Biometrics 00, 1–20 DOI: 10.1111/j.1541-0420.2005.00454.x
0000 0000

Flexible Variable Selection for Recovering Sparsity in Nonadditive Nonparametric Models

Zaili Fang[1], Inyoung Kim[1,*] and Patrick Schaumont[2]

[1]Department of Statistics, Virginia Tech., Blacksburg, Virginia, U.S.A.

[2]Department of Electrical & Computer Engineering, Virginia Tech., Blacksburg, Virginia, U.S.A.

[*]To whom correspondence should be addressed

[*]*email:* inyoungk@vt.edu

Summary: Variable selection for recovering sparsity in nonadditive and nonparametric models with high dimensional variables has been challenging. This problem becomes even more difficult due to complications in modeling unknown interaction terms among high dimensional variables. There is currently no variable selection method to overcome these limitations. Hence, in this paper we propose a variable selection approach that is developed by connecting a kernel machine with the nonparametric regression model. The advantages of our approach are that it can: (1) recover the sparsity, (2) automatically model unknown and complicated interactions, (3) connect with several existing approaches including linear nonnegative garrote and multiple kernel learning, and (4) provide flexibility for both additive and nonadditive nonparametric models. Our approach can be viewed as a nonlinear version of a nonnegative garrote method. We model the smoothing function by a Least Squares Kernel Machine (LSKM) and construct the nonnegative garrote objective function as the function of the sparse scale parameters of kernel machine to recover sparsity of input variables whose relevances to the response are measured by the scale parameters. We also provide the asymptotic properties of our approach. We show that sparsistency is satisfied with consistent initial kernel function coefficients under certain conditions. An efficient coordinate descent/backfitting algorithm is developed. A resampling procedure for our variable selection methodology is also proposed to improve the power.

Key words: Kernel learning; LASSO; Multivariate smoothing function; Nonnegative garrote; Sparsistency; Variable selection.

1. Introduction

The variable selection problem is important in many research areas such as genomics, data mining, image analysis, text and speech analysis, and other areas with high dimensional data. In general, the input variables form an interacting network with one another, and modeling these interactions is complicated due to high order interaction terms. Most variable selection approaches in multi-dimensional nonparametric models are performed in terms of function components selection, that is, modeling the function components (including nonlinear interactions) additively and then selecting significant components. Examples of these variable selection approaches are Component Selection and Smoothing Operator (COSSO) (Lin and Zhang, 2006), Sparse Additive Models (SAMs) (Ravikumar et al., 2009), and the extension of SAMs, Variable Selection using Adaptive Nonlinear Interaction Structure in High dimensions (VANISH) (Radchenko and James, 2010). However, when the number of input variables is large and their interactions are complicated, modeling each interaction term is extremely expensive and these function components approaches may not be efficient. Liu et al. (2007) established the connection between the least squares kernel machine and linear mixed models. Zou et al. (2010) employed a nonparametric regression model with a Gaussian process, which simultaneously considers all possible interactions using the kernel. Maity and Lin (2011) proposed a score test for detecting a gene effect in the presence of possible gene-gene interactions using garrote kernel machines. They focused on testing, not variable selection. In addition, their model and our model are also different. Maity and Lin (2011)'s Gaussian kernel depends on one parameter, while ours depend on many parameters which are the number of variables.

In this paper we will focus on variable selection approaches based on the kernel machine method because the family of kernel functions is rich for regression smoothing. They have the flexibility for various models including additive functional ANOVA and nonadditive smoothing functions. For example, since any symmetric positive definite matrix is a valid Gram matrix (a symmetric matrix embedding a finite set of observations specified by the kernel function $k_j(\cdot,\cdot)$), an additive Gram matrix $\sum_{j=1}^{p}\xi_j K_j$ (ξ_js are nonnegative hyperparameters) can be used for the functional ANOVA $f\left(\boldsymbol{x}^T\right) = \sum_{j=1}^{p} f_j(x_j)$, where K_j is the Gram matrix for the jth function space f_j and $\boldsymbol{x}^T = (x_1, ..., x_p)$. According to the the Representer Theorem (Kimeldorf and Wahba, 1971), a nonparametric function can be represented using a kernel function, $f_j(x) = \sum_{l=1}^{n}\alpha_l k_j(x_l, x)$ (the dual representation), where $k_j(\cdot,\cdot)$ is the kernel function for the jth function component and α_ls are the kernel function coefficients. With penalty on the norm (or pseudo norm) of the jth function component f_j, $\|f_j\|_{\mathcal{H}_{K_j}}$, sparsity of the function components can be recovered (Lin and Zhang, 2006; Bach, 2008).

However, with nonadditive smoothing function, the kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$ is usually a nonlinear function of multidimensional $\boldsymbol{x}$, such as the Gaussian kernel function. In a model with such a kernel function, the response can no longer be expressed in terms of additive function components, and no sparse function components are available. Therefore variable selection for recovering the sparsity of $\boldsymbol{x}$ within the nonadditive function becomes challenging. To the best of our knowledge, no variable selection method based on the kernel machine has been established for nonadditive smoothing function models simultaneously recovering sparsity of input variables in a nonadditive smoothing function.

Thus, the goal of this paper is to develop a new variable selection approach on kernel machine, which is able to recover sparsity of input variables in a nonadditive and nonparametric smoothing function on Gaussian process kernels. By considering kernel functions of the form $k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left\{-\sum_{j=1}^{p}\xi_j(x_j - x_j')^2\right\}$, we model the smoothing function with a general kernel function with hyperparameters ξ_j's. By shrinking these scale parameter ξ_j's, we can select the variables. In this way, our approach can be applied to either additive or nonadditive models by choosing a different K structure. To recover sparsity of ξ_j's, an efficient coordinate descent/backfitting algorithm has been developed to achieve the regularization path for ξ_j's. In Supplementary Materials, we show how the incoherence conditions can be developed and show how sparsistency can be established under certain conditions. We also provide the asymptotic properties of our method.

The remainder of this paper is organized as follows. In Section 2, we first define the optimization function of our approach on the kernel model. We discuss the connection of our approach with the linear nonnegative garrote model and also with the kernel machine learning problem in Section A of Supplementary Materials. In Section 3, we propose our coordinate-descent updating algorithm for the solution path of the scaling parameters. In Section 4, we discuss the incoherence conditions for consistency and sparsistency. In Section 5, we present several simulation examples. We apply our method to real datasets. Section 7 contains concluding remarks.

2. Flexible Nonparametric Modeling

In this section we explain how our approach is developed under nonadditive and nonparametric model with high dimensional variables. We refer to our approach as "nonnegative garrote on kernel machine" (NGK) in the rest of the paper.

2.1 *Nonparametric Model Using Kernel Machine*

Consider an n-observation and p-predictor dataset $(\mathbf{y}, X)$, where $X = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_p]$ and $\mathbf{x}_j = (x_{j1}, ..., x_{jn})^T$ is an $n \times 1$ vector for the jth predictor, $j = 1, ...p$. In other words, $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n]^T$ where $\boldsymbol{x}_i^T$ is a $1 \times p$ vector of predictors of ith observation, $i = 1, ...n$. Among those p predictors, $q \leqslant p$ of them are considered as the true predictors. For simplicity, $\mathbf{y}$ is centered, i.e. $\sum y_i = 0$. We also standardize X such that $\sum_{l=1}^n x_{jl} = 0$ and $\sum_{l=1}^n x_{jl}^2 = 1, j = 1, ..., p$. In this paper, we only consider that the functions lie in the Reproducing Kernel

Hilbert Spaces (RKHS), $\mathcal{H}_K$. According to the Representer Theorem, the nonparametric regression model can be expressed in (Kimeldorf and Wahba, 1971)

$$\mathbf{y} = \mathbf{f}(X) + \boldsymbol{\epsilon} = K\boldsymbol{\alpha} + \boldsymbol{\epsilon}, \tag{1}$$

where $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2 I)$ and K is the kernel matrix corresponding to the function space $\mathcal{H}_K$, generated by a positive definite kernel function, $k(\boldsymbol{x}, \boldsymbol{x}')$. K is also known as a “Gram matrix” of the kernel function.

From the above expression, the smooth nonlinear function $f(\boldsymbol{x})$ is thus expressed as $\sum_{i=1}^{n} \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x})$, where $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_n)^T$ is an $n \times 1$ vector of the coefficients of the kernel function. $\boldsymbol{\alpha}$ is estimated by solving the least squares kernel machine, which minimizes the least squares error with penalized norm $\|\mathbf{f}\|^2_{\mathcal{H}_K} = \boldsymbol{\alpha}^T K \boldsymbol{\alpha}$ induced by the kernel of the function space $\mathcal{H}_K$,

$$\frac{1}{2}\|\mathbf{y} - K\boldsymbol{\alpha}\|^2 + \frac{1}{2}\lambda_0 \boldsymbol{\alpha}^T K \boldsymbol{\alpha}, \tag{2}$$

and the solution is

$$\hat{\boldsymbol{\alpha}} = (\lambda_0 I + K)^{-1}\mathbf{y}, \tag{3}$$

where $\lambda_0 > 0$ is a smoothing parameter that balances the tradeoff between goodness of fit and smoothing the curve or high dimensional surface.

2.2 *Nonnegative Garrotte on Kernel*

More specifically, we developed our NGK approach under Model (1) for kernels. A kernel matrix can be viewed as applying a componentwise function on the similarity matrix among observations. The similarity metric between $\boldsymbol{x}$ and $\boldsymbol{x}'$ can be either the negative squared Euclidean distance, $-\|\boldsymbol{x} - \boldsymbol{x}'\|^2$, or the angle (dot product), $\boldsymbol{x}^T\boldsymbol{x}'$. Both of the similarity metrics can be written in an additive form in terms of p predictors, i.e, $-\|\boldsymbol{x} - \boldsymbol{x}'\|^2 = -\{(x_1 - x_1')^2 + ... + (x_p - x_p')^2\}$ and $\boldsymbol{x}^T\boldsymbol{x}' = x_1x_1' + ... + x_px_p'$. By this additivity, the kernel matrix can be expressed as either a linear or nonlinear function of the additive form.

For example, the Gram matrix by the dot product $\boldsymbol{x}^T\boldsymbol{x}'$ among observations is the linear polynomial kernel

$$K(X) = \rho X X^T = \sum_{j=1}^{p} \rho \mathbf{x}_j \mathbf{x}_j^T = \sum_{j=1}^{p} \rho D^j,$$

where $D^j = \mathbf{x}_j\mathbf{x}_j^T$, with (k,l)th entry $d_{kl}^j = x_{jk}x_{jl}, 1 \leqslant k, l \leqslant n$, and ρ is a scale parameter. Unlike a linear kernel, the Gaussian kernel can be expressed in a nonlinear function form because the Gram matrix with entries produced by the exponential function of $-\|\boldsymbol{x} - \boldsymbol{x}'\|^2$ is

$$K(X) = \exp\left(\rho \sum_{j=1}^{p} D^j\right),$$

where D^j is the matrix with (k,l)th entry $d_{kl}^j = -(x_{jk} - x_{jl})^2$ and the (k,l)th entry of matrix $\sum_{j=1}^{p} D^j$ is $-\|\boldsymbol{x}_k - \boldsymbol{x}_l\|^2 = -\sum_{j=1}^{p}(x_{jk} - x_{jl})^2$.

More generally, let us consider a nonnegative scale parameter $\boldsymbol{\xi} = (\xi_1, ..., \xi_p)$ with ξ_j corresponding to each predictor $\mathbf{x}_j$. Then both kernels can be expressed as

$$K(\boldsymbol{\xi}, X) = g\left(\sum_{j=1}^{p} \xi_j D^j\right), \tag{4}$$

where $\xi_j \geqslant 0, j = 1, ..., p$, and function $g(\cdot)$ is a componentwise function of matrix entries. That is, for a linear polynomial kernel, all $\xi_j = \rho$ and $g(\cdot)$ is the identity function and for a Gaussian kernel, all $\xi_j = \rho$ and $g(\cdot) = \exp(\cdot)$.

By introducing such nonnegative parameters to the kernel matrix, we can develop a variable selection approach for the nonparametric regression model (1). That is, we apply an extra penalty on $\boldsymbol{\xi}$ such that optimization problem (2) is subject to $\xi_j \geqslant 0$ and $\sum \xi_j \leqslant c$ (c is a positive real number), which results in the optimization problem

$$\frac{1}{2}\left\|\mathbf{y} - K(\boldsymbol{\xi}, X)\boldsymbol{\alpha}\right\|^2 + \frac{1}{2}\lambda_0 \boldsymbol{\alpha}^T K(\boldsymbol{\xi}, X)\boldsymbol{\alpha} + n\lambda \sum \xi_j, \tag{5}$$

where $\lambda > 0$ is a tuning parameter. This method is thus referred to as “nonnegative garrote on kernel machine.”

3. Methodology

In this section, we provide an efficient algorithm to solve the objective function (5).

3.1 *Backfitting Algorithm to Update* $\boldsymbol{\xi}$

We choose initial estimate $\tilde{\boldsymbol{\alpha}} = (\lambda_0 I + K(\rho))^{-1}\mathbf{y}$, which is the least squares kernel machine solution. Liu et al. (2007) obtained this least squares kernel machine solution as a restricted maximum likelihood (REML) estimator and also showed it as the best linear unbiased predictor (BLUP). Hence, our initial estimator is a reasonable and consistent initial estimator. In Section 4 we will also show under certain conditions that the estimation consistency of $\boldsymbol{\xi}$ can be achieved.

With fixed initial consistent $\tilde{\boldsymbol{\alpha}}$, the algorithm to update $\boldsymbol{\xi}$ becomes efficient. The algorithm we propose in the following can be viewed as the non-linear version of the coordinate descent algorithm for nonnegative garrotes. In special cases where linear polynomial kernels or other additive multiple kernels are considered, our algorithm is equivalent to the least angle regression selection (LARS) algorithm for functional ANOVA. The steps of our algorithm are summarized as follows:

- *Step* 1 Initialize $\tilde{\boldsymbol{\alpha}} = (\lambda_0 I + K(\rho))^{-1}\mathbf{y}$ and $\lambda_0 = \sigma^2/\sigma_\alpha^2$ by setting all $\xi_j = \rho$ and fitting the least squares kernel machine by REML estimation, which is obtained under the mixed model point a view, where

$$\begin{aligned} \mathbf{y}|\tilde{\boldsymbol{\alpha}}, \boldsymbol{\xi} &\sim N\left(K(\boldsymbol{\xi}, X)\tilde{\boldsymbol{\alpha}}, \sigma^2 I\right), \\ \tilde{\boldsymbol{\alpha}}|\boldsymbol{\xi} &\sim N\left(\mathbf{0}, \sigma_\alpha^2 K(\boldsymbol{\xi}, X)^{-1}\right). \end{aligned}$$

- *Step* 2 Determine the initial λ for which all $\hat{\xi}_j^{(0)} = 0$

$$\lambda^{(0)} = \max_j \left\{ n^{-1} \left(\tilde{\mathbf{y}} - K(0)\tilde{\boldsymbol{\alpha}}\right)^T \left(K_j'(0)\tilde{\boldsymbol{\alpha}}\right) \right\},$$

  where $\tilde{\mathbf{y}} = \mathbf{y} - \frac{\lambda_0}{2}\tilde{\boldsymbol{\alpha}}$.

- *Step* 3 Update $\hat{\boldsymbol{\xi}}$ coordinate wise at $\lambda^{(k+1)}$ with given $\tilde{\boldsymbol{\alpha}}$ by the following equation until converge:

$$\hat{\xi}_j = \left[\tilde{\xi}_j + \frac{(\tilde{\mathbf{y}} - K\tilde{\boldsymbol{\alpha}})^T K'_j \tilde{\boldsymbol{\alpha}} - n\lambda^{(k+1)}}{\left(K'_j\tilde{\boldsymbol{\alpha}}\right)^T \left(K'_j\tilde{\boldsymbol{\alpha}}\right)} \right]_+, \tag{6}$$

where $\tilde{\xi}_j$ denotes the previously updated $\hat{\xi}_j$, and K and K'_j are calculated from previously updated $\tilde{\xi}_j$'s.

- *Step* 4 Decrease λ and repeat step 3.
- *Step* 5 Stop when model selection criterion reaches minimum or last $\lambda = 0$.

In Step 3, we derive the updating equation (6) for $\hat{\boldsymbol{\xi}}$ via an approximation of $K(\boldsymbol{\xi})$. At given λ, assuming the current iteration $\tilde{\boldsymbol{\xi}}$ is close to the minimum solution $\hat{\boldsymbol{\xi}}$, the kernel matrix can be extended in one coordinate direction around $\tilde{\xi}_j$:

$$K(\tilde{\boldsymbol{\xi}}_{-j}, \hat{\xi}_j) = K(\tilde{\boldsymbol{\xi}}) + (\hat{\xi}_j - \tilde{\xi}_j)\left(\frac{\partial K}{\partial \xi_j}\right)_{\tilde{\boldsymbol{\xi}}} + O(\|\hat{\xi}_j - \tilde{\xi}_j\|^2),$$

where $(-j)$ denotes exclusion of ξ_j. In simple notation

$$K(\tilde{\boldsymbol{\xi}}_{-j}, \hat{\xi}_j) \approx K + (\hat{\xi}_j - \tilde{\xi}_j)K'_j, \tag{7}$$

where $K = K(\tilde{\boldsymbol{\xi}})$ and $K'_j = \left(\frac{\partial K}{\partial \xi_j}\right)_{\tilde{\boldsymbol{\xi}}}$. We can express $K'_j = K \circ D^j$ for a Gaussian kernel, where "$\circ$" denotes the Schur product or entrywise product of two matrices, while $K'_j = D^j$ for a linear polynomial kernel.

The updating solution of $\hat{\xi}_j$ given $\tilde{\boldsymbol{\xi}}$ is achieved by plugging (7) into (5) and solving $\hat{\xi}_j = \arg\min (5)$ given $\tilde{\boldsymbol{\alpha}}$ and λ_0. We notice that expression (6) is similar to the backfitting algorithm (Ravikumar et al., 2009) in nonparametric additive models, except our algorithm is a version of backfitting on nonadditive models by considering the ξ_j updating step as

- *Step* a Initialize $\hat{\xi}_j = \hat{\xi}_j^{(k)}, j = 1, ..., p$ with $\tilde{\boldsymbol{\alpha}}$ given.
- *Step* b (1) Compute the residual, $\mathbf{r}_j = \tilde{\mathbf{y}} - K\tilde{\boldsymbol{\alpha}} + \tilde{\xi}_j K'_j \tilde{\boldsymbol{\alpha}}$.

  (2) Project the residual onto $\mathbf{z}_j = K'_j\tilde{\boldsymbol{\alpha}}$ and $P_j = \mathbf{z}_j^T \mathbf{r}_j$.

  (3) Update $\hat{\xi}_j = \left(\frac{P_j - n\lambda}{\|\mathbf{z}_j\|^2}\right)_+$.
- *Step* c Repeat b until the individual $\hat{\xi}_j$'s do not change.

Our algorithm has two main advantages: first, it works for $p > n$, and second it works with nonadditive kernels.

3.2 *Model selection using selection probability*

Variable selection depends on how to select the penalty parameter. However, as we discussed before, NGK variable selection is a rather new topic within the kernel machine framework. There is no similar work available to provide a perfect criterion for selecting penalty parameter λ. The performance of any criterion not only depends on the model, but also depends on the data structure. Hence, in our study, we propose to select variables according to the selection probability or frequency of individual variables. This probability is achieved by some resampling procedures with variables selected by least squares kernel machine BIC for each single resampling, where the least squares kernel machine BIC is defined as

$$BIC = \log(RSS) + \frac{df\ \log(n)}{n},$$

where $RSS = (\mathbf{y} - \hat{\mathbf{f}})^T(\mathbf{y} - \hat{\mathbf{f}})$. For the given minimum solution $\hat{\boldsymbol{\xi}}$, the estimated function $\mathbf{f}$ can be expressed as $\hat{\mathbf{f}} = S\mathbf{y}$, where S is the smoothing matrix. For the least squares error kernel machine, $S = K(\hat{\boldsymbol{\xi}})\left(\lambda_0 I + K(\hat{\boldsymbol{\xi}})\right)^{-1}$, the degrees of freedom of the kernel machine smoother S is defined as $df = \text{Trace}(S)$. We propose two resampling procedures: one is based on bootstrap for large sample size and the other is based on permutation for small sample size. Our resampling procedures are further described in B.8 of Supplementary Materials and Section 6.2, respectively.

4. Some Theoretical Properties

Consistency in variable selection problem includes two aspects: estimation consistency and model selection consistency. Between the two, one does not necessarily imply the another. The model consistency is also called sparsistency, shorthand for "sparsity pattern consistency" (Ravikumar et al., 2009). We first show that the NGK estimator is $\sqrt{n/\log(p)}$ consistent (see

Theorem 1 of Supplementary Materials B.2). Lemma 2 states the necessary and sufficient conditions for $\hat{\boldsymbol{\xi}}$ to be consistent (See Supplementary Material A.3). In this section, we summarize incoherence condition for recovery of sparsity and show how we derive this condition (See Supplementary Material B.4) and recovery of sparsity (See Theorem 2 of Supplementary Material B.5.)

We first define some notations. Let $\boldsymbol{\xi}^*$ and $\hat{\boldsymbol{\xi}}$ represent the true $\boldsymbol{\xi}$ and minimum solution of (5), respectively. Suppose vector $\boldsymbol{\xi}^*$ is sparse, i.e. some $\xi_j^* = 0$. Without loss of generality, denote $\boldsymbol{\xi}^* = (\xi_1^*, ..., \xi_p^*)^T = \left(\boldsymbol{\xi}_1^{*T}, \boldsymbol{\xi}_0^{*T}\right)^T$, where $\boldsymbol{\xi}_1^*$ is the vector of the first q nonzero ξ_i^*'s and $\boldsymbol{\xi}_0^*$ is the zero vector. Define the nonzero index set of $\boldsymbol{\xi}^*$ as $\mathcal{A} := \{j \in \{1, ..., p\} | \xi_j^* > 0\}$, and denote $\hat{\mathcal{A}} := \{j \in \{1, ..., p\} | \hat{\xi}_j > 0\}$ as the nonzero index set of $\hat{\boldsymbol{\xi}}$. Note that $\mathcal{A}$ has relatively small cardinality $q = |\mathcal{A}|$, the number of true nonzero ξ_j's. Hence, the estimation consistency requires $\hat{\boldsymbol{\xi}} - \boldsymbol{\xi}^* \to \mathbf{0}$ as $\log(p)/n \to 0$, and the sparsistency requires $\lim_n P(\hat{\mathcal{A}} = \mathcal{A}) \to 1$.

Denote the true $\boldsymbol{\alpha}$ vector as $\boldsymbol{\alpha}^*$. For a given $\boldsymbol{\alpha}^*$ and estimate $\tilde{\boldsymbol{\alpha}}$ we define the following matrices and their respective partitions:

$$Z = [\mathbf{z}_1, ..., \mathbf{z}_p] = [Z_1, Z_0] = \left[\left\{K_j'(\boldsymbol{\xi}^*)\boldsymbol{\alpha}^*\right\}_{1 \leqslant j \leqslant q}, \left\{K_j'(\boldsymbol{\xi}^*)\boldsymbol{\alpha}^*\right\}_{q+1 \leqslant j \leqslant p}\right] \tag{8}$$

$$\tilde{Z} = [\tilde{\mathbf{z}}_1, ..., \tilde{\mathbf{z}}_p] = [\tilde{Z}_1, \tilde{Z}_0] = \left[\left\{K_j'(\boldsymbol{\xi}^*)\tilde{\boldsymbol{\alpha}}\right\}_{1 \leqslant j \leqslant q}, \left\{K_j'(\boldsymbol{\xi}^*)\tilde{\boldsymbol{\alpha}}\right\}_{q+1 \leqslant j \leqslant p}\right], \tag{9}$$

where $K_j'(\boldsymbol{\xi}^*) = \left.\frac{\partial K}{\partial \xi_j}\right|_{\boldsymbol{\xi}^*}$, obtained by taking the partial derivative of the componentwise entries of K. Some covariance matrices are also defined as

$$\begin{aligned} \Sigma_{11} &= \left(n^{-1} Z_1^T Z_1\right), \quad \Sigma_{01} = \left(n^{-1} Z_0^T Z_1\right), \\ \tilde{\Sigma}_{11} &= \left(n^{-1} \tilde{Z}_1^T \tilde{Z}_1\right), \quad \tilde{\Sigma}_{01} = \left(n^{-1} \tilde{Z}_0^T \tilde{Z}_1\right), \end{aligned} \tag{10}$$

where Σ_{11} and $\tilde{\Sigma}_{11}$ are assumed to be invertible.

4.1 *Incoherence Condition for Recovery of Sparsity*

We consider the following zero noise incoherence condition on the Z matrix:

$$\Sigma_{01}\Sigma_{11}^{-1}\mathbf{1} - \frac{\lambda_0}{2n\lambda} Z_0^T P \boldsymbol{\alpha}^* \preceq (1 - \gamma)\mathbf{1}, \tag{11}$$

where $\gamma \in (0,1]$, and $P = [I - Z_1(Z_1^T Z_1)^{-1} Z_1^T]$ is a projection matrix. (11) has its counterpart in LASSO variable selection (12) and is derived from $\boldsymbol{\alpha}^*$. With given additional conditions, it is equivalent to $\tilde{\Sigma}_{01}\tilde{\Sigma}_{11}^{-1}\mathbf{1} - \frac{\lambda_0}{2n\lambda}\tilde{Z}_0^T\tilde{P}\tilde{\boldsymbol{\alpha}} \preceq (1-\tilde{\gamma})\mathbf{1}$, which is derived from $\tilde{\boldsymbol{\alpha}}$ (see Supplementary Materials B.5), where $\tilde{\gamma} \in (0,1]$ and $\tilde{P} = [I - \tilde{Z}_1(\tilde{Z}_1^T\tilde{Z}_1)^{-1}\tilde{Z}_1^T]$.

5. Simulation Results

We conducted several simulation studies to demonstrate the advantage of NGK in Section 5.1-5.3. Further simulation studies are also summarized in Supplementary Materials A.6. We also provide simulation result of sensitivity test on the choice of initial $\tilde{\boldsymbol{\alpha}}$ in B.7 of Supplementary Materials.

5.1 *Comparison with Linear LASSO*

In many cases, even though the underlying true model is nonlinear, variable selection using linear LASSO can be easily used since algorithms for linear LASSO are already available (e.g, LARS). These algorithms might work well as long as the following incoherence condition is satisfied,

$$\left| X_0^T X_1 (X_1^T X_1)^{-1} \mathrm{sgn}(\boldsymbol{\beta}_1^*) \right| \preceq \mathbf{1}, \tag{12}$$

where X_0 and X_1 are the matrices of irrelevant and relevant predictors, and $\boldsymbol{\beta}_1^*$ represents the vector of true nonzero β_j's.

In this section we show a special case that using the NGK method sparsity of input variables can be recovered, while linear LASSO fails due to unsatisfied condition (12).

We use the same 3-variable setting by Zhao and Yu (2006), where they used simulation to demonstrate the incoherence condition in linear LASSO. First, we generated iid random variables $\mathbf{x}_1$, $\mathbf{x}_2$, $\boldsymbol{\epsilon}$ and $\mathbf{e}$ from $N(0,1)$ with sample size $n = 100$. The third predictor $\mathbf{x}_3$ is generated by $\mathbf{x}_3 = a\mathbf{x}_1 + b\mathbf{x}_2 + c\mathbf{e}$, where $a = 2/3$, $b = 2/3$ and $c = 1/3$, and the response is generated by

$$\mathbf{y} = \beta_1^*\mathbf{x}_1 + \beta_2^*\mathbf{x}_2 + \boldsymbol{\epsilon},$$

where $\beta_1^* = 2$ and $\beta_2^* = 3$. Denote $X_1 = [\mathbf{x}_1, \mathbf{x}_2]$ and $X_0 = [\mathbf{x}_3]$. Zhao and Yu (2006) showed that, with this setting, $\left(\frac{1}{n}X_0^T X_1\right)\left(\frac{1}{n}X_1^T X_1\right)^{-1} = \left(\frac{2}{3}, \frac{2}{3}\right)$, thus the incoherence condition (12) for linear LASSO is never satisfied with $\mathrm{sgn}(\beta_1^*) = \mathrm{sgn}(\beta_2^*)$.

However the incoherence condition (11) of NGK provides a different incoherence condition that is satisfied. To demonstrate this, we consider using a linear polynomial kernel. Thus with $\boldsymbol{\xi}_1^* = (\xi_1^*, \xi_2^*)^T$ and $\xi_3^* = 0$, we have $K(\boldsymbol{\xi}^*) = \xi_1^*\mathbf{x}_1\mathbf{x}_1^T + \xi_2^*\mathbf{x}_2\mathbf{x}_2^T$. Using the notation in Section 4, we obtain

$$\begin{aligned}\tilde\Sigma_{01}\tilde\Sigma_{11}^{-1}\mathbf{1} &= \begin{bmatrix} a\tilde{\boldsymbol{\alpha}}^T\mathbf{x}_3\mathbf{x}_1^T\tilde{\boldsymbol{\alpha}} & b\tilde{\boldsymbol{\alpha}}^T\mathbf{x}_3\mathbf{x}_2^T\tilde{\boldsymbol{\alpha}} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{\alpha}}^T\mathbf{x}_1\mathbf{x}_1^T\tilde{\boldsymbol{\alpha}} & 0 \\ 0 & \tilde{\boldsymbol{\alpha}}^T\mathbf{x}_2\mathbf{x}_2^T\tilde{\boldsymbol{\alpha}} \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= a\frac{\tilde{\boldsymbol{\alpha}}^T\mathbf{x}_3}{\tilde{\boldsymbol{\alpha}}^T\mathbf{x}_1} + b\frac{\tilde{\boldsymbol{\alpha}}^T\mathbf{x}_3}{\tilde{\boldsymbol{\alpha}}^T\mathbf{x}_2}\end{aligned} \tag{13}$$

and

$$\begin{aligned}\frac{\lambda_0}{2n\lambda}\tilde Z_0^T\tilde P\tilde{\boldsymbol{\alpha}} &= \frac{\lambda_0}{2n\lambda}\tilde Z_0^T(I - \tilde Z_1(\tilde Z_1^T\tilde Z_1)^{-1}\tilde Z_1^T)\tilde{\boldsymbol{\alpha}} \\ &= \frac{\lambda_0}{2n\lambda}\tilde Z_0^T\left[I - \left(\frac{1}{n}\mathbf{x}_1\mathbf{x}_1^T + \frac{1}{n}\mathbf{x}_2\mathbf{x}_2^T\right)\right]\tilde{\boldsymbol{\alpha}} \\ &= \frac{\lambda_0}{2n\lambda}\tilde{\boldsymbol{\alpha}}\left(\mathbf{x}_3\mathbf{x}_3^T - a\mathbf{x}_3\mathbf{x}_1^T - b\mathbf{x}_3\mathbf{x}_2^T\right)\tilde{\boldsymbol{\alpha}}.\end{aligned} \tag{14}$$

In equations (13)-(14), we use the fact that for independent random normals, $\frac{1}{n}\mathbf{x}_i^T\mathbf{x}_j = \delta_{ij}, i, j = 1, 2$ and $\frac{1}{n}\mathbf{x}_3^T\mathbf{x}_j = a$ or b for $j = 1$ or 2. Given $\tilde{\boldsymbol{\alpha}} = (\lambda_0 I + K(\tilde{\boldsymbol{\xi}}))^{-1}\mathbf{y}$ with $\tilde{\boldsymbol{\xi}} = (1, 1, 1)^T$, we can calculate the left-hand side of (11). For demonstration with one simulation example, we calculate two incoherence condition curves vs λ and λ_0, respectively. For the first curve vs. λ, we fix $\lambda_0 = 0.0026$ estimated by REML. For the second curve, we fix $\lambda = 1.516$, where we choose the model with minimum BIC and vary λ_0.

Figure 1(a)-(b) show two plots: (a) is for the incoherence condition values vs. λ when $\lambda_0 = 0.0026$ and (b) is for the incoherence condition values vs. λ_0 when $\lambda = 1.5616$. They show that for certain λ and λ_0 values, the incoherence condition values are smaller than one, thus condition (11) is satisfied and there is a possibility that the variable selection procedure of NGK can recover sparsity of those irrelevant variables. Figure 1(c)-(d) also shows the plot of the regularization path for linear LASSO (c) and NGK (d). It can be seen that for linear LASSO, β_3 is always non-zero on the path except when $\lambda = 0$, which means linear LASSO will always select β_3. However, the regularization path of NGK shows that for some λ, $\xi_3 = 0$, but both ξ_1 and ξ_2 are greater than zero, providing the possibility to select the correct variable set. The dashed line in Figure 1(d) indicates where we based model selection on minimum BIC.

[Figure 1 about here.]

5.2 *Simulation Example 2*

In this example, we consider fixed f and generate response y using

$$y = f + \epsilon = 10\cos(x_1) + 3x_2^2 + 5\sin(x_3) + 6\exp(x_4/3)x_4 + 8\cos(x_5) + x_5x_2x_1 + \epsilon, \qquad (15)$$

where $\epsilon \sim N(0,1)$ and $x_j \sim U(0,1), j = 1, ..., p$. Function f in this simulation is similar to the one used in Liu et al. (2007). In this example, we consider $p = 10$ total predictors where the first $q = 5$ are relevant. Again, three settings with sample sizes $n = 64$, 128, and 256 were generated with a total of 200 runs per setting.

The selection frequency of 200 runs are listed in Table B.9.3 and a selected example of the solution paths for two NGK methods and the BIC curves are shown in Figure B.9.2. Here five statistics of 200 runs are summarized in Table 1. They are "False Positive Rate (FP-rate)", "False Negative Rate (FN-rate)", "Model Size (MS)", "Residual Sum of Squares (RSS)" and "Mean Squared Error (MSE)", where FP-rate $= \frac{\#False\,Positive}{\#False\,Positive + \#True\,Negative}$, FN-rate $= \frac{\#False\,Negative}{\#False\,Negative + \#True\,Positive}$, RSS $= \sum_i^n (y_i - \hat{f}_i)^2/n$ and MSE $= \sum_i^n (f_i - \hat{f}_i)^2/n$ are calculated for each individual run. It can be seen that all three methods have the same zero FN rate. When the sample size is $n = 64$, COSSO approach has the worst estimation accuracy while NGK performs well. NGK is comparable to COSSO in terms of FP rate.

When sample size increases, NGK methods also perform well in terms of FP rate. It can be seen that with increasing sample size, COSSO still provide worst estimation accuracy, while NGK methods seem to estimate more accurately and to perform well in term of FP rate. In this example, the Gaussian kernel NGK method is considered to be the best method not only because it performs as well as the other methods in terms of FN and FP rates, but also because it has the best estimation accuracy. In simulation, we set the variance of ϵ as 1. RSS values of COSSO are less than true value 1, while our NGK approaches are closer to 1. This means that NGK approaches can more accurately estimate the variance of ϵ than COSSO.

[Table 1 about here.]

5.3 *Simulation Example 3*

In this example, we consider a special case with $p = 80$ and $n = 64$. Since $p > n$, we found that COSSO does not work well because it provides very unstable and numerical problems. Hence we compare our two NGK approachs corresponding to the Gaussian and linear polynomial kernel NGK methods using our backfitting algorithm. Example 3 has the same true function as Example 2. The first five predictors are relevant and a total of 400 runs have been simulated. Since computing becomes more intensive when n is large, we only demonstrate the results with $n = 64$. Figure A.9.3 shows example solution paths for Example 3 by the Gaussian and linear polynomial kernel NGK methods. In both cases the number of variables selected by BIC is greater than 5.

Because of the number of predictors, we portray the selection frequency or probability of each variable for 400 runs in Figure 2, which shows that the first five variables have selection probability very close to 1.0 for both methods. In addition, both methods show the same behavior in that the first five variables are clearly separated from the remaining 75 variables in terms of selection probability. However, the linear polynomial kernel method has a slightly higher FN rate (lower selection probability for five true variables) than the

Gaussian kernel approach. This advantage of Gaussian kernel may be contributed to the high order interaction term in (15), and linear polynomial kernel is a multiple kernel which does not count interactions. From Table 2, we can see the FP and FN rates for Example 3. Compared to Example 2, the FP rate of Example 3 for the Gaussian kernel approach is comparable as 0.09 and 0.08, respectively. For the linear polynomial kernel method, the FP rate increases slightly. The major difference is that in Example 2 FN-rates are zero for both methods, but are nonzero in Example 3. This is reasonable since inclusion of many irrelevant predictors deteriorates variable selection performance. We also realized that the average model size is greater than 5 from Table 2, which reflects the fact that including more irrelevant predictors will result in more irrelevant predictors being selected.

[Table 2 about here.]

[Figure 2 about here.]

As a referee recommends, we further conduct simulation to compare our NGK approach with LASSO. We chosed a tuning parameter of LASSO using BIC too. This result is also summaried in Table 2. The performance of LASSO was pretty good in terms of FP and FN rates. However, LASSO provided the larger RSS and MSE. These results means that LASSO works well for variable selection with less accurate function estimation. The estimated model size using LASSO is smaller than the true one.

6. Applications

In this section, we describe the application of our method in two practical settings. For the first application, we used polynomial kernel NGK, and the result is summarized in Supplementary Materials B.8. For the second application, we applied Gaussian kernel NGK, and the result is presented in this section.

6.1 *Gene Selection in Pathway Data*

We apply our Gaussian kernel NGK method to a set of diabetes data from Mootha et al. (2003). They provided pathway based analysis to classify two phenotypes, 17 normal and 18 Type II diabetes patients. A pathway is a predefined set of genes that serve a particular cellular or physiological function. They showed that pathway based analysis can detect coordinate subtle changes among a set of genes. It is known that genes in a pathway are not independent of one another and interact with unknown structure. The top significant pathways related to the diabetes disease have been identified (Mootha et al., 2003). Pathway 133 ("Oxidative phosphorylation"), pathway 4 ("Alanine and aspartate metabolism") and pathway 140 ("MAP00252-Alanine-and-aspartate metabolism") are three interesting ones which contain a total of 58, 18 and 22 genes, respectively. We apply our approach for each pathway.

For each pathway we label the genes by their appearance index, gene 1, gene 2 and so on. Note the same gene index from two pathways does not imply the same gene. Since the 18 genes in pathway 4 are all included the 22 genes in pathway 140, we use the gene index of pathway 140 to label genes in both pathways. Thus, genes 4, 5, 19 and 20 do not appear in pathway 4. Hence, in this application, the data set structure is $(\mathbf{y}, X)$ with a total of $n = 35$ observations and $p = 58, 18$ and 22 predictors, respectively. The response is the outcome of glucose levels.

Figures B.9.8 (a), (c) and (e) in Supplementary Materials plot the solution paths of the ξ_j's corresponding to genes for three pathways. Figures B.9.8 (b), (d) and (f) in Supplementary Materials show the BIC curves to select genes where a total of 13, 7 and 9 genes are selected, respectively. The index sets for the selected genes of the three pathways by the Gaussian kernel NGK method are $\hat{\mathcal{A}}_{133} = \{1, 4, 5, 14, 19, 23, 29, 31, 34, 41, 51, 53, 57\}$, $\hat{\mathcal{A}}_4 = \{8, 10, 11, 12, 13, 14, 21\}$ and $\hat{\mathcal{A}}_{140} = \{5, 8, 10, 11, 12, 13, 14, 18, 21\}$. However, as discussed for our application (B.7 of Supplementary Materials) and simulation of Example 3 (Section 5.3), variable selection depending on single draw may not be powerful even if the observation number is large. In this data, there are only 35 observations. So, we make the final model selection by using selection probability which is described in the following paragraphs.

6.2 *Model Selection Using Resampling Approach*

In this section we propose using a residual permutation procedure to repeat the variable selection process and counting the total frequency/probability of each predictor.

- *Step 1*: Apply the Gaussian kernel NGK variable selection method to the original dataset using the backfitting algorithm introduced in Section 3 and obtain the selected variables $\hat{\boldsymbol{\xi}} = (\hat{\xi}_j)^T_{j \in \hat{\mathcal{A}}}$. Use $\hat{\boldsymbol{\xi}}$ to fit the Gaussian kernel machine again to obtain new $\hat{\boldsymbol{\alpha}}$ and new λ_0 by REML such that $\hat{\mathbf{y}} = K(\hat{\boldsymbol{\xi}})\hat{\boldsymbol{\alpha}}$. Obtain the residual $\hat{\boldsymbol{\epsilon}} = \mathbf{y} - \hat{\mathbf{y}}$. Center $\hat{\boldsymbol{\epsilon}}$ by subtracting its mean.
- *Step 2*: Permute the residual $\hat{\boldsymbol{\epsilon}}$ to get new $\hat{\boldsymbol{\epsilon}}^*$ and simulate outcomes as $\mathbf{y}^* = K(\hat{\boldsymbol{\xi}})\hat{\boldsymbol{\alpha}} + \hat{\boldsymbol{\epsilon}}^*$.
- *Step 3*: Based on the new dataset $(\mathbf{y}^*, X)$ with fixed initial $\hat{\boldsymbol{\alpha}}$ and fixed λ_0, apply the NGK variable selection method again and obtain the selected gene set.
- *Step 4*: Repeat Steps 2-3 for a large number of iterations (e.g, 3,000 times).
- *Step 5*: Obtain the empirical probability/frequency of selecting each variable.

The results of NGK permutation procedure are summarized in Figure 3. If we take 60% as the threshold, the sets of genes selected are $\tilde{\mathcal{A}}_{133} = \{4, 5, 14, 19, 23, 31, 34, 41, 53\}$, $\tilde{\mathcal{A}}_4 = \{8, 10, 11, 12, 21\}$ and $\tilde{\mathcal{A}}_{140} = \{5, 12, 21\}$, respectively. Because pathway 4 is a subset of pathway 140, we plot the results of the two pathways in one plot, Figure 3(b). Compared with $\hat{\mathcal{A}}$, we see that $\tilde{\mathcal{A}} \subset \hat{\mathcal{A}}$. For example, for pathway 133 four extra genes selected using a single NGK step are $\{1, 29, 51, 57\}$. Especially for gene 1, the selection probability is less than 20% by permutation approach.

Interesting observations for pathway 4 and pathway 140 are found in Figure 3(b). We observe that some of the genes are not significantly related to the response, such as genes $\{1, 2, 3, 7, 9, 15, 22\}$. In both pathways, the selection probabilities remains small for those genes. Another observation is that some genes are significantly related to the response and retain a high selection probability in both pathways (gene 21, for example). Genes $\{10, 11, 12, 13, 14\}$ appear to group a gene segment with similar selection probability. An interesting gene is gene 5, which does not appear in pathway 4. Gene 5 has the highest selection probability in pathway 140. While gene 5 is present in pathway 140, the selection probabilities of $\{8, 10, 11\}$ are smaller than in pathway 4. This may indicate that some possible interaction or correlation may occur between gene 5, gene 8 and the gene segment$\{10, 11\}$, but researchers need to further invetigate biological justification.

[Figure 3 about here.]

7. Discussion

In this paper, we have proposed a new variable selection approach to recover sparsity of the multivariate input variable in a nonadditive smoothing function. Our approach can be addressed as a nonnegative garrote variable selection procedure with kernel machine. The method we proposed has several advantages: (1) it can recover sparsity as well as model any order interactions automatically; (2) it is applicable not only to nonadditive smoothing functions, but also to additive model by choosing a different kernel; and (3) it establishes a connection among several existing methods including linear nonnegative garrote and kernel learning. The main contribution of the paper is to (a) consider a non-linear variant of kernel learning and (b) to develop this to the non-negative garrote.

We note that $Q(\xi)$ is convex when ξ_is are all small but are not guaranteed to be convex when some of them are large. Because of the page limitation, this result is summarized in Table B.9.4 of supplementary material. The robustness of the initial values might be a sign of the convergence of our algorithm. However, it is important to theoretically show the convergence in general cases, which requires future work.

Furthermore, in this paper, we suggested resampling variable selection procedures in two cases: when n is large and when n is small. Thus consistency and convergence rate of resampling/bootstrapping on NGK approaches are interesting future topics as well. The intensive simulation studies and derivation of theoretical justification will be required in furture research. Derivation of theoretical distribution will be useful to provide guidance on determining this threshold. Note that in our data analysis, we chose the threshold 0.6 using cross-validation.

Kernel can be chosen by treating Kernel section as model selection. Liu et al. (2007) provided Kernel selection criterion using AIC and BIC. However, if researchers believe that variables are independent of each other, a polynomial kernel is a possible choice. Otherwise, Gaussian kernel may be a practical choice because Gaussian can be expressed as a sum of higher order polynomial kernels. We note that our NGK approach is useful for continuouse variables. If variables are categorical variables, both polynomial and Gaussian kernels are not useful because the correlation or distance among variables are not meaningful. Other Kernel (Korsgaard et al, 1998) can represent the meaning of categorical variables.

We apply our approach for each pathway. We observe different selection probabilities among Gene 5, Gene 8, and Genes {10,11}, which are shown in both two pathways. But some of them have high selection probabilities in one pathway, while they have small selection probabilities in the other pathway. These results need to be further investigated to assess the biological meanings.

Supplementary Materials

Web Appendices, Tables, Figures, and program code referenced are available under the Paper Information link at the Biometrics website `http://www.tibs.org/biometrics`.

REFERENCES

Bach, F. (2008). Consistency of the Group Lasso and Multiple Kernel Learning. *Journal of Machine Learning Research*, 9, 1179-1225.

Breiman, L. (1995). Better Subset Regression Using the Nonnegative Garrote. *Technometrics*, 37, 373-384.

Kimeldorf, G. and Wahba, G. (1971). Some Results on Tchebychefian Spline Functions. *Journal of Mathematical Analysis and Applications*, 33, 82-95.

Korsgaard, I., Madsenm P. and Jensen, J. (1998). Bayesian inference in the semiparametric log normal frailty model using Gibbs sampling. *Genetics Selection Evolution*, 30, 241-256.

Lanckriet, G., Cristianini, N., Bartlett, P., Ghaoui, L. E. and Jordan, M. I. (2004). Learning the Kernel Matrix with Semi-Definite Programming. *Journal of Machine Learning Research*, 5, 27-72

Lin, Y. and Zhang, H. H. (2006). Component Selection and Smoothing in Multivariate Nonparametric Regression. *The Annals of Statistics*, 34, 2272-2297.

Liu, D., Lin, X. and Ghosh, D. (2007). Semiparametric Regression of Multi-Dimensional Genetic Pathway Data: Least Squares Kernel Machines and Linear Mixed Models. *Biometrics*, 63, 1079-1088.

Micchelli, C. A. and Pontil, M. (2005). Learning the Kernel Function via Regularization. *Journal of Machine Learning Research*, 6, 1099-1125.

Maity, A. and Lin, X. (2011). Powerful tests for detecting a gene effect in the presence of possible gene-gene interactions using garrote kernel machines. *Biometrics*, 67, 1271-1284.

Mootha, V. K. Lindgren, C. M., Eriksson, K., Subramanian, A., Sihag, S., and et al (2003). PGC-l alpha-Responsive Genes Involved in Oxidative Phosphorylation are Coordinately Downregulated in Human Diabetes. *Nature Genetics*, 34, 267-273.

Radchenko, P. and James, G. M. (2010). Variable Selection Using Adaptive Nonlinear Interaction Structures in High Dimensions. *Journal of the American Statistical Association*, 105, 1541-1553.

Rakotomamonjy, A., Bach, F., Canu, S. and Grandvalet, Y. (2008). SimpleMKL. *Journal of Machine Learning Research*, 9, 2491-2521.

Ravikumar, P., Lafferty, J., Liu, H. and Wasserman, L. (2009). Sparse Additive Models. *Journal of the Royal Statistical Society, Series B*, 71, 1009-1030.

Yuan, M. (2007). Nonnegative Garrote Component Selection in Functional ANOVA Models. *Proceedings of AI and Statistics, AISTATS*, 660-666.

Zhao, P. and Yu, B. (2006). on Model Selection Consistency of Lasso. *Journal of Machine Learning Research*, 7, 2541-2563.

Zou, F., Huang, H., Lee, S. and Hoeschele, I. (2010). Nonparametric Bayesian Variable Selection with Applications to Multiple Quantitative Trait Loci Mapping with Epistasis and Gene-Environment Interaction. *Genetics*, 186, 385-394.

Received 0000 0000. *Revised 0000* 0000.

Accepted 0000 0000.

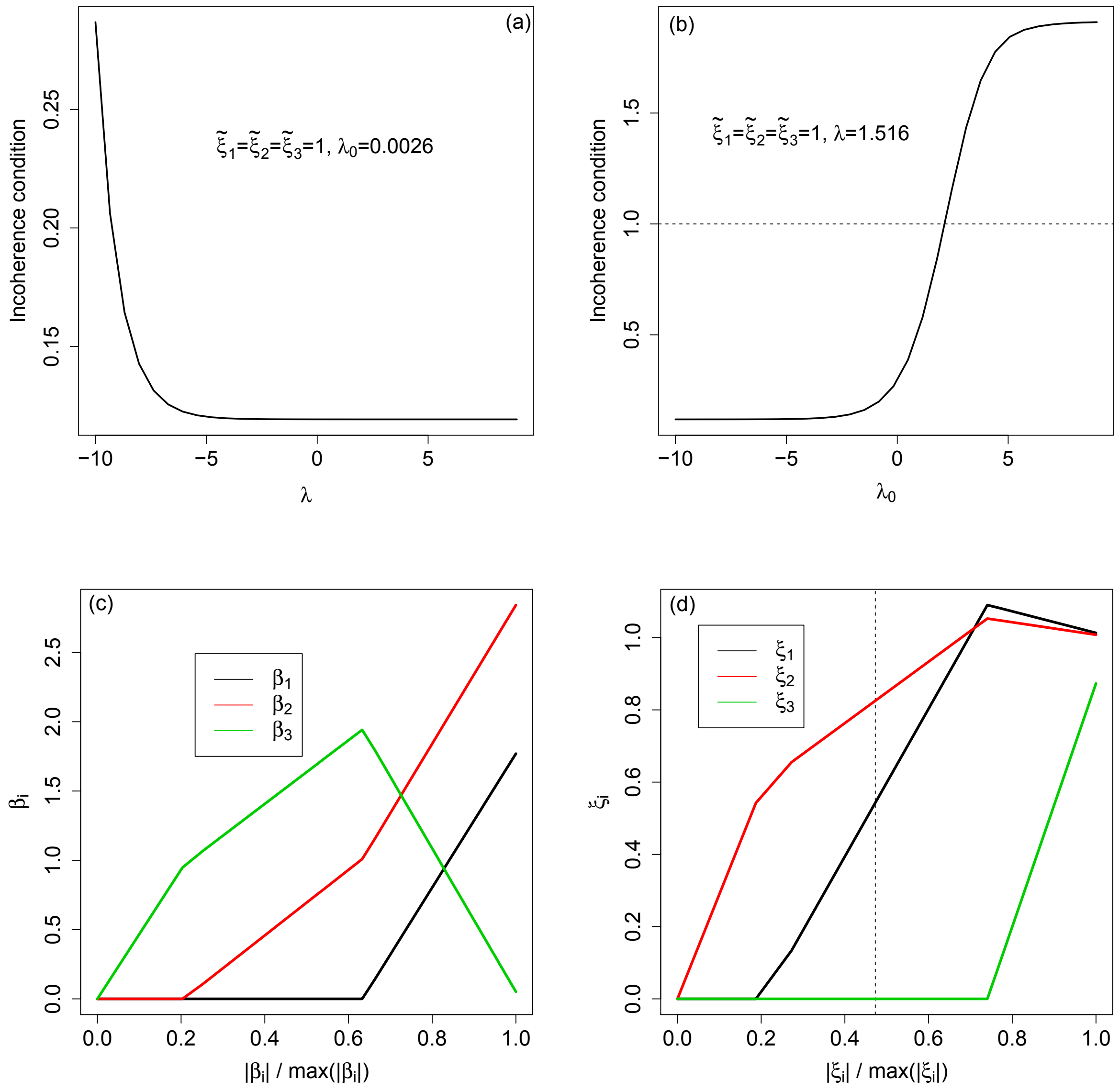

Figure 1. Incoherence condition values which depends on λ and λ_0 and Comparison of solution paths between LASSO and NGK: (a) Incoherence condition values vs. λ with λ_0 fixed at 0.0026, (b) Incoherence condition values vs. λ_0 with λ fixed at 1.516, where these two fixed values were obtained such that the condition is satisfied at these two values, (c) solution path of β_i's for linear LASSO, and (d) solution path of ξ_i's for NGK. All plots use initial $\tilde{\boldsymbol{\alpha}} = \Delta^{-1}(\tilde{\boldsymbol{\xi}})y$ with $\tilde{\boldsymbol{\xi}} = (1, 1, 1)^T$.

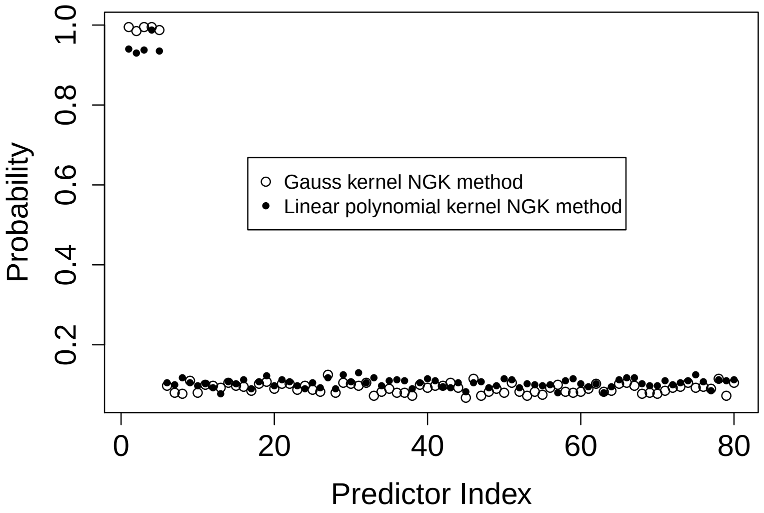

Figure 2. Selection probability of each predictor in Simulation Example 3 for 400 runs using two NGK methods.

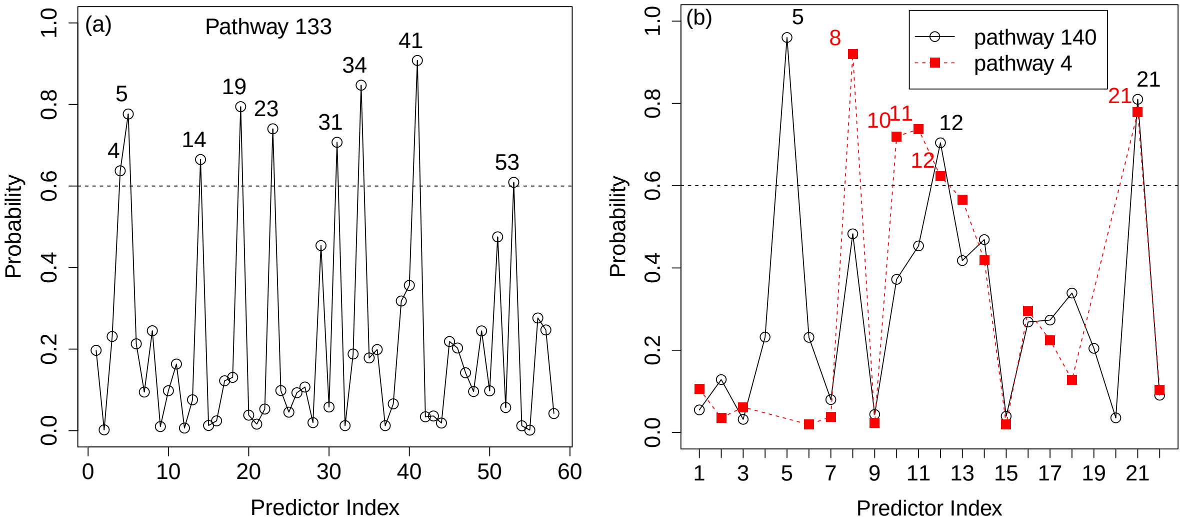

Figure 3. Selection probability of each gene using the residual permutation method for pathway 133 (a), and pathway 140 and 4 (b), with a total of 3000 runs for each pathway.

Table 1
Simulation results of Simulation Example 2 for 200 runs.

		FP-rate	FN-rate	MS	RSS	MSE
$n = 64$	NGK Gauss	0.09(0.11)	0.00(0.00)	5.59(0.83)	1.02(0.24)	0.34(0.09)
	NGK Poly	0.05(0.09)	0.00(0.00)	5.34(0.56)	1.14(0.20)	0.35(0.08)
	COSSO	0.04(0.08)	0.00(0.00)	5.32(0.61)	0.84(0.20)	0.99(0.18)
$n = 128$	NGK Gauss	0.02(0.05)	0.00(0.00)	5.01(0.31)	1.15(0.17)	0.27(0.05)
	NGK Poly	0.04(0.07)	0.00(0.00)	5.23(0.46)	1.20(0.15)	0.31(0.05)
	COSSO	0.01(0.04)	0.00(0.00)	5.06(0.27)	0.95(0.14)	1.02(0.13)
$n = 256$	NGK Gauss	0.01(0.03)	0.00(0.00)	5.04(0.18)	1.12(0.12)	0.20(0.05)
	NGK Poly	0.01(0.03)	0.00(0.00)	5.03(0.17)	1.22(0.11)	0.29(0.03)
	COSSO	0.01(0.03)	0.00(0.00)	5.05(0.21)	0.98(0.09)	1.01(0.09)

Table 2

Simulation results of Simulation Example 3 for 400 runs.

		FP-rate	FN-rate	MS	RSS	MSE
$n = 64$	NGK Gauss	0.08(0.04)	0.004(0.027)	11.82(3.50)	1.57(0.29)	1.01(0.23)
	NGK Poly	0.08(0.10)	0.036(0.122)	12.07(12.91)	1.38(1.47)	0.96(1.36)
	LASSO	0.03(0.01)	0.003(0.023)	7.40(1.18)	1.95 (0.32)	1.17(0.31)