Received: 30 June 2022 | Revised: 8 September 2022 | Accepted: 30 September 2022

DOI: 10.1002/cem.3452

RESEARCH ARTICLE

Journal of CHEMOMETRICS WILEY

On partial least-squares estimation in scalar-on-function regression models

Semanur Saricam[1] | **Ufuk Beyaztas**[2] | **Baris Asikgil**[3] | **Han Lin Shang**[4]

[1]Graduate School of Natural and Applied Sciences, Mimar Sinan Fine Arts University, Istanbul, Turkey

[2]Department of Statistics, Marmara University, Istanbul, Turkey

[3]Department of Statistics, Mimar Sinan Fine Arts University, Istanbul, Turkey

[4]Department of Actuarial Studies and Business Analytics, Macquarie University, Sydney, New South Wales, Australia

Correspondence
Ufuk Beyaztas, Department of Statistics, Marmara University, 34722 Kadikoy-Istanbul, Turkey.
Email: ufuk.beyaztas@marmara.edu.tr

Funding information
Scientific and Technological Research Council of Turkey (TUBITAK), Grant/Award Number: 120F270

Abstract

Scalar-on-function regression, where the response is scalar valued and the predictor consists of random functions, is one of the most important tools for exploring the functional relationship between a scalar response and functional predictor(s). The functional partial least-squares method improves estimation accuracy for estimating the regression coefficient function compared to other existing methods, such as least squares, maximum likelihood, and maximum penalized likelihood. The functional partial least-squares method is often based on the SIMPLS or NIPALS algorithm, but these algorithms can be computationally slow for analyzing a large dataset. In this study, we propose two modified functional partial least-squares methods to efficiently estimate the regression coefficient function under the scalar-on-function regression. In the proposed methods, the infinite-dimensional functional predictors are first projected onto a finite-dimensional space using a basis expansion method. Then, two partial least-squares algorithms, based on re-orthogonalization of the score and loading vectors, are used to estimate the linear relationship between scalar response and the basis coefficients of the functional predictors. The finite-sample performance and computing speed are evaluated using a series of Monte Carlo simulation studies and a sugar process dataset.

KEYWORDS

Bidiag1, Bidiag2, bidiagonalization, NIPALS, SIMPLS

1 | INTRODUCTION

Developments in technology and the prevalence of complex and high-dimensional data measured over a continuum, such as time, space, depth, and wavelength, have increased the adoption of functional data analysis (FDA) tools in many disciplines. In recent decades, numerous tools have been developed to analyze such data. See, for example, previous studies[1–5] that discuss a range of theoretical developments and applications in FDA tools. In this study, our focus is restricted to the scalar-on-function regression model (SoFRM), which investigates the functional relationship between a scalar response and functional predictor variables. SoFRM has received significant attention in the literature and has been successfully applied in many fields, such as bioscience, meteorology, chemometrics, and engineering (see, e.g., previous studies[1,3,6–13]).

Let $\mathcal{H}$ denote a separable $\mathcal{L}_2(\mathcal{I})$ Hilbert space of square-integrable and real-valued functions defined on the bounded and closed interval $\mathcal{I}$, $f:\mathcal{I}\to\mathbb{R}$ satisfying $\int_{\mathcal{I}} f^2(t)dt<\infty$. Here, $\mathcal{L}_2(\mathcal{I})$-functions correspond to infinite-dimensional vectors of a finite norm. Let us consider a random sample $\{Y_i,\mathcal{X}_i(t):i=1,...,n\}$ from a population

$\{Y \in \mathbb{R}, \mathcal{X} \in \mathcal{L}_2(\mathcal{I})\}$, where Y is a scalar-valued response variable and $\mathcal{X}$ is a $\mathcal{L}_2$ continuous stochastic process with elements defined in $\mathcal{H}$. Without loss of generality, we assume that both Y and $\mathcal{X}(t)$ are mean-zero processes, that is, $\mathrm{E}[Y] = \mathrm{E}[\mathcal{X}(t)] = 0$, and $t \in [0, 1]$. Then, the SoFRM used to explore the relationship between Y and $\mathcal{X}(t)$ is defined by

$$Y = \int_0^1 \mathcal{X}(t)\beta(t)dt + \epsilon, \tag{1.1}$$

where $\beta(t) \in \mathcal{L}_2[0, 1]$ is the regression coefficient function and ϵ is the random error term having a normal distribution with mean zero and variance σ^2.

The main aim in Model (1.1) is to estimate the regression coefficient function $\beta(t)$, which is a difficult task as $\beta(t)$ belongs to an infinite-dimensional space. It is well known that the use of the least-squares (LS) criterion to estimate $\beta(t)$ leads to the integral Wiener–Hopf equation:

$$\mathrm{E}[Y\mathcal{X}(t)] = \int_0^1 \mathrm{E}[\mathcal{X}(t)\mathcal{X}(s)]\beta(s)ds, \tag{1.2}$$

which is an ill-posed problem because the covariance operator of $\mathcal{X}(t)$ is generally not invertible (see, e.g., Aguilera et al.[9]). Several methods have been proposed to solve the inverse problem in (1.2). The common procedure in these methods is to implement the dimension-reduction paradigm. With this approach, the functional predictor and the regression coefficient function are first projected into a finite-dimensional space via a basis function expansion. Then, the regression model of scalar response on the vector of basis expansion coefficients is used to approximate the regression problem of scalar response on the functional predictor.

Several basis function expansion methods have been developed to project the infinite-dimensional SoFRM in (1.1) into a finite-dimensional space. First, the general basis expansion includes polynomial, Bernstein polynomial, *B*-spline, Fourier, radial, wavelet, and orthogonal basis functions. Using general basis function expansion methods coupled with the LS in estimating SoFRM was first proposed by Ramsay and Silverman.[1] Several extensions have been subsequently proposed to obtain better estimates for the SoFRM. For example, Marx and Eilers[14] used *B*-spline basis to project the functional objects in SoFRM. Then, they used penalized log-likelihood with the *P*-spline framework to estimate the regression parameter. Using *B*-spline basis and ridge regression, Cardot et al[15] proposed an estimator for $\beta(t)$. Matsui et al[16] considered radial basis functions, along with maximum likelihood (ML) and penalized maximum likelihood (PML) methods, to estimate the multivariate version of the SoFRM. Goldsmith et al[17] developed fast-fitting methods for the SoFRM, where the functional predictor is approximated via *B*-spline basis and the regression parameter function is estimated using penalized spline regression. Zhao et al[18] used a wavelet basis to project the functional predictor onto the finite-dimensional space and a LASSO estimator to approximate the regression parameter function. However, general basis function expansion methods may require a large number of basis functions to approximate the functional regression coefficient, causing overfitting of the model and low prediction accuracy. In addition, the use of traditional estimation techniques, such as LS and ML, may produce unstable estimates due to the multicollinearity problem (see, e.g., Matsui et al[19] for more details). Moreover, using estimation methods based on regularization techniques, such as PML, penalized spline regression, and ridge regression, considerably increases the computing time.[20]

Second, functional principal component (FPC) regression projects the functional objects in SoFRM onto finite-dimensional orthonormal FPC bases. The FPC regression is a data-driven method that uses the empirical covariance between the functional predictors when extracting the orthogonal latent components. In other words, it uses the information of functional predictors extracted from their empirical covariance function when constructing FPCs, which makes FPC more informative than the general basis expansion functions. Because the FPC regression method produces orthogonal latent components, it bypasses the multicollinearity problem in SoFRM and can produce efficient parameter estimates in terms of computing time. Thus, many estimation methods in SoFRM have been constructed based on FPC regression (see, e.g., previous studies[10,21–28] and references therein). On the other hand, most of the information in the covariance of functional predictors is captured by a few FPCs, and they are not necessarily important for modeling the response variable. All or some of the most important terms accounting for the interaction between the basis functions and functional predictors might come from higher order principal components.[29] Third, functional partial least squares (FPLS) regression projects the functional objects into orthonormal latent components and the corresponding scores. Unlike FPC regression, FPLS regression uses the information of both the response and predictor variables when projecting the SoFRM into the finite-dimensional space, that is FPLS components are obtained by maximizing the

covariance between the response and functional predictor. Compared with FPCs, FPLS produces more informative components with fewer terms.[29] In addition, the numerical analyses of Aguilera et al[7] showed that FPLS produces more efficient parameter estimates than FPC. Therefore, FPLS is generally preferred over FPC to estimate SoFRM (see, e.g., previous studies[7,9,28–37]). Therefore, in this study, we consider the FPLS method to estimate the SoFRM.

The aforementioned FPLS methods have been built based on two well-known algorithms, SIMPLS and NIPALS (mostly SIMPLS), which require iterative eigendecomposition. Compared with SIMPLS, the NIPALS algorithm requires more computing time (see, e.g., Beyaztas and Shang[20] and Björck and Indahl[38]), and thus, it may be slow or even computationally infeasible for sparse and/or large-scale data sets. On the other hand, while SIMPLS is a faster algorithm than NIPALS, it generally fails to produce numerically precise results in estimating regression coefficients and model fitting.[38,39] In light of these results, while the FPLS regression methods constructed based on the SIMPLS and NIPALS algorithms provide better inferences than the general basis expansion functions and FPC regression, the use of these methods may be insufficient for three cases. (1) When the functional predictor consists of such ultra-dense curves, that is the number of time points per curve is very large. To obtain an improved estimate for the regression parameter function and an improved approximation of the SoFRM, a relatively large number of basis functions may be needed to project the functional predictors onto a finite-dimensional space. (2) A large number of functional predictors are needed in the model to better explain the variation in the scalar response. In this case, the dimension of the model becomes larger compared with only one functional predictor case. (3) The combination of two cases. In such cases, the NIPALS-based FPLS method may be computationally infeasible, while the SIMPLS-based FPLS method may produce unstable regression coefficient estimates and model fitting results.

In this paper, we propose two modified FPLS regression methods to improve the computing speed and numerical precision of FPLS regression in estimating SoFRM. In the proposed methods, the infinite-dimensional functional predictors are first projected onto a finite-dimensional space using a basis expansion method. Then, we consider two PLS algorithms, Bidiag1 and Bidiag2, introduced by Manne,[40] to estimate the regression problem of scalar response on the basis coefficients of the functional predictors. The Bidiag1 and Bidiag2 algorithms are based on Krylov subspace and do not require the predictors' deflation steps. Thus, they are computationally faster than the SIMPLS and NIPALS algorithms. For example, Zhou[41] proposed a fast implementation of the FPLS method based on the Krylov subspace for the function-on-function regression model, the results of which showed that the proposed Krylov subspace-based FPLS method outperforms existing FPLS methods in terms of computing speed. The Bidiag1 and Bidiag2 algorithms include full re-orthogonalization of both score and loading vectors and, thus, they are considered both necessary and sufficient for numerical precision (see, e.g., Björck and Indahl[38]). Our numerical results, discussed in Section 3, reveal that the FPLS methods based on the Bidiag1 and Bidiag2 algorithms produce improved results in terms of computational efficiency. In addition, our results indicate that the proposed methods produce stable regression parameter estimates and model fitting results.

The remainder of this paper is structured as follows. Section 2 presents the methodology. In Section 3, several Monte Carlo experiments and chemometric data analyses are performed, and the results are compared favorably with existing methods. Section 4 concludes the paper, along with some ideas on how the methodology presented here can be extended.

2 | METHODOLOGY

For the SoFRM in (1.1), the FPLS method produces orthogonal components as linear functionals of the functional predictor:

$$\eta = \int_0^1 \mathcal{X}(t)w(t)dt,$$

where $w(t) \in \mathcal{L}_2[0,1]$ is the FPLS weight function defining the FPLS components, which are obtained by optimizing the squared covariance between the scalar response and functional predictor as follows:

$$\underset{\mathrm{w},\,\|\mathrm{w}\|^2=1}{\operatorname{argmax}} \operatorname{Cov}^2\left(\int_0^1 \mathcal{X}(t)w(t)dt, Y\right). \tag{2.3}$$

Let $C_{Y\mathcal{X}}$ and $C_{\mathcal{X}Y}$, respectively, denote the cross-covariance operator, evaluating the contribution of $\mathcal{X}(t)$ to Y and its adjoint as follows:

$$\begin{aligned} C_{Y\mathcal{X}} &: \mathcal{L}_2[0,1] \mapsto \mathbb{R}, f \mapsto \int_0^1 \mathrm{Cov}^2(\mathcal{X}(t), Y) f(t) dt, \\ C_{\mathcal{X}Y} &: \mathbb{R} \mapsto \mathcal{L}_2[0,1], x \mapsto f(t) = x\mathrm{Cov}(\mathcal{X}(t), Y). \end{aligned}$$

From the $\mathcal{L}_2$ continuity of $\mathcal{X}(t)$, we define a self-adjoint, positive, and compact operator $\mathcal{U} = C_{\mathcal{X}Y} \circ C_{Y\mathcal{X}}$.[31] The spectral analyses of $\mathcal{U}$ provide a set of positive eigenvalues, denoted by λ, associated with orthonormal FPLS weight functions $w(t)$ (which are also called eigenfunction) as a solution of

$$\mathcal{U}w = \lambda w. \tag{2.4}$$

Accordingly, the maximization problem defined in (2.3) can be redefined as

$$\max_{w \in \mathcal{L}_2[0,1]} \frac{\langle \mathcal{U}w, w \rangle}{\langle w, w \rangle}.$$

Let us denote by $w_1(t)$ the eigenfunction generated by $\mathcal{U}$ associated to its largest eigenvalue, denoted by λ_1, that is, $\mathcal{U}w_1 = \lambda_1 w_1$. Then, the first FPLS component, denoted by η_1, is defined as follows:

$$\eta_1 = \int_0^1 \mathcal{X}(t) w_1(t) dt.$$

The FPLS is an iterative method. Thus, the subsequent FPLS components are computed iteratively, where at each iteration h, the FPLS component is computed, taking into account the information extracted from the previous iteration. More precisely, let $\mathcal{X}_0 = \mathcal{X}$ and $Y_0 = Y$. In addition, let $\mathcal{X}_h$ and Y_h denote the residuals of the linear regressions of $\mathcal{X}_{h-1}$ on the hth FPLS component η_h and Y_{h-1} on η_h, respectively, as follows:

$$\begin{aligned} \mathcal{X}_h(t) &= \mathcal{X}_{h-1}(t) - p_h(t)\eta_h, \\ Y_h &= Y_{h-1} - c_h \eta_h, \end{aligned}$$

where $p_h(t) = \mathrm{E}[\mathcal{X}_{h-1}(t)\eta_h]/\mathrm{E}\left[\eta_h^2\right]$ and $c_h = \mathrm{E}[Y_{h-1}(t)\eta_h]/\mathrm{E}\left[\eta_h^2\right]$. Then, the h^{th} FPLS component is computed as follows:

$$\eta_h = \int_0^1 \mathcal{X}_{h-1}(t) w_h(t) dt,$$

where the h^{th} FPLS weight function, $w_h(t) \in \mathcal{L}_2[0,1]$ is given by

$$w_h = \underset{w,\, \|w\|^2=1}{\mathrm{argmax}} \mathrm{Cov}^2\left(\int_0^1 \mathcal{X}_{h-1}(t) w(t) dt, Y_{h-1} \right).$$

In other words, $w_h(t)$ is the weight function corresponding to the largest eigenvalue of $\mathcal{U}_{h-1}w_h = \lambda_h w_h$, where $\mathcal{U}_{h-1} = C_{\mathcal{X}Y}^{h-1} \circ C_{Y\mathcal{X}}^{h-1}$ and $C_{\mathcal{X}Y}^{h-1}$ and $C_{Y\mathcal{X}}^{h-1}$ are the cross-covariance operators of $\mathcal{X}_{h-1}(t)$ and Y_{h-1}, respectively.

2.1 | Regression coefficient estimation using basis expansion

The elements of functional predictor variable $\mathcal{X}(t)$ in (1.1) belong to an infinite-dimensional space. However, they are observed in practice in a finite set of discrete-time points. Thus, the functional forms of the elements of discretely observed functional predictors are first reconstructed before fitting the SoFRM. For this purpose, the commonly used practical approach is based on approximating the infinite-dimensional forms of the functional variables in a finite-dimension space spanned by a set of basis functions (see, e.g., Ramsay[1] and Aguilera et al.[9]). With this approach, using

a relatively large number of basis expansion functions K, the elements of the functional predictor can be expressed as a linear combination of basis functions $\phi_k(t) \in \mathcal{L}_2[0,1]$ and the basis coefficient a_{ik} as follows:

$$\begin{aligned} \mathcal{X}_i(t) &= \sum_{k=1}^{K} a_{ik}\phi_k(t),\ i = 1, ..., n, \\ \mathcal{X}(t) &= \sum_{k=1}^{K} \boldsymbol{a}_k\phi_k(t) = \boldsymbol{a}^\top \boldsymbol{\phi}(t), \end{aligned}$$

where $\boldsymbol{a} \in \mathbb{R}^{n\times K} = (a_{ik})_{i=1,...,n;\, k=1,...,K}$ is the matrix of basis coefficients and $\boldsymbol{\varphi}(t) = [\varphi_1, ..., \varphi_K]^\top$. From (2.4), the FPLS weight functions can be expressed in the same basis $\phi(t)$ as follows:

$$\boldsymbol{w}_h(t) = \sum_{k=1}^{K} w_{hk}\phi_k(t) = \boldsymbol{w}_h^\top \boldsymbol{\phi}(t), \quad h \geq 1,$$

where $w_h = [w_{h1}, ..., w_{hK}]^\top$ denotes the vector of basis expansion coefficients. In addition, the regression parameter function $\beta(t)$ can be expressed in the basis $\phi(t)$:

$$\beta(t) = \sum_{k=1}^{K} \beta_k\phi_k(t) = \boldsymbol{\beta}^\top \boldsymbol{\phi}(t),$$

where $\boldsymbol{\beta} = [\beta_1, ..., \beta_K]^\top$.

Let $\boldsymbol{\Phi} = \int_0^1 \boldsymbol{\phi}(t)\boldsymbol{\phi}^\top(t)dt$ be the $K \times K$ symmetric inner product matrix. Let also $\boldsymbol{\Phi}^{1/2}$ denote the square root of $\boldsymbol{\Phi}$. Preda and Saporta[31] and Aguilera et al[9] showed that the FPLS regression of Y on $\mathcal{X}(t)$ is equivalent to the PLS regression of Y on the random design matrix $\boldsymbol{A} = (\boldsymbol{\Phi})\boldsymbol{a}$. Both approaches produce the same PLS components at each step h of the PLS algorithm. At step h of the FPLS algorithm, the FPLS weight function $w_h(t)$ is obtained in terms of basis expansion coefficients as follows:

$$w_h = (\boldsymbol{\Phi}^{-1/2})^\top \tilde{w}_h,$$

where $\tilde{w}_h$, similar to (2.4), is the eigenvector corresponding to the largest eigenvalue of

$$(\boldsymbol{\Phi}^{-1/2})^\top C_{h-1}(\boldsymbol{\Phi}^{-1/2})\tilde{w} = \lambda\tilde{w},\ \tilde{w} \in \mathbb{R}^p,\ \langle \tilde{w}, \tilde{w} \rangle = \mathbb{1},$$

with C_{h-1} being the $K \times K$ matrix with entries $C_{h-1}(j,k) = \mathrm{Cov}(a_{h-1,j}Y_{h-1})\mathrm{Cov}(a_{h-1,k}Y_{h-1}),\ j,k \in 1, ..., K$. In other words, the FPLS weight function $w(t)$ can be obtained in the finite-dimensional space of basis expansion coefficients based on the following eigenvalue/eigenvector problem:

$$C_{h-1}\Phi w = \lambda w,\ w \in \mathbb{R}^p,\ \langle w, \Phi w \rangle = \mathbb{1}.$$

The results presented above indicate that the finite-dimensional problem of the FPLS regression of Y on $\mathcal{X}(t)$ can be reduced to the finite-dimensional PLS regression with predictor $\boldsymbol{A}$ and matrix $\boldsymbol{\Phi}$. Let $\boldsymbol{T}^h$ denote the matrix of first h PLS components obtained from the PLS regression of Y on $\boldsymbol{A}$. Then, the approximate form of Model (1.1) in terms of the first h number of PLS components is expressed as follows:

$$\hat{Y}_h = \mathbb{1}\hat{\gamma}_{0h} + \boldsymbol{T}_h\hat{\boldsymbol{\gamma}}_h = \mathbb{1}\hat{\boldsymbol{\gamma}}_{0h} + \boldsymbol{A}\boldsymbol{\Phi}\hat{\boldsymbol{\beta}}_h,$$

where $\hat{\gamma}_{0h}$ is the estimated intercept, $\hat{\boldsymbol{\gamma}}_h$ is the estimated parameter vector of the the regression of Y on $\boldsymbol{T}_h$, and $\hat{\boldsymbol{\beta}}_h = [\hat{\beta}_{h1}, ..., \hat{\beta}_{hK}]^\top$ is the estimated vector of basis coefficients:

$$\hat{\boldsymbol{\beta}}_h = \left(\boldsymbol{\Phi}^{-1/2}\right)^\top \boldsymbol{V}^\top \hat{\boldsymbol{\gamma}}_h,$$

where $\boldsymbol{V} = [\tilde{w}_1, ..., \tilde{w}_h]^\top$. Finally, the regression parameter function $\beta_h(t)$ is estimated as follows:

$$\hat{\beta}_h(t) = \sum_{k=1}^{K} \hat{\beta}_{hk} \phi_k(t).$$

2.2 | Bidiagonalization and PLS

Many existing PLS approaches are based on two well-known rank-reduction orthogonal projections, that is, the SIMPLS and NIPALS. The NIPALS algorithm performs deflation steps of the predictor, and thus, it may be computationally infeasible. Infeasibility occurs when many functional predictors are used in the model or when many basis expansion functions K are used to reconstruct the functional forms of the discretely observed predictors. On the other hand, while the SIMPLS is a faster algorithm than the NIPALS, it generally fails to produce numerically precise results in estimating regression coefficients and model fitting. To overcome these problems, we consider two alternative algorithms based on the Golub–Kahan bidiagonalization (Bidiag1 and Bidiag2) proposed by Björck and Indahl.[38]

Let us consider the following minimization problem, that is, linear LS problem for the PLS approximation of the SoFRM (1.1):

$$\min \|\boldsymbol{B}\|_2 \text{ such that } \|\boldsymbol{AB} - Y\|_2 = \min, \ \boldsymbol{A} \in \mathbb{R}^K, \ Y \in \mathbb{R}. \tag{2.5}$$

Here, there is no assumption on the size or rank of $\boldsymbol{A}$ so that for any size or rank of $\boldsymbol{A}$, there is a unique solution for the minimization problem (2.5), $\boldsymbol{B}^\dagger = \boldsymbol{A}^\dagger Y$, where $\boldsymbol{A}^\dagger$ denotes the pseudo-inverse of $\boldsymbol{A}$. Similar to Björck,[42] we consider the following subproblem for the PLS approximations $\boldsymbol{B}_m$ (i.e., the approximate m-component PLS solutions as defined by Björck[42]) to the problem (2.5):

$$\min \|\boldsymbol{AB}_m - Y\|_2, \text{ subject to} \boldsymbol{B}_m \in \mathcal{K}_m(\boldsymbol{A}^\top \boldsymbol{A}, \boldsymbol{A}^\top Y), \ m = 1, 2, ..., \tag{2.6}$$

where $\mathcal{K}_m(\boldsymbol{A}^\top \boldsymbol{A}, \boldsymbol{A}^\top Y) = \text{span}\{\boldsymbol{A}^\top Y, (\boldsymbol{A}^\top \boldsymbol{A})\boldsymbol{A}^\top Y, ..., (\boldsymbol{A}^\top \boldsymbol{A})^{m-1}\boldsymbol{A}^\top Y\}$ denotes the Krylov subspace. For $1 \le m \le H$, the subproblem (2.6) has full rank so that the corresponding PLS solution $\boldsymbol{B}_m$ is uniquely defined. Following Lemma 2.1 of Björck,[42] the maximum dimension of the sequence of Krylov subspaces in (2.6) is bounded by the number H of nonzero singular values of $\boldsymbol{A}$. Let $\boldsymbol{U} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{V} \in \mathbb{R}^{K \times K}$ denote the square matrices being orthogonal bases for $\mathbb{R}^n$ and $\mathbb{R}^K$, respectively. What follows hold for any orthogonal basis:

$$\begin{aligned} \boldsymbol{AB} &= Y \Leftrightarrow \boldsymbol{U}^\top \boldsymbol{AB}, \\ &= \boldsymbol{U}^\top Y \Leftrightarrow \boldsymbol{U}^\top \boldsymbol{AVV}^\top \boldsymbol{B}, \\ &= \mathcal{B}\boldsymbol{z} = \boldsymbol{q}, \end{aligned}$$

where $\mathcal{B} = \boldsymbol{U}^\top \boldsymbol{AV}$, $\boldsymbol{z} = \boldsymbol{V}^\top \boldsymbol{B}$, and $\boldsymbol{q} = \boldsymbol{U}^\top Y$. The orthogonal basis matrices generated by PLS are special and such that $\mathcal{B} = \boldsymbol{U}^\top \boldsymbol{AV}$ is bidiagonal. Note also that PLS does not generate the full basis matrices as assumed here, so in the analysis, one has to use the formula in Björck and Indahl.[38] Then, the Krylov subspace is related by

$$\mathcal{K}_m(\boldsymbol{A}^\top \boldsymbol{A}, \boldsymbol{A}^\top Y) = \boldsymbol{V}\mathcal{K}_m(\mathcal{B}^\top \mathcal{B}, \mathcal{B}\boldsymbol{q}).$$

The PLS approximation to the regression parameter is obtained as a solution to $\boldsymbol{B}_m = \boldsymbol{V}_m \boldsymbol{z}_m$, where $\boldsymbol{z}_m$ is the solution of the subproblem

$$\min \|\mathcal{B}\boldsymbol{z} - \boldsymbol{q}\|_2, \text{ subject to } \boldsymbol{z} \in \mathcal{K}_m(\mathcal{B}^\top \mathcal{B}, \mathcal{B}\boldsymbol{q}).$$

In the available FPLS methods, the standard PLS algorithms are used to solve the minimization problem (2.5). For example, let us consider the NIPALS algorithm. It starts with the initial values $\boldsymbol{A}_0 = \boldsymbol{A}$ and $Y_0 = Y$. Then, the NIPALS algorithm is as follows:

$$\begin{aligned}
&\boldsymbol{v}_m = \boldsymbol{A}_{m-1}^{\top} Y_{m-1};\ \boldsymbol{v}_m = \boldsymbol{v}_m / \|\boldsymbol{v}_m\|_2,\ m = 1, 2, ..., \\
&\boldsymbol{u}_m = \boldsymbol{A}_{m-1}\boldsymbol{v}_m;\ \boldsymbol{u}_m = \boldsymbol{u}_m / \|\boldsymbol{u}_m\|_2, \\
&(\boldsymbol{p}_m^{\top}, \zeta_m) = \boldsymbol{u}_m^{\top}(\boldsymbol{A}_{m-1}, Y_{m-1}), \\
&(\boldsymbol{A}_m, Y_m) = (\boldsymbol{A}_{m-1}, Y_{m-1}) - \boldsymbol{u}_m(\boldsymbol{p}_m^{\top}, \zeta_m),
\end{aligned}$$

where $\boldsymbol{u}_m$, $\boldsymbol{v}_m$, and $\boldsymbol{p}_m$ denote the score, loading weight, and loading vectors, respectively. In the NIPALS algorithm, each step requires three matrices and/or vector multipliers and one matrix deflation. When considering the floating-point (flop) arithmetic operations (see, e.g., Björck[42] for more information about the flop), each step of the NIPALS algorithm requires $8nK$ flops per PLS factor, where K denotes the number of basis expansion functions and n denotes the number of observations. In the algorithm, the orthogonal projections $\boldsymbol{u}_m$ are subtracted from the modified data, which can also be expressed via the modified Gram–Schmidt relation; $(\boldsymbol{A}_m, Y_m) = (1 - \boldsymbol{u}_m\boldsymbol{u}_m^{\top})(\boldsymbol{A}_{m-1}, Y_{m-1})$. If we sum the equations in the algorithm, then we have

$$\boldsymbol{A} = \boldsymbol{U}_m\boldsymbol{P}_m^{\top} + \boldsymbol{A}_m,\ Y = \boldsymbol{U}_m\boldsymbol{q}_m + Y_m,$$

where $\boldsymbol{q}_m = [\zeta_1, ..., \zeta_m]^{\top}$, $\boldsymbol{P}_m = [\boldsymbol{p}_1, ..., \boldsymbol{p}_m]^{\top}$, and $\boldsymbol{U}_m = [\boldsymbol{u}_1, ..., \boldsymbol{u}_m]^{\top}$. Accordingly, the PLS approximation of the regression coefficient is given by $\boldsymbol{B}_m = \boldsymbol{V}_m\boldsymbol{z}_m$, where $\boldsymbol{z}_m$ is obtained by solving

$$\begin{aligned}
(\boldsymbol{P}_m^{\top}\boldsymbol{V}_m)\boldsymbol{z}_m &= \boldsymbol{q}_m, \\
\boldsymbol{U}_m^{\top}(Y - \boldsymbol{A}\boldsymbol{V}_m\boldsymbol{z}_m) &= \boldsymbol{0}.
\end{aligned}$$

Herein, the expressions, $\mathcal{B}_m = \boldsymbol{U}_m^{\top}\boldsymbol{A}\boldsymbol{V}_m = \boldsymbol{P}_m^{\top}\boldsymbol{V}_m$, are the upper bidiagonal matrices, and $\mathcal{B}_m^{-1}$ is a full upper triangular matrix and equation $\boldsymbol{B}_m\boldsymbol{z}_m = \boldsymbol{q}_h$ is solved in $2m$ flops by back substitition (see, e.g., Björck and Indahl[38]). It should be noted that in NIPALS a rounding error will cause the matrix $\boldsymbol{P}_m^{\top}\boldsymbol{V}_m$ to be full, and in the standard NIPALS method, the equation $(\boldsymbol{P}_m^{\top}\boldsymbol{V}_m)\boldsymbol{z}_m = \boldsymbol{q}_m$ is solved as a full linear system. As proven by Elden[43] and Björck,[42] the NIPALS produces orthogonal matrices $\boldsymbol{U}_m$ and $\boldsymbol{V}_m$, which form the orthogonal bases for the Krylov subspaces:

$$\mathcal{R}(\boldsymbol{V}_m) = \mathcal{K}_m(\boldsymbol{A}^{\top}\boldsymbol{A}, \boldsymbol{A}^{\top}Y),\ \mathcal{R}(\boldsymbol{U}_m) = \mathcal{K}_m(\boldsymbol{A}\boldsymbol{A}^{\top}, \boldsymbol{A}\boldsymbol{A}^{\top}Y).$$

In addition, Elden[43] and Björck[42] showed that the following relationships hold for the upper bidiagonal $\mathcal{B}_m$:

$$\boldsymbol{A}\boldsymbol{V}_m = \boldsymbol{U}_m\mathcal{B}_m,\ \boldsymbol{A}^{\top}\boldsymbol{U}_m = \boldsymbol{V}_m\mathcal{B}_m^{\top} + \theta_{m+1}\boldsymbol{v}_{m+1}\boldsymbol{e}_m^{\top}, \tag{2.7}$$

where θ_m is the normalizing constant and $\boldsymbol{e}_m$ denotes the m^{th} column of the unit matrix.

From the results presented above, orthogonal bases for the Krylov subspaces play a crucial role in PLS algorithms. Theoretically, a sequence of Krylov vectors can be computed by applying the Gram–Schmidt orthogonalization. These vectors converge to the left and right singular vectors of the singular value of $\boldsymbol{A}$, which makes them rapidly become almost linearly dependent (see, e.g., Björck and Indahl[38] and Björck[42]). As expressed by Björck and Indahl,[38] because the loss of orthogonality grows exponentially with the number of steps, there may be a complete loss of orthogonality after only a few steps, even if a double numerical precision (with numerical precision 10^{-16}) is used by the Krylov vectors. This outcome results in loss of orthogonality in the Gram–Schmidt process and, thus, poor numerical precision for the several PLS algorithms. Consult Björck[42] and Björck and Indahl[38] for more information about the loss of orthogonality in the Krylov vectors and its consequences.

We consider two alternative PLS algorithms (Bidiag1 and Bidiag2) proposed by Björck and Indahl[38] instead of SIMPLS and NIPALS, to approximate the SoFRM (1.1). In the context of approximating (1.1), the procedure of Björck and Indahl[38] provides a direct algorithm for the bidiagonal reduction of rectangular matrix $\boldsymbol{A} \in \mathbb{R}^K$, where the bidiagonal matrices are chosen as products of Householder matrices (see, e.g., Björck[42] and Golub and Kahan[44]). The Householder matrices are elementary orthogonal, and it has been proven that algorithms based on orthogonal transformations with Householder matrices ensure the ideal stability properties (see, e.g., previous studies[45–47]). This bidiagonal reduction approach can easily be used to obtain PLS approximations.

The Bidiag2 algorithm starts with computing $(m = 1)$

$$\begin{aligned} \theta_1 \boldsymbol{v}_1 &= \boldsymbol{A}^\top Y, \\ \rho_1 \boldsymbol{u}_1 &= \boldsymbol{A}\boldsymbol{v}_1. \end{aligned}$$

Then, the following recursion algorithm is used to compute the quantities for $m \geq 2$:

$$\begin{aligned} \theta_m \boldsymbol{v}_m &= \boldsymbol{A}^\top \boldsymbol{u}_{m-1} - \rho_{m-1}\boldsymbol{v}_{m-1}, \\ \rho_m \boldsymbol{u}_m &= \boldsymbol{A}\boldsymbol{v}_m - \theta_m \boldsymbol{u}_{m-1}, \end{aligned}$$

where the normalizing constants ρ_m and θ_m, which are the m^{th} row of the upper bidiagonal $\mathcal{B}$, can be obtained from the normalizing conditions, that is, $\|\boldsymbol{u}_m\|_2 = \|\boldsymbol{v}_m\|_2 = \mathbb{1}$. In matrix form, the Bidiag2 algorithm can be expressed as follows:

$$\begin{aligned} \boldsymbol{V}_m(\theta \boldsymbol{e}_1) &= \boldsymbol{A}^\top Y, \\ \boldsymbol{A}\boldsymbol{V}_m &= \boldsymbol{U}_m \mathcal{B}_m, \\ \boldsymbol{A}^\top \boldsymbol{U}_m &= \boldsymbol{V}_m \mathcal{B}^\top + \theta_{m+1}\boldsymbol{v}_{m+1}\boldsymbol{e}_m^\top, \end{aligned} \tag{2.8}$$

where $\boldsymbol{e}_1$ and $\boldsymbol{e}_m$ are the first and m^{th} basis vectors in $\mathbb{R}^m$. Following Indahl,[48] the loading matrix for $\boldsymbol{A}$ is $\boldsymbol{P}_m = \boldsymbol{A}^\top \boldsymbol{U}_m$ so that multiplying (2.8) by $\boldsymbol{U}_m^\top$ from the left results in a bidiagonal matrix in the form $\mathcal{B}_m = \boldsymbol{P}_m^\top \boldsymbol{V}_m$. For more information, see presious studies.[38,42,48]

As concluded by Björck and Indahl,[38] the use of (full) re-orthogonalization in the Bidiag2 algorithm is essential. In their numerical analyses, Björck and Indahl[38] performed the Bidiag2 algorithm using one-sided re-orthogonalization. Their results showed that one-sided re-orthogonalization results in a substantial loss of numerical precision compared with full re-orthogonalization. This is also reported by Wu and Manne,[49] who concluded that the instability problems associated with the Bidiag2-based PLS could be eliminated using the re-orthogonalization steps. In addition, the numerical analyses of Andersson[39] showed that, while it is computationally fast, the Bidiag2 algorithm without re-orthogonalization results in a substantial loss of numerical precision compared with several other existing PLS algorithms.

In the Bidiag1 algorithm, the procedure starts with computing $(m = 1)$:

$$\begin{aligned} \gamma_1 \boldsymbol{u}_1 &= Y, \\ \alpha_1 \boldsymbol{v}_1 &= \boldsymbol{A}^\top \boldsymbol{u}_1. \end{aligned}$$

Then, for $m \geq 2$, the algorithm uses the following recursion to compute the quantities:

$$\begin{aligned} \gamma_m \boldsymbol{u}_m &= \boldsymbol{A}\boldsymbol{v}_{m-1} - \alpha_{m-1}\boldsymbol{u}_{m-1}, \\ \alpha_m \boldsymbol{v}_m &= \boldsymbol{A}^\top \boldsymbol{u}_m - \gamma_m \boldsymbol{v}_{m-1}, \end{aligned}$$

where the normalizing constants α_m and γ_{m+1} are the elements of the m^{th} column of the lower bidiagonal matrix $\mathcal{G}$ (see, e.g., Björck and Indahl[38]). Similar to Bidiag2, the Bidiag1 algorithm can be represented in the matrix form as follows:

$$\begin{aligned} \boldsymbol{U}_{m+1}(\gamma_1 \boldsymbol{e}_1) &= Y, \\ \boldsymbol{A}\boldsymbol{V}_m &= \boldsymbol{U}_{m+1}\mathcal{G}_m, \\ \boldsymbol{A}^\top \boldsymbol{U}_{m+1} &= \boldsymbol{V}\mathcal{G}_m^\top + \alpha_{m+1}\boldsymbol{v}_m \boldsymbol{e}_m^\top. \end{aligned}$$

Note that the vectors $\boldsymbol{v}_m$ in the Bidiag1 algorithm are equal to those presented in the Bidiag2 algorithm. On the other hand, the vectors $\boldsymbol{u}_m$ are orthogonal basis vectors for the Krylov subspace $\mathcal{K}_m(\boldsymbol{A}\boldsymbol{A}^\top, Y)$ (see, e.g., previous studies[38,42,43]).

Finally, the regression coefficients obtained using the Bidiag1 and Bidiag2 algorithms are obtained as $\boldsymbol{B}_m = \boldsymbol{V}_m \boldsymbol{z}_m$, where $\boldsymbol{z}_m$ satisfies the lower bidiagonal system for Bidiag1 and upper bidiagonal system for Bidiag2 as follows:

$$\mathcal{B}_m \boldsymbol{z}_m = \boldsymbol{q}_m,\ \mathcal{G}_m \boldsymbol{z}_m = \boldsymbol{q}_m.$$

These bidiagonal systems can be solved with minimum computational effort using back substitution (see, e.g., Björck and Indahl[38]). Contrary to the NIPALS and SIMPLS algorithms, no deflation is performed for $\boldsymbol{A}$ in the Bidiag1 and Bidiag2 algorithms. Instead, only the vector-matrix multiplications, that is, $\boldsymbol{AV}$ and $\boldsymbol{AU}$, are used, which reduces the computational cost compared to traditional algorithms. Contrary to the NIPALS algorithm, the Bidiag1 and Bidiag2 algorithms require $8(n-m)(K-m)$ flops per step.

3 | NUMERICAL RESULTS

Several Monte Carlo experiments and empirical analysis of sugar processing data are conducted to compare the proposed Bidiag1- and Bidiag2-based FPLS methods with the existing benchmark SIMPLS- and NIPALS-based FPLS methods in terms of computational efficiency and numerical precision. Björck and Indahl[38] pointed out that the Bidiag1 and Bidiag2 algorithms produce identical results, confirmed by our numerical analysis. Björck and Indahl[38] concluded that Bidiag2 might be the preferred alternative over Bidiag1 as Bidiag2 has slightly simpler implementation. Thus, in our numerical analyses, we only present the results of the Bidiag2-based FPLS method. We note that all the numerical analyses considered in this study are performed using R 4.1.1. on an Intel Core i7 6700HQ 2.6 GHz PC. An example R code for the proposed as well as existing methods can be found at https://github.com/SemanurSaricam/SoFRM_FPLS.

3.1 | Monte Carlo experiments

Throughout the Monte Carlo experiments, a simple SoFRM is considered, and the following process is used to generate the elements of functional predictor:

$$\mathcal{X}(t) = \sum_{k=1}^{4} \alpha_k \phi_k(t),\ t \in [0,1],$$

where $\alpha_k \sim \mathcal{N}(4, 3k^{-1/2})$ and $\phi_k(t) = 4\sin(k\pi t^2) + \cos(4k\pi t^2)$. The regression parameter function is generated using a cosine process as follows:

$$\beta(t) = 2\cos(\pi t),\ t \in [0,1].$$

Then, the values of the response variable are generated as follows:

$$Y = \int_0^1 \mathcal{X}(t)\beta(t)dt + \epsilon,$$

where $\epsilon \sim \mathcal{N}(0,1)$. In the experiments, the elements of the functional predictor are generated at 1001 equally spaced points in the interval $[0,1]$. A graphical display of the generated data and parameter function is presented in Figure 1.

To compare the numerical precision of the methods, we generate $n = 1000$ observations for the functional predictor and scalar response variables and $K = [25, 50, 100, 250, 500, 750]$ cubic B-spline basis functions are used to project the functional observations onto the finite-dimensional space of basis expansion coefficients. For each K, 500 Monte Carlo simulation data with pseudo-random seeds are generated. For each simulation, the following mean squared error (MSE) and mean integrated squared error (MISE) are computed using the first five FPLS components to evaluate the numerical precision of the proposed methods:

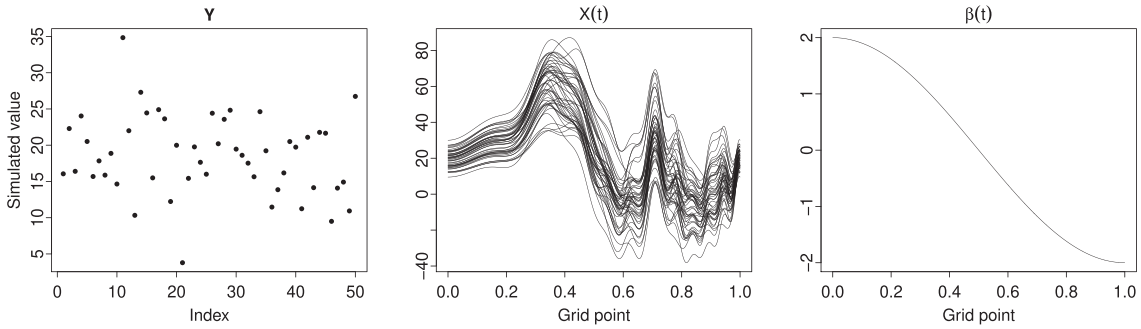

FIGURE 1 Graphical display of the generated scalar response (left panel), functional predictor (middle panel), and regression parameter function (right panel)

TABLE 1 Comparison of numerical precision: Computed mean MSE and MISE values

		Method	
Criterion	***K***	**SIMPLS**	**NIPALS and Bidiag2**
MSE	25	1.0489	1.0189
	50	1.0312	0.9760
	100	0.9767	0.9655
	250	0.9517	0.9474
	500	0.7173	0.7173
	750	0.7173	0.7173
MISE	25	0.7083	0.7083
	50	0.7173	0.7172
	100	0.7175	0.7175
	250	0.7172	0.7172
	500	1.0149	0.9909
	750	1.0244	0.9950

Note: The Bidiag2- and NIPALS-based FPLS methods produced identical results, and thus, their results are given in a joint column.

$$\text{MSE} = \frac{1}{1000}\sum_{i=1}^{1000}(Y_i - \hat{Y}_i)^2,$$

$$\text{MISE} = \int_0^1 [\beta(t) - \hat{\beta}(t)]^2 dt.$$

The mean MSE and MISE values computed over 500 Monte Carlo simulations are presented in Table 1. The results show that the Bidiag2 (and Bidiag1)-based FPLS methods produce the same MSE and MISE values as the NIPALS-based FPLS method. The SIMPLS-based FPLS method produces worse performance compared with other methods. The instability of the SIMPLS method is due to its use of higher order Krylov vectors as the initial vectors (see, e.g., Björck and Indahl[38] for more information about the instability of SIMPLS algorithm).

In addition, we compare the Bidiag2 (and Bidiag1)-based FPLS methods with the SIMPLS- and NIPALS-based FPLS methods in terms of their computing times. While doing so, the computing times of the methods are recorded for only the calculations performed in the finite-dimensional space of B-spline basis expansion coefficients. This is because the process of projecting functional variables into the finite-dimensional space is the same for all the methods. To compare the computing times of the methods, we consider three cases. Case 1: By fixing $n = 1000$ and $h = 5$, the computing times of the methods are recorded for $K = [25, 50, 100, 250, 500, 750]$.

Case 2: By fixing $n = 1000$ and $K = 500$, the computing times of the methods are recorded for $h = [1, 5, 10, 25, 50, 100]$.

Case 3: By fixing $h = 5$ and $K = 500$, the computing times of the methods are recorded for $n = [1000, 2000, 3000, 4000, 5000, 10,000]$.

The recorded computing times for all methods and cases are presented in Figure 2. This figure shows that all the methods have similar computing times when n, K, and h are relatively small. On the other hand, the Bidiag2 (and Bidiag1)-based FPLS methods require less computing than the SIMPLS- and NIPALS-based FPLS methods when one of the components $[n, K, h]$ is large. The SIMPLS and NIPALS algorithms require three times more computing times than the Bidiag2 (and Bidiag1) algorithm for Case 1 (when $K = 750$), 5.4 times more computing time than Bidiag2 (and Bidiag1) for Case 2 (when $h = 100$), and 2.4 times more computing time than Bidiag2 (and Bidiag1) for Case 3 (when $n = 10,000$).

The results of our Monte Carlo experiments indicate that the proposed Bidiag2 (and Bidiag1)-based FPLS method produces stable numerical precision in terms of MSE and MISE, requiring considerably less computing time compared with the existing FPLS methods. Considering that most of the FPLS methods are based on the SIMPLS, which fails to produce numerically precise results, the proposed methods can be considered a favorable alternative to the SIMPLS-based FPLS methods. In addition, based on the computing times results presented in Figure 3, the proposed methods can be considered a faster alternative to the NIPALS-based FPLS methods, and they are recommended especially for ultra-dense and/or multiple functional predictor cases.

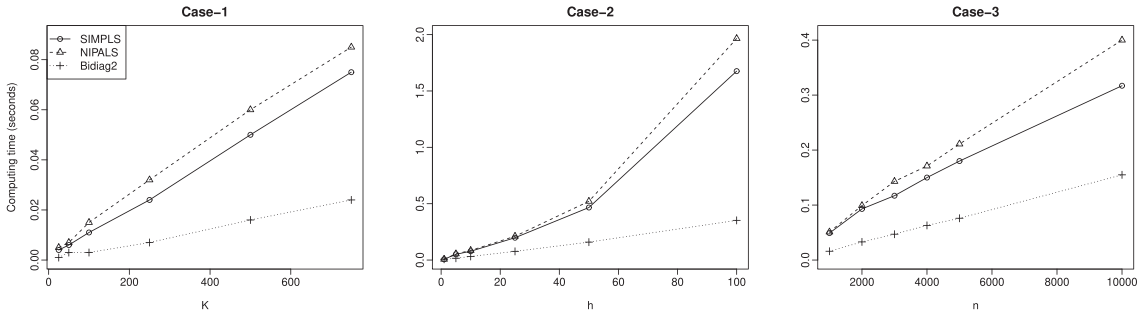

FIGURE 2 Estimated computing times for the SIMPLS-, NIPALS-, and Bidiag2-based FPLS methods for increasing K (left panel), h (middle panel), and n (right panel) values

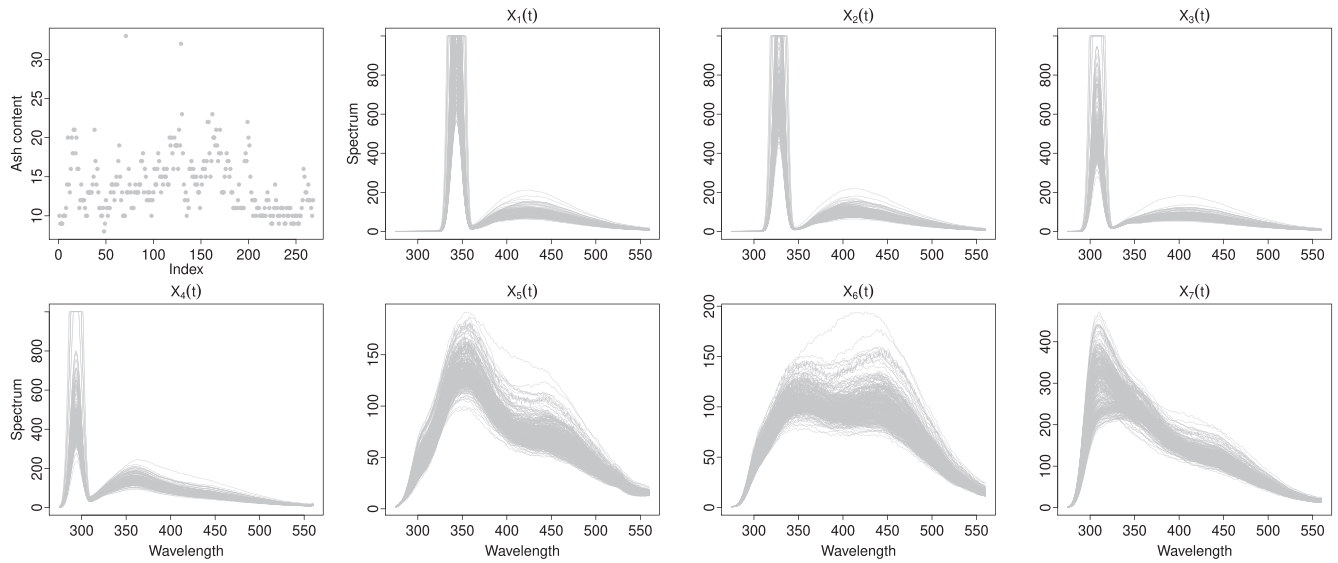

FIGURE 3 A graphical display of the 268 ash content and the emission spectra of sugar samples obtained at excitation wavelengths $[230, 240, 255, 290, 305, 325, 340] = [\mathcal{X}_1(t), \mathcal{X}_2(t), \mathcal{X}_3(t), \mathcal{X}_4(t), \mathcal{X}_5(t), \mathcal{X}_6(t), \mathcal{X}_7(t)]$

3.2 | Sugar process data

The sugar process dataset, initially described by Munck et al[50] and Bro,[51] is available at https://models.life.ku.dk/Sugar_Process. The sugar was dissolved in unbuffered water in this dataset, and the solution was measured spectrofluorometrically. The emission spectra were measured in 0.5 nm intervals from 275 to 560 (571 wavelengths in total) at seven excitation wavelengths (nm) [230, 240, 255, 290, 305, 325, 340] for each of the analyzed 268 sugar samples. In addition, a quality parameter, "ash content," a measure of the number of inorganic impurities in the refined sugar, was measured fluorometrically from the 268 sugar samples. Similar to Smaga and Matsui[52] and Gartheiss et al,[53] our aim with this dataset is to predict the ash content using seven functional predictors (emission spectra of sugar samples obtained at different excitation wavelengths). A graphical display of the ash content and emission spectra of sugar samples is presented in Figure 3.

With the variables in the sugar process data, we consider the following SoFRM:

$$\begin{aligned} Y &= \sum_{m=1}^{7} \int_{275}^{560} \mathcal{X}_m(t)\beta_m(t)dt, \quad t \in [275, 560], \\ &= \int_{275}^{560} \mathcal{X}^{\top}(t)\boldsymbol{\beta}(t)dt, \end{aligned}$$

TABLE 2 The computed MSE values of all the PLS methods for sugar process data

		Method		
K	***h***	**SIMPLS**	**NIPALS**	**Bidiag2**
10	5	13.0601 (**0.002**)	3.1774 (0.003)	3.1774 (**0.002**)
	10	7.4927 (**0.003**)	1.9937 (0.007)	1.9937 (**0.003**)
	15	5.9533 (0.009)	1.7325 (0.010)	1.7325 (**0.004**)
	25	88.6348 (0.010)	1.3863 (0.012)	1.3863 (**0.005**)
	50	263.2891 (0.015)	1.2340 (0.019)	1.2340 (**0.009**)
25	5	16.4864 (**0.003**)	3.2498 (0.004)	3.2498 (**0.003**)
	10	12.7225 (**0.004**)	2.0418 (0.008)	2.0418 (**0.004**)
	15	3.7031 (0.007)	1.6610 (0.011)	1.6610 (**0.005**)
	25	18.7357 (0.013)	0.8592 (0.018)	0.8592 (**0.008**)
	50	195.0939 (0.026)	0.5022 (0.046)	0.5022 (**0.017**)
50	5	36.5621 (0.007)	4.0226 (0.007)	4.0226 (**0.004**)
	10	8.9968 (0.010)	2.0560 (0.013)	2.0560 (**0.007**)
	15	14.4892 (0.013)	1.5631 (0.019)	1.5631 (**0.008**)
	25	17.2403 (0.019)	0.6636 (0.039)	0.6636 (**0.010**)
	50	58.2207 (0.042)	0.0463 (0.080)	0.0463 (**0.024**)
100	5	40.6034 (0.011)	4.1210 (0.014)	4.1210 (**0.007**)
	10	8.4830 (0.015)	1.9945 (0.032)	1.9945 (**0.009**)
	15	1.7659 (0.024)	1.5066 (0.042)	1.5066 (**0.015**)
	25	12.3059 (0.033)	0.4561 (0.068)	0.4561 (**0.020**)
	50	0.7924 (0.081)	0.0005 (0.147)	0.0005 (**0.061**)
250	5	40.7240 (0.027)	4.1154 (0.034)	4.1154 (**0.015**)
	10	10.0299 (0.038)	1.9839 (0.075)	1.9839 (**0.022**)
	15	4.0632 (0.052)	1.3790 (0.125)	1.3790 (**0.029**)
	25	22.2482 (0.095)	0.2930 (0.189)	0.2930 (**0.041**)
	50	0.3186 (0.285)	0.0001 (0.432)	0.0001 (**0.120**)

Note: Recorded computing times (in seconds) are given in brackets, and the fastest methods are marked in bold.

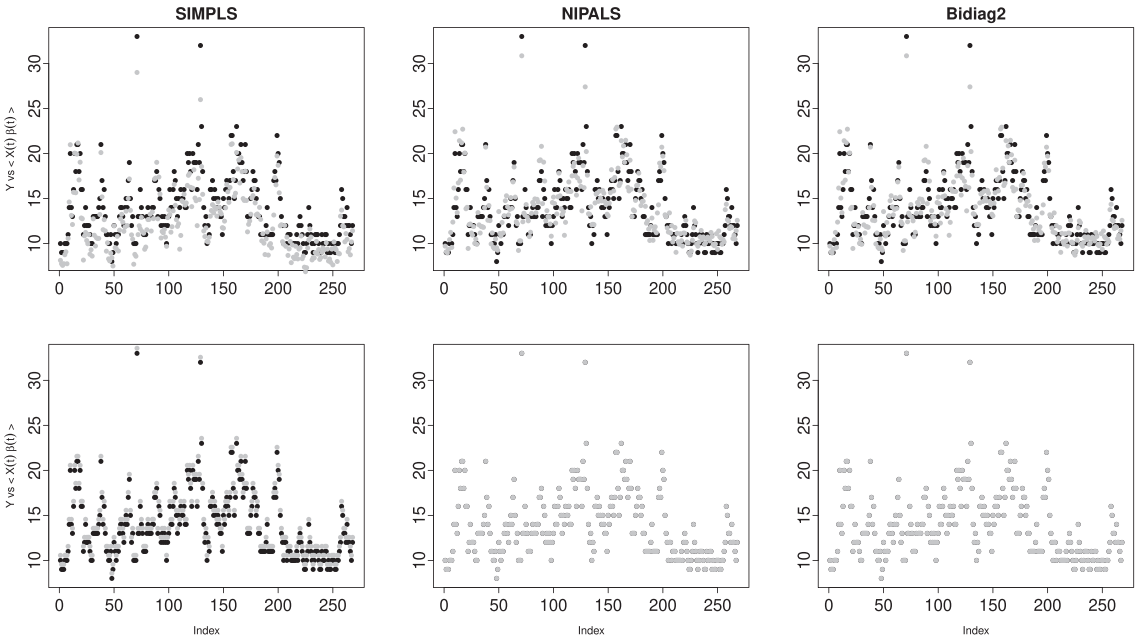

FIGURE 4 Plots Y versus $\langle \mathcal{X}(t)\hat{B}(t)\rangle$ when $[K,h]=[25,15]$ (first row) and $[K,h]=[250,50]$ (second row). In the plots, black points denote the observed ash content, while gray points denote the fitted ash content values by the methods. Note that for the case $[K,h]=[250,50]$ (second row), the observed and fitted ash content values are overlapped for the NIPALS- and Bidiag2-based FPLS methods

where $\mathcal{X}(t)=[\mathcal{X}_1(t),\mathcal{X}_2(t),...,\mathcal{X}_7(t)]^\top$ and $\boldsymbol{\beta}=[\beta_1(t),\beta_2(t),...,\beta_7(t)]^\top$. For this dataset, all the FPLS methods are used to estimate the above method, and the MSE values are computed for different sets of K and h values. In addition, for each K and h, the elapsed times are recorded. Our results are presented in Table 2. According to our results, as in the Monte Carlo experiments, the proposed Bidiag2 (and Bidiag1)-based FPLS method produces similar MSE values to the NIPALS-based FPLS method. The SIMPLS-based FPLS method produces considerably worse performance when compared with the Bidiag2- and NIPALS-based FPLS methods for all K and h values. When a small number of basis functions is used to project the functional predictors (e.g., $K=10$), the SIMPLS-based method produces from 3.43 (when $h=15$) to 213.36 (when $h=50$) times larger MSE values than those of Bidiag2 and NIPALS-based methods. While the predictive performance of the SIMPLS-based method increases with increasing K, it still has poor predictive performance, much worse than the predictive performance of Bidiag2- and NIPALS-based methods. For example, when $K=250$, the SIMPLS-based method produces from 2.94 (when $h=15$) to 3186 (when $h=50$) times larger MSE values than the Bidiag2- and NIPALS-based methods. The numerical instability of the SIMPLS algorithm is also expressed by Andersson[39] and Björck and Indahl,[38] and a possible explanation for the instability of the SIMPLS algorithm is its use of higher order Krylov vectors as the initial vectors. Besides its relatively inferior predictive performance, the SIMPLS-based method is also computationally slower than the proposed Bidiag2 (and Bidiag1)-based FPLS method. The proposed Bidiag2-based method is generally the fastest method.

In addition, we presented the plots of observed and fitted ash content values (i.e., Y vs $\langle \mathcal{X}(t),\hat{B}(t)\rangle$) for two cases, $[K,h]=[25,15]$ and $[K,h]=[250,50]$, to present the model fitting performance of the methods. The results are given in Figure 4. This figure shows that the proposed methods produce improved-fitted values for the ash content when compared to the SIMPLS-based FPLS method. The NIPALS-based FPLS method produces the same results with more computing times than the proposed methods.

4 | CONCLUSION

The SoFRM has become a general framework for investigating the functional relationship between a scalar response and functional predictor(s). Several methods have been proposed to estimate the parameters of this model. The numerical and theoretical studies discussed in Section 1 have demonstrated that the dimension-reduction techniques,

including FPLS, produce improved results in terms of parameter estimates and computing time in the estimation of SoFRM compared with other existing methods. The existing FPLS methods are based on two well-known algorithms, NIPALS and SIMPLS. Among these algorithms, NIPALS may be computationally infeasible when many functional predictors are used in the model or when a large number of basis functions is used to reconstruct the functional forms of the discretely observed predictors. On the other hand, SIMPLS may produce poor performance in parameter estimation and prediction because of its instability.

We introduce two modified FPLS algorithms to stably and efficiently estimate the regression coefficient function in the scalar-on-function regression. The proposed methods are based on the Bidiag1 and Bidiag2 algorithms with reorthogonalized scores and loading vectors introduced by Björck and Indahl.[38] The proposed methods' numerical precision and computational efficiency are evaluated via a series of Monte Carlo experiments and a chemometric data analysis, namely, of sugar process data. We demonstrate that the proposed methods produce stable parameter and model fitting estimates. Our results also demonstrate that, compared with existing FPLS methods, the proposed methods require significantly less computing time. From our numerical analyses, the SIMPLS-based FPLS method produces the (most unstable) worst performance. For instance, in the analysis of the sugar process data, the SIMPLS-based method produces from 2.94 to 3186 times more MSE values than the other methods. Because most of the existing FPLS methods are based on SIMPLS, which fails to produce numerically precise results, the proposed methods can be considered a favorable alternative to the SIMPLS-based FPLS method.

The methodology presented in the study can be extended to other FDA tools. For example, we outline two possible future directions for the proposed methods. First, the proposed methods can be used to estimate function-on-function regression models, where the elements of both the response and predictor variables consist of random curves. The infinite-dimensional function-on-function regression model is approximated by a multivariate regression model. Compared with the SoFRM, the estimation of function-on-function regression models generally requires more computing time since more than one component of the response variable needs to be estimated. The proposed methods can be used to estimate this model more efficiently than existing methods. For example, in such models, $20 \times 20 = 400$ tensor product basis expansion functions are used to approximate the quadratic and interaction effects of the functional predictors if $K = 20$ basis expansion functions are used to approximate a specific functional predictor. Second, the FRM with quadratic and interaction effects of functional predictors requires considerably more basis coefficients and parameters to be estimated in a model (see, e.g., previous studies[54–59]). Therefore, such models require much more computing time than the main effect models in the estimation step. The proposed methods can easily be extended to these models to reduce the computational cost of the estimation phase of such models.

ACKNOWLEDGMENTS

The authors would like to thank two reviewers for their careful reading of our manuscript and valuable suggestions and comments, which have helped us produce an improved version of our manuscript. This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) (Grant 120F270).

PEER REVIEW

The peer review history for this article is available at https://publons.com/publon/10.1002/cem.3452.

DATA AVAILABILITY STATEMENT

Data are available in http://models.life.ku.dk/Sugar_Process.

ORCID

Ufuk Beyaztas https://orcid.org/0000-0002-5208-4950

REFERENCES

1. Ramsay JO, Silverman BW. *Functional Data Analysis*. Springer; 1997.
2. Ramsay JO, Silverman BW. *Applied Functional Data Analysis*. Springer; 2002.
3. Ferraty F, Vieu P. *Nonparametric Functional Data Analysis*: Springer; 2006.
4. Horvath L, Kokoszka P. *Inference for Functional Data With Applications*. Springer; 2012.
5. Cuevas A. A partial overview of the theory of statistics with functional data. *J Stat Plan Inference*. 2014;147:1-23.
6. Cardot H, Ferraty F, Sarda P. Functional linear models. *Stat Probab Lett*. 1999;45:11-22.

7. Aguilera AM, Escabias M, Preda C, Saporta G. Using basis expansions for estimating functional PLS regression applications with chemometric data. *Chemom Intell Lab Syst*. 2010;104:289-305.
8. Ivanescu AE, Staicu AM, Scheipl F, Greven S. Penalized function-on-function regression. *Comput Stat*. 2015;30(2):539-568.
9. Aguilera AM, Aguilera-Morillo MC, Preda C. Penalized versions of functional PLS regression. *Chemom Intell Lab Syst*. 2016;154:8-92.
10. Reiss PT, Goldsmith J, Shang HL, Odgen RT. Methods for scalar-on-function regression. *Int Stat Rev*. 2017;85(2):228-249.
11. Dziak JJ, Coffman DL, Reimherr M, Petrovich J, Li R, Shiffman S, Shiyko MP. Scalar-on-function regression for predicting distal outcomes from intensively gathered longitudinal data: interpretability for applied scientists. *Stat Surv*. 2019;36:150-180.
12. Capezza C, Lepore A, Menafoglio A, Palumbo B, Vantini S. Control charts for monitoring ship operating conditions and CO_2 emissions based on scalar-on-function regression. *Appl Stoch Models Bus Ind*. 2020;36(3):477-500.
13. Beyaztas U, Shang HL. A robust functional partial least squares for scalar-on-multiple-function regression. *J Chemom*. 2022;36(4):e3394.
14. Marx BD, Eilers PHC. Generalized linear regression on sampled signals and curves: a P-spline approach. *Technometrics*. 1999;41(1):1-13.
15. Cardot H, Ferraty F, Sarda P. Spline estimators for the functional linear model. *Stat Sinica*. 2003;13(3):2159-2179.
16. Matsui H, Araki Y, Konishi S. Multivariate regression modeling for functional data. *J Data Sci*. 2008;6:313-331.
17. Goldsmith J, Bobb J, Crainiceanu CM, Caffo B, Reich D. Penalized functional regression. *J Comput Graphical Stat*. 2011;20(4):830-851.
18. Zhao Y, Odgen RT, Reiss PT. Wavelet-based LASSO in functional linear regression. *J Comput Graphical Stat*. 2012;21(3):600-617.
19. Matsui H, Kawano S, Konishi S. Regularized functional regression modeling for functional response and predictors. *J Ind Math*. 2009;1:17-25.
20. Beyaztas U, Shang HL. On function-on-function regression: partial least squares approach. *Environ Ecol Stat*. 2020;27(1):95-114.
21. Yao F. Functional principal component analysis for longitudinal and survival data. *Stat Snica*. 2007;17(3):965-983.
22. Hall P, Horowitz JL. Methodology and convergence rates for functional linear regression. *Ann Stat*. 2007;35(1):70-91.
23. Lee ER, Park BU. Sparse estimation in functional linear regression. *J Multivariate Anal*. 2012;105(1):1-17.
24. Maronna RA, Yohai VJ. Robust functional linear regression based on splines. *Comput Stat Data Anal*. 2013;65:46-55.
25. Goldsmith J, Scheipl F. Estimator selection and combination in scalar-on-function regression. *Comput Stat Data Anal*. 2014;70:362-372.
26. Wang J-L, Chiou J-M, Müller H-G. Functional data analysis. *Ann Rev Stat Appl*. 2016;3:257-295.
27. Shin H, Lee S. An RKHS approach to robust functional linear regression. *Stat Sinica*. 2016;26:255-272.
28. Kalogridis I, Van Aelst S. Robust functional regression based on principal components. *J Multivariate Anal*. 2019;173:393-415.
29. Delaigle A, Hall P. Methodology and theory for partial least squares applied to functional data. *Ann Stat*. 2012;40(1):322-352.
30. Saporta G. Mèthodes exploratoires d'analyse de données temporelles. *Cahiers du BURO Université Pierre et Marie Curie*. 1981;37–38:7-194.
31. Preda C, Saporta G. PLS regression on a stochastic process. *Comput Stat Data Anal*. 2005;48(1):149-158.
32. Preda C, Saporta G, Lévéder C. PLS classification for functional data. *Comput Stat*. 2007;22(2):223-235.
33. Reiss PT, Ogden RT. Functional principal component regression and functional partial least squares. *J Am Stat Assoc: Theory Methods*. 2007;102(479):984-996.
34. Kramer N, Boulesteix A-L, Tutz G. Penalized partial least squares with applications to b-spline transformations and functional data. *Chemom Intell Lab Syst*. 2008;94(1):60-69.
35. Aguilera AM, Escabias M, Valderrama MJ, Aguilera-Morillo MC. Functional analysis of chemometric data. *Open J Stat*. 2013;3(5):334-343.
36. Yu D, Kong L, Mizera I. Partial functional linear quantile regression for neuroimaging data analysis. *Neurocomputing*. 2016;195:74-87.
37. Febrero-Bande M, Galeano P, Gonzalez-Manteiga W. Functional principal component regression and functional partial least-squares regression: an overview and a comparative study. *Int Stat Rev*. 2017;85(1):61-83.
38. Björck A, Indahl UG. Fast and stable partial least squares modelling: a benchmark study with theoretical comments. *J Chemom*. 2017;31(8):e2898.
39. Andersson M. A comparison of nine PLS1 algorithms. *J Chemom*. 2009;23(10):518-529.
40. Manne R. Analysis of two partial-least-squares algorithms for multivariate calibration. *Chemom Intell Lab Syst*. 1987;2:187-197.
41. Zhou Z. Fast implementation of partial least squares for function-on-function regression. *J Multivariate Anal*. 2021;185:104769.
42. Björck A. Stability of two direct methods for bidiagonalization and partial least squares. *SIAM J Matrix Anal Appl*. 2014;35(1):279-291.
43. Elden L. Partial least-squares vs. Lanczos bidiagonalization—I: analysis of a projection method for multiple regression. *Comput Stat Data Anal*. 2004;46(1):11-31.
44. Golub GH, Kahan W. Calculating the singular values and pseudoinverse of a matrix. *SIAM Ser B*. 1965;2(2):205-224.
45. Higham NJ. *Accuracy and Stability of Numerical Algorithms*. SIAM; 2002.
46. Byers R, Xu H. A new scaling for newton's iteration for the polar decomposition and its backward stability. *SIAM J Matrix Anal Appl*. 2008;30(2):822-843.
47. Smoktunowicz A, Wrobel I. Numerical aspects of computing the Moore-Penrose inverse of full column rank matrices. *BIT Comp Sci Numer Math*. 2012;52(2):503-524.
48. Indahl UG. The geometry of PLS1 explained properly: 10 key notes on mathematical properties and some alternative algorithmic approaches to PLS1 modelling. *J Chemom*. 2014;28(3):168-180.
49. Wu W, Manne R. Fast regression methods in a Lanczos (or PLS-1) basis. Theory and applications. *Chemom Intell Lab Syst*. 2000;51(2):145-161.

50. Munck L, Nørgaard L, Engelsen SB, Bro R, Andersson CA. Chemometrics in food science—a demonstration of the feasibility of a highly exploratory, inductive evaluation strategy of fundamental scientific significance. *Chemom Intell Lab Syst*. 1998;44(1–2):31-60.
51. Bro R. Exploratory study of sugar production using fluorescence spectroscopy and multi-way analysis. *Chemom Intell Lab Syst*. 1999; 46(2):133-147.
52. Smaga L, Matsui H. A note on variable selection in functional regression via random subspace method. *Stat Methods Appl*. 2018;27(3): 455-477.
53. Gartheiss J, Maity A, Staicu A-M. Variable selection in generalized functional linear models. *Stat*. 2013;2(1):86-101.
54. Usset J, Staicu A-M, Maity A. Interaction models for functional regression. *Comput Stat Data Anal*. 2016;94:317-329.
55. Fuchs K, Scheipl F, Greven S. Penalized scalar-on-functions regression with interaction term. *Comput Stat Data Anal*. 2015;81:38-51.
56. Luo R, Qi X. Interaction model and model selection for function-on-function regression. *J Comput Graphical Stat*. 2019;28(2):309-322.
57. Matsui H. Quadratic regression for functional response models. *Econ Stat*. 2020;13:125-136.
58. Sun Y, Wang Q. Function-on-function quadratic regression models. *Comput Stat Data Anal*. 2020;142:106814.
59. Beyaztas U, Shang HL. A partial least squares approach for function-on-function interaction regression. *Comput Stat*. 2021;36(2): 911-939.

How to cite this article: Saricam S, Beyaztas U, Asikgil B, Shang HL. On partial least-squares estimation in scalar-on-function regression models. *Journal of Chemometrics*. 2022;e3452. doi:10.1002/cem.3452