

FunCast: a forecasting model for functional data using covariates

Selman Sezgin^{1,2*}, Julien Jacques¹, Kahina Mokrani², Sylvain Allio²

^{1*}ERIC, Université Lumière Lyon 2, Université Claude Bernard Lyon 1, Lyon, 69007, France.

²Orange Innovation, Orange Research, Belfort, 90000, France.

*Corresponding author(s). E-mail(s): selman.sezgin@univ-lyon2.fr;

Contributing authors: Julien.Jacques@univ-lyon2.fr; kahina.mokrani@orange.com; sylvain.allio@orange.com;

Abstract

Functional Data Analysis (FDA) is a mature area of statistics in which data are represented as smooth functions rather than finite-dimensional vectors, with concrete applications in fields such as medicine and finance. The functional approach allows for more accurate modeling of problems involving dynamic variables that interact over time. Despite significant advances in FDA, little work has been devoted to the task of forecasting the future of a function over a given horizon based on the past of functional covariates. In this work, we develop a functional forecasting model, FunCast, designed to address this problem. FunCast expands the future of the target function in a fixed functional basis and predicts its expansion coefficients using its past and the past of the covariates within the scalar-on-function regression framework. By also expanding the covariates and model parameters into functional bases, we derive an explicit closed-form solution for the optimal parameters of FunCast. A theoretical analysis establishes a convergence rate for FunCast under well-conditioning assumptions. We also propose a practical method to ease hyperparameter tuning by reducing the effective number of hyperparameters. Finally, the performance of FunCast is demonstrated on both synthetic functional data and the real-world Cycling dataset.

Keywords: Functional Data Analysis, Functional Regression, Stochastic Process, Parametric Modeling, Basis Function Expansion.

1 Introduction

Functional data analysis (FDA) is an active branch of statistics that deals with data in the form of smooth curves rather than finite-dimensional vectors. Medicine (Yi et al. 2022), biology (Baladandayuthapani et al. 2008), economics and finance (Das et al. 2019), demography (Hyndman and Booth 2008) and transportation (Albarrán-Lozano et al. 2017; Chiou

et al. 2019) are examples of fields where FDA has been successfully applied. For a comprehensive review of FDA applications, see (Ullah and Finch 2013). Research topics in FDA are numerous and include functional regression (Tamo Tchomgui et al. 2024), smoothing (Amir et al. 2024), classification (Rossi and Villa 2006), and clustering (Jacques and Preda 2014). The book (Ramsay and Silverman 2005) is a comprehensive and progressive exposition of FDA. In this work, we study a

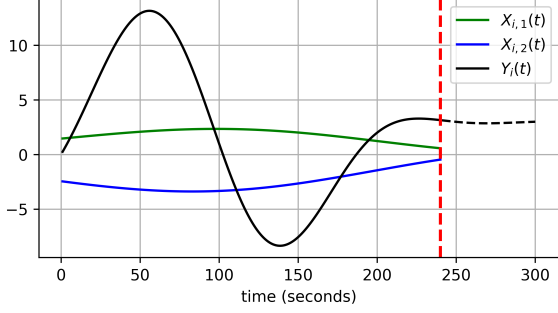


Fig. 1 Illustration of the forecasting problem. The i -th realization of the function of interest (in black) and covariates (in blue and green) is shown. In this figure, the functions are observed up to time 240, and the function of interest is predicted between times 240 and 300.

functional prediction problem in longitudinal data sets where several curves of a set of functions are observed (Giordani et al. 2020). More specifically, we consider a set of trajectories of a function of interest Y and the joint trajectories of functional covariates. We aim to find a functional model that, given the initial part of Y over the *past segment* $[0, T]$ and that of the covariates, forecasts the continuation of Y over the *future segment* $[T, T + H]$, with $T, H > 0$. For model training, we consider the n -sample of functional data:

$$\{Y_i(t), X_{i,1}(t), \dots, X_{i,p}(t)\} \quad (1)$$

for $t \in [0, T + H]$ and $i \in [1 : n]$, where $[1 : n]$ denotes the set of integers w that satisfy $1 \leq w \leq n$. The i -th realization contains $Y_i(t)$ and covariates $X_{i,1}(t), \dots, X_{i,p}(t)$. Model training consists in learning a statistical relationship between the functions $Y_i(t), X_{i,1}(t), \dots, X_{i,p}(t)$ on $[0, T]$ and $Y_i(t)$ on $[T, T + H]$. After training, given a new realization $Y_{n+1}(t), X_{n+1,1}(t), \dots, X_{n+1,p}(t)$ where the functions are only observed up to time T , the model forecasts the continuation of $Y_{n+1}(t)$ between T and $T + H$. The forecasting problem studied in this paper is illustrated in Figure 1. In practice, functions are not observed continuously over time but on a discrete set of time points, with m_1 measures on $[0, T]$ and m_2 measures on $[T, T + H]$. In the following, to simplify the presentation of the model, m_1 and m_2 are assumed to be constant with respect to the realization index i . However, the present work extends straightforwardly to the case where this assumption does

not hold. From this discrete perspective, classical learning models such as multivariate regression or time series are thus potential solutions. However, in addition to ignoring the underlying smooth functions, traditional multivariate regression with vector-valued outputs does not model temporal dependencies between the variables of the output vector, and the time series framework treats each realization independently of the others. In FDA, the curve prediction problem is typically addressed within the framework of functional time series (Damon and Guillas 2005; Jiao et al. 2023; Gao and Shang 2017; Shah et al. 2022). However, this framework does not solve our problem because it typically assumes an autoregressive dependency between two consecutive observations whereas the realization in (1) is independent. This problem does not fit into a function-on-function regression framework, since the domain of $Y_i(t)$, namely $[T, T + H]$, differs from that of the functional predictors, which is $[0, T]$. To our knowledge, the model in (Goldberg et al. 2010, 2014) is the closest to our work but our problem is more general because the authors do not consider the presence of covariates. In this work, we propose a functional prediction model, named FunCast, which predicts the future values of Y based on the past observations of Y and covariates. FunCast expands the future of Y into a fixed functional basis and predicts the expansion coefficients, which are then used to reconstruct a smooth continuation of Y . The estimation of these coefficients relies on the scalar-on-function regression framework (Reiss et al. 2017), where each coefficient is independently estimated from the past of Y and the covariates. This approach leads to the parametric functional model FunCast, where a set of functional parameters is optimized to fit the empirical data. To reduce the dimensionality of the problem, these parameters are expanded in functional bases, yielding an equivalent finite-dimensional optimization problem. The coefficients of the parameters are then optimized to minimize the model prediction error on the data. By also projecting the covariates onto functional bases, the optimization reduces to an ordinary least squares problem, for which an exact and explicit solution is provided. As a result, the optimal parameters of FunCast are expressed explicitly in terms of the observed data. In a theoretical analysis, we further show that,

in terms of prediction error, FunCast achieves a convergence rate of $\mathcal{O}\left(\frac{TH}{nm_2}\right)$ for well-conditioned problems. A discussion on the tuning of FunCast hyperparameters is then conducted. We propose a transformation method to reduce the number of hyperparameters to be selected without compromising the predictive performance of FunCast. Finally, experiments on both synthetic and real-world functional data are conducted to evaluate the performance of FunCast and compare it to competitors. This paper is organized as follows: Section 2 introduces the assumptions and notations of the FunCast model; Section 3 presents the optimization of FunCast functional parameters; Section 4 provides a theoretical analysis of FunCast convergence rate; Section 5 is implementation oriented and discusses the choice of FunCast hyperparameters; Section 6 presents numerical simulations on synthetic functional data; Section 7 applies FunCast to real-world data and Section 8 concludes with a summary and perspectives.

2 Hypotheses of the functional model

2.1 Model definition

Let $Y_i^{(1)}(t)$ denote the restriction of $Y_i(t)$ to the segment $[0, T]$ and $Y_i^{(2)}(t)$ the restriction to $[T, T + H]$. Using the same notation, the restrictions $X_{i,\ell}^{(1)}(t)$ for the covariates are introduced. Thus, the model proposed in this work predicts $Y_i^{(2)}(t)$ based on $Y_i^{(1)}(t)$ and $X_{i,1}^{(1)}(t), \dots, X_{i,p}^{(1)}(t)$. To simplify notation, we set $X_{i,0}^{(1)}(t) = Y_i^{(1)}(t)$. Consequently, the past of Y is treated by the model as one of the covariates used to predict the future of Y . Throughout this work, we assume that $X_{i,0}(t), \dots, X_{i,p}^{(1)}(t)$ belong to the Hilbert space $H_1 = L^2([0, T])$ of square-integrable functions on $[0, T]$. The inner product and norm in H_1 are defined as $\langle f; g \rangle_{H_1} = \int_0^T f(t)g(t)dt$ and $\|f\|_{H_1} = \left(\int_0^T f(t)^2 dt\right)^{1/2}$, respectively. Similarly, we assume that $Y_i^{(2)}(t)$ belongs to $H_2 = L^2([T, T + H])$, with the inner product and norm denoted by $\langle \cdot; \cdot \rangle_{H_2}$ and $\|\cdot\|_{H_2}$. The functional forecast model proposed in this work, referred to as FunCast, takes the following form: For all $t \in$

$[T, T + H]$,

$$\hat{Y}_i^{(2)}(t; \alpha, \beta) = \alpha(t) + \sum_{k=1}^K \left(\sum_{\ell=0}^p \langle X_{i,\ell}^{(1)}; \beta_{k,\ell} \rangle_{H_1} \right) \psi_k(t) \quad (2)$$

where $\alpha(t)$ is a real-valued function defined on $[T, T + H]$, $\psi = (\psi_1, \dots, \psi_K)$ is a basis of functions in H_2 , and $\beta = (\beta_{k,\ell} \mid k \in [1 : K], \ell \in [0 : p])$ is a collection of functional parameters in H_1 . The parameter space to which β belongs is denoted by $\mathcal{P}$. Consequently, $\mathcal{P}$ can be identified with $H_1^{K(p+1)}$. The function $\alpha(t)$ models systematic trends in the future of Y . In this work, we focus on a less general version of FunCast by setting $\alpha(t) = 0$. This assumption simplifies the model and is empirically justified, as setting $\alpha(t)$ to zero yielded satisfactory performance in our experiments. That said, extending this work to a non-zero and optimized $\alpha(t)$ remains a research direction. FunCast is then parameterized solely by β and is simply denoted by $\hat{Y}_i^{(2)}(t; \beta)$. Before detailing the optimization of the parameters β based on functional data, the following section introduces the key idea that motivates the formulation of FunCast, using concepts from FDA.

2.2 Model construction

A classical assumption in FDA is that the underlying functions belong to the Hilbert space of square-integrable functions which is an infinite-dimensional space. To reduce the dimensionality of the problem and facilitate statistical processing, it is common to project the functional data into functional bases (Ramsay and Silverman 2005). The basis functions are smooth and sufficiently elementary and general to approximate complex functions through mixtures. Examples of standard functional bases include the Fourier basis for periodic phenomena (Happ and Greven 2018), wavelets (Antoniadis et al. 2013), and spline bases which are widely used for non-periodic functions (Aguilera and Aguilera-Morillo 2013). The information contained in a function with a continuous domain is encoded by a limited number of scalars which are the expansion coefficients of the function with respect to a basis. FunCast leverages the idea that forecasting the future of Y can be reduced to predicting a finite set of its basis expansion coefficients. This approach has

two main advantages: first, it transforms the initial functional prediction problem into a simpler multivariate prediction problem, and second, it ensures that the forecast is smooth because it is, by definition, expressed as a linear combination of basis functions. Consider the approximation of $Y_i^{(2)}(t)$ in a basis ψ :

$$\hat{Y}_i^{(2)}(t) = \sum_{k=1}^K a_{i,k} \psi_k(t). \quad (3)$$

For each $k = 1, \dots, K$, in FunCast it is assumed that $a_{i,k}$ is estimated through a scalar-on-function regression based on the $p+1$ functional predictors $X_{i,0}^{(1)}(t), \dots, X_{i,p}^{(1)}(t)$ using the sample:

$$\left\{ X_{i,0}^{(1)}(t), \dots, X_{i,p}^{(1)}(t), a_{i,k} \right\}$$

for $t \in [0, T]$ and $i \in [1 : n]$. The standard linear functional model (Reiss et al. 2017) for this task is:

$$a_{i,k} = \sum_{\ell=0}^p \int_{t=0}^T X_{i,\ell}^{(1)}(t) \beta_{k,\ell}(t) dt, \quad (4)$$

where $\beta_{k,\ell}(t)$ are functional parameters. The resulting K similar scalar-on-function regressions are assumed to be independent, leading to $K(p+1)$ functional parameters. By substituting (4) into (3), the formulation of $\hat{Y}_i^{(2)}(t)$ in FunCast is recovered.

3 Model optimization

The optimization of FunCast consists in determining the functional parameters in model (2) that best fit the data using the principle of empirical risk minimization (Bach 2024).

3.1 Risk functions

In this section, we first introduce a so-called *functional* risk, defined as the mean squared error over the n realizations using the norm in H_2 . More precisely, the functional risk of FunCast for parameters $\beta \in \mathcal{P}$ is defined as:

$$\mathcal{R}_f(\beta) = \frac{1}{n} \sum_{i=1}^n \left\| Y_i^{(2)} - \hat{Y}_i^{(2)}(\cdot; \beta) \right\|_{H_2}^2. \quad (5)$$

Although the risk $\mathcal{R}_f$ naturally generalizes the traditional mean squared error in multivariate problems, it assumes that functions are continuously observed over time. Since functional data are typically observed discretely, the theoretical risk $\mathcal{R}_f$ is approximated by an *empirical risk* $\mathcal{R}_{\text{emp}}$ computed on finite grids. Let $0 < t_1 < \dots < t_{m_1} \leq T$ denote the measurement times of the covariates on $[0, T]$, and $T < t_{m_1+1} < \dots < t_{m_1+m_2} \leq T+H$ the measurement times of the future of Y on $[T, T+H]$. A function observed n times over $[0, T+H]$ can thus be represented longitudinally by a grid with n rows and $m_1 + m_2$ columns. The empirical risk of FunCast for parameters $\beta \in \mathcal{P}$ is defined as:

$$\mathcal{R}_{\text{emp}}(\beta) = \frac{1}{n} \sum_{i=1}^n \mathcal{R}_{\text{emp}}^{(i)}(\beta) \quad (6)$$

where

$$\mathcal{R}_{\text{emp}}^{(i)}(\beta) = \frac{1}{m_2} \sum_{j=m_1+1}^{m_1+m_2} \left(Y_i^{(2)}(t_j) - \hat{Y}_i^{(2)}(t_j; \beta) \right)^2$$

Unlike $\mathcal{R}_f$, the risk $\mathcal{R}_{\text{emp}}$ is strictly empirical as it relies on a finite set of observation points, which enables the practical optimization of β . However, $\mathcal{R}_{\text{emp}}$ does not fully capture the functional nature of the considered problem. Finally, note that under certain assumptions on the measurement times t_j , the risk $\mathcal{R}_{\text{emp}}$ is an approximation of $\mathcal{R}_f$. From a probabilistic perspective, if the t_j are uniformly distributed, then $\mathcal{R}_{\text{emp}}$ is as a Monte Carlo approximation of the functional integrals in $\mathcal{R}_f$. From an analytical perspective, if the t_j are equidistant, then $\mathcal{R}_{\text{emp}}$ is a Riemann sum approximation of $\mathcal{R}_f$. Finally, the optimal parameters $\hat{\beta}$ of FunCast are defined as the minimizers of the empirical risk:

$$\hat{\beta} = \arg \min_{\beta \in \mathcal{P}} \mathcal{R}_{\text{emp}}(\beta). \quad (7)$$

The functions in $\hat{\beta}$ are denoted by $\hat{\beta}_{k,\ell}$.

3.2 Basis expansion

Computing $\hat{\beta}$ remains computationally challenging in practice, as $K(p+1)$ parameters in H_1 need to be estimated. As discussed in Section 2.2, a

common technique in FDA to construct a finite-dimensional problem is to perform basis function expansions (Ramsay and Silverman 2005). Therefore, the following assumptions are made.

Assumption 1 Let $q_0, \dots, q_p$ be positive integers. For any $\beta \in \mathcal{P}$, the parameter $\beta_{k,\ell}(t)$ is expanded in a known basis $\mathbf{B}_\ell = (B_{\ell,1}, \dots, B_{\ell,q_\ell})$ such that

$$\beta_{k,\ell}(t) = \sum_{r=1}^{q_\ell} b_{k,\ell,r} B_{\ell,r}(t) = \mathbf{b}_{k,\ell}^\top \mathbf{B}_\ell(t),$$

using the vector notation $\mathbf{b}_{k,\ell} = [b_{k,\ell,1}, \dots, b_{k,\ell,q_\ell}]^\top \in \mathbb{R}^{q_\ell}$ and $\mathbf{B}_\ell(t) = [B_{\ell,1}(t), \dots, B_{\ell,q_\ell}(t)]^\top \in \mathbb{R}^{q_\ell}$.

Assumption 2 Let $h_0, \dots, h_p$ be positive integers. The covariate $X_{i,\ell}^{(1)}(t)$ is expanded in a known basis $\boldsymbol{\theta}_\ell = (\theta_{\ell,1}, \dots, \theta_{\ell,h_\ell})$ in H_1 such that

$$X_{i,\ell}^{(1)}(t) = \sum_{j=1}^{h_\ell} c_{i,\ell,j} \theta_{\ell,j}(t) = \mathbf{c}_{i,\ell}^\top \boldsymbol{\theta}_\ell(t),$$

using the vector notation $\mathbf{c}_{i,\ell} = [c_{i,\ell,1}, \dots, c_{i,\ell,h_\ell}]^\top \in \mathbb{R}^{h_\ell}$ and $\boldsymbol{\theta}_\ell(t) = [\theta_{\ell,1}(t), \dots, \theta_{\ell,h_\ell}(t)]^\top \in \mathbb{R}^{h_\ell}$.

The bases $\boldsymbol{\psi}$, $\boldsymbol{\theta}_\ell$, and $\mathbf{B}_\ell$, for $\ell \in [0 : p]$, are chosen by the user based on the nature of the data and can be treated as hyperparameters of the model. We denote $q = \sum_{\ell=0}^p q_\ell$, such that Kq is the total number of basis functions to expand any parameters β in $\mathcal{P}$. We also introduce an expansion coefficient extraction operator $\kappa : \mathcal{P} \rightarrow \mathbb{R}^{Kq}$, which maps each functional parameters $\beta \in \mathcal{P}$ to the vector $\mathbf{b} = \kappa(\beta)$ defined as follows: $\mathbf{b} = [\mathbf{b}_0^\top, \dots, \mathbf{b}_p^\top]^\top$ with $\mathbf{b}_\ell = [\mathbf{b}_{1,\ell}^\top, \dots, \mathbf{b}_{K,\ell}^\top]^\top \in \mathbb{R}^{Kq_\ell}$. This operator is bijective. The inverse operator κ^{-1} generates functional parameters $\beta \in \mathcal{P}$ from any expansion coefficients $\mathbf{b} \in \mathbb{R}^{Kq}$.

3.3 From functional to multivariate setting

In this section, we show that under Assumptions 1 and 2, the model (2) admits a vector representation, and that the empirical risk also has a matrix form. This allows us to derive an explicit formulation of the optimal parameters $\hat{\beta}$ based on the problem data.

Theorem 1 Let $\beta \in \mathcal{P}$ and $\mathbf{b} = \kappa(\beta)$. Then, the model (2) takes the following form: $\hat{Y}_i(t; \beta) = \mathbf{x}_i(t)^\top \mathbf{b}$, where $\mathbf{x}_i(t)$ is a time-dependent vector in $\mathbb{R}^{Kq}$ defined in Equation (8).

Proof of Proposition 1. In the following, $\mathcal{M}_{d,d'}(\mathbb{R})$ denotes the space of real matrices with d rows and d' columns. Defining the inner product matrix

$$\mathbf{J}_{\theta,B,\ell} = \int_0^T \boldsymbol{\theta}_\ell(u)^\top \mathbf{B}_\ell(u) du \in \mathcal{M}_{h_\ell, q_\ell}(\mathbb{R}),$$

and using the expansions of $X_{i,\ell}^{(1)}(t)$ and $\beta_{k,\ell}(t)$ from Assumptions 2 and 1, we have

$$\begin{aligned} \hat{Y}_i^{(2)}(t; \beta) &= \sum_{k=1}^K \sum_{\ell=0}^p \mathbf{c}_{i,\ell}^\top \mathbf{J}_{\theta,B,\ell} \mathbf{b}_{k,\ell} \psi_k(t) \\ &= \sum_{\ell=0}^p \mathbf{c}_{i,\ell}^\top \mathbf{J}_{\theta,B,\ell} \left(\sum_{k=1}^K \psi_k(t) \mathbf{b}_{k,\ell} \right). \end{aligned}$$

Let $\mathbf{b}_\ell \in \mathbb{R}^{Kq_\ell}$ be the column-wise concatenation of the vectors $\mathbf{b}_{\ell,1}, \dots, \mathbf{b}_{\ell,K}$, and define the matrix $\mathbf{M}_{\psi,\ell}(t) = [\psi_1(t) \mathbf{I}_{q_\ell} \cdots \psi_K(t) \mathbf{I}_{q_\ell}] \in \mathcal{M}_{q_\ell, Kq_\ell}(\mathbb{R})$, where $\mathbf{I}_{q_\ell}$ denotes the identity matrix in $\mathcal{M}_{q_\ell, q_\ell}(\mathbb{R})$. Then, we obtain $\hat{Y}_i^{(2)}(t; \beta) = \sum_{\ell=0}^p \mathbf{c}_{i,\ell}^\top \mathbf{J}_{\theta,B,\ell} \mathbf{M}_{\psi,\ell}(t) \mathbf{b}_\ell$. Let $h = \sum_{\ell=0}^p h_\ell$, $\mathbf{c}_i \in \mathbb{R}^h$ be the column-wise concatenation of the vectors $\mathbf{c}_{i,0}, \dots, \mathbf{c}_{i,p}$, and define the following block-diagonal matrix: $\mathbf{M}_{\theta,B,\psi}(t) = \text{Diag}(\mathbf{J}_{\theta,B,\ell} \mathbf{M}_{\psi,\ell}(t))_{\ell=0}^p \in \mathcal{M}_{h, Kq}(\mathbb{R})$. Then, using $\mathbf{b} = \kappa(\beta)$, we obtain $\hat{Y}_i^{(2)}(t; \beta) = \mathbf{c}_i^\top \mathbf{M}_{\theta,B,\psi}(t) \mathbf{b}$. And finally, defining

$$\mathbf{x}_i(t) = \mathbf{M}_{\theta,B,\psi}(t)^\top \mathbf{c}_i, \quad (8)$$

the proposition is proved. $\square$

Let $\mathbf{y} = [\mathbf{y}_1^\top, \dots, \mathbf{y}_n^\top]^\top \in \mathbb{R}^{nm_2}$ with $\mathbf{y}_i = [Y_i(t_{m_1+1}), \dots, Y_i(t_{m_1+m_2})]^\top \in \mathbb{R}^{m_2}$, and $\mathbf{X} = [\mathbf{X}_1^\top \cdots \mathbf{X}_n^\top]^\top \in \mathcal{M}_{nm_2, Kq}(\mathbb{R})$ with

$$\mathbf{X}_i = [\mathbf{x}_i(t_{m_1+1}) \cdots \mathbf{x}_i(t_{m_1+m_2})]^\top \in \mathcal{M}_{m_2, Kq}(\mathbb{R}).$$

A direct implication of Proposition 1 is that the empirical risk defined in Equation (6) is given by $\mathcal{R}_{\text{emp}}(\beta) = \frac{1}{nm_2} \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2^2$ where $\mathbf{b} = \kappa(\beta)$ and $\|\cdot\|_2$ denotes the Euclidean norm. Consequently, the optimal parameters $\hat{\beta}$ admit an explicit formulation in terms of their expansion coefficients.

Theorem 2 The expansion coefficients $\hat{\mathbf{b}} = \kappa(\hat{\beta})$ are given by

$$\hat{\mathbf{b}} = \begin{cases} (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} & \text{if } \mathbf{X}^\top \mathbf{X} \text{ is invertible,} \\ (\mathbf{X}^\top \mathbf{X})^+ \mathbf{X}^\top \mathbf{y} & \text{otherwise,} \end{cases}$$

where $(\mathbf{X}^\top \mathbf{X})^+$ is the Moore-Penrose pseudo-inverse of $\mathbf{X}^\top \mathbf{X}$.

Proof of Proposition 2 It suffices to show that $\hat{\mathbf{b}} = \arg \min_{\mathbb{R}^{Kq}} \frac{1}{nm_2} \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2^2$. Assume, for contradiction, that there exists $\mathbf{b}' \in \mathbb{R}^{Kq}$ such that $\|\mathbf{y} - \mathbf{X}\mathbf{b}'\|_2^2 < \|\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\|_2^2$. Then, setting $\beta' = \kappa^{-1}(\mathbf{b}')$, we would have $\mathcal{R}_{\text{emp}}(\beta') < \mathcal{R}_{\text{emp}}(\hat{\beta})$, which contradicts the definition of β in Equation (7). $\square$

4 Theoretical convergence analysis

4.1 Assumptions

We now establish theoretical guarantees for FunCast under the following assumptions. Provided they exist, the smallest and largest eigenvalues of a matrix $\mathbf{A}$ are denoted by $\lambda_{\min}(\mathbf{A})$ and $\lambda_{\max}(\mathbf{A})$, respectively. Moreover, the trace of $\mathbf{A}$ is denoted by $\text{Tr}(\mathbf{A})$. The expectation and variance of a random variable U are denoted by $E[U]$ and $V[U]$, respectively. For a random vector $\mathbf{U}$, the same notations are used to denote the expectation vector and the covariance matrix. Without loss of generality, in this section, the covariates $X_{i,0}^{(1)}(t), \dots, X_{i,p}^{(1)}(t)$, which include the past of Y , are deterministic. Consequently, the matrix $\mathbf{X}$ is also deterministic. Otherwise, the following results can still be established conditionally on $\mathbf{X}$.

Assumption 3 There exist parameters $\beta^* \in \mathcal{P}$ such that $Y_i^{(2)}(t) = \hat{Y}_i^{(2)}(t; \beta^*) + \varepsilon_i(t)$, where $\varepsilon_i(t)$ is a white noise process, meaning that $E[\varepsilon_i(t)] = 0$, $V[\varepsilon_i(t)] = \sigma^2 < \infty$, and $E[\varepsilon_i(t)\varepsilon_i(t')] = 0$ for $t \neq t'$. Additionally, $\varepsilon_i(t)$ and $X_{i,\ell}^{(1)}(t)$ are assumed to be uncorrelated. The functions in β^* are denoted by $\beta_{k,\ell}^*(t)$.

Assumption 4 The matrix $\mathbf{X}^\top \mathbf{X}$ is invertible.

Assumption 5 There exists a finite constant $M_X > 0$ such that $|X_{i,\ell}^{(1)}(t)| \leq M_X$ for $i \in [1 : n]$, $\ell \in [0 : p]$ and $t \in [0, T]$.

Assumption 6 There exists a finite constant $M_\psi > 0$ such that for all $k \in [1 : K]$ and $t \in [T, T + H]$, $|\psi_k(t)| \leq M_\psi$.

In Assumption 3, β^* should not be confused with $\hat{\beta}$. On the one hand, $\hat{\beta}$ is defined as the minimizer of the empirical risk (6) based on the discrete measurements of the functions; on the other hand, β^* refers to hypothetical parameters that are assumed to underlie the data. There is no guarantee that β^* minimizes the empirical risk. However, Lemma 2 shows an asymptotic similarity between β^* and $\hat{\beta}$, as the square distance in H_1 between parameter $\hat{\beta}_{k,\ell}$ and $\beta_{k,\ell}^*$ is $\mathcal{O}\left(\frac{Kq\sigma^2}{nm_2}\right)$. Moreover, regarding the additive noise $\varepsilon_i(t)$, it follows that any discrete sampling $(\varepsilon_i(t_1), \dots, \varepsilon_i(t_m))$ forms a random vector with zero mean and covariance matrix $\sigma^2 \mathbf{I}_m$. Assumption 4 ensures that the problem is well-conditioned. Moreover, it is known that $\lambda_X := \lambda_{\min}(\mathbf{X}^\top \mathbf{X}) > 0$. Assumption 5 holds in most applications, and standard data preprocessing steps, particularly normalization, make it possible to bound the values of the covariates. Finally, Assumption 6 holds for most standard functional bases (e.g., cubic splines, Fourier basis, polynomial basis).

4.2 Convergence rate

We now derive a convergence rate for the predictive risk of FunCast, defined as the error between the prediction using estimated parameters $\hat{\beta}$ and the true parameters β^* . The *predictive risk* is defined as the expected squared norm of the prediction error over H_2 :

$$\mathcal{R}_{\text{pred}}(\hat{\beta}, \beta^*) = \frac{1}{n} \sum_{i=1}^n \mathcal{R}_{\text{pred}}^{(i)}(\hat{\beta}, \beta^*),$$

where

$$\mathcal{R}_{\text{pred}}^{(i)}(\hat{\beta}, \beta^*) = E \left[\left\| \hat{Y}_i^{(2)}(\cdot; \hat{\beta}) - \hat{Y}_i^{(2)}(\cdot; \beta^*) \right\|_{H_2}^2 \right]$$

Theorem 3 Under Assumptions 3 to 6, the predictive risk satisfies

$$\mathcal{R}_{pred}(\hat{\beta}, \beta^*) = \mathcal{O}\left(\frac{K^3 p^2 q T H \sigma^2}{n m_2}\right). \quad (9)$$

Interpretation of the convergence rate

The formulation of the convergence rate in (9) provides a theoretical explanation of the impact of the problem parameters. First, a high noise level σ^2 degrades the model predictions since empirical measurements of Y deviate further from the true values of Y . Next, K , p , and q define the model complexity, i.e., the number of parameters to estimate. Recall that the model relies on estimating $K(p+1)$ functional parameters, or equivalently, on estimating Kq scalars. The factor H/m_2 represents the sampling rate of the future curve of Y . Under-sampling deteriorates prediction quality since a sufficiently high number of measurement points is required to accurately predict the future of Y . Finally, the factor TH/n indicates that if the considered functions are defined over long time intervals, the model requires more realizations to achieve convergence.

Impact of m_1 and h_ℓ

Unlike what is observed for the future of Y , the sampling rate m_1 of the past covariates does not appear in the convergence rate. The same applies to the number of expansion coefficients h_ℓ used to represent the covariate X_ℓ . However, these parameters have practical implications. First, for Assumption 2 to hold, it is necessary that $m_1 \geq h_\ell$ to prevent the expansion problem from being underdetermined, which ensures that m_1 is not too small. Next, as presented in Section 3.3, h_ℓ plays a role in matrix multiplications by affecting the dimensions of the matrices (e.g. Equation (8)). Consequently, intuitively, low values of h_ℓ may compromise the well-conditioning of the problem, and more specifically, the validity of Assumption 4. Although the convergence rate does not explicitly depend on the discretization of the past interval, appropriate choices of m_1 and h_ℓ are required to ensure valid basis expansions and numerical stability.

4.3 Proof of Proposition 3

In this section, classical results from matrix algebra are used (Soch et al. 2024). From Assumption 3 and using the notations and results from Section 3, we have $\mathbf{y} = \mathbf{X}\mathbf{b}^* + \boldsymbol{\varepsilon}$, where $\mathbf{b}^* = \kappa(\beta^*)$ and $\boldsymbol{\varepsilon} = [\boldsymbol{\varepsilon}_1^\top, \dots, \boldsymbol{\varepsilon}_n^\top]^\top \in \mathbb{R}^{nm_2}$ with $\boldsymbol{\varepsilon}_i = [\varepsilon_i(t_{m_1+1}), \dots, \varepsilon_i(t_{m_1+m_2})] \in \mathbb{R}^{m_2}$. This setting corresponds to the ordinary least squares (OLS) framework, and the coefficients $\hat{\mathbf{b}}$ established in Proposition 2 are, in this section, the OLS estimator of $\mathbf{b}^*$, leading to the following lemmas.

Lemma 1 An upper bound of the expected quadratic error between $\hat{\mathbf{b}}$ and $\mathbf{b}^*$ is $E\left[\|\hat{\mathbf{b}} - \mathbf{b}^*\|_2^2\right] \leq \frac{Kq\sigma^2}{\lambda_X n m_2}$.

Proof of Lemma 1 The OLS estimator $\hat{\mathbf{b}}$ satisfies

$$E\left[\|\hat{\mathbf{b}} - \mathbf{b}^*\|_2^2\right] = \frac{\sigma^2}{n m_2} \text{Tr}\left((\mathbf{X}^\top \mathbf{X})^{-1}\right).$$

Let $\lambda_1, \dots, \lambda_{Kq}$ be the eigenvalues of $\mathbf{X}^\top \mathbf{X}$. Then $\text{Tr}\left((\mathbf{X}^\top \mathbf{X})^{-1}\right) = \frac{1}{\lambda_1} + \dots + \frac{1}{\lambda_{Kq}} \leq \frac{Kq}{\lambda_X}$. $\square$

In the following, for $\ell = 0, \dots, p$, we define the symmetric matrix $\mathbf{G}_\ell \in \mathcal{M}_{q_\ell, q_\ell}(\mathbb{R})$ as $[\mathbf{G}_\ell]_{r, r'} = \int_0^T B_{\ell, r}(t) B_{\ell, r'}(t) dt$. Let $\lambda_B = \max\{\lambda_{\max}(\mathbf{G}_\ell); \ell = 0, \dots, p\}$. Note that if the basis $\mathbf{B}_\ell$ is orthonormal in H_1 , then $\lambda_B = 1$.

Lemma 2 For all $k = 1, \dots, K$ and $\ell = 0, \dots, p$: $E\left[\|\beta_{k, \ell}^* - \hat{\beta}_{k, \ell}\|_{H_1}^2\right] \leq \frac{\lambda_B K q \sigma^2}{\lambda_X n m_2}$.

Proof of Lemma 2 For $k = 1, \dots, K$, $\ell = 0, \dots, p$ and $r = 1, \dots, q_\ell$, let $\delta_{k, \ell, r} = b_{k, \ell, r}^* - \hat{b}_{k, \ell, r}$. Then,

$$\begin{aligned} \|\beta_{k, \ell}^* - \hat{\beta}_{k, \ell}\|_{H_1}^2 &= \int_0^T \left(\beta_{k, \ell}^*(t) - \hat{\beta}_{k, \ell}(t)\right)^2 dt \\ &= \int_0^T \left(\sum_{r=1}^{q_\ell} \delta_{k, \ell, r} B_{\ell, r}(t)\right)^2 dt \\ &= \int_0^T \sum_{r=1}^{q_\ell} \sum_{r'=1}^{q_\ell} \delta_{k, \ell, r} \delta_{k, \ell, r'} B_{\ell, r}(t) B_{\ell, r'}(t) dt \\ &= \sum_{r=1}^{q_\ell} \sum_{r'=1}^{q_\ell} \delta_{k, \ell, r} \delta_{k, \ell, r'} [\mathbf{G}_\ell]_{r, r'} \\ &= (\mathbf{b}_{k, \ell}^* - \hat{\mathbf{b}}_{k, \ell})^\top \mathbf{G}_\ell (\mathbf{b}_{k, \ell}^* - \hat{\mathbf{b}}_{k, \ell}) \end{aligned}$$

$$\begin{aligned} &\leq \lambda_{\max}(\mathbf{G}_\ell) \left\| \mathbf{b}_{k,\ell}^* - \hat{\mathbf{b}}_{k,\ell} \right\|_2^2 \quad (\text{a}) \\ &\leq \lambda_B \left\| \mathbf{b}_{k,\ell}^* - \hat{\mathbf{b}}_{k,\ell} \right\|_2^2. \end{aligned}$$

where (a) holds for any symmetric matrix. Then we use the fact that $\left\| \mathbf{b}_{k,\ell}^* - \hat{\mathbf{b}}_{k,\ell} \right\|_2^2 \leq \left\| \hat{\mathbf{b}} - \mathbf{b}^* \right\|_2^2$ and use Lemma 1. $\square$

We now continue the proof of Proposition 3. For $i = 1, \dots, n$, we have

$$\hat{Y}_i^{(2)}(t; \hat{\beta}) - \hat{Y}_i^{(2)}(t; \beta^*) = \sum_{k=1}^K a_k \psi_k(t),$$

with

$$a_k = \sum_{\ell=0}^p \langle X_{i,\ell}^{(1)}; \hat{\beta}_{k,\ell} - \beta_{k,\ell}^* \rangle_{H_1}.$$

Next, $\left\| \sum_{k=1}^K a_k \psi_k \right\|_{H_2} \leq \sum_{k=1}^K |a_k| \|\psi_k\|_{H_2} \leq M_\psi \sqrt{H} \sum_{k=1}^K |a_k|$, using the triangle inequality in H_2 followed by Assumption 6. Consequently,

$$\begin{aligned} \left\| \hat{Y}_i^{(2)}(\cdot; \hat{\beta}) - \hat{Y}_i^{(2)}(\cdot; \beta^*) \right\|_{H_2}^2 &\leq M_\psi^2 H \left(\sum_{k=1}^K |a_k| \right)^2 \\ &\leq M_\psi^2 H K \sum_{k=1}^K |a_k|^2, \end{aligned}$$

using the Cauchy–Schwarz inequality in $\mathbb{R}^K$. We now derive an upper bound for $|a_k|^2$. First,

$$|a_k| \leq \sum_{\ell=0}^p \left| \langle X_{i,\ell}^{(1)}; \hat{\beta}_{k,\ell} - \beta_{k,\ell}^* \rangle_{H_1} \right| \quad (\text{a})$$

$$\leq \sum_{\ell=0}^p \left\| X_{i,\ell}^{(1)} \right\|_{H_1} \left\| \hat{\beta}_{k,\ell} - \beta_{k,\ell}^* \right\|_{H_1} \quad (\text{b})$$

$$\leq M_X \sqrt{T} \sum_{\ell=0}^p \left\| \hat{\beta}_{k,\ell} - \beta_{k,\ell}^* \right\|_{H_1}, \quad (\text{c})$$

where (a) is the triangle inequality, (b) is the Cauchy–Schwarz inequality in H_1 , and (c) follows from Assumption 5. Then,

$$\begin{aligned} |a_k|^2 &\leq M_X^2 T \left(\sum_{\ell=0}^p \left\| \hat{\beta}_{k,\ell} - \beta_{k,\ell}^* \right\|_{H_1} \right)^2 \\ &\leq M_X^2 T (p+1) \sum_{\ell=0}^p \left\| \hat{\beta}_{k,\ell} - \beta_{k,\ell}^* \right\|_{H_1}^2, \end{aligned}$$

where the Cauchy–Schwarz inequality in $\mathbb{R}^{p+1}$ is used. Consequently,

$$\begin{aligned} &\left\| \hat{Y}_i^{(2)}(\cdot; \hat{\beta}) - \hat{Y}_i^{(2)}(\cdot; \beta^*) \right\|_{H_2}^2 \\ &\leq M_\psi^2 M_X^2 K T H (p+1) \sum_{k=1}^K \sum_{\ell=0}^p \left\| \hat{\beta}_{k,\ell} - \beta_{k,\ell}^* \right\|_{H_1}^2. \end{aligned}$$

Taking the expectation and using Lemma 2 completes the proof.

5 Hyperparameter Selection

In this section, the hyperparameter selection for FunCast is discussed. The optimized parameter vector $\hat{\mathbf{b}}$ defined in Proposition 2 is computed not only from the data but also from the function bases $(\theta_{\ell,1}, \dots, \theta_{\ell,h_\ell})$, $(B_{\ell,1}, \dots, B_{\ell,q_\ell})$, and $(\psi_1, \dots, \psi_K)$. These bases ensure the smooth representation of, respectively, predictor covariates, functional parameters, and forecasting of Y . We assume here that the functional bases are fixed. For functional data of general form, cubic spline bases can be considered, as they adapt efficiently to various types of functions, provided their number is properly chosen (Ramsay and Silverman 2005). Therefore, the hyperparameters $\boldsymbol{\eta}_0$ of FunCast reduce to the basis sizes: $\boldsymbol{\eta}_0 = [K, h_0, \dots, h_p, q_0, \dots, q_p]$, leading to $3 + 2p$ hyperparameters. A challenge in using FunCast is the high number of hyperparameters, which increases with the number of covariates. Without covariates, the model has three hyperparameters (K, h_0, q_0) . Each additional covariate introduces two new hyperparameters (h_ℓ, q_ℓ) , making the choice of $\boldsymbol{\eta}_0$ difficult in practice. To address this issue, we propose a practical solution to reduce the number of hyperparameters by automating certain selections. Instead of directly initializing $\boldsymbol{\eta}_0$, the user initializes intermediate hyperparameters $\boldsymbol{\eta}_1$, and a procedure $\mathcal{H}$ maps $\boldsymbol{\eta}_1$ to a final hyperparameter vector $\boldsymbol{\eta}_0$ of size $3 + 2p$. The advantage of $\boldsymbol{\eta}_1$ is that it is both smaller than $\boldsymbol{\eta}_0$ and independent of the number of covariates. We now detail the definition of $\boldsymbol{\eta}_1$ and the procedure $\mathcal{H}$. The construction of $\boldsymbol{\eta}_0$ from $\boldsymbol{\eta}_1$ consists of three steps.

Step 1: The first component of $\boldsymbol{\eta}_1$ is K , the number of basis functions in the expansion of the future of Y , which is therefore directly included in $\boldsymbol{\eta}_0$.

Step 2: The next hyperparameters to be initialized in $\boldsymbol{\eta}_0$ are $h_0, \dots, h_p$. The integer h_ℓ represents the size of the functional basis used to approximate the function $X_\ell^{(1)}(t)$. In general, selecting an optimal number of basis functions is a challenging task. A value that is too low may lead to a poor approximation, failing to capture the detailed behavior of the function. Conversely, a value that is too high may result in overfitting, where noise is also approximated. A possible solution is to leverage the n realizations of $X_\ell^{(1)}(t)$ and introduce a criterion to automatically determine h_ℓ . Let $\hat{X}_{i,\ell}^{(1)}(t)$ be the smoothed approximation of $X_{i,\ell}^{(1)}(t)$. Motivated by (Ramsay and Silverman 2005) in Chapter 4, we consider the following Root Residual Sum of Squares criterion:

$$\text{RRSS}_\ell = \frac{\sqrt{\sum_{i=1}^n \sum_{j=1}^{m_1} \left(X_{i,\ell}^{(1)}(t_j) - \hat{X}_{i,\ell}^{(1)}(t_j) \right)^2}}{m_1 - h_\ell}. \quad (10)$$

Intuitively, this criterion ensures a faithful approximation of the function while penalizing large values of h_ℓ . Thus, h_ℓ is initialized internally as the value that minimizes the criterion RRSS_ℓ , and is consequently not an hyper-parameter to tune.

Step 3: Finally, the remaining hyperparameters to be initialized in $\boldsymbol{\eta}_0$ are $q_0, \dots, q_p$. The integer q_ℓ represents the size of the basis used to approximate the K functional parameters $\beta_{1,\ell}(t), \dots, \beta_{K,\ell}(t)$. For interpretability purposes, parameters with low roughness are preferred. From a hyperparameter perspective, this translates into choosing $q_\ell \leq h_\ell$. Therefore, we set $q_\ell = (1-s)h_\ell$, where $s \in [0, 1]$ is a smoothness coefficient. Intuitively, a value of s close to 0 results in rough parameters, whereas a value close to 1 leads to smooth parameters. Thus, q_ℓ is computed based on h_ℓ , which was determined internally in the previous step. This normalized coefficient s is the second component of $\boldsymbol{\eta}_1$.

Finally, the intermediate hyperparameters are simply given by $\boldsymbol{\eta}_1 = [K, s]$, and the mapping $\mathcal{H}(\boldsymbol{\eta}_1)$ correctly leads to hyperparameters $\boldsymbol{\eta}_0$ that can be directly used in FunCast.

Discussions

Below are some remarks regarding this method:

- In step 1, the criterion (10) was chosen in this work, but it is not the only possible criterion. For instance, the numerator can be replaced by any other error metric without altering the hyperparameter construction procedure.
- The hyperparameters $\boldsymbol{\eta}_1$ do not influence the outcome of step 1. Consequently, when testing two different sets of hyperparameters $\boldsymbol{\eta}_1 \neq \boldsymbol{\eta}'_1$, the values of $h_0, \dots, h_p$ need to be computed only once using the criterion.
- $\mathcal{H}$ is not surjective; thus, by basing hyperparameter selection on the exploration of $\boldsymbol{\eta}_1$, the exploration of $\boldsymbol{\eta}_0$ is inherently restricted. This may lead to suboptimal hyperparameters. However, experiments not reported in this paper show, on the one hand, that exploring $\boldsymbol{\eta}_1$ is computationally faster due to the reduced number of hyperparameters and the introduction of the normalized parameter s , and on the other hand, that this hyperparameter tuning does not significantly degrade the forecasting performance of FunCast.

6 Application on synthetic data

6.1 Synthetic functional data

In this section, FunCast is applied to synthetic data. In the experiments, T and H are integers such that $T + H = 300$, and the functions are measured every second at the time points: $t_1 = 1, \dots, t_{m_1} = T, t_{m_1+1} = T + 1, \dots, t_{m_1+m_2} = T + H$. We introduce $\tau = \frac{T}{T+H}$ as the proportion of the observed curve. Let $\mu_1 < \dots < \mu_{10}$ be equidistant points satisfying $\mu_1 = 0$ and $\mu_u = \mu_{u-1} + \frac{T+H}{10}$ for $u = 1, \dots, 10$. We define the following function: $G_u(t) = \exp(-(t - \mu_u)^2/10^4)$. In the synthetic data, the function $Y(t)$ has a single covariate $X(t)$. First, the denoised functions are generated according to the following equations: $x(t) = \sum_{u=1}^{10} x_u G_u(t)$ and $y(t) = \gamma_0(t) + \gamma(t)x(t)$, where $\gamma_0(t) = 10^{-2}t$, $\gamma(t) = \sin(2\pi \cdot 5 \times 10^{-3} \cdot t)$ and finally $x_u \sim \mathcal{U}[-1, 1]$. Then, a Gaussian noise is added to both functions: $X(t) = x(t) + \varepsilon_X(t)$ and $Y(t) = y(t) + \varepsilon_Y(t)$ where $\varepsilon_X(t) \sim \mathcal{N}(0, \sigma_X^2)$ and $\varepsilon_Y(t) \sim \mathcal{N}(0, \sigma_Y^2)$. Additive Gaussian noise is characterized by the signal-to-noise ratio (SNR), which is expressed in decibels (dB). For a given SNR, the corresponding noise level σ_X satisfies

$\text{SNR} = 10 \log_{10} (P_x / \sigma_X^2)$ dB. In this expression, P_x is the power of $x(t)$, defined by $P_x = \frac{1}{T+H} \sum_{j=1}^{m_1+m_2} x(t_j)^2$. Similarly, the power P_y of the function $y(t)$ is defined, and the noise level σ_Y satisfies the same relation with the SNR. The use of the SNR provides a normalized measure for managing the addition of noise in the functional data. The SNR adapts the noise level to the amplitudes of the functions. For a given SNR, it is generally the case that $\sigma_X \neq \sigma_Y$, which allows for injecting Gaussian noise of similar strength into $X(t)$ and $Y(t)$ relative to their amplitude. Of course, the lower the SNR, the stronger the noise. Critical and unfavorable SNR values include $\text{SNR} = 0$ dB when the function and the noise are of equal amplitude, or $\text{SNR} < 0$ dB when the function is masked by the noise. Figure 2 illustrates how the SNR affects the functional data.

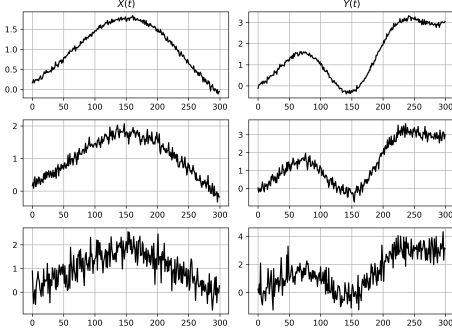


Fig. 2 A realization of $X(t)$ and $Y(t)$ with three different SNR values. From top to bottom, the SNR values are SNR = 30 dB, 20 dB, and 10 dB.

6.2 Simulation procedure

A simulation consists of generating synthetic functional data with training, validation, and test realizations, followed by tuning the model hyperparameters using the validation data, and finally performing inference on the test data. These steps are detailed below.

Step 1: Data generation. Functional data are randomly generated with n_{train} training samples, n_{val} validation samples, and n_{test} test samples. In the following experiments, we set $n_{\text{val}} = 100$ and $n_{\text{test}} = 2000$. The specific values of T and H , n_{train} , and the SNR vary depending on the scenario. The simulation scenarios are introduced in Section 6.4.

Step 2: Hyperparameter selection. The hyperparameters $\eta_1 = [K, \alpha]$, defined in Section 5, are tuned using cross-validation. A grid search is performed where K is explored in $\{5, 10, 15\}$ and s in $\{0.1, 0.4, 0.7, 1\}$. The metric used to evaluate the hyperparameter combinations is the Root Mean Squared Error (RMSE) on the validation data. The RMSE is formally defined in Section 6.3. Cubic B-spline functions are used as basis functions for ψ and the basis introduced in Assumptions 2 and 1.

Step 3: Model optimization. Once the hyperparameters are selected, the functional parameters of FunCast are optimized by computing $\hat{\beta}$ according to Proposition 2.

Step 4: Inference. The model is used to forecast the future values of Y on the test data. Specifically, inference consists of computing the vectors $[\hat{Y}_i(T+1; \hat{\beta}), \dots, \hat{Y}_i(T+H; \hat{\beta})]$ for $i \in [1 : n_{\text{test}}]$ according to Proposition 1. Finally, the predictions are evaluated using the error metrics defined in Section 6.3.

6.3 Performance metrics

This section presents the metrics used to assess the forecasting quality of our model as well as that of competing models.

Root Mean Squared Error (RMSE)

This standard metric quantifies the deviation between the actual values of the future of Y and the predicted values using the following formula:

$$\text{RMSE} = \sqrt{\frac{1}{n_{\text{test}} m_2} \sum_{i=1}^{n_{\text{test}}} \sum_{j=m_1+1}^{m_1+m_2} (Y_i(t_j) - \hat{Y}_i(t_j))^2}.$$

The RMSE is a positive metric and approaches zero as predictions become more accurate. Among its properties, it is worth noting that RMSE has the same unit as the target function Y and that this metric is sensitive to outlier predictions.

Symmetric Mean Absolute Percentage Error (SMAPE)

The SMAPE (Shcherbakov et al. 2013) is a robust relative error metric that is expressed as a percentage:

$$\text{SMAPE}(\%) = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \text{SMAPE}^{(i)}(\%)$$

where

$$\text{SMAPE}^{(i)}(\%) = \frac{100}{m_2} \sum_{j=m_1+1}^{m_1+m_2} \frac{|Y_i(t_j) - \hat{Y}_i(t_j)|}{\frac{1}{2} (|Y_i(t_j)| + |\hat{Y}_i(t_j)|)}.$$

The closer the SMAPE is to zero, the more accurate the prediction. Moreover, unlike the traditional Mean Average Percentage Error metric, SMAPE prevents divergence when some target values are close to zero.

6.4 Scenarios

Several simulation scenarios are introduced to evaluate the robustness of our model. The parameters characterizing these scenarios are the number of training realizations n_{train} , the ratio τ of the past interval $[0, T]$ to the total interval $[0, T + H]$, and the SNR. Two values were chosen for each parameter. Specifically, we have $n_{\text{train}} \in \{50, 100\}$, $\tau \in \{40\%, 80\%\}$, and $\text{SNR} \in \{10\text{dB}, 20\text{dB}\}$, leading to a total of eight parameter combinations. The scenarios are summarized in Table 1.

Scenario	n_{train}	τ (%)	SNR (dB)
(S1)	100	40	10
(S2)	100	40	20
(S3)	100	80	10
(S4)	100	80	20
(S5)	50	40	10
(S6)	50	40	20
(S7)	50	80	10
(S8)	50	80	20

Table 1 Summary of the simulation scenarios.

6.5 Competitors

The competitor models chosen for this study are classified into two categories.

Multiple-Inputs Multiple-Outputs (MIMO) regression

In this work, the problem of forecasting the future of Y can be formulated as a regression from $\mathbb{R}^{2m_1}$ to $\mathbb{R}^{m_2}$, where the m_1 discrete values of the past of X and Y are used to forecast the m_2 future discrete values of Y . The selected competitors for this task are classical statistical models: linear regression (LIN), ridge regression (RIDGE), lasso regression (LASSO), XGBoost (XGB), and support vector regression (SVR). These models are designed for univariate target prediction, whereas the future of Y is multivariate with m_2 points. Therefore, in the implementation, for each competitor, m_2 independent regression models from $\mathbb{R}^{2m_1}$ to $\mathbb{R}$ are trained to predict Y . Finally, the last selected regression competitor is the multi-layer perceptron (MLP), which, unlike the previous competitors, directly returns the m_2 future values of $Y(t)$. The architecture of this neural network is as follows: an input linear layer of size $2m_1$, a dense layer with ReLU activation, and an output linear layer of size m_2 . The optimizer is Adam and the loss function is the Mean Squared Error.

Time series forecasting

Time series models provide an alternative approach for forecasting the future of Y . Unlike our functional model and the MIMO regression competitors, this approach treats each realization $\{X_i(t), Y_i(t)\}$ independently. The first time series competitor is a univariate autoregressive model based on XGBoost, referred to as XGB(TS). For each test realization $i = 1, \dots, n_{\text{test}}$, XGB(TS) is trained on the past values of $Y_i(t)$, i.e., $[Y_i(t_1), \dots, Y_i(t_{m_1})]$, to forecast the next value based on a given number of previous values. Then, the m_2 future values of $Y_i(t)$ are predicted iteratively. Both during training and inference, XGB(TS) does not include the covariate X . The second competitor is the multivariate time series model Vector Autoregression (VAR), which, for each test realization $i = 1, \dots, n_{\text{test}}$, jointly processes $Y_i(t)$ and $X_i(t)$ to capture the relationships between the two functions. VAR is trained on the m_1 past points of $Y_i(t)$ and $X_i(t)$ to sequentially

forecast the m_2 future points of $Y_i(t)$. VAR also forecasts the m_2 future points of $X_i(t)$, but this prediction is not considered in the experiments.

Hyperparameters and software implementation

As with FunCast, the hyperparameters of the competitors were selected through grid search with cross-validation. The hyperparameters tested for each competitor are detailed in Table A1. All models were implemented in R, except for MLP, which was coded in Python with PyTorch. The experiments were conducted on a Windows 11 system with an AMD Ryzen 5 PRO 7530U CPU and 32 GB of RAM. For fairness, the hyperparameter grids for FunCast and the competing methods are of comparable size.

6.6 Results

The experimental results on synthetic data are provided in Tables 2 and 3, with respectively the RMSE and SMAPE of all models for the eight scenarios. The results show that FunCast achieves the lowest RMSE and SMAPE values across all considered scenarios. In the most favorable scenario (S4), the SMAPE of FunCast is 7.69%, while in the most unfavorable scenario (S5), the SMAPE reaches 37%. Notably, the regularization techniques RIDGE and LASSO significantly improve performance compared to simple linear regression (LIN). The L^2 penalty in RIDGE constrains the magnitude of the parameters, thereby reducing the model variance, while the L^1 penalty in LASSO automatically performs variable selection by setting some parameters to zero. Ultimately, regularization reduces the impact of noise and improves the generalization of these two competitors. It is also observed that the performance of VAR is significantly degraded in scenarios (S1), (S2), (S5), and (S6). This can be explained by the fact that these are precisely the scenarios where the covariates are observed over a short interval, while the future of Y is forecasted over a long interval. In these cases, VAR does not have enough training points to converge. In all scenarios, the time series models XGB(TS) and VAR generally yield lower performance. This is due, on the one hand, to the construction of the synthetic data, where extrapolating the continuation of the Y curve solely from its past (and also from the past

of X in the case of VAR) appears challenging, and on the other hand, to the propagation of errors during the sequential forecast of Y .

7 Application on real-world data

7.1 Cycling Data

In this section, FunCast and the competitors are evaluated on real functional data. The Cycling dataset, used in this work and examined in (Jacques and Samardžić 2022) and (Tamo Tchomgui et al. 2024), contains recordings of several parameters over a 30-minute duration for 216 cycling sessions. These parameters are: power developed by the cyclist (in Watts), road slope (in percent), heart rate (in beats per minute), speed (in kilometers per hour), pedaling cadence (in revolutions per minute), altitude (in meters), and outdoor temperature (in degrees Celsius). The function Y to be forecasted in the future is power. According to (Jacques and Samardžić 2022), the parameters with the greatest impact on power are slope (SLOPE), heart rate (HR), and speed (KPH), which are therefore considered potential covariates. As in (Tamo Tchomgui et al. 2024), a logarithmic transformation is applied to the power. Figure 3 displays several curves of these parameters over 30 seconds at the 20th minute.

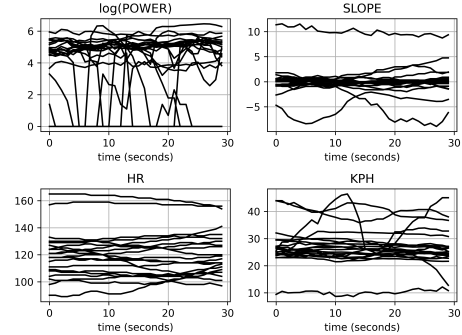


Fig. 3 For 20 cyclists, visualization of the logarithm of power output, slope, heart rate, and speed.

7.2 Results

The forecasting performance of FunCast on the Cycling data is evaluated under two scenarios. The data are taken from the 20th minute onward. The

Model	(S1)	(S2)	(S3)	(S4)	(S5)	(S6)	(S7)	(S8)
LIN	50 (87.4)	15.1 (19.1)	38.9 (92.2)	34.7 (80)	781 (3.74 · 10 ³)	57.7 (167)	40.2 (71.3)	14.8 (18.1)
RIDGE	0.781 (6.88 · 10 ⁻³)	0.421 (1.15 · 10 ⁻²)	0.67 (2.82 · 10 ⁻³)	0.226 (2.59 · 10 ⁻³)	0.804 (1.03 · 10 ⁻²)	0.452 (1.36 · 10 ⁻²)	0.684 (5.88 · 10 ⁻³)	0.238 (4.87 · 10 ⁻³)
LASSO	0.798 (9.7 · 10 ⁻³)	0.436 (1.49 · 10 ⁻²)	0.667 (3.19 · 10 ⁻³)	0.234 (2.11 · 10 ⁻³)	0.844 (1.55 · 10 ⁻²)	0.513 (2.24 · 10 ⁻²)	0.702 (6.97 · 10 ⁻³)	0.259 (5.66 · 10 ⁻³)
XGB	0.85 (1.09 · 10 ⁻²)	0.536 (1.45 · 10 ⁻²)	0.709 (5.05 · 10 ⁻³)	0.291 (7.74 · 10 ⁻³)	0.907 (1.59 · 10 ⁻²)	0.608 (2.67 · 10 ⁻²)	0.751 (1.28 · 10 ⁻²)	0.37 (2.36 · 10 ⁻²)
SVR	0.85 (1.52 · 10 ⁻²)	0.441 (1.89 · 10 ⁻²)	0.732 (1.4 · 10 ⁻²)	0.249 (1.89 · 10 ⁻³)	0.876 (1.28 · 10 ⁻²)	0.485 (2.55 · 10 ⁻²)	0.709 (1.17 · 10 ⁻²)	0.248 (3.76 · 10 ⁻³)
MLP	0.824 (1.08 · 10 ⁻²)	0.404 (1.66 · 10 ⁻²)	0.681 (1.58 · 10 ⁻²)	0.228 (4.54 · 10 ⁻²)	0.835 (1.51 · 10 ⁻²)	0.426 (2.11 · 10 ⁻²)	0.686 (1.01 · 10 ⁻²)	0.231 (1.39 · 10 ⁻²)
XGB(TS)	1.57 (1.3 · 10 ⁻²)	1.47 (1.8 · 10 ⁻²)	0.931 (1.64 · 10 ⁻²)	0.654 (1 · 10 ⁻²)	1.56 (1.45 · 10 ⁻²)	1.45 (2.1 · 10 ⁻²)	0.934 (2.97 · 10 ⁻²)	0.65 (1.21 · 10 ⁻²)
VAR	> 10 ³	> 10 ³	1.25 (2.6 · 10 ⁻¹)	1.35 (1.41 · 10 ⁻¹)	> 10 ³	> 10 ³	1.24 (2.95 · 10 ⁻¹)	1.37 (1.63 · 10 ⁻¹)
FunCast	0.716 (5.28 · 10 ⁻³)	0.313 (4.81 · 10 ⁻³)	0.624 (2.48 · 10 ⁻³)	0.199 (6.05 · 10 ⁻⁴)	0.729 (1.17 · 10 ⁻²)	0.325 (1.114 · 10 ⁻²)	0.627 (3.08 · 10 ⁻³)	0.201 (1.21 · 10 ⁻³)

Table 2 RMSE of FunCast and competitors on synthetic data for the scenarios introduced in Section 6.4. The mean value and standard deviation over the $n_{\text{simu}} = 30$ simulations are reported.

Model	(S1)	(S2)	(S3)	(S4)	(S5)	(S6)	(S7)	(S8)
LIN	> 100	> 100	> 100	> 100	> 100	> 100	> 100	> 100
RIDGE	38.8 (3.55 · 10 ⁻¹)	21.1 (3.7 · 10 ⁻¹)	24.1 (2.9 · 10 ⁻¹)	8.68 (2.19 · 10 ⁻¹)	39.4 (4.28 · 10 ⁻¹)	21.9 (5.3 · 10 ⁻¹)	24.5 (3.62 · 10 ⁻¹)	9.1 (2.32 · 10 ⁻¹)
LASSO	39.1 (3.58 · 10 ⁻¹)	21.3 (4.88 · 10 ⁻¹)	24 (3.04 · 10 ⁻¹)	8.95 (2.1 · 10 ⁻¹)	40.5 (56.5 · 10 ⁻¹)	24 (6.31 · 10 ⁻¹)	25 (3.73 · 10 ⁻¹)	9.86 (2.85 · 10 ⁻¹)
XGB	41.3 (4.82 · 10 ⁻¹)	25.2 (5.7 · 10 ⁻¹)	25.3 (3.13 · 10 ⁻¹)	11.4 (4.55 · 10 ⁻¹)	43.3 (8.55 · 10 ⁻¹)	28.6 (1.27)	26.5 (5.54 · 10 ⁻¹)	13.6 (8.87 · 10 ⁻¹)
SVR	41 (5.48 · 10 ⁻¹)	22.1 (5.65 · 10 ⁻¹)	25.4 (8.19 · 10 ⁻¹)	9.42 (1.85 · 10 ⁻¹)	41.8 (7.59 · 10 ⁻¹)	23.2 (7.52 · 10 ⁻¹)	25.2 (3.94 · 10 ⁻¹)	9.38 (2.05 · 10 ⁻¹)
MLP	40.4 (3.44 · 10 ⁻¹)	20.2 (5.97 · 10 ⁻¹)	24.5 (5.52 · 10 ⁻¹)	8.84 (1.47)	40.6 (5.49 · 10 ⁻¹)	20.9 (9.07 · 10 ⁻¹)	24.5 (3.92 · 10 ⁻¹)	8.88 (3.8 · 10 ⁻¹)
XGB(TS)	71.6 (8 · 10 ⁻¹)	66.4 (1.28)	33.9 (8.22 · 10 ⁻¹)	23.4 (6.01 · 10 ⁻¹)	71.2 (9.22 · 10 ⁻¹)	65.4 (1.43)	34 (2.27)	23.3 (7.58 · 10 ⁻¹)
VAR	> 100	> 100	41.5 (6.05)	39 (3.84)	> 100	> 100	42 (7.51)	39.6 (3.96)
FunCast	36.7 (3.8 · 10 ⁻¹)	17.1 (2.71 · 10 ⁻¹)	22.7 (2.62 · 10 ⁻¹)	7.69 (1.63 · 10 ⁻¹)	37 (4.3 · 10 ⁻¹)	17.4 (4.95 · 10 ⁻¹)	22.7 (3.39 · 10 ⁻¹)	7.7 (1.48 · 10 ⁻¹)

Table 3 SMAPE (in %) of FunCast and competitors on synthetic data for the scenarios introduced in Section 6.4. The mean value and standard deviation over the $n_{\text{simu}} = 30$ simulations are reported.

first scenario (S1)’ consists of predicting power over a long horizon using only one covariate. Specifically, we set $T = 240$, $H = 60$, and consider SLOPE as the sole covariate. The second scenario (S2)’ focuses on short-horizon power prediction, including the three covariates SLOPE, HR, and KPH. In (S2)’, we set $T = 30$ and $H = 30$. The same competitors as in Section 6 are considered, with the same hyperparameter settings described in Table A.1. Among the 216 recordings, 80% are used for training, 10% for validation, and 10% for testing, with this split being randomly assigned for each simulation. Tables 4 and 5 summarize the model performances based on the RMSE metric.

The experimental results show that FunCast yields more accurate predictions than the competitors in scenario (S1)’, whereas scenario (S2)’ highlights the limitations of FunCast. On one hand, reducing T and H favors the competitors, as the complexity of the problem is reduced. On the other hand, incorporating multiple covariates appears to introduce noise in the prediction of FunCast. Based on these observations, FunCast seems to be more competitive for functions defined over long time intervals and involving a limited number of covariates. This suggests, for

future research, the potential benefit of incorporating regularization mechanisms into FunCast to enhance its generalization ability. That said, based on the median value, FunCast ranks among the top three performing models. Finally, it is worth noting that FunCast is computationally lighter than the selected MIMO regression competitors. As explained in Section 6.5, a MIMO competitor consists of m_2 independent prediction models, resulting in a computational complexity proportional to m_2 . In contrast, FunCast is a single model, and the computation of its optimal parameters, primarily based on matrix inversion in Proposition 2, is independent of m_2 , making it a suitable solution for high-frequency functional data.

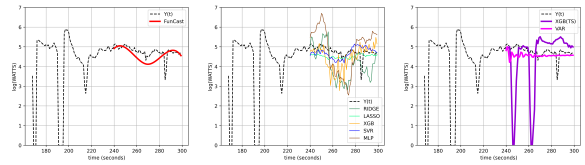


Fig. 4 Inference example in scenario (S1)’ with Cycling data.

Model	Min.	1st Qu.	Median	3rd Qu.	Max.
LIN	21.5	52.7	70.96	162.2	557.7
RIDGE	1.385	1.562	1.706	1.778	2.054
LASSO	1.22	1.473	1.585	1.675	2.112
XGB	1.343	1.555	1.632	1.725	2.049
SVR	1.327	1.469	1.608	1.776	2.163
MLP	1.36	1.63	1.721	1.837	2.005
XGB(TS)	1.569	1.831	1.952	2.15	2.498
VAR	1.33	1.573	1.752	1.965	2.274
FunCast	1.22	1.467	1.582	1.677	2.111

Table 4 RMSE of FunCast and the competitors on Cycling data with $T = 240$, $H = 60$ and SLOPE as covariate. Summary over 30 simulations.

Model	Min.	1st Qu.	Median	3rd Qu.	Max.
LIN	2.471	2.798	3.204	3.498	7.93
RIDGE	1.317	1.512	1.617	1.705	1.845
LASSO	1.427	1.477	1.536	1.648	1.78
XGB	1.432	1.751	1.797	1.832	1.97
SVR	1.317	1.54	1.698	1.797	1.852
MLP	1.351	1.523	1.677	1.787	2.042
XGB(TS)	1.392	1.869	2.082	2.183	2.501
VAR	1.938	44.89	> 100	> 100	> 100
FunCast	1.358	1.531	1.633	1.805	2.023

Table 5 RMSE of FunCast and the competitors on Cycling data with $T = 30$, $H = 30$ and SLOPE, HR and KPH as covariates. Summary over 30 simulations.

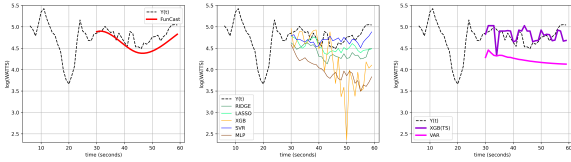


Fig. 5 Inference example in scenario (S2)' with Cycling data.

For comparison purposes, computation times for scenario (S1)' are reported in Table 6, where the computational advantage of FunCast can be observed relative to MIMO regressions, which require optimizing multiple independent models. Of course, these times should be interpreted in light of the model implementations. For example, the efficiency and maturity of PyTorch allow for low execution times for the tested MLP model.

Finally, Figures 4 and 5 show a representative inference example where FunCast performs favorably, for scenarios (S1)' and (S2)', respectively. It can be clearly seen that FunCast produces smoother predictions, as the future of Y is, by assumption, expressed in a functional basis.

8 Conclusion

Functional data analysis is a well-established field in statistics with numerous real-world applications. In this work, we focused on the problem of forecasting the future of a function Y based on past observations of functional covariates. Given the limited number of existing solutions for this task in the literature, we developed a parametric functional model that predicts the expansion coefficients of the future of Y in a fixed functional basis, using the framework of scalar-on-function

Model	Fine-tune time (per hyperparam. set)	Inference time
LIN		10.3 (0.251)
RIDGE	16.2 (0.846)	5.21 (0.4)
LASSO	11.2 (0.709)	0.221 (7.13 $\cdot 10^{-2}$)
XGB	27.7 (1.04)	25.6 (13.6)
SVR	10.4 (0.132)	3.85 (0.193)
MLP	2.74 (0.196)	1.02 (0.228)
XGB(TS)	9.09 (0.128)	5.37 (0.145)
VAR	0.734 (4.01 $\cdot 10^{-2}$)	0.47 (2.08 $\cdot 10^{-2}$)
FunCast	6.17 (7.01 $\cdot 10^{-2}$)	1.39 (0.403)

Table 6 Execution times (in seconds) of the models on scenario (S1)’ for the Cycling dataset over 10 simulations. The left column reports the fine-tuning time divided by the size of the corresponded hyperparameter grid. The right column reports the inference time on the test data. Mean values and standard deviations are reported. Fine-tuning is not performed with linear regression.

regression. We showed that the optimal parameters of FunCast admits an explicit closed-form solution, facilitating both interpretability and efficient implementation. We derived a convergence rate for the predictive error of FunCast under standard regularity and well-conditioning assumptions. From a practical standpoint, we proposed a strategy to reduce the effective hyperparameter space, based on automated basis selection and a smoothness-controlling parameter. Finally, experimental simulations on synthetic functional data and on the Cycling dataset demonstrated that our model generally outperforms existing traditional methods. The experiments on real data also revealed the limitations of our approach and suggested potential directions for improvement.

Several perspectives emerge from this work. A first direction is to incorporate regularization strategies into the model. Depending on the number of covariates and the chosen hyperparameters, the model may involve a large number of functional parameters. To improve generalization ability, it would be beneficial to limit (or eliminate) the impact of some of these parameters by introducing ℓ^1 or ℓ^2 penalization. A second direction is using conformal prediction (Tchomgui et al. 2024) to compute confidence bands for the future of Y to enhance the model interpretability.

Another perspective is to introduce mixture models (Tamo Tchomgui et al. 2024) to handle data heterogeneity. Finally, as discussed in Section 2.1, it would be beneficial to add an additional functional parameter that corresponds to a general trend.

Appendix A Competitors details

Details of FunCast competitors are shown in Table A1.

Competitor	Tuned hyperparameters	Software implementation
LIN	None	R 4.4, <code>glmnet</code> library
RIDGE	λ : Regularization intensity by penalizing the L_2 norm of the parameters to reduce overfitting. Values: 10^{-3} , 10^{-2} , 10^{-1} , 1.	R 4.4, <code>glmnet</code> library
LASSO	λ : Regularization intensity by penalizing the L_1 norm of the parameters to reduce overfitting. Values: 10^{-3} , 10^{-2} , 10^{-1} , 1.	R 4.4, <code>glmnet</code> library
XGB	max_depth : Maximum depth of the trees, impacting model complexity. Values: 3, 7, 10. learning_rate : Factor reducing the weight of each new tree; a lower value increases generalization but slows convergence. Values: 10^{-4} , 10^{-3} , 10^{-2} , 10^{-1} .	R 4.4, <code>xgboost</code> library
SVR	C : Penalty controlling the trade-off between fitting accuracy and regularization; a higher value increases overfitting. Values: 10^{-1} , 1, 10. epsilon : Insensitivity margin where prediction errors are not penalized. Values: 10^{-2} , 10^{-1} . kernel : Function projecting data into a higher-dimensional space for better separability. Values: <code>linear</code> , <code>radial</code> .	R 4.4, <code>e1071</code> library Meyer et al. (2019)
MLP	units : Number of neurons in the hidden layer. Values: 64, 128. batch_size : Number of samples used for each weight update. Values: 16, 32. learning_rate : Learning rate for weight updates. Values: 10^{-3} , 10^{-2} . epochs : Total number of passes over the training samples; a low value prevents convergence, while a high value may lead to overfitting. Values: 100, 200, 300.	Python 3.12, <code>pytorch</code> library
XGB(TS)	lags : Number of lags used to predict the next value of Y . Values: 5, 10, 20, 50. max_depth : Maximum depth of the trees, impacting model complexity. Values: 3, 10. learning_rate : Factor reducing the weight of each new tree; a lower value increases generalization but slows convergence. Values: 10^{-2} , 10^{-1} .	R 4.4, <code>xgboost</code> library
VAR	lags : Number of lags used to predict the next value of X and Y ; internally determined using the Akaike Information Criterion (AIC) with maximum lag of 20. trend : Trend term included in the model. Values: <code>none</code> , <code>const</code> , <code>trend</code> , <code>both</code> .	R 4.4, <code>vars</code> library

Table A1 Competitor models with tested hyperparameters and software implementation.

References

- Aguilera, A.M., Aguilera-Morillo, M.: Comparative study of different b-spline approaches for functional data. *Mathematical and Computer Modelling* **58**(7-8), 1568–1579 (2013)
- Antoniadis, A., Brossat, X., Cugliari, J., Poggi, J.-M.: Clustering functional data using wavelets. *International Journal of Wavelets, Multiresolution and Information Processing* **11**(01), 1350003 (2013)
- Albarrán-Lozano, I., Alonso-González, P.J., Arribas-Gil, A.: Dependence evolution in the spanish disabled population: a functional data analysis approach. *Journal of the Royal Statistical Society: Series A (Statistics in Society)* **180**(2), 657–677 (2017)
- Amir, W.A.F.W., Misro, M.Y., Mohd, M.H.: Flexible functional data smoothing and optimization using beta spline. *AIMS Mathematics* **9**(9), 23158–23181 (2024)
- Bach, F.: *Learning Theory from First Principles*. MIT press, Cambridge, MA (2024)
- Baladandayuthapani, V., Mallick, B.K., Young Hong, M., Lupton, J.R., Turner, N.D., Carroll, R.J.: Bayesian hierarchical spatially correlated functional data analysis with application to colon carcinogenesis. *Biometrics* **64**(1), 64–73 (2008) <https://doi.org/10.1111/j.1541-0420.2007.00846.x>
- Chiou, J.-M., Liou, H.-T., Chen, W.-H.: Modeling time-varying variability and reliability of free-way travel time using functional principal component analysis. *IEEE Transactions on Intelligent Transportation Systems* **22**(1), 257–266 (2019)
- Das, S., Demirer, R., Gupta, R., Mangisa, S.: The effect of global crises on stock market correlations: Evidence from scalar regressions via functional data analysis. *Structural Change and Economic Dynamics* **50**, 132–147 (2019)
- Damon, J., Guillas, S.: Estimation and simulation of autoregressive hilbertian processes with exogenous variables. *Statistical Inference for Stochastic Processes* **8**, 185–204 (2005)
- Giordani, P., Perna, S., Bianchi, A., Pizzulli, A., Tripodi, S., Matricardi, P.M.: A study of longitudinal mobile health data through fuzzy clustering methods for functional data: The case of allergic rhinoconjunctivitis in childhood. *PLOS ONE* **15**(11), 1–23 (2020) <https://doi.org/10.1371/journal.pone.0242197>
- Goldberg, Y., Ritov, Y., Mandelbaum, A.: The Best Linear Unbiased Estimator for Continuation of a Function (2010). <https://arxiv.org/abs/1005.1863>
- Goldberg, Y., Ritov, Y., Mandelbaum, A.: Predicting the continuation of a function with applications to call center data. *Journal of Statistical Planning and Inference* **147**, 53–65 (2014)
- Gao, Y., Shang, H.L.: Multivariate functional time series forecasting: Application to age-specific mortality rates. *Risks* **5**(2), 21 (2017)
- Hyndman, R.J., Booth, H.: Stochastic population forecasts using functional data models for mortality, fertility and migration. *International Journal of Forecasting* **24**(3), 323–342 (2008)
- Happ, C., Greven, S.: Multivariate functional principal component analysis for data observed on different (dimensional) domains. *Journal of the American Statistical Association* **113**(522), 649–659 (2018)
- Jiao, S., Aue, A., Ombao, H.: Functional time series prediction under partial observation of the future curve. *Journal of the American Statistical Association* **118**(541), 315–326 (2023)
- Jacques, J., Preda, C.: Functional data clustering: a survey. *Advances in Data Analysis and Classification* **8**, 231–255 (2014)
- Jacques, J., Samardžić, S.: Analysing cycling sensors data through ordinal logistic regression with functional covariates. *Journal of the Royal Statistical Society Series C: Applied Statistics* **71**(4), 969–986 (2022)

- Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F., Chang, C.-C., Lin, C.-C., Meyer, M.D.: Package ‘e1071’. The R Journal, 1–67 (2019)
- Reiss, P.T., Goldsmith, J., Shang, H.L., Ogden, R.T.: Methods for scalar-on-function regression. International Statistical Review **85**(2), 228–249 (2017)
- Ramsay, J.O., Silverman, B.W.: Functional Data Analysis. Springer, New York (2005)
- Rossi, F., Villa, N.: Support vector machine for functional data classification. Neurocomputing **69**(7-9), 730–742 (2006)
- Soch, J., et al.: StatProofBook/StatProof-Book.github.io: The Book of Statistical Proofs. Zenodo. Version 2023 (2024). <https://doi.org/10.5281/zenodo.4305949>. <https://doi.org/10.5281/zenodo.4305949>
- Shcherbakov, M.V., Brebels, A., Shcherbakova, N.L., Tyukov, A.P., Janovsky, T.A., Kamaev, V.A., *et al.*: A survey of forecast error measures. World applied sciences journal **24**(24), 171–176 (2013)
- Shah, I., Jan, F., Ali, S.: Functional data approach for short-term electricity demand forecasting. Mathematical problems in engineering **2022**(1), 6709779 (2022)
- Tchomgui, J.S.T., Barriac, V., Fraysse, G., Jacques, J., Chrétien, S.: Functional linear regression for the prediction of streaming video qoe. In: 2024 20th International Conference on Network and Service Management (CNSM), pp. 1–7 (2024). IEEE
- Tamo Tchomgui, J.S., Jacques, J., Fraysse, G., Barriac, V., Chretien, S.: A mixture of experts regression model for functional response with functional covariates. Statistics and Computing **34**(5) (2024) <https://doi.org/10.1007/s11222-024-10455-z>
- Ullah, S., Finch, C.F.: Applications of functional data analysis: A systematic review. BMC medical research methodology **13**, 1–12 (2013)
- Yi, Y., Billor, N., Liang, M., Cao, X., Ekstrom, A., Zheng, J.: Classification of eeg signals: An interpretable approach using functional data analysis. Journal of Neuroscience Methods **376**, 109609 (2022) <https://doi.org/10.1016/j.jneumeth.2022.109609>