Multi-fidelity surrogate modeling for time-series outputs

Baptiste Kerleguer

SIAM/ASA J. UNCERTAINTY QUANTIFICATION
Vol. 11, No. 2, pp. 514--539

Multifidelity Surrogate Modeling for Time-Series Outputs*

Baptiste Kerleguer\dagger

Abstract. This paper considers the surrogate modeling of a complex numerical code in a multifidelity framework when the code output is a time series and two code levels are available: a high-fidelity and expensive code level and a low-fidelity and cheap code level. The goal is to emulate a fast-running approximation of the high-fidelity code level. An original Gaussian process regression method is proposed that uses an experimental design of the low- and high-fidelity code levels. The code output is expanded on a basis built from the experimental design. The first coefficients of the expansion of the code output are processed by a cokriging approach. The last coefficients are processed by a kriging approach with covariance tensorization. The resulting surrogate model provides a predictive mean and a predictive variance of the output of the high-fidelity code level. It is shown to have better performance in terms of prediction errors than standard dimension reduction techniques.

Key words. Gaussian processes, time-series outputs, tensorized covariance, dimension reduction

MSC codes. 60G15, 62F15, 62G08

DOI. 10.1137/20M1386694

1. Introduction.

Advances in scientific modeling have led to the development of complex and computationally expensive codes. To solve the problem of computation time for tasks such as optimization or calibration of complex numerical codes, surrogate models are used. The surrogate modeling approach consists in building a surrogate model of a complex numerical code from a data set computed from an experimental design. A well-known method to build surrogate models is Gaussian process (GP) regression. This method, also called kriging, was originally proposed by [16] for geostatistics. This method has subsequently been used for computer experiments and in particular in the field of uncertainty quantification (UQ); see [32, 34].

It is common for complex codes to have different versions that are more or less accurate and more or less computationally expensive. The particular case that interests us is when codes are hierarchical; i.e., they are classified according to their computational cost and their accuracy. The more accurate the code, the more expensive it is. The autoregressive scheme presented by [14] is the first major result in the field of multifidelity GP regression. This technique has been amended by [21] in order to reduce the overall cokriging problem to several independent kriging problems. The papers [10, 29] present different application cases, and [9] is a synthesis of the use of multifidelity for surrogate modeling. In [22], the author introduces objective prior for the hyperparameters of the autoregressive model.

*Received by the editors December 17, 2020; accepted for publication (in revised form) November 10, 2022; published electronically May 12, 2023.
https://doi.org/10.1137/20M1386694

\dagger CEA, DAM, DIF, F-91297 Arpajon, France; Centre de Math\'ematiques Appliqu\'ees, Ecole Polytechnique, Institut Polytechnique de Paris, 91128 Palaiseau Cedex, France (baptiste.kerleguer@cea.fr).

New methods for multifidelity surrogate modeling have been introduced. In particular, deep GPs have been proposed to solve cases where the interactions between code levels are more complex [27]. This type of methods can deal with UQ [4], but they are time consuming and do not scale up the dimension of outputs. Neural networks have also been used to emulate multifidelity computer codes. In particular, the method proposed in [23] is a neural network method with an AR(1)-based model and is scalable to high-dimensional outputs. However, UQ is not taken into account in this method.

Among the codes with high-dimensional outputs, we are interested in those whose outputs are functions of one variable. When they are sampled, such outputs are called time series. Previous work has solved the problem of functional outputs only in the single-fidelity case. Two methods have been considered to solve the single-fidelity problem: reduce the dimension of the outputs [25] or adapt the covariance kernel [30]. In the context of dimension reduction, surrogate models generally neglect the UQ of the dimension reduction, which can be problematic for the quantification of prediction uncertainty. Moreover, large data sets (containing many low-fidelity data) lead to ill-conditioned covariance matrices that are difficult to invert. As proposed in [28], it is possible to strongly constrain the covariance kernel, which makes it possible to improve the estimation compared to the dimension reduction method. However, this method implies that the covariance must be separable, which reduces the use cases. Knowing that the AR(1) multifidelity model for GP regression uses cokriging, [26] presents an interesting approach for cokriging in the context of functional outputs, which is based on dimension reduction. An approach to multifidelity with functional outputs is presented in [11] for multivariate Hilbert space valued random fields.

In this work, we introduce an original approach to the construction of a surrogate model in the framework of hierarchical multifidelity codes with time-series outputs. The main idea is to combine a reduction method of the output dimension that fits well with the autoregressive model of multifidelity cokriging and a special single-fidelity method that allows to treat time-series output by GP regression with covariance tensorization. We address the case of one high-fidelity code and one low-fidelity code.

In section 2, we present the problem of multifidelity surrogate model for time-series outputs. Then we present our model to solve this problem.

The following three sections present the three parts of the method. In section 3, we present the method to reduce the dimension. Section 4 presents the multifidelity regression method for scalar outputs. The GP regression with the tensorized covariance method is presented in section 5.

In section 6, we find the calculation of the posterior mean and variance of the model. This section is divided into two subsections: one where we make the assumption that the orthogonal part is zero and one where we relax this assumption. In each subsection, we use two different constructions: the Dirac distribution and the cross-validation--based (CVB) method of dimension reduction.

The results presented in the numerical example in section 7 confirm that the processing of the orthogonal part is important. In this example, we test the different methods presented in the paper as well as neural network state-of-the-art multifidelity methods, and we assess their performance in terms of prediction errors and UQ.

2. The AR(1) multifidelity model with tensorized covariance and projection. The autoregressive multifidelity model has been introduced by [14]. The authors in [21] have simplified the computation. We present our AR model in section 4. In parallel, GP regression has been used with covariance tensorization in [30] and improved in [3, 28]. This method allows to extend GP regression to time-series outputs. We have exploited this method to build our own methodology for time-series regression, and we present it in section 5.

Let us consider a complex numerical code where the input is a point $\boldsymbol{x} \in Q$, with Q being a domain in $\mathbb{R}^d$ and the output a function of a one-dimensional variable. We are interested in hierarchical codes, which means that there are several code levels that can be classified according to their fidelity. In this work, we focus on only two code levels: a high-fidelity code and a low-fidelity code. In what follows, H represents the high-fidelity code and L the low-fidelity code, and the generic notation is $\mathrm{F} \in \{\mathrm{H}, \mathrm{L}\}$. For any given input $\boldsymbol{x}$, we run the F code and observe the data $z_{\mathrm{F}}(\boldsymbol{x}, t)$ for all t in a fixed regular grid $\{t_u, u = 1, \ldots, N_t\}$ in $[0, 1]$. However, the cost of the code allows only a limited number of code calls. This induces the use of the experimental design $D_{\mathrm{F}} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N_{\mathrm{F}})}\}$. N_{F} is the number of observations of the code F. The $N_t \times N_{\mathrm{F}}$ matrix containing the observations for $\boldsymbol{x} \in D_{\mathrm{F}}$ is $\boldsymbol{Z}^{\mathrm{F}}$. Our goal is to predict the values of $(z_{\mathrm{H}}(\boldsymbol{x}, t_u))_{u=1,\ldots,N_t}$ given $(\boldsymbol{Z}^{\mathrm{H}}, \boldsymbol{Z}^{\mathrm{L}})$ for a point $\boldsymbol{x} \in Q$ with the quantification of the prediction uncertainty. We model the prior knowledge of the code output $(z_{\mathrm{L}}, z_{\mathrm{H}})$ as a GP $(Z_{\mathrm{L}}, Z_{\mathrm{H}})$. We denote by $\mathcal{Z}^{\mathrm{F}}$ the random vector containing the random variables $(Z_{\mathrm{F}}(\boldsymbol{x}^{(i)}, t_u))_{\substack{u=1,\ldots,N_t \\ u=1,\ldots,N_t}}$. The combination of $\mathcal{Z}^{\mathrm{L}}$ and $\mathcal{Z}^{\mathrm{H}}$ is $\mathcal{Z}$.

2.1. Decomposition. We recall that $(Z_{\mathrm{L}}, Z_{\mathrm{H}})$ is a stochastic process. The temporal grid is $t_u, u \in \{1,\ldots,N_t\}$. We assume a hierarchical model. First, we define a model for the basis, determined by an orthogonal matrix $\boldsymbol{\Gamma}$, in the space $\mathbb{R}^{N_t}$ of the time-series outputs. Second, given the basis, we propose a model for the $\boldsymbol{x}$-dependent coefficients of the decomposition of $(Z_{\mathrm{L}}, Z_{\mathrm{H}})$ onto this basis.

Without prior knowledge, the matrix $\boldsymbol{\Gamma}$ is assumed to have a noninformative distribution in the orthogonal group O_{N_t}, the group of $N_t \times N_t$ orthogonal matrices.

Projection. Let N be an integer, smaller than the time dimension N_t. Let $\boldsymbol{\Gamma}$ be an orthogonal matrix; as presented in section 3, the columns of $\boldsymbol{\Gamma}$ are $\{\gamma_i\}_{i=1,\cdots,N_t}$. As detailed in subsection 6.1.3, the full computation of all N_t surrogate models may not give good results.

This leads to the idea of the introduction of the orthogonal subspaces $S_N^{\parallel}$ and $S_N^{\perp}$, where $S_N^{\parallel} = \mathrm{span}\{\gamma_1, \ldots, \gamma_N\}$ and $S^{\perp} = \mathrm{span}\{\gamma_{N+1}, \ldots, \gamma_N\}$.

With a given basis $\boldsymbol{\Gamma}$, it is possible to decompose the code outputs. The decomposition over the basis $\boldsymbol{\Gamma}$ gives us coefficients. We decompose Z_{H} and Z_{L} over the subspace $S_N^{\parallel}$. The rest are denoted $Z_{\mathrm{H}}^{\perp}$ and $Z_{\mathrm{L}}^{\perp}$. Consequently, we get

$$Z_{\mathrm{L}}(\boldsymbol{x}, t_u) = Z_{\mathrm{L}}^{\parallel}(\boldsymbol{x}, t_u) + Z_{\mathrm{L}}^{\perp}(\boldsymbol{x}, t_u) = \sum_{i=1}^{N} A_{i,\mathrm{L}}(\boldsymbol{x})\gamma_i(t_u) + Z_{\mathrm{L}}^{\perp}(\boldsymbol{x}, t_u) \tag{2.1}$$

and

$$Z_{\mathrm{H}}(\boldsymbol{x}, t_u) = Z_{\mathrm{H}}^{\parallel}(\boldsymbol{x}, t_u) + Z_{\mathrm{H}}^{\perp}(\boldsymbol{x}, t_u) = \sum_{i=1}^{N} A_{i,\mathrm{H}}(\boldsymbol{x})\gamma_i(t_u) + Z_{\mathrm{H}}^{\perp}(\boldsymbol{x}, t_u). \tag{2.2}$$

We are able to describe the code outputs with the basis $\mathbf{\Gamma}_N = \{\Gamma_i\}_{i=1,\dots,N}$ of S_N, the coefficients $A_{i,H}$ and $A_{i,L}$, and the orthogonal parts $Z^{\perp}_L$ and $Z^{\perp}_H$. Let $(\alpha_{i,\mathrm{ind}_F}(\mathbf{x}))_{i=1}^{N}$ be the $\mathbb{R}^{N_t}$-valued function:

$$\alpha_{i,\mathrm{ind}_F}(\mathbf{x}) = \sum_{u=1}^{m} z_{\mathrm{ind}_F}(\mathbf{x}, t_u)\Gamma_i(t_u). \tag{2.3}$$

We denote by $\alpha_i^{\mathrm{ind}_F}$ the $1 \times N_F$ row vector $(\alpha_{i,\mathrm{ind}_F}(\mathbf{x}^{(j)}))_{j=1}^{N_F}$ that contains the available data. The full data set is $\alpha = (\alpha^L, \alpha^H)$. The expression of the posterior of $\alpha_{i,\mathrm{ind}_F}$ is given in section 4.

We will use the method presented in section 4 for all $i \le N$. Given $\mathbf{\Gamma}$,

$$\begin{cases} A_{i,H}(\mathbf{x}) = \rho_{i,L}(\mathbf{x})\tilde{A}_{i,L}(\mathbf{x}) + \delta_i(\mathbf{x}), \\ \tilde{A}_{i,L}(\mathbf{x}) \perp \delta_i(\mathbf{x}), \\ \rho_{i,L}(\mathbf{x}) = g_i(\mathbf{x})^T\beta_{i,\rho_L}, \end{cases} \tag{2.4}$$

where

$$[\delta_i(\mathbf{x})|\,\mathbf{\Gamma}, \beta_{i,H}, \sigma_{i,H}] \sim \mathcal{GP}\left(f_{i,H}^T(\mathbf{x})\beta_{i,H}, \sigma_{i,H}^2\, r_{i,H}(\mathbf{x},\mathbf{x}')\right)$$

and $\tilde{A}_{i,L}(\mathbf{x})$ is a GP conditioned by α^L. Its distribution is the one of $[A_{i,L}(\mathbf{x})|\,\mathbf{\Gamma}, \mathcal{A}^L = \alpha^L, \beta_L, \sigma_L]$, where the law of $[A_{i,L}(\mathbf{x})|\,\mathbf{\Gamma}, \beta_{i,L}, \sigma_{i,L}]$ is of the form

$$[A_{i,L}(\mathbf{x})|\,\mathbf{\Gamma}, \beta_{i,L}, \sigma_{i,L}] \sim \mathcal{GP}\left(f_{i,L}^T(\mathbf{x})\beta_{i,L}, \sigma_{i,L}^2\, r_{i,L}(\mathbf{x},\mathbf{x}')\right). \tag{2.5}$$

g_i are vectors of q regression functions, $f_{i,\mathrm{ind}_F}(\mathbf{x})$ are vectors of p_{ind_F} regression functions, $r_{i,\mathrm{ind}_F}(\mathbf{x},\mathbf{x}')$ are correlation functions, β_{i,ind_F} are p_{ind_F}–dimensional vectors, β_{i,ρ_L} are q-dimensional vectors, and $\sigma^2_{i,\mathrm{ind}_F}$ are positive real numbers.

For the orthogonal part projected onto $S_N^{\perp}$, the method is different. The hypothesis is that the projection $Z^{\perp}_L(\mathbf{x}, t_u)$ of $Z_L(\mathbf{x}, t_u)$ has a negligible influence on the projection $Z^{\perp}_H(\mathbf{x}, t_u)$ of $Z_H(\mathbf{x}, t_u)$. Our assumption is that $Z^{\perp}_H(\mathbf{x}, t_u)$ is a GP with a tensorized covariance. The method we will use on $Z^{\perp}_H(\mathbf{x}, t_u)$ is described in section 5.

Note that the value $N = 0$ corresponds to full single fidelity; in this case, we use only GP regression with covariance tensorization as in section 5. For $N = N_L$, the dimension reduction is minimal, and cokriging is applied to all pairs $(A_{i,L}, A_{i,H})$ for $i \le N_L$. We will see in section 7 that the optimal N is in fact positive but smaller than N_L.

3. The basis $\mathbf{\Gamma}$.

In this section, we present different models for the random orthogonal matrix $\mathbf{\Gamma}$. Its law depends on the available information. If we have access to a lot of information based on the output of our code, we can use a Dirac distribution concentrated on one orthogonal matrix $\mathbf{\gamma}$ (it is a form of plug-in method). In contrast, the least informative law is the uniform law, i.e., the Haar measure over the group of orthogonal matrices. In order to make the best use of the available information, i.e., the known results of the code, an empirical law can be used.

3.1. Dirac distribution.

We can choose the distribution of the random matrix $\mathbf{\Gamma}$ as a Dirac distribution concentrated on a well-chosen orthogonal matrix $\mathbf{\gamma}$. This matrix is chosen

when the basis is known a priori or if the basis can be efficiently estimated from the observed code outputs. Motivated by the remark below (2.1), the matrix $\boldsymbol{\Gamma}$ can be computed using the singular value decomposition (SVD) of the code outputs.

The general idea is to choose subsets $\tilde{\mathcal{D}}_{\mathrm{F}} \subset \mathcal{D}_{\mathrm{F}}$ of size $\tilde{N}_{\mathrm{F}}$ and to apply an SVD on the $N_t \times (\tilde{N}_{\mathrm{H}} + \tilde{N}_{\mathrm{L}})$ matrix $\boldsymbol{\Xi}_{\mathrm{HL}}$ that contains the observed values $(z_{\mathrm{H}}(\boldsymbol{x}, t_u))_{u=1,\dots,N_t, \boldsymbol{x} \in \tilde{\mathcal{D}}_{\mathrm{H}}}$ and $(z_{\mathrm{L}}(\boldsymbol{x}, t_u))_{u=1,\dots,N_t, \boldsymbol{x} \in \tilde{\mathcal{D}}_{\mathrm{L}}}$. The SVD gives

$$\boldsymbol{\Xi}_{\mathrm{HL}} = \boldsymbol{\Psi} \boldsymbol{\Upsilon} \boldsymbol{\Phi}^{\mathrm{T}}. \tag{3.1}$$

The choice of $\boldsymbol{\Gamma}$ is $\boldsymbol{\Psi}$.

The first idea is to mix all available data, high- and low-fidelity: $\tilde{\mathcal{D}}_{\mathrm{H}} = \mathcal{D}_{\mathrm{H}}$ and $\tilde{\mathcal{D}}_{\mathrm{L}} = \mathcal{D}_{\mathrm{L}}$. However, we typically have that $N_{\mathrm{L}} \gg N_{\mathrm{H}}$, so the basis is mainly built from the low-fidelity data. In addition, the small differences in the data between high- and low-fidelity code outputs that would be useful to build the basis have negligible impact because they are overwhelmed by the low-fidelity data. This method is not appropriate in our framework.

We have to choose between high- and low-fidelity. High-fidelity has the advantage of being closer to the desired result. However, it is also almost impossible to validate the chosen $\boldsymbol{\Gamma}$ because the high-fidelity data size N_{H} is small. The low-fidelity data set is larger; hence, the estimation of $\boldsymbol{\Gamma}$ is more robust. In order to choose $\boldsymbol{\Gamma}$, we therefore suggest to use the low-fidelity data and to calculate the SVD with $\tilde{\mathcal{D}}_{\mathrm{H}} = \emptyset$ and $\tilde{\mathcal{D}}_{\mathrm{L}} = \mathcal{D}_{\mathrm{L}}$.

3.2. Uniform distribution.

We can choose a random matrix $\boldsymbol{\Gamma}$ using the Haar measure on the orthogonal group $\mathcal{O}_{N_t}$, the group of $N_t \times N_t$ orthogonal matrices. This is the uniform orthogonal matrix law.

To generate a random matrix from the Haar measure over $\mathcal{O}_{N_t}$, one can first generate an $N_t \times N_t$ matrix with independent and identically distributed coefficients with the standard normal distribution, then apply the Gram--Schmidt process onto the matrix. As shown in [5], this generator produces a random orthogonal matrix with the uniform orthogonal matrix law. This method completely ignores the available data and is not appropriate in our framework.

3.3. CVB distribution.

The downside of the Dirac distribution is that the uncertainties on the basis estimation are not taken into account. A CVB method to assess the uncertainty estimation is therefore considered.

The proposed method uses only the low-fidelity data because it is assumed that there are too few high-fidelity data to implement this method, so $\tilde{\mathcal{D}}_{\mathrm{H}} = \emptyset$. For the construction of the basis, we try to have different sets to evaluate the basis in order to have empirical estimates of the moments of the basis vectors. Let k be a fixed integer in $\{1, \dots, N_{\mathrm{L}}\}$. Let $J = \{J_1, \dots, J_k\}$ be a random set of k elements in $\{1, \dots, N_{\mathrm{L}}\}$, with uniform distribution over the subsets of k elements in $\{1, \dots, N_{\mathrm{L}}\}$. The empirical distribution for the matrix $\boldsymbol{\Gamma}$ is defined as follows: For any bounded function $f : \mathcal{O}_{N_t} \to \mathbb{R}$,

$$\mathbb{E}[f(\boldsymbol{\Gamma})] = \frac{1}{\binom{N_{\mathrm{L}}}{k}} \sum_{\{j_1,\dots,j_k\} \subset \{1,\dots,N_{\mathrm{L}}\}} f(\boldsymbol{\Psi}_{[J_1,\dots,J_k]}), \tag{3.2}$$

where $\mathbf{U}_{[J_1,\ldots,J_k]}$ is the matrix of the left singular vectors of the SVD of $(z_L(\boldsymbol{x}^{(i)}, t_u))_{\substack{u \in \{1,\ldots,N_t\} \\ i \in \{1,\ldots,N_L\}\setminus\{J_1,\ldots,J_k\}}}$. This distribution depends on the choice of k, which will be discussed in section 7.

4. Multifidelity coefficients A_i. In this section, we want to build a surrogate model of a code $\alpha_H(\boldsymbol{x})$ whose input $\boldsymbol{x}$ is in $Q \subset \mathbb{R}^d$ and whose scalar output is in $\mathbb{R}$. The construction of a surrogate model for complex computer code is difficult because of the lack of available experimental outputs. We consider the situation in which a cheaper and approximate code $\alpha_L(\boldsymbol{x})$ is available. In this section, we apply the regression method presented by [14], reviewed in [8], and improved in [21].

We model the prior knowledge of the code output (α_L, α_H) as a GP (A_L, A_H). The vector containing the values of $\alpha_{\mathrm{F}}(\boldsymbol{x})$ at the points of the experimental design D_{F} are denoted by α^{F}, and A^{F} is the Gaussian vector containing $A_{\mathrm{F}}(\boldsymbol{x})$, $\boldsymbol{x} \in D_{\mathrm{F}}$. The combination of A^L and A^H is A. So is α, the combination of α^L and α^H. We present the recursive model of multifidelity introduced by [21]. The experimental design is constructed such that $D_H \subset D_L$. We assume that the low-fidelity code is computationally cheap and that we have access to a large experimental design for the low-fidelity code, i.e., $N_L \gg N_H$.

We consider the hierarchical model introduced by [21],

$$\text{(4.1)} \qquad \begin{cases} A_H(\boldsymbol{x}) = \rho_L(\boldsymbol{x})\tilde{A}_L(\boldsymbol{x}) + \delta(\boldsymbol{x}), \\ \tilde{A}_L(\boldsymbol{x}) \perp \delta(\boldsymbol{x}), \\ \rho_L(\boldsymbol{x}) = g^T(\boldsymbol{x})\beta_\rho, \end{cases}$$

where $\perp$ means independence, ^{T} stands for the transpose,

$$\text{(4.2)} \qquad [\delta(\boldsymbol{x})\,|\,\beta_H, \sigma_H] \sim \mathcal{GP}\left(f^T(\boldsymbol{x})\beta_H, \sigma_H^2 r_H(\boldsymbol{x}, \boldsymbol{x}^r)\right),$$

and $\tilde{A}_L(\boldsymbol{x})$ is a GP conditioned by the values α^L. Its distribution is the one of $[A_L(\boldsymbol{x})\,|\,A^L = \alpha^L, \beta_L, \sigma_L]$ with

$$\text{(4.3)} \qquad [A_L(\cdot)\,|\,\beta_L, \sigma_L] \sim \mathcal{GP}\left(f^T(\boldsymbol{x})\beta_L, \sigma_L^2 r_L(\cdot, \boldsymbol{x}^r)\right).$$

Therefore, the distribution of $\tilde{A}_L(\boldsymbol{x})$ is Gaussian with mean $\mu_{\tilde{A}_L}(\boldsymbol{x})$ and variance $\sigma^2_{\tilde{A}_L}(\boldsymbol{x})$:

$$\text{(4.4)} \qquad \mu_{\tilde{A}_L}(\boldsymbol{x}) = f_L^T(\boldsymbol{x})\beta_L + r_L^T(\boldsymbol{x})C_L^{-1}\left(\alpha^L - F_L\beta_L\right),$$

$$\text{(4.5)} \qquad \sigma^2_{\tilde{A}_L}(\boldsymbol{x}) = \sigma_L^2\left(r_L(\boldsymbol{x}, \boldsymbol{x}) - r_L^T(\boldsymbol{x})C_L^{-1}r_L(\boldsymbol{x})\right).$$

Here

- $\mathcal{GP}$ means a GP;
- $g_L(\boldsymbol{x})$ is a vector of q_L regression functions;
- $f_{\mathrm{F}}(\boldsymbol{x})$ are vectors of p_{F} regression functions;
- $r_{\mathrm{F}}(\boldsymbol{x}, \boldsymbol{x}^r)$ are correlation functions;

- β_{F} are p_{F}-dimensional vectors;
- σ^2_{F} are positive real numbers;
- β_ρ is a q-dimensional vector of adjustment parameters;
- $C_{\mathrm{F}} = (r_{\mathrm{F}}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}))_{i,j=1}^{N_{\mathrm{F}}}$ is the $N_{\mathrm{F}} \times N_{\mathrm{F}}$ correlation matrix of A^{F};
- $r_{\mathrm{F}}(\mathbf{x}) = (r_{\mathrm{F}}(\mathbf{x}, \mathbf{x}^{(i)}))_{i=1}^{N_{\mathrm{F}}}$ is the N_{F}-dimensional vector of correlations between $A_{\mathrm{F}}(\mathbf{x})$ and A^{F};
- F_{F} is the $N_{\mathrm{F}} \times p_{\mathrm{F}}$ matrix containing the values of $f^{\mathrm{T}}(\mathbf{x})$ for $\mathbf{x} \in \mathcal{D}_{\mathrm{F}}$.

For $\mathbf{x} \in Q$, the conditional distribution of $A_{\mathrm{H}}(\mathbf{x})$ is

$$\Big[A_{\mathrm{H}}(\mathbf{x}) \,\Big|\, A = \alpha, \beta, \beta_\rho, \sigma^2\Big] \sim \mathrm{N}\Big(\mu_{A_{\mathrm{H}}}(\mathbf{x}), \sigma^2_{A_{\mathrm{H}}}(\mathbf{x})\Big), \tag{4.6}$$

where $\beta = (\beta_{\mathrm{H}}^{\mathrm{T}}, \beta_{\mathrm{L}}^{\mathrm{T}})^{\mathrm{T}}$ is the $p_{\mathrm{H}} + p_{\mathrm{L}}$-dimensional vector of regression parameters, $\sigma^2 = (\sigma^2_{\mathrm{L}}, \sigma^2_{\mathrm{H}})$ are the variance parameters

$$\begin{aligned}\mu_{A_{\mathrm{H}}}(\mathbf{x}) = {} & g^{\mathrm{T}}(\mathbf{x})\beta_\rho \mu_{\tilde{A}_{\mathrm{L}}}(\mathbf{x}) + f^{\mathrm{T}}(\mathbf{x})\beta_{\mathrm{H}} \\ & + r^{\mathrm{T}}_{\mathrm{H}}(\mathbf{x}) C^{-1}_{\mathrm{H}} \Big(\alpha_{\mathrm{H}} - \rho_{\mathrm{L}}(D_{\mathrm{H}}) \odot \alpha_{\mathrm{L}}(D_{\mathrm{H}}) - F_{\mathrm{H}}\beta_{\mathrm{H}}\Big),\end{aligned} \tag{4.7}$$

and

$$\sigma^2_{A_{\mathrm{H}}}(\mathbf{x}) = \Big(g^{\mathrm{T}}(\mathbf{x})\beta_\rho - r^{\mathrm{T}}_{\mathrm{H}}(\mathbf{x}) C^{-1}_{\mathrm{H}} F_{\mathrm{H}}\Big)^2 \big(F_{\mathrm{L}}^{\mathrm{T}} C_{\mathrm{H}}^{-1} F_{\mathrm{H}}\big)^{-1} + \sigma^2_{\mathrm{L}} g^{\mathrm{T}}(\mathbf{x}) C_{\mathrm{L}} g(\mathbf{x}) + \sigma^2_{\mathrm{H}} \Big(1 - r^{\mathrm{T}}_{\mathrm{H}}(\mathbf{x}) C^{-1}_{\mathrm{H}} r_{\mathrm{H}}(\mathbf{x})\Big). \tag{4.8}$$

The notation $\odot$ is the element by element matrix product, $\rho_{\mathrm{L}}(D_{\mathrm{H}})$ is the N_{H}-dimensional vector containing the values of $\rho_{\mathrm{L}}(\mathbf{x})$ for $\mathbf{x} \in D_{\mathrm{H}}$, and $\alpha_{\mathrm{L}}(D_{\mathrm{H}})$ is the N_{H}-dimensional vector containing the values of $\alpha_{\mathrm{L}}(\mathbf{x})$ at the points of D_{H}.

The prior distributions of the parameters β and σ are given in Appendix A. The hyperparameters of the covariance kernels r_{L} and r_{H} can be estimated by maximum likelihood or by leave-one-out (LOO) cross validation [2]. The nested property of the experimental design sets $D_{\mathrm{H}} \subset D_{\mathrm{L}}$ is not necessary to build the model, but it is simpler to estimate the parameters with this assumption [36]. Moreover, the ranking of codes and the low computer cost of the low-fidelity code allow for a nested design for practical applications.

5. The orthogonal part $Z^{\perp}_{\mathrm{H}}$. In this subsection, we address GP regression for a simple-fidelity code with time-series output. For the calculation of surrogate models with functional outputs, there are two different techniques. The simplest ones are dimension reduction techniques as presented in [1, 25]. An alternative is presented here; this method is GP regression with covariance tensorization. The method is presented in [30] and the estimation of the hyperparameters is from [28].

In this section and the following ones, we consider that the output is a time-dependent function observed on a fixed time grid $\{t_u\}_{u=1,\dots,N_t}$, with $N_t \gg 1$, which is called a time series.

The experimental design in a times-series output case is very different from a scalar output case. In particular, for a value $\mathbf{x}$ in the experimental design D, all the t_u for $u = 1, \dots, N_t$ are in the experimental design. The $N_x \times N_t$ matrix containing the observations is $\mathbf{Z}_{\mathrm{obs}} = (z(\mathbf{x}^{(i)}, t_u))_{\substack{u=1,\dots,N_t \\ i=1,\dots,N_x}}$. In GP regression, we model the prior knowledge of the code output as a GP $Z(\mathbf{x}, t_u)$ with $\mathbf{x} \in Q$ and $u = 1, \dots, N_t$ with a covariance function C given by (5.2) and a mean function μ given by (5.1). We focus our attention to the case $N_t > N_x$. We assume

that the covariance structure can be decomposed into two different functions representing the correlation in $\boldsymbol{x}$ and the correlation in t. If we choose well both functions, the kriging calculation is possible [28, 30].

In the following, we present a simplification of the method proposed in [28]. The a priori $\mathbb{R}^{N_t}$-valued mean function is assumed to be of the form

$$\mu(\boldsymbol{x}) = Bf(\boldsymbol{x}), \tag{5.1}$$

where $f(\boldsymbol{x})$ is a given $\mathbb{R}^M$-valued function and $B \in \mathcal{M}_{N_t \times M}(\mathbb{R})$ is to be estimated. We define by F the $N_x \times M$ matrix $[f^T(\boldsymbol{x}^{(i)})]_{i=1,\dots,N_x}$.

The a priori covariance function $C(t_u, t_{u'}, \boldsymbol{x}, \boldsymbol{x}')$ can be expressed with the $N_t \times N_t$ matrix R_t and the correlation function $C_x : Q \times Q \to [0, 1]$ with $C_x(\boldsymbol{x}, \boldsymbol{x}) = 1$:

$$C(t_u, t_{u'}, \boldsymbol{x}, \boldsymbol{x}') = R_t(t_u, t_{u'}) C_x(\boldsymbol{x}, \boldsymbol{x}'). \tag{5.2}$$

The covariance in time is expressed as a matrix because the temporal grid is finite and fixed. The covariance ``matrix'' (here a tensor) of $(Z(\boldsymbol{x}^{(j)}, t_u))_{\substack{u=1,\dots,N_t \\ j=1,\dots,N_x}}$ is

$$R = R_t \otimes R_x, \tag{5.3}$$

with $(R_x)_{k,l} = C_x(\boldsymbol{x}^{(k)}, \boldsymbol{x}^{(l)})$ $k, l = 1, \dots, N_x$. $\otimes$ is the Kronecker product described in [33, section 4.5.5].

If R_x and R_t are not singular, then the a posteriori distribution of the $\mathbb{R}^{N_t}$-valued process Z given the covariance functions and the observations $\boldsymbol{Z}$ is Gaussian,

$$(Z(\boldsymbol{x}, t_u))_{u=1,\dots,N} \mid R_t, C_x, \boldsymbol{Z} \sim \mathcal{GP}(\mu_\Lambda(\boldsymbol{x}), R_\Lambda(\boldsymbol{x}, \boldsymbol{x}') R_t), \tag{5.4}$$

with the N_t-dimensional posterior mean,

$$\mu_\Lambda(\boldsymbol{x}) = \boldsymbol{Z} \mathbf{R}^{-1} r_x(\boldsymbol{x}) + B u(\boldsymbol{x}), \tag{5.5}$$

where $r_x(\boldsymbol{x})$ is the N_x-dimensional vector $(C_x(\boldsymbol{x}, \boldsymbol{x}^{(j)}))_{j=1,\dots,N_x}$. The posterior covariance function $R_\Lambda(\boldsymbol{x}, \boldsymbol{x}')$ is

$$R_\Lambda(\boldsymbol{x}, \boldsymbol{x}') = c_\Lambda(\boldsymbol{x}, \boldsymbol{x}')(1 + v_\Lambda(\boldsymbol{x}, \boldsymbol{x}')). \tag{5.6}$$

The functions that are used in the regression are

$$\begin{cases} u(\boldsymbol{x}) = f(\boldsymbol{x}) - F^T R_x^{-1} r_x(\boldsymbol{x}), \\ c_\Lambda(\boldsymbol{x}, \boldsymbol{x}') = C_x(\boldsymbol{x}, \boldsymbol{x}') - r_x(\boldsymbol{x})^T R_x^{-1} r_x(\boldsymbol{x}'), \\ v_\Lambda(\boldsymbol{x}, \boldsymbol{x}') = u(\boldsymbol{x})^T (F^T R_x^{-1} F)^{-1} u(\boldsymbol{x}') c^{-1}(\boldsymbol{x}, \boldsymbol{x}'), \end{cases} \tag{5.7}$$

and

$$B_\Lambda = \boldsymbol{Z} R_x^{-1} F (F^T R_x^{-1} F)^{-1}. \tag{5.8}$$

The correlation function C_x is assumed to be a Matérn $\frac{5}{2}$ kernel with a tensorized form (see [34, Chapter 4]),

$$C_x(\mathbf{x}, \mathbf{x}') = \prod_{i=1}^{d} \left(1 + \frac{\sqrt{5}|x_i - x_i'|}{\ell_{x_i}} + \frac{5|x_i - x_i'|^2}{3\ell_{x_i}^2}\right) \exp\left(-\frac{\sqrt{5}|x_i - x_i'|}{\ell_{x_i}}\right), \tag{5.9}$$

with $\boldsymbol{\ell}_{\mathbf{x}} = (\ell_{x_1}, \ldots, \ell_{x_d})$ the vector of correlation lengths. Other choices are of course possible. R_t is estimated using R_x^{-1} and the observations $\mathbf{Z}$ by maximum likelihood, as in [28],

$$\hat{R}_t = \frac{1}{N_x} \left(\mathbf{Z} - \hat{\mathbf{b}}\right) R_x^{-1} \left(\mathbf{Z} - \hat{\mathbf{b}}\right)^T, \tag{5.10}$$

where $\hat{\mathbf{b}}$ is the $N_t \times N_x$ matrix of empirical means $\hat{b}_{u,i} = \frac{1}{N_x} \sum_{j=1}^{N_x} (\mathbf{Z})_{u,j}$ $\forall i = 1, \ldots, N_x$ and $u = 1, \ldots, N_t$.

It remains only to estimate the vector of correlation lengths $\boldsymbol{\ell}_{\mathbf{x}} = (\ell_{x_1}, \ldots, \ell_{x_d})$ to determine the function C_x. As presented in [28], the maximum likelihood estimation is not well defined for $\boldsymbol{\ell}_{\mathbf{x}}$. Indeed the approximation of R_t by (5.10) is singular because $N_x < N_t$. In fact, we do not need to invert R_t, as seen in (5.4)--(5.8). The method generally used to estimate the correlation lengths is cross validation and, in our case, LOO. The LOO mean square error that needs to be minimized is

$$\varepsilon^2(\boldsymbol{\ell}_{\mathbf{x}}) = \sum_{k=1}^{N_x} \left\| \mu_{\mathcal{N}}^{(-k)}(\mathbf{x}^{(k)} | \mathbf{Z}^{(-k)}, \boldsymbol{\ell}_{\mathbf{x}}) - \mathbf{Z}_{(\cdot)}(\mathbf{x}^{(k)}) \right\|^2, \tag{5.11}$$

where $\hat{\mu}^{(-k)}(\mathbf{x}^{(k)} | \mathbf{Z}^{(-k)}, \boldsymbol{\ell}_{\mathbf{x}})$ is the $\mathbb{R}^{N_t}$-valued prediction mean obtained with the correlation length vector $\boldsymbol{\ell}_{\mathbf{x}}$, using all observations except the kth, at the point $\mathbf{x}^{(k)}$ and $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^{N_t}$. We can use an expression of $\varepsilon^2(\boldsymbol{\ell}_{\mathbf{x}})$ that does not require multiple regression, as in [2, 6]. For more detail, see Appendix B.

6. Posterior law of the multifidelity model.

This section is divided into two parts. The first part presents truncation in which $Z_H^{\perp}$ is neglected. The second part presents the computation without neglecting $Z_H^{\perp}$

6.1. With truncation.

The goal of this section is to calculate the posterior distribution of $\mathcal{Z}_H(\mathbf{x}, t_u)$, where $\mathcal{Z}_H^{\perp}$ is null. The equations are computed with $N = N_t$, and a discussion about the value of N is proposed in subsection 6.1.3. The problem can be split into two parts: the multifidelity regression of the basis coefficients knowing $\boldsymbol{\gamma}$ and the integration with respect to the distribution of $\boldsymbol{\gamma}$. The Dirac and CVB distributions described in section 3 can be used to define the law of $\boldsymbol{\gamma}$.

Multifidelity surrogate modeling of the coefficients. By applying the model proposed in section 4, we can therefore deduce the prediction mean and variance. Their expressions are given in Appendix D.1.

6.1.1. Dirac law of $\boldsymbol{\gamma}$.

Here we assume that the law of $\boldsymbol{\gamma}$ is Dirac at $\boldsymbol{\gamma}^*$. Consequently, the posterior distribution of $\mathcal{Z}_H(\mathbf{x}, t)$ is Gaussian. In order to characterize the law of $\mathcal{Z}_H(\mathbf{x}, t_u)$, it is necessary and sufficient to compute its mean and variance.

Mean. The posterior mean is

$$\mathbb{E}[Z_H(\boldsymbol{x}, t_u)|A=\alpha] = \sum_{i=1}^{N_t} \gamma_i(t_u)\,\mathbb{E}[A_{i,H}(\boldsymbol{x})|A=\alpha], \tag{6.1}$$

where the expectation $\mathbb{E}[A_{i,H}(\boldsymbol{x})|A=\alpha]$ is given by (D.1).

ariance. The posterior variance is

$$\mathbb{V}[Z_H(\boldsymbol{x}, t_u)|A=\alpha] = \sum_{i=1}^{N_t} \gamma_i^2(t_u)\,\mathbb{V}[A_{i,H}(\boldsymbol{x})|A=\alpha], \tag{6.2}$$

where the variance $\mathbb{V}[A_{i,H}(\boldsymbol{x})|A=\alpha]$ is given by (D.2).

6.1.2. CVB law of $\boldsymbol{\Gamma}$. Because the law is different from Dirac, the posterior distribution of $Z_H(\boldsymbol{x}, t)$ is not Gaussian anymore. However, we can characterize the posterior mean and the variance of $Z_H(\boldsymbol{x}, t_u)$.

We denote $\mathbb{E}_\alpha[\cdot] = \mathbb{E}[\cdot|A=\alpha]$, $\mathbb{V}_\alpha[\cdot] = \mathbb{V}[\cdot|A=\alpha]$, $\mathbb{E}_{\boldsymbol{\Gamma}|\mathcal{Z}}[\cdot] = \mathbb{E}[\cdot||\mathcal{Z}=\boldsymbol{\gamma}]$, and $\mathbb{V}_{\boldsymbol{\Gamma}|\mathcal{Z}}[\cdot] = \mathbb{V}[\cdot||\mathcal{Z}=\boldsymbol{\gamma}]$.

Mean. The linearity of the expectation and the law of total expectation give

$$\mathbb{E}_\alpha[Z_H(\boldsymbol{x}, t_u)] = \sum_{i=1}^{N_t} \mathbb{E}_\alpha[\Gamma_i(t_u)]\,\mathbb{E}_\alpha[A_{i,H}(\boldsymbol{x})|\boldsymbol{\Gamma}], \tag{6.3}$$

where the expectation $\mathbb{E}_\alpha[A_{i,H}(\boldsymbol{x})|\boldsymbol{\Gamma}]$ is given by (D.1).

Variance. The law of total variance gives

$$\mathbb{V}_\alpha[Z_H(\boldsymbol{x}, t_u)] = \mathbb{V}_\alpha[\mathbb{E}_\alpha[Z_H(\boldsymbol{x}, t_u)|\boldsymbol{\Gamma}]] + \mathbb{E}_\alpha[\mathbb{V}_\alpha[Z_H(\boldsymbol{x}, t_u)|\boldsymbol{\Gamma}]]. \tag{6.4}$$

By Appendix D.2, we get

$$\begin{aligned}\mathbb{V}_\alpha[Z_H(\boldsymbol{x}, t_u)] = &\sum_{i=1}^{N_t} \mathbb{V}_\alpha[\Gamma_i(t_u)]\,\mathbb{E}_\alpha[A_{i,H}(\boldsymbol{x})|\boldsymbol{\Gamma}]] \\ &+ \sum_{i,j=1, i\neq j}^{N_t} \mathrm{Cov}_\alpha(\Gamma_i(t_u)\,\mathbb{E}_\alpha[A_{i,H}(\boldsymbol{x})|\boldsymbol{\Gamma}], \Gamma_j(t_u)\,\mathbb{E}_\alpha[A_{j,H}(\boldsymbol{x})|\boldsymbol{\Gamma}]) \\ &+ \sum_{i=1}^{N_t} \mathbb{E}_\alpha\Big[\Gamma_i^2(t_u)\,\mathbb{V}_\alpha[A_{i,H}(\boldsymbol{x})|\boldsymbol{\Gamma}]\Big]\end{aligned} \tag{6.5}$$

Equations (6.3) and (6.5) are combinations of expectations of explicit functions of $\boldsymbol{\Gamma}$. We can compute the result using our knowledge of the law of $\boldsymbol{\Gamma}$. The expectation of a function of $\boldsymbol{\Gamma}$ is given by (3.2).

6.1.3. Truncation. There is a problem with the surrogate modeling of the coefficients of the decomposition with indices larger than N_L. Indeed, we typically have $N_L < N_t$, so the vectors Γ_i with indices larger than N_L of the basis are randomly constructed, which is not suitable for building surrogate models. To solve this problem, it is possible to truncate the sum. Only the first N coefficients, with $N \leq N_L$, are calculated. This would be reasonable if the contributions of the terms $A_{i,H}(\boldsymbol{x})\Gamma_i(t_u)$ for $i > N$ were negligible. However, it turns out that these terms are often not collectively negligible (see section 7), and the truncation method does not achieve a good bias-variance trade-off even when optimizing with respect to N (e.g., by a cross-validation procedure). The high- and low-fidelity outputs do not necessarily have the same forms. Thus, it is possible that an important part of the high-fidelity code is neglected because it is not taken into account by the subspace spanned by $\{\Gamma_i\}_{i\leq N}$. We will therefore propose in the next section an original method to tackle this problem.

6.2. Without truncation. In this section, we first make a quick reminder of the methods presented in section 5. Moreover, with different assumptions about the law of $\boldsymbol{\Gamma}$, we present the regression using the model and the data.

Multifidelity of coefficients. As in subsection 6.1, we compute the N multifidelity models of the first N coefficients of the expansion of the code output given $\boldsymbol{\Gamma}$. If we apply the method proposed in section 4, we can therefore deduce the prediction mean and variance, as in subsection 6.1.

Tensorized covariance regression. The orthogonal part of the regression is computed using the method presented in section 5. The adaptation is that the regression must be carried out in subspace $S_N^{\perp}$ given $\boldsymbol{\Gamma}$,

$$\boldsymbol{Z}^{\perp}_{\mathcal{T}_N^{\perp}} : \boldsymbol{x}, \boldsymbol{\Gamma} \mapsto \boldsymbol{Z}_{\mathcal{T}_N^{\perp}} - \sum_{i=1}^{N} \alpha_{i,H}(\boldsymbol{x}, \boldsymbol{\Gamma}) \Big(\sum_{u=1,\cdots,N_t, \boldsymbol{x} \in \mathcal{D}_t} \Big) \tag{6.6}$$

where $\alpha_{i,H}$ is given by (2.3).

This does not have any consequence on the $\boldsymbol{x}$ part but only on the t part. Contrarily to $Z_H^{||}(\boldsymbol{x}, t_u)$, only one surrogate model is needed for $Z_H^{\perp}(\boldsymbol{x}, t_u)$. The detail of how we can deal with $Z_H^{\perp}(\boldsymbol{x}, t_u)$ and $Z_H^{||}(\boldsymbol{x}, t_u)$ is explained in Appendix C.

6.2.1. Dirac law of $\boldsymbol{\Gamma}$. Here we assume that $\boldsymbol{\Gamma}$ is known and its distribution is Dirac at $\boldsymbol{\gamma}$. Consequently, $Z_H(\boldsymbol{x}, t)$ is a GP by linear combination of independent GPs. Its posterior distribution is completely determined if we can evaluate its mean and covariance.

Mean. The $\Gamma_i(t_u)$'s are constant and equal to $\gamma_i(t_u)$. Consequently,

$$\mathbb{E}_{\alpha}[Z_H(\boldsymbol{x}, t_u) | N] = \sum_{i=1}^{N} \mathbb{E}_{\alpha}[A_{i,H}(\boldsymbol{x})] \gamma_i(t_u) \tag{6.7}$$

and

$$\mathbb{E}_{\mathcal{T}_N^{\perp}}[Z_H(\boldsymbol{x}, t_u) | N, \boldsymbol{\ell}_{\boldsymbol{x}}] = \sum_{i=1}^{N} \mathbb{E}_{\alpha}[A_{i,H}(\boldsymbol{x})] \Gamma_i(t_u) + \mathbb{E}_{\mathcal{T}_N^{\perp}}\Big[Z_H^{\perp}(\boldsymbol{x}, t_u) | N, \boldsymbol{\ell}_{\boldsymbol{x}} \Big], \tag{6.8}$$

where $\mathbb{E}_{\alpha}[A_{i,H}(\boldsymbol{x})]$ is given by (D.1) and $\mathbb{E}_{\mathcal{T}_N^{\perp}}[Z_H^{\perp}(\boldsymbol{x}, t_u) | N, \boldsymbol{\ell}_{\boldsymbol{x}}]$ by (5.5).

Variance. The formula of the variance is

$$\mathbb{V}_{\alpha}[A_{i,H}(\boldsymbol{x}) \Gamma_i(t)] = \mathbb{V}_{\alpha}[A_{i,H}(\boldsymbol{x})] \gamma_i(t)^2. \tag{6.9}$$

The uncorrelation of the coefficients $\boldsymbol{A}_{i,H}(\boldsymbol{x})$ gives $\mathrm{Cov}_{\alpha}[A_{i,H}(\boldsymbol{x}), \boldsymbol{A}_{j,H}(\boldsymbol{x})] = 0$ for $i = j$ and $\mathrm{Cov}_{\alpha}[A_{i,H}(\boldsymbol{x}) \gamma_i(t_u), Z^{\perp}(\boldsymbol{x}, t_u)] = 0$. The expression of the variance becomes simple,

$$\mathbb{V}_{\mathcal{T}_N^{\perp}}[Z_H(\boldsymbol{x}, t_u) | N, \boldsymbol{\ell}_{\boldsymbol{x}}] = \sum_{i=1}^{N} \mathbb{V}_{\alpha}[A_{i,H}(\boldsymbol{x})] \gamma_i(t_u)^2 + \mathbb{V}_{\mathcal{T}_N^{\perp}}\Big[Z_H^{\perp}(\boldsymbol{x}, t_u) | N, \boldsymbol{\ell}_{\boldsymbol{x}} \Big], \tag{6.10}$$

where $\mathbb{V}_{\alpha}[A_{i,H}(\boldsymbol{x})]$ is given by (D.2) and $\mathbb{V}_{\mathcal{T}_N^{\perp}}[Z_H^{\perp}(\boldsymbol{x}, t_u) | N, \boldsymbol{\ell}_{\boldsymbol{x}}]$ is given by (5.4).

6.2.2. VB law of $\boldsymbol{\Gamma}$. The posterior distribution of $Z_H(\boldsymbol{x}, t)$ is not Gaussian anymore. However, to predict the output of the high-fidelity code and to quantify the prediction uncertainty, we are able to compute the posterior mean and variance of $Z_H(\boldsymbol{x}, t)$.

Mean. We can decompose the process into two parts:

$$Z_H(\boldsymbol{x}, t_u) = Z_H(\boldsymbol{x}, t_u) + Z^{\perp}(\boldsymbol{x}, t_u). \tag{6.11}$$

The linearity of the expectation gives us

$$\mathbb{E}_{Z|N,\boldsymbol{\ell}}[Z_H(\boldsymbol{x}, t_u)|N, \boldsymbol{\ell}] = \sum_{i=1}^{N_H} \mathbb{E}_{Z|N,\boldsymbol{\ell}}[A_{i,H}(\boldsymbol{x})\Gamma_i(t_u)] + \mathbb{E}_{Z|N,\boldsymbol{\ell}}\left[Z_H^{\perp}(\boldsymbol{x}, t)|N, \boldsymbol{\ell}\right]. \tag{6.12}$$

The theorem of total expectation gives us

$$\mathbb{E}_{Z|N}\left[Z_H(\boldsymbol{x}, t_u)|N\right] = \sum_{i=1}^{N_H} \mathbb{E}_{Z|\alpha,\boldsymbol{\gamma}}[\Gamma_i(t_u)]\mathbb{E}_{Z|\alpha,\boldsymbol{\gamma}}[A_{i,H}(\boldsymbol{x})|\boldsymbol{\gamma}] \tag{6.13}$$

and therefore

$$\begin{aligned}\mathbb{E}_{Z|\alpha,\boldsymbol{\gamma},N,\boldsymbol{\ell}}[Z_H(\boldsymbol{x}, t_u)|N, \boldsymbol{\ell}] &= \sum_{i=1}^{N_H} \mathbb{E}_{Z|\alpha,\boldsymbol{\gamma},N,\boldsymbol{\ell}}[\Gamma_i(t_u)]\mathbb{E}_{Z|\alpha,\boldsymbol{\gamma},N,\boldsymbol{\ell}}[A_{i,H}(\boldsymbol{x})|\boldsymbol{\gamma}] \\ &\quad + \mathbb{E}_{Z|N,\boldsymbol{\ell}}\left[\mathbb{E}_{Z|N}^{\perp}\left[Z_H^{\perp}(\boldsymbol{x}, t_u)|N, \boldsymbol{\ell}, \boldsymbol{\Gamma}\right]|N, \boldsymbol{\ell}\right],\end{aligned} \tag{6.14}$$

where $\mathbb{E}_{\alpha}[A_{i,H}(\boldsymbol{x})|\boldsymbol{\gamma}]$ is given by (D.1) and $\mathbb{V}_{\alpha}[Z_H^{\perp}(\boldsymbol{x})|\boldsymbol{\gamma}]$ is given by (5.5). Equation (6.14) is a combination of expectations of explicit functions of $\boldsymbol{\gamma}$, which can be computed by (3.2).

Variance. The theorem of the total variance gives us

$$\begin{aligned}\mathbb{V}_{Z|\alpha,\boldsymbol{\gamma},N,\boldsymbol{\ell}}[Z_H(\boldsymbol{x}, t_u)|N, \boldsymbol{\ell}] &= \mathbb{V}_{Z|\alpha,\boldsymbol{\gamma},N,\boldsymbol{\ell}}[\mathbb{E}_{Z|\alpha,\boldsymbol{\gamma},N,\boldsymbol{\ell}}[Z_H(\boldsymbol{x}, t_u)|\boldsymbol{\gamma}, N, \boldsymbol{\ell}]|N, \boldsymbol{\ell}] \\ &\quad + \mathbb{E}_{Z|\alpha,\boldsymbol{\gamma},N,\boldsymbol{\ell}}[\mathbb{V}_{Z|\alpha,\boldsymbol{\gamma},N,\boldsymbol{\ell}}[Z_H(\boldsymbol{x}, t_u)|\boldsymbol{\gamma}, N, \boldsymbol{\ell}]|N, \boldsymbol{\ell}].\end{aligned} \tag{6.15}$$

By Appendix D.3, we get

$$\begin{aligned}\mathbb{V}_{Z|N,\boldsymbol{\ell}}[Z_H(\boldsymbol{x}, t_u)|N, \boldsymbol{\ell}] &= \mathbb{V}_{Z|N,\boldsymbol{\ell}}\left[\mathbb{E}_{Z|N,\boldsymbol{\ell}}\left[Z_H^{\perp}(\boldsymbol{x}, t_u)|\boldsymbol{\Gamma}, N, \boldsymbol{\ell}\right]|N, \boldsymbol{\ell}\right] \\ &\quad + \mathbb{E}_{Z|N,\boldsymbol{\ell}}\left[\mathbb{V}_{\alpha}\left[Z_H(\boldsymbol{x}, t_u)|\boldsymbol{\gamma}, N, \boldsymbol{\ell}\right]|N, \boldsymbol{\ell}\right] \\ &\quad + \sum_{i=1}^{N_H} \mathbb{V}_{Z|N,\boldsymbol{\ell}}[\Gamma_i(t_u)]\mathbb{V}_{\alpha}[A_{i,H}(\boldsymbol{x})|\boldsymbol{\gamma}] \\ &\quad + \sum_{i=1}^{N_H} \mathbb{E}_{Z|N,\boldsymbol{\ell}}[\Gamma_i(t_u)^2]\mathbb{V}_{\alpha}[A_{i,H}(\boldsymbol{x})|\boldsymbol{\gamma}] \\ &\quad + \sum_{i,j=1;i\neq j}^{N_H} \mathrm{Cov}_{Z|N,\boldsymbol{\ell}}[\Gamma_i(t_u)]\mathbb{E}_{\alpha}[A_{i,H}(\boldsymbol{x})|\boldsymbol{\gamma}], \Gamma_j(t_u)]\mathbb{E}_{\alpha}[A_{j,H}(\boldsymbol{x})|\boldsymbol{\gamma}]] \\ &\quad + 2\sum_{i=1}^{N_H} \mathrm{Cov}_{Z|N,\boldsymbol{\ell}}\left[\Gamma_i(t_u)]\mathbb{E}_{\alpha}[A_{i,H}(\boldsymbol{x})|\boldsymbol{\gamma}], \mathbb{E}_{Z|N,\boldsymbol{\ell}}\left[Z_H^{\perp}(\boldsymbol{x}, t_u)|\boldsymbol{\Gamma}, N, \boldsymbol{\ell}\right]|N, \boldsymbol{\ell}\right],\end{aligned} \tag{6.16}$$

where $\mathbb{V}_{\alpha}[A_{i,H}(\boldsymbol{x})|\boldsymbol{\gamma}]$ is given by (D.2) and $\mathbb{V}_{\alpha}[Z_H^{\perp}(\boldsymbol{x})|\boldsymbol{\gamma}]$ is given by (5.6). Equation (6.16) is a combination of expectations and variances of explicit functions of $\boldsymbol{\gamma}$, which can be computed by (3.2).

6.2.3. Effective dimension. For the formulas in subsection 6.2 to be valid, N must be fixed. We may choose N by knowledge of the physical system or of the code, but it is impossible in most cases due to the high-/low-fidelity differences. The best solution is generally to determine N by a K-fold cross-validation procedure.

The criterion that we choose to maximize is

$$Q^2_N(t_u) = 1 - \frac{\sum_{k=1}^{N_H} \left[z_H(\boldsymbol{x}^{(k)}, t_u) - \mathbb{E}_{Z_H}[Z_H(\boldsymbol{x}^{(k)}, t_u)|\boldsymbol{\gamma}, N, \boldsymbol{\ell}, \boldsymbol{x}^{(-k)}]\right]^2}{N_H \mathbb{V}[z_H(D_H, t_u)]}, \tag{6.17}$$

Table 1
Distributions of the input variables.

M_S	k	θ_0	$\dot{\theta}_0$	y_0
$\mathcal{U}(10, 12)$	$\mathcal{U}(1; 1.4)$	$\mathcal{U}(\frac{\pi}{4}; \frac{\pi}{3})$	$\mathcal{U}(0; \frac{1}{10})$	$\mathcal{U}(0; 0.2)$

where $\mathbb{V}[z_H(D_H, t_u)]$ is the empirical variance of the observed values:

$$\mathbb{V}[z_H(D_H, t_u)] = \frac{1}{N_H}\sum_{k=1}^{N_H} z_H(\boldsymbol{x}^{(k)}, t_u)^2 - \left(\frac{1}{N_H}\sum_{k=1}^{N_H} z_H(\boldsymbol{x}^{(k)}, t_u)\right)^2.$$

The procedure we propose starts with the dimension 0. For the case $N = 0$, the surrogate model depends only on high-fidelity regression:

- We compute the surrogate model for all $N = 0, \ldots, N_L$.
- We calculate the mean in t_u of $Q_N^2(t_u)$:

$$\hat{Q}^2_N = \frac{1}{N_t}\sum_{u=1}^{N_t} Q_N^2(t_u).$$

We compare the $\hat{Q}^2_N$ values and the value N with the largest $\hat{Q}^2_N$ chosen. In order to evaluate the surrogate model in the next section, we compute $\hat{Q}^2 = \max_N \hat{Q}^2_N$.

7. Illustration: Double-pendulum simulator. The purpose of this section is to apply the methods proposed in the previous sections to a mechanical example. The example is based on a simulator of a pendulum attached to a spring-mass system. We have two codes: The high-fidelity code numerically solves Newton's equation, and low-fidelity code simplifies the equation by linearization for small angles of the pendulum motion and solves the system.

7.1. Characteristics of the outputs.

The physical system. The system can be seen as a dual-oscillator cluster. The first oscillator is a spring-mass system whose axis is perpendicular to the gravitational axis. The parameters of this system are the mass of the system M_S and the spring stiffness k. The initial position of the mass is denoted y_0, and its initial velocity is 0. The second oscillator is a pendulum. A schematic representation of the system is presented in Figure 1. The parameters are the mass m and the length of the pendulum ℓ, which are fixed. The initial value of the angle is θ_0, and its derivative is $\dot{\theta}_0$. By Newton's law of motion, the dynamics is governed by a system of two coupled ODEs. However, we do not have a closed-form expression that gives the solution of the system. This forces us to use computer codes. The output signal is the position of the mass m at time $t \in \{t_1, \ldots, t_{N_t}\}$ with $N_t = 101$. The input vector is $\boldsymbol{x} = \{M_S, k, y_0, \theta_0, \dot{\theta}_0\}$. The input variables are assumed to be independent and identically distributed with uniform distributions as described in Table 1.

The two different code levels. We propose two codes. The high-fidelity code numerically solves the coupled system of ODEs by an Euler's derivation of the position y and the angle θ for each t_u. This gives functions $\theta(t_u)$ and $y(t_u)$. The low-fidelity code assumes that the

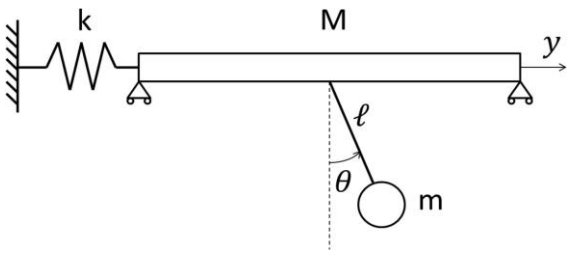

Figure 1. *The double-pendulum system with its parameters.*

angle of the \theta pendulum is small so that the linearization of $\sin(\theta)$ makes it possible to get a simpler form of the expression of the two coupled ODEs and a faster resolution.

Code analysis. A sensitivity analysis is carried out for information purposes, but it is not used in the forthcoming surrogate modeling. The sensitivity analysis makes it possible to determine the effective dimension of our problem. We compare outputs of the high- and low-fidelity codes and the associated Sobol indices on Figure 2. We estimate Sobol indices by the method described in [31] and implemented in the R library [13] by using a Monte Carlo sample of size 10^5 for each code. No surrogate model was used to estimate the indices in Figure 2. The indices are computed as in [12, equation 11] for each t_u. The areas shown in Figure 2 are the compilation of all the incides calculated in each t_u. The difference between the sum of the indices of order 1 and 1 is considered as interactions. The main result is that the two codes depend on the same input variables. The four most important input variables are y_0, k, M, and θ_0. It can be seen in Figure 2 that the interactions between codes are mostly linear. Maximum amplitude points of $z_L(x, t_u)$ and $z_H(x, t_u)$, which are more complex interactions between codes, can be seen in the horizontal part of the bottom graph of Figure 2.

7.2. Comparison between methods. The experimental designs used to compare the methods are presented in [19]. They are constructed from two independent maximum LHS (Latin hypercube sampling) designs with $N_H = 10$ and $N_L = 100$ points. The low-fidelity design is then modified so that the designs are nested. Only the points of the low-fidelity design closest to the points of the high-fidelity design are moved. To generate these designs, the R packages [7, 17] are used. A random uniform nested design can also be used, but we choose a more effective design for GP regression. The test design is composed of 4000 points randomly chosen in the hypercube determined by the supports of the uniform distributions described in Table 1.

In this section, we want to demonstrate the interest of the method presented in section 2. For this, we will compare several methods:

- The multifidelity method is presented in subsection 6.1 with a Dirac distribution of $\boldsymbol{\Gamma}$, called the SVD method.
- The multifidelity method uses GP regression of the orthogonal part with covariance-tensorization, and the distribution of $\boldsymbol{\Gamma}$ is Dirac at $\boldsymbol{\gamma}$, the matrix of the SVD of the observed low-fidelity code outputs. Its prediction is computed as in subsection 6.2 and called the Dirac method.

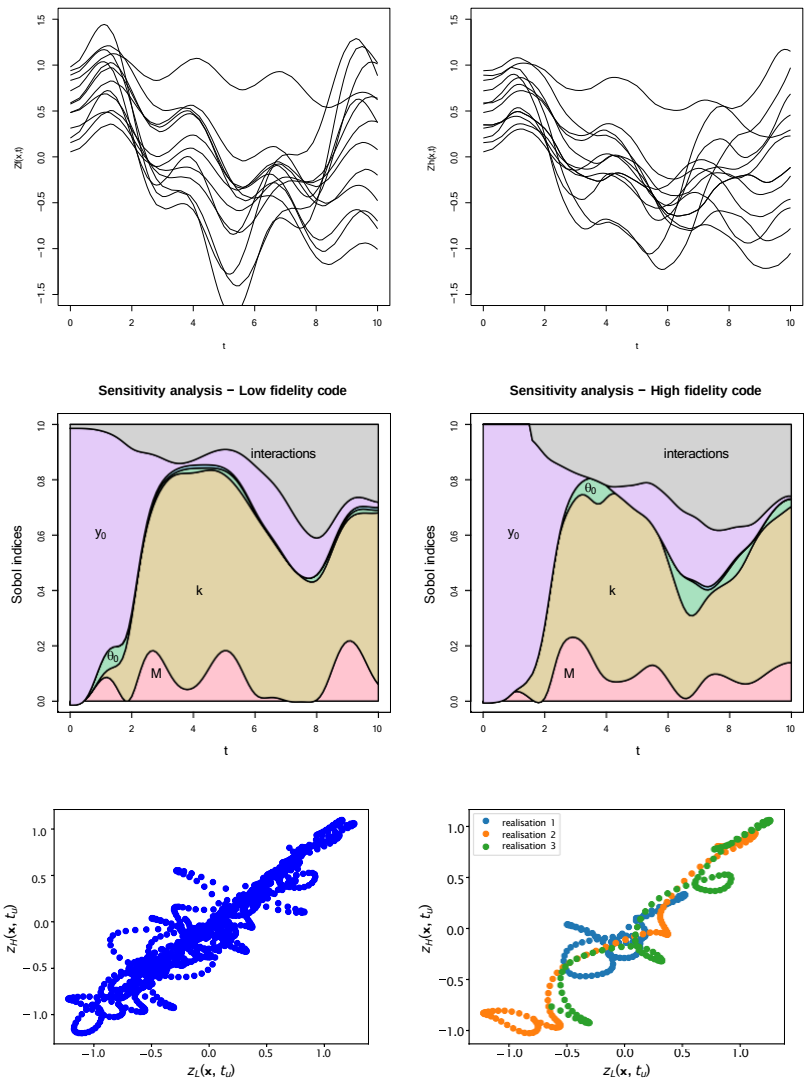

Figure 2. *Comparison between low-fidelity (top left) and high-fidelity (top right) code outputs. Sobol indices for high- and low-fidelity codes (center left: low-fidelity; center right: high-fidelity). For each time t in the time grid, we report the first-order Sobol indices, and ``interactions"" stands for the sum of the Sobol indices of order larger than 2. Finally, the bottom plots represent the interactions between codes (plots of* $(z_L(x,t_u), z_H(x,t_u))_{u=1}^{N_t}$ *for different x). The bottom left graph is for 10 values and the bottom right graph for 3 values of* \bfitx.

\bullet The multifidelity method is presented in subsection 6.2 with the CVB distribution, called the CVB method.

\bullet The neural network (NN) method is presented in [23]. We extend this method for time-series outputs by considering N_t-dimensional outputs for the low- and high-fidelity neural networks and by removing the physical inspired NN (PINN) part. The PINN part is removed in order to reduce the parameters and allows the learning phase to be faster and more accurate; see [35] for more details. We used the parameters proposed

in the article, i.e., 2 hidden layers for each network with 20 neurons per layer. We also tested the NN method up to 100 neurons per layer, but the best results were obtained with approximately 20 neurons.

The method we would like to highlight is the CVB method.

CVB basis. The law of $\boldsymbol{\Gamma}$ needs to be determined in order to compute or estimate the moments (6.3), (6.5), (6.14), and (6.16). The distribution of $\boldsymbol{\Gamma}$ is the CVB distribution described in subsection 3.3. As shown by (3.2), it depends on the size k of the random subset I. Here we choose $k = 4$. Because it is too expensive to compute the sum over all $\binom{N_L}{k}$ different subsets $\{j_1, \ldots, j_k\}$, we estimate the expectation (3.2) by an empirical average over $n = 64$ realizations I_j of the random subset I,

$$\mathbb{E}[f(\boldsymbol{\Gamma})] \simeq \frac{1}{n}\sum_{j=1}^{n} f(\Psi_{I_j}), \tag{7.1}$$

where Ψ_{I_j} is the matrix of the left singular vectors of the SVD of $(z_L(\boldsymbol{x}^{(i)}, t_u))_{\substack{u\in\{1,\ldots,N_t\}\\ i\in\{1,\ldots,N_L\}\setminus I_j}}$.

We have checked that the stability with respect to k is conserved if $1 < k < N_L - N_H$ and that the stability with regard to n is valid if $n > \max(k, 50)$. We have tested the construction of the CVB basis for all k values in this range and found that changes in k do not influence the basis significantly.

The computational cost of calculating the basis is very important in particular because it is impossible for us to calculate it for all subsets. A method to compute the basis with only a cost of $O(N_t^2)$ is given in [24], whereas we compute it with $O(N_L^2 N_t)$ by our method. The gain is, however, very small, especially if $N_L \ll N_t$ which is our case. We have therefore not implemented this method in the results presented in this paper.

Prediction of the orthogonal part. A simple model for the a priori mean function is chosen as $M = 1$ and $f(\boldsymbol{x}) = 1$. Consequently, $F^T R_X^{-1} F = \sum_{i,j} \{R_X^{-1}\}_{i,j}$.

Multifidelity regression of the coefficients. Our implementation of the multifidelity regression is based on [17]. We use an informative prior for the regression of the coefficients. For more information, refer to [19, section 3.4.4]. In this example, the size of the priors is $q = p_L = p_H = 1$. Considering the relation between the two codes, we choose $b^{\rho} = 1$. The trend is supposed to be null; consequently, $b_H^{\beta} = b_L = 0$. The variances are $\sigma_L = 0.5$ and $\sigma_H = 0.5$ with $V_H^{\beta} = 2$ and $V_L = 2$. The parameters for the inverse gamma distribution are $m_L = m_H = 0.2$ and $\varsigma_L = \varsigma_H = 1.5$. We have checked the robustness of the results with respect to the hyperparameters of the prior distributions. Alternatively, the article [22] presents noninformative priors for the autoregressive cokriging.

Prediction. In order to estimate the errors of the surrogate models, we calculate their Q^2's and report them in Figure 3. To compute Q^2 for a model, we calculate the difference between the validation set of size 4000 and the predictions of the model. We have averaged the estimates of the Q^2 over 40 different experimental designs.

The SVD method gives a very interesting result because the Q^2 is almost always higher than 0.8. However, in Figure 4, we can see that it does not capture the form of the times series. The Q^2 of the Dirac and CVB methods are larger than the ones of the other methods. The error is also less variable as a function of t. And the variance is much lower for both

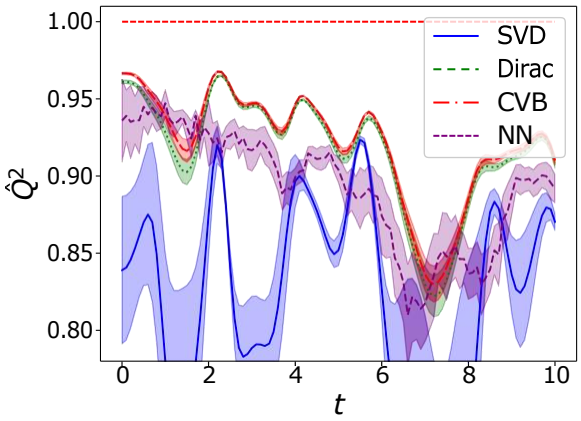

Figure 3. *Comparison between the methods in terms of time-dependent* $\hat{Q}^2$. *Averages over* 40 *random experimental designs are computed. The colored fields represent the confidence intervals determined by* \pm 1.96 *empirical standard deviation. Here* $N_H = 10$, $N_L = 100$, *and* $N_t = 101$.

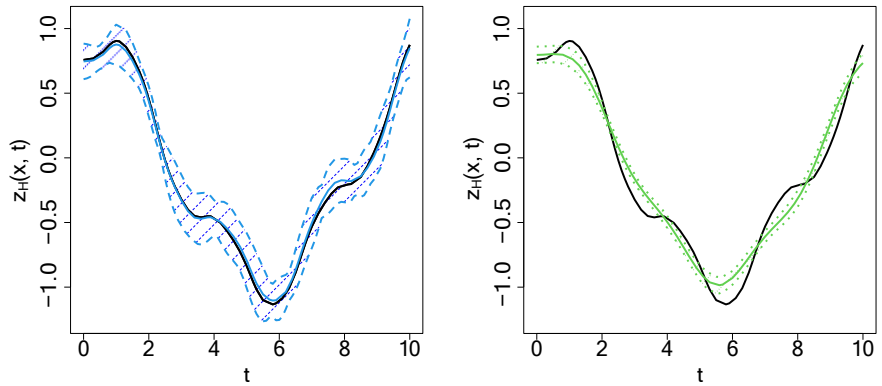

Figure 4. *Comparison between the predictions of the CVB method (left) and the SVD method (right). The black solid line is the exact high-fidelity time series, the colored solid line is the prediction mean, and the dashed lines are the confidence intervals. In this example, the value of N obtained by cross validation is* 8. *The uncertainty interval is* 1.96 *times the standard deviation.*

methods. However, even if there is a difference between the Dirac and CVB methods, it is not possible to say that the CVB method is better in this application. The difference between the Dirac and CVB methods is small in our example.

The variance of the prediction is very important for the quantification of prediction uncertainty. All formulas are given in the previous sections, and we illustrate the results in Figure 4. We can see that the variance of the projection method is not accurate and overestimates the quality of the prediction. This method is not acceptable for prediction. The Dirac method and the CVB method have almost the same variance. If we compare to the variance of the SVD method, it means that most of the uncertainty relates to the orthogonal part. This leads to the conclusion that this part is important in the regression.

In order to understand the interest of the method with covariance tensorization for the orthogonal part, we study in more detail the orthogonal part. First, we study the role of the

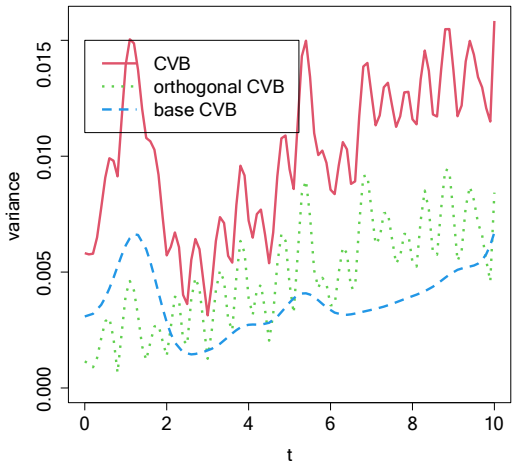

Figure 5. *Estimation of the different time-dependent prediction variance terms for the CVB method.*

value of N . Here $N_L = 100$, so the possible values of N are between 0 and 100. We find that the optimal value of N for 40 learning sets is between 8 and 10. Even when the value of N_H is increased, N remains constant in the 8 to 10 range. This means that the low-fidelity code can give reliable information on the high-fidelity code output projection into an eight-dimensional space. The high-fidelity code output is, however, higher dimensional, and it is important to predict the orthogonal part with a dedicated method, namely, the proposed covariance tensorization method.

We have carried out extensive numerical simulations with values of N_H in the range $[5 : 20]$ and values of N_L in the range $[50 : 1000]$. If only very small data sets are available ($5 \leq N_H \leq 7$), the prediction is not satisfactory whatever the method. Moreover, for values of N_L greater than 200, there is no significant change except for the NN method, which improves to the level of the CVB method for the largest data sets. There is also a decrease in prediction uncertainty with the increase of the N_H number, as can be expected. At the same time as the prediction uncertainty increases when N_H is decreased, there is also a decrease in prediction performance but independently of the method. The code we used is available in [15].

8. Discussion. The objective of this work is to propose a method that generates a surrogate model in the context of multifidelity and time-series outputs and that quantifices the prediction uncertainty. The method we propose is based on three main ingredients: dimension reduction, cokriging, and covariance tensorization. The model we present is based on multifidelity (cokriging) regression. By reducing the output dimension, multifidelity regression becomes possible. To take into account all the information contained in the data sets, the part that cannot be treated with the previous method is predicted by GP regression with covariance tensorization.

First, we have presented different ways to build the basis that allows to represent the high- and low-fidelity code outputs. Second, we have presented a model that allows to estimate the high-fidelity code outputs from data collected from the high- and low-fidelity codes. The combination of a multifidelity part and a single-fidelity part with tensorized covariance is the

central point of the proposed method. The performance of our model has been tested on a mechanical example. We have been able to use multifidelity in a very convincing way to build a robust surrogate model better than any other method presented so far.

There are several ways to extend the method presented in this article. Sequential experimental designs in a multifidelity context have already been dealt with by [20]. However, they deserve to be extended to the case of time-series outputs. We can consider regression problems for more than two levels of code. It is conceivable in this case to build several levels of bases which from code to code would improve the basis and thus reduce the orthogonal part. In addition, high-dimensional outputs are not different from time-series outputs as considered in this paper. It is therefore conceivable to adapt this method to more general functional outputs.

Appendix A. Multifidelity priors for AR(1) model.

In the following, we define the priors needed to use the AR(1) model defined in section 4.

The goal of a Bayesian prediction is to integrate the uncertainty of the parameter estimation into the predictive distribution as in [18]. Here the parameters are σ, β, and ρ. As explained in [21], the result is not Gaussian, but we can obtain expressions of the posterior mean $\mathbb{E}[A_H(\boldsymbol{x}) \mid A = \alpha]$ and variance $\mathbb{V}[A_H(\boldsymbol{x}) \mid A = \alpha]$. It is possible to consider informative or noninformative priors for the parameters [21, 22]. Here we consider informative conjugate priors:

$$\left[\frac{1}{\sigma^2}\right] \sim \mathcal{IG}(m_L, \sigma^2_L), \tag{A.1}$$

$$[\beta_L \mid \sigma^2] \sim \mathcal{N}_{p_L}(b_L, \sigma^2 V_L), \tag{A.2}$$

$$\left[\frac{1}{\sigma^2} \mid A^L = \alpha^L, \beta_L, \sigma^2_L\right] \sim \mathcal{IG}(m_H, \sigma^2_H), \tag{A.3}$$

$$\left[\begin{pmatrix}\beta_\rho \\ \beta_{A^L}\end{pmatrix} \mid A^L = \alpha^L, \sigma^2_H, \beta_L, \sigma^2_L\right] \sim \mathcal{N}_{q+p_H}\left(b_H = \begin{pmatrix} b_H^\rho \\ b_H^\beta \end{pmatrix}, \sigma^2_H V_H = \sigma^2_H \begin{pmatrix} V^\rho & 0 \\ 0 & V_H^\beta \end{pmatrix}\right). \tag{A.4}$$

Here

- b_L is a vector of size p_L;
- b^ρ is a vector of size q;
- b_H^β is a vector of size p_H;
- V_H^β is a $p_H \times p_H$ matrix;
- V^ρ is a $q \times q$ matrix;
- V_L is a $p_L \times p_L$ matrix;
- m_{IF} and σ^2_{IF} are positive real numbers, and $\mathcal{IG}$ stands for the inverse gamma distribution.

By using these informative conjugate priors, we obtain the following a posteriori distributions as in [21]:

$$\left[\frac{1}{\sigma^2} \mid A^L = \alpha^L\right] \sim \mathcal{IG}(d_L, Q_L), \tag{A.5}$$

$$[\beta_L \mid A^L = \alpha^L, \sigma^2] \sim \mathcal{N}_p(C_L \mathcal{W}_L, C_L), \tag{A.6}$$

$$\left[\frac{1}{\sigma^2} \mid A = \alpha\right] \sim \mathcal{IG}(d_H, Q_H), \tag{A.7}$$

$$[\beta_H, \beta_\rho \mid A = \alpha, \sigma^2] \sim \mathcal{N}_{p+q}(C_H \mathcal{W}_H, C_H). \tag{A.8}$$

Here

- $d_{\mathrm{F}} = \frac{n_{\mathrm{F}}}{2} + m_{\mathrm{F}}$;
- $Q_{\mathrm{F}} = (y_{\mathrm{F}} - H_{\mathrm{F}}\hat{\beta}_{\mathrm{F}})^{\mathrm{T}} C_{\mathrm{F}}^{-1}(y_{\mathrm{F}} - H_{\mathrm{F}}\hat{\beta}_{\mathrm{F}})$;
- $Q_{\mathrm{F}} = \tilde{Q}_{\mathrm{F}} + \sigma_{\mathrm{F}}^2 + (b_{\mathrm{F}} - \hat{\beta}_{\mathrm{F}})^{\mathrm{T}} (V_{\mathrm{F}} + (H_{\mathrm{F}}^{\mathrm{T}} C_{\mathrm{F}}^{-1} H_{\mathrm{F}}))^{-1} (b_{\mathrm{F}} - \hat{\beta}_{\mathrm{F}})$;
- $C_{\mathrm{F}} = \left[H_{\mathrm{F}}^{\mathrm{T}} C_{\mathrm{F}}^{-1} H_{\mathrm{F}} + \frac{V_{\mathrm{F}}^{-1}}{\sigma_{\mathrm{F}}^2} \right]^{-1}$;
- $\nu_{\mathrm{F}} = H_{\mathrm{F}}^{\mathrm{T}} C_{\mathrm{F}}^{-1} \alpha^{\mathrm{F}} + \frac{V_{\mathrm{F}}^{-1}}{\sigma_{\mathrm{F}}^2} b_{\mathrm{F}}$;
- H_{F} is defined by $H_{\mathrm{L}} = F_{\mathrm{L}}$ and $H_{\mathrm{H}} = [G^{\mathrm{L}} \odot (\alpha^{\mathrm{H}} \mathbf{1}_{q_{\mathrm{L}}}^{\mathrm{T}})\, F_{\mathrm{H}}]$;
- G^{L} is the $N_{\mathrm{H}} \times q$ matrix containing the values of $g^{\mathrm{T}}(\boldsymbol{x}_{\mathrm{L}}^{(i)})$ for $\boldsymbol{x} \in D_{\mathrm{H}}$;
- $\mathbf{1}_{q_{\mathrm{L}}}$ is a q-dimensional vector containing 1;
- $\hat{\beta}_{\mathrm{F}} = (H_{\mathrm{F}}^{\mathrm{T}} C_{\mathrm{F}}^{-1} H_{\mathrm{F}})^{-1} H_{\mathrm{F}}^{\mathrm{T}} C_{\mathrm{F}}^{-1} \alpha^{\mathrm{F}}$.

Consequently, the posterior distribution of $\mathcal{A}_{\mathrm{H}}(\boldsymbol{x})$ has the following mean and variance:

$$\mathbb{E}[\mathcal{A}_{\mathrm{H}}(\boldsymbol{x}) \mid A = \alpha] = h_{\mathrm{H}}^{\mathrm{T}}(\boldsymbol{x}) C_{\mathrm{H}} \nu_{\mathrm{H}} + r^{\mathrm{T}}(\boldsymbol{x}) C_{\mathrm{H}}^{-1} \left(\alpha^{\mathrm{H}} - H_{\mathrm{H}} C_{\mathrm{H}} \nu_{\mathrm{H}} \right), \tag{A.9}$$

$$\mathbb{V}[\mathcal{A}_{\mathrm{H}}(\boldsymbol{x}) \mid A = \alpha] = \left(h_{\mathrm{H}}^{\mathrm{T}}(\boldsymbol{x}) - r_{\mathrm{H}}^{\mathrm{T}}(\boldsymbol{x}) C_{\mathrm{H}}^{-1} H_{\mathrm{H}} \right) C_{\mathrm{H}} \left(h_{\mathrm{H}}^{\mathrm{T}}(\boldsymbol{x}) - r_{\mathrm{H}}^{\mathrm{T}}(\boldsymbol{x}) C_{\mathrm{H}}^{-1} H_{\mathrm{H}} \right)^{\mathrm{T}}$$

$$+ \hat{\sigma}_{\mathrm{H}}^2(\boldsymbol{x}) + \varepsilon(\boldsymbol{x}) \sigma_{\tilde{A}_{\mathrm{L}}}^2(\boldsymbol{x}) + \frac{Q_{\mathrm{H}}}{2(d_{\mathrm{H}} - 1)} \left(1 - r_{\mathrm{H}}^{\mathrm{T}}(\boldsymbol{x}) C_{\mathrm{H}}^{-1} r_{\mathrm{H}}(\boldsymbol{x}) \right), \tag{A.10}$$

where $\hat{\mu}_{\mathrm{L}}(\boldsymbol{x}) = g^{\mathrm{T}}(\boldsymbol{x}) \hat{\beta}_{\rho}$, $\hat{\beta}_{\rho} = [C_{\mathrm{H}} \nu_{\mathrm{H}}]_{i=p_{\mathrm{H}}+1,\ldots,p_{\mathrm{H}}+q}$, and $\varepsilon_{\rho}(\boldsymbol{x}) = g^{\mathrm{T}}(\boldsymbol{x}) \tilde{C}_{\mathrm{H}} g_{\mathrm{L}}(\boldsymbol{x})$ with $\tilde{C}_{\mathrm{H}} = [C_{\mathrm{H}}]_{i,j=p_{\mathrm{H}}+1,\ldots,p_{\mathrm{H}}+q}$.

The posterior mean $\mathbb{E}[A_{\mathrm{H}}(\boldsymbol{x}) \mid A = \alpha]$ is the predictive model of the high-fidelity response, and the posterior variance $\mathbb{V}[\mathcal{A}_{\mathrm{H}}(\boldsymbol{x}) \mid A = \alpha]$ represents the predictive variance of the model.

Appendix B. LOO formula and discussion.

LOO without loop. In order to quickly minimize the LOO error with respect to the vector of correlation lengths $\ell_{\boldsymbol{x}}$, there exist formulas to evaluate $\varepsilon^2(\ell_{\boldsymbol{x}})$ with matrix products [2], [6]. The LOO optimization problem is equivalent to minimize a function $\mathcal{fcv}(\ell_{\boldsymbol{x}})$ given by

$$\mathcal{fcv}(\ell_{\boldsymbol{x}}) = z^{\mathrm{T}} R_{\ell_{\boldsymbol{x}}}^{-1} \mathrm{diag}\left(R_{\ell_{\boldsymbol{x}}}^{-1} \right)^{-2} R_{\ell_{\boldsymbol{x}}}^{-1} z, \tag{B.1}$$

where z is a vector that collects all the data.

Considering (5.3) and the mixed-product property, the inverse of a Kronecker product, and the formula $\mathrm{diag}(A \otimes B) = \mathrm{diag}A \otimes \mathrm{diag}B$, the cost function can be expressed as

$$\mathcal{fcv}(\ell_{\boldsymbol{x}}) = z^{\mathrm{T}} \left[R_t^{-1} \mathrm{diag}\left(R_t^{-1} \right)^{-2} R_t^{-1} \right] \otimes \left[R_x^{-1} \mathrm{diag}\left(R_x^{-1} \right)^{-2} R_x^{-1} \right] z. \tag{B.2}$$

However, the term in R_t is impossible to calculate because R_t is not invertible. $R_t^{-1}\,\mathrm{diag}(R_t^{-1})^{-2}R_t^{-1}$ can be approximated by I_{N_t} in order to have a tractable problem. This assertion is equivalent to the hypothesis

$$R_t^2 = \mathrm{diag}(R_t^{-1})^{-2}. \tag{B.3}$$

This assumption can be seen as the fact that the error is estimated by taking into account only the spatial distribution of the covariance. Indeed, to calculate the error, only the matrix R_x is used, even if the value of R_t is calculated by maximum likelihood later in the GP regression method.

Thus, the minimization described in (5.11) makes it possible to calculate the correlation lengths by minimizing

$$\boldsymbol{f}_{\mathbf{cv}}(\boldsymbol{\ell}_{\mathrm{fix}}) \simeq z^T I_{N_t} \otimes \mathrm{diag}\left(R_x^{-1}\mathrm{diag}\left(R_x^{-1}\right)^{-2}R_x^{-1}\right) z, \tag{B.4}$$

where I_{N_t} is the $N_t \times N_t$ identity matrix. The main interest of this method is to give an approximate value of the error and to make the optimization much faster.

Optimization with hypothesis (B.3). Efficient minimization algorithms require to have the derivative of the function so that it does not have to be calculated by finite differences. Thanks to the simplification (B.4), it is possible to calculate the derivative of the LOO error [2],

$$\begin{aligned}\frac{\partial}{\partial \ell_k}\boldsymbol{f}_{\mathbf{cv}}(\boldsymbol{\ell}_{\mathrm{fix}}) = 2z^T I_{N_t} \otimes R_x^{-1}\mathrm{diag}\left(R_x^{-1}\right)^{-2}\left(R_x^{-1}\frac{\partial R_x}{\partial \ell_k}R_x^{-1}\right)\mathrm{diag}\left(R_x^{-1}\right)^{-1}R_x^{-1}z\\ -2z^T I_{N_t} \otimes R_x^{-1}\mathrm{diag}\left(R_x^{-1}\right)^{-2}R_x^{-1}\frac{\partial R_x}{\partial \ell_k}R_x^{-1}z,\end{aligned} \tag{B.5}$$

with

$$\left(\frac{\partial R_{x,l}}{\partial \ell_1}\right)_{i,j} = \frac{\ell_k^{-1}(x_{k,j}-x_{k,i})^2}{|x_{k,j}-x_{k,i}|}h_{\frac{5}{2}}^r\left(\frac{|x_{k,j}-x_{k,i}|}{\ell_k}\right) \tag{B.6}$$

and

$$h_{\frac{5}{2}}^r(x) = -\frac{5}{3}x\left(1+\sqrt{5}x\right)\exp\left(-\sqrt{5}x\right). \tag{B.7}$$

The method used to calculate the value of $\boldsymbol{\ell}_{\mathrm{fix}}$ is the Nelder--Mead method with only one starting point because starting from more points will be more costly and the function $\boldsymbol{f}_{\mathbf{cv}}$ is close to quadratic and consequently does not need multiple starting points.

Optimization without hypothesis (B.3). When hypothesis (B.3) does not hold, a way must be found to calculate $\boldsymbol{\ell}_{\mathrm{fix}}$ without this assumption. By a regularization of the matrix R_t, it is possible to calculate $\boldsymbol{f}_{\mathbf{cv}}(\boldsymbol{\ell}_{\mathrm{fix}})$ and its derivative by (5.11), (B.2), and (B.8). However, the solution will be a regularized solution and not an exact solution.

There are different types of regularization that allow matrices to be inverted. Two methods have been investigated here. The first one is standard (Tikonov regularization):

$$R_t^{-1} = \left(R_t^T R_t + \mathrm{precision}^2 I_{N_t}\right)^{-1}R_t^T. \tag{B.8}$$

Table 2

Benchmark of the different LOO optimization techniques for estimating the $f(\boldsymbol{x}, t) = \cos(4\pi(x_2 + 1)t)\sin(3\pi x_1 t)$ function using the separable covariance method. Loop LOO processes the error by computing the approximation, full LOO processes the regularized analytic expression, and simplified LOO processes the simplified one given by (B.4).

	Loop LOO	Full LOO	Simplified LOO
ε_{LOO}	$7.06\ 10^{-4}$	$8.72\ 10^{-4}$	$8.14\ 10^{-4}$
time	18.41 s	3.47 min	0.17 s

The second one is

$$R_t^{-1} = (R_t + \varepsilon I_N)^{-1}. \tag{B.9}$$

It has the disadvantage of being more sensitive to ε than the first one, which is why it will not be used.

However, in the calculation of the determinant, this adjustment may have advantages. Denoting by $R_t^{-1} = V C^{-1} U^T$, the SVD of R_t with $C^{-1} = \operatorname{diag}\left(\frac{1}{\sigma_i + \varepsilon}\right)$, whereas the same decomposition gives for (B.8) $C^{-1} = \operatorname{diag}\left(\frac{\sigma_i}{\sigma_i^2 + \varepsilon\sigma_i}\right)$. This is the reason why the two adjustments presented are not used in the same case. Indeed, $\frac{1}{\sigma_i + \varepsilon}$ is less efficient for the calculation of a determinant but more efficient for the calculation of the inverse of R_t.

$$\begin{aligned}\frac{\partial f_{\text{cv}}(\ell)}{\partial \ell_k} = 2z^T R_t^{-1}\operatorname{diag}\left(R_t^{-1}\frac{\partial R_t}{\partial \ell_k}R_t^{-1}\right)R_t^{-1}z \\ \otimes R_x^{-1}\operatorname{diag}\left(R_x^{-1}\frac{\partial R_x}{\partial \ell_k}R_x^{-1}\right)\left(\operatorname{diag}\left(R_t^{-1}\right)\otimes R_x^{-1}\right)z \\ -2z^T R_t^{-1}\operatorname{diag}\left(R_t^{-1}\right)R_t^{-1}\otimes_x R_x^{-1}\operatorname{diag}\left(R_x^{-1}\frac{\partial R_x}{\partial \ell_k}R_x^{-1}\right)R_x^{-1}z.\end{aligned} \tag{B.10}$$

Equations (B.6) and (B.7) are still valid.

This complete approach was compared to the LOO calculation using a loop. However, the calculation time of Kronecker products is too long compared to the calculation of the simple error with a loop. Moreover, the differences in the errors of the different methods are negligible. Thus, this solution is only recommended when the calculation of (B.2) and (B.10) is optimized.

Table 2 shows that the gain in calculation time by the simplified method is significant even though the error difference is very small. The extremely long time for the complete LOO is mainly due to an implementation of the Kronecker product that is not very effective in our implementation.

Appendix C. Tensorized covariance of the orthogonal part.

For R_x, we assume that C_x is chosen in the Matérn-5/2 class of functions. The function only depends on the correlation length vector ℓ. The matrix R_t is estimated as described in section 5 by the matrix $\widehat{R}_t$ given by

$$\widehat{R}_t = \frac{1}{N_x}\left(\widehat{Z}_{\text{interpolation}} - \Psi^{\perp}\right)R_x^{-1}\left(\widehat{Z}_{\text{interpolation}} - \Psi^{\perp}\right)^T,$$

where $\boldsymbol{\Lambda}^{\perp}$ is the $N_t \times N_x$ matrix of empirical means $\Lambda^{\perp}_{u,i} = \frac{1}{N_x} \sum_{j=1}^{N_x} \big[\boldsymbol{\mathit{fitZ}}^{\perp}_{\text{interpolation}}\big]_{u,j} \quad \forall i = 1, \ldots, N_x$ and $u = 1, \ldots, N_t$. Its range is indeed in $S_N^{\perp}$.

The prediction mean is the sum of two terms, $\boldsymbol{\mathit{fitZ}}^{\perp}_{\text{interpolation}} R_x^{-1} r_x(\boldsymbol{\mathit{fitx}})$, which is $S_N^{\perp}$-valued, and $B_{\Lambda} u(\boldsymbol{\mathit{fitx}})$, also $S_N^{\perp}$-valued, because

$$B_{\Lambda} = \boldsymbol{\mathit{fitZ}}^{\perp}_{\text{interpolation}} R_x^{-1} F \Big(F^{\mathsf{T}} R_x^{-1} F \Big)^{-1}, \tag{C.1}$$

with F the $N_{\mathrm{F}} \times M$ matrix $[f^{\mathsf{T}}(\boldsymbol{\mathit{fitx}}^{(i)})]_{i=1,\ldots,N_{\mathrm{eff}}}$. Consequently, we have

$$Z^{\perp}(\boldsymbol{\mathit{fitx}}, t_u) \,|\, \boldsymbol{\ell}_{\boldsymbol{\mathit{fitx}}}, \boldsymbol{\Gamma}, N, \boldsymbol{\mathit{fitZ}}^{\perp}_{\text{interpolation}} \sim \mathcal{GP}(\mu_{\Lambda}(\boldsymbol{\mathit{fitx}}), R_{\Lambda}(\boldsymbol{\mathit{fitx}}, \boldsymbol{\mathit{fitx}}')), \tag{C.2}$$

where the mean is given by (5.5) and the covariance by (5.6) with $\boldsymbol{\mathit{fitZ}}^{\perp}_{\text{interpolation}}$ as the observed inputs. LOO estimation of the vector of correlation lengths $\boldsymbol{\ell}_{\boldsymbol{\mathit{fitx}}}$ given $\boldsymbol{\Gamma}$ and N is carried out by the method presented in Appendix B.

Appendix D. Expressions of some expectations and variances.

D.1. Computation for uncorrelated GPs Given $\boldsymbol{\Gamma}, (A_{i,\mathrm{H}}(\boldsymbol{\mathit{fitx}}, t_u), A_{i,\mathrm{L}}(\boldsymbol{\mathit{fitx}}, t_u))_{\substack{\boldsymbol{\mathit{fitx}} \in \mathcal{Q} \\ u=1,\ldots,N_t}}$ are independent with respect to i. This independence makes it possible to generate N_t independent surrogate models, with mean and variance given by (A.9) and (A.10):

$$\mathbb{E}[A_{i,\mathrm{H}}(\boldsymbol{\mathit{fitx}}) \,|\, \boldsymbol{\Gamma}, A = \alpha] = r_{i,\mathrm{H}}^{\mathsf{T}}(\boldsymbol{\mathit{fitx}}) C_{i,\mathrm{H}}^{-1} \nu_{i,\mathrm{H}} + r_{i,\mathrm{H}}^{\mathsf{T}}(\boldsymbol{\mathit{fitx}}) C_{i,\mathrm{H}}^{-1} \Big(\alpha_i^{\mathrm{H}} - H_{i,\mathrm{H}} C_{i,\mathrm{H}}^{-1} \nu_{i,\mathrm{H}} \Big), \tag{D.1}$$

$$\begin{aligned}\mathbb{V}[A_{i,\mathrm{H}}(\boldsymbol{\mathit{fitx}}) \,|\, \boldsymbol{\Gamma}, A = \alpha] = {} & \Big(1 + \operatorname{tr}\big(\boldsymbol{\mathit{fitx}}\big) + \varepsilon_{i,\mathrm{H}}(\boldsymbol{\mathit{fitx}}) \Big) \sigma^2_{\mathrm{H},i} \tilde{A}_{\mathrm{L},i}(\boldsymbol{\mathit{fitx}}) \\ & + \frac{Q_{i,\mathrm{H}}}{2(d_{i,\mathrm{H}} - 1)} (1 - r_{i,\mathrm{H}}^{\mathsf{T}}(\boldsymbol{\mathit{fitx}}) C_{i,\mathrm{H}}^{-1} r_{i,\mathrm{H}}(\boldsymbol{\mathit{fitx}}))\end{aligned}$$

$$+ \Big(h_{i,\mathrm{H}}^{\mathsf{T}}(\boldsymbol{\mathit{fitx}}) - r_{i,\mathrm{H}}^{\mathsf{T}}(\boldsymbol{\mathit{fitx}}) C_{i,\mathrm{H}}^{-1} H_{i,\mathrm{H}} \Big) C_{i,\mathrm{H}} \Big(h_{i,\mathrm{H}}^{\mathsf{T}}(\boldsymbol{\mathit{fitx}}) - r_{i,\mathrm{H}}^{\mathsf{T}}(\boldsymbol{\mathit{fitx}}) C_{i,\mathrm{H}}^{-1} H_{i,\mathrm{H}} \Big)^{\mathsf{T}}. \tag{D.2}$$

D.2. Variance for projection The law of total variance gives

$$\mathbb{V}_{\alpha}[Z_{\mathrm{H}}(\cdot, t_u)] = \mathbb{V}_{\alpha}[\mathbb{E}_{\alpha}[Z_{\mathrm{H}}(\cdot, t_u) \,|\, \boldsymbol{\Gamma}]] + \mathbb{E}_{\alpha}[\mathbb{V}_{\alpha}[Z_{\mathrm{H}}(\cdot, t_u) \,|\, \boldsymbol{\Gamma}]]. \tag{D.3}$$

The variance term can be expressed as follows:

$$\begin{aligned}\mathbb{V}_{\alpha}[\mathbb{E}_{\alpha}[Z_{\mathrm{H}}(\cdot, t_u) \,|\, \boldsymbol{\Gamma}]] &= \mathbb{V}_{\alpha}\Big[\sum_{i=1}^{N_t} \Gamma_i(t_u) \mathbb{E}_{\alpha}[A_{i,\mathrm{H}}(\cdot) \,|\, \boldsymbol{\Gamma}]\Big] \\ &= \sum_{i=1}^{N_t} \mathbb{V}_{\alpha}[\Gamma_i(t_u) \mathbb{E}_{\alpha}[A_{i,\mathrm{H}}(\cdot) \,|\, \boldsymbol{\Gamma}]] \\ &+ \sum_{i,j=1, i \neq j}^{N_t} \mathrm{Cov}_{\alpha}\big(\Gamma_i(t_u) \mathbb{E}_{\alpha}[A_{i,\mathrm{H}}(\cdot) \,|\, \boldsymbol{\Gamma}], \Gamma_j(t_u) \mathbb{E}_{\alpha}[A_{j,\mathrm{H}}(\cdot) \,|\, \boldsymbol{\Gamma}]\big)\end{aligned} \tag{D.4}$$

where $\mathbb{E}_{\alpha}[A_{i,\mathrm{H}}(\boldsymbol{\mathit{fitx}}) \,|\, \boldsymbol{\Gamma}]$ is given by (D.1). The expectation term can be expressed as

$$\begin{aligned}\mathbb{E}_{\alpha}[\mathbb{V}_{\alpha}[Z_{\mathrm{H}}(\cdot, t_u) \,|\, \boldsymbol{\Gamma}]] &= \mathbb{E}_{\alpha}\Big[\sum_{i=1}^{N_t} \mathbb{V}_{\alpha}[A_{i,\mathrm{H}}(\cdot) \Gamma_i(t_u) \,|\, \cdot]\Big] + \sum_{i,j=1, i \neq j}^{N_t} \mathrm{Cov}_{\alpha}(A_{i,\mathrm{H}}(\cdot) \Gamma_i(t_u), A_{j,\mathrm{H}}(\cdot) \Gamma_j(t_u) \,|\, \boldsymbol{\Gamma}) \\ &= \sum_{i=1}^{N_t} \mathbb{E}_{\alpha}\Big[\Gamma_i^2(t_u) \mathbb{V}_{\alpha}[A_{i,\mathrm{H}}(\cdot) \,|\, \boldsymbol{\Gamma}]\Big] \\ &+ \sum_{i,j=1, i \neq j}^{N_t} \mathbb{E}_{\alpha}[\Gamma_i(t_u) \Gamma_j(t_u) \mathrm{Cov}(A_{i,\mathrm{H}}(\cdot), A_{j,\mathrm{H}}(\cdot) \,|\, \boldsymbol{\Gamma})]\end{aligned} \tag{D.5}$$

where $\mathbb{V}_{\alpha}[A_{i,H}(\boldsymbol{x})\Gamma_i(t_u)\,|\,\boldsymbol{\Gamma}]$ is given in (D.2) and $\mathrm{Cov}_{\alpha}(A_{i,H}(\boldsymbol{x}), A_{j,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}) = 0$ if $i \neq j$. Consequently,

$$\begin{aligned}\mathbb{V}_{\alpha}[Z_H(\boldsymbol{x},t_u)] &= \sum_{i=1}^{N_r}\mathbb{V}_{\alpha}[\Gamma_i(t_u)]_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}]\\ &+\sum_{i,j=1,i\neq j}^{N}\mathrm{Cov}_{\alpha}([\Gamma_i(t_u)]_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}],[\Gamma_j(t_u)]_{\alpha}[A_{j,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}])\\ &+\sum_{i=1}^{N_r}\mathbb{E}_{\alpha}\Big[\Gamma_i^2(t_u)\mathbb{V}_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}]\Big].\end{aligned} \tag{D.6}$$

D.3. Variance for tensorization of covariance and projection The theorem of the total variance gives us

$$\begin{aligned}\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H(\boldsymbol{x},t_u)\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}] &= \mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\\ &+\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}].\end{aligned} \tag{D.7}$$

The two terms of (6.15) are

$$\begin{aligned}&\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}] =\\ &\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H^{\perp}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}] + \mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N]\end{aligned} \tag{D.8}$$

and

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}] =\\ &\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H^{\perp}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\\ &+\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[2\mathrm{Cov}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H^{\parallel}(\boldsymbol{x},t_u),Z_H^{\perp}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\\ &+\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H^{\parallel}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N]\,|\,N].\end{aligned} \tag{D.9}$$

The uncorrelation of the $A_{i,H}(\boldsymbol{x},t_u)$ coefficients given $\boldsymbol{\Gamma}$ gives $\mathrm{Cov}_{\alpha}[A_{i,H}(\boldsymbol{x}),A_{j,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}] = 0$ for $i \neq j$ and $\mathrm{Cov}_{\boldsymbol{x}\boldsymbol{t}Z}[A_{i,H}(\boldsymbol{x})\Gamma_i(t_u),Z_H^{\perp}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma}] = 0$. This leads us to simplify (D.8) and (D.9) into

$$\begin{aligned}\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]] &= \mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{E}_{\alpha}[Z_H^{\perp}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\\ &+\sum_{i=1}^{N}\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[\Gamma_i(t_u)]_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}]\\ &+\sum_{i,j=1;i\neq j}^{N}\mathrm{Cov}_{\boldsymbol{x}\boldsymbol{t}Z}[\Gamma_i(t_u)]_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}],\Gamma_j(t_u)]_{\alpha}[A_{j,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}]]\\ &+2\sum_{i=1}^{N}\mathrm{Cov}_{\boldsymbol{x}\boldsymbol{t}Z}[\Gamma_i(t_u)]_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}],\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H^{\perp}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\end{aligned} \tag{D.10}$$

and

$$\begin{aligned}\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]] &= \mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{V}_{\alpha}[Z_H^{\perp}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\\ &+\sum_{i=1}^{N}\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[\Gamma_i(t_u)^2]_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\end{aligned} \tag{D.11}$$

The full formula of the variance can be expressed as

$$\begin{aligned}\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H(\boldsymbol{x},t_u)\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}] &= \mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{E}_{\alpha}[Z_H^{\perp}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\\ &+\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[\mathbb{V}_{\alpha}[Z_H(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\\ &+\sum_{i=1}^{N}\mathbb{V}_{\boldsymbol{x}\boldsymbol{t}Z}[\Gamma_i(t_u)]_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}]\\ &+\sum_{i=1}^{N}\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[\Gamma_i(t_u)^2]_{\alpha}[A_i(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}]\\ &+\sum_{i,j=1;i\neq j}^{N}\mathrm{Cov}_{\boldsymbol{x}\boldsymbol{t}Z}[\Gamma_i(t_u)]_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}],\Gamma_j(t_u)]_{\alpha}[A_{j,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}]]\\ &+2\sum_{i=1}^{N}\mathrm{Cov}_{\boldsymbol{x}\boldsymbol{t}Z}[\Gamma_i(t_u)]_{\alpha}[A_{i,H}(\boldsymbol{x})\,|\,\boldsymbol{\Gamma}],\mathbb{E}_{\boldsymbol{x}\boldsymbol{t}Z}[Z_H^{\perp}(\boldsymbol{x},t_u)\,|\,\boldsymbol{\Gamma},N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}]\,|\,N,\boldsymbol{\ell}_{\boldsymbol{x}\boldsymbol{t}}],\end{aligned} \tag{D.12}$$

Acknowledgments. The author thanks Prof. Josselin Garnier and Dr. Claire Cannamela for their guidance and advice.

REFERENCES

[1] G. Aversano, A. Bellemans, Z. Li, A. Coussement, O. Gicquel, and A. Parente, *Application of reduced-order models based on PCA \& kriging for the development of digital twins of reacting flow applications*, Comput. Chem. Eng., 121 (2019), pp. 422--441.

[2] F. Bachoc, *Cross validation and maximum likelihood estimations of hyper-parameters of Gaussian processes with model misspecification*, Comput. Statist. Data Anal., 66 (2013), pp. 55--69.

[3] S. Conti, J. P. Gosling, J. E. Oakley, and A. O'Hagan, *Gaussian process emulation of dynamic computer codes*, Biometrika, 96 (2009), pp. 663--676.

[4] K. Cutajar, M. Pullin, A. Damianou, N. Lawrence, and J. Gonz\'alez, *Deep Gaussian Processes for Multi-Fidelity Modeling*, preprint, arXiv:1903.07320v1, 2019.

[5] P. Diaconis, *What is a random matrix*, Not. AMS, 52 (2005), pp. 1348--1349.

[6] O. Dubrule, *Cross validation of kriging in a unique neighborhood*, J. Int. Assoc. Math. Geol., 15 (1983), pp. 687--699.

[7] D. Dupuy, C. Helbert, and J. Franco, *DiceDesign and DiceEval: Two R packages for design and analysis of computer experiments*, J. Stat. Softw., 65 (2015), pp. 1--38, http://www.jstatsoft.org/v65/i11/.

[8] A. I. Forrester, A. S\'obester, and A. J. Keane, *Multi-fidelity optimization via surrogate modelling*, Proc. R. Soc. A Math. Phys. Eng. Sci., 463 (2007), pp. 3251--3269.

[9] M. Giselle Fern\'andez-Godino, C. Park, N. H. Kim, and R. T. Haftka, *Issues in deciding whether to use multifidelity surrogates*, AIAA J., 57 (2019), pp. 2039--2054.

[10] J. Goh, D. Bingham, J. P. Holloway, M. J. Grosskopf, C. C. Kuranz, and E. Rutter, *Prediction and computer model calibration using outputs from multifidelity simulators*, Technometrics, 55 (2013), pp. 501--512.

[11] O. Grujic, A. Menafoglio, G. Yang, and J. Caers, *Cokriging for multivariate Hilbert space valued random fields: Application to multi-fidelity computer code emulation*, Stoch. Environ. Res. Risk Assess., 32 (2018), pp. 1955--1971.

[12] B. Iooss and P. Lema\^itre, *A review on global sensitivity analysis methods*, in Uncertainty Management in Simulation-Optimization of Complex Systems: Algorithms and Applications, Springer, (2015), pp. 101--122.

[13] B. Iooss, S. D. Veiga, A. Janon, G. Pujol, with contributions from B. Broto, K. Boumhaout, T. Delage, R. E. Amri, J. Fruth, L. Gilquin, J. Guillaume, L. Le Gratiet, P. Lemaitre, A. Marrel, A. Meynaoui, B. L. Nelson, F. Monari, R. Oomen, O. Rakovec, B. Ramos, O. Roustant, E. Song, J. Staum, R. Sueur, T. Touati, and F. Weber, *Sensitivity: Global Sensitivity Analysis of Model Outputs*, R package version 1.22.1, https://CRAN.R-project.org/package=sensitivity, 2020.

[14] M. Kennedy and A. O'Hagan, *Predicting the output from a complex computer code when fast approximations are available*, Biometrika, 87 (2000), pp. 1--13.

[15] B. Kerleguer, *MultiFi Time-Series*, https://github.com/ehbihenscoding/MultiFiTimeSeries, 2022.

[16] D. G. Krige, *A statistical approach to some basic mine valuation problems on the witwatersrand*, J. South Afr. Inst. Min. Metall., 52 (1951), pp. 119--139.

[17] L. Le Gratiet, *MuFiCokriging: Multi-Fidelity Cokriging Models*, R package version 1.2, https://CRAN.R-project.org/package=MuFiCokriging, 2012.

[18] L. Le Gratiet, *Bayesian analysis of hierarchical multifidelity codes*, SIAM/ASA J. Uncertain. Quantif., 1 (2013), pp. 244--269.

[19] L. Le Gratiet, *Multi-Fidelity Gaussian Process Regression for Computer Experiments*, Ph.D. thesis, Universit\'e Paris-Diderot--Paris VII, 2013, https://tel.archives-ouvertes.fr/tel-00866770.

[20] L. Le Gratiet and C. Cannamela, *Cokriging-based sequential design strategies using fast cross-validation techniques for multi-fidelity computer codes*, Technometrics, 57 (2015), pp. 418--427.

[21] L. Le Gratiet and J. Garnier, *Recursive co-kriging model for design of computer experiments with multiple levels of fidelity*, Int. J. Uncertain. Quantif., 4 (2014), pp. 364--386.
[22] P. Ma, *Objective Bayesian analysis of a cokriging model for hierarchical multifidelity codes*, SIAM/ASA J. Uncertain. Quantif., 8 (2020), pp. 1358--1382.
[23] X. Meng and G. E. Karniadakis, *A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems* , J. Comput. Phys., 401 (2020), 109020.
[24] B. Mertens, T. Fearn, and M. Thompson, *The efficient cross-validation of principal components applied to principal component regression*, Stat. Comput., 5 (1995), pp. 227--235.
[25] S. Nanty, C. Helbert, A. Marrel, N. Pérot, and C. Prieur, *Uncertainty quantification for functional dependent random variables*, Comput. Statist., 32 (2017), pp. 559--583.
[26] D. Nerini, P. Monestiez, and C. Manté, *Cokriging for spatial functional data*, J. Multivariate Anal., 101 (2010), pp. 409--418.
[27] P. Perdikaris, M. Raissi, A. Damianou, N. D. Lawrence, and G. E. Karniadakis, *Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling* , Proc. R. Soc. A Math. Phys. Eng. Sci., 473 (2017), 20160751.
[28] G. Perrin, *Adaptive calibration of a computer code with time-series output* , Reliab. Eng. Syst. Safety, 196 (2020), 106728.
[29] G. Pilania, J. E. Gubernatis, and T. Lookman, *Multi-fidelity machine learning models for accurate bandgap predictions of solids*, Comput. Mater. Sci., 129 (2017), pp. 156--163.
[30] J. Rougier, *Efficient emulators for multivariate deterministic functions*, J. Comput. Graph. Statist., 17 (2008), pp. 827--843.
[31] A. Saltelli, P. Annoni, I. Azzini, F. Campolongo, M. Ratto, and S. Tarantola, *Variance based sensitivity analysis of model output: Design and estimator for the total sensitivity index* , Comput. Phys. Commun., 181 (2010), pp. 259--270.
[32] T. J. Santner, B. J. Williams, W. Notz, and B. J. Williams, *The Design and Analysis of Computer Experiments*, Springer, New York, 2003.
[33] C. F. Van Loan and G. H. Golub, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 1983.
[34] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, MA, 2006.
[35] X. Zhang, F. Xie, T. Ji, Z. Zhu, and Y. Zheng, *Multi-fidelity deep neural network surrogate model for aerodynamic shape optimization*, Comput. Methods Appl. Mech. Engrg., 373 (2021), 113485.
[36] Q. Zhou, Y. Wu, Z. Guo, J. Hu, and P. Jin, *A generalized hierarchical co-kriging model for multi-fidelity data fusion*, Struct. Multidiscip. Optim., 62 (2020), pp. 1885--1904.