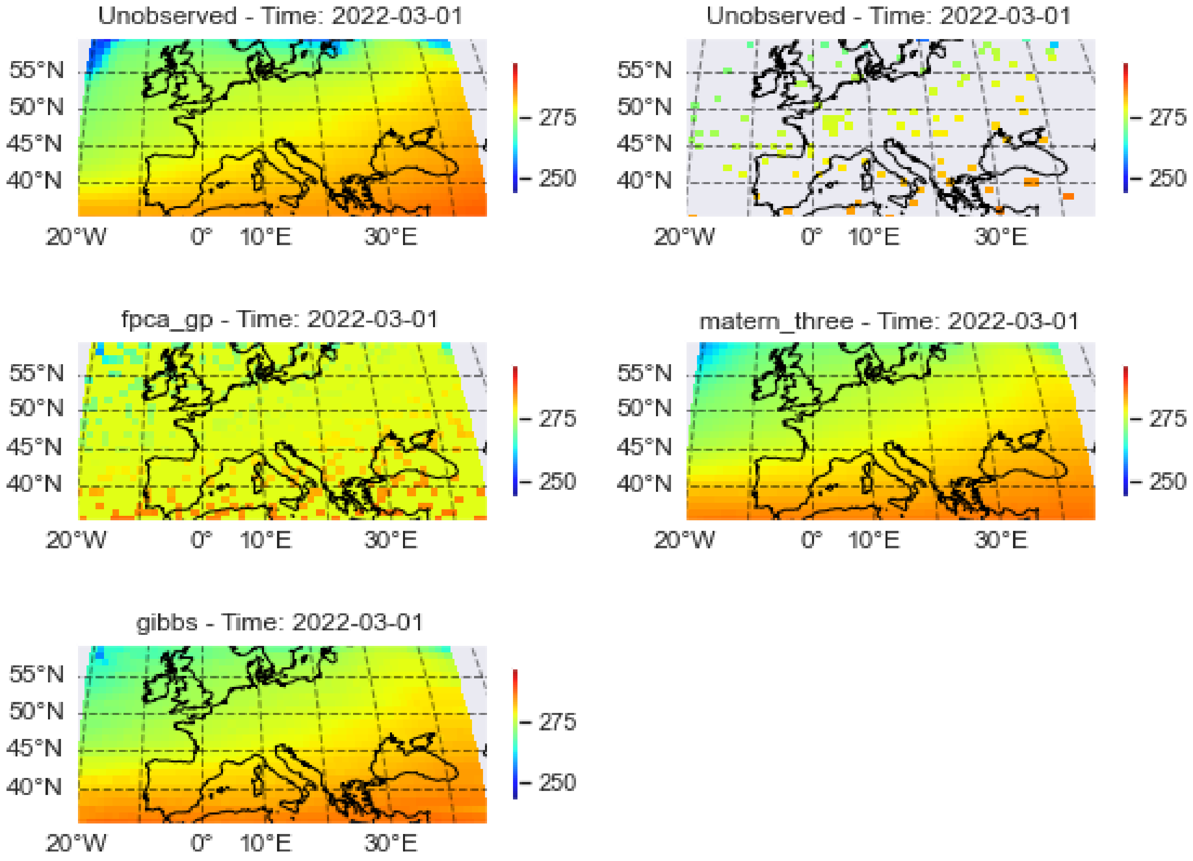
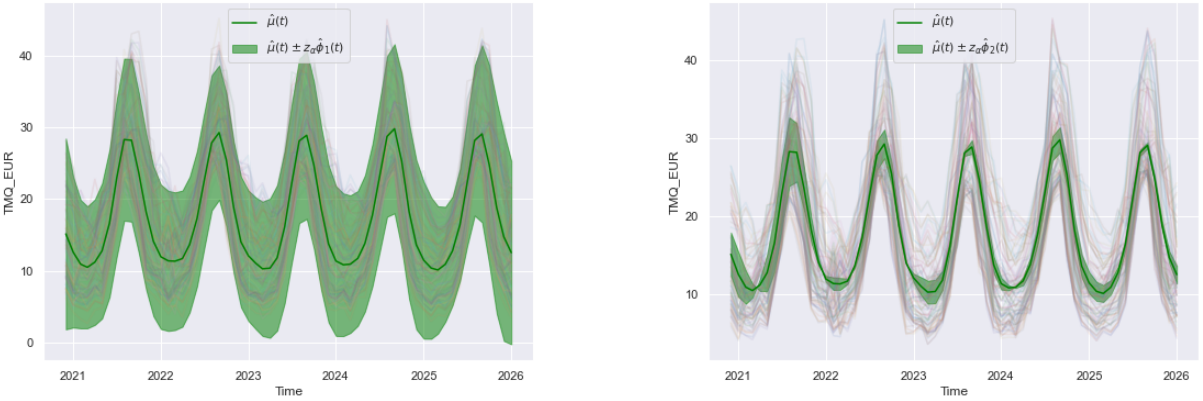

Unobserved - Time: 2022-03-01
Unobserved - Time: 2022-03-01
fpca_gp - Time: 2022-03-01
matern_three - Time: 2022-03-01
gibbs - Time: 2022-03-01
55°N
50°N
45°N
40°N
20°W
0°
10°E
30°E
275
250

Fig. 7.25 An indicative example of the CPACE model performance on reconstruction for the European study of the CESM-LE TREFHT variable using the various models in the European study. Clockwise from the top left we have the true data, the observed data at a particular time point, the Matérn Three model reconstruction, the Gibbs model reconstruction, and the White model reconstruction under the CPACE framework.



(a) First eigenfunction.

(b) Second eigenfunction.

Fig. 7.26 The first two eigenfunctions impact on the mean function from the PACE and CPACE decomposition framework for the TMQ variable in the European study. Here the shaded region shows the impact of two standard deviations from the mean function using the corresponding eigenfunction. Sample curves from the dataset are plotted for context.

European study compared to the Global study, by construction. The mean estimator we have used, described in Section 5.3 and Theorem 5.1, will converge to the true mean given more observations. We also have that the variability across space is less in the European study than that of the Global study. This means that functions tend to be closer to the mean function, and so our estimate is often less skewed by unusual functions in this study. One important point is that in the Global study we will tend to have two sets of functions with alternate periods. These correspond to the functions from the two hemispheres of the globe respectively. This added variability is not present in the European study.

Second, in addition to the level shift of the first eigenfunction, in Figure 7.26a, the second eigenfunction captures the variability in the peaks and troughs. This is similar to the other European studies. Again this provides confidence in the PACE and CPACE methodologies as they seem to pull out natural eigenfunctions.

Next, we summarise the various models ability to reconstruct the validation dataset. This is given in Table 7.18. As can be seen, and alike other variable for the European study, both the PACE and CPACE models perform comparatively. The distribution of these metrics is displayed in Figure 7.27, which highlights why we have chosen the Matérn kernel as best in class. This is encouraging from the CPACE models, since it highlights we do not lose any predictive ability using this Gaussian process framework. Equally, the same reasoning applies as to why we do not see much improvement using the spatial kernels from the other European studies, namely the added density of observations means each function with partially observed data is more well observed, meaning the models lean towards using this information to aid in predictions. Thus, the spatial models gain less of a bonus from incorporating this additional spatial information. It is reassuring to see from this that the spatial kernels aren't becoming overly confident on the spatial component of the modelling. This can occur due to numerical issues when estimating the hyperparameters of the spatial kernels, which cause the spatial kernel to essentially neglect

Table 7.18 Results for reconstruction of the validation data for the TMQ variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	RMSE	MAE
pace	2.5156 (0.0615)	1.8872 (0.0402)
fpca_gp	2.5206 (0.0614)	1.8905 (0.0393)
matern_three	**2.3960 (0.0561)**	**1.7995 (0.0446)**
gibbs	2.4649 (0.0663)	1.8489 (0.0426)

Table 7.19 Results for reconstruction of the test data for the TMQ variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	RMSE	MAE
pace	5.0394 (0.0966)	4.0714 (0.1078)
fpca_gp	5.0394 (0.0966)	4.0714 (0.1078)
matern_three	**2.4122 (0.0514)**	**1.8100 (0.0414)**
gibbs	2.5766 (0.0953)	1.9292 (0.0670)

the fact that the temporal component is present by making them exceedingly large. This is discussed more in Chapter 8.

Next, we consider the performance of our models on the test dataset. As mentioned in Section 6.1.2, this corresponds to functional data which we have not fully observed during estimation of model components and kernel hyperparameters where applicable. Table 7.19 presents these results.

Again, we see the Matérn model being best in class. Although the Gibbs model performs just as well, and both spatial kernels outperform the non-spatial models. It is encouraging to see that the pace and fpca_gp models perform equally, as these models are theoretically equivalent as explained in Chapter 5. This is a good indication that even on real world datasets using the CPACE framework is not detrimental to prediction performance in application. For an illustration of the improvements the CPACE framework with spatial kernels can make, we display an example of a curve prediction across the whole temporal domain in Figure 7.28. Here, we can see clearly, how the spatial models are informed by the neighbouring observed curves to help predict the level shift in the curves, but also help in predicting the amplitude shift to the mean function. It is interesting to note how similar the Gibbs and Matérn models predict here. However, in this case the Gibbs model confidence band for the prediction seems much more reasonable, whereas the Matérn model seems to be overly confident.

Finally, we consider the image reconstruction metrics for the models. Given the similar performance between the Gibbs and Matérn models, as with other studies, the difference between image reconstruction metrics which consider perceived similarity between the true surfaces and the reconstruction, may often be the metric of interest in choosing which to use for real world applications. These metric results are given in Table 7.20. This confirms the Matérn models best in class performance. However, two other important points can be raised from this. Firstly; the fpca_gp model performs slightly better in mean than

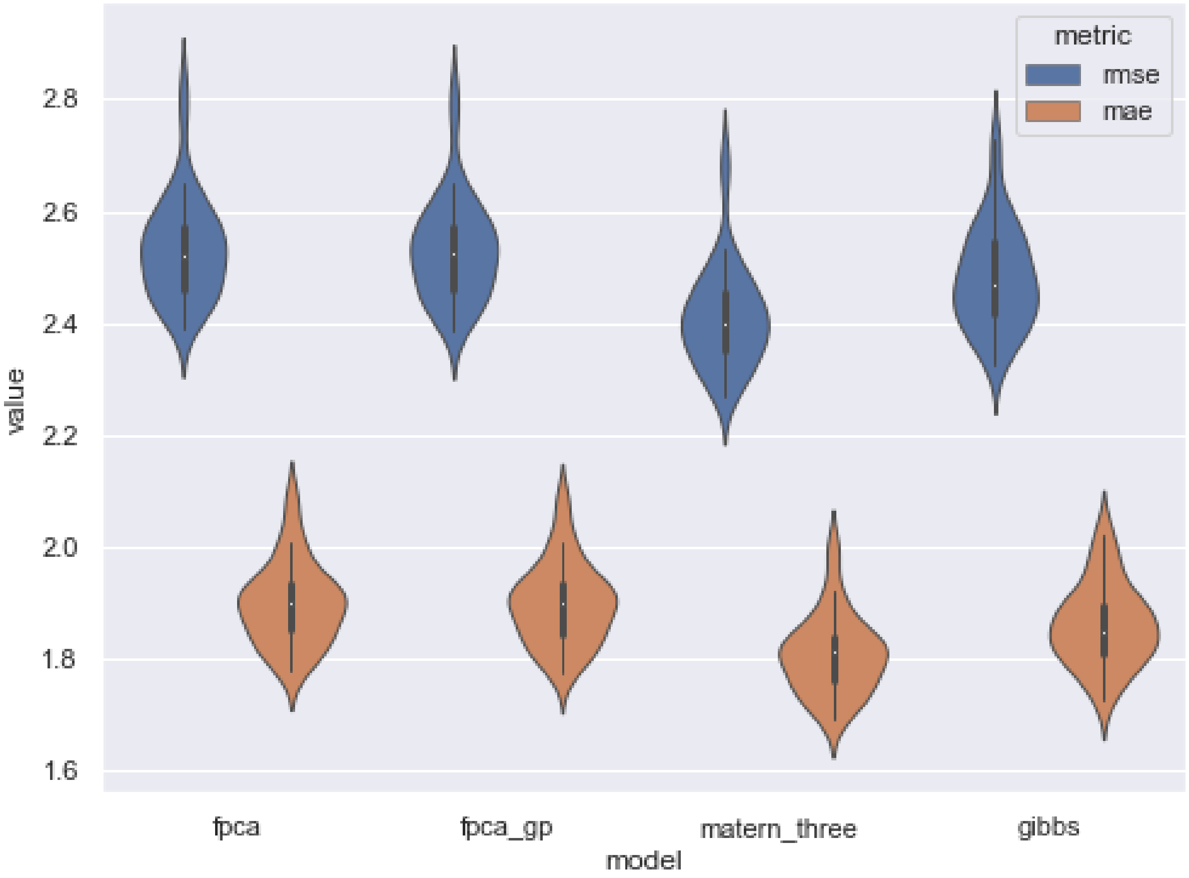


Fig. 7.27 Distribution plot of the RMSE and MAE metrics on validation data for the TMQ variable in the European study.

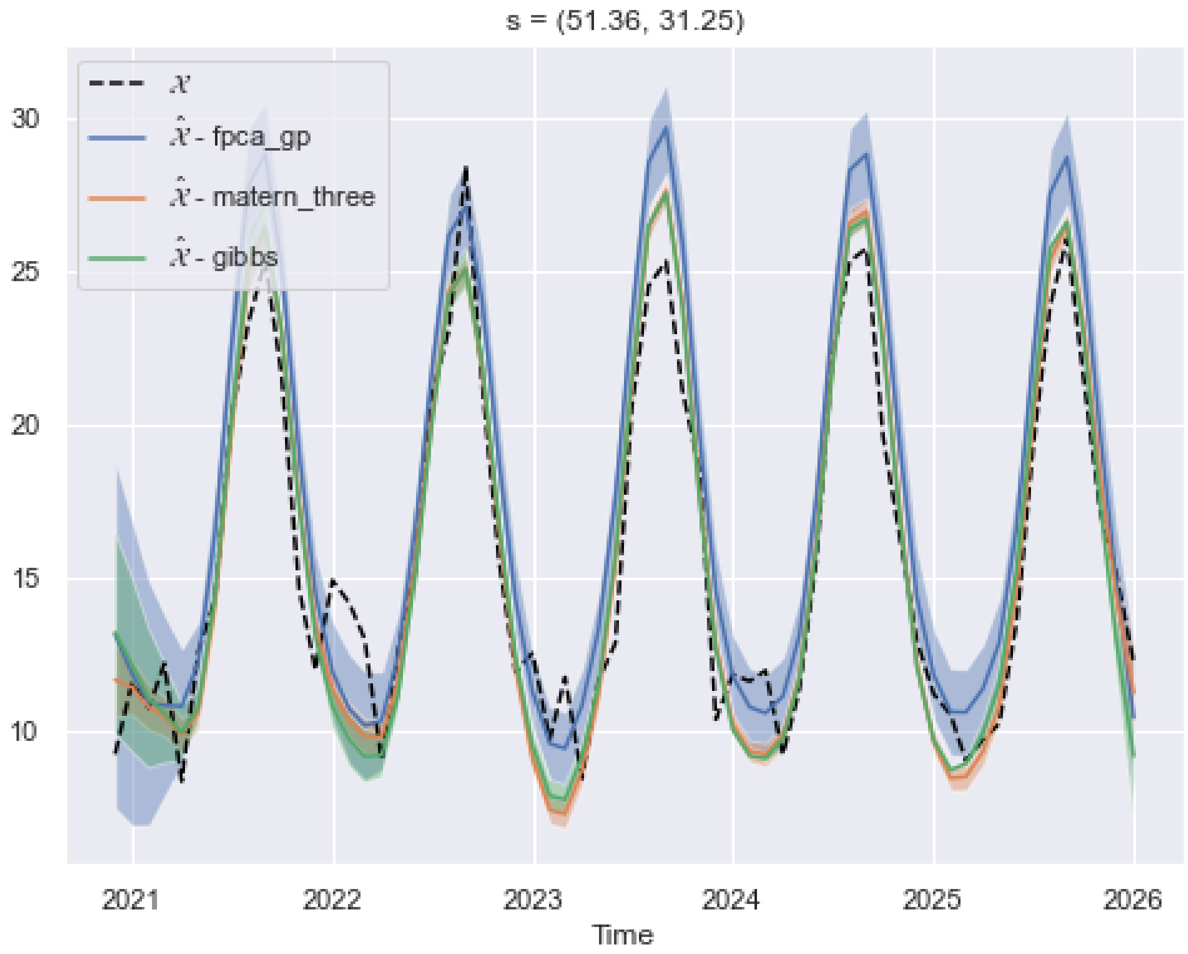


Fig. 7.28 An indicative example of the CPACE model performance on reconstruction of the CESM-LE TMQ variable in the European study from the test dataset.

Table 7.20 Results for reconstruction of the full data for the TMQ variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	PSNR	SSIM
`pace`	14.3970 (0.1016)	0.2674 (0.0068)
`fpca_gp`	14.3976 (0.1021)	0.2679 (0.0064)
`matern_three`	**20.5960 (0.2607)**	**0.7951 (0.0073)**
`gibbs`	19.9990 (0.3397)	0.7650 (0.0168)

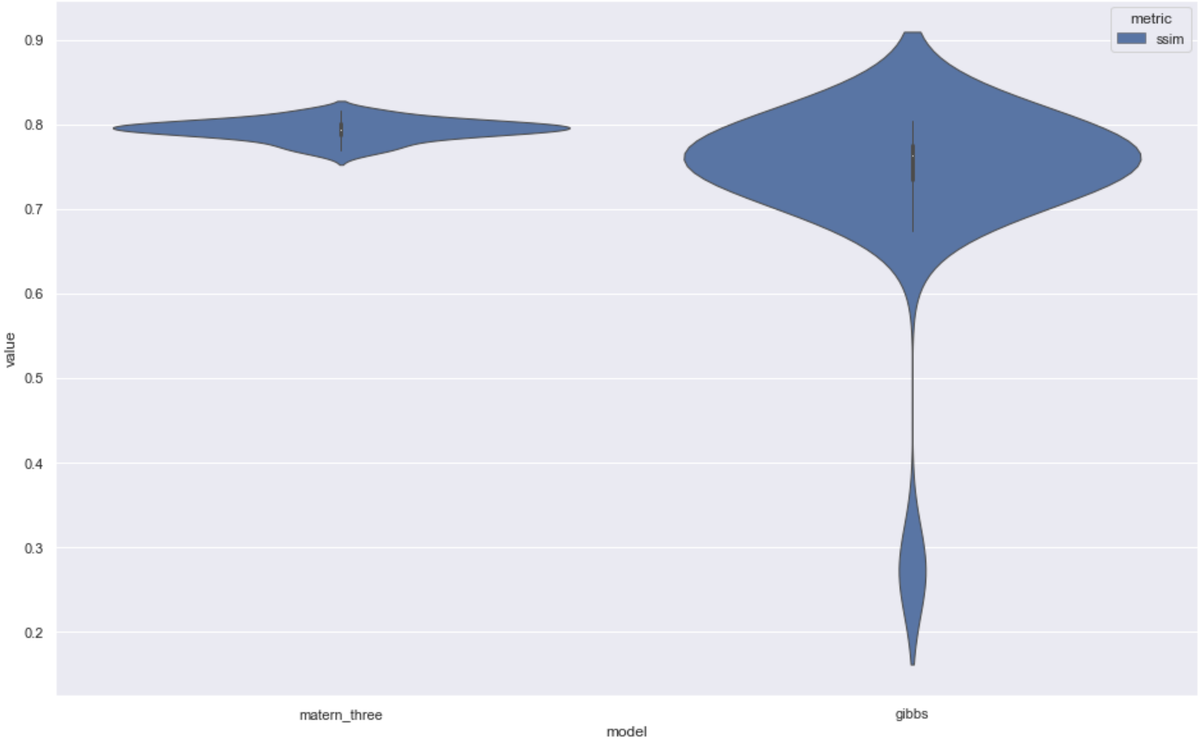


Fig. 7.29 Distribution plot of the SSIM metric on full data for the TMQ variable in the European study for the Matérn and Gibbs kernel.

the PACE model. As seen with the simulation study in Section 6.6; this is due to the ability to tune the kernel variances. Secondly; we see a slightly bigger variance of the Gibbs model results. We look at what causes this for the SSIM metric in Figure 7.29, which considers the distribution of the SSIM metrics for this study between the Matérn and Gibbs models. We can see that while the Gibbs model performs worse on average, it occasionally performs better. This indicates it might not be a clear cut decision that the Matérn model is always the correct kernel to choose for this study.

Irrespective of this we can see clearly a performance increase in using the CPACE framework with spatial kernels. Whilst this is not unexpected and ties in nicely with the Global study and the simulation study, it is encouraging to see that the CPACE framework continues to outperform the PACE framework on higher density observed functional data which is under consideration in this study. Finally, for illustration we display an example reconstruction for all models for a specific point in time across the full spatial domain in Figure 7.30. Here, we can clearly see the visual improvements which are represented in

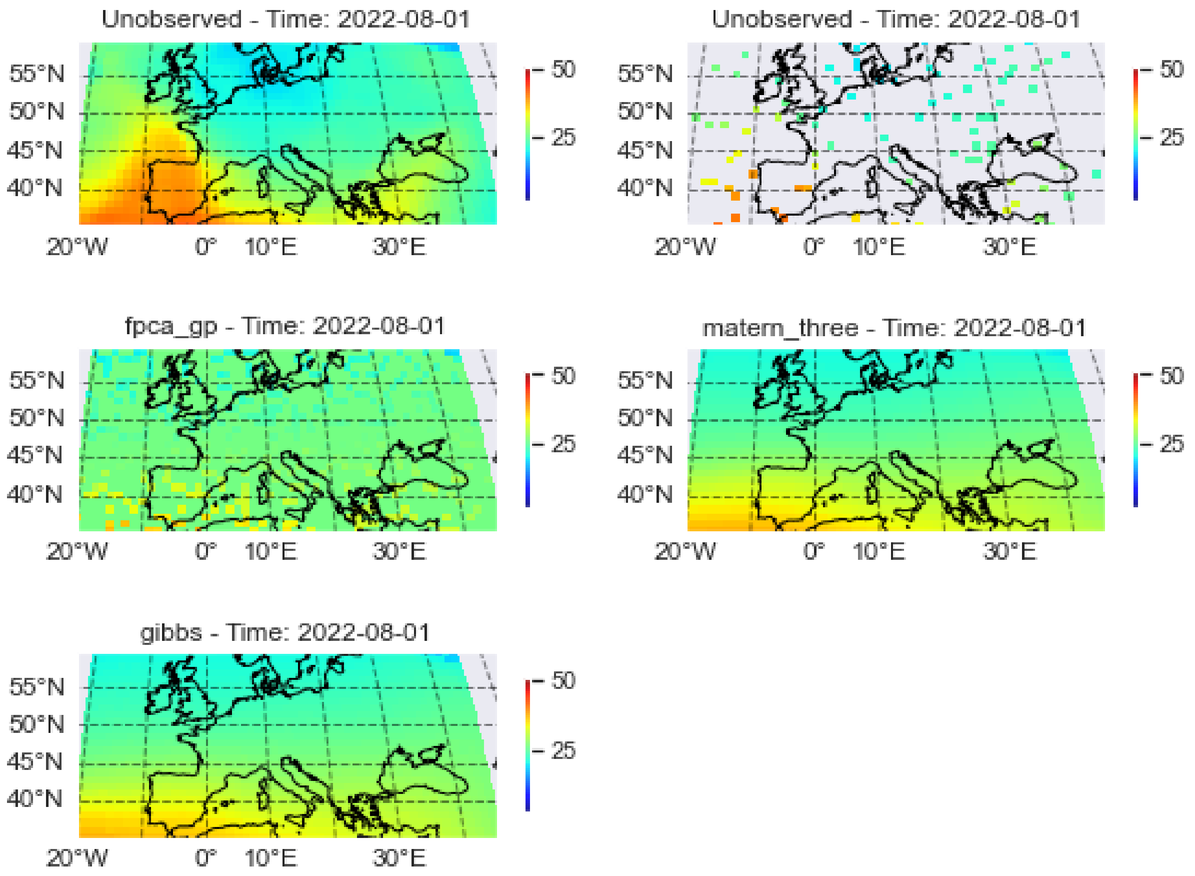

Unobserved - Time: 2022-08-01
Unobserved - Time: 2022-08-01
fpca_gp - Time: 2022-08-01
matern_three - Time: 2022-08-01
gibbs - Time: 2022-08-01
55°N
50°N
45°N
40°N
20°W
0°
10°E
30°E
50
25

Fig. 7.30 An indicative example of the CPACE model performance on reconstruction for the European study of the CESM-LE TMQ variable using the various models in the European study. Clockwise from the top left we have the true data, the observed data at a particular time point, the Matérn Three model reconstruction, the Gibbs model reconstruction, and the White model reconstruction under the CPACE framework.

the metrics throughout this section. However, we can also see that there are some areas in which we just don't capture the spatial variation correctly. These correspond to the areas of the highest peaks and troughs, where we fail to completely capture this extreme rapid variation from the mean curve. This is likely because the spatial variation changes quite rapidly over time in this dataset. The CPACE model assumes separate but constant spatial structure over time for each eigenfunction, and so with our 5 components in the models we do not capture these extreme cases of spatial variation, which occur infrequently over time. Increasing the number of components K in our models would likely alleviate this issue, however for this study we have not considered how to choose K. A discussion on this, not relating to these studies, is given in Chapter 8. More modest variation across space is well captured however, since this tends to be more consistent over time.

7.5.4 Wind Speed (U10)

In this section we present the results of the European study for the Wind speed variable. We begin by considering the training metrics for this study. That is the RMSE and MAE for our models ability to reconstruct partially observed functions. Table 7.21 displays the

Table 7.21 Results for reconstruction of the validation data for the U10 variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	RMSE	MAE
pace	1.1300 (0.0374)	0.8958 (0.0295)
fpca_gp	1.1277 (0.0375)	0.8930 (0.0288)
matern_three	**1.0906 (0.0351)**	**0.8642 (0.0273)**
gibbs	1.1130 (0.0381)	0.8807 0.0281

Table 7.22 Results for reconstruction of the test data for the U10 variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	RMSE	MAE
pace	1.6801 (0.0340)	1.2104 (0.0276)
fpca_gp	1.6801 (0.0340)	1.2104 (0.0276)
matern_three	**1.1354 (0.0425)**	**0.8908 (0.0242)**
gibbs	1.3265 (0.0613)	1.0053 (0.0299)

mean and standard deviation of these metrics from the 40 simulations which are part of the CESM-LE dataset. Again, as we have seen in the other European studies, all models perform equally well. The Matern̈ is chosen as best in class because of its lower mean metric scores. However, all models are within one standard deviation of each other, hence this is not a definitive indication that this kernel will perform best over all simulations.

Figure 7.31 gives an illustration of a reconstruction of a partially observed function. We can see that, on the whole, the models capture the periodic nature well, but struggle to capture the amplitude changes between observations. This is not an unusual sight for models with a periodic temporal component.

A similar set of results is obtained by looking at the performance of the models on the test data, that is, the prediction of completely unobserved functional data. Table 7.22 displays these results. We can clearly see a preference for the spatial kernels under the CPACE framework. Unlike the European studies for the previous variables of interest, the Matérn model is clearly the best model for this variable. An example reconstruction of a test data point is given in Figure 7.32 which highlights the difference between models. We can see clearly that they all fail to capture the variation in the peak in the middle of the domain, but the spatial kernel models capture the beginning and end of the temporal domain much better than the non-spatial kernel. The Matérn model seems to slightly outperform the Gibbs kernel in this one example in this area too.

Finally, we consider the ability of the models under the image reconstruction metrics as described in Section 7.1. We display these results in Table 7.23. As expected, we see the Matérn model performing best in class. However, it is interesting to note that unlike the the previous variables in the European study, the ability to reconstruct the U10 variable is much less. We can see a clear example of the struggles of modelling this data through Figure 7.33, which shows the reconstruction across the whole domain at a particular point in time. Here, it is obvious all models missed the large increase in windspeed in the

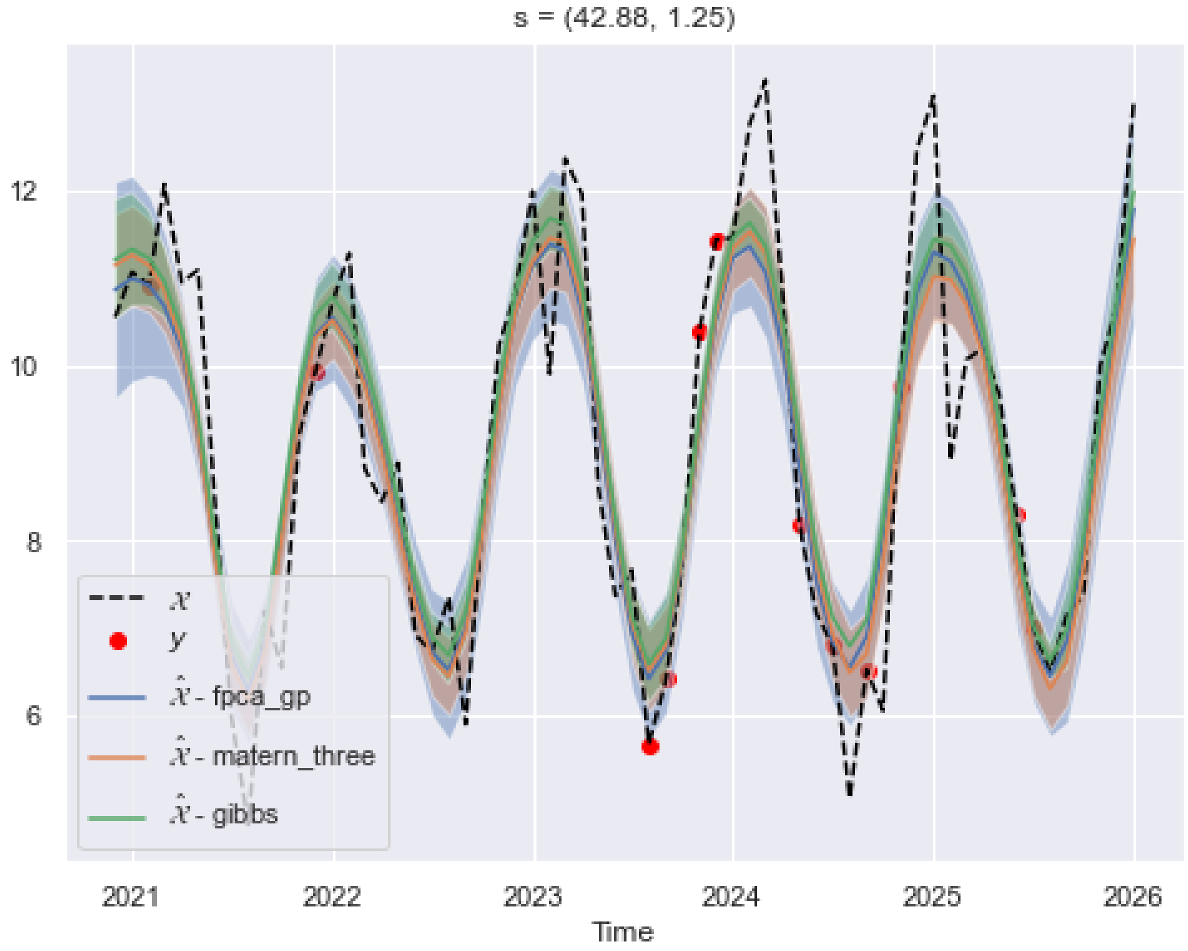


Fig. 7.31 An indicative example of the CPACE model performance on reconstruction of the CESM-LE U10 variable in the European study from the validation dataset.

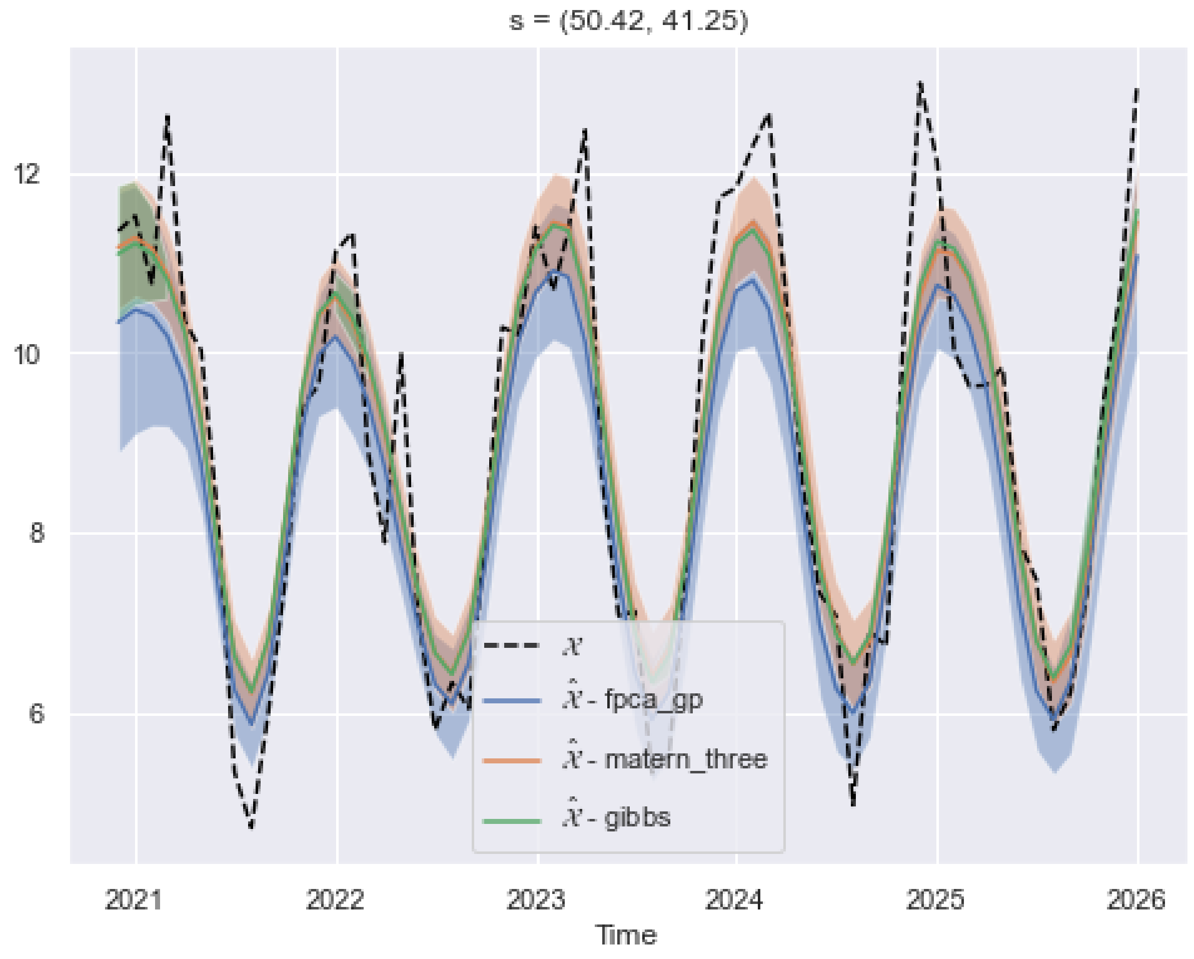


Fig. 7.32 An indicative example of the CPACE model performance on reconstruction of the CESM-LE U10 variable in the European study from the test dataset.

Table 7.23 Results for reconstruction of the full data for the U10 variable in the European study from the CESM-LE dataset. Bold indicates best in class.

Model	PSNR	SSIM
pace	15.7694 (0.1047)	0.2681 (0.0034)
fpca_gp	15.7721 (0.1054)	0.2688 (0.0033)
matern_three	**18.7107 (0.1809)**	**0.4674 (0.0139)**
gibbs	17.5800 (0.3304)	0.4257 (0.0173)

Atlantic. Similar to the study on the precipitation, Section 7.5.3, this is because this phenomena is short lived, and the spatial variation is not constant through time. The short lived nature of such phenomena make it difficult for the PACE and CPACE frameworks to pick up as eigenfunctions, since overall it does not account for much variation. Further, the change in spatial variation this causes is not picked up by the CPACE framework, as it would be required to be an eigenfunction which would then get its own spatial variation over time. One way to capture this would be to let K, the number of components of the model be increased. However, in these studies we have not considered how to choose K, a discussion on this is presented in Chapter 8.

It is also interesting to note, that in Figure 7.33, we can see the Gibbs kernel and the Matérn kernel disagreeing on the structure. One example is the area over the Baltic sea; where the Gibbs model prefers a more variable structure here than the Matérn kernel. This is a prime example of the added flexibility of the non-stationary Gibbs kernel.

7.5.5 Discussion

In the above, we have considered comparing the PACE methodology with our CPACE framework with a number of different kernels on a subset of the full CESM-LE dataset. We can see that in all cases, alike with the Global studies, the CPACE spatially dependent models outperform the PACE framework. This is again intuitive due to the geographic nature of the dataset which clearly has a spatial component that is not accounted for in a PACE framework.

Our results suggest a Matérn kernel is typically the most appropriate kernel of the three considered. This is usually closely followed by the Gibbs kernel, and in some cases the Gibbs kernel is preferred, especially when considering the model which may maximise the image reconstruction metrics. We can see a clear improvement in reconstructing the smaller European dataset than compared to the Global study. Again, this is quite expected due to the construction of this study having denser observations. This allows for a better estimate of the mean and covariance structure, which are used in both the PACE and CPACE framework. It is encouraging to note that the CPACE framework still outperforms the PACE framework in this setting, that is the denser observations do not reduce the need for a spatial kernel. Although we do see a slight decrease in comparative performance of the CPACE framework on partially observed functions when densely observed.

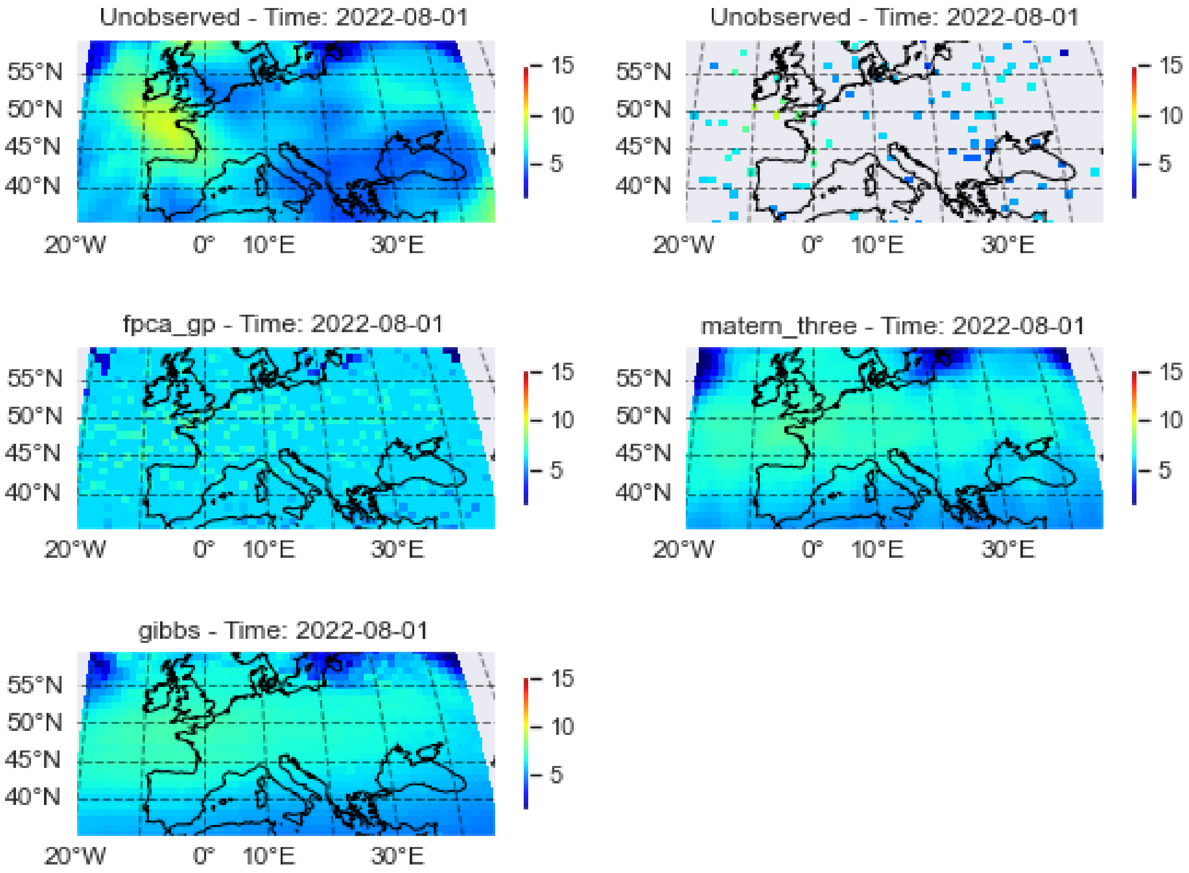

Unobserved - Time: 2022-08-01
Unobserved - Time: 2022-08-01
fpca_gp - Time: 2022-08-01
matern_three - Time: 2022-08-01
gibbs - Time: 2022-08-01
55°N
50°N
45°N
40°N
20°W
0°
10°E
30°E
15
10
5

Fig. 7.33 An indicative example of the CPACE model performance on reconstruction for the European study of the CESM-LE U10 variable using the various models. Clockwise from the top left we have the true data, the observed data at a particular time point, the Matérn Three model reconstruction, the Gibbs model reconstruction, and the White model reconstruction under the CPACE framework.

The wind speed, U10, and precipitation, TMQ, in general were the most difficult to model. As explained in Section 7.5.4 and Section 7.5.3 respectively. This is because of non-constant spatial variation over time, and the limited number of component eigenfunctions we have used in this study.

The European studies illustrate that the CPACE framework is quite applicable to smaller spatial scale studies.

7.6 Summary

In the above global and European studies, Section 7.4 and Section 7.5 respectively, we compared how the CPACE framework handles our CESM-LE data. As described in Section 2.1, this is often used as a real world EO dataset. The data generating process of this dataset is far removed from the assumed model that the CPACE framework is based on. We see encouraging results from both studies. We found a good performance increase on the pixel related metrics of the RMSE and MAE. More importantly for application perhaps, we notice a good increase in the SSIM and PSNR for the spatially dependent CPACE models. This indicates these models are ideal candidates to help reproduce imagery from partial or completely missing EO data.

We have explored the difference the CPACE framework makes on both densely observed data, as in the European study, and on very sparsely observed data in the Global study. The improvements seen using the same framework in both is encouraging, and as discussed in Chapter 8, the setup and implementation for various spatial domains requires little fine tuning.

We note that improvements from the PACE methodology was seen across all variables of interest, however all models found the U10 and TMQ datasets most difficult to predict. This we reasoned is due to these variables having the most complicated generating procedure, with examples of spatial variation which is non-constant through time. This is something which would require a complete eigenfunction and the corresponding spatial kernel to capture in the CPACE model and we studied models with a fixed 5 components which typically wasn't enough to capture this independently. The reasoning on how to select K, the number of components in the CPACE model, is briefly discussed in Chapter 8 and is not something we considered for these studies.

It is notable, that in the eigen decomposition of all the variables, across the European and global studies, the first eigenfunction typically related to a level shift of the mean function. This is both interesting and encouraging, as this often corresponds to a shift in the mean function as you travel across latitude bands of the globe. This highlights an important aspect of the PACE modelling; which is its ability to explain datasets. This is something that the CPACE framework keeps with the same eigen decomposition. It is encouraging to see that this application resulted in such interpretable eigenfunctions.

Chapter 8

Implementation of CPACE model

In Chapter 5 we presented the CPACE framework to model functional data that is observed with correlation. We have shown the improvements that such a framework can achieve on simulated and real world data in Chapter 6 and Chapter 7 respectively. The framework relies on adapting the PACE methodology and viewing the collection of functional data as a realisation from a larger random field. To this end, we proposed to model such a realisation as a Gaussian process whose kernel function is informed by the principal components from the PACE framework. As such there are a number of components which need to be estimated, namely the mean function and covariance surface, the eigenfunctions and eigenvalues of the covariance surface, the noise variance, and the between function spatial kernel hyperparameters. We have shown in Chapter 5 theoretically how one would estimate these, and shown in Theorems 5.1, 5.2 the validity of estimating the mean and covariance functions using a penalised B-spline approach for correlated functional data.

However, implementing these approaches often requires subtleties. This can be due to numerical stability in calculation or computational feasibility due to dataset size. In this chapter we look at the implementation details we have used, the reasoning behind them and the benefits they give. We begin by looking at our approach to estimating the penalised B-spline smoother which is used to estimate our mean, covariance surfaces, and eigenfunctions. This section is focused on the implementation details used within the CPACE framework, however the same considerations apply to the functional time series model (described in Chapter 4). Following this; we consider implementation details used in the simulation study, and application to the CESM-LE dataset regarding numerically stable results. Finally, we consider our approach to kernel hyperparameters estimation with a view on reducing computation time and applicability to large datasets. In addition, we focus on our implementation of the Gibbs kernel for use within this framework.

8.1 Penalised B-Spline Smoothing

The theoretical aspects of penalised spline smoothing using B-splines have been discussed in Section 3.3 with the validity of using these for mean function and covariance surface estimation in Theorem 5.1 and Theorem 5.2 respectively. Here, we focus on a discussion of the implementation details.

To begin, we consider how to specify the penalty matrix $\boldsymbol{P}$ in Equation (3.12). Theoretically, as mentioned in Section 3.3.2, this matrix is formed with $(l,m)^{\text{th}}$ element being the inner product between $L\left(B_l\right)$ and $L\left(B_m\right)$, where L is some linear differential operator and B_l, B_m are the l^{th} and m^{th} basis functions respectively. In this work we have considered only the simple case where L is the first or higher order derivative. Specifying a more complicated form of L is indeed a common action in functional data analysis, [63], however it often is used when some underlying known process is being captured through this. For this work, we only use this as a form to ensure our estimated functions from this approach are smooth, with the smoothness being represented by the order of the derivative.

The next consideration is the actual computation of $\boldsymbol{P}$. Since, in general, the inner product does not have a nice analytical form, a numerical approach is used. One could use numerical integration, however often we encounter numerical instabilities with such approaches. For our implementation we follow Wood's method for computing such penalties, [86]. They compute the penalty matrix, $\boldsymbol{P} = \boldsymbol{c}^{\mathsf{T}}\boldsymbol{S}\boldsymbol{c}$ where $\boldsymbol{S}$ is a banded diagonal matrix. The coefficients of $\boldsymbol{S}$ can be computed efficiently by following the algorithm in [86]; this uses the fact that evaluating the B-spline basis and it's derivatives is quick and relatively easy with standard methods, and that the integral for each element of $\boldsymbol{S}$ is constructed from the sum of integrals over some polynomial segments of order p which is decided by the derivative chosen. This offers the ability to mix and match derivative based penalties with B-splines and [86] gives a computationally stable method for calculation. In addition, both [86] and [83] show a method to extend this approach to tensor product of B-splines which we utilise in our implementation of both the CPACE framework (for the covariance surface) and our functional time series model in Chapter 4. This methodology is both computationally stable and efficient.

Finally, one must consider how to choose the smoothing parameter; $\omega \in \mathbb{R}^+$ in Equation (3.12). As discussed in Section 3.3.2 we utilise the GCV to choose the smoothing parameter. There are plenty of schemes designed to minimise the GCV with respect to ω, many of which are discussed in [85]. These consider the more general case where the model is a generalised additive model, whereas in our models the link function is simply the identity. We opt to use the direct fitting method, as described in [84]. This highlights a hierarchical method to estimating the smoothness parameters via first estimating the coefficients of the system through a penalised iteratively re-weighted least squares (P-IRLS) approach. We can safely ignore the iteration because we can directly estimate the coefficients from Equation (3.12). We utilise their proposed stable approach to solving Equation (3.12), by considering the QR decomposition of the design matrix adjusted by the square root of the penalty matrix by appending it to the rows, [84]. Then with this fixed, estimates of the derivatives of the GCV with respect to the smoothing parameter ω can be achieved.

We diverge from [84] at this point since we choose not to utilise the analytic approach suggested. We choose to use automatic differentiation to calculate this derivative, discussed further in Section 8.4. This approach allows for speed of development, with no need to directly implement the relatively complicated updates to calculate the analytic derivative. This numerical derivative along with the calculated GCV for a particular smoothing parameter is then fed into a numerical optimisation routine.

For our spline smoothing we opt to utilise the N-ADAM optimiser routine, [16]. This is a first order gradient based optimiser routine which incorporates the Nesterov momentum into the popular ADAM optimiser, [16]. This was chosen purely for the performance and speed of iteration, and is utilised in the smoothing models which are within both the FTSM model in Chapter 4 and the CPACE model in Chapter 5. As the N-ADAM optimiser is purely gradient-based, good initialisation is crucial. Another issue and implementation detail for the penalised B-spline smoothing is the initialisation of the smoothing parameter and the setup of the design matrix in Equation (3.12). We initialise the smoothing parameter to a variety of values independently, and choose to continue the minimisation routine with the best performing initial smoothing parameter. That is we initially uniformly choose smoothing parameters from between $1e-6$ and $1e4$. We run the B-Spline fitting procedure with these smoothing parameters independently and calculate the GCV score associated to each. We then proceed with optimising the smoothing parameter with the initial starting value which gave the smallest GCV score from our initial choices. This is a common approach to initialising smoothing parameters. In our experiments and CPACE study we opted to use 32 initial guesses for smoothing parameter.

The setup of the design matrix for the penalised B-spline smoother which we utilise throughout the CPACE framework is also of importance. The normalisation of the design matrix, $\boldsymbol{B}$, and the response vector $\boldsymbol{Y}_i$ from Equation (3.12) is a standard approach to this. We follow, [82], and firstly normalise both the response $\boldsymbol{Y}_i$ and the design matrix by the norm of the design matrix $\boldsymbol{B}$. This aids in numerical stability.

Next, in Section 8.2, we discuss implementation considerations with respect to the choice of the number of components, K, in the eigen decomposition.

8.2 Choosing the Number of Components

The number of components in the eigen decomposition of the covariance surface of our data generating procedure is a key component in both the PACE and CPACE framework. It has many parallels with the number of components chosen in a PCA decomposition with multivariate data. Each component, or eigenfunction, will capture a mode of variation present in the dataset.

In theory, letting the number of components tend to infinity will allow perfect representation of the covariance surface. However, each component we keep in the eigen decomposition of the covariance surface will add to the computational complexity of the model. Each component is represented with a penalised B-spline smoother in our CPACE framework, and thus the coefficients in that representation will need estimating. In addition, extra computation is needed for the Gaussian process framework, which is discussed in Section 8.3. So as always, there is a trade-off between choosing K large enough to capture sufficient variation whilst maintaining computation feasibility.

One approach, is to choose K such that the eigenfunction explain $\alpha\%$ of the total variance. This approach relies on first computing the full eigen decomposition of an estimated covariance matrix into its eigenvectors and eigenvalues. This estimated covariance matrix is the discretised version of the covariance surface described in Section 5.4. The eigenvalues correspond to the variance of the respective eigenvectors. From this we can then choose the first K components which capture $\alpha\%$ of the total variation. This is a common holistic method in PCA, [39]. We use this method in our implementation of the CPACE framework for both the simulation study and the application to the CESM-LE dataset with $\alpha = 0.95$, which led to our choice of $K = 5$ in both cases.

However, there is no guarantee that this holistic method will lead to the optimum choice of K for reconstructive ability of the CPACE framework. There are a variety of other methods, discussed in [39] and more recently by Josse and Husson in [40]. One such method which seems promising would be the selection of K through a form of cross-validation, [40]. Intuitively, this would be performing an hierarchical model selection step, which would involve selecting K, estimating all the model's hyperparameters, then evaluating on a proportion of the observed data to obtain a loss score. We repeat this for a selection of K values, then choose the K which would minimise this score. In this work, we haven't considered such an exhaustive search for K since it would lead to increased model estimation time. However, such an approach certainly would be a good direction for future work on the CPACE framework.

8.3 CPACE Gaussian Process Implementation

As discussed in Chapter 5 the CPACE framework is built upon viewing functional data as a realisation from a Gaussian process with a specific structured field. Gaussian processes are known to offer some nice properties; such as finite moments and analytical predictive variance, [80]. However, it is well known that Gaussian process models scale poorly with increase in data size. In this section, we present our implementation of the CPACE framework which helps to address these issues, whilst retaining the positive properties that a Gaussian process model has.

We first consider the construction of the space-time covariance kernel that the CPACE framework assumes. The functional form of which is given in Equation (5.7). We repeat this below for convenience.

$$a_{\mathcal{X}}\left(\boldsymbol{s}, t, \boldsymbol{s}', t'\right) = \boldsymbol{\phi}^{\intercal}(t) \mathrm{Diag}\left(a_1(\boldsymbol{s}, \boldsymbol{s}'), a_2(\boldsymbol{s}, \boldsymbol{s}'), \cdots, a_K(\boldsymbol{s}, \boldsymbol{s}')\right) \boldsymbol{\phi}(t').$$

This form gives an idea of how the model is computed, however a naive construction by applying this functional form to every space-time point we wish to calculate for can be computationally intensive. This is for a couple of reasons. Firstly, considering we wish to calculate the covariance between N spatial locations, at each location we observe

J_i temporal points, $\mathcal{X} = \{(\boldsymbol{s}_i, t_{ij})\,; i = 1, 2, \dots, N, j = 1, 2, \dots, J_i\} \subset \mathcal{S} \times \mathcal{T}$. Our total number of observation points is then given by $|\mathcal{X}| = \sum_{i=1}^{N} J_i$.

A direct matrix implementation would require the construction of the vector of eigenfunctions which is $K \times 1$, and the diagonal $K \times K$ matrix of spatial covariance. This would have to be repeated $|\mathcal{X}| \times |\mathcal{X}|$ times; once for each location. When $|\mathcal{X}|$ becomes large either through frequent temporal observations or more spatial locations, this direct method would require either extreme parallelisation or long computation time. This becomes exaggerated when the K spatial kernels are expensive to compute in themselves, such as the case with the Gibbs kernel that we have discussed in Section 7.2.

One method to increase the speed of calculating the covariance matrix, $\Sigma_{\mathcal{X}}$, is to consider a vectorised approach. This was utilised in [48] for the SPACE model, albeit under a different framework. This involves creating the matrix $\boldsymbol{U} = [U_1, U_2, \dots, U_K]$ where $U_k \in \mathbb{R}^{|\mathcal{X}| \times N}$ is the sub matrix of columns which has the following form:

$$\begin{bmatrix}
\phi_k(t_{11}) & 0 & 0 & \dots & 0 \\
\phi_k(t_{12}) & \ddots & \ddots & \ddots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
\phi_k(t_{1J_1}) & 0 & \ddots & \ddots & \vdots \\
0 & \phi_k(t_{21}) & 0 & \ddots & \vdots \\
0 & \phi_k(t_{21}) & 0 & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \phi_k(t_{2J_2}) & 0 & \ddots & \vdots \\
\vdots & 0 & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \phi_k(t_{N1}) \\
\vdots & \ddots & \ddots & \ddots & \phi_k(t_{N2}) \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & \dots & \phi_k(t_{NJ_N})
\end{bmatrix}.$$

The corresponding matrix $\boldsymbol{A} \in \mathbb{R}^{KN \times KN}$ can be formed such that $\Sigma_{\mathcal{X}} = \boldsymbol{U}\boldsymbol{A}\boldsymbol{U}^{\intercal}$ where $\boldsymbol{A} = \operatorname{diag}(A_1, A_2, \dots, A_K)$ is the block diagonal matrix. The k^{th} block A_k is found simply by evaluating the spatial kernel $a_k(\boldsymbol{s}, \boldsymbol{s}')$ at all pairs of spatial points $\boldsymbol{s}_i$ for $i = 1, 2, \dots, N$.

This approach has a number of advantages. Firstly, computation of the spatial kernel component is greatly sped up by having to only evaluate the component spatial kernels each on $N \times N$ positions. This is a reduction by a factor of K. Typically the spatial kernel will be more computationally intensive to compute than the eigenfunctions so this saves a great deal of computation time. Secondly, this representation is inherently sparse, which has two implications. We can speed up the matrix multiplications between the sparse representation for $\boldsymbol{U}$ and the block diagonal matrix of $\boldsymbol{A}$, using a matrix multiplication algorithm for sparse matrices. Many linear algebra routines have functionality built in to accommodate this. In our implementation we use the `tensorflow` library which has this capability, [1]. We also use this sparse representation to accommodate much larger

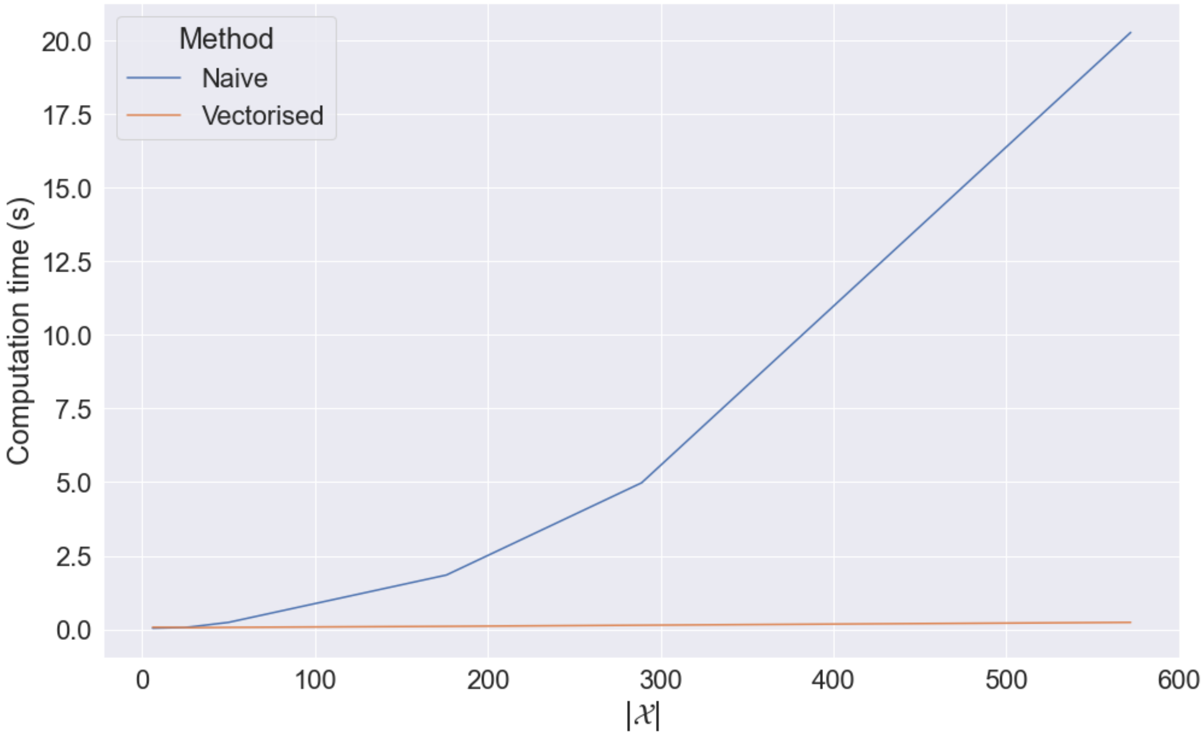


Fig. 8.1 A illustration for the computer time taken to calculate $\Sigma_{\mathcal{X}}$ for the naive and vectorised methods for multiple sizes of observed data. We use $K = 5$ for our illustrative example. The above computation times were taken from a single machine running with the Darwin kernel Version 21.1.0 on an i386 processor with 16Gb of RAM.

datasets. This is simply down to the fact we do not have to store large matrices in this representation. In fact, in our implementation for the CPACE model we only need to store the sparse representation for $\boldsymbol{U}$ and the blocks of $\boldsymbol{A}$.

We illustrate the comparative time taken for this approach to the naive one in Figure 8.1. We can see the similarity in computation time for small values of the datasets, but we can also see the much shallower increase in computation time between the naive method and vectorised method. The reason for such great speed up is given above.

This method for calculation of $\Sigma_{\mathcal{X}}$ has an additional use in the calculation of the log marginal likelihood, given in Equation (5.74). Renowned for being the bottleneck of a standard Gaussian process model, [80], the minimisation of the log marginal likelihood is typically time consuming due to the need to calculate the inverse and determinant of the covariance matrix of the process for the training observations. This can be particularly challenging since the standard approach for doing so involves the Cholesky decomposition of $\boldsymbol{\Sigma}(\boldsymbol{Y}, \boldsymbol{Y})$. The algorithm used to compute this Cholesky decomposition has computational complexity of $O(\frac{|\mathcal{X}|^3}{6})$. This does not scale particularly well as $|\mathcal{X}|$ gets larger. However, we can use the structure of $\Sigma_{\mathcal{X}}$ to improve this in our CPACE model.

In particular, noting that $\boldsymbol{\Sigma}(\boldsymbol{Y}, \boldsymbol{Y}) = \Sigma_{\mathcal{X}} + \sigma_{\varepsilon}^2 \boldsymbol{I}$ has a very particular structure which is given below;

$$\boldsymbol{\Sigma}(\boldsymbol{Y}, \boldsymbol{Y}) = \boldsymbol{U}\boldsymbol{A}\boldsymbol{U}^{\intercal} + \sigma_{\varepsilon}^2 \boldsymbol{I}.$$

We can use the Woodbury matrix identity to calculate it's inverse, and the matrix determinant lemma to calculate its determinant.

The Woodbury matrix identity states that inversion of a rank k matrix can be computed by applying a rank k correction to the inverse of the original matrix, [87]. In our model we have:

$$\left(\boldsymbol{U}\boldsymbol{A}\boldsymbol{U}^{\intercal} + \sigma_{\varepsilon}^{2}\boldsymbol{I}\right)^{-1} = p\boldsymbol{I} - p\boldsymbol{U}\left(\boldsymbol{A}^{-1} + p^{2}\boldsymbol{U}^{\intercal}\boldsymbol{U}\right)^{-1}\boldsymbol{U}^{\intercal}, \tag{8.1}$$

where $p = \frac{1}{\sigma_{\varepsilon}^{2}}$ is the precision. Here this has reduced the large $|\mathcal{X}| \times |\mathcal{X}|$ matrix inversion into two smaller inversions of size $(K \times N) \times (K \times N)$. We note now that the inversion of the matrix $\boldsymbol{A}$ can be simplified further due to its block diagonal structure. This inversion actually only requires K lots of $N \times N$ inversions. The outer inversion, $\left(\boldsymbol{A}^{-1} + p^{2}\boldsymbol{U}^{\intercal}\boldsymbol{U}\right)$ is not inverted directly as for the marginal log likelihood we require $\boldsymbol{\Sigma}\left(\boldsymbol{Y},\boldsymbol{Y}\right)^{-1}\boldsymbol{Y}$. Hence by postmultiplying Equation (8.1) by $\boldsymbol{Y}$ we can use the Cholesky factor of this outer matrix and a Cholesky solve against $\boldsymbol{U}^{\intercal}\boldsymbol{Y}$ to calculate this component in a numerical stable way.

For the marginal log likelihood we also require the determinant of $\boldsymbol{\Sigma}\left(\boldsymbol{Y},\boldsymbol{Y}\right)$. The matrix determinant lemma, [26], is the analogue to the Woodbury matrix identity for their determinants. By using the matrix determinant lemma the determinant of $\boldsymbol{\Sigma}\left(\boldsymbol{Y},\boldsymbol{Y}\right)$ can be found as:

$$\det\left(\boldsymbol{\Sigma}\left(\boldsymbol{Y},\boldsymbol{Y}\right)\right) = \det\left(\boldsymbol{A}^{-1} + p^{2}\boldsymbol{U}^{\intercal}\boldsymbol{U}\right)\det(\boldsymbol{A})\det(\sigma_{\varepsilon}^{2}\boldsymbol{I}).$$

Again, this works in tandem with the Cholesky decomposition of the larger outer matrix and if the inner inversion of $\boldsymbol{A}$ is also calculated with a Cholesky decomposition the full determinant follows easily from the Woodbury inversion in Equation (8.1). Note that these formulas are exact and no approximations are required, thus we retain the exact properties of the Gaussian process if we follow the above implementation whilst reducing the complexity of calculating the expensive operations in the marginal log likelihood from $O(\frac{|\mathcal{X}|^{3}}{6})$ to $O((KN)^{3})$. This essentially means that the number of time points we observe at each spatial location is not a limiting factor in our estimation procedure. Since our computation time only has cubic growth with the number of spatial points and model components. Figure 8.2 highlights the effect of using the above method on computation time for a matrix inversion. As can be seen from this the above implementation using the Woodbury inversion technique reduces the computational burden significantly.

The above implementation for our CPACE model makes the model feasible on relatively large datasets. However, it still requires multiple evaluations of the marginal log likelihood to estimate the hyperparameters of the spatial kernels. We discuss our approach to this in the following section.

8.3.1 Maximising the Marginal Log Likelihood

We have stated in Chapter 5 that any hyperparameters of the CPACE model, most prominently the spatial kernel hyperparameters, are estimated by minimising the negative log marginal likelihood of the model; this is equivalent to maximising the marginal log

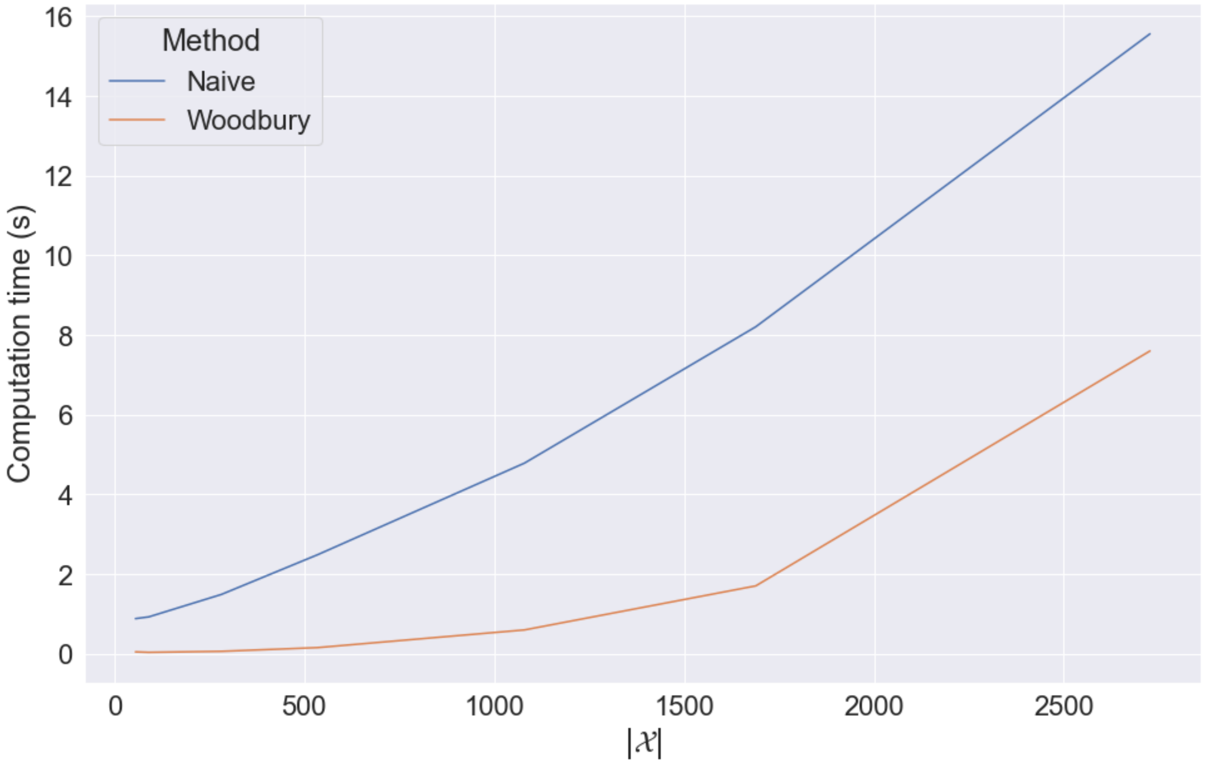


Fig. 8.2 An illustration for the computer time taken to calculate $\Sigma_{\mathcal{X}}$ inverse for the naive and vectorised methods for multiple sizes of observed data. This illustrative example uses $K = 5$ number of components in the CPACE model. The above times were taken from a single machine running with Darwin kernel Version 21.1.0 on an i386 processor with 16Gb of RAM.

likelihood. To do so we follow our minimisation methodology from Section 8.1 and use a gradient descent based algorithm. In particular, we also use automatic differentiation to calculate our gradients. This is discussed in Section 8.4.

In our implementation, we have opted to use a stochastic gradient descent (SGD), [72], algorithm for this. Whilst in principle any gradient descent algorithm should perform fine, we found through experimentation that a SGD algorithm tended to perform best. Our intuitive reasoning for this is that, due to the large number of hyperparameters in the CPACE framework, it is relatively easy for the gradient descent algorithms to get stuck in local optima. The stochastic nature of SGD essentially randomly perturbs the true direction of travel by approximating the gradient from a random subset of data. This added variability gives it the chance to overcome any small local optima caused by specific hyperparameters, whereas deterministic approached like N-ADAM, [16], tended not to find these.

Two more items are important in implementing such a gradient descent approach to maximising the marginal log likelihood; the initialisation of the algorithm, and the stopping criterion. To initialise our minimisation routine we opt for random starts. In particular, we pick a number of places from the likely domain of the hyperparameters. For each choice we calculate the marginal log likelihood, then start the SGD initialised at the point which has maximum marginal log likelihood. Whilst this does not guarantee to aide in finding global optima it is a often used heuristically, [72]. The reasoning being it gives the descent algorithm a chance of starting in a location where the global optimum may exist. The exact number of initialisation attempts is chosen by the user and is usually problem dependent. For example; in our simulation study, Chapter 6, we use 32 random initialisations. The trade off for choosing this is purely down to time available for training. There is a similar trade off used in deciding when to stop the gradient descent algorithm.

As with any minimisation procedure, how to stop is often a tricky problem. For SGD it is of particular importance, as the stochastic nature may mean you can step away from the gradient only to return to it later in the routine. Early stopping is a classic heuristic approach to prevent wasted computation when this happens, [72]. In our implementation for our simulation study we decided to stop when the absolute relative change in gradient was less than a tolerance of 0.001% or the relative change of the maximum log likelihood was less than a tolerance of $1e^{-5}\%$. Again, these stopping criterion are usually problem dependent, [72]. The larger tolerance for stopping can often reduce estimation time but may mean you end up with sub-optimal hyperparameters. A narrow tolerance may mean you produce more optimal hyperparameter estimation, but can cause the computational time to increase largely.

Finally, to help speed up computation of this minimisation procedure we implement batched stochastic gradient descent, [46]. We consider our training datasets in batches, and perform the update for the hyperparameters on each batch in sequence. Our training data is split into b sized batches, b is chosen heuristically. Typical choices are $16, 32$, and 64. A single iteration, or epoch, of estimation of our hyperparameters results in batching our training data then looping through all batches of our training data and performing a SGD update step. This effectively adds two improvements to our estimation procedure. There is a dramatic speed up since now gradients of the marginal log likelihood are computed over the batch, so the batch size, b, is determining our maximum size of the computationally complex Cholesky decomposition as described in Section 8.3.1. We note that when we split our training data up into batches, we split entirely on the location, thus we are splitting N into batches. This means that all temporal observations for a particular functional data are contained in the same batch.

This batched approach to estimating kernel hyperparameters is well used in Machine learning, [46]. The speed up of computation offered by this is bought at the price of added noise. In practice, this approach works well with SGD and is widely used, [46], [72].

In this section we have detailed our estimation procedure for the hyperparameters in the CPACE framework. As with most Gaussian process implementations we estimate these through maximising the marginal log likelihood. We have detailed our implementation of this maximisation approach using a batched gradient descent algorithm. What remains to be specified is how we obtain the gradients with respect to the hyperparameters which are used in such gradient based algorithms. We do so in the following section.

8.4 Automatic Differentiation

In our CPACE framework, described in Chapter 5, there are two main places where we need to estimate hyperparameters; in the estimation of eigen functions through penalised B-Spline smoothing and in the estimation of our spatial kernel hyperparameters. We have outlined our implementation of estimating these in Section 8.1 and Section 8.3.1 respectively. Each of these require a gradient based optimiser, and thus need gradients with respect to hyperparameters. In this section we outline how we go about obtaining these.

There are three popular methods for computing gradients and Hessians of complex mathematical functions using a computer; numeric, symbolic, and automatic differentiation. Numeric differentiation uses the method of finite differences to approximate the derivative. It can introduce rounding errors through the discretisation of the problem as well as issues surrounding cancellation causing the estimated gradient to be a poor approximation of the exact gradient. Symbolic differentiation manipulates expression through mathematics, using known gradients of simple operations and a combination of the product and chain rules of derivatives. These are then combined to a single expression, which represents the gradient of the function which is then evaluated at locations of interest. It is often inappropriate to use since converting computer code into a single mathematical expression of the gradient can be difficult, especially if there are conditions such as; `if` and `else` statements. This can lead to inefficient code, [6].

Automatic differentiation is a series of techniques that allows one to efficiently and accurately evaluate derivatives of numeric functions expressed as computer programs, [59]. It exploits the fact that a computer program executes a series of elementary mathematical operations and functions. It is broadly split into two categories; forward mode, and reverse mode. We discuss reverse mode automatic differentiation briefly as it is the implementation we use in this work. For more detailed discussion on these approaches we refer the reader to [59].

Reverse mode automatic differentiation; calculates the (partial) derivative(s) at a point by using a forward and reverse pass through the elementary functions. We give a simple example to illustrate.

Suppose we wish to evaluate the partial derivatives of:

$$z = x_1 x_2 + cos(x_1),$$

with respect to x_1 and x_2 at $x_1 = 3$, $x_2 = 4$. Automatic differentiation begins by breaking this up into our component functions; exactly as a computer program would.

$$\begin{aligned} w_1 &= x_1, \\ w_2 &= x_2, \\ w_3 &= w_1 w_2, \\ w_4 &= \cos(w_1), \\ w_5 &= w_3 + w_4, \\ z &= w_5. \end{aligned}$$

The forward pass then simply evaluates these components at $x_1 = 3$ and $x_2 = 4$ and saves the results. That is:

$$\begin{aligned} w_1 &= x_1 = 3, \\ w_2 &= x_2 = 4, \\ w_3 &= w_1 w_2 = 12, \\ w_4 &= \cos(w_1) = -0.99, \\ w_5 &= w_3 + w_4 = 11.01, \\ z &= w_5 = 11.01. \end{aligned}$$

The backward pass then starts at the last node and evaluates the derivative with respect to its parent components. So firstly, $\frac{dz}{dw_5} = 1$. Then, w_5 depends linearly on w_3 and w_4, so we calculate the derivative of w_5 with respect to these; then can use the chain rule to accumulate this up to the derivative of z with respect to these components, keeping each. That is:

$$\begin{aligned} \frac{dz}{dw_3} &= \frac{dz}{dw_5}\frac{dw_5}{dw_3} = 1 \times 1, \\ \frac{dz}{dw_4} &= \frac{dz}{dw_5}\frac{dw_5}{dw_4} = 1 \times 1. \end{aligned}$$

We can continue on in this fashion. Then $\frac{dz}{dw_2} = \frac{dz}{dw_3}\frac{dw_3}{dw_2} = w_1 = 2$ as we know w_1 from our forward pass. Similarly $\frac{dz}{dw_1} = \frac{dz}{dw_3}\frac{dw_3}{dw_1} + \frac{dz}{dw_4}\frac{dw_4}{dw_1} = w_2 - \sin(w_1) = 4 - \sin(3)$. Evaluating this gives our partial derivates:

$$\begin{aligned}\frac{dz}{dx_1} &= 2,\\ \frac{dz}{dx_2} &= 4 - \sin(3).\end{aligned}$$

This procedure is essentially creating a graph of relatively simple computations, of which shared results can be utilised. Software is readily available which deals with the creating, storing, and efficient evaluation of such graphs. For example, the `tensorflow` library, [1].

The above illustration is easily extended to multivariate functions and can handle high numbers of partial derivatives, [59]. This is because we can compute all partial derivatives that we are interested in a single flow through both the forward and reverse path. As long as we have a few components with known derivatives, such as $\sin, \cos, \exp, \dots,$ we can then quickly evaluate derivatives of complex functions with respect to many parameters. This methodology does not suffer particularly from round off error due to never subtracting similar numbers, unlike in finite difference approaches which is the basis for the method.

In our implementation of the CPACE framework we make use of reverse mode automatic differentiation. We use the Python package, `tensorflow` which is capable of calculating gradients through this exact approach, [1]. This means our gradient based optimisation procedures for penalised B-spline smoothing and spatial kernel hyperparameters can be achieved, even when the number of parameters we need derivatives for is extremely high such as the case for the Gibbs kernel. We discuss this particular kernel more in the following section.

8.5 Gibbs Kernel

This Gibbs kernel is our main non-stationary kernel of interest in both our simulation study and application to the CESM-LE study; Chapter 6 and Chapter 7 respectively. The general form of which, previously discussed in Section 6.1.3 and given here for convenience is:

$$k\left(s_i, s_j\right) = \sum_{q=1}^{Q} \sqrt{\frac{2l_q(s_i)l_q(s_j)}{l_q(s_i)^2 + l_q(s_j)^2}} \exp\left(-\frac{(s_i - s_j)^2}{l_q(s_i)^2 + l_q(s_j)^2}\right),$$

where s_i, s_j are the points of evaluation, Q is the number of components in the kernel and l_q is the length scale function for component q. In this section we discuss the implementation of the length scale model, l_q for $q = 1, 2, \dots, Q$ that we use for the simulation and CESM-LE study.

We mentioned briefly in Sections 6.1, 7.2 that we use a neural network model for the length scale, l_q. This is not the only model that could be used. For example, [62] considers parametrising their length scale models using a latent Gaussian process. In our implementation of the Gibbs kernel, we have opted to use a neural network model for a couple of reasons. Firstly, given a sufficient structure they are flexible and can approximate a variety of surfaces. Secondly, they are somewhat agnostic to specific construction. That is as long as the network structure we use is sufficiently large, we can be somewhat lazy about exact network structure. That is not to say we should be lazy in constructing these hyperparameter models, but that given they are not our main area of interest in the CPACE framework, we should be happy that we can give a general construction which should perform reasonably well. Finally, they are fast to evaluate and fast to train, [1]. In particular, such a network model fits in naturally to the use of `tensorflow` to implement the CPACE framework.

Having decided on a structure of the neural network, for example in our simulation study we used a model with two hidden layers each containing 32 neurons using the ReLu activation function, there is still the practical consideration to ensure the length scale model it represents is positive. The simplest way to do this is to model the log length scale using the neural network and exponentiate the output. This is how we have implemented the Gibbs kernel in both the simulation study and application to the CESM-LE dataset.

Estimation of the parameters of the length scale model follows, as in Section 8.3, where they are estimated in conjunction with the other hyperparameters of the CPACE framework. This is where having an efficient and fast calculation of the log marginal likelihood and its gradient with respect to the hyperparameters becomes powerful. Since without this using a length scale model which relies on a large number of hyperparameters would be infeasible in terms of computational cost.

Finally, we give an indicative example of the output from an estimated length scale model using the above approach in Figure 8.3. This is taken from a single simulation for the pressure variable from the CESM-LE dataset. The full model results, which includes this simulation, are given in Section 7.4.1. We use this to highlight that the neural network approach gives a reasonable method for implementing the Gibbs kernel in the CPACE framework. As we can see, it captures interesting variation in the length scale which corresponds to distinct areas of the globe where pressure often has differing correlation structure.

Whilst we haven't attempted to show that using such a neural network model is the definitive way to implement the length scale models in the Gibbs kernel; through our simulation studies and application to CESM-LE dataset this approach has faired well. It often converges to reasonably interpretable structures and the results for the CPACE models utilising this implementation were comparable to those that used a more standard kernel.

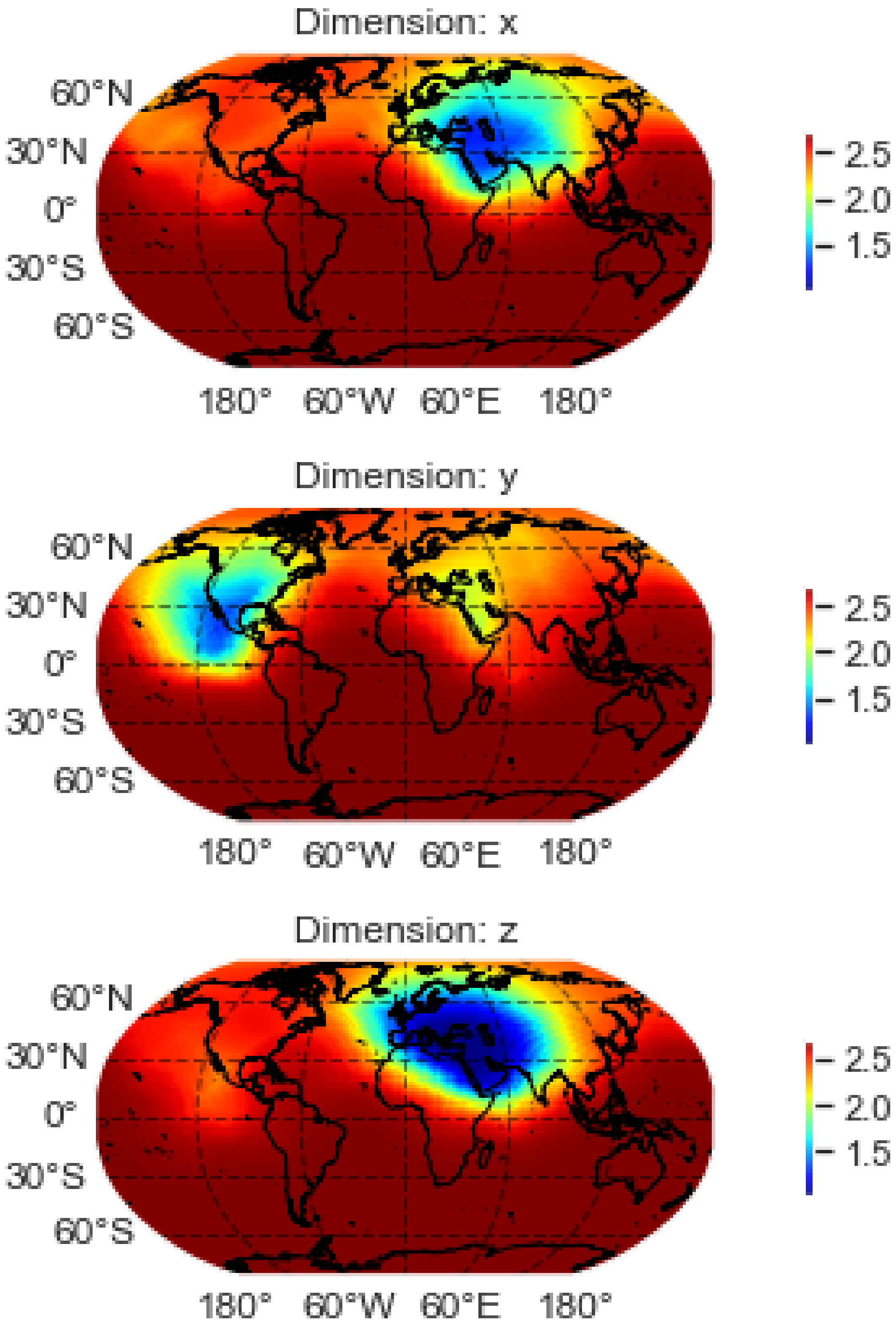


Fig. 8.3 An indicative example of the length scale model used in our implementation of the Gibbs kernel. The full length scale model for each dimension of the dataset is plotted in the corresponding graphs. Here we show the first component of the Gibbs kernel only and for a single spatial kernel of the CPACE framework. We use this to highlight the reconstructive ability of a relative simple neural network model for representing the full unknown length scale model used in the Gibbs kernel.

Chapter 9

Conclusion and Future Work

The aim of this thesis has been to consider the role that models based on functional data analysis techniques can be used to represent Earth Observation data. In particular, we have focused on developing methods which can help explain the dataset's spatial and temporal variation in a parsimonious way. We have provided a novel methodology, CPACE, which achieves this through building upon a functional principal components decomposition of the data to inform the temporal component of the spatial-temporal covariance function under a Gaussian process framework. We have highlighted this framework's performance through a simulation study and in application to the CESM-LE dataset.

In Chapter 1 we have detailed the specific subtleties of Earth Observation data; namely their inherent spatial and temporal dependency, and lattice like collection over the spatial domain. We have provided a description of such a dataset by viewing it as a collection of partially observed functional data. There are two such possible representations; the first being where the functional domain is time, the second being space. We have reviewed the literature regarding spatio-temporal methods applicable to EO data; where we highlight the need to often specify a parametric form of spatial-temporal covariance. We argue that this is often a complex task and often the parametric covariance function can obscure the model from being parsimonious; this motivates the use of a principal components analysis in our proposed CPACE methodology.

In Chapter 2 we have introduced our study dataset. The CESM-LE dataset is extremely well studied in the EO field, with many publications using this dataset as a study. We have shown the key characteristics of our variables of interest; namely pressure, temperature, wind speed and precipitation. We have detailed the necessary preprocessing to utilise this data which, in the most, was to scale down the dataset size to be more manageable. We have highlighted the various spatial and temporal correlations present in the variables which motivates the use of this dataset as a study of interest. In addition, we have 40 realisations of this dataset for each variable of interest. This makes this an ideal dataset to compare proposed models against as we have repeated studies.

In Chapter 4 we have considered our first approach for handling EO data in a functional data framework; namely considering the data as a collection of surfaces indexed over time. This is typically not the canonical form for functional data, since the functional domain considered is space. However, we have found that this approach works well when our objective is to forecast the functional time series. We have considered this approach on a number of simulated scenarios, with the functional time series approach performing at least as well a standard analysis, which uses time as the functional domain and ignores spatial dependency. We have found that this increase in performance was most noticeable during forecasting results, which suggest this technique may be preferable to standard analysis when this is our objective. However, under application to the CESM-LE dataset this technique is less successful. We reason this is due to the representation of our functional surfaces being over simplistic. That is; our use of smoothing splines to represent the functional principal component surfaces in our functional time series methodology was too restrictive when trying to recreate the complex surfaces of the CESM-LE dataset. In particular, we had difficulties representing small scale spatial variation. This resulted in overly smooth forecasts. This motivates our creation of the CPACE framework which aimed to overcome this.

We have presented the CPACE framework in Chapter 5. This framework considers the canonical form of functional data with time as the functional domain and introduces spatial dependency between functional observations. Here we have highlighted how to introduce spatial dependency between functional principal components; that can capture the spatial variation observed in the data. This builds on the PACE, [90], and SPACE ,[48], frameworks by considering the model as a larger Gaussian process where the covariance function is informed by the functional principal components. The advantage of this formulation is that we can utilise the nice properties of Gaussian process regression to both estimate hyperparameters and obtain predictive distributions rather than just a mean prediction. We have shown under certain assumptions that both the mean function and covariance surfaces can be estimated consistently with penalised spline regression. These assumptions essentially require a limit to the spatial dependency that can be observed in the dataset for these estimators to remain consistent. These assumptions; whilst strong, are often common in spatial statistics, [11].

We have tested the CPACE framework against a variety of simulated scenarios; where it performs at least as well as the PACE framework. We have shown that the CPACE framework gives the flexibility to accommodate a range of spatial dependencies through the use of standard spatial covariance kernels. Additionally it has the capability of allowing for more complex spatial kernels where appropriate. We have considered the use of a non-stationary Gibbs kernel as an example of such a kernel; and have shown the advantage that this can have on data simulated with non-stationary spatial dependency.

Finally, we have applied the CPACE framework to the CESM-LE dataset; in two studies. One across a large spatial scale with a high degree of sparsity in our observations and another across a smaller spatial scale with less sparsity. We have seen that the CPACE framework performs well and have exhibited its use with three different spatial kernels, the Matérn One Half, Matérn Three Halves, and the Gibbs kernel. We have found the estimation procedure for the model, detailed in Chapter 5 and Chapter 8, works well to find reasonable models for the spatial dependency. As well as this; the framework gives interpretable eigenfunctions which means the framework gives us an explanatory view of our chosen variables from the CESM-LE dataset. Coupled together with the good results for interpolation on relatively sparse data; the CPACE framework is a promising model for application to EO data.

Interestingly, in our application to the CESM-LE; the CPACE framework performs better on the pressure (PS) and temperature (TREFHT) variables. This we have reasoned is due to the smoother spatial variation of these variables compared to the wind speed (U10) and precipitation (TMQ) variables. The pressure and temperature variables exhibit some non-constant spatial variation over time; which is something the CPACE framework cannot capture as it assumes the spatial dependency is constant over time for each component. It may be possible to capture such variation by increasing the number of components of the model, however this increases the computational complexity of the model and thus the time taken for estimating the model.

9.1 Future Work

There are a number of interesting research directions that have not been explored in this thesis.

Considering first our initial attempt of modelling functional data through functional time series type models. An avenue of possible future work would be to consider differing spatial smoothing methodologies other than the penalised regression used in the functional time series methodology in Chapter 4. This may be through the use of local linear smoothers; such as those used in the estimation of the covariance surface in [90] or something more sophisticated. Since we found that the over smoothing of the eigenfunctions in the functional time series methodology was the major limiting factor; it would be of interest to see if this can be overcome through a differing methodology. This may then make such modelling more appealing to EO applications. This would be advantageous as we noted the appeal of this approach for forecasting rather than interpolation.

Another clear avenue for future work is to study our CPACE framework under new real world applications. We have limited our study of the CPACE framework to a variety of simulation studies and the application of the CESM-LE data. An interesting future application would be to apply this framework for image reconstruction of satellite imagery; for example those studied in [55] where the aim would be to reconstruct imagery obscured by clouds. This is a large growth area of EO data, [4], and thus the desire for interpretable models which can handle this data is increasing. Such future applications may require extra attention to the implementation of the model as described in Chapter 8.

Similarly, one may consider differing spatial kernels; or possibly introduce kernels which include extra covariate information in our CPACE framework. This could be achieved through appropriate extension of the spatial domain to be the spatial/covariate domain and introducing an appropriate kernel for such a domain. Whilst this isn't necessarily an extension of the CPACE framework such extensions may well help in modelling EO data as the additional covariate information could be used to create differing covariance for certain areas of interest. The added difficulty here would be the creation of valid spatial/covariate based kernels.

A particularly interesting future direction for the CPACE framework would be to consider extending it to allow for correlation between eigenfunctions. Currently; the framework assumes K independent score processes, $\zeta_k(\boldsymbol{s})$, which are then combined to form the full covariance model given in Equation (5.7). Adjusting this to allow for correlation between the score processes may allow the model to capture additional variation which is specific to certain locations of the spatial domain without having to increase the number of principal components, K. In essence this would be adjusting Equation (5.5) to:

$$\operatorname{cov}\left(\zeta_p, \zeta_q\right) = a_{pq},$$

where a_{pq} is some covariance kernel for $p, q = 1, 2, \ldots, K$. This full structure would be extremely flexible, and is mentioned in [48] in the context of their SPACE framework, however they note that this may be subject to unreliable estimation. Additionally there will be some difficulties in this approach as we would lose the diagonal structure of the full covariance; which may hinder the quick evaluation of the kernel, and the calculation of the log marginal likelihood which has been discussed in Chapter 8.

Similarly, one may consider how to extend the CPACE framework so that the spatial dependency can change over time. As we saw in the CESM-LE study in Chapter 7 the lack of this ability caused relatively poor performance in interpolating the wind speed variable. It would be interesting to consider capturing this by modifying the assumption that the spatial dependency for a single principal component is constant through time.

Another approach for future development of the CPACE framework is the extension to modelling multiple variables together. That is, essentially having a multivariate response variable, and considering the relationship between these response variables by extending the CPACE framework. Similar work has been derived for Gaussian process regression of multivariate responses, [9]. This would allow the joint modelling of multiple variables; and possibly using one to help inform the other for interpolation or forecasting. This again would have application to EO data, in particular satellite remote sensing, where often another source of imagery is used to infer missing observations of another, [55].

Finally; an interesting future direction for the implementation of the CPACE framework would be the refinement of the estimation procedure. The current implementation works well for datasets of similar size to those discussed in this thesis. However; the estimation procedure still uses the full log marginal likelihood calculation, albeit with some dimensionality reduction using the Woodbury identity. One could easily consider how to extend this procedure to utilise some of the more recent advances in scaleable Gaussian process modelling. One could consider the use of Global approximations such as subset of regressors or variational sparse approximations, [50], and intertwine this with the CPACE framework to make this more scaleable. Additionally, one could consider implementing the model with a desire to utilise GPU's rather than CPU's. This would specifically allow for more complex non-stationary kernels such as the Gibbs kernel built upon neural network models for the length scale components. This would certainly allow the model to be more applicable to a range of large datasets, which are becoming more common in the EO space.

References

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. (2016). TensorFlow: A System for Large-scale Machine Learning. *arXiv*, page 21.

[2] Abramowitz, M. and Stegun, I. A., editors (2013). *Handbook of Mathematical Functions: with Formulas, Graphs, and Mathematical Tables.* Dover books on mathematics. Dover Publ, New York, NY, 9. dover print.; [nachdr. der ausg. von 1972] edition.

[3] Aguilera, A. M., Ocaña, F. A., and Valderrama, M. J. (1999). Forecasting Time Series by Functional PCA. Discussion of Several Weighted Approaches. *Computational Statistics*, 14(3):443–467.

[4] Aschbacher, J. and Milagro-Pérez, M. P. (2012). The European Earth Monitoring (GMES) Programme: Status and Perspectives. *Remote Sensing of Environment*, 120:3–8.

[5] Aston, J. A. D., Tavakoli, S., and Pigoli, D. (2017). Tests for Separability in Nonparametric Covariance Operators of Random Surfaces. In Aneiros, G., G. Bongiorno, E., Cao, R., and Vieu, P., editors, *Functional Statistics and Related Fields*, pages 243–250. Springer International Publishing, Cham. Series Title: Contributions to Statistics.

[6] Baydin, A. G., Pearlmutter, B. A., Radul, A. A., and Siskind, J. M. (2018). Automatic Differentiation in Machine Learning: a Survey. *Journal of Machine Learning Research*, 18(153):1–43.

[7] Billingsley, P. (1995). *Probability and Measure.* Wiley series in probability and mathematical statistics. Wiley, New York, 3rd ed edition.

[8] Bjorck, A. (1996). *Numerical Methods for Least Squares Problems.* SIAM, Philadelphia.

[9] Chen, Z., Wang, B., and Gorban, A. N. (2020). Multivariate Gaussian and Student-t Process Regression for Multi-output Prediction. *Neural Comput & Applic*, 32(8):3005–3028.

[10] Cressie, N. and Huang, H.-C. (1999). Classes of Nonseparable, Spatio-Temporal Stationary Covariance Functions. *Journal of the American Statistical Association*, 94(448):1330–1339.

[11] Cressie, N. A. C. (2015). *Statistics for Spatial Data.* John Wiley & Sons, Inc, Hoboken, NJ, revised edition edition.

[12] Cressie, N. A. C. and Wikle, C. K. (2011). *Statistics for Spatio-Temporal Data.* Wiley series in probability and statistics. Wiley, Hoboken, N.J.

[13] Cummins, D. J., Filloon, T. G., and Nychka, D. (2001). Confidence Intervals for Nonparametric Curve Estimates: Toward More Uniform Pointwise Coverage. *Journal of the American Statistical Association*, 96(453):233–246.

[14] De Boor, C. (2001). *A Practical Guide to Splines: with 32 Figures.* Number v. 27 in Applied mathematical sciences. Springer, New York, rev. ed edition.

[15] Deb, S. and Tsay, R. S. (2019). Spatio-Temporal Models with Space-Time Interaction and Their Applications to Air Pollution Data. *STAT SINICA*.

[16] Dozat, T. (2016). Incorporating Nesterov Momentum into Adam. *ICLR 2016 workshop.*

[17] Fan, J., Gijbels, I., Hu, T.-C., and Huang, L.-S. (1996). A Study of Variable Bandwidth Selectiion for Local Polynomial Regression. *Statistica Sinica*, 6(1):113–127. Publisher: Institute of Statistical Science, Academia Sinica.

[18] Ferraty, F. and Vieu, P. (2006). *Nonparametric Functional Data Analysis: Theory and Practice.* Springer series in statistics. Springer, New York. OCLC: ocm70261207.

[19] Fletcher, R. (2008). *Practical Methods of Optimization.* A Wiley-Interscience publication. Wiley, Chichester, 2. ed., reprinted in paperback, june 2008 edition.

[20] Fuentes, M. (2006). Testing for Separability of Spatial–Temporal Covariance Functions. *Journal of Statistical Planning and Inference*, 136(2):447–466.

[21] Genton, M. G. (2001). Classes of Kernels for Machine Learning: A Statistics Perspective. *Journal of Machine Learning Research*, 2(Dec):299–312.

[22] George, B. and Aban, I. (2015). Selecting a Separable Parametric Spatiotemporal Covariance Structure for Longitudinal Imaging Data. *Statist. Med.*, 34(1):145–161.

[23] Gibbs, M. (1998). *Bayesian Gaussian Processes for Regression and Classification.* PhD thesis, University of Cambridge.

[24] Gneiting, T. (2002). Nonseparable, Stationary Covariance Functions for Space–Time Data. *Journal of the American Statistical Association*, 97(458):590–600.

[25] Guinness, J. and Fuentes, M. (2016). Isotropic Covariance Functions on Spheres: Some Properties and Modeling Considerations. *Journal of Multivariate Analysis*, 143:143–152.

[26] Harville, D. A. (1997). Determinants. In *Matrix Algebra From a Statistician's Perspective*, pages 179–208. Springer New York, New York, NY.

[27] Hastie, T., Tibshirani, R., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer series in statistics. Springer, New York, NY, 2nd ed edition.

[28] Higdon, D. (2002). Space and Space-Time Modeling using Process Convolutions. In Anderson, C. W., Barnett, V., Chatwin, P. C., and El-Shaarawi, A. H., editors, *Quantitative Methods for Current Environmental Issues*, pages 37–56. Springer London, London.

[29] Hooker, G. and Roberts, S. (2016). Maximal Autocorrelation Functions in Functional Data Analysis. *Stat Comput*, 26(5):945–950.

[30] Hooker, G., Roberts, S., and Shang, Lin, H. (2015). Maximal Autocorrelation Factors for Function-valued Spatial/Temporal Data. In *Weber, T., McPhee, M.J. and Anderssen, R.S. (eds) MODSIM2015, 21st International Congress on Modelling and Simulation.* Modelling and Simulation Society of Australia and New Zealand.

[31] Hore, A. and Ziou, D. (2010). Image Quality Metrics: PSNR vs. SSIM. In *2010 20th International Conference on Pattern Recognition*, pages 2366–2369, Istanbul, Turkey. IEEE.

[32] Hu, W., Fuglstad, G., and Castruccio, S. (2022). A Stochastic Locally Diffusive Model with Neural Network-based Deformations for Global Sea Surface Temperature. *Stat*, 11(1).

[33] Hurrell, J. W., Holland, M. M., Gent, P. R., Ghan, S., Kay, J. E., Kushner, P. J., Lamarque, J.-F., Large, W. G., Lawrence, D., Lindsay, K., Lipscomb, W. H., Long, M. C., Mahowald, N., Marsh, D. R., Neale, R. B., Rasch, P., Vavrus, S., Vertenstein, M., Bader, D., Collins, W. D., Hack, J. J., Kiehl, J., and Marshall, S. (2013). The Community Earth System Model: A Framework for Collaborative Research. *Bull. Amer. Meteor. Soc.*, 94(9):1339–1360.

[34] Hyndman, R. J. and Athanasopoulos, G. (2021). *Forecasting: principles and practice.* OTexts. OCLC: 1296327289.

[35] Hyndman, R. J. and Booth, H. (2008). Stochastic Population Forecasts using Functional Data Models for Mortality, Fertility and Migration. *International Journal of Forecasting*, 24(3):323–342.

[36] Hyndman, R. J. and Shahid Ullah, M. (2007). Robust Forecasting of Mortality and Fertility Rates: A Functional Data Approach. *Computational Statistics & Data Analysis*, 51(10):4942–4956.

[37] Hyndman, R. J. and Shang, H. L. (2009). Forecasting Functional Time Series. *Journal of the Korean Statistical Society*, 38(3):199–211.

[38] Iaco, S. D., Myers, D. E., and Posa, D. (2002). Nonseparable Space-Time Covariance Models: Some Parametric Families. *Mathematical Geology*, 34(1):23–42.

[39] Jolliffe, I. T. (2002). Choosing a Subset of Principal Components or Variables. In *Principal Component Analysis*, pages 111–149. Springer, New York, NY.

[40] Josse, J. and Husson, F. (2012). Selecting the Number of Components in Principal Component Analysis using Cross-validation Approximations. *Computational Statistics & Data Analysis*, 56(6):1869–1879.

[41] Kaiser, H. F. (1958). The Varimax Criterion for Analytic Rotation in Factor Analysis. *Psychometrika*, 23(3):187–200.

[42] Karhunen, K. (1946). Zur Spektraltheorie Stochastischer Prozesse. *Ann. Acad. Sci. Finnicae, Ser. A*, 1:34.

[43] Kay, J. E., Deser, C., Phillips, A., Mai, A., Hannay, C., Strand, G., Arblaster, J. M., Bates, S. C., Danabasoglu, G., Edwards, J., Holland, M., Kushner, P., Lamarque, J.-F., Lawrence, D., Lindsay, K., Middleton, A., Munoz, E., Neale, R., Oleson, K., Polvani, L., and Vertenstein, M. (2015). The Community Earth System Model (CESM) Large Ensemble Project: A Community Resource for Studying Climate Change in the Presence of Internal Climate Variability. *Bulletin of the American Meteorological Society*, 96(8):1333–1349.

[44] Khabbazan, S., Vermunt, P., Steele-Dunne, S., Ratering Arntz, L., Marinetti, C., van der Valk, D., Iannini, L., Molijn, R., Westerdijk, K., and van der Sande, C. (2019). Crop Monitoring Using Sentinel-1 Data: A Case Study from The Netherlands. *Remote Sensing*, 11(16):1887.

[45] Knott, G. D. (2000). *Interpolating Cubic Splines.* Number 18 in Progress in computer science and applied logic. Birkhäuser, Boston Basel Berlin.

[46] Li, M., Zhang, T., Chen, Y., and Smola, A. J. (2014). Efficient Mini-batch Training for Stochastic Optimization. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 661–670, New York New York USA. ACM.

[47] Lindgren, F., Rue, H., and Lindström, J. (2011). An Explicit Link between Gaussian Fields and Gaussian Markov Random Fields: the Stochastic Partial Differential Equation Approach: Link between Gaussian Fields and Gaussian Markov Random Fields. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(4):423–498.

[48] Liu, C., Ray, S., and Hooker, G. (2017). Functional Principal Component Analysis of Spatially Correlated Data. *Stat Comput*, 27(6):1639–1654.

[49] Liu, C., Ray, S., Hooker, G., and Friedl, M. (2012). Functional Factor Analysis for Periodic Remote Sensing Data. *Ann. Appl. Stat.*, 6(2).

[50] Liu, H., Ong, Y.-S., Shen, X., and Cai, J. (2020). When Gaussian Process Meets Big Data: A Review of Scalable GPs. *IEEE Trans. Neural Netw. Learning Syst.*, 31(11):4405–4423.

[51] Loève, M. (1946). Fonctions Aléatoires à Décomposition Orthogonale Exponentielle. *La Revue Scientifique*, 84:159–162.

[52] Lu, Z., Qu, Y., and Qiao, S. (2014). Geodetic Datum and Geodetic Control Networks. In *Geodesy*, pages 71–130. Springer Berlin Heidelberg, Berlin, Heidelberg.

[53] Lukas, M. A. (2006). Robust Generalized Cross-validation for Choosing the Regularization Parameter. *Inverse Problems*, 22(5):1883–1902.

[54] Lukas, M. A., De Hoog, F. R., and Anderssen, R. S. (2012). Performance of Robust GCV and Modified GCV for Spline Smoothing: Robust GCV and modified GCV criteria. *Scandinavian Journal of Statistics*, 39(1):97–115.

[55] Meraner, A., Ebel, P., Zhu, X. X., and Schmitt, M. (2020). Cloud removal in Sentinel-2 Imagery using a Deep Residual Neural Network and SAR-optical Data Fusion. *ISPRS Journal of Photogrammetry and Remote Sensing*, 166:333–346.

[56] Militino, A. F., Ugarte, M. D., and Pérez-Goya, U. (2018). An Introduction to the Spatio-Temporal Analysis of Satellite Remote Sensing Data for Geostatisticians. In Daya Sagar, B., Cheng, Q., and Agterberg, F., editors, *Handbook of Mathematical Geosciences*, pages 239–253. Springer International Publishing, Cham.

[57] Mitchell, M. W., Genton, M. G., and Gumpertz, M. L. (2006). A likelihood Ratio Test for Separability of Covariances. *Journal of Multivariate Analysis*, 97(5):1025–1043.

[58] Muro, J., Canty, M., Conradsen, K., Hüttich, C., Nielsen, A., Skriver, H., Remy, F., Strauch, A., Thonfeld, F., and Menz, G. (2016). Short-Term Change Detection in Wetlands Using Sentinel-1 Time Series. *Remote Sensing*, 8(10):795.

[59] Neidinger, R. D. (2010). Introduction to Automatic Differentiation and MATLAB Object-Oriented Programming. *SIAM Rev.*, 52(3):545–563.

[60] Oliver, C. and Quegan, S., editors (2004). *Understanding Synthetic Aperture Radar Images.* SciTech Publishing, Raleigh, NC.

[61] O'Sullivan, F. (1986). A Statistical Perspective on Ill-Posed Inverse Problems. *Statist. Sci.*, 1(4).

[62] Paciorek, C. J. and Schervish, M. J. (2006). Spatial modelling using a new class of nonstationary covariance functions. *Environmetrics*, 17(5):483–506.

[63] Ramsay, J. O. and Silverman, B. W. (2010). *Functional Data Analysis.* Springer Series in Statistics. Springer, New York, NY, 2. ed edition.

[64] Raspini, F., Bianchini, S., Ciampalini, A., Del Soldato, M., Solari, L., Novali, F., Del Conte, S., Rucci, A., Ferretti, A., and Casagli, N. (2018). Continuous, semi-automatic Monitoring of Ground Deformation using Sentinel-1 Satellites. *Sci Rep*, 8(1):7253.

[65] Rossi, R. E., Dungan, J. L., and Beck, L. R. (1994). Kriging in the Shadows: Geostatistical Interpolation for Remote Sensing. *Remote Sensing of Environment*, 49(1):32–40.

[66] Ruppert, D., Wand, M. P., and Carroll, R. J. (2003). *Semiparametric Regression.* Cambridge series in statistical and probabilistic mathematics. Cambridge University Press, Cambridge ; New York.

[67] Sampson, P. D. and Guttorp, P. (1992). Nonparametric Estimation of Nonstationary Spatial Covariance Structure. *Journal of the American Statistical Association*, 87(417):108–119.

[68] Schmidt, A. M. and Guttorp, P. (2020). Flexible Spatial Covariance Functions. *Spatial Statistics*, 37:100416.

[69] Shen, H., Li, X., Cheng, Q., Zeng, C., Yang, G., Li, H., and Zhang, L. (2015). Missing Information Reconstruction of Remote Sensing Data: A Technical Review. *IEEE Geosci. Remote Sens. Mag.*, 3(3):61–85.

[70] Shi, J. Q. and Choi, T. (2011). *Gaussian Process Regression Analysis for Functional Data.* CRC Press, Boca Raton, FL. OCLC: ocn491888782.

[71] Singha, S., Bellerby, T. J., and Trieschmann, O. (2013). Satellite Oil Spill Detection Using Artificial Neural Networks. *IEEE J. Sel. Top. Appl. Earth Observations Remote Sensing*, 6(6):2355–2363.

[72] Sra, S., Nowozin, S., and Wright, S. J., editors (2012). *Optimization for Machine Learning.* Neural information processing series. MIT Press, Cambridge, Mass. OCLC: ocn701493361.

[73] Stein, M. L. (1999). *Interpolation of Spatial Data: Some Theory for Kriging.* Springer series in statistics. Springer, New York.

[74] Switzer, P. and Green, A. (1984). Min/Max Autocorrelation Factors for Multivariate Spatial Imaging. *Department of Statistics, Stanford University, Stanford, CA.*, page 14. Technical Report No.6.

[75] Wahba, G. (1977). Practical Approximate Solutions to Linear Operator Equations When the Data are Noisy. *SIAM J. Numer. Anal.*, 14(4):651–667.

[76] Wahba, G. (1985). A Comparison of GCV and GML for Choosing the Smoothing Parameter in the Generalized Spline Smoothing Problem. *Ann. Statist.*, 13(4).

[77] Wahba, G. (1990). *Spline Models for Observational Data.* Number 59 in CBMS-NSF Regional Conference series in applied mathematics. Society for Industrial and Applied Mathematics, Philadelphia, Pa.

[78] Wang, Z. and Bovik, A. (2009). Mean Squared Error: Love it or leave it? A New Look at Signal Fidelity Measures. *IEEE Signal Process. Mag.*, 26(1):98–117.

[79] Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. (2004). Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. on Image Process.*, 13(4):600–612.

[80] Williams, C. K. I. and Rasmussen, C. E. (2006). *Gaussian Processes for Machine Learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass. OCLC: ocm61285753.

[81] Wold, S., Esbensen, K., and Geladi, P. (1987). Principal Component Analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52.

[82] Wood, S. N. (2006a). *Generalized Additive Models: An Introduction with R*. Texts in statistical science. Chapman & Hall/CRC, Boca Raton, FL. OCLC: ocm64084887.

[83] Wood, S. N. (2006b). Low-Rank Scale-Invariant Tensor Product Smooths for Generalized Additive Mixed Models. *Biometrics*, 62(4):1025–1036.

[84] Wood, S. N. (2008). Fast Stable Direct Fitting and Smoothness Selection for Generalized Additive Models. *J Royal Statistical Soc B*, 70(3):495–518.

[85] Wood, S. N. (2011). Fast Stable Restricted Maximum Likelihood and Marginal Likelihood Estimation of Semiparametric Generalized Linear Models: Estimation of Semiparametric Generalized Linear Models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(1):3–36.

[86] Wood, S. N. (2017). P-splines with Derivative Based Penalties and Tensor Product Smoothing of Unevenly Distributed Data. *Stat Comput*, 27(4):985–989.

[87] Woodbury, M. A. (1950). Inverting Modified Matrices. In *Memorandum Rept. 42, Statistical Research Group*. Princeton Univ.

[88] Xiao, L. (2019). Asymptotic Theory of Penalized Splines. *Electron. J. Statist.*, 13(1).

[89] Xiao, L. (2020). Asymptotic Properties of Penalized Splines for Functional Data. *Bernoulli*, 26(4).

[90] Yao, F., Müller, H.-G., and Wang, J.-L. (2005). Functional Data Analysis for Sparse Longitudinal Data. *Journal of the American Statistical Association*, 100(470):577–590.

[91] Zhang, C., Li, W., and Travis, D. J. (2009). Restoration of Clouded Pixels in Multispectral Remotely Sensed Imagery with Cokriging. *International Journal of Remote Sensing*, 30(9):2173–2195.