Variable Selection for Scalar-on-function Regression Models

Yuhang Yang

May 20, 2022

Abstract

Variable selection in regression models has become an important topic in many application fields, due to a great increase in data availability. In this project, we consider scalar-on-function regression models, in which the predictors can be functional data, i.e. curves. Although grouped versions of some shrinkage methods have been recently adapted to these models to select relevant functional predictors, these methods have not been tested on models with many functional predictors yet. We study two functional variable selection methods, one based on the group lasso penalty, and another one based on the group SCAD penalty, and we compare their selection results through simulation studies. Results show effectiveness in the lasso-based method but issues arise with the other method. Possible reasons are briefly discussed and might require further research in the future.

1 Introduction

Variable selection has been a commonly studied area in regression analysis and is generally introduced in many statistics textbooks. Classic variable selection methods include subset selection, best subset selection and stepwise selection: other more recent popular methods include shrinkage methods, for example ridge and lasso. In recent decades, grouped version of these shrinkage methods have become popular and proven to be effective in selecting predictors that have some pre-defined grouping structures (see e.g., Yuan and Lin, 2006 [1]; Meier et al., 2008 [2]).

While the literature mainly focuses on linear regression analysis and generalized linear models, some recent studies have extended these methods to functional data analysis. In functional regression models, the predictors and/or the response, which are usually not observed directly, instead; they are observed at certain time points and usually with noise. There are three types of functional linear models, based on which variables are functional: functional response and scalar predictors, scalar response and functional predictors, and both functional predictors and functional response. In this project, we focus on variable selection for scalar-on-function regression models. Recent studies adapting group lasso penalty (Gertheiss et al., 2013) [3] and group SCAD penalty (Matsui and Konishi, 2011) [4] to scalar-on-function regression have shown some success. However, the models considered in the simulations and real data examples of these articles contain only a small number of

predictors. We are interested in the performance of these two methods in more complex models and in comparing them. Therefore, we carry out a simulation study based on the design in Gertheiss et al. (2013) [3], we analyze the results and further identify some potential issues and limitations.

The remainder of this report is given as follows. In section 2, we outline methods of ridge, lasso and grouped lasso. In section 3, we introduce functional data analysis and functional linear models. We then discuss the case where we have a scalar response and functional independent variables, and we explain in detail two variable selection methods. Section 4 focuses on our simulation studies, where we extend the studies performed in the two reference articles to more complex models to evaluate the performance and compare the two methods. Some brief discussion is included in this section. Section 5 concludes this project.

2 Shrinkage methods

This section is based on Hastie et al. (2017) [5] and James et al. (2017) [6]. In general, if we are interested in studying and summarizing the relationship between a dependent variable and one or more independent variables, we could consider using linear regression models. By fitting the model, we can obtain a set of coefficient estimates, which would allow us expressing the average change in the dependent variable if there is an increase or decrease in the independent variables. Estimated coefficients can also be used for making prediction on data that we have not seen yet. The variable we are interested in predicting is called the response variable, while independent variables are called predictors, explanatory variables, regressors, or covariates. A common representation of the linear regression model is:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon, \tag{1}$$

where Y is the quantitative response variable, X_1,...,X_p are a set of predictors, and ϵ is the error term. $X_i's$ can be either categorical or quantitative. In the case of a categorical variable with d levels (categories), we need to transform it into $d-1$ dummy variables to be included in the model. Dummy variables take the form of 0 or 1, and represent the presence of certain level associate with the categorical variable. The error ϵ is assumed to be independent and identically distributed and have a normal distribution with mean 0 and variance σ^2. $\beta_i's$ are the unknown theoretical coefficients. One important goal in regression is to estimate the coefficients, as well as the unknown population variance σ^2. We frequently use ordinary least squares (OLS) method to obtain the estimations. Assume we have a sample of n observations $(x_{1i}, ..., x_{pi}, y_i)$ for $i = 1, ..., n$. The multiple linear regression minimization problem can be written as:

$$\hat{\beta}^{\text{OLS}} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ji}\beta_j \right)^2. \tag{2}$$

$\hat{\beta}^{\text{OLS}}$ has closed-form solution $\hat{\beta} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$, where $\mathbf{X}$ is an $n \times (p+1)$ the design matrix with 1 in first column and the rest of columns corresponding to the observed values of $X_1, \ldots, X_p$, and $\mathbf{y}$

is a vector of length n consisting of observed $y_1, \ldots, y_n$.

Even if OLS estimator is unbiased (i.e., E[$\hat{\beta}$]=β) and it is optimal in the class of linear unbiased estimators under the linear regression model hypothesis, it has issues in some cases. For example, multicollinearity causes a major problem in the OLS estimator. Indeed, if we have two or more predictors that are highly correlated with each other, then $\hat{\beta}_i^{\text{OLS}}$ will have high variance, resulting in unstable estimates and poor performance in prediction. Another case in which OLS have issues is when we have a large number of predictors and the model estimated by OLS can easily overfit the data and produce inaccurate prediction. In addition, if the number of predictors exceeds the number of observations, OLS might not be unique. A possible solution to these cases is to employ shrinkage methods, in which we add a penalty term to the minimization problem in (2).

Shrinkage methods add constrains to the coefficient estimates, which would shrink the coefficient estimates toward zero or exactly zero depending on the method. In this project, we consider ridge, lasso, group lasso, SCAD (smoothly clipped absolute deviation) and we also briefly introduce group SCAD. The SCAD penalty can be viewed as an improved version of lasso penalty with less shrinkage to larger coefficients. The grouped versions of these methods are useful when, instead having individual predictors, we have pre-defined groups of predictors; the grouped approach consists in modifying the penalty function so that the penalty is applied to each group of predictors. These shrinkage methods result in biased estimators of the coefficients, however they often improve the fit of the regression model by reducing the variance with respect to the OLS estimator, allowing us to build a model that has better prediction accuracy. In addition, the penalty term allows to estimate the coefficients also when we have more predictors than observations ($n > p$). Finally, some of these shrinkage methods also permit to automatically perform variable selection.

2.1 Ridge regression

Ridge estimation of the regression model parameters $\beta_0, ..., \beta_p$ can be written in the form:

$$\hat{\beta}^{\text{ridge}} = \underset{\beta}{\text{argmin}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}, \tag{3}$$

where λ is a tuning parameter. When $\lambda = 0$, ridge estimates are equivalent to the OLS estimates. Increasing λ shrinks coefficients more toward zero, and λ is generally determined by cross validation. Note that, when we apply ridge method, penalty is not applied to the intercept. This is because the intercept is simply the expected value of Y when all $X_i's$ are zero and is not associated with any predictor. Unlike least squares estimates, ridge estimates are not scale equivariant, meaning that the estimates obtained by ridge method can be significantly impacted by multiplying any predictor by a constant. Such scaling changes the sum of squares of the coefficients and results in unevenly penalized coefficients. Therefore, data input are usually standardized before applying the ridge regression so that each predictor has mean zero and variance one. Note that the constraint in ridge regression is the ℓ_2 norm of the vector of coefficients and ridge regression does not perform variable selection. Indeed,

coefficient estimates are shrank toward zero but will never reach exactly zero. Ridge estimator has the following closed form solution:

$$\hat{\beta}^{\text{ridge}} = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}, \tag{4}$$

where $\mathbf{X}$ and $\mathbf{y}$ are the design matrix and the response vector previously defined. Bias-variance tradeoff applies to the ridge regression: by adding the penalty term to the objective function in the minimization problem, we better control the variance of the coefficient estimator, trading a small bias for a lower variance. Indeed, ridge estimator is biased since the expected value of $\hat{\beta}^{\text{ridge}}$ is not β. As λ increases, the bias increases and variance decreases. The λ selected by cross validation should achieve the smallest value of $bias^2 + variance^2$.

2.2 Lasso regression

Similarly to ridge, lasso estimates can be written as:

$$\hat{\beta}^{\text{lasso}} = \underset{\beta}{\text{argmin}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}, \tag{5}$$

where λ is a tuning parameter which is usually optimized through cross validation. As in the ridge regression, $\lambda = 0$ gives the OLS estimates and increasing λ shrinks coefficients towards zero. Also as in the ridge regression, no penalty is added to the intercept and data standardization is usually applied before applying the lasso. However, the ℓ_1 norm of the vector of coefficients is used as penalty in the lasso; as a consequence, the lasso performs variable selection. Indeed, as λ increases some coefficient estimates will become exactly zero. Bias-variance tradeoff also applies for lasso: the estimator is biased but has lower variance. Importantly, lasso estimator does not have a closed form solution, hence computational optimization techniques are employed to numerically compute the solution.
This is an example plot of the lasso output (Figure 1). The credit dataset was introduced in James et al. (2017) [6] and considers 10 variables to predict credit card balance. As λ increases, some coefficient estimates become exactly zero, that is, the corresponding predictor is not included in the model.

2.3 Group lasso

Lasso performs well in cases where we have multiple predictors but only a few of them are indeed associated with the response. Group lasso represents a generalization of the lasso method. It can be applied when the predictors belong to some pre-defined groups, and we want to either include or exclude all the variables belonging to the same group. An intuitive example would be categorical variables with many levels. Assume we have a categorical variable with d levels, then we would include $d-1$ dummy variables in the model. For each dummy variable, it is 0 if the corresponding level is not observed and is 1 if it is observed in the data. In this case, it would not make sense to keep only

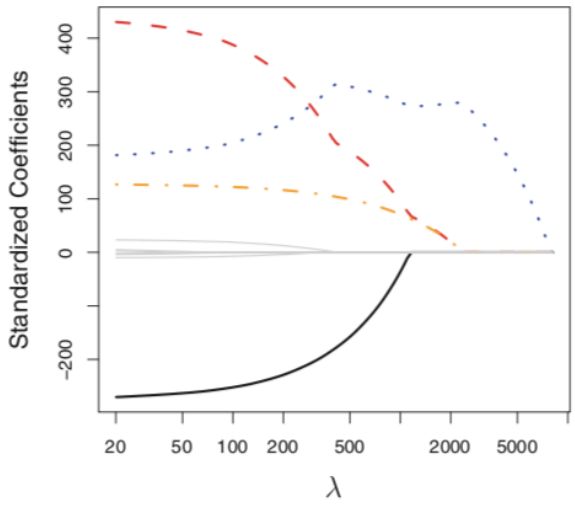


Figure 1: Lasso example output plot from credit. Each line shows the value of a standardized coefficient as a function of the parameter λ. As λ increases, some of the standardized coefficients become exactly zero. Reprinted from An introduction to statistical learning: With applications in R. (p. 220), by James et al., 2017, Springer.[6]

one or two dummy variables in the model; instead, we should either keep or remove the entire group of dummy variables generated from the same categorical variable. The group lasso problem can be represented as:

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \sum_{l=1}^{L} x_{ijl} \beta_{jl} \right)^2 + \lambda \sum_{l=1}^{L} \left(\beta_l' K_l \right)^{1/2} \right\}, \tag{6}$$

where l indexes to the l^{th} group, and K_l is a positive definite matrix (and can be the identity matrix).

The group lasso was studied along with group least angle regression selection (LARS) and group non-negative garrotte in regression with grouped variables by Yuan and Lin (2006) [1]. Based on the simulation results and a real data example, group lasso performed effectively. But due to the piecewise nonlinear solution path, group lasso may have higher computational cost in large scale problems (Yuan and Lin, 2006) [1]. Group lasso for logistic regression was studied by Meier et al. (2008) [2]. Another effective algorithm was proposed based on the group lasso. In the simulation study, group lasso favored large models and included many noise variables. In the real data example case, group lasso had the best prediction performance (Meier et al., 2008 [2]).

2.4 SCAD penalized regression

The SCAD penalty was proposed by Fan and Li (2001) [7] and the resulting estimator has many good properties simultaneously, namely the unbiasedness, sparsity and continuity. Note again that

the lasso and ridge are both biased estimators. Lasso has sparsity property whereas ridge does not have, and they are both continuous estimators. In the multiple linear regression setting (without grouped variables), the SCAD penalized least squares can be written as:

$$\hat{\beta}^{\text{SCAD}} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} P(|\beta_j|) \right\}. \tag{7}$$

The SCAD penalty is the continuous differentiable penalty function $P_\lambda(|\beta_j|) = \lambda P(|\beta_j|)$ defined by:

$$P'_\lambda(|\beta_j|) = \lambda \left\{ I(|\beta_j| \leq \lambda) + \frac{(a\lambda - |\beta_j|)_+}{(a-1)\lambda} I(|\beta_j| > \lambda) \right\}, \tag{8}$$

where λ and $a > 2$ are a pair of tuning parameters and can be selected by cross validation. However, performing such cross validation is usually computational expensive in practice. Based on the Bayesian statistical point of view and simulation, Fan and Li (2001) [7] suggested $a = 3.7$ would be a reasonable choice for most problems. SCAD solution is given by:

$$\hat{\beta}_j^{SCAD} = \begin{cases} \left(\left| \hat{\beta}_j^{\text{OLS}} \right| - \lambda \right)_+ \operatorname{sign}\left(\hat{\beta}_j^{\text{OLS}} \right) & \text{if } \left| \hat{\beta}_j^{\text{OLS}} \right| \leq 2\lambda; \\ \left\{ (a-1)\hat{\beta}_j^{\text{OLS}} - \operatorname{sign}\left(\hat{\beta}_j^{\text{OLS}} \right) a\lambda \right\} / (a-2) & \text{if } 2\lambda < \left| \hat{\beta}_j^{\text{OLS}} \right| \leq a\lambda; \cdot \\ \hat{\beta}_j^{\text{OLS}} & \text{if } \left| \hat{\beta}_j^{\text{OLS}} \right| > a\lambda \end{cases} \tag{9}$$

Compared with other shrinkage method, SCAD estimator has advantage of unbiasedness, meaning that for large coefficients, they would be shrunk less severely and achieve nearly unbiased estimator. SCAD penalty can also perform variable selection and produce a sparse set of solution.

The group SCAD penalty developed by Wang et al. (2007) [8] works by modifying (7) to adjust for the pre-defined group structure of predictors. Similar to group lasso penalty, let K_j denote to group sizes of a predictor X_j. The minimization problem is:

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \sum_{k=1}^{K} x_{ij\ell}\beta_{j\ell} \right)^2 + \lambda \sum_{j=1}^{p} \sum_{k=1}^{K} P_\lambda \left(\|\beta_{jk}\|_2 \right) \right\}. \tag{10}$$

Group SCAD has analogous form $\min_\theta \left\{ \|z - \theta\|^2 + P_\lambda(\|\theta\|) \right\}$. The penalty is defined as:

$$P_\lambda(|\beta|) = \begin{cases} \lambda|\beta| & \text{if } |\beta| \leq \lambda \\ -\frac{\left(|\beta|^2 - 2a\lambda|\beta| + \lambda^2\right)}{2(a-1)} & \text{if } \lambda < |\beta| < a\lambda \\ \frac{(a+1)\lambda^2}{2} & \text{if } |\beta| > a\lambda \end{cases} . \tag{11}$$

$a = 3.7$ is also used for group SCAD penalty.

3 Functional data analysis

This section is based on Kokoszka and Reimherr (2017) [9] and Ramsay and Silverman (2006) [10]. Functional Data Analysis (FDA) refers to the statistical analysis of data which can be considered

as smooth curves (or smooth surfaces). Such data arise commonly as longitudinal data and time series data. For example, children growth data that repeatedly measures heights of the same group of children at different ages can be considered as functional data and analyzed with FDA.

In general, we do not observe the functional datum directly. On the contrary, we only have noisy, discrete observations of it. In particular, we observe the smooth curve $x(t)$ only on a discrete grid of points $t_1, ..., t_J$, with measurement error, i.e. we have $y_j = x(t_j) + \epsilon_j$, where y_j are the observed values and ϵ_j are the measurement errors. As a consequence, the first step in functional data analysis consists in computing the curve $x(t)$ from raw data (t_j, y_j) for $j = 1, ..., J$.

Basis expansion is commonly employed for computing curves from raw data. Basis function procedures take the form $x(t) \approx \sum_{k=1}^{K} c_k \phi_k(t)$, where ϕ_1,...,ϕ_k are K basis functions and c_1,...,c_k are coefficients, generally obtained from the observed data by smoothing techniques. There exists several basis systems in literature, among which we have the Fourier basis and the B-splines basis. Fourier basis system models the periodic behavior of the curves:

$$\hat{x}(t) = c_0 + c_1 \sin \omega t + c_2 \cos \omega t + c_3 \sin 2\omega t + c_4 \cos 2\omega t + \ldots, \tag{12}$$

$\phi_0(t) = 1, \phi_{2r-1}(t) = \sin r\omega t, \phi_{2r}(t) = \cos r\omega t$ and $\omega = \frac{2\pi}{T}$, where T is the period. In particular, B-splines basis functions are commonly employed to model non-periodic curves and consists of piecewise polynomials on subintervals separated by knots. Additional basis systems include wavelets, exponential and power bases, step-function basis, etc.

After the basis functions have been determined, we can use smoothing methods to find c_1,...,c_k. There are, in general, two smoothing methods that are commonly used: regression splines and roughness penalty. The first one uses OLS, and the objective function to minimize is

$$\sum_{j=1}^{J} \left[y_j - \sum_{k=1}^{K} c_k \phi_k (t_j) \right]^2. \tag{13}$$

In regression splines, the level of smoothing can be changed by tuning the number of basis K: small K leads to a very smooth curve $\hat{x}(t)$ and large K leads to a more wiggly curve $\hat{x}(t)$. The estimate $\hat{\mathbf{c}}$ is $\hat{\mathbf{c}} = (\mathbf{\Phi}'\mathbf{\Phi})^{-1} \mathbf{\Phi}'\mathbf{y}$, where $\mathbf{\Phi}$ is defined as a $J \times K$ matrix consisting of $\phi_k(t_j)$. The second one is similar to shrinkage methods where we add a penalty term to the objective function to be minimized. For the penalty, we mainly consider taking the second order derivative of basis expansion, which can be written as:

$$\text{PEN}_2 (x) = \int \left(D^2 x\right)^2 = \int \left[\sum_{k=1}^{K} c_k D^2 \phi_k(t) \right]^2 dt. \tag{14}$$

Then the objective function that we minimize is:

$$\sum_{j=1}^{J} \left(y_j - c_1\phi_1(t) - \cdots - c_K\phi_K(t)\right)^2 + \lambda \int \left(c_1\phi_1''(t) + \cdots + c_K\phi_K''(t)\right)^2 dt, \tag{15}$$

where λ is a tuning parameter as in the shrinkage methods setting. In this case, we usually employ a large number of basis K, while λ determines the level of smoothness of the resulting curve $\hat{x}(t)$:

larger $\lambda's$ correspond to smoother curves and the optimal value of lambda is usually selected via cross-validation.

3.1 Functional linear models

Functional linear models can be viewed as an extension of linear models to functional data, where either the response variable or one or more predictors, or a combination of both are functional. In this subsection, we briefly introduce three cases: a functional response and a scalar independent variable, a scalar response and a functional independent variable, and a functional response and a functional independent variable.
Consider the following three models corresponding to the above cases:

$$Y(t) = \beta_0(t) + \beta(t)X + \epsilon(t) \tag{16}$$

$$Y = \beta_0 + \int_0^T \beta(s)X(s)ds + \epsilon \tag{17}$$

$$Y(t) = \beta_0(t) + \int_0^T \beta(s,t)X(s)ds + \epsilon(t). \tag{18}$$

The focus of this project will be the scalar response case with multple functional predictors. For simplicity, we start considering the case of only one functional predictor, as in equation (16).

To obtain the estimates for the scalar-on-function regression model, the simplest approach is to discretize the covariate function. Then we can write the model as:

$$y_i = \beta_0 + \sum_{j=1}^{J} X_{ij}\beta_j + e_i \quad i = 1, 2, \ldots, n. \tag{19}$$

This will generate a system with many unknowns, which will often result in an infinite sets of solutions if the sample size is not big enough with respect to J. Hence, this approach is often unrealistic to use. Resulting infinite sets of possible solutions and is often unrealistic to use. A potential solution is to consider the functional variable on a coarser time scale, which would reduce the number of parameters we need to estimate. However, there are still problems with this method. The model generated will be difficult to interpret and will use up degrees of freedom in the data. Thus, using regularization methods is a necessary step in scalar-on-function regression.

There are two types of regularization methods, which can be employed to fit a scalar-on-function model. The first one consists in using restricted basis functions. Specifically, the regression coefficients $\beta's$ can be written as $\beta(s) = \sum_{k=1}^{K_\beta} b_k\theta_k(s)$ and the functional variable $X(s)$ can be approximated as $X(s) \approx \sum_{k=1}^{K_x} c_k\phi_k(s)$, where $\theta_k(s)$ and $\phi_k(s)$ are sets of basis functions, b_k and c_k are coefficients. K_β and K_x are the numbers of basis functions. Then we can write $\hat{y}_i = \beta_0 + \int_0^T X(s)\beta(s)ds = \beta_0 + \int_0^T \mathbf{C}\phi(s)\theta(s)'\mathbf{b}ds$, where $\mathbf{C}$ is a coefficient matrix with dimensions $J \times K_x$ and $\mathbf{b}$ is a vector of length K_β. The choice of basis system and K depends on the data. In general, we want to choose K that is not too large, so that the model is interpretable and has reasonable degrees of freedom.

The second regularization method consists in applying a roughness penalty, following the same approach discussed previously in the smoothing context. In particular, we can write the penalized residual sum of squares as:

$$\text{PENSSE}_\lambda(\beta_0, \beta) = \sum_{i=1}^{n} \left[y_i - \beta_0 - \int X(s)\beta(s)ds \right]^2 + \lambda \int [L\beta(s)]^2 ds, \tag{20}$$

where L is a linear differential operator chosen based on the data and λ is a tuning parameter generally selected by cross-validation.

3.2 Variable selection for scalar-on-function regression

In addition to the estimation of the functional regression model, another important aspect to consider is variable selection. The intuition behind is similar to the linear regression case, that is, variable selection is helpful in identifying important predictors, reducing the model size, and improving prediction performance. Many studies over the past decade focus on variable selection for functional regression models with scalar response and functional predictors, while there are fewer proposals for scalar-on-function models.

The proposed procedures can be divided into several categories, such as shrinkage methods, testing procedures, Bayesian framework and application of machine learning algorithms. For shrinkage methods, Pannu and Billor (2017) [11] proposed a LAD-group lasso penalty and considered the situation where data has outliers. Roche (2019) [12] use adaptations of the lasso method to select variables in multivariate functional linear regression while Huang et al. (2016) [13] employ data-driven functional principal components (FPC) basis with least absolute deviation (LAD) loss and a penalty term for robust variable selection. Kong et al. (2016) [14] used FPC for basis expansion and group SCAD and SCAD as penalty terms for functional and non-functional predictors, respectively. Lian (2013) [15] also use FPC basis expansions and perform variable selection using the same technique as Matsui and Konishi (2011) [4], i.e. using L_1 type regularization and specifically the group SCAD penalty. Other types of penalties have also been studied, for example, ℓ_1/ℓ_2-type penalty in Matsui and Umezu (2019) [16], adaptive elastic net penalty in Matsui (2021) [17], and smoothly adaptively centered ridge in Belli (2021) [18]. In the context of variable selection for generalized functional linear models, Matsui (2014) [19] consider to adopt the lasso as the L_1 type penalty for multiclass logistic regression, and Matsui (2019) [20] propose sparse group lasso-type penalty for a similar type of models, allowing the selection of decision boundaries in addition to selection of variables. Zhu and Cox (2009) [21] use FPC basis and group lasso regularization and allow both functional predictors and scalar covariates in the model. Fuchs et al. (2016) [22] propose a classification tool for functional data with multinomial logit model and sparsity-inducing penalty. For the testing procedures, Collazos et al. (2016) [23] consider likelihood ratio test, and Swihart et al. (2014) [24] propose to employ restricted likelihood ratio tests to test for important functional predictors. Bayesian approach can also be applied to variable selection, for example Zhu et al. (2010) [25] study Bayesian hierarchical model for classification. They

use orthonormal basis expansion or FPC to reduce the dimension of functional data and a hybrid Metropolis–Hastings/Gibbs sampler (George and McCulloch, 1997 [26]) for posterior sampling, which selects relevant variables simultaneously. In addition, several studies introduce the concept of variable importance. Möller et al. (2016) [27] apply random forests for functional covariates and propose a variable importance curve, which could be potentially used for variable selection. Gregorutti et al. (2015) [28] study group variable selection with random forests and the applications of the grouped importance measure. Mielniczuk and Teisseyre (2014) [29] consider subsets of variables and measure variable importance using t-statistics. This method is also studied by Smaga and Matsui (2018) [30], who propose two algorithms based on random subspace method for selecting functional variables. Another different approach is distance correlation, proposed by Székely et al. (2007) [31] and used in Febrero-Bande et al. (2019) [32] for selecting variables in functional additive regression models, where predictors can be of different nature (functional, scalar, multivariate, directional, etc.), and the effect of each predictor can be linear or nonlinear. Lastly, Liu et al. (2018) [33] study functional variable selection via Gram–Schmidt (FGS) orthogonalization.

In this project, we compare the variable selection procedure proposed by Gertheiss et al. (2013) [3] with the procedure proposed by Matsui and Konishi (2011) [4] via simulation studies, hence we briefly summarize both methods below.

Gertheiss et al. (2013) [3] propose to use a sparsity-smoothness penalty technique to select important functional variables for both the normal response case and the binary response case. The proposed method is first presented for the normal response case then modified to apply to the binary response case. Assume we have scalar response $y_1, \ldots, y_n$, $X_1, \ldots, X_p$ squared integrable random curves and $X_{i1}, \ldots, X_{ip}$ with $i = 1, \ldots, n$ independent realizations of these random curves. Further, assume that the $X'_{ij}s$ are observed at a dense grid of time points $t_{j1}, \ldots, t_{jN_j}$. Then the functional linear model can be written as:

$$y_i = \beta_0 + \sum_{j=1}^{p} \int X_{ij}(t)\beta_j(t)dt + \epsilon_i, \tag{21}$$

where ϵ_i are independent random errors with mean 0 and variance σ^2. Each coefficient β_j is estimated by basis expansion with a rich pre-set basis function and the approximation is $\beta_j(t) = \sum_{r=1}^{q} b_{jr}\theta_{jr}(t)$, where r indexes the number of basis function used and b_{jr} is the basis coefficient estimate for the basis functions $\theta_{jr}(t)$. Then the functional linear model is approximated by applying Riemann sum:

$$\int X_{ij}(t)\beta_j(t)dt \approx \sum_r \left\{ \Delta_j \sum_l X_{ij}\left(t_{jl}\right)\theta_{jr}\left(t_{jl}\right) \right\} b_{jr} = Z_{ij}^{\top} b_j, \tag{22}$$

where $b_j = \left(b_{j1}, \ldots, b_{jq}\right)^{\top}$, $Z_{ij} = \left(Z_{ij1}, \ldots, Z_{ijq}\right)^{\top}$, $Z_{ijr} = \Delta_j \sum_l X_{ij}\left(t_{jl}\right)\theta_{jr}\left(t_{jl}\right)$ and $\Delta_j = t_{jl} - t_{j,l-1}$. The proposed sparsity-smoothness penalty technique, introduced by Meier et al. (2009) [34] is defined as:

$$P_{\lambda,\varphi}\left(\beta_j\right) = \lambda\left(\left\|\beta_j\right\|^2 + \varphi\left\|\beta_j''\right\|^2\right)^{1/2}, \tag{23}$$

and can be rewritten as $P_{\lambda,\varphi}(\beta_j) = \lambda\left(b_j^\top(\Psi_j + \varphi\Omega_j)\, b_j\right)^{1/2}$ where Ψ_j is the $q \times q$ matrix with the (r,k) element equal to $(\Psi_j)_{rk} = \int \theta_{jr}(t)\theta_{jk}(t)dt$, $r,k = 1,\ldots,q$, and Ω_j is the $q \times q$ matrix with the (r,k) element equal to $(\Omega_j)_{rk} = \int \theta''_{jr}(t)\theta''_{jk}(t)dt$, $r,k = 1,\ldots,q$. This is a general group lasso type penalty and can be further reduced to $P_{\lambda,\varphi}(\beta_j) = \lambda\left(b_j^\top K_{\varphi,j} b_j\right)^{1/2}$ by letting $K_{\varphi,j} = \Psi_j + \varphi\Omega_j$. Then the functional linear model with penalty has the form:

$$\sum_{i=1}^{p}\left(y_i - \beta_0 - Z_{ij}^\top b_j\right)^2 + \lambda\sum_{j=1}^{p}\left(b_j^\top K_{\varphi,j} b_j\right)^{1/2}, \tag{24}$$

where λ and φ are tuning parameters that can be selected by cross-validation. There is also an adaptive penalty proposed, which is defined as:

$$P_{\lambda,\varphi}(\beta_j) = \lambda\left(W_j\|\beta_j\|^2 + \varphi V_j\left\|\beta''_j\right\|^2\right)^{1/2}, \tag{25}$$

where W_j and V_j are weights chosen in a data-adaptive way (Meier et al., 2009) [34]. These penalties for functional linear models can be extended to the generalized response case by replacing the first quadratic form in (23) with the (log-)likelihood function corresponding to y_i. Through simulation studies and real data examples, the proposed procedure is proven to be effective.

Matsui and Konishi (2011) [4] also study the functional linear model (20) but use Gaussian basis functions (Ando et al., 2008 [35]) for basis expansion: $x_{ij}(t) = \sum_{r=1}^{q} w_{ijr}\phi_{jr}(t) = \boldsymbol{w}_{ij}^T\boldsymbol{\phi}_j(t)$, where r indexes the number of basis function used. The Gaussian basis functions are defined as $\phi_{jr}(t) = \exp\left\{-\frac{(t-m_{jr})^2}{2\nu_j s_{jr}^2}\right\}$ $(r = 1,\ldots,q)$, where m_r is the center, s_r^2 is the dispersion of the corresponding basis function, and ν_r is a hyperparameter. The number of coefficients and parameters of Gaussian basis functions are pre-determined. If the coefficient functions $\beta_j(t)$ for the functional predictor $X_j(t)$ is written in terms of the same basis expansions, then (20) can be re-written in this form:

$$\begin{aligned} y_i &= \beta_0 + \sum_{j=1}^{p}\int_{\mathcal{T}} \boldsymbol{c}_{ij}^T\phi_j(t)\boldsymbol{\phi}_j^T(t)\boldsymbol{b}_j dt + \epsilon_i \\ &= \boldsymbol{z}_i^T\boldsymbol{b} + \epsilon_i, \end{aligned} \tag{26}$$

where $\boldsymbol{z}_i = \left(1, \boldsymbol{c}_{i1}^T\boldsymbol{J}_{\phi_1},\ldots,\boldsymbol{c}_{ip}^T\boldsymbol{J}_{\phi_j}\right)^T$, $\boldsymbol{b} = \left(\beta_0, \boldsymbol{b}_1^T,\ldots,\boldsymbol{b}_j^T\right)^T$ and $\boldsymbol{J}_{\phi_j} = \int_{\mathcal{T}}\phi_j(t).\ \phi_j^T(t)dt$ are $r \times r$ cross product matrices. Assuming Gaussian basis functions are used, the (l,k)-th element of J_{ϕ_j} has the form: $J_{\phi_j}^{(l,k)} = \frac{\sqrt{2\pi\nu_j s_{jl}^2 s_{jk}^2}}{\sqrt{s_{jl}^2 + s_{jk}^2}}\exp\left\{-\frac{\left(m_{jl}-m_{jk}\right)^2}{2\nu_j\left(s_{jl}^2+s_{jk}^2\right)}\right\}$. Moreover, the functional regression model (21) has probability density function: $f\left(y_i \mid \boldsymbol{x}_i; \boldsymbol{b}, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{\left(y_i - \boldsymbol{z}_i^T\boldsymbol{b}\right)^2}{2\sigma^2}\right\}$. To estimate the model parameters $\boldsymbol{\theta} = \left\{\boldsymbol{b}, \sigma^2\right\}$, they considered maximizing a penalized log-likelihood function with group SCAD penalty:

$$l_\lambda(\boldsymbol{\theta}) = l(\boldsymbol{\theta}) - n\sum_{j=1}^{p} P_\lambda\left(\|\boldsymbol{b}_j\|_2\right), \tag{27}$$

where $l(\theta)$ is the log-likelihood function $l(\boldsymbol{\theta}) = \sum_{i=1}^{n} f(y_i \mid \boldsymbol{x}_i; \boldsymbol{\theta})$. $P_\lambda(\cdot)$ is the group SCAD penalty, which has the properties of sparsity, continuity and unbiasedness and is defined in equation (11). Finally, $\|\boldsymbol{b}_j\|_2 = \sqrt{\boldsymbol{b}_j^T \boldsymbol{G}_j \boldsymbol{b}_j}$, where G_j is a $r \times r$ positive semi-definite matrix. Due to the difficulty in deriving the SCAD estimator analytically, Matsui and Konishi (2011) [4] use the iterative procedure developed by Fan and Li (2001) [7] to approximate the SCAD penalty as:

$$P_\lambda\left(\|\boldsymbol{b}_j\|_2\right) \approx P_\lambda\left(\left\|\boldsymbol{b}_j^{(0)}\right\|_2\right) + \frac{1}{2}\frac{P'_\lambda\left(\left\|\boldsymbol{b}_j^{(0)}\right\|_2\right)}{\left\|\boldsymbol{b}_j^{(0)}\right\|_2}\left(\boldsymbol{b}_j^T\boldsymbol{b}_j - \boldsymbol{b}_j^{(0)^T}\boldsymbol{b}_j^{(0)}\right) \tag{28}$$

for $b_j \approx b_j^{(0)}$, where $b^{(0)}$ and $\sigma^{(0)2}$ are initial values and $\boldsymbol{b}^{(0)} = \left(\beta_0^{(0)}, \boldsymbol{b}_1^{(0)^T}, \ldots, \boldsymbol{b}_j^{(0)^T}\right)^T$ can be ridge estimator or maximum likelihood estimator with generalized inverse. Consequently, the $k+1$-th update for b and σ^2 under the Gaussian model are given as:

$$\boldsymbol{b}^{(k+1)} = \left(Z^T Z + n\sigma^{(k)2}\Sigma\left(\boldsymbol{b}^{(k)}\right)\right)^{-1} Z^T \boldsymbol{y} \tag{29}$$

$$\sigma^{(k+1)2} = \frac{1}{n}\left(\boldsymbol{y} - Z\boldsymbol{b}^{(k+1)}\right)^T\left(\boldsymbol{y} - Z\boldsymbol{b}^{(k+1)}\right), \tag{30}$$

where $Z = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)^T$ and $\boldsymbol{y} = (y_1, \ldots, y_n)^T$. They are updated until convergence so that the regularized estimator $\hat{\boldsymbol{b}}$, $\hat{\sigma^2}$ are obtained. For selecting the regularization parameter λ, the authors try several selection criteria, including the GIC (Konishi and Kitagawa, 1996 [36]) defined as $\mathrm{GIC} = -2\sum_{i=1}^{n} f(y_i \mid \boldsymbol{x}_i; \boldsymbol{\theta}) + 2\,\mathrm{tr}\left\{R(\hat{\boldsymbol{\theta}})^{-1}Q(\hat{\boldsymbol{\theta}})\right\}$, where $R(\boldsymbol{\theta})$, $Q(\boldsymbol{\theta})$ are defined as following: $R(\boldsymbol{\theta}) = -\frac{1}{n}\sum_{i=1}^{n}\frac{\partial^2\left\{\log f(y_i|\boldsymbol{x}_i;\boldsymbol{\theta}) - \boldsymbol{b}^T\Sigma(\boldsymbol{b})\boldsymbol{b}/2\right\}}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}$ and $Q(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\frac{\partial\left\{\log f(y_i|\boldsymbol{x}_i;\boldsymbol{\theta}) - \boldsymbol{b}^T\Sigma(\boldsymbol{b})\boldsymbol{b}/2\right\}}{\partial\boldsymbol{\theta}}\frac{\partial\{\log f(y_i|\boldsymbol{x}_i;\boldsymbol{\theta})\}}{\partial\boldsymbol{\theta}^T}$; BIC given by $\mathrm{BIC} = -2\sum_{\alpha=1}^{n} f(y_\alpha \mid \boldsymbol{x}_\alpha; \boldsymbol{\theta}) + \hat{d}f \log n$; and GCV defined as $\mathrm{GCV} = \frac{1}{n}\frac{\|\boldsymbol{y} - Z\hat{\boldsymbol{b}}\|^2}{(1-\hat{d}f/n)^2}$ with effective degrees of freedom: $\hat{df} = \mathrm{tr}\left\{Z\left(Z^TZ + n\hat{\sigma}^2\Sigma(\hat{\boldsymbol{b}})\right)^{-1}Z^T\right\}$. By minimizing these criteria, the corresponding model is selected as the optimal one. Through simulation studies and real data examples, the proposed procedure is proven to be effective.

4 Simulation Studies

We conducted simulation studies to compare the performance of the two methods detailed in the previous section. The design is based largely on the simulations in Gertheiss et al. (2013) [3], that we extend to more complex models with many functional predictors. The dataset is of the form $\{X_{i1}(t) : t \in \mathcal{T}_1\}, \ldots, \{X_{i30}(t) : t \in \mathcal{T}_{30}\}, Y_i,\ i = 1, ..., 300.$ $\mathcal{T}_j$ is a set of 300 equidistant points in $(0, 300)$, and the functional predictors X_{ij} are constructed as $x_{ij}(t) = \sum_{r=1}^{5}\left(b_{ijr}\sin\left(t\pi\left(5 - b_{ijr}\right)/150\right) - m_{ijr}\right) +$ 15. Instead of considering a total of 10 functional predictors (of which 5 active) as in Gertheiss simulation stidy, we consider 30 functional predictors of which 10 are relevant to predict the response variable. The true coefficient functions $\beta_j(t)'s$ have gamma or exponential like shape, but used different scales compared to the original design, so that $\beta_1(t), \ldots, \beta_6(t)$ have gamma like shape and

decreasing effective size while $\beta_9(t)$ and $\beta_{10}(t)$ have more linear exponential shape compared to $\beta_7(t)$ and $\beta_8(t)$. For generating the response variable, we generated $Y_i = \alpha + \sum_{j=1}^{10} \int_0^{300} \beta_j(t) X_{ij}(t) dt + \epsilon_i$ for the functional linear model, and the error follows normal distribution with mean 0 and variance 10. We also generated a binary response, $Y_i \sim \text{Bernoulli}\left[\exp\left(\eta_i\right) / \left\{1 + \exp\left(\eta_i\right)\right\}\right]$ with $\eta_i = \alpha + \sum_{j=1}^{10} \int_0^{300} \beta_j(t) X_{ij}(t) dt$ for the generalized functional linear model. Note that both variable selection methods are applied to the functional linear model, but only the method proposed by Gertheiss et al. (2013) [3] is applied to the generalized functional linear model. For simplicity, we will refer to the method proposed by Gertheiss et al. (2013) [3] as method 1, and to the method proposed by Matsui and Konishi (2011) [4] as method 2 for the rest of this section.

The implementation of method 1 does not require any extensive modification to accommodate the larger number of functional predictors. As in the original design, the simple, standard, adapt 1, and adapt 2 approaches are considered when comparing the number of predictors selected. The simple approach uses 30 basis functions (B-Spline basis) to control the smoothness and then apply the proposed method with penalty (23) and $\varphi = 0$. The standard approach is the standard functional group lasso. The adapt 1 and adapt 2 are based on penalized functional regression (PFR; Goldsmith et al., 2011a) [37] and the smoothing parameter estimation method used is REML. The adapt 1 approach is the adaptive penalized estimation (25) with weights W_j and V_j chosen adaptively while adapt 2 fixed $V_j = 1$ based on adapt 1. To get the initial estimates of W_j and V_j in adapt 1 and adapt 2, we fit a PFR and use the estimated coefficient functions to calculate W_j and V_j as $w_j = 1/\left\|\breve{\beta}_j\right\|$ and $v_j = 1/\left\|\breve{\beta}_j''\right\|$. The `pfr()`function in R `refund` package (Goldsmith et al., 2021) [38] has been updated since the publication of the original article. Therefore, we employ the `pfr_old()`function instead of the `pfr()`function throughout our simulations.

Table 1 shows the proportion of times each functional predictor is selected in a total of 50 simulated runs for the four approaches for method 1 in the normal response case. Each column represents one functional predictor, and each row corresponds to one approach. The numbers range from 0 to 1 and are calculated as the number of times the functional predictor is selected divided by 50 runs. For example, 1 means that the predictor is selected in all 50 runs. From the table, we can see that all four approaches tend to select larger model. Almost all relevant functional predictors (columns 1 – 10) are selected, while the rest of the functional predictors seem to be selected randomly in some runs. Adapt 1 and adapt 2 perform better and select smaller models, with about 18 functional predictors selected, compared to standard and simple approaches. Standard approach select the largest model among the four approaches, with an average of about 26 functional predictors included. For the four approaches, the true positive rates are 1, 0.998, 0.998, and 1 respectively. The standard error for each functional predictor is also computed and shown in Table 4 in the Appendix.

Table 2 shows the results for the binary response case. This table is constructed in the same way as the normal response case. We can see that the selection of true functional predictors follows a similar pattern, which is as expected based on the original design. All approaches favor large models, and irrelevant functional predictors are randomly included in some runs. Compared to the

Approach \ Predictor	1	2	3	4	5	6	7	8	9	10	
standard	1	1	1	1	1	1	1	1	1	1	
adapt1	1	1	1	1	1	0.98	1	1	1	1	
adapt2	1	1	1	1	1	0.98	1	1	1	1	
simple	1	1	1	1	1	1	1	1	1	1	
Approach \ Predictor	11	12	13	14	15	16	17	18	19	20	
standard	0.72	0.76	0.8	0.76	0.72	0.82	0.7	0.84	0.9	0.8	
adapt1	0.34	0.36	0.38	0.4	0.36	0.3	0.34	0.48	0.36	0.48	
adapt2	0.34	0.36	0.42	0.38	0.4	0.28	0.36	0.38	0.38	0.48	
simple	0.7	0.66	0.74	0.8	0.62	0.72	0.68	0.82	0.74	0.74	
Approach \ Predictor	21	22	23	24	25	26	27	28	29	30	**AvgSize**
standard	0.76	0.76	0.82	0.78	0.76	0.88	0.78	0.74	0.78	0.78	**25.66**
adapt1	0.36	0.34	0.4	0.36	0.34	0.5	0.42	0.48	0.34	0.36	**17.68**
adapt2	0.3	0.34	0.4	0.34	0.44	0.44	0.44	0.48	0.28	0.38	**17.6**
simple	0.56	0.56	0.76	0.64	0.76	0.86	0.76	0.72	0.7	0.72	**24.26**

Table 1: Selection results for method 1 in the normal response case.

normal response case, the average model sizes are slightly smaller in the binary case except for the adapt 2 approach. Adapt 1 approach has the best performance. One significant finding is that all four approaches occasionally exclude some relevant functional predictors. For example, as shown in column 6 of the table, adapt 1 approach exclude this predictor in almost half of the 50 runs, while the standard approach selected this predictor more frequently, but still excluded it in about 10 runs. For the four approaches, the true positive rates are 0.98, 0.942, 0.954, and 0.976 respectively, slightly lower than in the normal response case. Therefore, method 1 seems to have more limited ability to select the correct functional predictors in the binary response case.

To further analyze the results in both scenarios, we present plots for the true functional coefficients versus the estimated coefficients for the 10 active functional predictors in Figure 2 and 3. The dashed lines correspond to the true functional coefficients, and the solid lines correspond to the estimated functional coefficients. The estimated functional coefficients are averaged over the 50 runs, and we select the approach that has best performance in each scenario, i.e. adapt 2 approach for the normal response case and adapt 1 approach for the binary response case. From these plots, we can see that the estimated coefficients capture most of the underlying true curve structure in both cases overall. The estimation in the normal response case is better than in the binary response case.

The implementation of method 2 has been very challenging. After obtaining the R code from the

Figure 2: True (dashed line) and estimated (solid line) coefficients for method 1 in the normal response case.

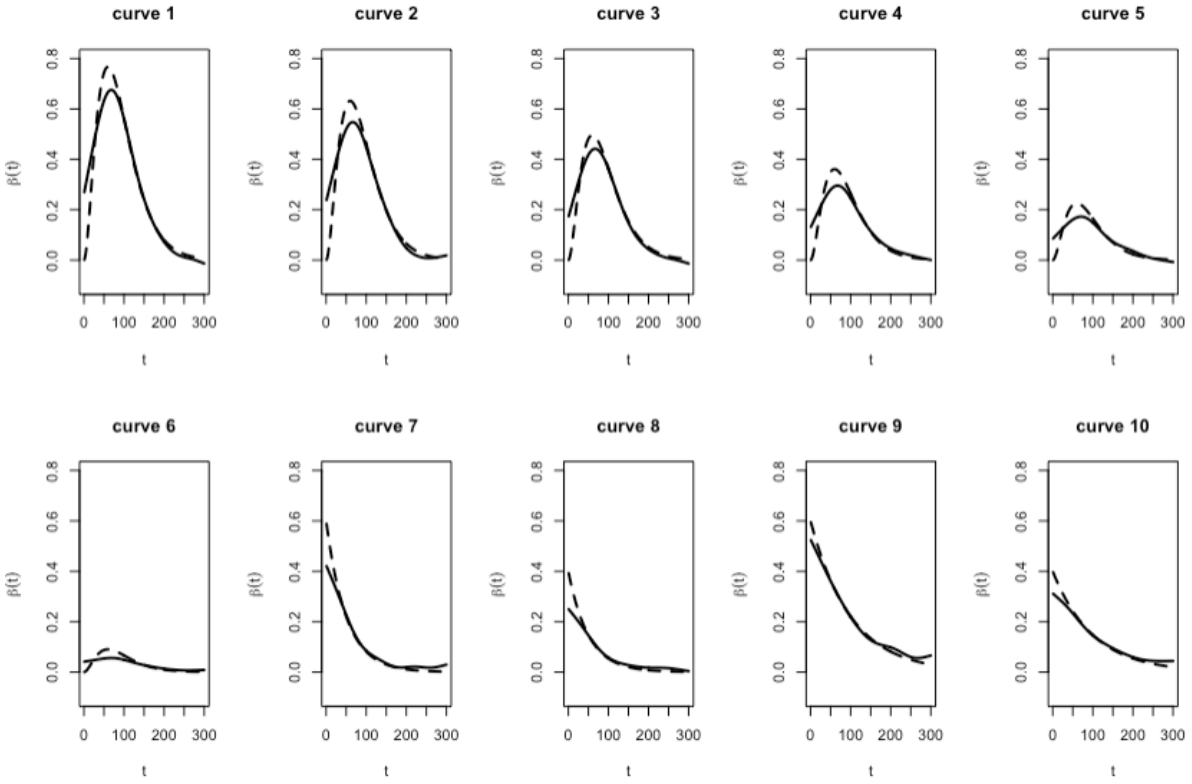
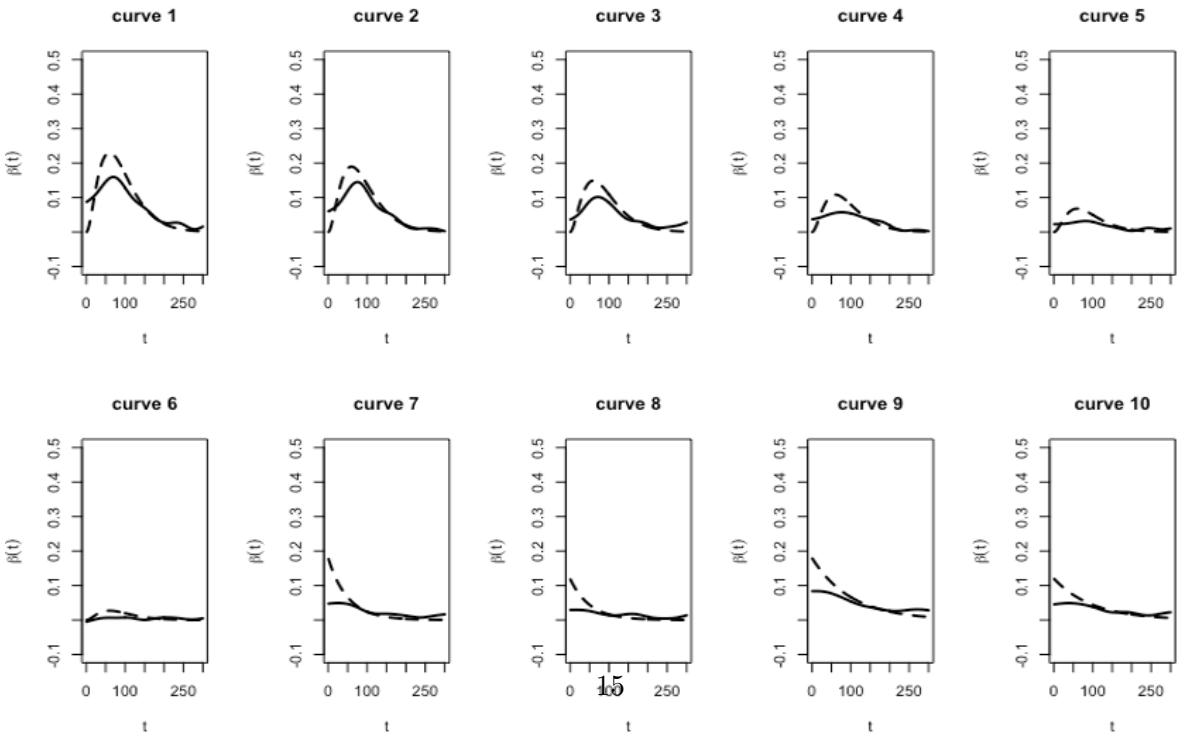


Figure 3: True (dashed line) and estimated (solid line) coefficients for method 1 in the binary response case.



Approach \ Predictor	1	2	3	4	5	6	7	8	9	10	
standard	1	1	1	1	1	0.82	1	0.98	1	1	
adapt1	1	1	1	1	0.96	0.54	1	0.92	1	1	
adapt2	1	1	1	1	0.96	0.64	1	0.94	1	1	
simple	1	1	1	1	1	0.78	1	0.98	1	1	
Approach \ Predictor	11	12	13	14	15	16	17	18	19	20	
standard	0.66	0.58	0.48	0.66	0.72	0.56	0.72	0.62	0.64	0.7	
adapt1	0.38	0.28	0.28	0.34	0.44	0.3	0.54	0.3	0.38	0.42	
adapt2	0.5	0.36	0.4	0.48	0.54	0.34	0.6	0.36	0.4	0.42	
simple	0.62	0.58	0.58	0.62	0.76	0.58	0.72	0.58	0.6	0.7	
Approach \ Predictor	21	22	23	24	25	26	27	28	29	30	**AvgSize**
standard	0.7	0.72	0.66	0.66	0.66	0.64	0.66	0.62	0.7	0.68	**22.84**
adapt1	0.5	0.38	0.46	0.34	0.38	0.4	0.48	0.38	0.44	0.46	**17.3**
adapt2	0.46	0.44	0.48	0.38	0.5	0.52	0.54	0.42	0.54	0.54	**18.76**
simple	0.66	0.74	0.64	0.58	0.7	0.68	0.7	0.6	0.72	0.64	**22.76**

Table 2: Selection results for method 1 in the binary response case

authors, we first tried to reproduce the results of the original simulation example (shown in Figure 4), but this was unsuccessful due to several reasons. First, we noticed a parameter `c` listed, however, we could not find this parameter being used anywhere in the R script that implements the method, and this parameter is not explained in the article neither. Hence, we ignored this parameter throughout our simulation study. Second, in the original article, the authors indicate that $a = 3.7$ is used as a parameter of SCAD penalty; however, in the R script both λ and a are selected by selection criteria, GCV, GIC, and BIC. In addition, we noticed that in some case, the grid of lambda does not cover the optimal value, and thus produced incorrect estimations and large MSE. Hence, we expanded the grid to fix this issue going forward. Moreover, out of the 100 simulation runs, Figure 4 shows that the proposed method select correct models frequently, as high as 90 times. Note that the original design simulate three functional predictors with two of them being relevant. This selection result is very different from the results we obtained. In our replication, out of 20 simulation runs, the method selected all three functional predictors in every run. We further investigated the selection results in detail, and determined that this result is due to a bad choice of lambda. Indeed, it appears that the algorithm can actually select the correct variables with a certain lambda value, however this optimal value is usually not the one selected by BIC, GIC, or GCV. For example, in one test run, the lambda that selects exactly two functional predictors is around 0.794, but the lambda selected by BIC, GIC,

	$n = 50$				$n = 100$			
	GCV	BIC	GIC	glasso	GCV	BIC	GIC	glasso
(c=0.05)								
AMSE	1.98	1.98	1.98	1.70	2.76	2.76	2.77	2.79
Correct	80	80	80	78	90	91	90	85
(c=0.1)								
AMSE	7.58	7.58	7.42	5.96	7.87	7.88	7.88	8.06
Correct	51	51	53	48	80	80	80	72

Figure 4: Results from the simulation example in the original paper. [4]

and GCV is around 0.126. So far, we have identified a missing component, the estimation of sigma hat, in the GCV selection criterion function for group SCAD, and we suspect it to be related with the problem. We might be attempting to modify this function later to include the estimation of sigma hat, but it is currently not within the scope of this project. Another limitation of this method is that the number of basis functions in basis expansion is fixed to 6, which might not work well in our design. It would be worth changing this setting in future studies.

With a significant effort to adjust the R script of method 2, we were able to test this method on the same dataset as method 1. To make the data smoothing more efficiently, we replaced the repeated code with a loop structure, so that larger number of functional predictors can be smoothed easily. We attempted parallel computing for data smoothing as well to reduce the amount of time it takes, but this modification had issues when the script was scheduled to run on a cluster, so it was not used for the final simulation. Table 3 shows the simulation results for method 2; it should be interpreted the same way as the results tables for method 1.

The approaches considered in method 2 are group SCAD penalty with tuning parameters selected by GCV, BIC, and GIC respectively, and group lasso penalty with tuning parameters selected by the same sets of selection criteria. The true positive rates are all 1 despite different approaches or selection criteria. It is obvious that many irrelevant functional predictors are selected by both algorithms consistently with the problems we discussed previously in the replication of original design. Therefore, we believe that this simulation results table do not accurately reflect the performance of method 2, and we expect to see similar efficiency as in original article if the code issues are addressed.

5 Conclusion

We studied the variable selection methods for scalar-on-function regression models and conducted simulation studies to evaluate and compare the performance of two selected methods on complex functional linear models. The main idea behind these variable selection methods is to employ a grouped version of lasso or SCAD to select relevant functional predictors. We started by reviewing the concept of shrinkage methods, and then we introduced functional data analysis and functional

Approach \ Predictor	1	2	3	4	5	6	7	8	9	10	
GCV_gscad	1	1	1	1	1	1	1	1	1	1	
BIC_gscad	1	1	1	1	1	1	1	1	1	1	
GIC_gscad	1	1	1	1	1	1	1	1	1	1	
GCV_glasso	1	1	1	1	1	1	1	1	1	1	
BIC_glasso	1	1	1	1	1	1	1	1	1	1	
GIC_glasso	1	1	1	1	1	1	1	1	1	1	
Approach \ Predictor	11	12	13	14	15	16	17	18	19	20	
GCV_gscad	0.96	0.92	0.94	0.98	0.94	0.9	0.96	0.94	0.94	0.98	
BIC_gscad	0.82	0.88	0.86	0.86	0.88	0.78	0.92	0.78	0.84	0.9	
GIC_gscad	0.96	0.94	0.96	0.98	0.96	0.92	0.98	0.94	0.94	0.98	
GCV_glasso	1	1	1	1	0.98	1	0.96	1	1	1	
BIC_glasso	1	1	1	1	0.98	1	0.96	1	1	1	
GIC_glasso	1	1	1	1	0.98	1	0.96	1	1	1	
Approach \ Predictor	21	22	23	24	25	26	27	28	29	30	**AvgSize**
GCV_gscad	0.96	0.92	0.98	0.94	0.98	0.94	0.96	0.96	0.96	0.9	**28.96**
BIC_gscad	0.92	0.8	0.96	0.86	0.92	0.84	0.88	0.94	0.92	0.8	**27.36**
GIC_gscad	0.98	0.96	0.98	0.94	0.98	0.96	0.96	0.96	0.98	0.92	**29.18**
GCV_glasso	1	1	1	1	1	1	1	1	1	0.96	**29.9**
BIC_glasso	1	1	1	1	1	1	1	1	1	0.96	**29.9**
GIC_glasso	1	1	1	1	1	1	1	1	1	0.96	**29.9**

Table 3: Selection results for method 2 (normal response case)

linear models. A literature review of recently developed variable selection methods was also conducted and briefly summarized. For the two variable selection methods we chose, we outlined the proposed methodologies, which used basis expansion to approximate functional predictors and different group type penalties. Lastly, the simulation studies and results were described and interpreted. The adaptive version of sparsity-smoothness penalty proposed by Gertheiss et al. (2013) [3] has the best variable selection performance and good estimation of the true coefficient functions. Even though this method favors large model, the relevant functional predictors are almost always selected. The method proposed by Matsui and Konishi (2011) [4] has rather unexpected performance and several potential issues. Though the method produced entirely different results as in original article, we managed to carry out our simulation study. If the limitations and concerns on the code implementing this method can be addressed, we expect to see good performance with this method too.

Acknowledgements

I would like to express my sincere gratitude and appreciation to my advisor Dr. Marzia A. Cremona, who has shown tremendous support for me, and provided guidance and advice through all stages of this project. I would like to thank Dr. Adam Rothman and Dr. Qian Qin for agreeing to be my co-advisor and committee member, and for providing brilliant comments and suggestions for my project. I thank Dr. Hidetoshi Matsui and Dr. Jan Gertheiss for sharing supplementary materials from their published research work so I could continue this project. I also would like to give special thanks to my program director Dr. Yuhong Yang and our former program coordinator Taryn Verley for working on the project and graduation requirements with me.

My biggest thanks to my parents Xueling Zeng and Jingming Yang, for supporting me on every decision I make, none of this would indeed be possible without you. Finally, I thank my cat Jackpot, for the emotional support and company.

References

[1] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

[2] Lukas Meier, Sara Van De Geer, and Peter Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008.

[3] Jan Gertheiss, Arnab Maity, and Ana-Maria Staicu. Variable selection in generalized functional linear models. *Stat*, 2(1):86–101, 2013.

[4] Hidetoshi Matsui and Sadanori Konishi. Variable selection for functional regression models via the l1 regularization. *Computational Statistics & Data Analysis*, 55(12):3304–3310, 2011.

[5] Hastie T, Tibshirani R, and Friedman J. *The elements of Statistical Learning, second edition: Data Mining, Inference, and prediction.* Springer, 2009.

[6] James G, Witten D, Hastie T, and Tibshirani R. *An introduction to statistical learning: With applications in R.* Springer, 2017.

[7] Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

[8] Lifeng Wang, Guang Chen, and Hongzhe Li. Group scad regression analysis for microarray time course gene expression data. *Bioinformatics*, 23(12):1486–1494, 2007.

[9] Kokoszka P and Reimherr M. *Introduction to functional data analysis.* CRC Press, Taylor Francis Group, 2017.

[10] Ramsay J.O and Silverman B.W. *Functional Data Analysis.* Springer, 2006.

[11] Jasdeep Pannu and Nedret Billor. Robust group-lasso for functional regression model. *Communications in statistics-simulation and computation*, 46(5):3356–3374, 2017.

[12] Angelina Roche. Variable selection and estimation in multivariate functional linear regression via the lasso. *arXiv preprint arXiv:1903.12414*, 2019.

[13] Lele Huang, Junlong Zhao, Huiwen Wang, and Siyang Wang. Robust shrinkage estimation and selection for functional multiple linear model through lad loss. *Computational Statistics & Data Analysis*, 103:384–400, 2016.

[14] Dehan Kong, Kaijie Xue, Fang Yao, and Hao H Zhang. Partially functional linear regression in high dimensions. *Biometrika*, 103(1):147–159, 2016.

[15] Heng Lian. Shrinkage estimation and selection for multiple functional regression. *Statistica Sinica*, pages 51–74, 2013.

[16] Hidetoshi Matsui and Yuta Umezu. Variable selection in multivariate linear models for functional data via sparse regularization. *Japanese Journal of Statistics and Data Science*, pages 1–15, 2019.

[17] Hidetoshi Matsui. Sparse varying-coefficient functional linear model. *arXiv preprint arXiv:2110.12599*, 2021.

[18] Edoardo Belli. Smoothly adaptively centered ridge estimator. *Journal of Multivariate Analysis*, page 104882, 2021.

[19] Hidetoshi Matsui. Variable and boundary selection for functional data via multiclass logistic regression modeling. *Computational Statistics & Data Analysis*, 78:176–185, 2014.

[20] Hidetoshi Matsui. Sparse group lasso for multiclass functional logistic regression models. *Communications in Statistics-Simulation and Computation*, 48(6):1784–1797, 2019.

[21] Hongxiao Zhu and Dennis D Cox. A functional generalized linear model with curve selection in cervical pre-cancer diagnosis using fluorescence spectroscopy. In *Optimality*, pages 173–189. Institute of Mathematical Statistics, 2009.

[22] Karen Fuchs, Wolfgang Pößnecker, and Gerhard Tutz. Classification of functional data with k-nearest-neighbor ensembles by fitting constrained multinomial logit models. *arXiv preprint arXiv:1612.04710*, 2016.

[23] Julian AA Collazos, Ronaldo Dias, and Adriano Z Zambom. Consistent variable selection for functional regression models. *Journal of Multivariate Analysis*, 146:63–71, 2016.

[24] Bruce J Swihart, Jeff Goldsmith, and Ciprian M Crainiceanu. Restricted likelihood ratio tests for functional effects in the functional linear model. *Technometrics*, 56(4):483–493, 2014.

[25] Hongxiao Zhu, Marina Vannucci, and Dennis D Cox. A bayesian hierarchical model for classification with selection of functional predictors. *Biometrics*, 66(2):463–473, 2010.

[26] Edward I George and Robert E McCulloch. Approaches for bayesian variable selection. *Statistica sinica*, pages 339–373, 1997.

[27] Annette Möller, Gerhard Tutz, and Jan Gertheiss. Random forests for functional covariates. *Journal of Chemometrics*, 30(12):715–725, 2016.

[28] Baptiste Gregorutti, Bertrand Michel, and Philippe Saint-Pierre. Grouped variable importance with random forests and application to multiple functional data analysis. *Computational Statistics & Data Analysis*, 90:15–35, 2015.

[29] Jan Mielniczuk and Paweł Teisseyre. Using random subspace method for prediction and variable importance assessment in linear regression. *Computational Statistics & Data Analysis*, 71:725–742, 2014.

[30] Łukasz Smaga and Hidetoshi Matsui. A note on variable selection in functional regression via random subspace method. *Statistical Methods & Applications*, 27(3):455–477, 2018.

[31] Gábor J Székely, Maria L Rizzo, and Nail K Bakirov. Measuring and testing dependence by correlation of distances. *The annals of statistics*, 35(6):2769–2794, 2007.

[32] Manuel Febrero-Bande, Wenceslao González-Manteiga, and Manuel Oviedo de la Fuente. Variable selection in functional additive regression models. *Computational Statistics*, 34(2):469–487, 2019.

[33] Ruiping Liu, Huiwen Wang, and Shanshan Wang. Functional variable selection via gram–schmidt orthogonalization for multiple functional linear regression. *Journal of Statistical Computation and Simulation*, 88(18):3664–3680, 2018.

[34] Lukas Meier, Sara Van de Geer, and Peter Bühlmann. High-dimensional additive modeling. *The Annals of Statistics*, 37(6B):3779–3821, 2009.

[35] Tomohiro Ando, Sadanori Konishi, and Seiya Imoto. Nonlinear regression modeling via regularized radial basis function networks. *Journal of Statistical Planning and Inference*, 138(11):3616–3633, 2008.

[36] Sadanori Konishi and Genshiro Kitagawa. Generalised information criteria in model selection. *Biometrika*, 83(4):875–890, 1996.

[37] Jeff Goldsmith, Jennifer Bobb, Ciprian M Crainiceanu, Brian Caffo, and Daniel Reich. Penalized functional regression. *Journal of computational and graphical statistics*, 20(4):830–851, 2011.

[38] Jeff Goldsmith, Fabian Scheipl, Lei Huang, Julia Wrobel, Chongzhi Di, Jonathan Gellar, Jaroslaw Harezlak, Mathew W. McLean, Bruce Swihart, Luo Xiao, Ciprian Crainiceanu, and Philip T. Reiss. *refund: Regression with Functional Data*, 2021. R package version 0.1-24.

Appendix

Approach \ Predictor	1	2	3	4	5	6	7	8	9	10
standard	0	0	0	0	0	0	0	0	0	0
adapt1	0	0	0	0	0	0.14	0	0	0	0
adapt2	0	0	0	0	0	0.14	0	0	0	0
simple	0	0	0	0	0	0	0	0	0	0
Approach \ Predictor	11	12	13	14	15	16	17	18	19	20
standard	0.45	0.43	0.4	0.43	0.45	0.39	0.46	0.37	0.3	0.4
adapt1	0.48	0.48	0.49	0.49	0.48	0.46	0.48	0.5	0.48	0.5
adapt2	0.48	0.48	0.5	0.49	0.49	0.45	0.48	0.49	0.49	0.5
simple	0.46	0.48	0.44	0.4	0.49	0.45	0.47	0.39	0.44	0.44
Approach \ Predictor	21	22	23	24	25	26	27	28	29	30
standard	0.43	0.43	0.39	0.42	0.43	0.33	0.42	0.44	0.42	0.42
adapt1	0.48	0.48	0.49	0.48	0.48	0.51	0.5	0.5	0.48	0.48
adapt2	0.46	0.48	0.49	0.48	0.5	0.5	0.5	0.5	0.45	0.49
simple	0.5	0.5	0.43	0.48	0.43	0.35	0.43	0.45	0.46	0.45

Table 4: Standard Errors for method 1 normal response

Approach \ Predictor	1	2	3	4	5	6	7	8	9	10
standard	0	0	0	0	0	0	0	0	0	0
adapt1	0	0	0	0	0	0.14	0	0	0	0
adapt2	0	0	0	0	0	0.14	0	0	0	0
simple	0	0	0	0	0	0	0	0	0	0
Approach \ Predictor	11	12	13	14	15	16	17	18	19	20
standard	0.45	0.43	0.4	0.43	0.45	0.39	0.46	0.37	0.3	0.4
adapt1	0.48	0.48	0.49	0.49	0.48	0.46	0.48	0.5	0.48	0.5
adapt2	0.48	0.48	0.5	0.49	0.49	0.45	0.48	0.49	0.49	0.5
simple	0.46	0.48	0.44	0.4	0.49	0.45	0.47	0.39	0.44	0.44
Approach \ Predictor	21	22	23	24	25	26	27	28	29	30
standard	0.43	0.43	0.39	0.42	0.43	0.33	0.42	0.44	0.42	0.42
adapt1	0.48	0.48	0.49	0.48	0.48	0.51	0.5	0.5	0.48	0.48
adapt2	0.46	0.48	0.49	0.48	0.5	0.5	0.5	0.5	0.45	0.49
simple	0.5	0.5	0.43	0.48	0.43	0.35	0.43	0.45	0.46	0.45

Table 5: Standard Errors for method 1 binary response

Approach \ Predictor	1	2	3	4	5	6	7	8	9	10
GCV_gscad	0	0	0	0	0	0	0	0	0	0
BIC_gscad	0	0	0	0	0	0	0	0	0	0
GIC_gscad	0	0	0	0	0	0	0	0	0	0
GCV_glasso	0	0	0	0	0	0	0	0	0	0
BIC_glasso	0	0	0	0	0	0	0	0	0	0
GIC_glasso	0	0	0	0	0	0	0	0	0	0
Approach \ Predictor	11	12	13	14	15	16	17	18	19	20
GCV_gscad	0.2	0.27	0.24	0.14	0.24	0.3	0.2	0.24	0.24	0.14
BIC_gscad	0.39	0.33	0.35	0.35	0.33	0.42	0.27	0.42	0.37	0.3
GIC_gscad	0.2	0.24	0.2	0.14	0.2	0.27	0.14	0.24	0.24	0.14
GCV_glasso	0	0	0	0	0.14	0	0.2	0	0	0
BIC_glasso	0	0	0	0	0.14	0	0.2	0	0	0
GIC_glasso	0	0	0	0	0.14	0	0.2	0	0	0
Approach \ Predictor	21	22	23	24	25	26	27	28	29	30
GCV_gscad	0.2	0.27	0.14	0.24	0.14	0.24	0.2	0.2	0.2	0.3
BIC_gscad	0.27	0.4	0.2	0.35	0.27	0.37	0.33	0.24	0.27	0.4
GIC_gscad	0.14	0.2	0.14	0.24	0.14	0.2	0.2	0.2	0.14	0.27
GCV_glasso	0	0	0	0	0	0	0	0	0	0.2
BIC_glasso	0	0	0	0	0	0	0	0	0	0.2
GIC_glasso	0	0	0	0	0	0	0	0	0	0.2

Table 6: Standard Errors for method 2