Nonlinear Functional Output Regression: A Dictionary Approach

Dimitri Bouche[1] **Marianne Clausel**[2] **François Roueff**[1] **Florence d'Alché-Buc**[1]
[1]LTCI, Télécom Paris, Institut Polytechnique de Paris [2] Université de Lorraine, CNRS, IECL

Abstract

To address functional-output regression, we introduce projection learning (PL), a novel dictionary-based approach that learns to predict a function that is expanded on a dictionary while minimizing an empirical risk based on a functional loss. PL makes it possible to use non orthogonal dictionaries and can then be combined with dictionary learning; it is thus much more flexible than expansion-based approaches relying on vectorial losses. This general method is instantiated with reproducing kernel Hilbert spaces of vector-valued functions as kernel-based projection learning (KPL). For the functional square loss, two closed-form estimators are proposed, one for fully observed output functions and the other for partially observed ones. Both are backed theoretically by an excess risk analysis. Then, in the more general setting of integral losses based on differentiable ground losses, KPL is implemented using first-order optimization for both fully and partially observed output functions. Eventually, several robustness aspects of the proposed algorithms are highlighted on a toy dataset; and a study on two real datasets shows that they are competitive compared to other nonlinear approaches. Notably, using the square loss and a learnt dictionary, KPL enjoys a particularily attractive trade-off between computational cost and performances.

1 INTRODUCTION

In a large number of fields such as Biomedical Signal Processing, Epidemiology Monitoring, Speech and Acoustics, Climate Science, each data instance consists in a high number of measurements of a common underlying phenomenon. Such high-dimensional data generally enjoys strong smoothness across features. To exploit that structure, it can be interesting to model the underlying functions rather than the vectors of discrete measurements we observe, opening the door to functional data analysis (FDA; Ramsay and Silverman, 2005). In practice, FDA relies on the assumption that the sampling rate of the observations is high enough to consider them as functions. Of special interest is the general problem of functional output regression (FOR) in which the output variable is a function and the input variable can be of any type, including a function.

While functional linear models have received a great deal of attention—see the additive linear model and its variations (Ramsay and Silverman, 2005; Morris, 2015, and references therein)—, nonlinear ones have been less studied. Reimherr et al. (2018) extend the function-to-function additive linear model by considering a tri-variate regression function in a reproducing kernel Hilbert space (RKHS). In non-parametric statistics, Ferraty and Vieu (2006) introduce variations of the Nadaraya-Watson kernel estimator for outputs in a Banach space. Oliva et al. (2015) rather project both input and output functions on orthogonal bases and regress the obtained output coefficients separately on the input ones using approximate kernel ridge regressions (KRR). Finally, extending kernel methods to functional data, Lian (2007) introduces a function-valued KRR. In that context Kadri et al. (2010, 2016) propose a solution based on the approximate inversion of an infinite-dimensional linear operator and studies richer kernels. We give more details on those methods and compare them with our approach in Section 6.1.

In this paper we introduce a novel dictionary-based approach to FOR. We learn to predict a function that is expanded on a dictionary while minimizing an empirical risk based on a functional loss. We call this approach *projection learning* (PL). It can be instantiated with any machine learning algorithm outputting vectors using a wide range of functional losses. PL also makes it possible to use non-orthonormal dictionaries. It represents a crucial advantage as complex functions generally cannot be well represented using

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

few vectors in conventional bases. They can however be compressed very efficiently using learnt redundant dictionaries (Mallat, 2008). Then, to solve FOR problems with complex output functions, PL combined with dictionary learning (DL) algorithms (Dumitrescu and Irofti, 2018) can be both fast and accurate. In practice functions are not fully observed; discrete observations are rather available. PL can accommodate such realistic case without making any assumptions on the sampling grids, either by learning with an estimated gradient or by plugging in an estimator in a closed-form functional solution.

Then, considering vector-valued RKHSs (vv-RKHS, Micchelli and Pontil, 2005), we introduce *kernel-based projection learning* (KPL). Vv-RKHSs extend the scope of kernel methods to vector-valued functions by means of operator-valued kernels (OVK)—see Section A of the Supplement for an introduction. They constitute a principled way of performing vector-valued nonlinear regression considering any type of input data for which a kernel can be defined (Shawe-Taylor and Cristianini, 2004). Learning typically relies on a representer theorem which remains valid for the KPL problem.

Contributions. We introduce PL, a novel dictionary-based approach to FOR. It can handle non orthonormal dictionaries and can thus be combined with dictionary learning. Then, we focus on KPL, an instantiation based on vv-RKHSs. For the functional square loss, we propose two estimators, one for fully observed output functions and another for partially observed ones. Both are backed with an excess risk bound. For an integral loss based on a differentiable ground loss, we solve KPL using first-order optimization and show that the gradient can easily be estimated from partially observed functions. Eventually, we study different robustness aspects of the proposed algorithms on a toy dataset; and demonstrate on two real datasets that they can be competitive with other nonlinear FOR methods while keeping the computational cost significantly lower.

Notations and context. We assimilate the spaces $(\mathbb{R}^d)^n$ and $\mathbb{R}^{d\times n}$. The concatenation of vectors $(u_i)_{i=1}^n \in \mathbb{R}^{d\times n}$ is denoted $\mathsf{vec}((u_i)_{i=1}^n) \in \mathbb{R}^{dn}$. For $n \in \mathbb{N}^*$, we use the shorthand $[n]$ for the set $\{1,\dots,n\}$. We denote by $\mathcal{F}(\mathcal{X},\mathcal{Y})$ the space of functions from $\mathcal{X}$ to $\mathcal{Y}$. For two Hilbert spaces $\mathcal{U}$ and $\mathcal{Y}$, $\mathcal{L}(\mathcal{U},\mathcal{Y})$ is the set of bounded linear operators from $\mathcal{U}$ to $\mathcal{Y}$ and $\mathcal{L}(\mathcal{U}) := \mathcal{L}(\mathcal{U},\mathcal{U})$. The adjoint of a linear operator A is denoted $\mathsf{A}^\#$. For $\mathcal{U} = \mathbb{R}^d$, we introduce $\mathsf{A}_{(n)} \in \mathcal{L}(\mathbb{R}^{dn},\mathcal{Y}^n)$ as $\mathsf{A}_{(n)} : \mathsf{vec}((u_i)_{i=1}^n) \longmapsto (\mathsf{A}u_1,\dots,\mathsf{A}u_n)$ and $\mathsf{A}_{\mathsf{mat},(n)} \in \mathcal{L}(\mathbb{R}^{d\times n},\mathcal{Y}^n)$ as $\mathsf{A}_{\mathsf{mat},(n)} : (u_i)_{i=1}^n \longmapsto (\mathsf{A}u_1,\dots,\mathsf{A}u_n)$. For $\mathsf{B} \in \mathbb{R}^{p\times q}, \mathsf{C} \in \mathbb{R}^{d\times n}$, $\mathsf{B}\otimes\mathsf{C} \in \mathbb{R}^{pd\times qn}$ denotes the Kronecker product. Finally $\mathsf{L}^2(\Theta)$ stands for the Hilbert space of real-valued square integrable functions on a given compact subset $\Theta \subset \mathbb{R}^q$; without loss of generality we suppose that $|\Theta| := \int_\Theta 1\mathrm{d}\theta = 1$.

2 PROJECTION LEARNING

2.1 Functional output regression

Let $\mathcal{X}$ be a measurable space and (X,Y) be a couple of random variables on $\mathcal{Z} := \mathcal{X}\times\mathsf{L}^2(\Theta)$ with joint probability distribution ρ. To introduce the FOR problem, we define a functional loss ℓ as a real-valued function over $\mathsf{L}^2(\Theta)\times\mathsf{L}^2(\Theta)$. Examples of functional losses include the functional square loss and more generally, any integral of a ground loss $l : \mathbb{R}\times\mathbb{R}\to\mathbb{R}$. Particularly, given such ground loss l, for $(y_0,y_1)\in\mathsf{L}^2(\Theta)\times\mathsf{L}^2(\Theta)$, a functional loss ℓ can be defined as:

$$\ell(y_0,y_1) = \int_\Theta l(y_0(\theta),y_1(\theta))\mathrm{d}\theta. \tag{1}$$

Specifically, taking the square loss as ground loss $l(y_0(\theta),y_1(\theta)) = (y_0(\theta)-y_1(\theta))^2$ we obtain the functional square loss $\ell_2(y_0,y_1) := \|y_0-y_1\|^2_{\mathsf{L}^2(\Theta)}$, widely used in the literature (Kadri et al., 2010).

Given such functional loss ℓ and a hypothesis class $\mathcal{G}\subset\mathcal{F}(\mathcal{X},\mathsf{L}^2(\Theta))$, we now define the FOR problem as

$$\min_{f\in\mathcal{G}} \mathcal{R}(f) := \mathbb{E}_{(\mathsf{X},\mathsf{Y})\sim\rho}\left[\ell(\mathsf{Y},f(\mathsf{X}))\right]. \tag{2}$$

However, we have access to the joint probability distribution ρ only through an observed sample. The aim is then to approximately solve the above problem using the available data. We study two possible settings.

In the first one, the output functions are *fully observed*. Our sample $\mathbf{z} := (x_i,y_i)_{i=1}^n$ then consists of $n\in\mathbb{N}$ i.i.d. realizations drawn from ρ, this setting coincides with the so-called *dense* one described in FDA (Kokoska and Reimherr, 2017). By contrast, in the *partially observed* setting (also referred to as the *sparse* one, described and studied in Kokoska and Reimherr (2017); Li and Hsing (2010); Cai and Yuan (2011)), the output functions are observed on grids which may be irregular, subject to randomness and potentially different for each function. Even though the former scenario is relatively frequent in theoretical works, the latter can be more realistic.

In the partially observed setting, we suppose that we only observe each y_i on a random sample of locations, $\theta_i := (\theta_{ip})_{p=1}^{m_i} \in \Theta^{m_i}$, drawn from a probability distribution μ. For the sake of simplicity, μ is chosen as the uniform distribution on Θ and the draws of locations are supposed to be independent. The learning problem depicted in Equation (2) has now to be solved using a partially observed functional output sample:

$$\widetilde{\mathbf{z}} := (x_i,(\theta_i,\widetilde{y}_i))_{i=1}^n, \tag{3}$$

where for all $i \in [n]$, $\theta_i \in \Theta^{m_i}$, $\widetilde{y}_i \in \mathbb{R}^{m_i}$ with $m_i \in \mathbb{N}^*$ the number of observations available for the i-th function, and for all $p \in [m_i]$, $\theta_{ip} \in \Theta$ and $\widetilde{y}_{ip} \in \mathbb{R}$.

In this paper, we propose a novel angle to address the FOR problem using both types of samples.

2.2 Approximated FOR

To tackle Problem (2), we propose to learn to predict expansion coefficients on a dictionary of functions $\phi := (\phi_l)_{l=1}^d \in \mathsf{L}^2(\Theta)^d$ with $d \in \mathbb{N}^*$ (considerations on the choice of this dictionary are postponed to Section 3). We then introduce the following linear operator:

Definition 2.1. (Projection operator) For a dictionary ϕ, the associated projection operator Φ is defined by $\Phi: \quad u \in \mathbb{R}^d \longmapsto \sum_{l=1}^d u_l \phi_l \in \mathsf{L}^2(\Theta)$.

We can give an explicit expression of $\Phi^{\#}$ as well as a matrix representation of $\Phi^{\#}\Phi$.

Lemma 2.1. *The adjoint of* Φ *is given by* $\Phi^{\#}: g \in \mathsf{L}^2(\Theta) \longmapsto (\langle \phi_l, g \rangle_{\mathsf{L}^2(\Theta)})_{l=1}^d \in \mathbb{R}^d$. *Thus we have* $\Phi^{\#}\Phi = (\langle \phi_l, \phi_s \rangle_{\mathsf{L}^2(\Theta)})_{l,s=1}^d$.

The core idea of PL is to define a simpler model $f(x) = \Phi h(x)$ in Problem (2), where $h : \mathcal{X} \longmapsto \mathbb{R}^d$ is a vector-valued function. This yields the problem

$$\min_{h \in \mathcal{H}} \mathcal{R}(\Phi \circ h), \tag{4}$$

that we can solve using a sample from one or the other of the two observation settings previously defined.

In the fully observed setting, we can minimize over $\mathcal{H} \subset \mathcal{F}(\mathcal{X}, \mathbb{R}^d)$ the empirical counterpart of the true risk based on $\mathbf{z}$, $\widehat{\mathcal{R}}(\Phi \circ h, \mathbf{z}) := \frac{1}{n}\sum_{i=1}^n \ell(y_i, \Phi h(x_i))$, with some additional penalty $\Omega_{\mathcal{H}} : \mathcal{H} \longrightarrow \mathbb{R}$ to control the model complexity:

$$\min_{h \in \mathcal{H}} \widehat{\mathcal{R}}(\Phi \circ h, \mathbf{z}) + \lambda \Omega_{\mathcal{H}}(h), \tag{5}$$

with $\lambda > 0$. In other words, we search a solution in the hypothesis space $\{f : x \longmapsto \Phi h(x),\ h \in \mathcal{H}\}$ and solve a function-valued problem at the price of solving a vector-valued one in $\mathcal{H}$. Even though a vector-valued function is learned, the loss remains a functional one. Moreover, any predictive model devoted to vectorial output regression (e. g. neural networks, random forests, kernel methods etc.) is eligible. We regularize our model through the vector-valued function h.

To tackle the partially observed setting, rather than formulating an empirical counterpart of the true risk based on $\widetilde{\mathbf{z}}$, we exploit specific properties of the learning algorithms proposed in Section 4. Namely in our closed form ridge estimator (Proposition 4.2) or in the gradient (Equation (11)), the output functions only appear through scalar products with elements of the dictionary. We can then estimate those from $((\theta_i, \widetilde{y}_i))_{i=1}^n$ and use a plug-in strategy. Interestingly, computing the gradient for the data attach term in Problem (5) shows that this is a feature of projection learning which is not specific to the vv-RKHS instantiation (see Section F.1 of the Supplement for details).

3 DICTIONARIES

In solving Problem (4) instead of Problem (2), we restrict the predictions of our model to $\text{Span}(\phi)$, the space of linear combinations of functions of ϕ. As a result ϕ must be chosen so that the functions $(y_i)_{i=1}^n$ can be approximated accurately by elements from $\text{Span}(\phi)$. To achieve this, several strategies are possible.

3.1 General dictionaries

Orthonormal and Riesz bases. We can consider families of functions known to provide sharp approximations of functions belonging to $\mathsf{L}^2(\Theta)$. Orthogonal bases such as Fourier bases or wavelets bases (DeVore et al., 1992), as well as Riesz bases (see Definition 5.1) such as splines (Oswald, 1990), have proved their efficiency in signal compression. In practice, a choice among those families can be made from observed properties of the output functions or prior information on the generating process. Then within a family, dictionaries with different parameters (number of functions and/or other parameters) can be considered. A cross-validation can be performed to select one.

Families of random functions, such as random Fourier features (RFFs, Rahimi and Recht, 2008a) can enjoy good approximation properties as well. Through the choice of such family, we approximate the output functions in a space that is dense in a RKHS (Rahimi and Recht, 2008b). The link with this RKHS can moreover be made explicit as a family is associated to a given kernel. The kernel can then be chosen by cross-validation and number of functions to include results from a precision/computation time trade-off.

3.2 Dictionary learning

When the output functions are too complex, selecting a dictionary can however be difficult. The choice of a family may not be evident and it may take too many atoms (functions) to reach a satisfying approximation precision. While functional principal component analysis (FPCA; Ramsay and Silverman, 2005) addresses the first issue by ensuring that $\text{Span}(\phi)$ is close to $\text{Span}((y_i)_{i=1}^n)$, it does not address the second one. If the functions at hand are too complex, a very large number of eigenfunctions will be necessary to reach an acceptable approximation quality. By opposition, dictionary learning (DL) solves both problems; it can generally synthesize faithfully the properties of a complicated set of functions while using very few atoms (Mairal et al., 2009). The DL problem is of the form

$$\min_{\phi\in\mathcal{C},\beta\in\mathbb{R}^{d\times n}} \frac{1}{n}\sum_{i=1}^{n}\left(\|y_i-\Phi\beta_i\|^2_{\mathsf{L}^2(\Theta)}+\tau\Omega_{\mathbb{R}^d}(\beta_i)\right), \quad (6)$$

where $\mathcal{C}$ is a set of constraint for the dictionary, $\Omega_{\mathbb{R}^d}:\mathbb{R}^d\longrightarrow\mathbb{R}$ is a penalty on the learned representation coefficients and $\tau>0$ is a trade-off parameter. $\mathcal{C}:=\{\phi\in\mathsf{L}^2(\Theta)^d,\ \|\phi_l\|^2_{\mathsf{L}^2(\Theta)}\leq 1,\ \ l\in[d]\}$ and $\Omega_{\mathbb{R}^d}:=\|.\|_1$ are the most common choices (Lee et al., 2007; Mairal et al., 2009), and most existing algorithms are based on alternating optimization schemes (Dumitrescu and Irofti, 2018, and references therein).

As opposed to other dictionary based methods (Oliva et al., 2015), KPL can handle the resulting non orthonormal dictionary and can thus benefit from the compression power of DL. Then combining the two, we obtain a FOR method that can deal directly with complex functional-output datasets at a low computational cost. Admittedly, solving Problem (6) has a cost, which must however be mitigated. Many efficient algorithms exist (Dumitrescu and Irofti, 2018) and the dictionary moreover needs to be learnt only once (when selecting other parameters through cross-validation, it needs only be learnt once per fold).

4 VV-RKHS INSTANTIATION

We now focus on projection learning using vv-RKHSs.

4.1 Vv-RKHSs and representer theorem

Let $\mathsf{K}:\mathcal{X}\times\mathcal{X}\longmapsto\mathcal{L}(\mathbb{R}^d)$ be an OVK and $\mathcal{H}_{\mathsf{K}}\subset\mathcal{F}(\mathcal{X},\mathbb{R}^d)$ its associated vv-RKHS. For $x\in\mathcal{X}$, we define $\mathsf{K}_x\in\mathcal{L}(\mathbb{R}^d,\mathcal{H}_{\mathsf{K}})$ as $\mathsf{K}_x:u\longmapsto\mathsf{K}_xu$, with $\mathsf{K}_xu:x'\longmapsto\mathsf{K}(x',x)u$. We consider Problem (5) taking $\mathcal{H}=\mathcal{H}_{\mathsf{K}}$ as vector-valued hypothesis class. Setting the regularization as $\Omega_{\mathcal{H}_{\mathsf{K}}}(h):=\|h\|^2_{\mathcal{H}_{\mathsf{K}}}$ yields the following instantiation of PL with vv-RKHS:

$$\min_{h\in\mathcal{H}_{\mathsf{K}}}\frac{1}{n}\sum_{i=1}^{n}\ell(y_i,\Phi h(x_i))+\lambda\|h\|^2_{\mathcal{H}_{\mathsf{K}}}. \quad (7)$$

To solve Problem (7), we show in Proposition 4.1 that it benefits from a representer theorem, which proof is given in Section B.1 of the Supplement. It can then be restated as a problem with dn variables.

Proposition 4.1. ***(Representer theorem)*** *For ℓ continuous and convex with respect to its second argument, Problem (7) admits a unique minimizer $h^\lambda_{\mathbf{z}}$. Moreover there exists $\alpha\in\mathbb{R}^{d\times n}$ such that*

$$h^\lambda_{\mathbf{z}}=\sum_{j=1}^{n}\mathsf{K}_{x_j}\alpha_j.$$

Choice of kernels. In vv-RKHSs, the choice of the kernel determines the regularization conveyed by the RKHS norm. In practice, the separable kernel is often used: $\mathsf{K}=k\mathsf{B}:(x_0,x_1)\longmapsto k(x_0,x_1)\mathsf{B}$ (Alvarez et al., 2012), with k a scalar kernel on $\mathcal{X}$ and $\mathsf{B}\in\mathbb{R}^{d\times d}$ a positive definite symmetric matrix encoding relations between the output variables. In KPL, B can encode prior information on the dictionary. A diagonal matrix can for instance penalize higher frequencies/scales more. We exploit this with wavelets in the experiments related to biomedical imaging in Section 6.4.

4.2 Ridge solution

In this section, we focus on the functional square loss.

Fully observed setting. By Proposition 4.1, Problem (7) can be rewritten as

$$\min_{\alpha\in\mathbb{R}^{d\times n}}\frac{1}{n}\left\|\mathbf{y}-\Phi_{(n)}\mathbf{K}\mathsf{vec}(\alpha)\right\|^2_{\mathsf{L}^2(\Theta)^n} + \lambda\langle\mathsf{vec}(\alpha),\mathbf{K}\mathsf{vec}(\alpha)\rangle_{\mathbb{R}^{dn}}, \quad (8)$$

where $\mathbf{y}:=(y_i)_{i=1}^n\in\mathsf{L}^2(\Theta)^n$, the kernel matrix is defined block-wise as $\mathbf{K}:=[\mathsf{K}(x_i,x_j)]_{i,j=1}^n\in\mathbb{R}^{dn\times dn}$; and vec and $\Phi_{(n)}$ are introduced in Section 1. We then derive a closed-form for fully observed output functions.

Proposition 4.2. ***(Ridge solution)*** *The minimum in Problem (8) is achieved by any $\alpha^*\in\mathbb{R}^{d\times n}$ verifying*

$$\left(\mathbf{K}(\Phi^\#\Phi)_{(n)}\mathbf{K}+n\lambda\mathbf{K}\right)\mathsf{vec}(\alpha^*):=\mathbf{K}\Phi^\#_{(n)}\mathbf{y}. \quad (9)$$

Such α^* exists. Moreover if $\mathbf{K}$ is full rank then $\left((\Phi^\#\Phi)_{(n)}\mathbf{K}+n\lambda\mathbf{I}\right)$ is invertible and α^* is such that

$$\mathsf{vec}(\alpha^*)=\left((\Phi^\#\Phi)_{(n)}\mathbf{K}+n\lambda\mathbf{I}\right)^{-1}\Phi^\#_{(n)}\mathbf{y}. \quad (10)$$

We define the ridge estimator as $h^\lambda_{\mathbf{z}}:=\sum_{j=1}^n\mathsf{K}_{x_j}\alpha^*_j$.

The proof is detailed in Section B.2 of the Supplement. $(\Phi^\#\Phi)_{(n)}$ is a block diagonal matrix with the Gram matrix $\Phi^\#\Phi$ of the dictionary repeated on its diagonal. Then if ϕ is orthonormal, Equation (10) simplifies to $\mathsf{vec}(\alpha^*)=(\mathbf{K}+n\lambda\mathbf{I})^{-1}\Phi^\#_{(n)}\mathbf{y}$.

Partially observed setting We can derive a solution for partially observed functions from Proposition 4.2. To that end, we remark that in Equation (10), the output functions only appear through the quantity $(\Phi_{(n)})^\#\mathbf{y}=\mathsf{vec}((\Phi^\#y_i)_{i=1}^n)\in\mathbb{R}^{dn}$ with for $i\in[n]$, $\Phi^\#y_i=\left(\langle y_i,\phi_l\rangle_{\mathsf{L}^2(\Theta)}\right)_{l=1}^d$. As a consequence, we propose to estimate those scalar products from the available observations and then to plug the obtained estimators into Equation (10).

Definition 4.1. (**Plug-in ridge estimator.**) For all $l \in [d]$ and $i \in [n]$, let $\widetilde{\nu}_{il} := \frac{1}{m_i}\sum_{p=1}^{m_i} \widetilde{y}_{ip}\phi_l(\theta_{ip})$ be the entries of $\widetilde{\nu} \in \mathbb{R}^{d\times n}$. Let $\widetilde{\alpha}^* \in \mathbb{R}^{d\times n}$ be such that $\mathsf{vec}(\widetilde{\alpha}^*) = \left((\Phi^{\#}\Phi)_{(n)}\mathbf{K} + n\lambda\mathbf{I}\right)^{-1}\mathsf{vec}(\widetilde{\nu})$. We then define the plug-in ridge estimator as $\widetilde{h}^{\lambda}_{\widetilde{\mathbf{z}}} := \sum_{j=1}^{n}\mathsf{K}_{x_j}\widetilde{\alpha}^*_j$.

We propose the following strategy to compute this estimator for a separable kernel $\mathsf{K} = k\mathsf{B}$.

Fast algorithm for plug-in ridge estimator. The matrix $\mathbf{K}$ can be rewritten as $\mathbf{K} = \mathsf{K}_{\mathcal{X}} \otimes \mathsf{B}$ with $\mathsf{K}_{\mathcal{X}} := (k(x_i, x_j))_{i,j=1}^{n} \in \mathbb{R}^{n\times n}$. Solving the linear system in Equation (10) has time complexity $\mathcal{O}(n^3d^3)$. However, $(\Phi_{(n)})^{\#}\Phi_{(n)} = \mathsf{I} \otimes (\Phi^{\#}\Phi)$, thus $(\Phi_{(n)})^{\#}\Phi_{(n)}\mathbf{K} = (\mathsf{I}\otimes(\Phi^{\#}\Phi))(\mathsf{K}_{\mathcal{X}}\otimes\mathsf{B})$. Using the mixed product property (Horn and Johnson, 1991, Lemma 4.2.10), we must solve $(\mathsf{K}_{\mathcal{X}}\otimes((\Phi^{\#}\Phi)\mathsf{B})+n\lambda\mathbf{I})\mathsf{vec}(\alpha) = \mathsf{vec}(\widetilde{\nu})$. Two classic resolution strategies can separate the contribution of n and d in the cubic term of the complexity. We can notice that the above linear system is equivalent to a discrete time Sylvester equation (Sima, 1996; Dinuzzo et al., 2011), which can be solved in $\mathcal{O}(n^3 + d^3 + n^2d + nd^2)$ time. Or if we wish to test many values of λ, using the Kronecker structure, we can deduce an eigendecomposition of $\mathsf{K}_{\mathcal{X}} \otimes ((\Phi^{\#}\Phi)\mathsf{B})$ from one of $\mathsf{K}_{\mathcal{X}}$ and one of $(\Phi^{\#}\Phi)\mathsf{B}$ (Horn and Johnson, 1991, Theorem 4.2.12) in $\mathcal{O}(n^3 + d^3)$ time. For a given $\alpha \in \mathbb{R}^{d\times n}$, the predicted function at a new input point $x \in \mathcal{X}$ is then given by $\Phi\mathsf{B}\alpha\mathsf{k}_{\mathbf{x}}(x)$ with $\mathsf{k}_{\mathbf{x}}(x) := (k(x, x_i))_{i=1}^{n}$.

Algorithm 1: Plug-in ridge estimator

```
Input: Sample $\widetilde{\mathbf{z}}$, matrices $\mathsf{B}$, $\Phi^{\#}\Phi$
Compute: kernel matrix $\mathsf{K}_{\mathcal{X}} = (k(x_i, x_j))_{i,j=1}^{n}$
Compute: estimates $\widetilde{\nu}$ of $(\langle y_i, \phi_d\rangle_{\mathsf{L}^2(\Theta)})_{i=1,l=1}^{n,d}$
Solve: $(\mathsf{K}_{\mathcal{X}} \otimes ((\Phi^{\#}\Phi)\mathsf{B}) + n\lambda\mathsf{I})\mathsf{vec}(\alpha) = \mathsf{vec}(\widetilde{\nu})$
Output: Representer coefficients $\alpha \in \mathbb{R}^{d\times n}$.
```

4.3 Iterative optimization

For other losses, since it is no longer possible to find a closed-form, we resort to iterative optimization.

Fully observed setting For K separable, using Proposition 4.1 and defining $\ell_{y_i}(y) := \ell(y_i, y)$; Problem (7) is rewritten as

$$\min_{\alpha\in\mathbb{R}^{d\times n}} \frac{1}{n}\sum_{i=1}^{n} \ell_{y_i}\left(\Phi\mathsf{B}\alpha\mathsf{k}_{\mathbf{x}}(x_i)\right) + \lambda\langle\mathsf{K}_{\mathcal{X}}, \alpha^{\mathrm{T}}\mathsf{B}\alpha\rangle_{\mathbb{R}^{n\times n}}.$$

The gradient of the objective is given by

$$\frac{1}{n}\mathsf{B}\Phi^{\#}_{\mathsf{mat},(n)}\mathsf{G}(\alpha)\mathsf{K}_{\mathcal{X}} + \lambda\mathsf{B}\alpha\mathsf{K}_{\mathcal{X}}, \qquad (11)$$

with $\mathsf{G}(\alpha) := (\nabla\ell_{y_i}(\Phi\mathsf{B}\alpha\mathsf{k}_{\mathbf{x}}(x_i)))_{i=1}^{n} \in \mathsf{L}^2(\Theta)^n$ and $\nabla\ell_{y_i} : \mathsf{L}^2(\Theta) \longmapsto \mathsf{L}^2(\Theta)$ the gradient of ℓ_{y_i}.

Partially observed setting. We notice that the entries of $\Phi^{\#}_{\mathsf{mat},(n)}\mathsf{G}(\alpha) \in \mathbb{R}^{d\times n}$ are the scalar products $\left(\langle\nabla\ell_{y_i}(\Phi\mathsf{B}\alpha\mathsf{k}_{\mathbf{x}}(x_i)), \phi_l\rangle_{\mathsf{L}^2(\Theta)}\right)_{l,i=1}^{d,n}$. For ℓ an integral loss (Equation (1)) based on a differentiable ground loss l, $\nabla\ell_{y_i} : y \longmapsto (\theta \longmapsto l(y_i(\theta), y(\theta)))$. We can thus estimate the columns $\Phi^{\#}\nabla\ell_{y_i}(\Phi\mathsf{B}\alpha\mathsf{k}_{\mathbf{x}}(x_i))$ as

$$\frac{1}{m_i}\sum_{p=1}^{m_i} l\left(y_i(\theta_{ip}), \phi(\theta_{ip})^{\mathrm{T}}\mathsf{B}\mathsf{k}_{\mathbf{x}}(x_i)\right)\phi(\theta_{ip}), \qquad (12)$$

where we have used the convention that for $\theta \in \Theta$, $\phi(\theta) := (\phi_l(\theta))_{l=1}^{d} \in \mathbb{R}^d$. The corresponding estimation of $\Phi^{\#}_{\mathsf{mat},(n)}\mathsf{G}(\alpha)$ can be plugged into Equation (11) to yield an estimated gradient.

Link with ridge estimator. In the partially observed setting, for the square loss, iterative optimization and the plug-in ridge estimator do not yield the same result. In fact, they correspond to two different ridge closed-forms (see Section F.2 of the Supplement). While the former is slower to compute than the latter it can be more robust (see Section 6.3).

5 THEORETICAL ANALYSIS

In this section we give two finite sample excess risk bounds. One for the ridge estimator in the fully observed setting and one for the plug-in ridge estimator in the partially observed setting. In the first case, we study the effect of the number of samples n, and in the second case that of both n and the number of observations per function m. We suppose that for all $i \in [n]$, $m_i = m$. We leave however a detailed analysis with respect to the size of the dictionary d (including approximation aspects) for future work. Our analysis is based on the framework of integral operators (Caponnetto and De Vito, 2007; Smale and Zhou, 2007) to which we give an introduction in the context of our problem in Section C of the Supplement.

5.1 Fully observed setting

In this section, we suppose that $\mathcal{X}$ is a separable metric space. We also need to relate the $\mathsf{L}^2(\Theta)$ norm of any $g \in \mathrm{Span}(\phi)$ to the square norm of its coefficients in the dictionary ϕ. To that end, a usual assumption is that it is a *Riesz family* (Casazza, 2000).

Definition 5.1. (Riesz family) $\phi \in \mathsf{L}^2(\Theta)^d$ is a Riesz family of $\mathsf{L}^2(\Theta)$ with constants (c_ϕ, C_ϕ) if it is linearly independent and for any $u \in \mathbb{R}^d$,

$$c_\phi \left\|u\right\|_{\mathbb{R}^d} \leq \left\|\sum_{l=1}^{d} u_l\phi_l\right\|_{\mathsf{L}^2(\Theta)} \leq C_\phi \left\|u\right\|_{\mathbb{R}^d}.$$

If in addition for all $l \in [d]$, $\|\phi_l\|_{\mathsf{L}^2(\Theta)} = 1$, it is a normed Riesz family.

Remark. Riesz families provide a natural generalization of orthonormal families as a normed Riesz family with $c_\phi = C_\phi = 1$ is orthonormal.

We make the following assumptions.

Assumption 5.1. *K is a vector-valued continuous kernel and there exists $\kappa > 0$ such that for $x \in \mathcal{X}$, $\|\mathsf{K}(x,x)\|_{\mathcal{L}(\mathbb{R}^d)} \leq \kappa$.*

Remark. We suppose that κ is independant from d. This is for instance the case if for $x \in \mathcal{X}$, $\mathsf{K}(x,x)$ is diagonal or block diagonal with bounded coefficients. More generally, we can rely on the fact that κ is bounded by the maximal $\|\cdot\|_1$-norm of the columns of $\mathsf{K}(x,x)$, which can easily be imposed to be be independent of d.

Assumption 5.2. *The dictionary ϕ is a normed Riesz family in $\mathsf{L}^2(\Theta)$ with upper constant C_ϕ.*

Remark. We do not use the lower constant c_ϕ.

Assumption 5.3. *There exist $h_{\mathcal{H}_\mathsf{K}} \in \mathcal{H}_\mathsf{K}$ such that $h_{\mathcal{H}_\mathsf{K}} = \inf_{h \in \mathcal{H}_\mathsf{K}} \mathcal{R}(\Phi \circ h)$.*

Remark. This is a standard assumption (Caponnetto and De Vito, 2007; Baldassarre et al., 2012; Li et al., 2019), it implies the existence of a ball of radius $R > 0$ in $\mathcal{H}_\mathsf{K}$ containing $h_{\mathcal{H}_\mathsf{K}}$, as a consequence $\|h_{\mathcal{H}_\mathsf{K}}\|_{\mathcal{H}_\mathsf{K}} \leq R$.

Assumption 5.4. *There exists $L \geq 0$ such that for all $\theta \in \Theta$, almost surely $|\mathsf{Y}(\theta)| \leq L$.*

We then have the following excess risk bound for the ridge estimator defined in Proposition 4.2. We prove it in Section E.1 of the Supplement.

Proposition 5.1. *Let $0 < \eta < 1$, taking $\lambda = \lambda_n^*(\eta/2) := 6\kappa C_\phi^2 \frac{\log(4/\eta)\sqrt{d}}{\sqrt{n}}$, with probability at least $1-\eta$*

$$\mathcal{R}(\Phi \circ h_{\mathbf{z}}^\lambda) - \mathcal{R}(\Phi \circ h_{\mathcal{H}_\mathsf{K}}) \leq 27 \left(\frac{B_0}{\sqrt{d}} + B_1 \sqrt{d} \right) \frac{\log(4/\eta)}{\sqrt{n}},$$

with $B_0 := (L + \sqrt{\kappa} C_\phi R)^2$ and $B_1 := \kappa C_\phi^2 R^2$.

This bound implies the consistency of the ridge estimator in the number of samples n.

5.2 Partially observed setting

To treat the partially observed setting, we need to make the following additional assumption.

Assumption 5.5. *There exists $M(d) \geq 0$ such that for all $\theta \in \Theta$ and for all $l \in [d]$, $|\phi_l(\theta)| \leq M(d)$.*

Remark. The dependence in d is specific to the family to which ϕ belongs; for wavelets we have $M(d) = 2^{r(\Theta,d)/2} \max_{\theta \in \Theta} |\psi(\theta)|$ with ψ the mother wavelet and $r(\Theta, d) \in \mathbb{N}$ the number of dilatations included in ϕ, whereas for a Fourier dictionary we have $M(d) = 1$.

We then have the following excess risk bound for the plug-in ridge estimator from Definition 4.1 which we prove in Section E.2 of the Supplement.

Proposition 5.2. *Let $0 < \eta < 1$, taking $\lambda = \lambda_n^*(\eta/3) := 6\kappa C_\phi^2 \frac{\log(6/\eta)\sqrt{d}}{\sqrt{n}}$, with probability at least $1-\eta$,*

$$\mathcal{R}(\Phi \circ \widetilde{h}_{\widetilde{\mathbf{z}}}^\lambda) - \mathcal{R}(\Phi \circ h_{\mathcal{H}_\mathsf{K}}) \leq \left(\frac{B_2(d)\sqrt{n}}{m^2} + \frac{B_3(d)}{m^{3/2}} + \frac{9C(d)^2}{2\sqrt{n}m} + \frac{B_4(d)}{\sqrt{n}} \right) \log(6/\eta),$$

with $C(d) := \frac{LM(d)}{C_\phi}$, $B_2(d) := 18\sqrt{d}\left(C(d) + \frac{R}{\sqrt{d}}\right)^2$, $B_3(d) := B_2(d) - 18\frac{R^2}{\sqrt{d}}$, $B_4(d) := \frac{81}{2}\left(\frac{B_0}{\sqrt{d}} + B_1\sqrt{d}\right)$ and B_0 and B_1 are defined as in Proposition 5.1.

We highlight that if $m \asymp \sqrt{n}$, then this bounds yields consistency for the plug-in ridge estimator.

6 NUMERICAL EXPERIMENTS

Section 6.3 is dedicated to the study of several aspects of robustness of KPL algorithms. Then we compare KPL with the nonlinear FOR methods presented in Section 6.1 on two datasets. In Section 6.4 we explore a biomedical imaging dataset with relatively small number of samples ($n = 100$) and partially observed functions, whereas in Section 6.5 we study a speech inversion dataset with relatively large number of samples ($n = 413$) and fully observed output functions.

We use the mean squared error (MSE) as metric. Given observed functions $((\theta_i, \widetilde{y}_i))_{i=1}^n$ and predicted ones $(\widehat{y}_i)_{i=1}^n \in \mathsf{L}^2(\Theta)$, we define it as MSE $:= \frac{1}{n}\sum_{i=1}^n \frac{1}{m_i}\sum_{p=1}^{m_i}(\widehat{y}_i(\theta_{ip}) - \widetilde{y}_{ip})^2$. The presented results are averaged either over 10 or 20 runs with different train/test splits. Full details of the experimental procedures are postponed to Section H of the Supplement.

6.1 Related works

We compare KPL to four existing nonlinear FOR methods that we present in this section. More detailed descriptions are given in Section G of the Supplement.

Functional kernel ridge regression (FKRR). Kadri et al. (2010, 2016) solve a functional KRR using function-valued-RKHSs. A representer theorem yields a closed-form solution computed by inverting an operator in $\mathcal{L}(\mathsf{L}^2(\Theta))^{n \times n}$. For a separable kernel $\mathsf{K}^{\text{fun}} = k\mathsf{L}$ with $\mathsf{L} \in \mathcal{L}(\mathsf{L}^2(\Theta))$, if an eigendecomposition of L is known in closed-form, an approximate solution is computed in $\mathcal{O}(n^3 + n^2 Jm)$ time, with J the number of eigenfunctions considered and m the size of the discretization grid. If not, a discretized problem is solved in $\mathcal{O}(n^3 + m^3 + n^2 m + nm^2)$ time.

Triple basis estimator (3BE). In (Oliva et al., 2015), the input and the output functions are represented by decomposition coefficients on two orthonormal families. The output coefficients are then regressed on the input ones using KRRs approximated with J RFFs in $\mathcal{O}(J^3+J^2d)$ time, with d the size of the output family. As 3BE is specific to function-to-function regression with scalar-valued inputs, we deal with vector-valued input functions (as in Section 6.5), directly through a kernel. We call this extension **one basis estimator (1BE)**; it is solved in $\mathcal{O}(n^3+n^2d)$ time. 1BE is in fact a particular case of the KPL plug-in ridge estimator with ϕ orthonormal and $\mathsf{K}=k\mathsf{I}$. However, our estimator offers the additional possibility to use non orthonormal dictionaries and to impose richer regularizations through kernels $\mathsf{K}=k\mathsf{B}$ with $\mathsf{B}\neq\mathsf{I}$. KPL can moreover can be used with a wide range of functional losses.

Kernel additive model (KAM). Reimherr et al. (2018) propose an additive function-to-function regression model using RKHSs. A representer theorem leads to a closed-from solution. Computations are performed in a truncated FPCA basis of size $J<n$. For a product of kernels, if the Kronecker structure is exploited (a possibility which is however not highlighted by the authors), the complexity is $\mathcal{O}(n^3+J^3+n^2J+nJ^2)$ time using a Sylvester solver. However, computing the matrix to form the linear system—matrix A in page 6 of (Reimherr et al., 2018)—is generally much more expensive; exploiting the product of kernels, n^2+J^2 double integrals must be computed which has time complexity $\mathcal{O}(n^2t^2+J^2m^2)$, with t the size of the input discretization grid. Those computations must moreover generally be repeated many times so as to tune the multiple kernel parameters.

Kernel Estimator (KE). Finally, an extension of the Nadaraya-Watson kernel estimator to Banach spaces is introduced and studied in (Ferraty et al., 2011).

6.2 Preliminary elements

Note on optimization. We compute the KPL plug-in ridge estimator as in Algorithm 1 with Sylvester solver. For iterative optimization, we use L-BFGS-B (Zhu et al., 1997); the estimates of partial second order informations improve convergence speed. For FKRR, of the two possible approaches from Section G of the Supplement, we use the faster Sylvester approach. For KAM we exploit the separability as well using a Sylvester solver.

Logcosh functional loss. As an example of a robust integral loss, for $\gamma>0$, we introduce $\ell_{\mathsf{lch}}^{(\gamma)}$. It is obtained by taking $l_{\mathsf{lch}}^{(\gamma)}:(a,b)\longmapsto {}^{1}\!/\!_{\gamma}\log(\cosh(\gamma(a-b))$ as ground loss in Equation (1). This ground loss behaves similarly to the Huber loss (Huber, 1964)—almost quadratically around 0 and almost linearly elsewhere. The parameter γ gives us control on its behaviour around 0, as it grows bigger, $l_{\mathsf{lch}}^{(\gamma)}$ tends to the absolute

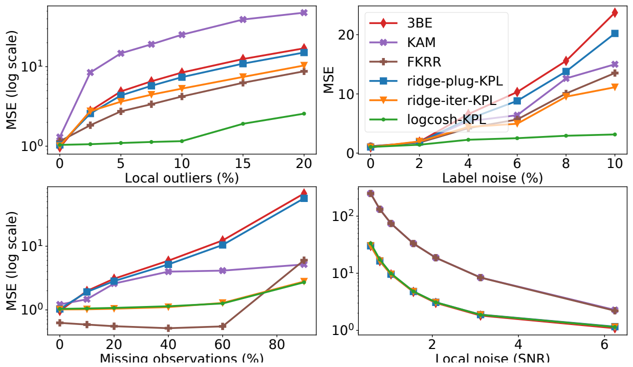

Figure 1: Several aspects of robustness.

loss (see Section H for examples). As opposed to our proposed integral loss $\ell_{\mathsf{lch}}^{(\gamma)}$, the extension of the Huber loss to $\mathsf{L}^2(\Theta)\times\mathsf{L}^2(\Theta)$ (e. g. Bauschke and Combettes, 2017, Example 13.7) is not differentiable everywhere.

6.3 Toy data

In this section, we take $\mathsf{K}=k\mathsf{I}$ with k a scalar-valued Gaussian kernel. We use a generated toy dataset: inputs are random mixtures of cubic B-splines (de Boor, 2001) centered at different locations and outputs are associated mixtures of Gaussian processes (drawn once and then fixed). The full generation procedure is described in Section H of the Supplement. We use $n_{\text{train}}=100$ samples for training and $n_{\text{test}}=100$ samples and use Fourier dictionaries for KPL and 3BE.

Corruption modalities. We study the effect of four types of corruptions of the training data: local outliers, label noise, missing observations and local noise. In the first case, observations from the output functions are replaced with random draws in their range. In the second case, some output functions are replaced with erroneous ones. In the third case we remove observations from the output functions uniformly at random. Finally, in the last one we add Gaussian noise to those observations. We then use the signal to noise ratio as x-axis; for a noise level σ and a sample $\widetilde{\mathbf{z}}$, we define it as SNR $:=\frac{1}{\sigma n}\sum_{i=1}^{n}\frac{1}{m_i}\sum_{p=1}^{m_i}|\widetilde{y}_{ip}|$.

Comments on the results. The evolution of the MSEs for several levels of corruption are displayed in Figure 1. For each type, at least one KPL algorithm is particularly robust which demonstrates the versatility of our framework. KPL can be combined with the functional logcosh loss to obtain a FOR algorithm that is robust to outliers (*logcosh-KPL*). Dealing with partially observed functions, KPL solved iteratively using estimated gradients works especially well (*ridge-iter-KPL*, *logcosh-KPL*). Finally all proposed KPL algorithms are robust to local noise.

Table 1: MSEs on the DTI dataset.

KE	0.231 ± 0.025
3BE	0.227 ± 0.017
KAM	0.222 ± 0.021
FKRR	0.215 ± 0.020
Ridge-KPL	0.211 ± 0.022
Logcosh-KPL	**0.209 ± 0.020**

6.4 Diffusion tensor imaging dataset (DTI)

Dataset. We now consider the DTI dataset.[1] It consists of 382 Fractional anisotropy (FA) profiles inferred from DTI scans along two tracts—corpus callosum (CCA) and right corticospinal (RCS). The scans were performed on 142 subjects; 100 multiple sclerosis (MS) patients and 42 healthy controls. MS is an auto-immune disease which causes the immune system to gradually destroy myelin, however the structure of this process is not well understood. Using the proxy of FA profiles, we propose to predict one tract (RCS) from the other (CCA). We consider only the first $n = 100$ scans of MS patients. Finally, we highlight that the functions are partially observed: significant parts of the FA profiles along the RCS tract are missing.

Experimental setting. We perform linear smoothing if necessary—for FKRR and KAM. We split the data as $n_{\text{train}} = 70$ and $n_{\text{test}} = 30$ and use wavelets dictionaries for 3BE and KPL. For KPL, we take a kernel of the form $\mathsf{K} = k\mathsf{D}$ with k a Gaussian kernel and D a diagonal matrix with diagonal decreasing with the corresponding wavelet scale. Finally, when using wavelets, we extend the signal symmetrically to avoid boundary effects. The MSEs are shown in Table 1.

Comments on the results. The studied methods perform almost equally well, with a slight advantage for ours. The combination of an efficient use of wavelets (well suited to non-smooth data) with the scale-dependant regularization induced by the kernel $\mathsf{K} = k\mathsf{D}$ may explain this.

6.5 Synthetic speech inversion dataset

Dataset. We consider a speech inversion problem: from an acoustic speech signal, we estimate the underlying vocal tract (VT) configuration that produced it (Richmond, 2002). Such information can improve performance in speech recognition systems or in speech synthesis. The dataset was introduced by Mitra et al. (2009); it is generated by a software synthesizing words

[1]This dataset was collected at Johns Hopkins University and the Kennedy-Krieger Institute and is freely available as a part of the *Refund* R package

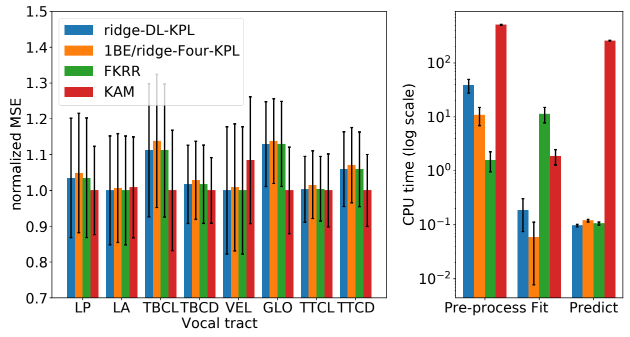

Figure 2: MSEs and CPU times on the speech dataset.

from an articulatory model. It consists of a corpus of $n = 413$ pronounced words with 8 distinct VT functions: lip aperture (LA), lip protrusion (LP), tongue tip constriction degree (TTCD), tongue tip constriction location (TTCL), tongue body constriction degree (TBCD), tongue body constriction location (TBCL), Velum (VEL) and Glottis (GLO).

Experimental setting. To match words of varying lengths, we extend symmetrically both the input sounds and the VT functions matching the longest word. We represent the sounds using 13 mel-frequency cepstral coefficients (MFCC), the input data thus consist of vector-valued functions. We split the data as $n_{\text{train}} = 300$ and $n_{\text{test}} = 113$. We normalize the output functions so that they take their values in $[-1, 1]$. To deal with the vector-valued functional inputs, we use an integral of Gaussian kernels on the standardized MFCCs (KPL, FKRR, 1BE/KPL). For KAM we take Laplace kernels for both input and output locations, and use a Gaussian kernel defined on $\mathbb{R}^{13}$ to compare the evaluations of the standardized MFCCs (see Section H of the Supplement for details on the employed kernels).

The MSEs for the 8 VTs (left panel) as well as an analysis of the computation times (right panel) are displayed in Figure 2. *Pre-process* entails all pre-processing operations (e. g. computing the the kernel matrices, learning the dictionary, computing the gram matrix of ϕ), *fit* measures the fitting time per se (solving the relevant linear system) and *predict* measures the prediction time on the test set (for all methods, it entails computing new kernel matrices). *ridge-DL-KPL* is the KPL ridge estimator with ϕ learnt by solving Problem (6) with $\mathcal{C}$ and $\Omega_{\mathbb{R}^d}$ as introduced in Section 3.2. *1BE/ridge-Four-KPL* corresponds to 1BE (or equivalently KPL with $\mathsf{K} = k\mathsf{I}$) with ϕ a Fourier family. To give an order of idea, we use 30 atoms for the learnt dictionaries while the numbers of atoms selected by cross-validation for the Fourier ones are around 100. We do not include KE in the figure as it performed poorly on this dataset.

Comments on the results. For 4 out of 8 VTs (LP, LA, TBCD, TTCL), the performances of the methods are comparable, with KAM being slightly more precise. On the remaining 4 VTs, ridge-DL-KPL, 1BE/ridge-Four-KPL and FKRR beat KAM on one (VEL) and are beaten by KAM on the 3 other (TBCL, GLO, TTCD). This could be explained by the fact that KAM predicts locally the functions while the other three methods have more of a global approach. Depending on the properties of the functions and the nature of the dependency between input and output functions, one or the other could be more favorable. However KAM's main weakness is its computational cost for pre-processing and prediction, which makes it unpractical to use on medium-sized datasets and impossible to use on larger ones. The particularily time-consuming operation in question is the computation of an analogous to the kernel matrix (see Section 6.1). The three other methods display very close MSEs, with 1BE/ridge-Four-KPL being a bit less precise than the two others. Ridge-DL-KPL and FKRR perform equally well. However for the former the main computational burden comes from a pre-processing operation (learning the dictionary), which is performed only once per dataset (or once per fold in a cross-validation); whereas for the latter it comes from fitting the method, which must be done many times so as to tune its parameters. Moreover for Ridge-DL-KPL, once a number of atoms yielding a good approximation has been found and the dictionary has been learnt, no further tuning must be performed for the outputs, whereas for FKRR an output kernel must be chosen.

7 CONCLUSION

We introduced PL, a general dictionary-based framework to address FOR. It can be used with a wide class of functional losses and non orthonormal dictionaries. Through an extensive study in the context of vv-RKHSs, we illustrated some aspects of its versatility and demonstrated that the approach is efficient and can be backed theoretically in some cases. For future research, PL could be instantiated using other hypothesis classes than vv-RKHS and the possibilities offered by dictionary learning could be investigated further.

Acknowledgements

The authors thank Zoltán Szabó for his insightful feedbacks. This work was supported by the Télécom Paris research chair on Data Science and Artificial Intelligence for Digitalized Industry and Services (DSAIDIS).

References

A. M. Alvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: a review. *Foundations and Trends in Machine Learning*, 4(3):195–266, 2012.

L. Baldassarre, L. Rosasco, and A. Barla. Multi-output learning via spectral filtering. *Machine Learning*, 87: 259–301, 2012.

H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2017.

T. T. Cai and M. Yuan. Optimal estimation of the mean function based on discretely sampled functional data: Phase transition. *The Annals of Statistics*, 39: 2330–2355, 2011.

A. Caponnetto and E. De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, pages 331–368, 2007.

P. G. Casazza. The art of frame theory. *Taiwanese journal of mathematics*, 4:129–201, 2000.

C. de Boor. *A practical guide to Splines - Revised Edition*. Springer, 2001.

R. A. DeVore, B. Jawerth, and V. Popov. Compression of wavelet decompositions. *American Journal of Mathematics*, 114(4):737–785, 1992.

F. Dinuzzo, C. S. Ong, P. Gehler, and G. Pillonetto. Learning output kernels with block coordinate descent. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 49–56, 2011.

B. Dumitrescu and P. Irofti. *Dictionary Learning, Algorithms and Applications*. Springer, 2018.

F. Ferraty and P. Vieu. *Nonparametric functional data analysis*. Springer, 2006.

F. Ferraty, A. Laksaci, A. Tadj, and P. Vieu. Kernel regression with functional response. *Electron. J. Statist.*, 5:159–171, 2011.

R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991.

P. J. Huber. Robust estimation of a location parameter. *Annals of Statistics*, 53:73–101, 1964.

H. Kadri, E. Duflos, P. Preux, S. Canu, and M. Davy. Nonlinear functional regression: a functional RKHS approach. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 9, pages 374–380, 2010.

H. Kadri, E. Duflos, P. Preux, S. Canu, A. Rakotomamonjy, and J. Audiffren. Operator-valued kernels for learning from functional response data. *Journal of Machine Learning Research*, 17:1–54, 2016.

P. Kokoska and M. Reimherr. *Introduction to Functional Data Analysis*. CRC Press, 2017.

H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems (NIPS) 19*, pages 801–808, 2007.

Y. Li and T. Hsing. Uniform convergence rates for nonparametric regression and principal component analysis in functional/longitudinal data. *The Annals of Statistics*, 38:3321–3351, 2010.

Z. Li, J.-F. Ton, D. Oglic, and D. Sejdinovic. Towards a unified analysis of random Fourier features. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97, pages 3905–3914, 2019.

H. Lian. Nonlinear functional models for functional responses in reproducing kernel Hilbert spaces. *Canadian Journal of Statistics*, pages 597–606, 2007.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages 689–696, 2009.

S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 2008.

C. A. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17(1):177–204, 2005.

V. Mitra, Y. Ozbek, H. Nam, X. Zhou, and C. Y. Espy-Wilson. From acoustics to vocal tract time functions. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4497–4500, 2009.

J. S. Morris. Functional regression. *The Annual Review of Statistics and Its Application*, 2:321–359, 2015.

J. Oliva, W. Neiswanger, B. Poczos, E. Xing, H. Trac, S. Ho, and J. Schneider. Fast function to function regression. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 38, pages 717–725, 2015.

P. Oswald. On the degree of nonlinear spline approximation in Besov-Sobolev spaces. *Journal of approximation theory*, 61(2):131–157, 1990.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems (NIPS) 20*, pages 1177–1184. Curran Associates, Inc., 2008a.

A. Rahimi and B. Recht. Uniform approximation of functions with random bases. In *46th Annual Allerton Conference on Communication, Control, and Computing*, pages 555–561, 2008b.

J. O. Ramsay and B. W. Silverman. *Functional data analysis*. Springer, 2005.

M. Reimherr, B. Sriperumbudur, and B. Taoufik. Optimal prediction for additive function on function regression. *Electronic Journal of Statistics*, 12:4571–4601, 2018.

K. Richmond. *Estimating Articulatory Parameters from the Acoustic Speech Signal*. PhD thesis, The Center for Speech Technology Research, Edinburgh University, 2002.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

V. Sima. *Algorithms for Linear-Quadratic Optimization*. Chapman and Hall/CRC, 1996.

S. Smale and D.-X. Zhou. Learning theory estimates via integral operators and their approximations. *Constructive Approximation*, pages 153–172, 2007.

C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transations on Mathematical Software*, 1997.