Bayesian RKHS-based methods in functional regression

Master's thesis

by

Antonio Coín Castro

Under the supervision of

José Ramón Berrendero Díaz

Antonio Cuevas González

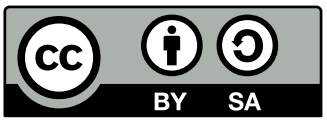
A document submitted in partial fulfillment of the requirements for the

Master's degree in Data Science

at

Universidad Autónoma de Madrid

September, 2022



This document has been typeset using a modification of the *mimosis* template at https://github.com/Pseudomanifold/latex-mimosis. The LaTeX source code is openly available at https://www.github.com/antcc/tfm, while the Python source code for the experiments carried out can be found in https://www.github.com/antcc/rk-bfr.

Abstract

This work is framed within the field of Functional Data Analysis, a branch of statistics in which the objects of interest are random functions in a functional space, instead of, say, random points in $\mathbb{R}^p$. Due to the infinite-dimensional nature of the data, the most common L^2-models for functional linear and logistic regression present some complications that require further simplifications, usually in the form of regularization or dimensionality reduction.

In this thesis we propose a novel Bayesian approach for functional linear and logistic regression models, based on the theory of reproducing kernel Hilbert spaces (RKHS's). These models build upon the RKHS associated with the covariance function of the underlying stochastic process, and can be viewed as a finite-dimensional approximation to the classical functional regression paradigm. The corresponding functional model (or the functional logistic equation in the case of binary response) is determined by a function living on a dense subspace of the RKHS of interest, which has a tractable parametric form based on linear combinations of the kernel. By imposing a suitable prior distribution on this space, we can perform data-driven inference via standard Bayes methodology. The posterior distribution can be estimated through Markov chain Monte Carlo methods, which do not require a complete specification of the posterior density.

We derive several prediction strategies from the approximate posterior distribution, including a Bayesian-motivated variable selection procedure. We show through a comprehensive set of experiments that these methods are competitive against other usual alternatives in terms of predictive performance, both in simulated examples and real data sets. Overall, our proposed model is simple with regard to interpretation and feasible with regard to implementation, while also enjoying the added flexibility of an ambient Bayesian framework.

KEYWORDS: functional data, linear regression, logistic regression, reproducing kernel Hilbert space, Bayesian inference, Markov chain Monte Carlo.

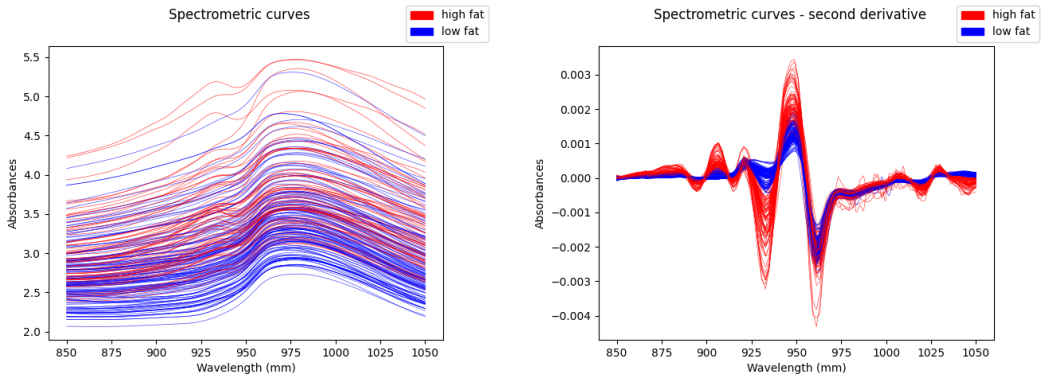
Contents

1 Introduction

Over the last few decades, situations involving data in the form of functions have become commonplace in many statistical scenarios, as more and more information is available worldwide with an ever-increasing level of granularity in the measurements. In particular, functional data problems are far from unheard of in the data science and machine learning community, since they have attracted the attention of researchers and practitioners equally. Medical data, weather indicators or stock exchange indices are examples of elements that benefit from a functional treatment, where the observations are regarded as single entities rather than as a conglomerate of individual points.

Under a functional framework, the objects of interest are *random functions* instead of random points in a finite-dimensional space. While in principle the functional data could be simply regarded as a discretized vector in a very high dimension (and indeed such a discretization is performed in practice), there are often many advantages in taking into account the functional nature of the data, ranging from modeling the possibly high correlation among points that are close in the domain, to extracting information that may be hidden in the derivatives of the function in question. Thus, the general idea is to assume the existence of an underlying sufficiently smooth function that corresponds to each (possibly noisy) functional observation, even though we only record it on a finite grid of points.

To see what this kind of data looks like, Figure 1a shows an example of a functional data set, which is known in the literature as the Tecator data set, and whose elements represent near-infrared absorbance curves of meat samples. The objective here is to predict the fat content based on this absorbance spectrum, separating the samples into those with "high" and "low" fat content. At first glance it does not seem that the trajectories contain much relevant information to help classify the samples. However, after a suitable smoothing of the data (e.g. by representing each function in a Fourier basis) we can take the derivatives of the curves. In this case, after differentiating twice a clearer pattern emerges (see Figure 1b), one from which inference and prediction will surely be easier.



(a) Original curves

(b) Second derivatives

Figure 1: Curves in the Tecator data set and their second derivatives (after smoothing).

In recent years numerous proposals have arisen on how to suitably deal with functional data, all of them encompassed under the term Functional Data Analysis (FDA), which essentially explores statistical techniques to process, model and make inference on data varying over a continuum. A partial survey on such techniques and methods is Cuevas (2014), while a more detailed exposition of the theory and applications can be found for example in Ramsay and Silverman (2005), Hsing and Eubank (2015) or the book by Horváth and Kokoszka (2012). As the name suggests, FDA techniques are heavily inspired by functional analysis tools and methods: Hilbert spaces, orthonormal systems, linear operators, and so on. In particular, a notion that also intersects with the classical theory of machine learning and pattern recognition, and that has gained traction in recent years, is that of reproducing kernel Hilbert spaces (RKHS's). We will demonstrate throughout this work how these spaces of functions possess properties that allow for an efficient treatment of functional data.

On the other hand, Bayesian inference methods are ubiquitous in the realm of statistics, and their usual non-parametric approach also makes use of random functions, though in a slightly different manner than in the FDA context. However, the two methodologies can certainly interact and benefit from one another, as we intend to show in this thesis. We will be particularly interested in Markov chain Monte Carlo (MCMC) methods, which allow us to approximate an arbitrary posterior distribution through a function proportional to its density.

In this work we are concerned with functional linear and logistic regression models, that is, situations where the goal is to predict a continuous or dichotomous variable from functional observations. Even though these problems can be formally stated with almost no differences from their finite-dimensional counterparts, there are some fundamental challenges as well as some subtle drawbacks that emerge as a result of working in infinite dimensions. Moreover, we will concentrate our efforts on the case in which the response is a scalar, though function-on-function and function-on-scalar regression are also interesting scenarios widely explored in the literature. To set a common framework, throughout this work we will consider a scalar response variable Y (either continuous or binary) which has some dependence on a stochastic L^2-process $X = X(t) = X(t, \omega)$ with trajectories in $L^2[0, 1]$ (i.e. a process with finite second moments and whose realizations are square-integrable functions indexed on $[0, 1]$). The underlying probability space $(\Omega, \mathscr{A}, \mathbb{P})$ is not important. We will further suppose that X is centered, that is to say, its mean function $m(t) = \mathbb{E}[X(t)]$ vanishes for all $t \in [0, 1]$. In addition, we will tacitly assume the existence of a *labeled* data set $\mathscr{D}_n = \{(X_i, Y_i) : i = 1, \dots, n\}$ of independent observations from (X, Y), where the functional observations are recorded on a common finite grid on $[0, 1]$. Our ultimate aim will be to accurately predict the response corresponding to unlabeled samples from X. Figure 2 depicts a typical data set used in functional regression, and we already saw in Figure 1 what a functional classification data set may look like.

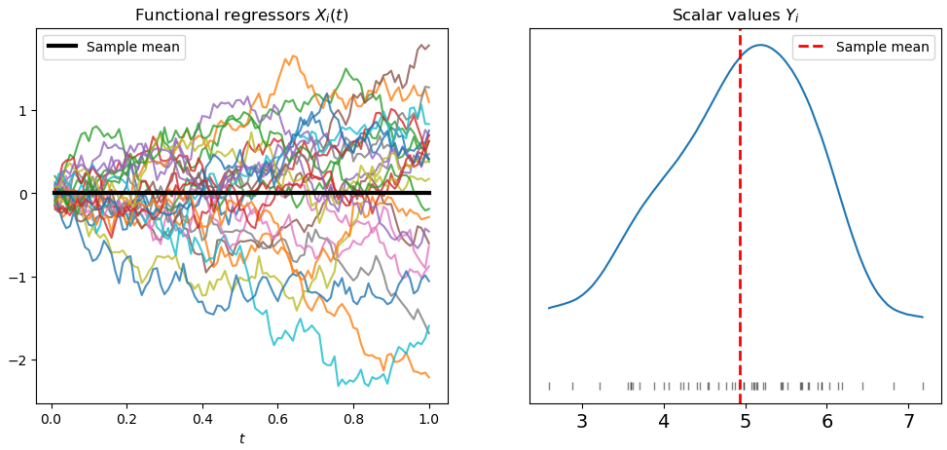


Figure 2: Simulated data set for functional regression. On the left we have $n = 50$ functional observations on an equispaced grid of $N = 100$ points on $[0, 1]$. To each observation corresponds a real number; the distribution of these responses is shown on the right.

1.1 Objectives and scope

We list below the main objectives of this work in no particular order.

1. To perform a brief but thorough literature review that contextualizes functional linear and logistic regression within statistics and machine learning, exploring the predominant models and techniques.
2. To propose a novel RKHS-based functional model for linear and logistic regression that builds on existing work and focuses on simplicity, both in terms of interpretation and implementation.
3. To describe and implement a general Bayesian approach for parameter estimation within the suggested model.
4. To introduce the tools needed to specify the proposed model and put it into practice, mainly reproducing kernel Hilbert spaces and Markov chain Monte Carlo methods.
5. To carry out an extensive experimental study to test these new models and compare them to existing methods, both in simulations and in real-world scenarios.

It is beyond the scope of this thesis to provide a complete review of FDA as a whole, or to delve too deeply into the details and inner workings of most models and techniques mentioned. Nevertheless, references are often provided throughout the text to point the interested reader towards more specialized resources.

1.2 Structure overview

In Chapter 2 we summarize the relevant literature related to our problem, along with a review of the basics of RKHS's and MCMC methods. Chapter 3 is devoted to explaining the Bayesian methodology and the functional regression models we propose. Then, in Chapter 4 we present a short discussion of theoretical and computational details that have led to the concrete specification of the model, as well as some validation techniques. The empirical results of the experimentation are contained in Chapter 5. Lastly, the conclusions drawn from this work and future paths of research are reviewed in Chapter 6.

2 Background and related work

FDA is undoubtedly an active area of research, which finds applications in a wide variety of fields, such as biomedicine, finance, meteorology or chemistry (see for example Ullah and Finch, 2013). Accordingly, there are many recent contributions on how to tackle functional data problems, both from a theoretical and practical standpoint. Chief among them is the approach of reducing the problem to a finite-dimensional one, for example using a truncated basis expansion or spline interpolation methods (e.g. Aguilera and Aguilera-Morillo, 2013; Müller and Stadtmüller, 2005). At the same time, much effort has also been put into the task of building a sound theoretical basis for FDA, generalizing different concepts to the infinite-dimensional framework. Examples of this endeavor include the definition of centrality measures and depth-based notions for functional data (e.g. Cuevas et al., 2007; Fraiman and Muniz, 2001; López-Pintado and Romo, 2009), an ANOVA test for functional data (Cuevas et al., 2004), a purely functional partial least squares algorithm (Delaigle and Hall, 2012b), a functional Mahalanobis distance (e.g. Berrendero et al., 2020a; Galeano et al., 2015), or an extension of Fisher's discriminant analysis for function-valued random elements (e.g. James and Hastie, 2001; Shin, 2008), among many others. Interestingly enough, these last two are examples of situations in which a RKHS-based approach provides useful insights.

Another technique found in the related literature is the use of Gaussian processes to model the functional behavior of the data (see for instance Shi and Choi, 2011). These ideas extend the theory of Gaussian process regression in classical finite-dimensional settings (e.g. Rasmussen, 2004), providing an alternative Bayesian approach to functional inference and prediction problems. Additional non-parametric methods for functional prediction and classification were notably explored in Ferraty and Vieu (2006).

L^2-models

The most common scalar-on-function linear regression model is the classical L^2-model, widely popularized since the first edition (1997) of the monograph by Ramsay and Silverman (2005). It can be seen as a generalization of the usual finite-dimensional model, replacing the scalar product in $\mathbb{R}^p$ for that of the functional space $L^2[0,1]$:

$$Y = \alpha_0 + \langle X, \beta \rangle + \varepsilon = \alpha_0 + \int_0^1 X(t)\beta(t)\,dt + \varepsilon, \tag{2.1}$$

where $\alpha_0 \in \mathbb{R}$, ε is a random error term independent from X with $\mathbb{E}[\varepsilon] = 0$, and the functional slope parameter $\beta = \beta(\cdot)$ is assumed to be a member of the infinite-dimensional space $L^2[0,1]$. A careful rearrangement of (2.1) shows that the model is equivalently expressed as $\Delta = \mathscr{K}\beta$, where $\Delta(\cdot)$ is the cross-covariance function of X and Y, and $\mathscr{K}$ is the covariance operator of the process X (to be defined in Section 2.1). This expression is a continuous analog of the normal equations that arise in finite-dimensional regression settings, and since the operator $\mathscr{K}$ is non-invertible in general, questions of existence and uniqueness of β need further study (see Cardot and Sarda, 2011).

Technical details aside, the inference on β is also hampered by the fact that $L^2[0,1]$ is an extremely wide space that contains many non-smooth or ill-behaved functions, so that any estimation procedure involving optimization on it would typically be hard. Although it can be tempting to simply discretize the observed values on a grid and proceed with standard multiple linear regression, this would result in an under-determined model in which the estimated parameters lose meaning. This happens essentially because the number of available parameters is infinite, but the number of equations is finite (see Ramsay and Silverman, 2005, Sec. 15.2). Thus, some regularization or dimensionality reduction techniques are needed for parameter estimation; see Reiss et al. (2017) for a summary of several widespread methods.

A common inference strategy is to expand both X and β on a certain data-driven orthonormal basis of $L^2[0,1]$, say $\{\phi_j\}$, up to a certain value $p \in \mathbb{N}$:

$$X(t) = \sum_{j=1}^{p} Z_j \phi_j(t), \qquad \beta(t) = \sum_{j=1}^{p} \beta_j \phi_j(t).$$

In this way, model (2.1) simplifies to

$$Y = \alpha_0 + \sum_{j=1}^{p} \beta_j Z_j,$$

which can then be solved in the usual manner. For example, when the chosen basis is composed of eigenfunctions of the covariance operator $\mathscr{K}$, this method is known as Functional Principal Component Regression (FPCR). Another class of methods focus on selecting an *a priori* basis for β (e.g. a spline basis) and solving a penalized least squares problem. In particular, for a truncated basis representation of the form $\beta(t) = \phi(t)'b$, with $\phi(t) = (\phi_1(t), \dots, \phi_p(t))'$ and $b \in \mathbb{R}^p$, a typical minimization problem would be

$$\underset{(\alpha_0, b) \in \mathbb{R} \times \mathbb{R}^p}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - \alpha_0 - \int_0^1 x_i(t)[\phi(t)'b]\, dt \right)^2 + \lambda \Omega(b) \right\},$$

where Ω is some penalty function and $\lambda > 0$ is a regularization parameter.

Turning our attention to logistic regression, a similar L^2-based functional logistic equation can be derived for the binary classification problem via the logistic function:

$$\mathbb{P}(Y = 1 \mid X) = \frac{1}{1 + \exp\{-\alpha_0 - \langle X, \beta \rangle\}}, \tag{2.2}$$

where $\alpha_0 \in \mathbb{R}$ and $\beta \in L^2[0, 1]$. In this situation, the most common way of estimating the slope function β is via its Maximum Likelihood Estimator (MLE). However, not only do the same complications as in the linear regression model apply in this situation, but there is also the additional problem that in functional settings the MLE does not exist with probability one under fairly general conditions (see Berrendero et al., 2022).

RKHS models

In spite of the apparent generality of model (2.1), it can be shown that it is not flexible enough to include "simple" finite-dimensional models based on linear combinations of the marginals of the process, such as $Y = \alpha_0 + \beta_1 X(t_1) + \cdots + \beta_p X(t_p) + \varepsilon$ for some constants $\beta_j \in \mathbb{R}$ and instants $t_j \in [0, 1]$; see Berrendero et al. (2020b) for additional details on this. It turns out that in both linear and logistic scenarios a natural and more general alterna-

tive to the L^2-model is the so-called Reproducing Kernel Hilbert Space (RKHS) model, which instead assumes the unknown functional parameter to be a member of the RKHS associated with the covariance function of the process X, making use of the scalar product of that space. As we will show later on, not only is this model simpler and arguably easier to interpret, but it also constrains the parameter space to smoother and more manageable functions. In fact, it does include a model based on finite linear combinations of the marginals of X as a particular case, which is especially appealing to practitioners confronted with functional data problems due to its simplicity. These RKHS-based models and their idiosyncrasies have been explored in Berrendero et al. (2019, 2020b) in the functional linear regression setting, and in Berrendero et al. (2022) for the case of functional logistic regression.

There are other proposals for models that change the habitat of the functional parameter, and there are even some in which β is supposed to live in a RKHS, most notably Yuan and Cai (2010). However, theirs are arbitrary RKHS's that do not directly exploit the relation with the process X (though the authors do use this connection to derive some asymptotic properties of their estimators), and therefore the approach is somewhat different. Incidentally, the RKHS associated with X also has some interesting properties that contribute to shed light on the near-perfect classification phenomenon for functional data, described by Delaigle and Hall (2012a) and further examined for example in the works of Berrendero et al. (2018) or Torrecilla et al. (2020).

A major aim of this work is to motivate these recently-proposed RKHS models inside the functional framework, while also providing efficient techniques to apply them in practice. Our main contribution is the proposal of a Bayesian approach to parameter estimation within the aforementioned RKHS models, in which a prior distribution is imposed on the unknown functional parameter to obtain a posterior distribution after seeing the data. Although setting a prior distribution on a functional space is generally a hard task, the specific parametric formulation of the RKHS models we propose greatly facilitates this (see Chapter 3 for details). A similar Bayesian scheme has recently been explored in Grollemund et al. (2019), albeit not within a RKHS framework.

Another set of techniques extensively studied in this context are variable selection methods, which aim to select the marginals $\{X(t_j)\}$ of the process that better summarize it according to some optimality criterion. As it happens, some variable selection methods have already been proposed in the RKHS framework (see for example Berrendero

et al., 2019; Bueno-Larraz and Klepsch, 2019), but in general they have their own dedicated algorithms and procedures. As will become apparent in the forthcoming chapters, given the nature of our suggested Bayesian model we can easily isolate the marginal posterior distribution corresponding to a finite set of points $\{t_j\}$, and thus provide a Bayesian-motivated variable selection process along with the other prediction methods that naturally arise within our model. In this way, in addition to making predictions about the input data, we can evaluate exactly which marginals of the functional explanatory variable contain the most relevant information. These points-of-impact selection models for functional predictors have also been considered in the literature; see Poß et al. (2020), Berrendero et al. (2016) or Ferraty et al. (2010) by way of illustration. Another example of a related strategy is the work of James et al. (2009), in which the authors propose a method to estimate $\beta(t)$ in such a way that it is exactly zero over some regions in the domain (a sort of "region selection" algorithm).

Bayesian inference

The term Bayesian inference usually refers to a wide class of methods that to some extent employ Bayes' theorem to update the initial probability assigned to a hypothesis (or a parameter) when new information is available. It can be seen as a general tool for modeling situations or problems that involve uncertainty; some examples include Bayesian hierarchical modeling, Bayesian regression or even Bayesian neural networks (see e.g. Bishop, 2006; Murphy, 2012). Specifically, in this work we will be interested in performing parameter estimation in a Bayesian framework, so that pre-existing information and beliefs about the parameters can be incorporated into the model in question.

One difficulty found in almost all Bayesian methods is that the posterior distribution, the main object of interest, is usually intractable due to the integral that appears as the normalizing constant in Bayes' rule. A way to bypass this limitation is to use conjugate distributions, where it can be rigorously proven that a certain combination of likelihood and prior distributions produces a posterior distribution in the same family as the prior. However, unless conjugate priors are used, a closed-form expression of the posterior is generally unattainable and some type of approximation is required. Apart from basic numerical integration, some well-performing methods in this regard are variational inference approaches (e.g. Blei et al., 2017) and MCMC methods (e.g. Brooks et al., 2011).

The former are based on approximating the posterior by another distribution restricted to a certain parametric family so that the Kullback-Leibler divergence between them is minimized, whereas the latter are iterative methods that directly provide approximate samples of the posterior distribution.

On a separate note, there are some recent works that tackle Bayesian inference from a functional perspective, mostly in relation to the distribution over functions induced by a Bayesian neural network. Some examples are the functional Bayesian neural networks proposed by Sun et al. (2019), variational implicit processes (Ma et al., 2019) and their deep variants (Ortega et al., 2022), or the functional variational inference techniques suggested in Ma and Hernández-Lobato (2021). Moreover, there are approaches to functional regression from a Bayesian perspective that impose a Gaussian process prior on the functional parameter β (e.g Lian et al., 2016), and it turns out that the RKHS's corresponding to these Gaussian processes, described for example in Van Der Vaart and Van Zanten (2008), are useful when studying contraction rates of the posterior distribution.

2.1 Reproducing kernel Hilbert spaces

In this section we present a brief exposition of some basic concepts regarding reproducing kernel Hilbert spaces. Although there are quite a few ways of introducing these spaces, we adopt a probabilistic point of view that will set the stage for the development of the subsequent theory. Although the following ideas can be extended to complex functions, we restrict ourselves to real-valued functions for the sake of simplicity; for a more detailed account, a good reference is the book by Berlinet and Thomas-Agnan (2004). We start by defining the concept of kernel functions, which are the foundation of these spaces.

Definition 2.1. We say that a function of two variables $K : \mathcal{T} \times \mathcal{T} \to \mathbb{R}$ is positive semidefinite[1] if

$$\sum_{i,j=1}^{p} a_i a_j K(t_i, t_j) \geq 0$$

for any $p \in \mathbb{N}$, any $(a_1, \dots, a_p) \in \mathbb{R}^p$ and any $(t_1, \dots, t_p) \in \mathcal{T}^p$. Note that this is equivalent to saying that the matrix $(K(t_i, t_j))_{i,j}$ is positive semidefinite for any choice of $p \in \mathbb{N}$ and $(t_1, \dots, t_p) \in \mathcal{T}^p$.

[1] Sometimes in the literature these functions are known simply as positive definite.

We will be interested mainly in positive semidefinite functions that are symmetric, which are usually referred to as *kernel functions*, and which happen to be the class of covariance functions of second order stochastic processes (Berlinet and Thomas-Agnan, 2004, Th. 27). Let us now show how a RKHS arises from a kernel function.

Suppose $X = X(t)$ is a L^2-stochastic process with trajectories in $L^2[0,1]$, and for simplicity assume $\mathbb{E}[X(t)] = 0$ for all $t \in [0,1]$. Let us denote by $K(t,s) = \mathbb{E}[X(t)X(s)]$ the covariance function of the process X. To construct the RKHS $\mathscr{H}(K)$ associated with the covariance function, we start by defining the functional vector space $\mathscr{H}_0(K)$ of all finite linear combinations of evaluations of K, that is,

$$\mathscr{H}_0(K) = \left\{ f \;:\; f(\cdot) = \sum_{i=1}^{p} a_i K(t_i, \cdot),\ p \in \mathbb{N},\ a_i \in \mathbb{R},\ t_i \in [0,1] \right\}. \tag{2.3}$$

Note that, as subsets, $\mathscr{H}_0(K) \subset L^2[0,1]$. However, this new space can be endowed with an inner product different from the one induced by $L^2[0,1]$, namely

$$\langle f, g \rangle_K = \sum_{i,j} a_i b_j K(t_i, s_j),$$

for $f(\cdot) = \sum_i a_i K(t_i, \cdot)$ and $g(\cdot) = \sum_j b_j K(s_j, \cdot)$. We show below that it is well defined, but before we note that functions in this space satisfy the so-called *reproducing property*:

$$f(t) = \langle K(t, \cdot), f \rangle_K, \quad \text{for all } t \in [0,1].$$

In particular, $K(t,s) = \langle K(t,\cdot), K(s,\cdot) \rangle_K$, which can be understood as saying that the kernel *reproduces* itself, hence the term "reproducing kernel".

Proposition 2.2. *$(\mathscr{H}_0(K), \langle \cdot, \cdot \rangle_K)$ is an inner product space.*

Proof. Firstly, consider $f(\cdot) = \sum_i a_i K(t_i, \cdot)$ and $g(\cdot) = \sum_j b_j K(s_j, \cdot)$, and observe that

$$\langle f, g \rangle_K = \sum_j b_j f(s_j) = \sum_i a_i g(t_i).$$

These equalities show that $\langle f, g \rangle_K$ depends only on f and g through their values, so it is independent of their representation in $\mathscr{H}_0(K)$. From this expression (and also from the original definition) it is straightforward to check linearity and symmetry, while positive

semidefiniteness is a direct consequence of K being a covariance function, and thus positive semidefinite. It remains to prove that $\langle f, f\rangle_K = 0$ implies $f = 0$. Indeed, for all $t \in [0,1]$ and $\varepsilon \in \mathbb{R}$, we have

$$0 \leq \langle f + \varepsilon K(t,\cdot), f + \varepsilon K(t,\cdot)\rangle_K = \langle f, f\rangle_K + 2\varepsilon\langle f, K(t,\cdot)\rangle_K + \varepsilon^2 K(t,t).$$

Now, if $\langle f, f\rangle_K = 0$, by letting $\varepsilon \to 0^{\pm}$ it follows that $f(t) = \langle f, K(t,\cdot)\rangle_K$ necessarily vanishes for all t, as desired. □

At this point, $\mathscr{H}(K)$ is defined to be the completion of $\mathscr{H}_0(K)$ under the norm induced by the scalar product $\langle\cdot,\cdot\rangle_K$, which informally amounts to adding all the limits of Cauchy sequences in $\mathscr{H}_0(K)$, turning it into a genuine Hilbert space with the inner product extended accordingly. Then, it is immediate to see that the reproducing property is retained. An important consequence is that $\mathscr{H}(K)$ is a space of actual functions and not of equivalence classes, since the values of the functions at particular points are in fact relevant, unlike in L^2-spaces.

Indeed, another way of characterizing a RKHS is via the continuity of all the *evaluation operators*, i.e., $\delta_t(f) := f(t)$ for $f \in \mathscr{H}(K)$ and $t \in [0,1]$. In this case, the interpretation is that if two functions are close in the RKHS norm, they are also pointwise close. Although in this definition there is no explicit mention of the kernel, it can be recovered through the Riesz representation theorem. We summarize below some interesting properties of $\mathscr{H}(K)$.

Proposition 2.3. *The following properties hold for the space $\mathscr{H}(K)$:*

(i) The evaluation operator δ_t is bounded for all $t \in [0,1]$.
(ii) Norm convergence implies pointwise convergence, and if K is continuous, it also implies uniform convergence.
(iii) If K is m-times continuously differentiable, then so is every function $f \in \mathscr{H}(K)$. In particular, if K is continuous, every function $f \in \mathscr{H}(K)$ is continuous.

Proof. To prove *(i)*, simply note that for any $t \in [0,1]$ and $f \in \mathscr{H}(K)$ the Cauchy-Schwarz inequality and the reproducing property tell us that

$$|\delta_t(f)| = |f(t)| = |\langle f, K(t,\cdot)\rangle_K| \leq \|K(t,\cdot)\|_K \|f\|_K = \sqrt{K(t,t)}\|f\|_K,$$

and consequently $\|\delta_t\| \leq \sqrt{K(t,t)}$. To see *(ii)*, observe that for any $t \in [0,1]$ we have

$$|f_n(t) - f(t)| = |\delta_t(f_n - f)| \leq \|\delta_t\| \|f_n - f\|_K,$$

as we have shown in *(i)* that the evaluation operator is bounded. Moreover, if K is continuous on its (compact) domain, $\|\delta_t\| \leq \sup_s \sqrt{K(s,s)} = M < \infty$ for all t, so the convergence is indeed uniform. Finally, we prove the second statement in *(iii)*, and refer the reader to Saitoh and Sawano (2016, Th 2.6) for a complete proof. If K is continuous, then it is clear that every function in $\mathscr{H}_0(K)$ is continuous. Let us now fix an arbitrary $f \in \mathscr{H}(K)$. By definition, there exists a sequence $\{f_n\} \subset \mathscr{H}_0(K)$ of continuous functions such that $\|f_n - f\|_K \to 0$. But then $f_n \to f$ uniformly by *(ii)*, and therefore f is continuous as the uniform limit of continuous functions. Note that this statement remains valid, with a slightly different proof, even when the underlying domain is not compact. □

For the sake of completeness, it is worth mentioning that the RKHS associated with a kernel function is unique, and indeed the Moore-Aronszajn theorem (e.g. Berlinet and Thomas-Agnan, 2004, Th. 3) states that there is a one-to-one correspondence between kernel functions and reproducing kernel Hilbert spaces. Let us now illustrate the definition in a few simple cases; see Saitoh and Sawano (2016, Ch. 1) for more involved examples.

Example 2.4. If X is the standard Brownian motion on $[0,1]$, it is well known that the associated covariance function is $K_{\text{bm}}(t,s) = \min\{t,s\}$. The corresponding RKHS is given by (Janson, 1997, Ex. 8.19):

$$\mathscr{H}(K_{\text{bm}}) = \{f \,:\, f \text{ is absolutely continuous, } f(0) = 0 \text{ and } f' \in L^2[0,1]\},$$

with inner product $\langle f, g\rangle_{K_{\text{bm}}} = \int f'g'$.

Example 2.5 (Finite-dimensional RKHS)**.** Setting aside our stochastic process framework for a moment, we can conceive examples outside of $\mathscr{T} = [0,1]$. Consider a vector-valued random variable $X = (X_1, \dots, X_p)$ with non-singular covariance matrix Σ. In this case the index set would be $\mathscr{T}_p = \{1, \dots, p\}$, so identifying functions on $\mathscr{T}_p$ with points in the p-dimensional Euclidean space, we have $\mathscr{H}(\Sigma) = \mathbb{R}^p$, with inner product $\langle x, y\rangle_\Sigma = x\Sigma^{-1}y$.

Example 2.6 (Non-RKHS Hilbert space)**.** As we pointed out before, the space of square integrable functions $L^2(\mathbb{R})$ *is not* a RKHS. There are many ways of seeing this; for example, in this space norm convergence does not imply pointwise convergence. Another possibility is to observe that the would-be reproducing kernel must be the Dirac delta function:

$$f(s) = \int_{-\infty}^{\infty} \delta(t, s) f(t)\, dt, \quad s \in \mathbb{R}.$$

However, $\delta(t, \cdot) \notin L^2(\mathbb{R})$.

The covariance operator

Note that, as expected, the covariance function of X plays a crucial role in characterizing the associated RKHS. An integral operator closely related to this covariance function is the so-called covariance operator, namely $\mathscr{K} : L^2[0,1] \to L^2[0,1]$ given by

$$\mathscr{K} f(\cdot) = \int_0^1 K(t, \cdot) f(t)\, dt, \quad f \in L^2[0,1],$$

which is self-adjoint and compact when K is continuous (e.g. Hsing and Eubank, 2015, Th. 4.6.2), so for the remaining of this work we will indeed suppose that K is continuous. As it turns out, the covariance function admits a spectral decomposition in terms of the eigenvalues and eigenfunctions of this operator, a sort of continuous generalization of the eigendecomposition of a symmetric positive semidefinite matrix.

Theorem 2.7 (Mercer's theorem)**.** *Let K be a continuous kernel on $[0,1]^2$. Then, there exists an orthonormal basis $\{\phi_j\}$ of $L^2[0,1]$ consisting of eigenfunctions of $\mathscr{K}$, whose corresponding eigenvalues are all positive[2] and form a non-increasing sequence $\{\lambda_j\} \to 0$, and such that*

$$K(t, s) = \sum_{j=1}^{\infty} \lambda_j \phi_j(t) \phi_j(s)$$

for all t and s, where the convergence is absolute and uniform.

In connection with our RKHS theory, it can be shown that the set $\{\sqrt{\lambda_j}\phi_j\}$ constitutes an orthonormal basis of $\mathscr{H}(K)$ (see e.g. Cucker and Zhou, 2007, Sec. 4.4). Furthermore,

[2] Even though the spectral theorem only guarantees that the eigenvalues of $\mathscr{K}$ are non-negative, there is no loss of generality in assuming $\lambda_j > 0$ for all $j \geq 1$; see Remark 3 in Cucker and Smale (2001, Ch. 3).

a similar orthogonal decomposition holds for the stochastic process X, an expansion that lies at the base of many L^2 functional regression models.

Theorem 2.8 (Karhunen-Loève expansion)**.** *With the assumptions and notations of Theorem 2.7, the centered process $X = X(t)$ with covariance function K admits the quadratic-mean representation*

$$X(t) = \sum_{j=1}^{\infty} \zeta_j \phi_j(t),$$

where the convergence is in $L^2(\Omega)$ and uniform in t. The ζ_j are independent zero-mean random variables with $\mathbb{E}[\zeta_i \zeta_j] = \delta_{ij}\lambda_j$, given explicitly by the formula

$$\zeta_j = \int_0^1 X(t)\phi_j(t)\, dt, \quad j \geq 1.$$

In addition to these theorems, it is worth mentioning that the covariance operator $\mathscr{K}$ provides several alternative definitions of $\mathscr{H}(K)$. For example, the RKHS can be identified with the image of the operator's square root, i.e., $\mathscr{H}(K) = \mathscr{K}^{1/2}(L^2[0,1])$, with inner product $\langle f, g\rangle_K = \left\langle \mathscr{K}^{-1/2}(f), \mathscr{K}^{-1/2}(g)\right\rangle$. Furthermore, we can think of the norm in $\mathscr{H}(K)$ as an L^2-like regularized norm, since this space can also be seen as the set $\mathscr{H}(K) = \{f \in L^2[0,1] : \sum_j \lambda_j^{-1}\langle f, \phi_j\rangle^2 < \infty\}$, with corresponding inner product $\langle f, g\rangle_K = \sum_j \lambda_j^{-1}\langle f, \phi_j\rangle\langle g, \phi_j\rangle$. Note that, since $\{\lambda_j\}$ tends to zero, this definition highlights the fact that functions in $\mathscr{H}(K)$ are smooth, not only in that they inherit the regularity of the kernel K, but also in the sense that their components in an orthonormal basis, namely $\langle f, \phi_j\rangle$, need to vanish quickly. A proof of the previous results can be consulted for example in Berlinet and Thomas-Agnan (2004, Sec. 3.2).

Loève's isometry

Until now we have seen that $\mathscr{H}(K)$ is a space related to X only through its covariance function. Nevertheless, we will now hopefully clarify that this space can indeed be seen as *the* Hilbert space inherently associated with the process X itself. Let us shift the perspective momentarily and consider the space $L^2(\Omega)$ of all zero-mean random variables with finite second moment, endowed with the usual norm $\|U\|^2 = \mathbb{E}[U^2]$. Then, the linear span of X in $L^2(\Omega)$ is given by

$$\mathscr{L}_0(X) = \left\{ U \in L^2(\Omega) \,:\, U = \sum_{i=1}^{p} a_i X(t_i) \,:\, p \in \mathbb{N},\ a_i \in \mathbb{R},\ t_i \in [0,1] \right\}.$$

It is well known that its completion in $L^2(\Omega)$, denoted $\mathscr{L}(X)$, is a Hilbert space, sometimes called the Hilbert space generated by X. In the words of Berlinet and Thomas-Agnan (2004), "$\mathscr{L}(X)$ *contains the random variables attainable by linear operations, including limits, on the measurements of the process*". As it happens, via *Loève's isometry* (Loève, 1948) one can establish a congruence Ψ_X between $\mathscr{H}(K)$ and $\mathscr{L}(X)$ (see Lemma 1.1 in Lukić and Beder, 2001). This isometry is essentially the continuous extension of the correspondence

$$\sum_{i=1}^{p} a_i X(t_i) \longleftrightarrow \sum_{i=1}^{p} a_i K(t_i, \cdot), \tag{2.4}$$

and can be formally defined, in terms of its inverse, as $\Psi_X^{-1}(U)(t) = \mathbb{E}[UX(t)]$ for all $U \in \mathscr{L}(X)$. Thus, $\mathscr{H}(K)$ can be regarded as an isometric copy of $\mathscr{L}(X)$, an approach that is often useful in statistics.

However, despite the close connection between the process X and the space $\mathscr{H}(K)$, special care must be taken when dealing with concrete realizations of the process. It can be shown that under rather general conditions the trajectories of X *do not belong* to the corresponding RKHS with probability one (e.g. Lukić and Beder, 2001, Cor. 7.1; Pillai et al., 2007, Th. 11). For instance, the trajectories of a Brownian motion are known to be nowhere differentiable, but as shown in Example 2.4, the elements of the associated RKHS are differentiable almost everywhere. A heuristic argument to justify this fact, given in Wahba (1990), is as follows: consider the Karhunen-Loève expansion of X, i.e.,

$$X(t) = \sum_{j=1}^{\infty} \zeta_j \phi_j(t),$$

and the truncated version $X_N(t)$ up to the N-th term. On the one hand, for each fixed t we have $X_N(t) \to X(t)$ in the quadratic mean sense (by the Karhunen-Loève theorem), but on the other hand observe that

$$\mathbb{E}\big[\|X_N(\cdot)\|_K^2\big] = \mathbb{E}\left[\sum_{j=1}^{N} \frac{\zeta_j^2}{\lambda_j}\right] = N \to \infty \quad (N \to \infty).$$

As a consequence, the expression $\langle x, f\rangle_K$ is ill-defined and lacks meaning when x is a realization of X. However, following Parzen's approach in his seminal work (e.g. Parzen, 1961, Th. 4E), we can leverage Loève's isometry and identify $\langle x, f\rangle_K$ with the image $\Psi_x(f) := \Psi_X(f)(\omega)$, for $x = X(\omega)$ and $f \in \mathscr{H}(K)$. This notation, viewed as a formal extension of the inner product, often proves to be useful and convenient. Some properties stemming from this interpretation are stated below (see Parzen, 1961, p. 974).

Proposition 2.9. *For every $t \in [0,1]$ and $f, g \in \mathscr{H}(K)$, the following relations hold:*

(i) $\langle X, K(t,\cdot)\rangle_K = X(t)$, *a particular reproducing property.*
(ii) $\mathbb{E}[\langle X, f\rangle_K] = 0$.
(iii) $\mathbb{E}[\langle X, f\rangle_K \langle X, g\rangle_K] = \langle f, g\rangle_K$.

Note that statement *(iii)* above provides yet another characterization of the inner product in $\mathscr{H}(K)$, where $\langle X, f\rangle_K$ and $\langle X, g\rangle_K$ are understood as the random variables representing f and g in $\mathscr{L}(X)$, respectively.

Applications in machine learning

Although not the main concern of this work, the theory of reproducing kernels and the associated kernel methods find applications in many areas of machine learning. For example, the well-known *kernel trick* can be seen as a specific usage of the reproducing property. First, a feature map $\Phi(x) \in \mathscr{H}$ is applied to the objects in the space of interest, transforming them into richer elements in a RKHS. Then, many computations can be efficiently carried out in an implicit manner by means of the corresponding kernel

$$K(x, y) = \langle \Phi(x), \Phi(y)\rangle_{\mathscr{H}}.$$

The fact that one does not need to explicitly compute $\Phi(x)$ is especially relevant for kernels with a simple expression but a possibly complex (e.g. infinite-dimensional) feature space, such as the Gaussian kernel $K(x, y) = \exp(-\gamma\|x - y\|^2)$.

Another example of the use of RKHS's is in the context of regularization problems, where the celebrated *representer theorems* provide a way of obtaining closed-form solutions to a penalized optimization problem. For instance, learning techniques that rely on empirical risk minimization (e.g. Vapnik, 1991) benefit from one such result by Schölkopf et al. (2001), stated below.

Theorem 2.10 (Generalized representer theorem). *Consider a RKHS of functions $\mathcal{H}(K)$ defined over a non-empty set $\mathcal{X}$, a training set $\{(x_i, y_i)\}_{i=1}^n \subset (\mathcal{X} \times \mathbb{R})^n$, an arbitrary error function $\mathcal{R} : (\mathcal{X} \times \mathbb{R}^2)^n \to \mathbb{R} \cup \{\infty\}$, and a strictly increasing real-valued function $\Omega : [0, \infty) \to \mathbb{R}$. Then, any minimizer f^* of the regularized empirical risk*

$$\mathcal{R}(\{(x_i, y_i, f(x_i))\}_{i=1}^n) + \Omega(\|f\|_K)$$

is of the form $f^*(\cdot) = \sum_{i=1}^n a_i K(x_i, \cdot)$, where $a_i \in \mathbb{R}$ for all $i = 1, \dots, n$.

2.2 Markov chain Monte Carlo

In the context of Bayesian inference, there are many situations in which we would like to sample from a distribution without using its explicit closed-form density, either because we do not know it, or because it is computationally difficult to do so. In these cases, we could work with a function $f(x)$ proportional to our target density $\pi(x)$. More to the point, in a Bayesian framework the posterior distribution is often intractable due to the normalizing integral constant, but we do know that *posterior* $\propto$ *prior* $\times$ *likelihood*.

When $x \in \mathbb{R}$, we can use Monte Carlo algorithms such as *rejection sampling*. In a nutshell, this method samples uniformly from the area under $f(x)$ by means of an auxiliary density that encompasses this area. Specifically, one needs to find a density function g with supp $f \subset$ supp g and $f \leq Mg$ for some $M > 1$. Then, sampling from g and accepting the proposal, say x, with probability $f(x)/Mg(x)$ yields an approximate set of samples from f. Nevertheless, these kinds of algorithms are rapidly affected by the so-called *curse of dimensionality* (see the schematic in Figure 3), so new techniques are needed when sampling from multidimensional distributions. This is where Markov chain Monte Carlo (MCMC) methods come into play.

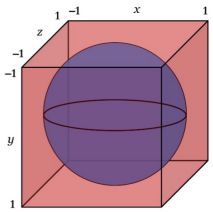


Figure 3: To cover the volume of $f(x)$ in higher dimensions we intuitively need a volume that grows exponentially with the dimension itself.

MCMC methods are based on the iterative construction of a Markov chain whose stationary distribution is the objective distribution $\pi(x)$. In this way, we can take the samples of a sufficiently advanced chain as approximate realizations of $X \sim \pi(x)$. When it comes to building the chains, the transition probabilities are assigned in such a way that regions with higher density with respect to $\pi(x)$ are favored (by means of the known proportional function $f(x)$). The dimensionality issues are somewhat mitigated, and on top of that there are specific tuning techniques to tackle them. However, the samples obtained from this procedure are not independent, though they can be made more or less so by *thinning* the chain and considering only one every few samples.

Another advantage of these methods is that they recover the marginal distributions directly. Suppose that after a successful MCMC run we have M multivariate approximate samples $\{\theta^{(m)*} = (\theta_1^{(m)*}, \dots, \theta_p^{(m)*})\}$ of a joint p-dimensional distribution. The marginal distribution of each variable, which would be theoretically computed as

$$\pi(\theta_i) = \int \pi(\theta)\, d\theta_1 \cdots d\theta_{i-1}\, d\theta_{i+1} \cdots d\theta_p,$$

can be approximated by just retaining the samples $\{\theta_i^{(m)*}\}$ corresponding to θ_i.

Metropolis-Hastings

The Metropolis-Hastings algorithm (Metropolis et al., 1953) is arguably the best known algorithm in this context. As a matter of fact, it is actually more of a general framework from which many other methods are derived. In its original formulation, it relies on a symmetric *proposal* distribution $g(x'|x_t)$ that generates a proposal for the next state given the current one (e.g. a Gaussian distribution centered on x_t). Approximate samples from the intended distribution can then be generated through the following iterative acceptance-based procedure:

1. Generate a proposal $x' \sim g(x'|x_t)$.
2. Accept the proposal with probability $\alpha = \min\{1, f(x')/f(x_t)\}$. If accepted, set $x_{t+1} = x'$; otherwise set $x_{t+1} = x_t$.

Figure 4 shows a schematic representation of the process. Note that the acceptance ratio $f(x')/f(x_t) = \pi(x')/\pi(x_t)$ is a measure of how more likely is x' to be a sample from π

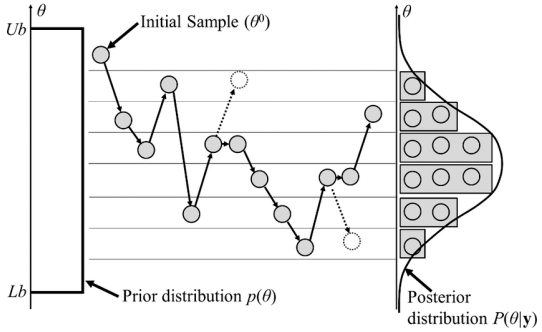


Figure 4: Representation of the Metropolis-Hastings algorithm (taken from Lee et al. (2015)). The discarded proposals are depicted as dashed circles.

than x_t. This specific acceptance value is also chosen so that the detailed balance equations of the chain are satisfied, i.e., $\pi(x)P(x'|x) = \pi(x')P(x|x')$, where $P(x|y)$ is the transition probability from state x to state y. These conditions essentially guarantee that π is the stationary distribution of the chain; see Robert et al. (1999) for additional details. An immediate generalization consists on choosing a general (possibly non-symmetric) jump distribution g, in which case the acceptance probability is rewritten as

$$\alpha = \min\left\{1, \frac{f(x')g(x_t \mid x')}{f(x_t)g(x' \mid x_t)}\right\}.$$

Although this algorithm performs well on a wide range of problems, it can be difficult to find a suitable jump distribution when the number of dimensions is high. If we know the distribution of each variable conditional on the rest of them, an alternative approach to sampling is *Gibb's algorithm* (S. Geman and D. Geman, 1984), which is a particular case of Metropolis-Hastings that considers separate samples for each dimension. Specifically, in each step we get unidimensional samples *in turn* (e.g. via rejection sampling) of each variable conditional on the rest of them, using always the more up-to-date values available:

$$\pi(x_l^{(i+1)} \mid x_1^{(i+1)}, \dots, x_{l-1}^{(i+1)}, x_{l+1}^{(i)}, \dots, x_L^{(i)}), \quad l = 1, \dots, L.$$

Lastly, there are several methods based on Metropolis-Hastings that are the subject of active research, both from a theoretical and computational standpoint. A good example are *Hamiltonian Monte Carlo* methods (e.g. Neal, 2011), which are a family of algorithms that introduce gradient information to guide the proposals in the sample space.

They are based on Hamiltonian dynamics, and in general they improve convergence speed due to a more intelligent choice of new points on the chain. With automatic differentiation being developed in the last few decades, these algorithms have become computationally feasible and have been widely adopted by the scientific community.

Affine-invariant ensemble sampler

An interesting and often desirable property of sampling algorithms is that they be *affine-invariant*, which means that they regard two distributions that differ in an affine transformation, say $\pi(x)$ and $\pi_{A,b}(Ax + b)$, as equally difficult to sample from. This is useful when one is working with very asymmetrical or skewed distributions, for an affine transformation can turn them into ones with simpler shapes.

Generally speaking, a MCMC algorithm can be described through a function R as $\Lambda(t+1) = R(\Lambda(t), \xi(t), \pi)$, where $\Lambda(t)$ is the state of the chain at instant t, π is the objective distribution, and $\xi(t)$ is a sequence of i.i.d. random variables that represent the random behavior of the chain. With this notation, the affine-invariance property can be characterized as (Goodman and Weare, 2010):

$$R(A\lambda + b, \xi(t), \pi_{A,b}) = AR(\lambda, \xi(t), \pi) + b,$$

for all A, b and λ, and almost all $\xi(t)$. This means that if we fix a random generator and run the algorithm twice, one time using π and starting in $\Lambda(0)$ and a second time using $\pi_{A,b}$ with initial point $\Gamma(0) = A\Lambda(0) + b$, then $\Gamma(t) = A\Lambda(t) + b$ for all t. In Goodman and Weare (2010) the authors consider an ensemble of samplers with the affine invariance property. Specifically, they work with a set $\Lambda = (\Lambda_1, \dots, \Lambda_L)$ of *walkers*, where $\Lambda_l(t)$ represents an individual chain at time t. At each iteration, an affine-invariant transformation is used to find the next point, which is constructed using the current values of the rest of the walkers (similar to Gibb's algorithm), namely the *complementary ensemble*

$$\Lambda_{-l}(t) = \{\Lambda_1(t+1), \dots, \Lambda_{l-1}(t+1), \Lambda_{l+1}(t), \dots, \Lambda_L(t)\}, \quad l = 1, \dots, L.$$

To maintain the affine invariance and the joint distribution of the ensemble, the walkers are advanced one by one following a Metropolis-Hastings acceptance scheme. There are mainly two types of moves.

STRETCH MOVE. For each walker $1 \leq l \leq L$ another walker $\Lambda_j \in \Lambda_{-l}(t)$ is chosen at random, and the proposal is constructed as

$$\Lambda_l(t) \to \Gamma = \Lambda_j + Z(\Lambda_l(t) - \Lambda_j),$$

where $Z \overset{i.i.d.}{\sim} g(z)$ satisfying the symmetry condition $g(z^{-1}) = zg(z)$. In particular, the suggested density is

$$g_a(z) \propto \begin{cases} \frac{1}{\sqrt{z}}, & \text{if } z \in [a^{-1}, a], \\ 0, & \text{otherwise.} \end{cases}, \quad a > 1.$$

Supposing $\mathbb{R}^p$ is the sample space, the corresponding acceptance probability (chosen so that the detailed balance equations are satisfied) is:

$$\alpha = \min\left\{1,\ Z^{p-1}\frac{\pi(\Gamma)}{\pi(\Lambda_l(t))}\right\}.$$

WALK MOVE. For each walker $1 \leq l \leq L$ a random subset $S_l \subseteq \Lambda_{-l}(t)$ with $|S_l| \geq 2$ is selected, and the proposed move is

$$\Lambda_l(t) \to \Gamma = \Lambda_l(t) + W,$$

where W is a normal distribution with mean 0 and the same covariance as the sample covariance of all walkers in S_l. The acceptance probability in this case is just the Metropolis ratio, i.e., $\alpha = \min\{1, \pi(\Gamma)/\pi(\Lambda_l(t))\}$.

From a computational perspective, the Python library *emcee*[3] (Foreman-Mackey et al., 2013) provides a parallel implementation of this algorithm. The idea is to divide the ensemble Λ into two equally-sized subsets $\Lambda^{(0)}$ and $\Lambda^{(1)}$, and then proceed on each iteration in the following alternate fashion:

1. Update *all* walkers in $\Lambda^{(0)}$ through one of the available moves explained above, using $\Lambda^{(1)}$ as the complementary ensemble.
2. Use the new values in $\Lambda^{(0)}$ to update $\Lambda^{(1)}$.

In this way the detailed balance equations are still satisfied, and each of the steps can benefit from the computing power of an arbitrary number of processors (up to $L/2$).

[3] https://emcee.readthedocs.io

3 Bayesian methodology for RKHS-based functional regression models

In this chapter we present the precise functional models and Bayesian methodologies explored in this work. The RKHS-based functional models under consideration (see Berrendero et al., 2019, 2022) are those obtained by substituting the functional parameter $\beta \in L^2[0,1]$ for $\alpha \in \mathscr{H}(K)$, replacing also the scalar product $\langle X, \beta \rangle$ for $\langle X, \alpha \rangle_K$ in the L^2-models (2.1) and (2.2). However, to further simplify things we will follow a parametric approach and suppose that α is in fact a member of the dense subspace $\mathscr{H}_0(K)$ defined in (2.3), i.e.:

$$\alpha(\cdot) = \sum_{j=1}^{p} \beta_j K(t_j, \cdot), \text{ for some } p \in \mathbb{N},\ \beta_j \in \mathbb{R} \text{ and } t_j \in [0,1]. \tag{3.1}$$

Moreover, as we said before, with a slight abuse of notation we will understand the expression $\langle x, \alpha \rangle_K$ as $\Psi_x(\alpha)$, where $x = X(\omega)$ and Ψ_x is Loève's isometry. Hence, taking into account that $\Psi_X(K(t,\cdot)) = X(t)$ by definition (see (2.4)), when α is as in (3.1) we can write $\langle x, \alpha \rangle_K \equiv \sum_{j=1}^{p} \beta_j x(t_j)$.

In this way we get a simpler, finite-dimensional approximation of the functional RKHS model, which we argue reduces the overall complexity of the model while still capturing most of the relevant information. When it comes to parameter estimation, a direct optimization of some loss function would probably require a tailored algorithm that took into account the whole functional trajectories $x(t)$ to select the appropriate times t_j. Indeed, such an idea is explored in Berrendero et al. (2022) for the logistic regression case, where the authors propose a "greedy max-max" method reminiscent of the EM algorithm

that alternates between estimating the coefficients and the time instants through a maximum likelihood approach.

At this point we propose to follow a Bayesian approach to estimate the parameters of the model, which we believe is in line with the idea of simplicity we pursue, and also introduces an additional layer of flexibility into the model. In this way, we can include problem-specific information through the use of prior distributions, and on top of that, this method works almost unaltered for both linear and logistic regression models. The general idea will be to impose a prior distribution on the functional parameter to eventually derive a posterior distribution after incorporating the available sample information.

In view of (3.1), to set a prior distribution on the unknown function α (that is, a prior distribution on the functional space $\mathscr{H}_0(K)$) it suffices to first consider a discrete distribution on p, and then impose p-dimensional continuous prior distributions on the coefficients β_j and the times t_j given p. Thanks to this parametric approach, the challenging task of setting a prior distribution on a space of functions is considerably simplified, while simultaneously not constraining the model to any specific distribution (in contrast to, for instance, Gaussian process regression methods). Moreover, note that starting from a probability distribution $\mathbb{P}_0$ on $\mathscr{H}_0(K)$ we can obtain a probability distribution $\mathbb{P}$ on $\mathscr{H}(K)$ merely by defining $\mathbb{P}(B) = \mathbb{P}_0(B \cap \mathscr{H}_0(K))$ for all Borel sets B. Consequently, our simplifying assumption on α is not actually very restrictive, since any prior distribution on $\mathscr{H}_0(K)$ can be directly extended to a prior distribution on $\mathscr{H}(K)$.

However, after some initial experimentation we found that, for practical and computational reasons, the value of p (the dimensionality of the model) is best fixed beforehand in a suitable way; see Chapter 4 for details. Thus, we will regard only the β_j and t_j as free parameters, and search for our functional parameter in the space

$$\mathscr{H}_{0,p}(K) = \left\{ \sum_{j=1}^{p} \beta_j K(t_j, \cdot) \ : \ \beta_j \in \mathbb{R},\ t_j \in [0,1] \right\}. \tag{3.2}$$

Even though we actually work on $\mathscr{H}_{0,p}(K)$, the discrete parameter p can still be selected in several meaningful ways that make use of the available data, and the set of feasible values is not very large in practice. Moreover, we could think of this approach as imposing a degenerate prior distribution on p, so it is in a way a particular case of the more general model discussed above.

In any case, after selecting a suitable prior distribution $\pi(\theta)$ for the finite-dimensional parameter vector θ (which will be specified shortly), we can resort to Bayes' theorem to perform the inference step, which in the case of i.i.d. samples amounts to

$$\pi(\theta \mid \mathscr{D}_n) \propto \left(\prod_{i=1}^{n} \pi(Y_i \mid X_i, \theta) \right) \pi(\theta). \tag{3.3}$$

In Sections 3.1 and 3.2 we proceed to specify the parameter spaces, prior distributions and concrete models for $\pi(Y|X,\theta)$ considered in the case of functional linear regression and functional logistic regression, respectively. Even though in (3.3) we have omitted the possibly intractable integral related to the normalizing constant, sampling from the (approximate) posterior distribution can still be accomplished via MCMC methods (see Chapter 4 for implementation details).

3.1 Functional linear regression

In the case of functional linear regression, the simplified RKHS model considered is

$$Y = \alpha_0 + \langle X, \alpha \rangle_K + \varepsilon = \alpha_0 + \sum_{j=1}^{p} \beta_j X(t_j) + \varepsilon, \tag{3.4}$$

where $\alpha(\cdot) = \sum_{j=1}^{p} \beta_j K(t_j, \cdot) \in \mathscr{H}_{0,p}(K)$, $\alpha_0 \in \mathbb{R}$, and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ is an error term independent from X. This model is essentially a finite-dimensional approximation from a functional perspective to the more general RKHS model that assumes $\alpha \in \mathscr{H}(K)$, proposed in Berrendero et al. (2019).

When p is fixed, the parameter space becomes $\Theta_p = \mathbb{R}^p \times [0,1]^p \times \mathbb{R} \times \mathbb{R}^+$, and in the sequel a generic element of this $(2p+2)$-dimensional space will be denoted by $\theta = (\beta_1, \ldots, \beta_p, t_1, \ldots, t_p, \alpha_0, \sigma^2) \equiv (b, \tau, \alpha_0, \sigma^2)$. Before proceeding further, observe that we can rewrite model (3.4) in a more explicit and practical fashion in terms of the available sample information in $\mathscr{D}_n$. Indeed, for $\theta \in \Theta_p$ the reinterpreted model assumes the form

$$Y_i \mid X_i, \theta \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(\alpha_0 + \sum_{j=1}^{p} \beta_j X_i(t_j),\ \sigma^2 \right), \quad i = 1, \ldots, n. \tag{3.5}$$

It is worth mentioning that the model remains linear in the sense that it fundamentally involves a random variable $\langle X, \alpha \rangle_K = \Psi_X(\alpha)$ belonging to the linear span of the process X in $L^2(\Omega)$. Also, note that given the time instants t_j, the model becomes a multiple linear model with the $X(t_j)$ as scalar covariates. As a matter of fact, this model is particularly suited as a basis for variable selection methods, and furthermore the general RKHS model entails the classical L^2-model (2.1) under certain conditions (see Berrendero et al., 2020b, Sec. 3). In addition, this model could be easily extended to the case of several covariates via an expression of type $Y = \alpha_0 + \left\langle X^1, \alpha_1 \right\rangle_K + \cdots + \left\langle X^q, \alpha_q \right\rangle_K + \varepsilon$. In that case, as argued in Grollemund et al. (2019) for a similar situation, if we were to set a prior distribution on all the parameters involved, we could recover the full posterior by looking alternately at the posterior distribution of each covariate conditional on the rest of them.

The Bayesian approach: prior and posterior

The prior distribution suggested for the parameter vector $\theta \in \Theta_p$ is given by

$$\begin{aligned} \pi(\alpha_0, \sigma^2) &\propto 1/\sigma^2, \\ \tau &\sim \mathcal{U}([0,1]^p), \\ b \mid \tau, \sigma^2 &\sim \mathcal{N}_p(b_0, g\sigma^2 \underbrace{(\mathcal{X}_\tau' \mathcal{X}_\tau + \eta I)}_{G_\tau}^{-1}), \end{aligned} \tag{3.6}$$

where I is the identity matrix, $\mathcal{X}_\tau$ is the data matrix $(X_i(t_j))_{i,j}$, and $b_0 \in \mathbb{R}^p$, $g \in \mathbb{R}$ and $\eta \in \mathbb{R}^+$ are hyperparameters of the model. On the one hand, note the use of a joint prior distribution on α_0 and σ^2, which is a widely used non-informative prior known in the standard linear regression setting as Jeffrey's prior (Jeffreys, 1946). In any event, the estimation of $\alpha_0 = \mathbb{E}[Y]$ is straightforward, so it could have been left out of the model altogether. On the other hand, the prior on b is a slight modification of the well-known Zellner's g-prior (Zellner, 1986), in which a regularizing term is added to avoid ill-conditioning problems in the Gram matrix, obtaining a ridge-like Zellner prior controlled by the tuning parameter η (Baragatti and Pommeret, 2012). All in all, with a slight abuse of notation the proposed prior distribution becomes $\pi(\theta) = \pi(b|\tau, \sigma^2)\pi(\tau)\pi(\alpha_0, \sigma^2)$.

As for the posterior distribution, we only compute a function proportional to its log-density, since that is all that is needed for a MCMC algorithm to work. A standard algebraic manipulation in (3.3) yields the following result.

Proposition 3.1. *Under the linear model* (3.5)*, the prior distribution implied in* (3.6) *produces the log-posterior distribution*

$$\log \pi(\theta \mid \mathscr{D}_n) \propto \frac{1}{2\sigma^2}\left(\|\mathbf{Y} - \alpha_0\mathbf{1} - \mathscr{X}_\tau b\|^2 + \frac{1}{g}(b - b_0)'G_\tau(b - b_0)\right) + (p + n + 2)\log\sigma - \frac{1}{2}\log|G_\tau|,$$

where $\mathbf{Y} = (Y_1, \dots, Y_n)'$ *and* $\mathbf{1}$ *is an* n*-dimensional vector of ones.*

Making predictions

In order to generate predictions, let us recall that when performing the empirical posterior approximation, on each of the M steps of the iterative MCMC algorithm we get an approximate sample $\theta^{(m)*} = (b^{(m)*}, \tau^{(m)*}, \alpha_0^{(m)*}, (\sigma^2)^{(m)*})$ of the posterior distribution $\pi(\theta|\mathscr{D}_n)$. Assuming now a previously unseen test set $\mathscr{D}'_{n'}$ in the same conditions as $\mathscr{D}_n$, we propose to construct three different kinds of predictors based on the MCMC samples, each of them following a different strategy.

Summarize-then-predict. If we consider a point-estimate statistic T that acts as a summary of the marginal posterior distributions, we can get the corresponding estimates $\hat{\theta} = (\hat{b}, \hat{\tau}, \hat{\alpha}_0, \hat{\sigma}^2) = T\{\theta^{(m)*}\} \equiv (T\{b^{(m)*}\}, T\{\tau^{(m)*}\}, T\{\alpha_0^{(m)*}\}, T\{(\sigma^2)^{(m)*}\})$, and then we can predict the responses in the usual way following model (3.4), i.e.:

$$\hat{Y}_i = \hat{\alpha}_0 + \sum_{j=1}^{p} \hat{\beta}_j X_i(\hat{t}_j), \quad i = 1, \dots, n'. \tag{3.7}$$

Note that in this case the variance σ^2 is treated as a nuisance parameter. Although it contributes to measure the uncertainty in the approximations, its estimates are discarded in the final prediction.

Predict-then-summarize. Alternatively, we can look at the approximate posterior distribution as a whole, and compute the predictive distribution of the simulated responses at each step of the chain following model (3.5):

$$\mathbf{Y}^{(m)*} := \left\{Y_i^{(m)*} \equiv Y_i \mid X_i, \theta^{(m)*} \;:\; i = 1, \dots, n'\right\}, \quad m = 1, \dots, M. \tag{3.8}$$

Then, we can take the mean of all such simulated responses as a proxy for each response variable, that is,

$$\hat{Y}_i = \frac{1}{M} \sum_{m=1}^{M} Y_i^{(m)*}, \quad i = 1, \dots, n'.$$

This method differs from the previous one in that it takes into account the full approximate posterior distribution instead of summarizing it directly.

Variable selection. Lastly, we can focus only on the marginal posterior distribution of $\tau|\mathscr{D}_n$ and select p time instants using a point-estimate statistic T as in our first strategy, but discarding the rest of the parameters. Specifically, we can consider the times $\hat{t}_j = T\{t_j^{(m)*}\}$ and reduce the original data set to just the $n \times p$ real matrix given by $\{X_i(\hat{t}_j) : i = 1, \dots, n,\ j = 1, \dots, p\}$. After this variable selection has been carried out, we can tackle the problem using a finite-dimensional linear regression model and apply any of the well-known prediction algorithms suited for this situation.

Note that these predictors can be obtained all at once after only one round of training (that is, an individual MCMC run to approximate the posterior distribution). As a consequence, what we have in practice is a single algorithm that can produce multiple predictors at the same computational cost, so that any of them can be chosen (or even switched back and forth) depending on the particularities of the problem at hand. Moreover, one could even contemplate an *ensemble model* in which some kind of aggregation of several of the available prediction methods is performed to produce a final result.

Besides, observe that the choice of a specific point estimator to summarize the posterior distribution results in a veiled assumption of an underlying loss function between the estimated and real parameters. In general, the mean is more sensitive to outliers and the median is more robust, but the latter assumes an l^1-type loss function while the former implicitly optimizes an l^2 loss. On the other hand, the mode is also a good candidate because it represents the point of highest probability density, following a *maximum a posteriori* (MAP) approach. At any rate, these decisions are strongly dependent on several factors such as the skewness or the number of modes in the resulting posterior distribution, and thus should be made on a case-by-case basis.

3.2 Functional logistic regression

In the case of functional logistic regression, we regard the response variable $Y \in \{0, 1\}$ as a Bernoulli random variable given $X = x \in L^2[0, 1]$, and as usual suppose that $\log(p(x)/(1 - p(x)))$ is linear in x, where $p(x) = \mathbb{P}(Y = 1|X = x)$. Then, following the approach suggested by Berrendero et al. (2022), a RKHS model might be given, in terms of the correspondence $\langle X, \alpha\rangle_K = \Psi_X(\alpha)$, by the equation

$$\mathbb{P}(Y = 1 \mid X) = \frac{1}{1 + \exp\{-\alpha_0 - \langle X, \alpha\rangle_K\}}, \quad \alpha_0 \in \mathbb{R},\ \alpha \in \mathscr{H}_{0,p}(K). \tag{3.9}$$

Indeed, note that this can be seen as a finite-dimensional approximation (but, still, with a functional interpretation) to the general RKHS functional logistic model proposed by these authors, which can be obtained by replacing $\mathscr{H}_{0,p}(K)$ with the whole RKHS space $\mathscr{H}(K)$. Now, if we aim at a classification problem, our strategy will be similar to that followed in the functional linear model: after incorporating the sample information, we can rewrite (3.9) as

$$Y_i \mid X_i, \theta \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(p_i), \quad i = 1, \dots, n, \tag{3.10}$$

with

$$p_i = \mathbb{P}(Y_i = 1 \mid X_i, \theta) = \frac{1}{1 + \exp\left\{-\alpha_0 - \sum_{j=1}^{p} \beta_j X_i(t_j)\right\}}, \quad i = 1, \dots, n, \tag{3.11}$$

where in turn $\alpha_0, \beta_j \in \mathbb{R}$ and $t_j \in [0, 1]$.

In much the same way as the linear regression model described above, this RKHS-based logistic regression model offers some advantages over the L^2-model (2.2). First and foremost, it has a more straightforward interpretation and allows for a workable Bayesian approach, as we will demonstrate below. Secondly, it can be shown that under mild conditions the general RKHS functional logistic model holds whenever the conditional distributions $X|Y = 0$ and $X|Y = 1$ are homoscedastic Gaussian processes, and in some cases it also entails the L^2-model (see Theorem 1 in Berrendero et al., 2022); this provides a sound theoretical motivation for the reduced model. Furthermore, a maximum

likelihood approach for parameter estimation (although not considered here) is possible as well. Indeed, the use of a finite-dimensional approximation mitigates the problem of non-existence of the MLE in the functional case. However, let us recall that even in finite-dimensional settings there are cases of quasi-complete separation in which the MLE does not exist (Albert and Anderson, 1984), though this issue can be circumvented using, for example, Firth's corrected estimator (Firth, 1993).

The Bayesian approach: prior and posterior

As far as prior distributions go, we propose to use the same ones as we did in (3.6) for the linear regression model. However, in this case the nuisance parameter σ^2 only appears as part of the hierarchical prior distribution, and not in the final model. The posterior distribution is again derived after a routine calculation.

Proposition 3.2. *Under the logistic model* (3.10)*, the prior distribution implied in* (3.6) *produces the log-posterior distribution*

$$\begin{aligned}\log \pi(\theta \mid \mathcal{D}_n) \propto & \sum_{i=1}^{n}\left[\left(\alpha_0 + \langle X_i, \alpha\rangle_K\right)Y_i - \log\left(1 + \exp\left\{\alpha_0 + \langle X_i, \alpha\rangle_K\right\}\right)\right] \\ & + \frac{1}{2}\log|G_\tau| - (p+2)\log\sigma - \frac{1}{2g\sigma^2}(b-b_0)'G_\tau(b-b_0).\end{aligned}$$

Remember that $\langle X_i, \alpha\rangle_K = \sum_{j=1}^{p}\beta_j X_i(t_j)$.

Making predictions

Bear in mind that in this case we are essentially approximating the probabilities p_i in (3.11), so before producing a response we need to transform the predicted values to a binary output in $\{0, 1\}$. According to the usual criterion of minimizing the misclassification probability, it is known that the Bayes optimal rule is recovered by predicting $\hat{Y} = 1$ whenever $\mathbb{P}(Y = 1|X) \geq 1/2$. Nevertheless, for a more general cost function one could consider other criteria that would lead to evaluating whether $\mathbb{P}(Y = 1|X) \geq \gamma$ for some threshold $\gamma \in [0, 1]$.

With this last strategy in mind, the summarize-then-predict approach on the approximate posterior distribution is analogous to the linear regression case:

$$\hat{Y}_i = \mathbb{I}\left(\left[1 + \exp\left\{-\hat{\alpha}_0 - \sum_{j=1}^{p} \hat{\beta}_j X_i(\hat{t}_j)\right\}\right]^{-1} \geq \gamma\right), \quad i = 1, \dots, n', \tag{3.12}$$

where $\mathbb{I}$ is the indicator function ($\mathbb{I}(P)$ is 1 if P is true and 0 otherwise). The hat estimates are obtained once again through a summary statistic T of the corresponding marginal posterior distributions. On the other hand, the prediction method that takes into account the entire posterior approximation (i.e. the predict-then-summarize approach) is somewhat different now, since there is the question of which response (the Bernoulli variables in (3.10) or the raw probabilities in (3.11)) to consider when averaging the posterior samples. Hence, there are primarily two possible outcomes.

Average sampled probability. If we choose to average the approximate probabilities $p_i^{(m)*} = \mathbb{P}(Y_i = 1 | X_i, \theta^{(m)*})$ computed following (3.11), the resulting predictor is

$$\hat{Y}_i = \mathbb{I}\left(\frac{1}{M}\sum_{m=1}^{M} p_i^{(m)*} \geq \gamma\right), \quad i = 1, \dots, n'.$$

Average sampled response. Deciding to average the approximate binary responses $Y_i^{(m)*}$ instead (see (3.8)) leads to computing the predictions as

$$\hat{Y}_i = \mathbb{I}\left(\frac{1}{M}\sum_{m=1}^{M} Y_i^{(m)*} \geq \gamma\right), \quad i = 1, \dots, n'.$$

In this case, each $Y_i^{(m)*}$ is a random variable that follows a Bernoulli distribution with parameter $p_i^{(m)*}$, for $m = 1, \dots, M$. Note that when $\gamma = 1/2$ this is equivalent to predicting $\hat{Y}_i$ according to the majority vote of all the $Y_i^{(m)*}$.

Lastly, the variable selection method is essentially the same as in the case of functional linear regression: we select p time instants from each trajectory based on a summary of the posterior distribution $\tau | \mathscr{D}_n$, and then feed the reduced data set to a finite-dimensional binary classification procedure.

4 Model choice, implementation and validation

In this chapter we gather together several remarks on the main choices made during the design and implementation of our model, as well as some examples of validation strategies that attempt to measure the goodness-of-fit of the model given the observed data.

4.1 Model specification

First we describe some problems and modeling decisions that we had to deal with throughout the development of the model.

Label switching

A well-known issue found in mixture-like models like the ones we propose is *label switching*, which in short refers to the non-identifiability of the components of the model caused by their interchangeability. In our case, this happens because the likelihood is symmetric with respect to the ordering of the parameters b and τ, i.e., $\pi(Y|X,\theta) = \pi(Y|X,\nu(\theta))$ for any permutation ν that rearranges the indices $j = 1,\dots,p$. Thus, since the components are arbitrarily ordered, they may be inadvertently exchanged from one iteration to the next in any MCMC algorithm. This can cause nonsensical answers when summarizing the marginal posterior distributions to perform inference, as different labelings might be mixed on each component (Stephens, 2000).

However, this phenomenon is perhaps surprisingly a condition for the convergence of the MCMC method: as pointed out by many authors (e.g. Celeux et al., 2000), a lack

of switching would indicate that not all modes of the posterior distribution were being explored by the sampler. For this reason, many ad-hoc solutions revolve around post-processing and relabeling the samples to eliminate the switching effect, but they generally do not prevent it from happening in the first place.

The most straightforward solutions consist on imposing an artificial identifiability constraint on the parameters to break the symmetry of their posterior distributions; see Jasra et al. (2005) and references therein. A common approach that seems to work well is to simply enforce an ordering in the parameters in question, which in our case would mean requiring for example that $\beta_i < \beta_j$ for $i < j$, or the analogous with the times τ. We have implemented a variation of this method described in Simola et al. (2021), which works by post-processing the samples and relabeling the components to satisfy the order constraint mentioned above, choosing either b or τ depending on which set of ordered parameters would produce the largest separation between any two of them (suitably averaged across all iterations of the chains).

This is an area of ongoing research, and thus there are other, more complex relabeling strategies, both deterministic and probabilistic. A summary of several such methods can be found for example in Rodríguez and Walker (2014) and Papastamoulis (2015). In particular, we tested the pivot method proposed by Marin et al. (2005), in which all samples are aligned to minimize their distance to a reference element (the "pivot"), which is chosen as the sample that maximizes the posterior density. However, the process of finding the appropriate permutation in each case was time-consuming, and the results were similar, if not worse, than the ones obtained with the simpler order constraints. For this reason, the latter were chosen as the default relabeling method of our algorithm, though the pivot method can still be enabled through a dedicated argument in the sampling procedure.

The choice of p

One of the key decisions in our Bayesian modeling scheme was whether to consider the number of components p as a member of the parameter space and integrate it into the model. While theoretically we could impose a prior distribution on p as well (e.g. a categorical distribution with a fixed maximum value), we found that it would have some unwanted practical implications. For instance, it would make the implementation more

complex, since the dimensionality of the parameters b and τ would need to be fixed at a certain maximum value beforehand, but the working value of p within the MCMC algorithm would vary from one iteration to the next. In this case we would have no immediate way of tracking down which set of parameters is "active" at any given time. A simple approach would be to always consider the first p parameters and ignore the rest, and we did indeed try this technique, but it gave rise to new difficulties and the results obtained were not good. In fact, the label switching issue is accentuated when p is allowed to vary (c.f. Grollemund et al., 2019, Sec. 2.3), and on top of that, the interpretation of, say, the first coefficient β_1 in a model with 3 components is different than the interpretation of the same coefficient in a model with only 2 components.

This inconsistency in the interpretation of the components when the dimensionality of the model increases or decreases can be mitigated using a particular type of MCMC method known as reversible-jump MCMC (Green, 1995). Theoretically, these algorithms are specifically designed to approximate the posterior distribution in mixture-like models when the number of components is unknown, allowing the underlying dimensionality to change between iterations. However, since they are not yet widely adopted in practice and a reference implementation is not available, we decided against using them in our applications.

Another possibility would be to adapt a purely Bayesian model selection technique to our framework (see Gelman et al., 2013; Piironen and Vehtari, 2017), or even derive some model aggregation methods to combine the posterior distributions obtained for different-sized models. These methods are usually based in computing a quantity known as the *Bayes factor*, which in turn requires the specific value of the normalizing integral constant we have been trying to avoid all along. In the end, for the sake of simplicity we decided to let p be an hyperparameter, so that we could use any model selection criteria (e.g. BIC, DIC, cross-validation, ...) to select its optimal value. As we will see shortly in Chapter 5, the experiments carried out indicate that even low values of p provide sufficient flexibility in most scenarios, and they suggest that our models are not very sensitive to small variations of this parameter.

Other hyperparameters

As for the default values of the rest of hyperparameters in the prior distributions in (3.6), several comments are in order:

- For the expected value b_0 we propose to use the MLE of b. Although the likelihood function is rather involved, an approximation of the optimal value is enough for our purposes. Our numerical studies suggest that the results are much better with this choice than, say, with a random or null vector.
- We found that the parameter g does not have as much influence on the final result, and the experimentation indicates that $g = 5$ is a good value.
- Lastly, we observed that the choice of η can have a considerable impact on the final estimator. That is why, in an effort to normalize its scale, we consider a compound parameter $\eta = \tilde{\eta}\lambda_{\max}(X_\tau' X_\tau)$, where $\lambda_{\max}(X_\tau' X_\tau)$ is the largest eigenvalue of the matrix $X_\tau' X_\tau$, and $\tilde{\eta} > 0$ is the actual tuning parameter (which can be selected for instance by cross-validation strategies). This standardization technique has been used previously in the literature; see for example Grollemund et al. (2019).

4.2 MCMC implementation

The MCMC method chosen for approximating the posterior distribution in our models is the affine-invariant ensemble sampler described in Section 2.2. As mentioned there, we utilize the computational implementation in the Python library *emcee*, which is both reliable and easy to use; it aims to be a general-purpose package that performs well in a wide class of problems. One advantage of this method, apart from the property of affine-invariance, is that it only requires us to specify a few hyperparameters, irrespective of the underlying dimension. This contrasts to, say, the $\mathcal{O}(N^2)$ degrees of freedom corresponding to the covariance matrix of an N-dimensional jump distribution in Metropolis-Hastings. Furthermore, following Goodman and Weare (2010) we can use the *integrated autocorrelation time* to compute an estimate of the effective sample size (i.e. the number of independent samples) after the sampling is done.

After selecting the number of iterations and the number of chains we want, we need to specify the initial points for each of them. As pointed out in Foreman-Mackey et al. (2013), a good initial choice is a Gaussian ball around a point in Θ_p that is expected to have a high probability with respect to the objective distribution. In our implementation we adopt this method, choosing an approximation of the MLE of θ as the central point in each case. To perform this approximation we employ the optimization suite of

the Python library *scipy*[4], and in particular we use the Basin-hopping algorithm (Wales and Doye, 1997). This is a two-phase stochastic method that combines global steps with local optimization, in the hope of avoiding getting stuck too quickly in local maxima. To reduce the effects of randomness, we run the algorithm a few times and retain the point with the highest likelihood, and to avoid biasing the sampler too much towards the specific point selected, we let a fraction of the initial points be random (within reasonable bounds). This approximation is also used to specify the hyperparameter b_0.

Other less relevant hyperparameters include the burn-in period for the chains, which is the number of initial samples discarded, or the amount of thinning performed, which is the number of consecutive samples discarded to reduce the correlation among them. Another computational decision we made is working with $\log \sigma$ instead of σ^2, so that the domain of this parameter is an unconstrained space, which apparently helps increase the efficiency of the method.

Finally, it is worth mentioning that we experimented with another MCMC framework called *pymc*[5]. This is a well-established project aimed at developing a probabilistic programming language for Bayesian modeling with emphasis on MCMC methods (Salvatier et al., 2016). This package is more complex to use and has a steeper learning curve, but at the same time offers a richer experience and a wide variety of samplers. Apart from standard Metropolis-Hastings techniques, the main algorithm in this library is the *No U-Turn Sampler* (NUTS), a Hamiltonian Monte Carlo method proposed in Hoffman and Gelman (2014) that performs an automatic adjustment of hyperparameters. However, while the results obtained were similar to that of *emcee*, the execution time of the NUTS sampler was considerably higher (most likely due to parametrization issues with the model). Nevertheless, we decided to include this package as part of the developed code for future use (the Metropolis-Hastings sampler is fully functional and shows comparable performance to *emcee*), although we do not report the corresponding experimental results.

4.3 Validation techniques

To conclude, it is worth mentioning that the Bayesian aspects of our model allow us to perform some model validation checks straight away. For example, we can derive credible

[4] https://docs.scipy.org/doc/scipy/
[5] https://docs.pymc.io

intervals for each of the parameters (see Figure 5), and in the case of linear regression, we can use the sampled values of σ^2 as a measure of the uncertainty of the predictions.

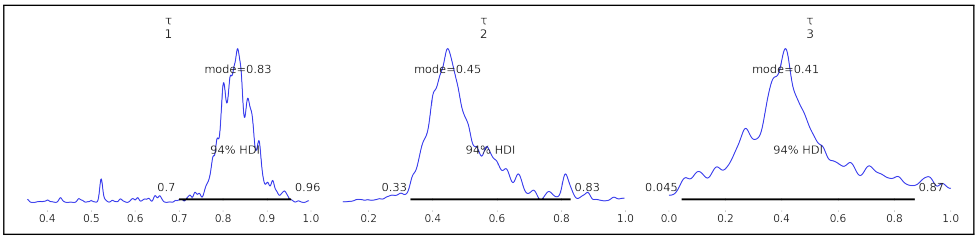


Figure 5: Example of 94% credible intervals on the posterior distribution of the impact points.

Moreover, we can perform various visual checks, such as a plot comparing both the observed and posterior predictive distribution of the responses (see Figure 6). The posterior predictive distribution for a new sample x is formally defined as

$$\pi(y \mid x, \mathscr{D}_n) = \int_\Theta \pi(y \mid x, \theta)\pi(\theta \mid \mathscr{D}_n)\, d\theta,$$

and in our context it can be approximated by the simulated responses $\mathbf{Y}^* \equiv \{\mathbf{Y}^{(m)*}\}$, following the notation of (3.8). This distribution accounts for the uncertainty about θ, and we do indeed use it for prediction in our predict-then-summarize approach.

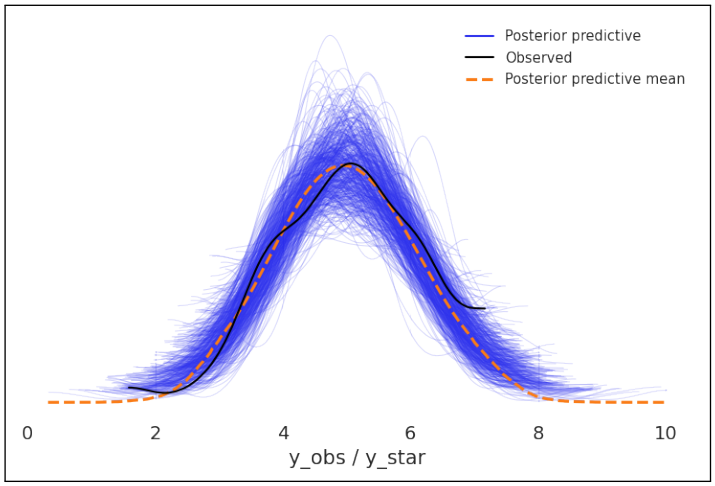


Figure 6: Example of posterior predictive graphical checks on a fitted linear regression model. We show a comparison between the observed distribution of the response variable and the posterior predictive distribution consisting on the approximate sampled responses.

In addition, we can calculate the so-called *Bayesian p-values* for several statistics, which are defined as $P(T(\mathbf{Y}^*) \leq T(\mathbf{Y})|\mathbf{Y})$, and are computed by simply measuring the proportion of the M estimates $T\{\mathbf{Y}^{(m)*}\}$ that fall below the observed value of the statistic, $T(\mathbf{Y})$. They are expected to be around 0.5 when the model accurately represents the data, and a deviation either way can be indicative of modeling issues; see Chapter 6 of Gelman et al. (2013) for details.

5 Experiments

In this chapter we present the results of the experiments carried out to test the performance of our models in different scenarios, both simulated and with real data. The structure of the code and other minor computational details are discussed in Appendix A.

Experimental setting

First and foremost, we consider $T = \{\text{mean, median, mode}\}$ for our summarize-then-predict approach to prediction (see (3.7) and (3.12)). As a result, in linear regression we have 4 prediction methods (one for each statistic and one for the predict-then-summarize approach) and 3 variable selection methods (one for each statistic), while in logistic regression there are similarly 5 prediction methods and 3 variable selection procedures. In each case, after variable selection is performed, we use a l^2-penalized multiple linear/logistic regression method to generate the corresponding predictions. All in all, we are looking at 7 (8 in the case of logistic regression) prediction methods, and although all of them are derived from a single MCMC run, we will treat them as separate for the purposes of the experimentation. In what follows we relax the notation and use the expressions "model", "prediction method" and "algorithm" almost interchangeably when referring to these 7 (or 8) cases.

We consider $n = 150$ training samples and $n' = 100$ testing samples on an equispaced grid of $N = 100$ points on $[0, 1]$ for the simulated data sets, and we do a 66%/33% train/test split on the real data sets. We then perform 5-fold cross validation (CV) on the training set to select the best values of p and η for each model, and after refitting the best model in each case on the whole training set, we evaluate it and measure the predictive performance on the test set. We look for p in the set $\{1, 2, \dots, 10\}$, while the possible values of η are $\{10^{-4}, 10^{-3}, \dots, 10^{2}\}$.

Because of execution time constraints, the hyperparameters of the MCMC method are not part of the CV process, and are selected manually based on an initial set of experiments, as well as recommendations from the original article. We use 64 chains and run them for 900 iterations in total, discarding the first 500 iterations as burn-in. Moreover, we use a weighted mixture of walk and stretch moves in the *emcee* sampler to advance the chains in each iteration, selecting the stretch move (the default) with probability 0.7 or the walk move with probability 0.3.

Lastly, since the use of MCMC algorithms introduces a source of stochasticity in the prediction procedure, we independently repeat the whole process 10 times (each with a different train/test configuration), and average the results across these executions. The metrics used to evaluate the performance of the models are the RMSE for linear regression and the accuracy for logistic regression. Recall that

$$\text{RMSE}(\{\hat{y}_i\}) = \sqrt{\frac{1}{n'}\sum_{i=1}^{n'}(y_i - \hat{y}_i)^2}$$

and

$$\text{Accuracy}(\{\hat{y}_i\}) = \frac{1}{n'}\sum_{i=1}^{n'}\mathbb{I}(y_i = \hat{y}_i).$$

Data sets

We consider a set of functional regressors common to linear and logistic regression problems. They are four Gaussian processes (GP's), each with a different covariance function:

(i) BM. A Brownian motion $K_1(t,s) = \min\{t,s\}$.
(ii) fBM. A fractional Brownian motion $K_2(t,s) = 1/2(s^{2H} + t^{2H} - |t-s|^{2H})$, with Hurst parameter $H = 0.8$.
(iii) O-U. An Ornstein-Uhlenbeck process $K_3(t,s) = e^{-|t-s|}$.
(iv) Gaussian. A Gaussian process with Gaussian kernel $K_4(t,s) = e^{-(t-s)^2/2\nu^2}$, where $\nu = 0.2$.

Also, when applicable, we fix a variance $\sigma^2 = 0.5$ for the error terms ε.

Linear regression data sets. We consider three different types of simulated data sets, all with a common value of $\alpha_0 = 5$.

- A finite-dimensional RKHS model where the response is generated as $Y = 5 - 5X(0.1) + X(0.4) + 10X(0.8) + \varepsilon$, for each of the four GP's mentioned above.
- A L^2-model with underlying coefficient function $\beta(t) = \log(1 + 4t)$, again for the same four GP's.
- A model based on non-GP regressors. Specifically, we use a geometric Brownian motion (GBM) defined as $X(t) = e^{BM(t)}$, where $BM(t)$ is a standard Brownian motion. In this case we consider two data sets, one with a RKHS response and one with a L^2 response, with the same parameters as above.

As for the real data sets, we use the (twice-differentiated) Tecator data set introduced in Chapter 1 to predict fat content of 193 meat samples, as well as what we call the Moisture (Kalivas, 1997) and Sugar (Bro, 1999) data sets. The first consists of near-infrared spectra of 100 wheat samples and the objective is to predict the samples' moisture content, whereas the second contains 268 samples of sugar fluorescence data in order to predict ash content. The three data sets are measured on a grid of 100, 101 and 115 equispaced points on $[0, 1]$, respectively.

Logistic regression data sets. Again we consider three different types of simulated data sets, with a common value of $\alpha_0 = -0.5$. In this case we randomly permute 10% of the labels to introduce some noise in the simulations.

- Four logistic finite-dimensional RKHS models with the same functional parameter as in the linear regression case (one for each GP). Specifically,

$$\mathbb{P}(Y = 1 \mid X) = \frac{1}{1 + \exp\{0.5 + 5X(0.1) - X(0.4) - 10X(0.8)\}}.$$

- Four logistic L^2-models with the same coefficient function as in the linear regression case, i.e.:

$$\mathbb{P}(Y = 1 \mid X) = \frac{1}{1 + \exp\left\{0.5 - \int_0^1 \log(1 + 4t)X(t)\, dt\right\}}.$$

- A "mixture" model based on non-GP regressors, in which we mix regressors from two different GP's (with equal probability $p = 1/2$) and label them according

to their origin. Firstly, we consider a homoscedastic case to distinguish between a standard Brownian motion and a Brownian motion with a mean function that is zero until $t = 0.5$, and then becomes $m(t) = 0.75t$. Secondly, we consider a heteroscedastic case to distinguish between a standard Brownian motion and a Brownian motion with variance 2, that is, with kernel $K(t, s) = 2\min\{t, s\}$.

Additionally, we use three real data sets well known in the literature. The first one is a subset of the Medflies data set (Carey et al., 1998), consisting on samples of the number of eggs laid daily by 534 flies over 30 days, to predict if their longevity is high or low. The second one is the Berkeley Growth Study data set (Tuddenham and Snyder, 1954), which records the height of 54 girls and 39 boys over 31 different points in their lives. Finally, we selected a subset of the Phoneme data set (Hastie et al., 1995), based on 200 digitized speech frames over 128 equispaced points to predict the phonemes "aa" and "ao".

Comparison algorithms

We have included a fairly comprehensive suite of comparison algorithms, chosen among the most common methods used in machine learning and FDA. There are purely functional methods, finite-dimensional models that work on the discretized data, and variable selection/dimension reduction procedures.

Linear regression comparison algorithms. We consider the following regression methods:

- Manual. Dummy variable selection method with a pre-specified number of components (equispaced on $[0, 1]$).
- Lasso. Linear least squares with l^1 regularization.
- Ridge. Linear least squares with l^2 regularization.
- PLS. Partial least squares for dimension reduction.
- PCA. Principal component analysis for dimension reduction.
- PLS1. Partial least squares regression (e.g. Wegelin, 2000).
- APLS. Functional partial least squares regression proposed by Delaigle and Hall (2012b).
- RMH. Recursive maxima hunting variable selection method proposed by Torrecilla and Suárez (2016).

- FLin. Functional L^2 linear regression model with fixed basis expansion.
- FPCA. Functional principal component analysis.
- FPLS1. Functional PLS regression through basis expansion, implemented as in Aguilera et al. (2010).

Logistic regression comparison algorithms. All the variable selection and dimension reduction algorithms from above are also considered in this case, plus the following classification methods:

- Log. Standard multiple logistic regression with l^2 regularization.
- LDA. Linear discriminant analysis.
- QDA. Quadratic discriminant analysis.
- RKVS. RKHS-based variable selection method proposed in Berrendero et al. (2018).
- APLS. Functional PLS used as a dimension reduction method, as proposed in Delaigle and Hall (2012a) in combination with the nearest centroid (NC) algorithm.
- FLog. Functional RKHS-based logistic regression algorithm proposed in Berrendero et al. (2022).
- FLDA. Implementation of the functional version of linear discriminant analysis proposed in Preda et al. (2007).
- MDC. Maximum depth classifier (e.g. Ghosh and Chaudhuri, 2005).
- FKNN. Functional K-nearest neighbors classifier with the L^2-distance.
- FNC. Functional nearest centroid classifier with the L^2-distance.

The main parameters of all these algorithms are selected by cross-validation, using the same 5 folds as our proposed models so that the comparisons are fair. In particular, regularization parameters are searched among 20 values in the logarithmic space $[10^{-4}, 10^4]$, the number of manually selected variables is one of $\{5, 10, 15, 20, 25, 50\}$, the number of components for dimension reduction and variable selection techniques is in the set $\{2, 3, 4, 5, 7, 10, 15, 20\}$, the number of basis elements for cubic spline bases is in $\{8, 10, 12, 14, 16\}$, the number of basis elements for Fourier bases is one of $\{3, 5, 7, 9, 11\}$, and the number of neighbors in the KNN classifier is in $\{3, 5, 7, 9, 11\}$.

Most algorithms have been taken from the libraries *scikit-learn*[6] and *scikit-fda*[7], the first oriented to machine learning in general and the second to functional data analysis

[6] https://scikit-learn.org
[7] https://fda.readthedocs.io

in particular. However, some methods were not found in these packages and had to be implemented from scratch. This is the case of the FLDA, FPLS and APLS methods, which we coded following the corresponding articles.

Results display

We have adopted a visual approach to presenting the experimentation results, using colored graphs to help visualize them. We felt that this was a better way of summarizing a large empirical study such as the one we have carried out. Alternatively, tables representing these results can be consulted in Appendix B.

We show the mean and standard deviation of the score obtained in each case across the 10 random runs, depicting our models in orange and the comparison algorithms in blue. We also show the global mean of all the comparison algorithms with a dashed vertical line, and we exclude extreme negative results from this mean to avoid distortion. Moreover, we separate complete prediction algorithms from two-stage methods, the latter being the ones that perform variable selection or dimension reduction prior to a multiple linear/logistic regression method.

We do not expect our models to defeat all the comparison algorithms in all cases (and indeed they do not), but we intend to show that they are competitive methods that on average match or improve the performance of other usual alternatives. After all, *there is no such thing as a free lunch*[8]!

5.1 Functional linear regression

Simulated data sets

In Figure 7 we see the results for the four GP regressors considered in the RKHS case. This is the most favorable case for us, as the underlying model coincides with our assumed model. Indeed, we can see that in most cases our algorithms are the ones with lower RMSE, save for a few exceptions, notably the Gaussian kernel. A subsequent anal-

[8] The phrase alludes to the No Free Lunch theorem, which loosely speaking asserts that all machine learning methods are equivalent when their performance is averaged across all possible problems. However, the practical implications of this statement are questionable (see e.g. Giraud-Carrier and Provost, 2005).

ysis showed that this particular data set is especially sensitive to the value of the hyperparameter η in our prior distribution, so a more customized approach would be needed to obtain better results.

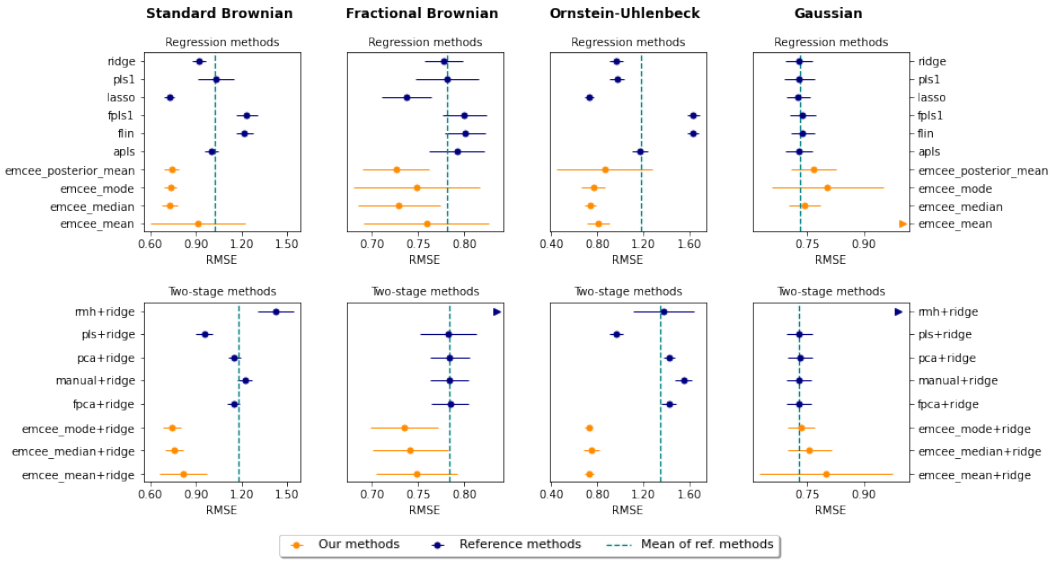


Figure 7: RMSE of predictors for simulated GP data that obeys an underlying RKHS model (lower is better).

In Figure 8 we see the results for the case with an underlying L^2-model, which would be our most direct competitor. In this case the outcome is satisfactory, since for the most part our models are on a par with the rest, even beating other methods that were designed with the L^2-model in mind. Moreover, whenever one of our models has a higher RMSE, the difference is pretty small in comparison. Note that some of our Bayesian models have a higher standard deviation, partly because there is an intrinsic randomness in the methods (apart from the train/test splits), and it can be the cause of the occasional worse performance. In relation to this, we observe that the methods that use the mean as a summary statistic tend to perform much worse. This is because the mean is very sensitive to outliers, so if at some point there is a chain that randomly deviates from the rest, the mean of the posterior distribution will be greatly impacted. In particular, very high values of the parameter σ^2 were reported sometimes, producing misleading results when averaged with the rest of the values.

Lastly, the results for the non-GP case can be seen in Figure 9, in which the regressors are realizations of a GBM. In this case we still get better results under the RKHS model,

while the results under the L^2-model are slightly worse. However, as before, the difference is small (except for the *emcee_mean* methods).

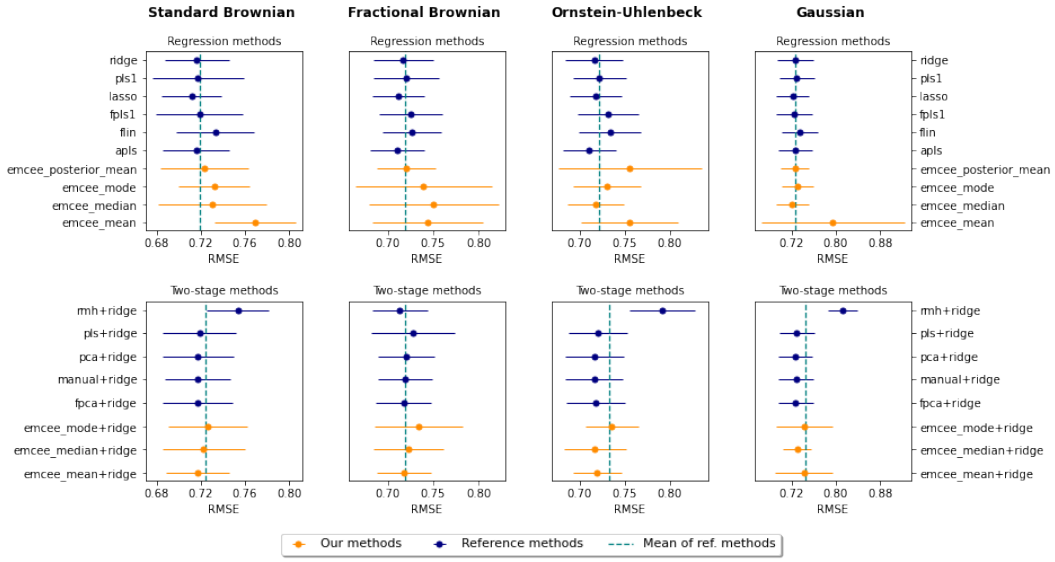


Figure 8: RMSE of predictors for simulated GP data that obeys an underlying L^2 model (lower is better).

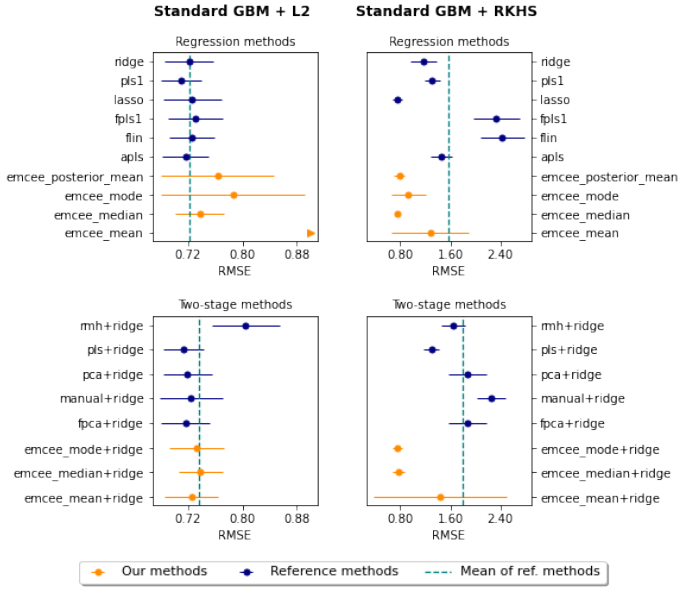


Figure 9: RMSE of predictors for simulated data with GBM regressors (lower is better).

Real data

Figure 10 shows the results for the real data sets. In these data sets there is a substantial difference in performance between some of our methods and the reference algorithms. However, the predict-then-summarize approach (represented as *posterior_mean*) seems to work quite well, always scoring near the mean RMSE of all the comparison algorithms. Moreover, our two-stage methods seem to outperform the summarize-then-predict methods in Moisture and Sugar, scoring again very close to the mean of the reference models.

We have to bear in mind that real data is more complex and noisy than simulated data, and it is possible that after a suitable pre-preprocessing we would obtain better results with our methods. However, our goal was to perform a general comparison without focusing too much on the specifics of any particular data set.

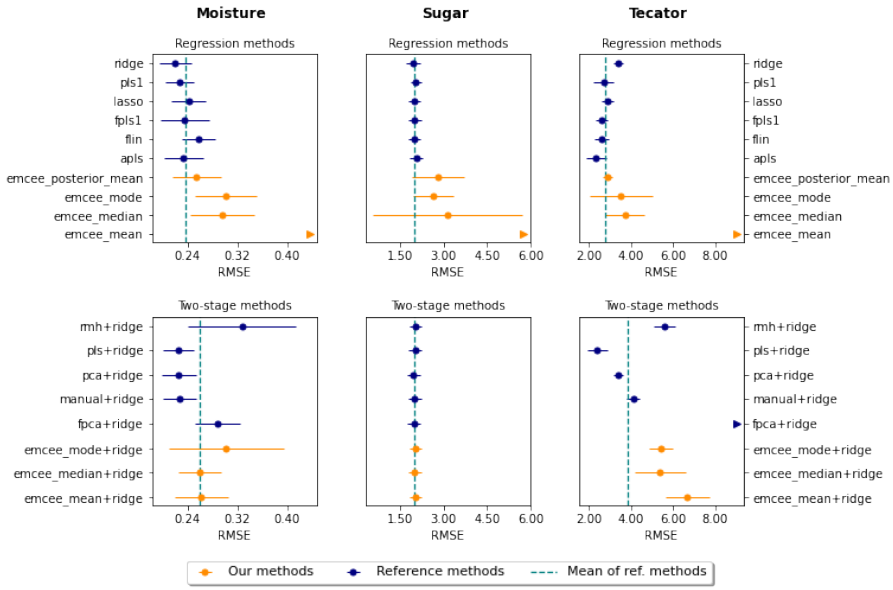


Figure 10: RMSE of predictors for real data sets (lower is better).

5.2 Functional logistic regression

Simulated data sets

In Figure 11 we see the results for the GP regressors in the logistic RKHS case. Our models perform fairly well in this advantageous case, although they are not always better than

the comparison methods. However, in most cases the differences observed account for only one or two misclassified samples.

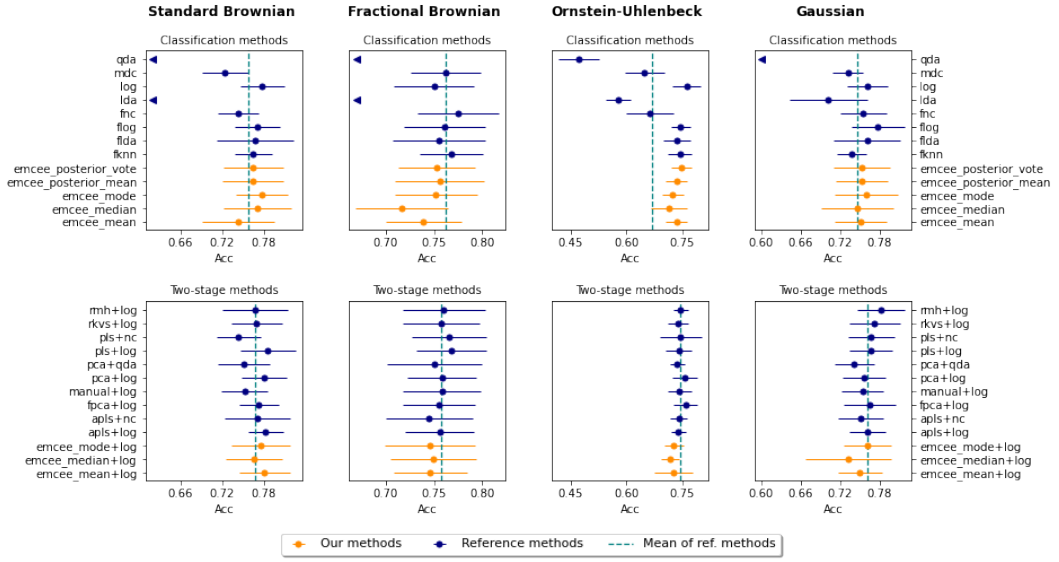


Figure 11: Accuracy of classifiers for simulated GP data that obeys an underlying logistic RKHS model (higher is better).

Continuing with Figure 12, we see that in the L^2 case the results are again promising, since our models score consistently on or above the mean of the reference models, and in many cases surpassing most of them. The predict-then-summarize approaches (*emcee_posterior_mean* and *emcee_posterior_vote*) are particularly good in this case, and in general have low standard errors. Moreover, the overall accuracy of all methods is poor (below 60%), so this is indeed a difficult problem in which even small increases in accuracy are relevant.

Finally, Figure 13 shows that our classifiers perform better than most comparison algorithms when separating two homoscedastic Gaussian processes, but they struggle in the heteroscedastic case. Incidentally, this heteroscedastic case of two zero-mean Brownian motions has a special interest, since it can be shown that the Bayes error is zero in the limit of dense monitoring (i.e. with an arbitrarily fine measurement grid), a manifestation of the "near-perfect" classification phenomenon analyzed for example in Torrecilla et al. (2020). Our results are in line with the empirical studies of this article, where the authors conclude that even though the asymptotic theoretical error is zero, most classi-

fication methods are suboptimal in practice (possibly due to the high collinearity of the data), with the notable exception of PCA+QDA.

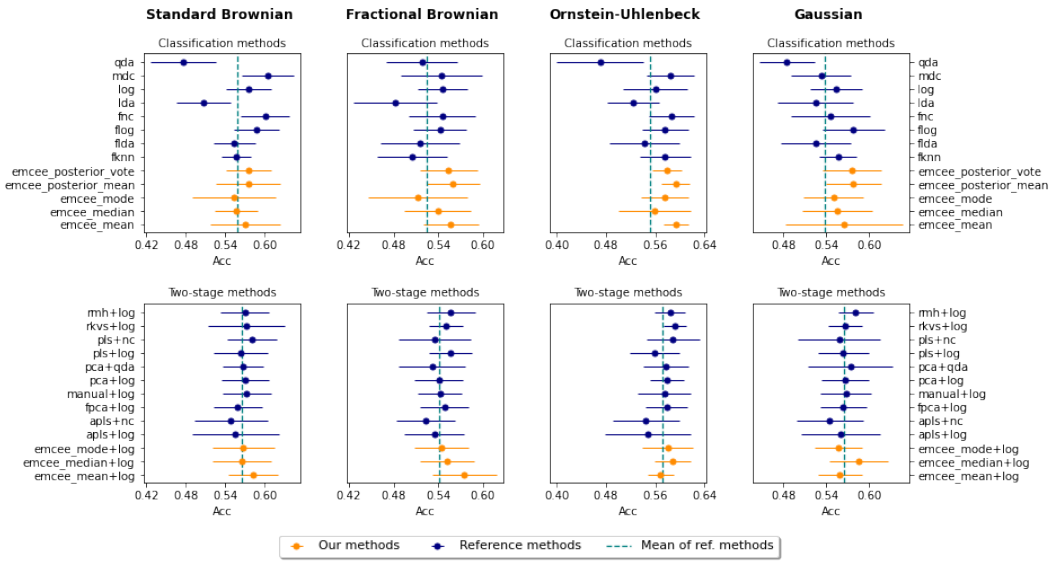


Figure 12: Accuracy of classifiers for simulated GP data that obeys an underlying logistic L^2-model (higher is better).

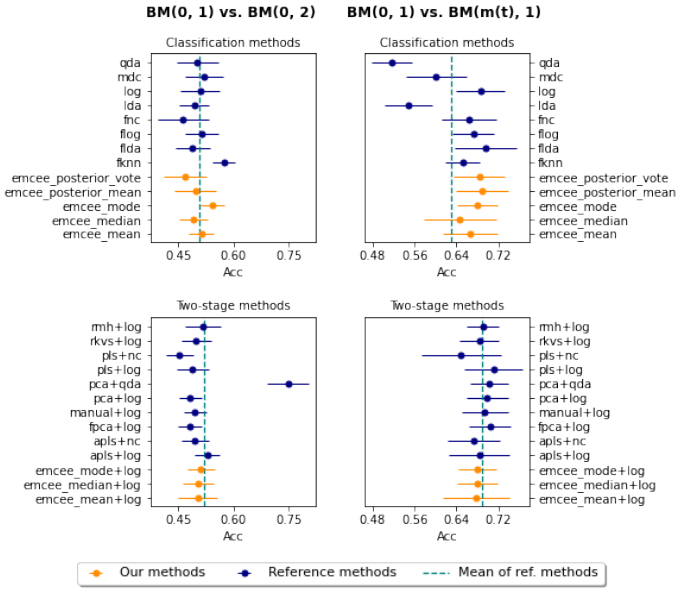


Figure 13: Accuracy of classifiers for simulated data coming from two different GP's, labeled according to their origin (higher is better).

Real data

As for the real data sets, in Figure 14 we see positive results in general, obtaining in most cases accuracies well above the mean of the reference models, and sometimes above most of them. In particular, the predict-then-summarize methods tend to have a good performance and achieve a lower standard error across executions, which is a trend that we also saw in the simulated data sets. As we have been seeing almost invariably, the models that use *emcee_mean* are the exception, and in all these data sets they perform steadily worse than the rest of our Bayesian models.

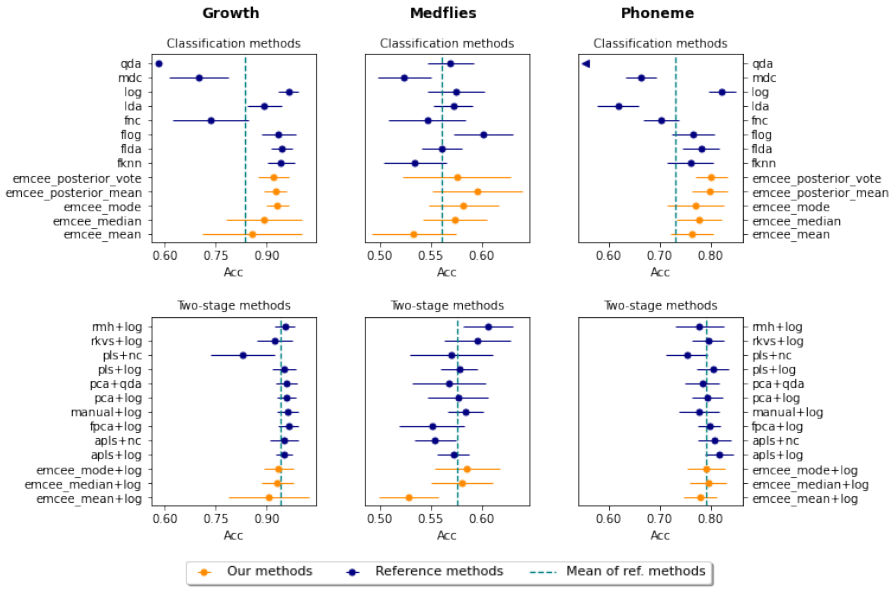


Figure 14: Accuracy of classifiers for real data sets (higher is better).

5.3 Additional experiments

In this section we present some experiments that are not directly related to measuring the predictive performance of our models relative to other methods.

Model misspecification

One requirement that our model should satisfy is that it ought to be able to recover the true parameter function when the underlying data generation model is a finite-dimensional

RKHS model. This is generally the case when the value of p in our model and the true value of p coincide, but what happens when we change the value of p in the model?

Take for example a RKHS data set with two components generated according to the formula $Y = 5 - 5X(0.1) + 10X(0.8) + \varepsilon$, with $\varepsilon \sim \mathcal{N}(0, 0.5)$. Figure 15 shows the resulting posterior distribution of the parameters $b = (\beta_1, \beta_2, \beta_3)$ and $\tau = (t_1, t_2, t_3)$ for a model with 3 components. As we can see, one of the coefficients has gone to zero to account for the overspecification of the model, while the other two have stabilized very close to the true parameters. The same goes for the time instants, except that in this case there is no default value to represent that a component is unused, so the time corresponding to the null coefficient oscillates back and forth. The estimated function (based for example in the mode of the posterior distributions) will not be perfect, essentially because of the noise in the response. But it should be close to the true parameter function $\alpha(t) = -5K(t, 0.1) + 10K(t, 0.8)$, providing a good predictive performance.

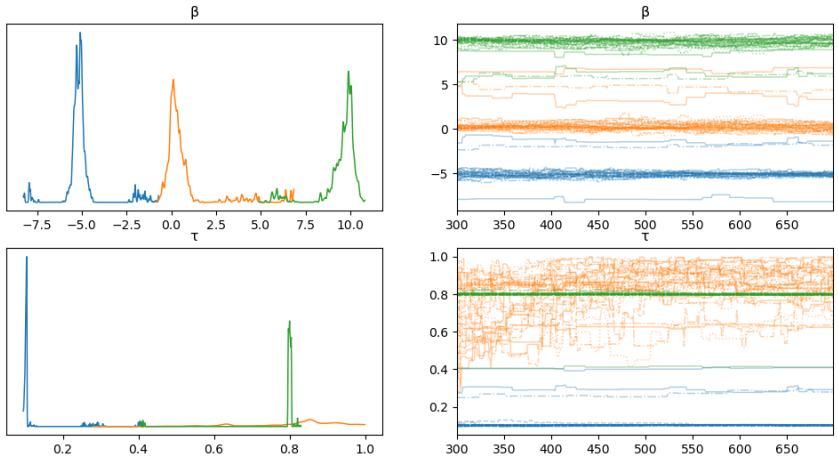


Figure 15: Estimated posterior distribution for b and τ in a linear model with $p = 3$ (left) and the corresponding trace evolution for 400 iterations in the MCMC sampler (right).

In contrast, if we consider now a model with $p = 4$ with the same data, we might obtain posterior distributions like the ones in Figure 16. In this situation two coefficients should go to zero, but that is no longer the case. For example, while the green component has a high density around 0, it also has a considerable mass around 10, effectively "competing" with the red component. This is a manifestation of the label switching issue, caused in this case by an excessive number of degrees of freedom in the model.

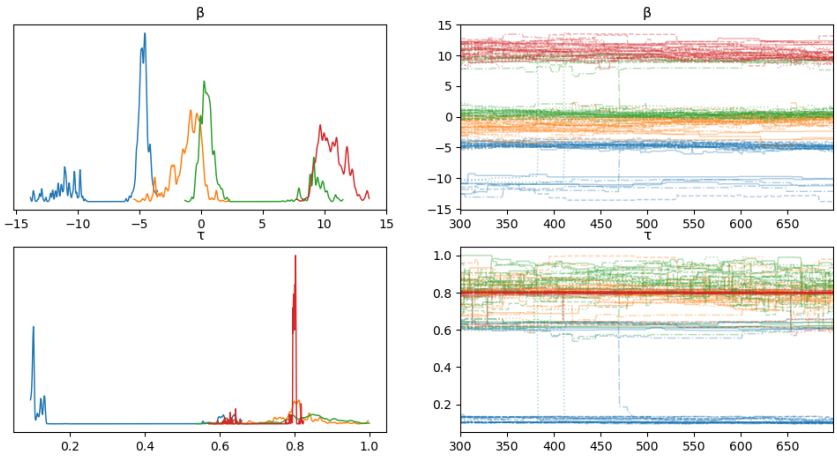


Figure 16: Estimated posterior distribution for b and τ in a linear model with $p = 4$ (left) and the corresponding trace evolution for 400 iterations in the MCMC sampler (right).

There is still another possible situation, one in which there is no label switching but the estimated function has four non-negligible components. This can happen because the different components exploit the additional freedom and "work together", so to speak. In this way we might obtain an estimate that does not resemble the true coefficient function, but that has a very low prediction error. However, this could also work to our detriment and cause the estimated function to be worse prediction-wise than simpler alternatives. This phenomenon is expected to strengthen as the difference between the true and assumed value of p grows larger.

Dependence on p

Another thing we wanted to look at was the dependence of the final prediction result on the chosen value of p, especially when there is no concept of "components" in the underlying model. We can take for example the homoscedastic mixture data set described earlier for the logistic regression problem, and fix the parameter $\eta = 0.01$. The corresponding mean accuracies (in 10 repetitions) for RKHS models with $p = 1, 2, \ldots 10$ components are shown in Figure 17, along with their standard errors (which are arguably not very informative).

It would appear that the methods that use the whole posterior distribution are more stable and somewhat independent of the value of $p > 1$. The other algorithms show

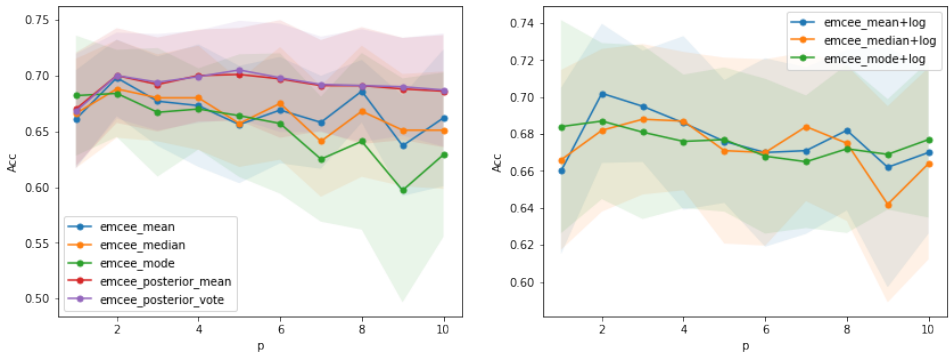


Figure 17: Mean accuracy in 10 repetitions for our logistic RKHS methods as a function of p, using $\eta = 0.01$ and the homoscedastic mixture data set. The corresponding standard errors are shown in faded colors.

a slight downward trend as p increases (although not so much in the variable selection methods), and in general their best results are obtained at $p = 2$ and $p = 8$. We expect that this effect or some small variation of it will remain valid in other situations; however, a profound study of this would be the subject of a different experiment altogether.

Execution times

We summarize below the mean execution time for each independent run of our CV experiment, grouping the simulated data sets by the underlying data generation strategy (since the results within these groups were very similar).

	RKHS	L^2	GBM	Tecator	Moisture	Sugar
Linear regression	260	245	264	230	264	285
	RKHS	L^2	Mixture	Medflies	Growth	Phoneme
Logistic regression	277	263	246	250	218	277

Table 5.1: Mean execution times (in minutes) for the independent runs of the CV experiments in Sections 5.1 and 5.2, grouped when possible by the underlying data generation strategy.

The reference algorithms used were all fast methods that can complete their whole CV loop in just a few seconds, so any comparison with them in this regard is uninteresting. However, this outcome was to be expected, since in general the MCMC methods are

known to have a high computational cost in exchange for the simplicity in modeling they provide. An individual MCMC run is not that expensive time-wise, but even for a modest 5-fold cross-validation loop the effects add up quickly to a considerable amount of time. In Figure 18 we show the evolution of the execution time (in seconds) versus the number of training samples in a generic case, which not surprisingly has a more or less linear behavior. This, however, is not an impediment for using our methods in practice.

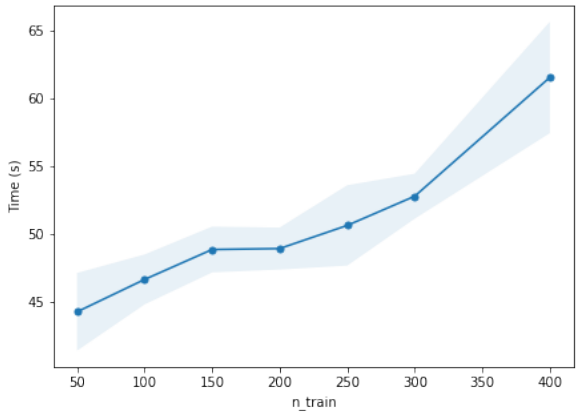


Figure 18: Mean training times of a logistic RKHS model with $p = 5$, as a function of the number of training samples (averaged across 10 independent runs). The values are surrounded by their corresponding standard deviations.

Non-reported experiments

Apart from all the experiments described above, we performed several tests with different configurations and alternatives. Although these trials did not produce better results in general, we mention some of them here to show how the model could be easily extended. In fact, most of these modifications, if not all, are still available in the provided code and can be enabled through specific parameters.

The first thing we tried was to make the parameter p part of the Bayesian model, imposing a categorical prior distribution on it. As explained in Chapter 4, this approach led to many complications and generally poor results. However, the idea of letting the model select how many components it should have based on the data is an interesting one, and in some situations this could greatly improve the results of variable selection procedures based on other model selection criteria.

We also tried changing some prior distributions. For instance, we swapped the improper prior distribution on σ^2 for a Cauchy distribution on $\mathbb{R}^+$. The results were similar, but we ultimately discarded this approach because it introduced yet another hyperparameter into the model with no clear benefit. In addition, we tried to impose a beta distribution on each of the time instants t_j, selecting the corresponding shape parameters in a semi-automatic way so that most of the density fell in the areas where the variance of the process was higher (and thus the effect of individual trajectories was more noticeable). This approach does work, but in our estimation the computational overhead introduced by the beta density function (sometimes resulting in a 1.5x increase in execution time) is not worth the mild improvements, if any.

A lot of minor things were tried, such as standardizing the regressors and/or the responses, smoothing the functional data, changing the proposal of moves in the MCMC sampler, or transforming the values of τ during sampling so that the domain was unconstrained. It is likely that some of these strategies would lead to better results in specific cases, but overall they did not improve the general performance of the prediction procedure.

6 Conclusions

In this work we have addressed some inference and prediction issues involved in the task of learning from functional data. In particular, we have tackled functional linear and logistic regression problems, presenting an alternative to the L^2-models predominant in these scenarios. Our proposed models rely heavily on the theory of RKHS's, offering finite-dimensional approximations to the regression function, but still with a functional perspective. This leads to a conceptually simpler approach that facilitates the interpretation and subsequent analysis, which we believe is a compelling feature for most practitioners. The underlying theoretical aspects are nevertheless far from trivial, so we have also explored some of the related mathematical background in this text.

With the aid of Bayesian methods, we have introduced a computationally feasible approach to parameter estimation within the proposed RKHS models. The flexibility is improved through the use of prior distributions that can introduce pre-existing information into the model. In addition, the fact that we have a complete posterior distribution at our disposal enables us to derive different prediction and variable selection methods, or even perform other types of inference such as the construction of credible intervals. Moreover, the use of MCMC methods to approximate the posterior distribution results in an implementation that is simpler than most existing alternatives. Although this simplicity comes at the expense of increased computational requirements, this computational cost is still reasonable and the model can be used in practice. The execution time only starts to be of some relevance when performing extensive simulations and comparisons, but it is not a serious burden for the use of our methods in practical cases.

We implemented the whole inference and prediction pipeline in the Python programming language, and then used it to carry out a large empirical study to assess the performance of our models against other usual alternatives, functional and otherwise. The results were generally satisfactory, proving that we have competitive methods that achieve good performance in different scenarios, often with low dimensionality and thus increased

tractability and interpretability. Lastly, we believe that with more computational resources we could consider a larger pool of hyperparameters in the model selection phase, and also increase the number of folds and random train/test splits, which would most likely contribute to better results.

6.1 Future work

Some lines of further research suggested by our work are the following:

- To delve further into the relationship between RKHS's and functional data problems, a connection that has proven to be fruitful in many scenarios. A first idea would be to extend the RKHS-based logistic regression model to a generalized functional linear model with an arbitrary link function.
- To try to derive some theoretical properties of our Bayesian predictors. For example, consistency and/or robustness results regarding the posterior distribution would be an excellent complement to the practical side of this work.
- To find other prior distributions for our parameters that perform better in general, or to eliminate the need of the hyperparameters b_0 or η.
- To experiment with other MCMC algorithms for posterior approximation. Specifically, it would be interesting to implement an efficient and reliable reversible-jump MCMC method in Python, which as we have already mentioned is a better fit for our particular Bayesian model. Moreover, we could also adopt a different approach and use variational inference methods to approximate the posterior.

A On the code developed

The Python source code developed for this work is available at the GitHub repository `https://github.com/antcc/rk-bfr`, under a GPLv3 license. The code is adequately documented and is structured in several directories as follows:

- In the `rkbfr` folder we find the files responsible for the implementation of our models, separated according to the functionality they provide.
- The `reference_methods` folder contains our implementation of the functional comparison algorithms that were not available through a standard Python library.
- The `utils` folder contains some utility files for simulation, experimentation and visualization.
- At the root folder we have files for executing our experiments, which accept many user-specified parameters (e.g. number of iterations, type of data set, ...). In particular, the script `results_cv.py` contains the code for our comparison experiments, while script the `results_all.py` allows the execution of our Bayesian methods without a cross-validation loop.

When possible, the code was implemented in a generic way that would allow for easy extensions or derivations. It was also developed with efficiency in mind, so many functions and methods exploit the vectorization capabilities of the *numpy*[9] and *scipy* libraries. Moreover, we followed closely the style of the *scikit-learn* and *scikit-fda* libraries, so our methods are fully compatible and could be integrated (after some minor tweaking) with both of them. The code for the experiments was executed in a machine with 4 cores, and a random seed with value 2022 was set for reproducibility. We provide a script file `launch.sh` that illustrates a typical execution.

Lastly, there are also Jupyter notebooks that demonstrate the use of our methods in a more visual way. Inside these notebooks there is a step-by-step guide on how one might

[9] `https://numpy.org`

execute our algorithms, accompanied by many graphical representations, and offering the possibility of changing multiple parameters to experiment with the code. In addition, there is also a notebook that can be used to generate all the tables and figures of this document pertaining to the experimental results.

B Tables of experimental results

Here we present the tables corresponding to the empirical studies in Sections 5.1 and 5.2. In each case the best and second best results are shown in **blue** and **bold**, respectively.

Functional linear regression

Estimator	BM	fBM	O-U	Sq. exp
emcee_mean	0.913 (0.310)	0.759 (0.068)	0.806 (0.098)	1.408 (1.359)
emcee_median	**0.729 (0.048)**	**0.729 (0.045)**	**0.740 (0.052)**	0.743 (0.041)
emcee_mode	0.735 (0.039)	0.748 (0.068)	0.769 (0.102)	0.803 (0.147)
emcee_posterior_mean	0.743 (0.047)	**0.726 (0.036)**	0.863 (0.416)	0.766 (0.061)
apls	1.003 (0.045)	0.792 (0.030)	1.167 (0.068)	**0.728 (0.035)**
flin	1.219 (0.056)	0.800 (0.022)	1.630 (0.051)	0.738 (0.030)
fpls1	1.235 (0.069)	0.800 (0.024)	1.631 (0.053)	0.738 (0.035)
lasso	**0.727 (0.034)**	0.738 (0.027)	**0.731 (0.039)**	**0.726 (0.032)**
pls1	1.032 (0.116)	0.782 (0.034)	0.974 (0.063)	0.729 (0.041)
ridge	0.920 (0.043)	0.778 (0.021)	0.965 (0.059)	**0.728 (0.035)**
emcee_mean+ridge	0.816 (0.154)	0.749 (0.044)	**0.734 (0.039)**	0.799 (0.175)
emcee_median+ridge	**0.759 (0.063)**	**0.741 (0.041)**	0.751 (0.065)	0.755 (0.058)
emcee_mode+ridge	**0.746 (0.058)**	**0.735 (0.036)**	**0.726 (0.038)**	0.735 (0.036)
fpca+ridge	1.149 (0.041)	0.784 (0.020)	1.420 (0.063)	**0.728 (0.033)**
manual+ridge	1.221 (0.050)	0.784 (0.021)	1.548 (0.072)	**0.727 (0.032)**
pca+ridge	1.153 (0.041)	0.784 (0.022)	1.422 (0.050)	0.730 (0.033)
pls+ridge	0.955 (0.053)	0.783 (0.031)	0.962 (0.059)	0.729 (0.035)
rmh+ridge	1.423 (0.117)	0.847 (0.043)	1.375 (0.266)	1.226 (0.117)

Table B.1: Mean RMSE of predictors for simulated GP data obeying an underlying RKHS model (lower is better). The corresponding standard errors are shown between brackets.

Estimator	BM	fBM	O-U	Sq. exp
emcee_mean	0.769 (0.037)	0.744 (0.061)	0.756 (0.054)	0.794 (0.129)
emcee_median	0.730 (0.049)	0.751 (0.071)	0.718 (0.031)	**0.722 (0.030)**
emcee_mode	0.732 (0.032)	0.739 (0.075)	0.730 (0.038)	0.730 (0.029)
emcee_posterior_mean	0.723 (0.040)	0.720 (0.032)	0.755 (0.079)	**0.726 (0.026)**
apls	**0.715 (0.030)**	**0.710 (0.030)**	**0.710 (0.029)**	**0.726 (0.031)**
flin	0.733 (0.035)	0.727 (0.033)	0.733 (0.035)	0.735 (0.032)
fpls1	0.718 (0.039)	0.726 (0.035)	0.731 (0.034)	**0.726 (0.033)**
lasso	**0.712 (0.027)**	**0.712 (0.028)**	0.717 (0.029)	**0.722 (0.029)**
pls1	0.717 (0.041)	0.720 (0.036)	0.722 (0.029)	0.729 (0.031)
ridge	0.716 (0.029)	0.717 (0.032)	**0.716 (0.032)**	0.727 (0.033)
emcee_mean+ridge	**0.717 (0.029)**	**0.718 (0.030)**	0.719 (0.027)	0.743 (0.052)
emcee_median+ridge	0.722 (0.038)	0.723 (0.038)	**0.717 (0.035)**	0.730 (0.025)
emcee_mode+ridge	0.726 (0.036)	0.735 (0.048)	0.736 (0.030)	0.743 (0.050)
fpca+ridge	**0.717 (0.032)**	**0.718 (0.030)**	0.718 (0.033)	**0.727 (0.031)**
manual+ridge	**0.717 (0.030)**	0.719 (0.030)	**0.716 (0.032)**	**0.728 (0.031)**
pca+ridge	**0.717 (0.032)**	0.720 (0.031)	**0.716 (0.032)**	**0.727 (0.031)**
pls+ridge	**0.719 (0.033)**	0.728 (0.046)	0.720 (0.033)	0.730 (0.031)
rmh+ridge	0.753 (0.029)	**0.713 (0.030)**	0.791 (0.037)	0.812 (0.027)

Table B.2: Mean RMSE of predictors for simulated GP data obeying an underlying L^2 model (lower is better). The corresponding standard errors are shown between brackets.

Estimator	GBM + L^2	GBM + RKHS
emcee_mean	0.948 (0.354)	1.278 (0.622)
emcee_median	0.737 (0.036)	**0.747 (0.031)**
emcee_mode	0.786 (0.106)	0.928 (0.275)
emcee_posterior_mean	0.763 (0.083)	0.786 (0.084)
apls	**0.716 (0.034)**	1.456 (0.170)
flin	0.726 (0.033)	2.427 (0.352)
fpls1	0.731 (0.040)	2.336 (0.365)
lasso	0.726 (0.042)	**0.759 (0.073)**
pls1	**0.710 (0.029)**	1.309 (0.122)
ridge	0.721 (0.035)	1.175 (0.205)
emcee_mean+ridge	0.725 (0.040)	1.432 (1.059)
emcee_median+ridge	0.738 (0.033)	**0.780 (0.093)**
emcee_mode+ridge	0.733 (0.040)	**0.760 (0.073)**
fpca+ridge	**0.716 (0.036)**	1.873 (0.302)
manual+ridge	0.724 (0.046)	2.253 (0.226)
pca+ridge	0.719 (0.036)	1.879 (0.304)
pls+ridge	**0.713 (0.030)**	1.299 (0.125)
rmh+ridge	0.805 (0.051)	1.640 (0.189)

Table B.3: Mean RMSE of predictors for simulated data with GBM regressors (lower is better). The corresponding standard errors are shown between brackets.

Estimator	Moisture	Sugar	Tecator
emcee_mean	1.268 (1.096)	9.207 (9.248)	9.811 (7.446)
emcee_median	0.296 (0.051)	3.130 (2.584)	3.714 (0.922)
emcee_mode	0.301 (0.049)	2.628 (0.700)	3.531 (1.494)
emcee_posterior_mean	0.255 (0.039)	2.813 (0.897)	2.918 (0.222)
flin	0.257 (0.026)	1.978 (0.210)	**2.604 (0.344)**
fpls1	0.236 (0.038)	1.993 (0.223)	**2.604 (0.294)**
lasso	0.242 (0.028)	**1.975 (0.199)**	2.892 (0.270)
pls1	**0.228 (0.023)**	2.045 (0.190)	2.704 (0.467)
ridge	0.221 (0.026)	1.952 (0.235)	3.387 (0.218)
apls	0.234 (0.031)	2.050 (0.238)	2.349 (0.470)
emcee_mean+ridge	0.262 (0.043)	2.020 (0.198)	6.673 (1.037)
emcee_median+ridge	0.260 (0.034)	1.995 (0.219)	5.393 (1.210)
emcee_mode+ridge	0.302 (0.092)	2.037 (0.200)	5.442 (0.563)
fpca+ridge	0.289 (0.035)	**1.976 (0.227)**	9.521 (0.603)
manual+ridge	**0.228 (0.026)**	1.987 (0.227)	4.126 (0.305)
pca+ridge	0.226 (0.027)	1.963 (0.234)	**3.388 (0.218)**
pls+ridge	0.226 (0.025)	2.012 (0.218)	2.415 (0.501)
rmh+ridge	0.327 (0.086)	2.031 (0.216)	5.580 (0.513)

Table B.4: Mean RMSE of predictors for real data sets (lower is better). The corresponding standard errors are shown between brackets.

Functional logistic regression

Estimator	BM	fBM	O-U	Sq. exp
emcee_mean	0.743 (0.052)	0.739 (0.040)	0.734 (0.029)	0.751 (0.039)
emcee_median	0.771 (0.048)	0.716 (0.048)	0.714 (0.049)	0.746 (0.055)
emcee_mode	**0.777 (0.037)**	0.752 (0.043)	0.724 (0.029)	0.760 (0.048)
emcee_posterior_mean	0.764 (0.044)	0.756 (0.046)	0.734 (0.029)	0.753 (0.040)
emcee_posterior_vote	0.765 (0.043)	0.753 (0.040)	**0.747 (0.027)**	0.753 (0.043)
fknn	0.765 (0.027)	**0.768 (0.033)**	0.743 (0.032)	0.738 (0.022)
flda	0.767 (0.055)	0.755 (0.048)	0.735 (0.036)	**0.761 (0.050)**
flog	0.771 (0.033)	0.761 (0.042)	0.745 (0.025)	0.777 (0.040)
fnc	0.743 (0.029)	0.775 (0.042)	0.661 (0.063)	0.755 (0.035)
lda	0.514 (0.054)	0.601 (0.030)	0.578 (0.032)	0.702 (0.059)
log	0.778 (0.031)	0.750 (0.042)	0.761 (0.039)	**0.761 (0.031)**
mdc	0.724 (0.033)	0.762 (0.037)	0.648 (0.052)	0.732 (0.023)
qda	0.499 (0.038)	0.488 (0.041)	0.472 (0.055)	0.483 (0.027)
emcee_mean+logistic	0.781 (0.036)	0.746 (0.038)	0.725 (0.051)	0.750 (0.034)
emcee_median+logistic	0.766 (0.041)	0.749 (0.045)	0.717 (0.024)	0.732 (0.066)
emcee_mode+logistic	0.776 (0.042)	0.746 (0.047)	0.726 (0.025)	0.761 (0.036)
apls+log	**0.783 (0.025)**	0.756 (0.036)	0.739 (0.020)	0.761 (0.028)
apls+nc	0.771 (0.048)	0.745 (0.045)	0.740 (0.022)	0.751 (0.034)
fpca+log	0.773 (0.028)	0.755 (0.038)	0.758 (0.032)	0.765 (0.039)
manual+log	0.753 (0.033)	0.758 (0.040)	0.742 (0.031)	0.754 (0.032)
pca+log	0.780 (0.032)	0.758 (0.036)	**0.756 (0.032)**	0.756 (0.033)
pca+qda	0.751 (0.037)	0.750 (0.049)	0.736 (0.019)	0.741 (0.030)
pls+log	0.786 (0.040)	0.768 (0.037)	0.740 (0.035)	0.766 (0.033)
pls+nc	0.744 (0.032)	**0.766 (0.039)**	0.745 (0.055)	0.767 (0.035)
rkvs+log	0.770 (0.037)	0.757 (0.040)	0.738 (0.026)	**0.772 (0.039)**
rmh+log	0.768 (0.047)	0.760 (0.043)	0.745 (0.019)	0.781 (0.036)

Table B.5: Mean accuracy of classifiers for simulated GP data obeying an underlying RKHS model (higher is better). The corresponding standard errors are shown between brackets.

Estimator	BM	fBM	O-U	Sq. exp
emcee_mean	0.571 (0.053)	**0.557 (0.037)**	**0.594 (0.021)**	0.565 (0.082)
emcee_median	0.557 (0.033)	0.539 (0.045)	0.559 (0.059)	0.556 (0.049)
emcee_mode	0.553 (0.063)	0.513 (0.067)	0.575 (0.038)	0.550 (0.042)
emcee_posterior_mean	0.575 (0.049)	**0.560 (0.036)**	**0.593 (0.022)**	**0.578 (0.039)**
emcee_posterior_vote	0.576 (0.034)	0.554 (0.039)	0.579 (0.023)	**0.576 (0.041)**
fknn	0.557 (0.022)	0.505 (0.047)	0.576 (0.042)	0.557 (0.026)
flda	0.554 (0.032)	0.516 (0.053)	0.543 (0.057)	0.526 (0.049)
flog	0.587 (0.034)	0.542 (0.036)	0.576 (0.038)	**0.578 (0.043)**
fnc	**0.601 (0.036)**	0.545 (0.045)	0.587 (0.037)	0.546 (0.056)
lda	0.507 (0.041)	0.482 (0.056)	0.524 (0.042)	0.525 (0.052)
log	0.576 (0.034)	0.546 (0.033)	0.560 (0.053)	0.554 (0.037)
mdc	**0.605 (0.039)**	0.544 (0.055)	0.584 (0.039)	0.533 (0.042)
qda	0.476 (0.050)	0.518 (0.048)	0.470 (0.071)	0.485 (0.039)
emcee_mean+logistic	**0.583 (0.038)**	**0.575 (0.043)**	0.569 (0.021)	0.559 (0.030)
emcee_median+logistic	0.565 (0.044)	0.552 (0.037)	**0.589 (0.029)**	**0.585 (0.041)**
emcee_mode+logistic	0.568 (0.047)	0.544 (0.036)	0.581 (0.041)	0.557 (0.033)
apls+log	0.556 (0.066)	0.535 (0.040)	0.548 (0.070)	0.560 (0.055)
apls+nc	0.549 (0.056)	0.523 (0.040)	0.545 (0.054)	0.545 (0.047)
fpca+log	0.559 (0.037)	0.548 (0.033)	0.579 (0.034)	0.564 (0.032)
manual+log	0.573 (0.037)	0.542 (0.029)	0.575 (0.043)	0.568 (0.035)
pca+log	0.570 (0.036)	0.541 (0.033)	0.579 (0.028)	0.567 (0.033)
pca+qda	0.567 (0.030)	0.532 (0.045)	0.577 (0.037)	0.574 (0.059)
pls+log	0.564 (0.042)	0.556 (0.028)	0.559 (0.041)	0.564 (0.036)
pls+nc	**0.581 (0.038)**	0.535 (0.048)	**0.589 (0.043)**	0.558 (0.057)
rkvs+log	0.572 (0.058)	0.550 (0.023)	**0.592 (0.018)**	0.567 (0.024)
rmh+log	0.570 (0.036)	**0.557 (0.033)**	0.584 (0.024)	**0.581 (0.025)**

Table B.6: Mean accuracy of classifiers for simulated GP data obeying an underlying L^2 model (higher is better). The corresponding standard errors are shown between brackets.

Estimator	Heteroscedastic	Homoscedastic
emcee_mean	0.513 (0.035)	0.667 (0.053)
emcee_median	0.492 (0.039)	0.647 (0.070)
emcee_mode	**0.543 (0.033)**	0.680 (0.038)
emcee_posterior_mean	0.497 (0.056)	**0.690 (0.050)**
emcee_posterior_vote	0.469 (0.058)	0.684 (0.048)
fknn	**0.574 (0.031)**	0.652 (0.034)
flda	0.489 (0.047)	**0.696 (0.059)**
flog	0.515 (0.045)	0.673 (0.040)
fnc	0.463 (0.069)	0.664 (0.053)
lda	0.493 (0.040)	0.548 (0.046)
log	0.509 (0.055)	0.686 (0.046)
mdc	0.521 (0.052)	0.601 (0.058)
qda	0.502 (0.056)	0.517 (0.039)
emcee_mean+logistic	0.503 (0.054)	0.678 (0.063)
emcee_median+logistic	0.504 (0.041)	0.680 (0.038)
emcee_mode+logistic	0.512 (0.036)	0.681 (0.036)
apls+log	**0.529 (0.034)**	0.684 (0.058)
apls+nc	0.496 (0.039)	0.674 (0.050)
fpca+log	0.481 (0.032)	**0.704 (0.041)**
manual+log	0.496 (0.029)	0.694 (0.044)
pca+log	0.483 (0.030)	0.699 (0.040)
pca+qda	**0.748 (0.055)**	0.703 (0.037)
pls+log	0.489 (0.043)	**0.711 (0.055)**
pls+nc	0.454 (0.037)	0.649 (0.076)
rkvs+log	0.499 (0.041)	0.684 (0.037)
rmh+log	0.516 (0.049)	0.691 (0.030)

Table B.7: Mean accuracy of classifiers for simulated data coming from two different GP's, labeled according to their origin (higher is better). The corresponding standard errors are shown between brackets.

Estimator	Growth	Medflies	Phoneme
emcee_mean	0.858 (0.147)	0.533 (0.041)	0.763 (0.041)
emcee_median	0.894 (0.112)	0.573 (0.032)	0.776 (0.044)
emcee_mode	0.932 (0.034)	0.582 (0.034)	0.770 (0.056)
emcee_posterior_mean	0.926 (0.032)	**0.596 (0.044)**	0.797 (0.035)
emcee_posterior_vote	0.919 (0.046)	0.575 (0.052)	**0.801 (0.031)**
fknn	0.942 (0.040)	0.534 (0.031)	0.760 (0.046)
flda	**0.945 (0.032)**	0.561 (0.020)	0.781 (0.037)
flog	0.935 (0.050)	**0.601 (0.029)**	0.766 (0.041)
fnc	0.735 (0.112)	0.546 (0.038)	0.703 (0.036)
lda	0.894 (0.052)	0.572 (0.019)	0.618 (0.040)
log	**0.965 (0.030)**	0.575 (0.028)	**0.822 (0.026)**
mdc	0.700 (0.087)	0.524 (0.026)	0.663 (0.031)
qda	0.581 (0.000)	0.569 (0.023)	0.457 (0.043)
emcee_mean+logistic	0.906 (0.118)	0.528 (0.029)	0.779 (0.033)
emcee_median+logistic	0.932 (0.049)	0.580 (0.031)	0.796 (0.036)
emcee_mode+logistic	0.935 (0.043)	0.585 (0.032)	0.791 (0.038)
apls+log	0.952 (0.026)	0.572 (0.016)	**0.816 (0.028)**
apls+nc	0.952 (0.041)	0.554 (0.020)	**0.807 (0.032)**
fpca+log	**0.965 (0.030)**	0.551 (0.032)	0.797 (0.021)
manual+log	**0.961 (0.032)**	0.584 (0.018)	0.778 (0.039)
pca+log	0.958 (0.029)	0.576 (0.030)	0.794 (0.030)
pca+qda	0.958 (0.032)	0.567 (0.036)	0.784 (0.034)
pls+log	0.952 (0.036)	0.578 (0.018)	0.804 (0.031)
pls+nc	0.829 (0.094)	0.570 (0.040)	0.754 (0.041)
rkvs+log	0.923 (0.052)	**0.596 (0.032)**	0.796 (0.031)
rmh+log	0.955 (0.030)	**0.606 (0.025)**	0.778 (0.048)

Table B.8: Mean accuracy of classifiers for real data sets (higher is better). The corresponding standard errors are shown between brackets.

Bibliography

Aguilera, A. M., Escabias, M., Preda, C., and Saporta, G. (2010). "Using basis expansions for estimating functional PLS regression: applications with chemometric data". In: *Chemometrics and Intelligent Laboratory Systems* 104 (2), pp. 289–305 (cit. on p. 45).

Aguilera, A. M. and Aguilera-Morillo, M. (2013). "Comparative study of different B-spline approaches for functional data". In: *Mathematical and Computer Modelling* 58 (7-8), pp. 1568–1579 (cit. on p. 5).

Albert, A. and Anderson, J. A. (1984). "On the existence of maximum likelihood estimates in logistic regression models". In: *Biometrika* 71 (1), pp. 1–10 (cit. on p. 30).

Baragatti, M. and Pommeret, D. (2012). "A study of variable selection using g-prior distribution with ridge parameter". In: *Computational Statistics and Data Analysis* 56 (6), pp. 1920–1934 (cit. on p. 26).

Berlinet, A. and Thomas-Agnan, C. (2004). *Reproducing kernel Hilbert spaces in probability and statistics*. Springer (cit. on pp. 10, 11, 13, 15, 16).

Berrendero, J. R., Bueno-Larraz, B., and Cuevas, A. (2019). "An RKHS model for variable selection in functional linear regression". In: *Journal of Multivariate Analysis* 170, pp. 25–45 (cit. on pp. 8, 23, 25).

Berrendero, J. R., Bueno-Larraz, B., and Cuevas, A. (2020a). "On Mahalanobis Distance in Functional Settings." In: *Journal of Machine Learning Research* 21 (9), pp. 1–33 (cit. on p. 5).

Berrendero, J. R., Cholaquidis, A., and Cuevas, A. (2020b). "On a general definition of the functional linear model". In: *arXiv preprint arXiv:2011.05441* (cit. on pp. 7, 8, 26).

Berrendero, J. R., Cuevas, A., and Torrecilla, J. L. (2016). "Variable selection in functional data classification: a maxima-hunting proposal". In: *Statistica Sinica*, pp. 619–638 (cit. on p. 9).

Berrendero, J. R., Cuevas, A., and Torrecilla, J. L. (2018). "On the use of reproducing kernel Hilbert spaces in functional classification". In: *Journal of the American Statistical Association* 113 (523), pp. 1210–1218 (cit. on pp. 8, 45).

Berrendero, J. R., Bueno-Larraz, B., and Cuevas, A. (2022). "On functional logistic regression: some conceptual issues". In: *arXiv preprint arXiv:1812.00721. Manuscript submitted for publication* (cit. on pp. 7, 8, 23, 29, 45).

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Vol. 4. 4. Springer (cit. on p. 9).

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). "Variational inference: A review for statisticians". In: *Journal of the American Statistical Association* 112 (518), pp. 859–877 (cit. on p. 9).

Bro, R. (1999). "Exploratory study of sugar production using fluorescence spectroscopy and multi-way analysis". In: *Chemometrics and Intelligent Laboratory Systems* 46 (2), pp. 133–147 (cit. on p. 43).

Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (2011). *Handbook of Markov Chain Monte Carlo*. CRC press (cit. on p. 9).

Bueno-Larraz, B. and Klepsch, J. (2019). "Variable Selection for the Prediction of C [0, 1]-Valued Autoregressive Processes using Reproducing Kernel Hilbert Spaces". In: *Technometrics* 61 (2), pp. 139–153 (cit. on p. 9).

Cardot, H. and Sarda, P. (2011). "Functional Linear Regression". In: *The Oxford Handbook of Functional Data Analysis*. Ed. by F. Ferraty and Y. Romain. Oxford University Press, pp. 21–46 (cit. on p. 6).

Carey, J. R., Liedo, P., Müller, H.-G., Wang, J.-L., and Chiou, J.-M. (1998). "Relationship of age patterns of fecundity to mortality, longevity, and lifetime reproduction in a large cohort of Mediterranean fruit fly females". In: *The Journals of Gerontology Series A: Biological Sciences and Medical Sciences* 53 (4), B245–B251 (cit. on p. 44).

Celeux, G., Hurn, M., and Robert, C. P. (2000). "Computational and inferential difficulties with mixture posterior distributions". In: *Journal of the American Statistical Association* 95 (451), pp. 957–970 (cit. on p. 33).

Cucker, F. and Smale, S. (2001). "On the mathematical foundations of learning". In: *Bulletin of the American mathematical society* 39 (1), pp. 1–49 (cit. on p. 14).

Cucker, F. and Zhou, D. X. (2007). *Learning theory: an approximation theory viewpoint*. Vol. 24. Cambridge University Press (cit. on p. 14).

Cuevas, A. (2014). "A partial overview of the theory of statistics with functional data". In: *Journal of Statistical Planning and Inference* 147, pp. 1–23 (cit. on p. 2).

Cuevas, A., Febrero, M., and Fraiman, R. (2004). "An anova test for functional data". In: *Computational statistics & data analysis* 47 (1), pp. 111–122 (cit. on p. 5).

Cuevas, A., Febrero, M., and Fraiman, R. (2007). "Robust estimation and classification for functional data via projection-based depth notions". In: *Computational Statistics* 22 (3), pp. 481–496 (cit. on p. 5).

Delaigle, A. and Hall, P. (2012a). "Achieving near perfect classification for functional data". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 74 (2), pp. 267–286 (cit. on pp. 8, 45).

Delaigle, A. and Hall, P. (2012b). "Methodology and theory for partial least squares applied to functional data". In: *The Annals of Statistics* 40 (1), pp. 322–352 (cit. on pp. 5, 44).

Ferraty, F., Hall, P., and Vieu, P. (2010). "Most-predictive design points for functional data predictors". In: *Biometrika* 97 (4), pp. 807–824 (cit. on p. 9).

Ferraty, F. and Vieu, P. (2006). *Nonparametric functional data analysis: theory and practice*. Vol. 76. Springer (cit. on p. 5).

Firth, D. (1993). "Bias reduction of maximum likelihood estimates". In: *Biometrika* 80 (1), pp. 27–38 (cit. on p. 30).

Foreman-Mackey, D., Hogg, D. W., Lang, D., and Goodman, J. (2013). "emcee: the MCMC hammer". In: *Publications of the Astronomical Society of the Pacific* 125 (925), p. 306 (cit. on pp. 22, 36).

Fraiman, R. and Muniz, G. (2001). "Trimmed means for functional data". In: *Test* 10 (2), pp. 419–440 (cit. on p. 5).

Galeano, P., Joseph, E., and Lillo, R. E. (2015). "The Mahalanobis distance for functional data with applications to classification". In: *Technometrics* 57 (2), pp. 281–291 (cit. on p. 5).

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian Data Analysis*. Chapman and Hall/CRC (cit. on pp. 35, 39).

Geman, S. and Geman, D. (1984). "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6 (6), pp. 721–741 (cit. on p. 20).

Ghosh, A. K. and Chaudhuri, P. (2005). "On maximum depth and related classifiers". In: *Scandinavian Journal of Statistics* 32 (2), pp. 327–350 (cit. on p. 45).

Giraud-Carrier, C. and Provost, F. (2005). "Toward a justification of meta-learning: Is the no free lunch theorem a show-stopper". In: *Proceedings of the ICML-2005 Workshop on Meta-learning*, pp. 12–19 (cit. on p. 46).

Goodman, J. and Weare, J. (2010). "Ensemble samplers with affine invariance". In: *Communications in applied mathematics and computational science* 5 (1), pp. 65–80 (cit. on pp. 21, 36).

Green, P. J. (1995). "Reversible jump Markov chain Monte Carlo computation and Bayesian model determination". In: *Biometrika* 82 (4), pp. 711–732 (cit. on p. 35).

Grollemund, P.-M., Abraham, C., Baragatti, M., and Pudlo, P. (2019). "Bayesian Functional Linear Regression with Sparse Step Functions". In: *Bayesian Analysis* 14 (1), pp. 111–135 (cit. on pp. 8, 26, 35, 36).

Hastie, T., Buja, A., and Tibshirani, R. (1995). "Penalized discriminant analysis". In: *The Annals of Statistics* 23 (1), pp. 73–102 (cit. on p. 44).

Hoffman, M. D. and Gelman, A. (2014). "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." In: *Journal of Machine Learning Research* 15 (1), pp. 1593–1623 (cit. on p. 37).

Horváth, L. and Kokoszka, P. (2012). *Inference for Functional Data with Applications*. Springer Series in Statistics. Springer New York (cit. on p. 2).

Hsing, T. and Eubank, R. (2015). *Theoretical foundations of functional data analysis, with an introduction to linear operators*. Vol. 997. John Wiley & Sons (cit. on pp. 2, 14).

James, G. and Hastie, T. (2001). "Functional linear discriminant analysis for irregularly sampled curves". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63 (3), pp. 533–550 (cit. on p. 5).

James, G., Wang, J., and Zhu, J. (2009). "Functional linear regression that's interpretable". In: *The Annals of Statistics* 37 (5A), pp. 2083–2108 (cit. on p. 9).

Janson, S. (1997). *Gaussian Hilbert Spaces*. Cambridge University Press (cit. on p. 13).

Jasra, A., Holmes, C. C., and Stephens, D. A. (2005). "Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modeling". In: *Statistical Science* 20 (1), pp. 50–67 (cit. on p. 34).

Jeffreys, H. (1946). "An invariant form for the prior probability in estimation problems". In: *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 186 (1007), pp. 453–461 (cit. on p. 26).

Kalivas, J. H. (1997). "Two data sets of near infrared spectra". In: *Chemometrics and Intelligent Laboratory Systems* 37 (2), pp. 255–259 (cit. on p. 43).

Lee, J., Sung, W., and Choi, J.-H. (June 2015). "Metamodel for Efficient Estimation of Capacity-Fade Uncertainty in Li-Ion Batteries for Electric Vehicles". In: *Energies* 8, pp. 5538–5554 (cit. on p. 20).

Lian, H., Choi, T., Meng, J., and Jo, S. (2016). "Posterior convergence for Bayesian functional linear regression". In: *Journal of Multivariate Analysis* 150, pp. 27–41 (cit. on p. 10).

Loève, M. (1948). "Fonctions aléatoires du second ordre". In: *Processus stochastiques et mouvement Brownien (P. Lévy)*. Paris: Gauthier-Villars, pp. 299–352 (cit. on p. 16).

López-Pintado, S. and Romo, J. (2009). "On the concept of depth for functional data". In: *Journal of the American statistical Association* 104 (486), pp. 718–734 (cit. on p. 5).

Lukić, M. and Beder, J. (2001). "Stochastic processes with sample paths in reproducing kernel Hilbert spaces". In: *Transactions of the American Mathematical Society* 353 (10), pp. 3945–3969 (cit. on p. 16).

Ma, C. and Hernández-Lobato, J. M. (2021). "Functional variational inference based on stochastic process generators". In: *Advances in Neural Information Processing Systems* 34, pp. 21795–21807 (cit. on p. 10).

Ma, C., Li, Y., and Hernández-Lobato, J. M. (2019). "Variational implicit processes". In: *International Conference on Machine Learning*. PMLR, pp. 4222–4233 (cit. on p. 10).

Marin, J.-M., Mengersen, K., and Robert, C. P. (2005). "Bayesian Modelling and Inference on Mixtures of Distributions". In: *Bayesian Thinking*. Ed. by D. Dey and C. Rao. Vol. 25. Handbook of Statistics. Elsevier, pp. 459–507 (cit. on p. 34).

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). "Equation of state calculations by fast computing machines". In: *The Journal of Chemical Physics* 21 (6), pp. 1087–1092 (cit. on p. 19).

Müller, H.-G. and Stadtmüller, U. (2005). "Generalized functional linear models". In: *The Annals of Statistics* 33 (2), pp. 774–805 (cit. on p. 5).

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press (cit. on p. 9).

Neal, R. M. (2011). "MCMC using Hamiltonian dynamics". In: *Handbook of Markov Chain Monte Carlo (S. Brooks, A. Gelman, G. Jones and X.-L. Meng)*. Chapman and Hall/CRC (cit. on p. 20).

Ortega, L. A., Santana, S. R., and Hernández-Lobato, D. (2022). "Deep Variational Implicit Processes". In: *arXiv preprint arXiv:2206.06720* (cit. on p. 10).

Papastamoulis, P. (2015). "label. switching: An R package for dealing with the label switching problem in MCMC outputs". In: *arXiv preprint arXiv:1503.02271* (cit. on p. 34).

Parzen, E. (1961). "An approach to time series analysis". In: *The Annals of Mathematical Statistics* 32 (4), pp. 951–989 (cit. on p. 17).

Piironen, J. and Vehtari, A. (2017). "Comparison of Bayesian predictive methods for model selection". In: *Statistics and Computing* 27 (3), pp. 711–735 (cit. on p. 35).

Pillai, N. S., Wu, Q., Liang, F., Mukherjee, S., and Wolpert, R. L. (2007). "Characterizing the Function Space for Bayesian Kernel Models." In: *Journal of Machine Learning Research* 8 (8) (cit. on p. 16).

Poß, D., Liebl, D., Kneip, A., Eisenbarth, H., Wager, T. D., and Barrett, L. F. (2020). "Superconsistent estimation of points of impact in non-parametric regression with functional predictors". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 82 (4), pp. 1115–1140 (cit. on p. 9).

Preda, C., Saporta, G., and Lévéder, C. (2007). "PLS classification of functional data". In: *Computational Statistics* 22 (2), pp. 223–235 (cit. on p. 45).

Ramsay, J. O. and Silverman, B. W. (2005). *Functional Data Analysis*. Springer (cit. on pp. 2, 6).

Rasmussen, C. E. (2004). "Gaussian Processes in Machine Learning". In: *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Revised Lectures*. Springer, pp. 63–71 (cit. on p. 5).

Reiss, P. T., Goldsmith, J., Shang, H. L., and Ogden, R. T. (2017). "Methods for scalar-on-function regression". In: *International Statistical Review* 85 (2), pp. 228–249 (cit. on p. 6).

Robert, C. P., Casella, G., and Casella, G. (1999). *Monte Carlo statistical methods*. Vol. 2. Springer (cit. on p. 20).

Rodríguez, C. E. and Walker, S. G. (2014). "Label switching in Bayesian mixture models: Deterministic relabeling strategies". In: *Journal of Computational and Graphical Statistics* 23 (1), pp. 25–45 (cit. on p. 34).

Saitoh, S. and Sawano, Y. (2016). *Theory of reproducing kernels and applications*. Springer (cit. on p. 13).

Salvatier, J., Wiecki, T. V., and Fonnesbeck, C. (2016). "Probabilistic programming in Python using PyMC3". In: *PeerJ Computer Science* 2, e55 (cit. on p. 37).

Schölkopf, B., Herbrich, R., and Smola, A. J. (2001). "A generalized representer theorem". In: *International conference on computational learning theory*. Springer, pp. 416–426 (cit. on p. 17).

Shi, J. Q. and Choi, T. (2011). *Gaussian process regression analysis for functional data*. CRC Press (cit. on p. 5).

Shin, H. (2008). "An extension of Fisher's discriminant analysis for stochastic processes". In: *Journal of Multivariate Analysis* 99 (6), pp. 1191–1216 (cit. on p. 5).

Simola, U., Cisewski-Kehe, J., and Wolpert, R. L. (2021). "Approximate Bayesian computation for finite mixture models". In: *Journal of Statistical Computation and Simulation* 91 (6), pp. 1155–1174 (cit. on p. 34).

Stephens, M. (2000). "Dealing with label switching in mixture models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 62 (4), pp. 795–809 (cit. on p. 33).

Sun, S., Zhang, G., Shi, J., and Grosse, R. (2019). "Functional variational Bayesian neural networks". In: *arXiv preprint arXiv:1903.05779* (cit. on p. 10).

Torrecilla, J. L., Ramos-Carreño, C., Sánchez-Montañés, M., and Suárez, A. (2020). "Optimal classification of Gaussian processes in homo-and heteroscedastic settings". In: *Statistics and Computing* 30 (4), pp. 1091–1111 (cit. on pp. 8, 50).

Torrecilla, J. L. and Suárez, A. (2016). "Feature selection in functional data classification with recursive maxima hunting". In: *Advances in Neural Information Processing Systems* 29 (cit. on p. 44).

Tuddenham, R. D. and Snyder, M. M. (1954). "Physical growth of California boys and girls from birth to eighteen years". In: *Publications in child development. University of California, Berkeley* 1 (2), pp. 183–364 (cit. on p. 44).

Ullah, S. and Finch, C. F. (2013). "Applications of functional data analysis: A systematic review". In: *BMC Medical Research Methodology* 13 (1), pp. 1–12 (cit. on p. 5).

Van Der Vaart, A. W. and Van Zanten, J. H. (2008). "Reproducing kernel Hilbert spaces of Gaussian priors". In: *Pushing the limits of contemporary statistics: contributions in*

honor of Jayanta K. Ghosh. Institute of Mathematical Statistics, pp. 200–222 (cit. on p. 10).

Vapnik, V. (1991). "Principles of risk minimization for learning theory". In: *Advances in Neural Information Processing Systems* 4 (cit. on p. 17).

Wahba, G. (1990). *Spline models for observational data*. SIAM (cit. on p. 16).

Wales, D. J. and Doye, J. P. (1997). "Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms". In: *The Journal of Physical Chemistry A* 101 (28), pp. 5111–5116 (cit. on p. 37).

Wegelin, J. A. (2000). *A survey of Partial Least Squares (PLS) methods, with emphasis on the two-block case*. Tech. rep. (cit. on p. 44).

Yuan, M. and Cai, T. T. (2010). "A reproducing kernel Hilbert space approach to functional linear regression". In: *The Annals of Statistics* 38 (6), pp. 3412–3444 (cit. on p. 8).

Zellner, A. (1986). "On assessing prior distributions and Bayesian regression analysis with g-prior distributions". In: *Bayesian Inference and Decision Techniques* (cit. on p. 26).

Acknowledgments

This research has been partially supported by grant PRE2020-095147 of the Spanish Ministry of Science and Innovation (MICINN). We also wish to acknowledge the computational resources provided by the Centro de Computación Científica-Universidad Autónoma de Madrid (CCC-UAM), which were instrumental in performing the experiments in this work.