Nonlinear Dimension Reduction with Kernel Sliced Inverse Regression

Yi-Ren Yeh, Su-Yun Huang, and Yuh-Jye Lee

Abstract—Sliced inverse regression (SIR) is a renowned dimension reduction method for finding an effective low-dimensional linear subspace. Like many other linear methods, SIR can be extended to nonlinear setting via the "kernel trick". The main purpose of this article is two-fold. We build kernel SIR in a reproducing kernel Hilbert space rigorously for a more intuitive model explanation and theoretical development. The second focus is on the implementation algorithm of kernel SIR for fast computation and numerical stability. We adopt a low rank approximation to approximate the huge and dense full kernel covariance matrix and a reduced singular value decomposition technique for extracting kernel SIR directions. We also explore kernel SIR's ability to combine with other linear learning algorithms for classification and regression including multiresponse regression. Numerical experiments show that kernel SIR is an effective kernel tool for nonlinear dimension reduction and it can easily combine with other linear algorithms to form a powerful toolkit for nonlinear data analysis.

Index Terms—dimension reduction, eigenvalue decomposition, kernel, reproducing kernel Hilbert space, singular value decomposition, sliced inverse regression, support vector machines.

1 Introduction

Dimension reduction is an important topic in machine learning and data mining. The main demand comes from complex data analysis, data visualization and parsimonious modeling [1], [2]. Modern data are usually complex, high-dimensional and with nonlinear structures. A dimension reduction technique helps us to characterize the key data structure using only a few main features ranked by their importance. It thus provides a way for data visualization to gain better intuitive insights of the underlying data. The most popular dimension reduction method is probably the principal component analysis (PCA), which is an unsupervised method. In contrast, the sliced inverse regression (SIR) [3] extracts the dimension reduction subspace, called the effective dimension reduction (*e.d.r.*) subspace, based on the covariance matrix of input attributes inversely regressed on the responses. SIR can be viewed as a supervised companion of PCA for linear dimension reduction. SIR has won its reputation to perform well in dimension reduction and related applications and has gained great attention in statistical literature [2], [3], [4], [5], [6], [7]. The work by Wu [8] extends the classical SIR to nonlinear dimension reduction via the kernel method. This extension is named kernel sliced inverse regression (KSIR), and is applied to support vector classification. In this article we reformulate KSIR in a reproducing kernel Hilbert space setting and emphasize on KSIR's implementation techniques, its ability to combine with other linear algorithms and its applications to classification as well as regression including multiresponse regression. We adopt a low rank approximation to approximate the huge and dense kernel covariance matrix and also introduce a reduced singular value decomposition technique for estimating kernel *e.d.r.* directions in KSIR implementation. These reduction techniques will speed up the computation and increase the numerical stability. In a nutshell, our learning framework for a complicated data set is as follows. We extract a kernel *e.d.r.* subspace, also named feature *e.d.r.* subspace, in which the nonlinear structure of data has been embedded. Then we can apply linear learning algorithms to the images of the original complicated data in this low-dimensional feature *e.d.r.* subspace. Under this learning framework we are able to generate nonlinear models for learning tasks such as regression and multi-class classification in an efficient way, compared to a very popular SVM tool LIBSVM [9].

The subsequent sections are organized as follows. Section 2 gives a brief review of the classical SIR. Section 3 introduces its kernel extension in a reproducing kernel Hilbert space (RKHS) framework and provides some insight into the technical conditions. Theory that leads to the estimation of feature dimension reduction subspace is given. In the same section, a fast implementation algorithm is prescribed and some numerical issues are discussed. Section 4 is on numerical experiments and results. Concluding remarks are in Section 5. Some further theoretical properties for KSIR are in the Appendix.

- *Yi-Ren Yeh is with the Department of Computer Science & Information Engineering, National Taiwan University of Science and Technology, Taipei , Taiwan 10607.*
  E-mail: D9515009@mail.ntust.edu.tw
- *Su-Yun Huang is with the Institute of Statistical Science, Academia Sinica, Taipei, Taiwan 11529.*
  E-mail: syhuang@stat.sinica.edu.tw
- *Yuh-Jye Lee is with the Department of Computer Science & Information Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan 10607.*
  E-mail: yuh-jye@mail.ntust.edu.tw

Manuscript received 10 Mar. 2008; revised 2 Oct. 2008; accepted 17 Nov. 2008.

2 SLICED INVERSE REGRESSION

Different from other dimension reduction methods, SIR summarizes a regression or classification model as follows:

$$\mathbf{y} = f(\beta_1'\mathbf{x}, \ldots, \beta_d'\mathbf{x}; \epsilon), \quad \beta_k, \mathbf{x} \in \mathbb{R}^p, \tag{1}$$

where d (often $\ll p$) is the effective dimensionality and $\{\beta_1, \ldots, \beta_d\}$ forms a basis of the effective dimension reduction subspace. The model (1) implies that most of the relevant information in $\mathbf{x}$ about $\mathbf{y}$ is contained in $\{\beta_1'\mathbf{x}, \ldots, \beta_d'\mathbf{x}\}$. The dimensionality of input attributes gets cut down from p to d. It takes the weakest form for linear dimension reduction and only assumes the existence of some low-dimensional linear subspace without imposing any parametric structure on f. With this model (1) and the linear design condition (LDC) defined below, SIR then extracts the pattern *e.d.r.* subspace by using the notion of inverse regression [3], [5]. The central inverse regression function is defined as follows:

$$\mathbf{g}(\mathbf{y}) = E(\mathbf{x}|\mathbf{y}) - E(\mathbf{x}) \in \mathbb{R}^p.$$

We say that $\{\beta_1, \ldots, \beta_d\}$ satisfies the LDC, if, for any $b \in \mathbb{R}^p$, the conditional expectation $E(b'\mathbf{x}|\beta_1'\mathbf{x}, \ldots, \beta_d'\mathbf{x})$ is affine linear in $\{\beta_1'\mathbf{x}, \ldots, \beta_d'\mathbf{x}\}$. That is, there exist some constants $c_0, c_1, \ldots, c_d$ such that

$$E(b'\mathbf{x}|(\beta_1'\mathbf{x}, \ldots, \beta_d'\mathbf{x})) = c_0 + c_1\beta_1'\mathbf{x} + \cdots + c_d\beta_d'\mathbf{x}. \tag{2}$$

With the model (1) and the LDC (2), it can be shown [3] that the basis of the pattern *e.d.r.* subspace can be estimated by the leading directions from the generalized eigenvalue decomposition of $Cov(\mathbf{g})$, denoted by $\Sigma_{E(\mathbf{x}|\mathbf{y})}$, with respect to $Cov(\mathbf{x})$, denoted by $\Sigma_{\mathbf{x}}$. SIR is devised to find such leading directions. In other words, with $\mathbf{x}$ being standardized, SIR finds the leading directions in that the central inverse regression function $\mathbf{g}(\mathbf{y})$ has the largest variation. These are the most informative directions in the input pattern space for describing $\mathbf{y}$. In practical supervised learning tasks, the joint distribution of the input vector $\mathbf{x}$ and the output variable $\mathbf{y}$ is unknown but fixed. We use bold-faced $\mathbf{x}$ and $\mathbf{y}$ for random vectors and variables and italic letters x and y for their realizations. Suppose we have a data set

$$\mathcal{D} := \{(x^1, y_1), \ldots, (x^n, y_n)\},$$

each pair (x^i, y_i) is an instance $x^i \in \mathbb{R}^p$ with its response or class label y_i. Let $A \in \mathbb{R}^{n\times p}$ be the data matrix of input attributes and $Y = (y_1, \ldots, y_n)' \in \mathbb{R}^n$ be the corresponding responses. Each row of A represents an observation, x^i. The empirical data version of sliced inverse regression finds the dimension reduction directions by solving the following generalized eigenvalue problem based on empirical data $\mathcal{D}$:

$$\Sigma_{E(A|Y_J)}\beta = \lambda\Sigma_A\beta, \tag{3}$$

where Σ_A is the sample covariance matrix of A, Y_J denotes the membership in slices and there are J many slices, and $\Sigma_{E(A|Y_J)}$ denotes the between-slice sample covariance matrix based on sliced means given by

$$\Sigma_{E(A|Y_J)} = \frac{1}{n}\sum_{j=1}^{J} n_j(\bar{x}^j - \bar{x})(\bar{x}^j - \bar{x})'.$$

Here $\bar{x}$ is the sample grand mean, $\bar{x}^j = \frac{1}{n_j}\sum_{i\in S_j} x^i$ is the sample mean for the jth slice and S_j is the index set for jth slice. Note that the slices are extracted from A according to the sorted responses Y. For classification, $\bar{x}^j$ is simply the sample mean of input attributes for the jth class. There is an equivalent way to modeling SIR by the following optimization problem

$$\max_{\beta\in\mathbb{R}^p} \ \beta'\Sigma_{E(A|Y_J)}\beta \ \text{ subject to } \beta'\Sigma_A\beta = 1. \tag{4}$$

The solution, denoted by β_1, gives the first *e.d.r.* direction such that slice means projected along β_1 are most spreading out, where β_1 is normalized with respect to the sample covariance matrix Σ_A. Repeatedly solving this optimization problem with the orthogonality constraints $\beta_k\Sigma_A\beta_l = \delta_{k,l}$, where $\delta_{k,l}$ is the Kronecker delta, the sequence of solutions $\beta_1, \ldots, \beta_d$ forms the *e.d.r.* basis. Some insightful discussion to enhance the SIR methodology and applications can be found in Chen and Li [4].

3 KERNEL EXTENSION OF SIR IN RKHS AND A FAST IMPLEMENTATION

In this section, we first briefly review the kernel extension by Wu [8], which is formulated in a kernel-spectrum based feature space. We reformulate Wu's KSIR in an equivalent reproducing kernel Hilbert space (RKHS) setting. This reformulation is more intuitive for model explanation and is better suitable for practical implementation and theoretical development.

3.1 KSIR in RKHS: a geometric framework and properties

The classical SIR is designed to find a *linear* transformation from the input space to a low dimensional *e.d.r.* subspace that keeps as much information as possible for the output variable $\mathbf{y}$. However, it does not work for nonlinear feature extraction and it fails to find linear directions being in the null space or having small angles to the null space of $\Sigma_{E(\mathbf{x}|\mathbf{y})}$. The following regression example taken from Friedman [10] is used for illustrative purpose for KSIR. This example has explanatory variables in $\mathbb{R}^{10}$:

$$\begin{aligned} \mathbf{y} &= f(\mathbf{x}_1, \ldots, \mathbf{x}_{10}; \epsilon) \\ &= 10\sin(\pi\mathbf{x}_1\mathbf{x}_2) + 20(\mathbf{x}_3 - 0.5)^2 + 10\mathbf{x}_4 + 5\mathbf{x}_5 + \epsilon \end{aligned} \tag{5}$$

where $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_{10})$ are independent and identically distributed (iid) uniform random variables over $[0, 1]$ and $\epsilon \overset{iid}{\sim} N(0, 1)$. SIR fails to find the direction along the $\mathbf{x}_3$-coordinate due to its symmetric structure to the vertical axis at $\mathbf{x}_3 = 0.5$. The variance of $E(\mathbf{x}_3|\mathbf{y})$ is zero

and hence scaled principal components based on $\Sigma_{E(\mathbf{x}|\mathbf{y})}$ will not find the direction of $\mathbf{x}_3$-axis. For the Friedman example, the function's key features can be easily described by a few nonlinear components extracted by KSIR. Experimental study of it can be found in a later section.

In SIR, the model assumption (1) says that there exists a linear dimension reduction subspace and that the underlying objective function f can be any linear or nonlinear form defined on that subspace. To look for a nonlinear extension, in Wu's [8] setting the original input space $\mathcal{X}$ is mapped to a high dimensional feature space $\mathcal{Z}$ via the feature map

$$\Phi : \mathcal{X} \subset \mathbb{R}^p \mapsto \mathcal{Z}, \tag{6}$$

where $\Phi(x)$ is the kernel spectrum for a certain positive definite kernel, i.e., $K(x,u) = \Phi(x)'\Phi(u)$. The following dimension reduction model in the feature space $\mathcal{Z}$ is assumed

$$\mathbf{y} = f(\beta_1'\mathbf{z}, \ldots, \beta_d'\mathbf{z}; \epsilon), \quad \beta_k, \mathbf{z} \in \mathcal{Z}. \tag{7}$$

In other words, there exist $\beta_1, \ldots, \beta_d \in \mathcal{Z}$ such that $\mathbf{y}$ and $\mathbf{z}$ are conditionally independent given $\{\beta_1'\mathbf{z}, \ldots, \beta_d'\mathbf{z}\}$. See Wu [8] for the kernel extension of SIR in the framework of $\mathcal{Z}$. As the feature space $\mathcal{Z}$ is often not explicitly known to us, we will then look for a substitute with explicit expression. As part of the key purposes of dimension reduction are feature extraction and data visualization, a concrete feature space is more convenient for explanatory purpose and for practical implementation as well as theoretical development. We, therefore, transform the feature space $\mathcal{Z}$ to an isometric isomorphic space, which is explicitly known and where data can be observed. Consider an alternative feature map $\Gamma : \mathcal{X} \mapsto \mathcal{H}_K$ given by

$$x \mapsto \Gamma(x) := K(x, \cdot). \tag{8}$$

Kernels used here are positive definite, also known as reproducing kernels. For a given positive definite kernel K, its associated Hilbert space consists of all finite kernel mixtures $\sum_{i=1}^m a_i K(x, u_i)$ and their limits, where $m \in \mathbb{N}$, $u_i \in \mathbb{R}^p$ and $a_i \in \mathbb{R}$ all can be arbitrary. This Hilbert space is known as the reproducing kernel Hilbert space generated by K, denoted by $\mathcal{H}_K$. Throughout this article we assume that all the reproducing kernels employed are (C1) symmetric (i.e., $K(x,u) = K(u,x)$) and measurable, (C2) of trace type, i.e., $\int_{\mathcal{X}} K(x,x)d\mu < \infty$ and (C3) for $x \neq u$, $K(x,\cdot) \neq K(u,\cdot)$ in $L_2(\mathcal{X}, \mu)$ sense for Lebesgue measure μ. The distribution μ does not need be the same as the distribution of $\mathbf{x}$. The original input space $\mathcal{X}$ is then embedded into a new feature space $\mathcal{H}_K$ via the transformation Γ. Each input point $x \in \mathcal{X}$ is mapped to an element $K(x,\cdot) \in \mathcal{H}_K$.

Let $\mathcal{J} : \mathcal{Z} \mapsto \mathcal{H}_K$ be a map from the spectrum-based feature space $\mathcal{Z}$ to the kernel associated Hilbert space $\mathcal{H}_K$ defined by $\mathcal{J}(\Phi(x)) = K(x,\cdot)$. By condition (C3) and the reproducing property

$$\langle K(x,\cdot), f(\cdot)\rangle_{\mathcal{H}_K} = f(x), \quad \forall f \in \mathcal{H}_K, \forall x \in \mathcal{X},$$

it is easy to verify that $\mathcal{J}$ is a one-to-one linear transformation satisfying

$$\|z\|_{\mathcal{Z}}^2 = \|\Phi(x)\|_{\mathcal{Z}}^2 = K(x,x) = \|K(x,\cdot)\|_{\mathcal{H}_K}^2 = \|\mathcal{J}(z)\|_2^2.$$

Thus, $\Phi(\mathcal{X})$ and $\Gamma(\mathcal{X})$ are isometrically isomorphic, and the two feature representations (6) and (8) are equivalent in this sense. We will work directly on the latter feature representation (8), which is explicit and concrete. The classical SIR solves a generalized spectrum decomposition of the between-slice covariance matrix in the pattern Euclidean space $\mathbb{R}^p$. Similarly, KSIR solves a generalized spectrum decomposition of the between-slice covariance operator in the feature RKHS $\mathcal{H}_K$. The following two definitions place the notions of feature *e.d.r.* subspace and LDC in the framework of $\mathcal{H}_K$.

Definition 1 (feature e.d.r. subspace in $\mathcal{H}_K$): Let $H = \{h_1, \ldots, h_d\}$ be a collection of elements in $\mathcal{H}_K$, and let $\mathcal{H}$ be the linear subspace spanned by elements in H. We say that $\mathcal{H}$ is a feature *e.d.r.* subspace of $\mathcal{H}_K$ if $\mathbf{y}$ and $\mathbf{x}$ are conditionally independent given $\{h_1(\mathbf{x}), \ldots, h_d(\mathbf{x})\}$, i.e., information about $\mathbf{y}$ in $\mathbf{x}$ is contained in $\{h_1(\mathbf{x}), \ldots, h_d(\mathbf{x})\}$. We name h_k's the feature *e.d.r.* directions, $\mathcal{H}$ the feature *e.d.r.* subspace and $h_k(\mathbf{x})$'s the feature *e.d.r.* variates.

One can picture $h_k \in \mathcal{H}_K$ as the image of $\beta_k \in \mathcal{Z}$ via $\mathcal{J}$ and β_k as the pre-image of h_k. This gives an interplay between the *e.d.r.* directions in $\mathcal{Z}$ and in $\mathcal{H}_K$. Note that, by the isomorphism and the kernel reproducing property, we have

$$\beta_k'\mathbf{z} = \langle h_k(\cdot), K(\mathbf{x},\cdot)\rangle_{\mathcal{H}_K} = h_k(\mathbf{x}).$$

Then the *e.d.r.* model (7) becomes

$$\mathbf{y} = f(h_1(\mathbf{x}), \cdots, h_d(\mathbf{x}); \epsilon),$$

where $h_1(\mathbf{x}), \ldots, h_d(\mathbf{x})$ are nonlinear functional variates of $\mathbf{x}$.

Let $\ell : \mathcal{H}_K \mapsto \mathbb{R}$ be an arbitrary linear functional. By Riesz representation Theorem [11], [12], there exists an $f \in \mathcal{H}_K$ so that

$$\ell(K(\mathbf{x},\cdot)) = \langle K(\mathbf{x},\cdot), f(\cdot)\rangle_{\mathcal{H}_K} = f(\mathbf{x}). \tag{9}$$

Conversely, any $f \in \mathcal{H}_K$ defines a linear functional through (9). Thus, the functional variate $f(\mathbf{x})$ can be viewed as a kernel extension of a linear variate $b'\mathbf{x}$. It leads to the functional version of the linear design condition for KSIR.

Definition 2 (linear design condition in $\mathcal{H}_K$): Let $H = \{h_1, \ldots, h_d\}$ be a collection of elements in $\mathcal{H}_K$. H is said to satisfy the linear design condition, if the following statement holds. For any $f \in \mathcal{H}_K$,

$$E(f(\mathbf{x})|h_1(\mathbf{x}), \ldots, h_d(\mathbf{x})) = c_0 + c_1 h_1(\mathbf{x}) + \cdots + c_d h_d(\mathbf{x}) \tag{10}$$

for some constants $c_0, c_1, \ldots, c_d$.

At the first glance, this LDC seems more restrictive than the one given by (2) for the classical SIR. Actually, it is not the case. Condition (10) says that the regression of f on h_k is affine linear where h_k's are kernel mixtures yet

to be estimated. As kernel functions are abundant and flexible building blocks, it can be shown that any smooth function can be well approximated by a kernel mixture [13], [14]. Therefore, condition (10) is not as restrictive as the LDC (2) in the Euclidean space. The philosophy is that, linearity in an Euclidean space is much stricter than linearity in an RKHS. The former is in terms of linear combinations of the standard basis $\{e_1, \ldots, e_p\}$ formed by coordinate axes, while the latter is in terms of linear combinations of kernels, which form a large body of functions of various shapes. An elliptically symmetric distribution of data scatter in the *e.d.r.* subspace will ensure the validity of LDC. Some mild departure from the elliptical symmetry in the *e.d.r.* subspace will not hurt the application of SIR [3], [7]. For KSIR, the kernel transform has mapped the data into a very high dimensional feature RKHS, and we look for low-dimensional projections therein. Low-dimensional projections from high-dimensional data are known to be able to improve the elliptical symmetry of data distribution [6], [15].

The goal of KSIR is to estimate the feature *e.d.r.* directions h_k's. Similar to the classical SIR, KSIR finds feature *e.d.r.* directions by solving a generalized spectrum decomposition problem.

Note that the feature data $K(x^i, \cdot)$ and the feature *e.d.r.* directions h_k's are in a high- or infinite-dimensional space $\mathcal{H}_K$. To store and compute the feature data in a computer, a finite basis set $\{K(\cdot, u_i) : u_i \in \mathcal{I}\}$ is used to represent (also to approximate) the feature data, where $\mathcal{I} \subset \mathcal{X}$ is a finite index set. We often take the following

$$\text{basis set}: \ \{K(\cdot, x^i)\}_{i=1}^n, \ \text{denoted by } \{K(\cdot, A)\}. \quad (11)$$

Then the feature data become $K = [K(x^i, x^j)]_{i,j=1}^n$, the usual kernel matrix. Though this matrix is symmetric, there is no distinction between rows and columns. But still it is convenient to regard each row as an observation and each column as a variable. There are occasions that the basis set is not the type (11), then there is a distinction between rows and columns. Other choices of basis sets are fine, too, as long as they represent a distribution similar to the distribution of training input attributes. It is the key idea of our approximation to KSIR in Section 3.3. For simplicity and better intuition, we will work on a finite basis approximation subspace in the main body of this article. The functional version is given in the Appendix for interested readers. With the representation by finite basis, the feature *e.d.r.* variates can be expressed as

$$h_k(\mathbf{x}) = K(\mathbf{x}, A)\alpha_k \ \text{ for some } \alpha_k = (\alpha_k^1, \ldots, \alpha_k^n)' \in \mathbb{R}^n,$$

while an arbitrary nonlinear variate $f(\mathbf{x})$ can be represented as

$$f(\mathbf{x}) = K(\mathbf{x}, A)a \ \text{ for some } a = (a_1, \ldots, a_n) \in \mathbb{R}^n.$$

Let $T = K(\mathbf{x}, A)'$ be the finite-basis representation of the random variate $f(\mathbf{x})$, which is a random column vector in $\mathbb{R}^n$. The LDC in Definition 2 can be restated as shown below using finite basis representation:

$$E(a'T|\alpha_1'T, \ldots, \alpha_d'T) = c_0 + c_1\alpha_1'T + \cdots + c_d\alpha_d'T, \ \ \forall a \in \mathbb{R}^n. \quad (12)$$

With the introduction of a finite basis set, the feature *e.d.r.* directions and the subspace can be described in terms of finite-dimensional vectors and vector spaces. By taking such basis set $K(\cdot, A)$, the KSIR procedure is simply the classical SIR with $T = K(\mathbf{x}, A)'$ and $\mathbf{y}$. Hence Theorem 3.1 of Li [3] leads to the following theorem.

Theorem 1: Assume the existence of a feature *e.d.r.* subspace $\mathcal{H} = \text{span}\{K(\mathbf{x}, A)\alpha_1, \ldots, K(\mathbf{x}, A)\alpha_d\}$ and that the LDC (12) holds. Then the central inverse regression vector falls into the subspace spanned by $\{\Sigma_T\alpha_1, \ldots, \Sigma_T\alpha_d\}$.

$$E(T|\mathbf{y}) - E(T) \in \text{span}\{\Sigma_T\alpha_1, \ldots, \Sigma_T\alpha_d\}, \quad (13)$$

where Σ_T is the covariance matrix of $T = K(\mathbf{x}, A)'$.

One may consider to use other kernel basis, e.g., say, $K(\cdot, \tilde{A})$ for some finite set $\tilde{A} \in \mathbb{R}^{\tilde{n} \times p}$. Then T becomes $K(\mathbf{x}, \tilde{A})'$. Theorem 1 should then be revised accordingly. Also Theorem 1 is stated in terms of a finite-dimensional approximation using the basis set $K(\cdot, A)$ for practical simplicity. A more general functional version of the theorem can be found in the Appendix. It formulates KSIR as a generalized spectrum decomposition of the between-slice covariance operator in an RKHS framework. From (13) we see that the central inverse regression of T on $\mathbf{y}$ degenerates at any directions orthogonal to $\text{span}\{\Sigma_T\alpha_1, \ldots, \Sigma_T\alpha_d\}$. Thus, the covariance matrix of $E(T|\mathbf{y}) - E(T)$ provides a way for estimating the feature *e.d.r.* subspace $\mathcal{H}$. In other words, we solve the following generalized eigenvalue problem:

$$\Sigma_{E(T|\mathbf{y})}\alpha = \lambda\Sigma_T\alpha \quad (14)$$

or equivalently successively solve the following optimization problem with orthonormality constraints

$$\max_{\alpha \in \mathbb{R}^n} \ \alpha'\Sigma_{E(T|\mathbf{y})}\alpha \ \text{ subject to } \alpha'\Sigma_T\alpha = 1. \quad (15)$$

After extracting the feature *e.d.r.* directions, we will map the feature data onto the feature *e.d.r.* subspace spanned by these directions. The nonlinear structure of data will be embedded in this feature *e.d.r.* subspace. Thus, a linear learning algorithm can be applied on this feature *e.d.r.* subspace directly. Note that equation (14) is only used for illustrative purpose. We will not solve this generalized eigenvalue problem directly. Instead, we implement KSIR in a different way for numerical consideration and fast computation. Implementation details are introduced in the following subsections.

3.2 Estimating KSIR directions via the reduced SVD

Since the rank of the between-slice covariance of kernel data, $E(T|\mathbf{y})$ in (14), is $(J-1)$, we do not need to solve the whole eigenvalue decomposition problem for this $n \times n$ matrix. Instead, we use the reduced singular value decomposition (SVD) technique to solve for leading $(J-1)$

components to save the computing time. Let π_j be the proportional size (or prior probability) of the jth slice. Consider the eigenvalue decomposition of $W'\Sigma_T^{-1}W$ as UDU', where W consists of centered and weighted slice means. Precisely, the jth column of W is given by

$$w_j = \sqrt{\pi_j}\left(E(T|\mathbf{y}_i, i \in S_j) - E(T)\right).$$

Since the eigenvector(s) associated with the zero eigenvalue has no information for $\mathbf{y}$, we may restrict U and D to eigenvectors with non-zero eigenvalues. For simplicity, assume there is only one zero eigenvalue. Then, U has size $J\times(J-1)$ and its columns are formed by eigenvectors and D is a $(J-1)\times(J-1)$ diagonal matrix with descending non-zero eigenvalues. From Theorem 1, we know the feature *e.d.r.* subspace can be estimated via the generalized eigenvalue problem (14). In the following proposition, we will give the orthonormal basis set, which forms the feature *e.d.r.* directions.

Proposition 2 (feature e.d.r. directions): The orthonormalized feature *e.d.r.* directions are given by columns of $\Sigma_T^{-1}WUD^{-\frac{1}{2}}$, where U and D are from the eigenvalue decomposition $W'\Sigma_T^{-1}W = UDU'$.

Proof:

In estimating the *e.d.r.* directions, we first transform the generalized eigenvalue problem (14) to the standard eigenvalue problem:

$$\Sigma_T{}^{-\frac{1}{2}}WW'\Sigma_T{}^{-\frac{1}{2}}z = \lambda z, \tag{16}$$

where $z = \Sigma_T{}^{\frac{1}{2}}\alpha$ and $WW' = \Sigma_{E(T|\mathbf{y})}$. Let $ZD^{\frac{1}{2}}U'$ be the SVD of $\Sigma_T^{-\frac{1}{2}}W$. We have $W'\Sigma_T^{-1}W = UDU'$ and $Z = \Sigma_T^{-\frac{1}{2}}WUD^{-\frac{1}{2}}$ are the solutions of (16). Thus, the *e.d.r.* directions V can be estimated as follows:

$$V = \Sigma_T{}^{-\frac{1}{2}}Z = \Sigma_T^{-1}WUD^{-\frac{1}{2}}. \tag{17}$$

The proof is completed. □

Proposition 2 provides a reduction of computing complexity in the eigenvalue decomposition by using the reduced singular value decomposition for solving only leading $(J-1)$ components, where $J \ll n$. It converts the larger eigenvalue problem for $\Sigma_T^{-1}\Sigma_{E(T|\mathbf{y})} \in \mathbb{R}^{n\times n}$ into a smaller one, $W'\Sigma_T^{-1}W \in \mathbb{R}^{J\times J}$. Note that if only a few leading directions are needed, we can use only leading k $(< J-1)$ columns from U and corresponding leading diagonal elements from D for further reduction.

Proposition 2 is a population version. In practical data analysis sample estimates based on the given data $\mathcal{D}$ are used to replace all the population-based quantities. The sample covariance matrices $\Sigma_{E(K|Y_J)}$ and Σ_K are used to replace the population covariance matrices $\Sigma_{E(T|\mathbf{y})}$ and Σ_T, respectively. Also the following sample estimates for the centered weighted slice means are used to replace their population versions:

$$w_j = \sqrt{n_j/n}\left(\frac{\mathbf{1}'_{n_j}K(A_{S_j}, A)}{n_j} - \frac{\mathbf{1}'_nK(A,A)}{n}\right)',$$

where $\mathbf{1}'_{n_j}K(A_{S_j}, A)/n_j$ and $\mathbf{1}'_nK(A,A)/n$ are respectively the jth slice sample mean and the grand mean of $K(A,A)$. Note that then $WW' = \Sigma_{E(K|Y_J)}$ is the between-slice sample covariance. The sample covariance matrix Σ_K is singular and is often having much lower effective rank than its size. This low-effective-rank phenomenon causes numerical instability and poor *e.d.r.* directions estimation. We will discuss this issue and its remedy in next subsection.

3.3 Low-rank approximation to KSIR matrix

In many real world applications, the effective rank of the covariance matrix of kernel data is very low. This causes the numerical instability and leads to inferior estimation of feature *e.d.r.* directions [16]. Adding a ridge-type regularization term, i.e., to add a small diagonal matrix εI to Σ_K, is a common way to solve the numerical instability. However, the ridge-type regularization encounters a heavy computing load (even not doable) when the full kernel dimension is large. An appropriate way to deal with this problem is to find a reduced-column approximation to K, denoted by $\tilde{K}$, so that $\tilde{K}$ has full column rank and its column space $\mathcal{C}(\tilde{K})$ provides a good approximation to $\mathcal{C}(K)$. This approximation will enhance the numerical stability without much information loss. Therefore, throughout this article we will adopt a reduced-column approximation instead of a ridge-type regularization for the singularity of within-slice covariance matrix. The reduced-column approximation will cut down the problem size of the generalized spectrum decomposition and will also speed up the computation.

Let $\tilde{P}$ be a projection matrix of size $n \times \tilde{n}$, which satisfies $\tilde{P}'\tilde{P} = I_{\tilde{n}}$. Given a reduced-column kernel data $\tilde{K} := K\tilde{P}$, the approximation of KSIR is to solve the following reduced generalized eigenvalue problem:

$$\Sigma_{E(\tilde{K}|Y_J)}\tilde{\alpha} = \lambda\Sigma_{\tilde{K}}\tilde{\alpha}, \tag{18}$$

which is of much smaller size, as $\tilde{n} \ll n$. With the use of reduced kernel $\tilde{K}$, the corresponding centered weighted slice means are given by $\tilde{W}_{\tilde{n}\times J}$ with the jth column

$$\tilde{w}_j = \sqrt{n_j/n}\left(\frac{\mathbf{1}'_{n_j}\tilde{K}_{S_j}}{n_j} - \frac{\mathbf{1}'_n\tilde{K}}{n}\right)',$$

where $\mathbf{1}'_{n_j}\tilde{K}_{S_j}/n_j$ and $\mathbf{1}'_n\tilde{K}/n$ are respectively the jth slice mean and the grand mean of $\tilde{K}$. We can also apply Proposition 2 to the reduced problem (18) and the resulting feature *e.d.r.* directions are given by $\tilde{V} = \Sigma_{\tilde{K}}^{-1}\tilde{W}\tilde{U}\tilde{D}^{-1/2}$, where $\tilde{U}$ and $\tilde{D}$ are the eigenvectors and eigenvalues for $\tilde{W}'\Sigma_{\tilde{K}}^{-1}\tilde{W}$. Here, $\Sigma_{\tilde{K}}^{-1}$ exists if a proper projection $\tilde{P}$ is used. The KSIR algorithm using a reduced kernel approximation is given in Algorithm 1. Two strategies for choosing a low-rank projection matrix $\tilde{P}$ are discussed below.

Reduced kernel approximation by optimal basis. The SVD gives the empirical optimal low-rank projection to get a reduced kernel. The SVD step aims to cut the number of kernel columns to its effective rank to avoid the numerical instability and to cope with the difficulty

Algorithm 1 Low-rank Projection KSIR Algorithm

Input: A reduced kernel matrix $\tilde{K}_{n\times\tilde{n}}$ and a J-sliced response (label) n-vector Y_J.
Output: KSIR directions $\tilde{V}_{\tilde{n}\times(J-1)}$ and associated eigenvalues $\tilde{d}_{(J-1)\times 1}$.
1. Compute the centered and weighted slice means $\tilde{W}_{\tilde{n}\times J}$;
2. Compute the covariance matrix $\Sigma_{\tilde{K}}$ of the reduced kernel;
3. Compute the eigenvalue decomposition of $\tilde{W}'\Sigma_{\tilde{K}}^{-1}\tilde{W}$ as $\tilde{U}\tilde{D}\tilde{U}'$;
   /* $O(J^3)$ for solving the eigenvalue problem */
   /* $\tilde{D}$ and $\tilde{U}$ consist of non-zero eigenvalues and associated eigenvectors */
   /* $O(\tilde{n}^3)$ for solving the linear system $\Sigma_{\tilde{K}}X = \tilde{W}$ to get $\Sigma_{\tilde{K}}^{-1}\tilde{W}$ */
4. $\tilde{V} \leftarrow \Sigma_{\tilde{K}}^{-1}\tilde{W}\tilde{U}\tilde{D}^{-\frac{1}{2}}$; $\tilde{d} \leftarrow \text{diagonal}\{\tilde{D}\}$.

encountered in *e.d.r.* directions estimation. Consider the SVD of the centered full kernel matrix:

$$\left(I_n - \frac{\mathbf{1}_n\mathbf{1}_n'}{n}\right)K = G\Lambda P',$$

where $G'G = I$, $P'P = I$ and Λ is a diagonal matrix with descending singular values. Often the diagonal elements decay to zero very fast [17] and we need only a small number of leading eigenvectors to approximate the centered K:

$$\left(I_n - \frac{\mathbf{1}_n\mathbf{1}_n'}{n}\right)K = G\Lambda P' \approx \tilde{G}\tilde{\Lambda}\tilde{P}',$$

where $\tilde{G}$ and $\tilde{P}$, both of the size $n \times \tilde{n}$, consist of $\tilde{n}$ leading columns of G and P, respectively, and $\tilde{\Lambda}$, of size $\tilde{n} \times \tilde{n}$, consists of leading diagonals of Λ. We will work on the reduced-column kernel matrix $\tilde{K} := K\tilde{P}$ for the KSIR procedure. Note that P can be obtained from the eigenvalue decomposition of $Cov(K)$:

$$Cov(K) := \Sigma_K = PSP' \approx \tilde{P}\tilde{S}\tilde{P}',$$

where $S = \Lambda^2$ and $\tilde{S} = \tilde{\Lambda}^2$. Also note that

$$Cov(\tilde{K}) := \Sigma_{\tilde{K}} = \frac{1}{n}\tilde{P}'K\left(I_n - \frac{\mathbf{1}_n\mathbf{1}_n'}{n}\right)K\tilde{P} = \tilde{P}'\Sigma_K\tilde{P} = \tilde{S},$$

which makes the inverse of $\Sigma_{\tilde{K}}$ readily there. In other words, $\Sigma_{\tilde{K}}^{-1}$ in KSIR algorithm's Step 3 can be obtained from resulting matrices in this SVD preprocessing step. The reduced-column matrix by leading singular vectors guarantees the linear independence among columns. However, this strategy only works for small to median sized kernel matrix, as for large kernel the computing cost for large eigenvalue problem is heavy and of complexity $O(n^3)$. Sometimes the large-scale data can go beyond the capacity of the memory size. This SVD step takes up most of the computing time in extracting KSIR directions. For massive data sets, it is not economic and sometimes can be computationally difficult or even impossible to compute the optimal basis from the full kernel. Thus, we provide another strategy to handle the large scale problem.

Reduced kernel approximation by random basis. For median to large sized data sets, we use the random subset reduced kernel to cut the kernel column size. In the random subset approach we choose $\tilde{P}$ as a column subset from I_n. In practice, it is not necessary to generate the full K nor to calculate the transformation $\tilde{K} := K\tilde{P}$. Instead we directly build up $\tilde{K}$ with selected columns only. The basic concept of random subset reduced kernel technique is to approximate the full kernel by the Nyström approximation:

$$K(A,A) \approx K(A,\tilde{A})K(\tilde{A},\tilde{A})^{-1}K(\tilde{A},A) = \tilde{K}K(\tilde{A},\tilde{A})^{-1}\tilde{K}', \tag{19}$$

where $\tilde{A}_{\tilde{n}\times p}$ is a random subset of A and $K(A,\tilde{A}) = \tilde{K}_{n\times\tilde{n}}$ is a reduced kernel consisting of partial columns of the full kernel. Note that

$$K(A,A)\alpha \approx \tilde{K}K(\tilde{A},\tilde{A})^{-1}\tilde{K}'\alpha = \tilde{K}\tilde{\alpha},$$

where $\tilde{\alpha} = K(\tilde{A},\tilde{A})^{-1}\tilde{K}'\alpha$ is an approximation to the full problem. It means we only use $\tilde{n}$ basis functions $\{K(\cdot,\tilde{A})\}$ for modeling functions in $\mathcal{H}_K$. See [17] for more technical details and statistical properties for the random subset approach. The resulting reduced kernel matrix $\tilde{K}$ has full column rank so that $\Sigma_{\tilde{K}}$ is well-conditioned. The singularity problem can be resolved and the computational cost can be further cut down. Our selection of reduced set is done by a stratified random sampling from the full training set A. From Algorithm 1, we can see the efficiency of reduced KSIR in extracting feature *e.d.r.* directions. The computational cost is one time of $O(\tilde{n}^3)$ for $\Sigma_{\tilde{K}}^{-1}$ and one time of $O(J^3)$ for the eigenvalue problem incurred, where $\tilde{n}$ ($\ll n$) is at most in the hundreds and J is at most in the tens in our later experiments.

4 Numerical Experiments

In this section, we will design numerical experiments to evaluate the information content about $\mathbf{y}$ contained in the feature *e.d.r.* subspace extracted by the proposed KSIR algorithm implemented with reduction techniques. We focus on four kinds of applications of KSIR, namely, data visualization, classification, regression, and multiresponse regression. All experimental examples have two main steps: extracting the feature *e.d.r.* subspace and running a linear learning algorithm such as Fisher linear discriminant analysis (FDA), linear support vector machine (SVM), regularized least squares (RLS), or linear support vector regression (SVR) on the feature *e.d.r.* variates. We evaluate the effectiveness of KSIR

combined with linear learning algorithms on five binary classification data sets, eight multi-class data sets and six regression data sets by comparing the results with the nonlinear SVM and nonlinear SVR using the benchmark algorithm LIBSVM [9]. All the experimental data sets are described in Tables 1 and 2. Note that we use the names `Kf_1000`, `CA_1000`, and `CA` instead of `Kin_fh_1000`, `Comp_Activ_1000`, and `Comp_Activ` respectively for convenience in regression data sets. The "R.S." columns record the proportion of random subset for reduced kernel approximation used in our experiments. A 10% reduced set is often enough for median sized data. For smaller sized data sets, one may increase the reduction ratio, while for larger data sets, one may decrease the reduction ratio. We hardly let the basis set size go beyond 400. The reduction ratio of basis set can also be treated as a parameter and determined through a tuning procedure. Or the basis set can grow by an incremental algorithm [18]. The `banana` and `splice` data sets are from [19]. The `tree` data set is taken from Image Processing Lab, University of Texas at Arlington[1]. The `adult` and the `web` data sets are both compiled by Platt [20]. The `medline`[2] is a text classification data set. One important characteristic of text classification data is the large number of variable dimensionality ($p \gg n$). Unlike the classical SIR, which works on $p \times p$ between-slice and total covariance matrices, our KSIR algorithm can easily handle this kind of data sets with $p \gg n$ by employing linear kernel and works on the $n \times n$ linear kernel data matrix without getting into the difficulty of large covariance matrices. We defer this special handling to Section 4.2. All the other data sets can be obtained from UCI Repository of machine learning data archive [21] and UCI Statlog collection. Note that we scale all the classification data sets in $[-1, 1]$ except for the `dna` and `medline` data sets. Except for classification and regression problems, we also explore the possible application of KSIR to multiresponse regression. All our codes are written in Matlab [22] and are available at `http://www.stat.sinica.edu.tw/syhuang/`.

TABLE 1
Description of classification data sets.

Data set	Classes	Training Size	Testing Size	Attributes.	R.S. (%)
`banana`	2	400	4900	2	10%
`tree`	2	700	11692	18	10%
`splice`	2	1000	2175	60	20%
`adult`	2	32561	16281	123	1%
`web`	2	49749	14951	300	1%
`Iris`	3	150	-	4	10%
`wine`	3	178	-	13	10%
`vehicle`	4	846	-	18	20%
`segment`	7	2310	-	19	10%
`dna`	3	2000	1186	180	10%
`satimage`	6	4435	2000	36	20%
`pendigits`	10	7494	3498	16	4%
`medline`	5	1250	1250	22095	100%

TABLE 2
Description of regression data sets.

Data set	Size	Attributes	R.S. (%)
`Housing`	506	13	15%
`Kf_1000`	1000	32	5%
`CA_1000`	1000	21	5%
`Kin_fh`	8129	32	5%
`CA`	8129	21	1%
`Friedman`	40768	10	1%

4.1 Data visualization

SIR and KSIR aim to extract a low dimensional feature *e.d.r.* subspace that contains the information about output variable $\mathbf{y}$ as much as possible. The former looks for such a subspace in the pattern Euclidean space, while the latter in the feature RKHS. Moreover, both SIR and KSIR algorithms rank the importance of *e.d.r.* directions by associated eigenvalues. Thus, we can use the first one or two directions to visualize the main data structure, which will be otherwise complex in high dimension. Here we show some data views in a 2-dimensional subspace obtained by PCA, SIR and KSIR, respectively. The first example is on `pendigits` data. Training data are used to extract the *e.d.r.* directions, and only 14 test points from each category are used to plot the low-dimensional views to avoid excessive ink. The results are shown in Figure 1. Figures 1(a) and 1(b) are 2D views via PCA and SIR. There are some obvious overlaps among these ten classes. Figure 1(c) is the 2D view along KSIR directions and Figure 1(d) zooms in a small crowded region for a better view. We can easily see that KSIR variates provide a much better discriminant power.

The next two examples are on regression data sets. The first one is the "`peaks`" function provided by Matlab. It creates a synthetic regression surface (without additive noise) using 2-dimensional inputs. The corresponding regression surface plot with 961 points is in Figure 2(a). In our study, we split this data set into 80% and 20% subsets for training and testing, respectively. Plots of 2D views by PCA, SIR and KSIR are in Figures 2(b)-(d). As the `peaks` data have only 2-dimensional inputs, there is not much sense to talk about dimension reduction in the 2D pattern space. The purpose of this data example is to show the relevant linear structure in feature subspace by KSIR as depicted in Figure 2(d). We can clearly see that the KSIR processing turns the nonlinear structure in pattern space into a prominent linear structure in kernel feature space. It provides an empirical justification to combine the KSIR with linear learning algorithms for other tasks on the feature *e.d.r.* subspace.

Another regression example is the Friedman's example (5). There are ten explanatory variables, but only five of them are effective and the rest are nuisance. There are linear component, $10\mathbf{x_4} + 5\mathbf{x_5}$, as well as nonlinear component, $\sin(\pi \mathbf{x_1} \mathbf{x_2})$, and a hidden *e.d.r.* direction $\mathbf{x_3}$ to SIR in this example. We split the data into 99% and

1. http://www-ee.uta.edu/EEweb/IP/training_data_files.htm
2. http://www.cc.gatech.edu/∼hpark/data.html

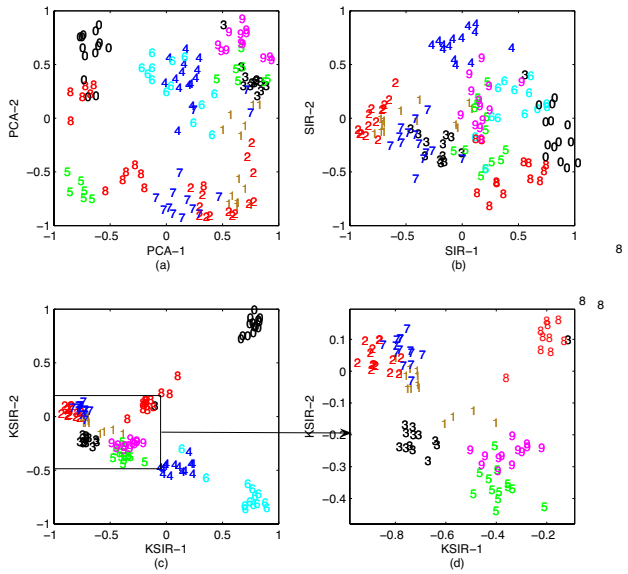

Fig. 1. 2D views of `pendigits` data by PCA, SIR, KSIR.

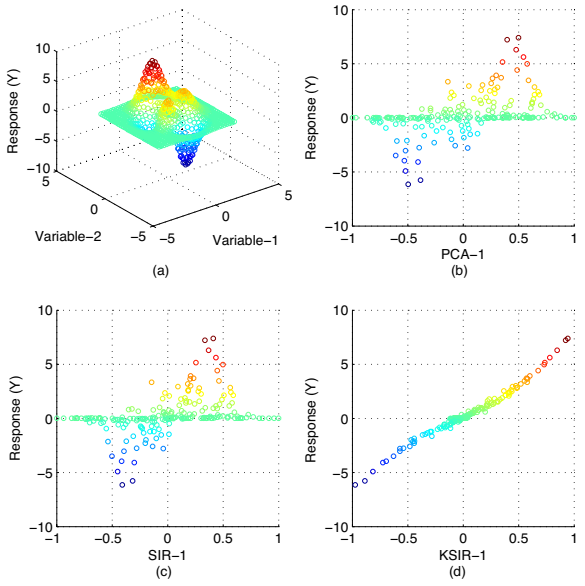

Fig. 2. 2D views of response vs. the 1st variate by PCA, SIR and KSIR with `peaks` data.

1% subsets for training and testing, respectively, for data visualization purpose. The leading ten eigenvalues by SIR and KSIR are respectively

$$\begin{aligned}\text{SIR}: &\ 0.7213, 0.0067, 0.0020, 0.0013, 0.0011,\\ &\ 0.0007, 0.0004, 0.0004, 0.0003, 0.0001;\\ \text{KSIR}: &\ 0.9417, 0.6639, 0.2010, 0.0435, 0.0214,\\ &\ 0.0120, 0.0111, 0.0105, 0.0097, 0.0092.\end{aligned}$$

The low-dimensional data views by PCA, SIR and KSIR

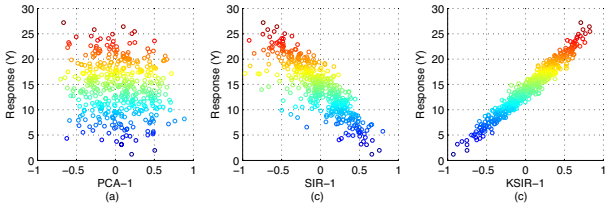

Fig. 3. 2D views of Friedman data by PCA, SIR and KSIR.

are shown in Figure 3. Obviously, none of the PCA directions capture a good effective subspace for the response surface. The first SIR direction does reflect a good description for the response, and the first KSIR direction is even better and it carries the best information content for the response among the three methods. Figure 3(c) shows a clearly good linearity of the first KSIR variate to the response. In summary, the effect of KSIR is not mere dimension reduction, it also maps the data to low-dimensional nonlinear features via kernel transformation so that the response can be well approximated by a linear form in terms of these extracted nonlinear features.

4.2 KSIR dimension reduction for classification

The dimension reduction provided by KSIR can be used as a data preprocess for later tasks such as classification or regression. In the process of applying SIR or KSIR, the role of each slice represents the clustering structure of the data. If we know the information about clustering structure in advance, it helps us to make slices when applying SIR or KSIR. In classification, the clustering structure has been defined through their class labels, and slices are made accordingly. We then estimate the central feature *e.d.r.* subspace and map the data onto this feature subspace for discriminant purpose. In a J-class problem, we slice the data sets into J slices according to the class labels. Thus, there are at most $J-1$ many independent *e.d.r.* directions, since the rank of $\Sigma_{E(A|Y_J)}$ or $\Sigma_{E(K|Y_J)}$ is at most $J-1$. After extracting the feature *e.d.r.* subspace, discriminant analysis becomes much computationally easier in this very low-dimensional subspace. Since we have turned the nonlinear structure in the pattern space into an *approximately linear* structure in the feature space via kernel transformation, direct application of linear learning algorithms on KSIR variates is often sufficient. In our classification experiments, we particularly pick the Fisher linear discriminant analysis (FDA) and the linear smooth support vector machine (SSVM) [23] as our baseline linear learning algorithms. One property of SSVM is that it is solved in the primal space and its computational complexity depends on the number of input attributes (here is the number of KSIR variates, which is at most $J-1$). A smaller number of columns implies less computational load. Note that as data are projected along the KSIR directions, discriminant analysis is computationally light. The FDA and SSVM acting on top of KSIR variates are numerically compared with

the standard nonlinear SVM benchmark algorithm LIBSVM.

For binary classification, the data sets put on experiments are already divided into training set and testing set in advance. We use the training set to build the model, including extracting the feature *e.d.r.* subspace and training for the final model in this *e.d.r.* subspace, then evaluate the resulting model using the testing set. As there is some stochastic variation due to reduced set selection, the procedure is repeated 10 times. We report the average error rate and the standard deviation of 10 times experiments. The upper panel of Table 3 lists the average error rates for these binary classifications. Reduced kernel approximation by a stratified random (training) subset is used depending on the data set size. The standard deviations reported in our numerical results are very small. It indicates that the reduced set sizes are large enough in our experiments from model robustness point of view. For binary classification, KSIR extracts a one-dimensional feature component for discriminant analysis. FDA in this one-dimensional subspace is a simple division of the line through the midpoint of two class centroids. The SSVM division of the line is a bit more complex than cutting through the midpoint. It is still a maximum margin criterion, but along a line instead of in a high-dimensional feature space. Results are compared with LIBSVM. Although FDA and SSVM are acting on top of a one-dimensional KSIR variate and reduced kernel approximation has been applied, the results in the KSIR columns are comparable to those in the LIBSVM column. It means that KSIR actually finds an effective projection direction in the kernel feature space and the reduced kernel approximation works well, too. For multi-class problems, the first four data sets are not pre-divided into training and testing sets. For these data sets, we use ten-fold cross validation and report their average error rates over 10 replicate runs of ten-fold partition. Results are listed in the middle panel of Table 3. For the rest multi-class data sets, there are separate testing sets. The results of ten replicate runs are reported in the lower panel of Table 3. In these multi-class classification problems, one-versus-one scheme is used for SSVM and LIBSVM. Note that for the `medline` data set, the variable dimensionality is 22095. For this example we look for linear *e.d.r.* directions in the original pattern space instead of kernel *e.d.r.* directions. As the dimensionality is so high that a direct application of the classical SIR, which calls for covariance matrices of size 22095, is not possible. Our KSIR algorithm can overcome this problem by using the linear kernel, $K = AA'$ (even the reduced linear kernel, $K = A\tilde{A}'$). In linear kernel setting, the computation cost is at most $O(n^3)$, where n is the sample size. This is the special handling for data sets with $n \ll p$, which is commonly seen in text mining and gene expression data sets. From Table 3, we see that a simple linear SSVM algorithm acting on a few leading KSIR variates can perform as good as LIBSVM. The performance of FDA on KSIR variates is a little worse than SSVM on KSIR variates and LIBSVM, but the difference is small.

TABLE 3
The error rate for FDA and linear SSVM on KSIR variates compared with LIBSVM on classification data sets.

Data set	KSIR+FDA mean (std)	KSIR+SSVM mean (std)	LIBSVM mean (std)
`banana`	0.1176 (0.0039)	0.1183 (0.0028)	0.1229
`tree`	0.1225 (0.0027)	0.1156 (0.0023)	0.1283
`splice`	0.1182 (0.0041)	0.1175 (0.0036)	0.1048
`adult`	0.1702 (0.0011)	0.1492 (0.0008)	0.1491
`web`	0.0164 (0.0007)	0.0152 (0.0004)	0.0090
`Iris`	0.0227 (0.0064)	0.0273 (0.0066)	0.0373 (0.0034)
`wine`	0.0163 (0.0053)	0.0125 (0.0042)	0.0187 (0.0042)
`vehicle`	0.1460 (0.0091)	0.1499 (0.0103)	0.1460 (0.0072)
`segment`	0.0297 (0.0025)	0.0282 (0.0020)	0.0285 (0.0016)
`dna`	0.0652 (0.0035)	0.0447 (0.0013)	0.0463
`satimage`	0.0926 (0.0031)	0.0914 (0.0034)	0.0870
`pendigits`	0.0229 (0.0011)	0.0193 (0.0027)	0.0180
`medline`	0.1208	0.1136	0.1106

Another issue is the computing time comparison. We record all the computing time CPU seconds in Table 4. All the experiments are executed in the same environment. The equipment of the computer is CPU P4 3.0GHz, Memory 1.5G and operating system XP. It can be seen that KSIR+FDA and KSIR+SSVM are often faster than LIBSVM in training time especially for large data sets and multi-class problems. For multi-class problems, we used "one-versus-one" scheme for SSVM and LIBSVM to decompose the problem into a series of binary classification subproblems and combine them by a majority vote. The KSIR+FDA and KSIR+SSVM only have to run the KSIR algorithm once, which consumes the major part of the computing time in one complete run of discriminant analysis. Once we have the *e.d.r.* variates with dimensionality $J-1$, a series of binary SSVMs or one FDA machine in this $(J-1)$-dimensional subspace is computationally light to carry out. In comparing the hybrid of KSIR with linear SSVM versus the LIBSVM, both have to build a series of $\binom{J}{2}$ many binary classifiers. The former needs a one-time-only KSIR process in a reduced feature subspace of dimensionality $\tilde{n}$ and then builds a series of binary classifiers in a $(J-1)$-dimensional KSIR extracted subspace, while the latter builds such a series of binary classifiers in a higher dimensional feature space of dimensionality about the size $2n/J$. We only report the training time comparison with LIBSVM. The testing time for linear learning algorithms (FDA and SSVM here, and RLS-SVR in next subsection) on KSIR test variates are prominently and uniformly faster than LIBSVM in regression and in classification, and thus we omit the report of testing time comparison. The efficient testing time is due to the fact that KSIR-based learning algorithms are acting on very low-dimensional KSIR variates, while LIBSVM is acting in the high dimensional feature space and its speed depends on the number of support vectors, which is much larger than the *e.d.r.* dimensionality.

TABLE 4
The training time (seconds) of FDA and linear SSVM on KSIR variates compared with LIBSVM on classification data sets.

Data set	KSIR+FDA	KSIR+SSVM	LIBSVM
banana	0.065	0.064	0.041
tree	0.076	0.078	0.075
splice	0.219	0.185	0.443
adult	4.863	4.867	219.3
web	16.18	16.37	149.5
dna	0.378	0.360	2.806
satimage	3.041	3.197	3.654
pendigits	1.245	1.775	2.402
medline	1.701	1.799	3.032

4.3 KSIR dimension reduction for regression

Different from classification problems, KSIR for regression can be more complicated than classification due to the lack of intuitive slices. In fact, we need to consider more factors, like the number of slices, their positioning and the dimensionality of the final *e.d.r.* subspace. For positioning of slices, we adopt a simple equal frequency strategy for it. For the number of slices, we run the experiments with various numbers of slices in the range of 15 to 45. We report the statistics such as mean, standard deviation and median, etc. of these experiments in Table 5. These results are quite robust to the number of slices.[3] Similar conclusion for SIR can be found in [3]. Thus, we fix at 30 slices in the rest of regression examples. In determining the number of extracted components, there are formal statistical tests [3], [24], [25], [26], [27] for the linear case. Extension of these formal statistical tests to the kernel case has to be carefully studied and justified, which can be an interesting and important future direction. Here we present a bootstrap-based graphical tool which combines the variability plot [28] together with the scree plot [29] for further suggestion of the dimensionality d. The scree plot gives a pictorial view of the eigenvalues and how they drop. In our empirical experience, the last few eigenvalues are relatively much smaller than the leading ones. The variability plot further quantifies the variability, or the stability which is one minus the variability, for each eigen-component. The variability of the jth component is defined to be

$$\rho_j = \frac{1}{B}\sum_{k=1}^{B}\left(1-\cos(\theta_k^{(j)})\right),$$

where $\theta_k^{(j)}$ is the angle between the jth eigen-component and its bootstrapped estimate (repeated B times). Low variability indicates the corresponding eigenvector is stable and is regarded as a major component, while high variability indicates a minor component and can be regarded as noise. Here, we present the variability and scree plots for `Housing` and `Friedman` data sets (see Figure 4) with $B = 50$. These plots suggest using a small dimensionality. From this observation, we run the regression experiments on a small dimensionality (3 dimensions) as well as $J-1$ dimensions.

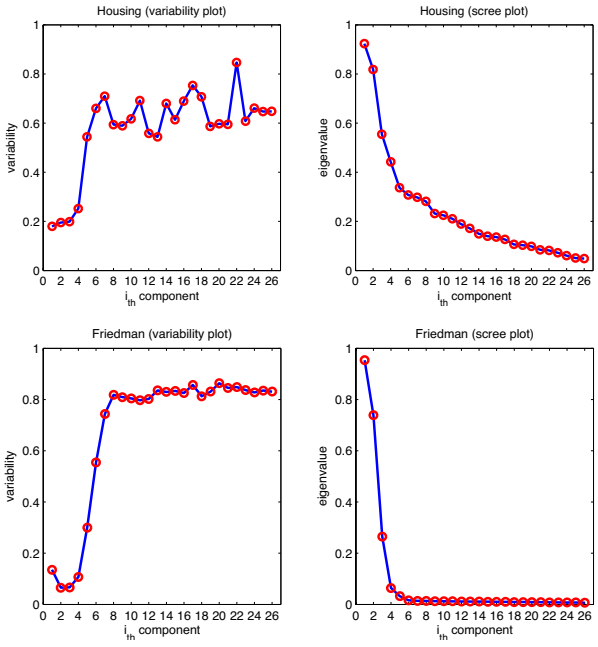

Fig. 4. The variability plot and scree plot for Housing and Friedman data sets

After extracting the KSIR directions, we apply the linear regularized least square (RLS) fit of the responses on KSIR variates. The testing results of KSIR+RLS are compared with nonlinear SVR provided in LIBSVM. Note that a reduced kernel approximation has been used in KSIR for all our regression examples. The R^2 values based on ten-fold cross validation are shown in Table 6. R^2 is a commonly used criterion for evaluation of regression goodness of fit. Its definition is given below:

$$R^2 = 1 - \sum_{i=1}^{n}(y_i-\hat{y}_i)^2 / \sum_{i=1}^{n}(y_i-\bar{y})^2,$$

where $\hat{y}_i$ is the fitted response and $\bar{y}$ is the grand mean. We can see that the difference in R^2 between KSIR(29)+RLS and LIBSVM is not significant. Note that LIBSVM is only a little better than KSIR(3)+RLS. In other words, a small dimensionality, such as 3, can still achieve roughly the same performance as using the full dimensionality of 29 for 30 slices. Often a few leading directions, say 3, can be well enough for describing the response without losing much information. That fits the purpose if the data visualization is required. Computing time comparison is reported in Table 7. KSIR+RLS is much faster than LIBSVM, especially for large data problems. All these results reflect the same phenomenon found in classification, that KSIR can find effective feature *e.d.r.* directions to speed up the computation for regression fit with satisfactory results.

3. For the `CA_1000` and `CA`, the low R^2 occurs at the slice number 15. By increasing the slice number, the low R^2 phenomenon easily disappears.

TABLE 5
R^2 using 30 slices and statistics for R^2 over 15-45 slices.

Data set	R^2 for 30 slices	R^2 over 15-45 slices				
		mean	std	median	max	min
Housing	0.8582	0.8584	0.0019	0.8580	0.8648	0.8553
Kf_1000	0.6480	0.6478	0.0008	0.6481	0.6490	0.6462
CA_1000	0.9704	0.9553	0.0299	0.9699	0.9748	0.8609
Kin_fh	0.6955	0.6954	0.0003	0.6953	0.6960	0.6948
CA	0.9774	0.9719	0.0143	0.9776	0.9789	0.9127
Friedman	0.9555	0.9555	0.0001	0.9556	0.9556	0.9554

TABLE 6
R^2 of RLS on 3 and 29 KSIR variates compared with LIBSVM on regression data sets.

Data set	KSIR(3)+RLS mean (std)	KSIR(29)+RLS mean (std)	LIBSVM mean (std)
Housing	0.8619 (0.0398)	0.8611 (0.0440)	0.8680 (0.0314)
Kf_1000	0.6457 (0.0479)	0.6473 (0.0462)	0.6470 (0.0459)
CA_1000	0.9606 (0.0094)	0.9702 (0.0055)	0.9783 (0.0056)
Kin_fh	0.6949 (0.0129)	0.6950 (0.0128)	0.7004 (0.0131)
CA	0.9733 (0.0047)	0.9770 (0.0031)	0.9821 (0.0022)
Friedman	0.9553 (0.0006)	0.9554 (0.0006)	0.9561 (0.0005)

TABLE 7
The training time (seconds) of RLS on 3 and 29 KSIR variates compared with LIBSVM on regression data sets.

Data set	KSIR(3)+RLS	KSIR(29)+RLS	LIBSVM
Housing	0.021	0.035	0.244
Kf_1000	0.015	0.024	0.371
CA_1000	0.016	0.042	0.502
Kin_fh	1.029	1.121	19.57
CA	0.167	0.376	27.50
Friedman	6.064	6.433	2242.1

4.4 KSIR dimension reduction for multiresponse regression

For multiresponse regression problems, we generate a synthetic data set modified from the Friedman example. 10 response variables and 10 predictor variables are generated as follows. The predictor variables $\mathbf{x}_1, \ldots, \mathbf{x}_{10}$ are independent and identically distributed uniformly over $[0, 1]$, and the 10 response variables are generated from the following setting:

$$\begin{aligned}
\mathbf{y_1} &= 10\sin(\pi \mathbf{x}_1 \mathbf{x}_2) + 5(\mathbf{x}_3 - 0.5)^2 + \mathbf{x}_4 + \mathbf{x}_5 + \epsilon, \\
\mathbf{y_2} &= 10\sin(\pi \mathbf{x}_1 \mathbf{x}_2) + 20(\mathbf{x}_3 - 0.5)^2 + 10\mathbf{x}_4 + 5\mathbf{x}_5 + \epsilon, \\
\mathbf{y_3} &= 5\mathbf{y_1} + \epsilon, \\
\mathbf{y_4} &= \mathbf{y_1} + \mathbf{y_2} + \epsilon, \\
\mathbf{y_5} \ \ldots \ \mathbf{y_{10}} &\sim N(0, 0.1,), \\
\epsilon &\sim N(0, 1).
\end{aligned}$$

A random sample of size 2000 is generated. Though the responses are in a 10-dimensional space, the effective dimensionality is actually only 2. Here are the steps for handling such a multiresponse regression using KSIR.

- Make kernel data (full or reduced) using predictor variables, denoted by K. Denote the observed multiresponses by Y.
- Compute correlation matrix of K and Y, denoted by R_{cor}, which is of size $m \times q$, where m is the column size of K and q is the column size of Y. (One may use covariance matrix as well.) Extract the leading k right singular vectors (here $k = 2$ for the modified Friedman example) of R_{cor} and project Y onto these k leading directions, i.e., $R_{cor} = USV'$ (U and V are orthogonal matrices and S is diagonal) and let $\tilde{Y} = Y\tilde{V}$, where $\tilde{V}$ is the leading k columns of V.
- Often $k \ll \dim(Y)$, and we call $\tilde{Y}$ the most correlated variates with K. Make slices using these low-dimensional most correlated variates (grid slicing along these $\tilde{Y}$ variates). In other words, we have J^k slices for the data.
- Solve the generalized eigenvalue problem of between-slice covariance with respect to the kernel data covariance to extract KSIR variates.
- Perform regression of Y on KSIR variates for an all-responses regression, or perform regression of $\tilde{Y}$ on KSIR variates for only the most correlated response variates.

In our procedure, we use $Cor(K, Y)$ (or one could use $Cov(K, Y)$ alternatively) to look for the most correlated directions between K and Y. These leading most correlated variates are used to make slices, and then KSIR is carried out based on these slices. Next, regression is made on KSIR variates. To examine the performance of the KSIR applied to multiresponse regression, we run the regression of Y on KSIR variates using RLS and compare the results with a coordinate-wise nonlinear SVR using LIBSVM (one run for each response and there are 10 in total). Figure 5 displays the resulting R^2 via the ten-fold cross validation for various numbers of slices. R^2 for multiresponse regression is defined as

$$R^2 = 1 - \sum_{i=1}^{n}\sum_{j=1}^{q}(y_{ij} - \hat{y}_{ij})^2 / \sum_{i=1}^{n}\sum_{j=1}^{q}(y_{ij} - \bar{y}_j)^2.$$

The results evidence the good performance of KSIR based the most correlated response variates. Besides, the results from Figure 5 also reflect the robustness on the number of slices.

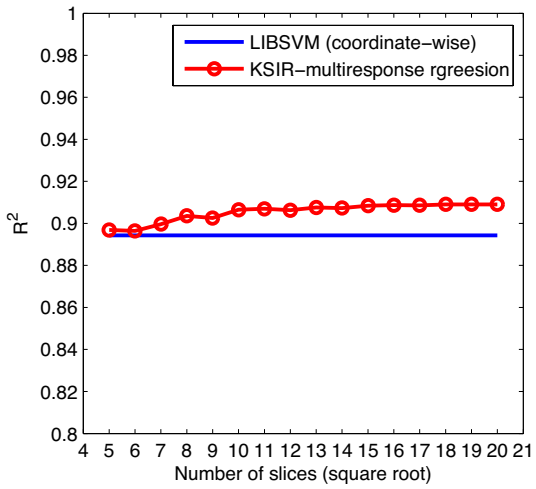

Fig. 5. R^2 for multiresponse modified Friedman example.

4.5 Kernels and parameter tuning

In nonparametric statistics, the choice of kernel type is often not crucial as long as the chosen kernel constitutes suitable building blocks for the underlying functional class. There will be some slight difference in estimation efficiency for different choice of kernels. A list of kernels for nonparametric estimation and their relative efficiencies can be found in [30] (Table 3.1). Gaussian kernel $K(x,u) = \exp(-\gamma\|x-u\|^2)$ is probably the most commonly used kernel in SVM-based learning algorithms and we adopt it for the kernel extension of SIR in our nonlinear numerical examples. On the other hand, the choice of kernel bandwidth is much more crucial than the kernel type. For either KSIR-based methods or for the conventional SVMs, there are two parameters involved, namely, the weight parameter C and the Gaussian kernel width parameter γ. The naive tuning procedure, a two-dimensional grid search, is time consuming. A remedy for it is to replace the bulldozer grid-search by a well-designed search scheme to reduce computing load. For instance, the nested uniform design (UD) model selection method [31] provides an economic alternative for parameter tuning. It has been numerically shown that there is no significant difference in testing accuracy by using a nested-UD search to replace the grid search. Thus, in our experimental study, the nested-UD is adopted for parameter tuning in LIBSVM. As for KSIR-based methods, computing time for parameter tuning is a lot lighter than that for conventional SVMs, even if the latter is equipped with a UD-based search. Tuning procedure for KSIR-based methods is *nearly* a one-dimensional search for γ, as the search for C is computationally light and negligible. KSIR-based methods are carried out in two stages. At the first stage, a parameter value for γ is needed for training KSIR *e.d.r.* subspace. At the second, a parameter value for C is needed for linear SSVM or RLS on KSIR variates. For each γ and its resulting *e.d.r.* subspace, an optimal C is determined over a range of grid points. This tuning procedure at the second stage for C is computationally light, as it is carried out in a very low-dimensional *e.d.r.* subspace and the time complexity of linear SSVM and RLS depend on the dimensionality of the *e.d.r.* subspace. For each fixed γ we can try a few C values to pair with this γ without much computing cost. This is another computing merit of KSIR approach in practical usage. Note that all the parameters C and γ of each data set in our experiments are reported in Table 8 and 9.

5 Conclusion

We have reformulated the KSIR [8] in a more rigorous and formal RKHS framework for theoretical development and for practical implementation of fast algorithm. The KSIR algorithm first maps the pattern space to an appropriate RKHS, and next extracts the main linear features in this embedded feature space. It takes class labels or regression response information into account and is a supervised dimension reduction method. After the extraction of the feature *e.d.r.* subspace, many supervised linear learning algorithms, such as FDA, SVM, SVR and possible others, can be applied to the images of input data in this feature *e.d.r.* subspace. This will generate a nonlinear learning model in the original input pattern space and achieve a very good performance for complex data analysis. We have also incorporated reduced kernel approximation to cut down the computational load and to resolve the numerical instability due to singularity in large between-slice covariance matrix. The singularity problem not only causes numerical instability but also leads to inferior *e.d.r.* directions estimation.

A few leading components extracted by KSIR can carry most of the relevant information about y in regression and in classification. It allows us to run linear learning algorithms in a very low dimensional feature *e.d.r.* subspace and to gain computational advantages without sacrificing the performance of learning algorithms. For example in solving nonlinear SVM multi-class problem, one has to solve $\binom{J}{2}$ nonlinear binary SVMs under the "one-versus-one" scheme. In KSIR-based approach, it only involves solving the KSIR problem once, and the remaining task is solved by a series of $\binom{J}{2}$ many linear binary SVMs in a $(J-1)$-dimensional space. Moreover, KSIR approach also has an advantage in tuning procedure. We have demonstrated these nice merits in our numerical experiments. Finally, using the first one or two components will help scientists or data analysts to gain a direct insight of data patterns, which will be otherwise complex in high dimensionality.

Appendix A
KSIR in an RKHS Framework

The classical SIR looks for *e.d.r.* directions in the pattern Euclidean space for maximum between-slice dispersion

TABLE 8
The parameters C and γ in each classification data set

Data set	KSIR+FDA	KSIR+SSVM		LIBSVM	
	γ	C	γ	C	γ
banana	8	3.16×10^4	5.76	1.00×10^2	4.19
tree	5.00×10^{-1}	1.00×10^3	1.34×10^{-1}	1	9.31×10^{-1}
splice	3.91×10^{-3}	1.78×10^5	1.61×10^{-3}	1.00×10^2	2.89×10^{-2}
adult	4.88×10^{-4}	5.62×10^3	4.49×10^{-4}	1.00×10^1	1.98×10^{-2}
web	5.00×10^{-1}	1.00×10^3	1.40×10^{-1}	1.00×10^1	3.76×10^{-2}
Iris	6.25×10^{-2}	1.00×10^3	1.98×10^{-1}	4.09×10^3	1.95×10^{-3}
wine	6.25×10^{-2}	1.00×10^1	1.94×10^{-2}	1.28×10^2	9.76×10^{-4}
vehicle	3.10×10^{-2}	3.16×10^4	3.10×10^{-2}	5.12×10^2	1.25×10^{-1}
segment	2.50×10^{-1}	5.62	4.78×10^{-1}	6.4×10^1	1
dna	9.76×10^{-4}	5.62×10^3	2.53×10^{-4}	1.00×10^2	2.53×10^{-2}
satimage	5.00×10^{-1}	5.62	3.20×10^{-1}	1.60×10^1	1
pendigits	6.25×10^{-2}	5.62	1.61×10^{-1}	1.00×10^1	4.65×10^{-1}
medline	-	1.00×10^{-1}	-	1.00×10^1	-

TABLE 9
The parameters C and γ in each regression data set

Data set	KSIR(3)+RLS		KSIR(29)+RLS		LIBSVM	
	C	γ	C	γ	C	γ
Housing	1.00×10^6	4.15×10^{-1}	1.00×10^6	4.15×10^{-1}	1.00×10^2	4.11×10^{-1}
Kf_1000	1.78×10^5	5.06×10^{-4}	5.62×10^3	5.06×10^{-4}	1	2.53×10^{-2}
CA_1000	1.00×10^3	2.00×10^{-3}	1.78×10^2	2.00×10^{-3}	3.16×10^2	1.15×10^{-1}
Kin_fh	3.16×10^1	9.90×10^{-3}	1	9.90×10^{-3}	1	2.66×10^{-2}
CA	1.79×10^2	2.80×10^{-3}	1.79×10^2	2.80×10^{-3}	1.00×10^2	1.15
Friedman	1.00×10^3	9.11×10^{-2}	1.00×10^3	9.11×10^{-2}	1.00×10^2	1.83×10^{-1}

with respect to overall dispersion. Based on the same idea of maximum between-slice dispersion, the kernel transform embeds the Euclidean pattern space $\mathcal{X}$ into an appropriate $\mathcal{H}_K$ by $\Gamma : \mathbf{x} \mapsto K(\mathbf{x}, \cdot)$. Next KSIR looks for *e.d.r.* directions in $\mathcal{H}_K$ that maximizes the between-slice dispersion with respect to the total dispersion. For technical details, we need to introduce a few more notations. Define below the grand mean function $m(\cdot)$, conditional mean function $m_{\mathbf{y}}(\cdot)$, covariance kernel Σ and between-slice covariance kernel Σ_B:

$$m(s) = E\{K(\mathbf{x}, s)\}, \tag{20}$$

$$m_{\mathbf{y}}(s) = E\left\{K(\mathbf{x}, s)|\mathbf{y}\right\}, \tag{21}$$

$$\Sigma(s,t) = E\left\{(K(\mathbf{x}, s) - m(s))(K(\mathbf{x}, t) - m(t))\right\}, \tag{22}$$

$$\Sigma_B(s,t) = E\left\{(m_{\mathbf{y}}(s) - m(s))(m_{\mathbf{y}}(t) - m(t))\right\}. \tag{23}$$

Σ and Σ_B are also called covariance operators, as they induce linear operators on $\mathcal{H}_K$ given by $(\Sigma f)(\cdot) = \int K(x, \cdot) f(x) dP_{\mathbf{x}}(x)$ and

$$(\Sigma_B f)(\cdot) = \int E\left(K(\mathbf{x}, \cdot)|\mathbf{y}\right) E\left(f(\mathbf{x})|\mathbf{y}\right) dP_{\mathbf{y}}(y).$$

KSIR solves the spectral decomposition of the between-slice covariance operator Σ_B with respect to the overall covariance operator Σ. That is, it aims to find leading directions $h_k \in \mathcal{H}_K$ to

$$\max\ \Sigma_B h_k = \lambda_k \Sigma h_k \ \text{ s.t. } \ \langle h_k, \Sigma h_k \rangle_{\mathcal{H}_K} = \delta_{kj}, \tag{24}$$

where $\delta_{kk} = 1$ and $\delta_{kj} = 0$ for $k \neq j$. Below is a functional version of Theorem 1.

Theorem 1* *Assume the existence of a feature e.d.r. subspace $\mathcal{H} = \text{span}\{h_1, \ldots, h_d\}$ in $\mathcal{H}_K$ and the validity of the LDC (10). Further assume that the covariance operator Σ is compact and non-singular and that $m_{\mathbf{y}} - m$ is in the range of $\Sigma^{1/2}$.* [4] *Then the central inverse regression curve $m_{\mathbf{y}} - m$ falls into the subspace $\Sigma(\mathcal{H})$.*

Proof of Theorem 1:* Note that for any $f, g \in \mathcal{H}_K$ we have $\langle m, f \rangle_{\mathcal{H}_K} = E\langle K(\mathbf{x}, \cdot), f(\cdot) \rangle_{\mathcal{H}_K} = Ef(\mathbf{x})$ and $\langle \Sigma f, g \rangle_{\mathcal{H}_K} = Cov\{f(\mathbf{x}), g(\mathbf{x})\}$. In particular, take $g = K(u, \cdot)$ and $f = h_k$, we have

$$\langle \Sigma h_k, g \rangle_{\mathcal{H}_K} = Cov\{h_k(\mathbf{x}), K(\mathbf{x}, u)\}. \tag{25}$$

Also the covariance of $h_i(\mathbf{x})$ and $h_j(\mathbf{x})$ is given by

$$Cov\{h_i(\mathbf{x}), h_j(\mathbf{x})\} = \langle \Sigma h_i, h_j \rangle_{\mathcal{H}_K}. \tag{26}$$

Let Σ_H denote the matrix with the (i,j)th entry given by (26) and let $H(\mathbf{x})$ be the random vector (column vector) given by $H(\mathbf{x}) = (h_1(\mathbf{x}), \ldots, h_d(\mathbf{x}))'$. Note that

$$m_{\mathbf{y}}(\cdot) - m(\cdot) = E\left\{E(K(\mathbf{x}, \cdot)|H(\mathbf{x})) - m(\cdot)\middle|\mathbf{y}\right\}.$$

By the LDC we have $E(K(\mathbf{x}, \cdot)|H(\mathbf{x})) - m(\cdot)$ is linear in $H(\mathbf{x})$, which leads to $E(K(\mathbf{x}, u)|H(\mathbf{x})) - m(u) = c(u)H(\mathbf{x})$, where $c(u) = Cov\{K(\mathbf{x}, u), H(\mathbf{x})'\}\Sigma_H^{-1}$ (a d-row vector) by the application of least squares linear regression. From (25) we have

$$c(u) = \langle \Sigma H(\cdot)', K(\cdot, u) \rangle_{\mathcal{H}_K} \Sigma_H^{-1} = (\Sigma H(u))' \Sigma_H^{-1}.$$

4. The compactness condition is to ensure that $\langle f, \Sigma g \rangle_{\mathcal{H}_K}$ is well-defined and the last condition on $m_{\mathbf{y}} - m$ is to ensure that $\langle m_{\mathbf{y}} - m, \Sigma^{-1}(m_{\mathbf{y}} - m) \rangle_{\mathcal{H}_K}$ exists and won't go unbounded.

$\Sigma H(u)$ is independent of $\mathbf{y}$ and is in the linear span of $\{\Sigma h_1(u), \ldots, \Sigma_K h_d(u)\}$. □

Theorem 1* says that $(m_{\mathbf{y}} - m)$ falls into the subspace $\Sigma(\mathcal{H})$. In other words, $\Sigma^{-1}(m_{\mathbf{y}} - m)$ is in the feature *e.d.r.* subspace. Thus, orthonormalized $\Sigma^{-1}(m_{\mathbf{y}} - m)$ can be used for the estimation of *e.d.r.* directions. The idea is first to slice the $\mathbf{y}$-variable into J slices and denote the slice means by m_j, i.e., $m_j(\cdot) = E\{K(\mathbf{x}, \cdot)|\mathbf{y} \in j\text{th slice}\}$. Next is to orthonormalize $\Sigma^{-1}(m_j - m)$, $j = 1, \ldots, J$, and use them for feature *e.d.r.* directions. The idea can be formalized in the following proposition. Let M be a $J \times J$ matrix with (j, j')th entry given by

$$M := \left[\sqrt{\pi_j \pi_{j'}}\, \langle m_j - m,\, \Sigma^{-1}(m_{j'} - m)\rangle_{\mathcal{H}_K}\right]_{jj'},$$

where π_j is the prior probability for the jth slice. Denote its decomposition by $M = UDU'$, where U consists of eigenvectors with non-zero eigenvalues and D is a diagonal matrix of non-zero eigenvalues. Let $W = (w_1, \ldots, w_J)$, J many functions arranged in row, where $w_j(u) = \sqrt{\pi_j}\,(m_j(u) - m(u))$ is the jth centered weighted slice mean.

Proposition 2* *$\Sigma^{-1}WUD^{-1/2}$ are e.d.r. directions.*

Proof of Proposition 2:* To show this proposition, we have to check that there exists a certain positive definite diagonal matrix Λ such that

$$\Sigma_B(\Sigma^{-1}WUD^{-1/2}) = \Sigma(\Sigma^{-1}WUD^{-1/2})\Lambda \quad \text{and}$$

$$(\Sigma^{-1}WUD^{-1/2})'\ \Sigma\ (\Sigma^{-1}WUD^{-1/2}) = I$$

hold. Note that Σ_B can be written as $\Sigma_B = WW'$. Then,

$$\Sigma_B(\Sigma^{-1}WUD^{-1/2}) = WW'\Sigma^{-1}WUD^{-1/2} = WUD^{1/2}.$$

That is, take $\Lambda = D$. The derivation of the second assertion straightforwardly goes as follows:

$$D^{-1/2}U'W'\Sigma^{-1}WUD^{-1/2} = D^{-1/2}U'MUD^{-1/2} = I.$$

The proof is completed. □

In summary, KSIR is derived based on the same notion as SIR but in an infinite-dimensional RKHS. Its procedure is simply the SIR procedure on kernel transformed data.

ACKNOWLEDGMENT

The authors thank Dr. Wu, Han-Ming for helpful discussion and thank four anonymous reviewers for many constructive comments. Special thanks go to an anonymous reviewer who has simplified our original proof for Proposition 2.

REFERENCES

[1] E. Alpaydm, *Introduction to Machine Learning*. The MIT Press, 2004.
[2] R. D. Cook, *Regression Graphics: Ideas for Studying Regressions Through Graphics*. John Wiley and Sons, 1998.
[3] K. C. Li, "Sliced inverse regression for dimension reduction (with discussion)," *Journal of the American Statistical Association*, vol. 86, pp. 316–342, 1991.
[4] C. Chen and K. C. Li, "Can SIR be as popular as multiple linear regression?" *Statistica Sinica*, vol. 8, pp. 289–316, 1998.
[5] N. Duan and K. C. Li, "Slicing regression: a link free regression method," *Annals of Statistics*, vol. 19, pp. 505–530, 1991.
[6] P. Hall and K. C. Li, "On almost linearity of low dimensional projection from high dimensional data," *Annals of Statistics*, vol. 21, pp. 867–889, 1993.
[7] K. C. Li, "Nonlinear confounding in high-dimensional regression," *Annals of Statistics*, vol. 25, pp. 577–612, 1997.
[8] H. M. Wu, "Kernel sliced inverse regression with applications on classification," *Journal of Computational and Graphical Statistics*, vol. 17, no. 3, pp. 590–610, 2008.
[9] C. C. Chang and C. J. Lin., "LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm," 2001.
[10] J. H. Friedman, "Multivariate adaptative regression splines," *Annals of Statistics*, vol. 19, pp. 1–67, 1991.
[11] G. Wahba, *Spline Models for Observational Data*. SIAM, 1990.
[12] G. Wahba, "Support vector machines, reproducing kernel Hillbert spaces, and randomized GACV," In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. The MIT Press, 1999.
[13] S. Saitoh, *Integral Transforms, Reproducing Kernels and Their Applications*. Addison Wesley Longman, 1997.
[14] J. R. Thompson and R. A. Tapia, *Nonparametric Function Estimation, Modeling, and Simulation*. SIAM, 1990.
[15] P. Diaconis and D. Freedman, "Asymptotics of graphical projection pursuit," *Annals of Statistics*, vol. 12, pp. 793–815, 1984.
[16] F. R. Bach and M. I. Jordan, "Kernel independent component analysis," *Journal of Machine Learning Research*, vol. 3, pp. 1–48, 2002.
[17] Y. J. Lee and S. Y. Huang, "Reduced support vector machines: a statistical theory," *IEEE Transactions on Neural Networks*, vol. 18, pp. 1–13, 2007.
[18] Y. J. Lee, H. Y. Lo, and S. Y. Huang, "Incremental reduced support vector machines," *International Conference on Informatics Cybernetics and System (ICICS2003)*, 2003.
[19] S. Mika, G. Rätsch, J. Weston, and B. Schölkpf and K.-R. Müller, "Fisher discriminant analysis with kernels," in *Neural Networks for Signal Processing IX*, 1999, pp. 41–48.
[20] J. C. Platt, "Sequential minimal optimization: a fast algorithm for training support vector machines," In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. The MIT Press, 1999.
[21] A. Asuncion and D. J. Newman, "UCI repository of machine learning databases. http://www.ics.uci.edu/∼mlearn/mlrepository.html," 2007.
[22] MATLAB, *User's Guide*. The MathWorks, Inc., Natick, MA 01760, 1992.
[23] Y. J. Lee and O. L. Mangasarian, "SSVM: A smooth support vector machine," *Computational Optimization and Applications*, vol. 20, pp. 5–22, 2001.
[24] L. Ferré, "Determining the dimension in sliced inverse regression and related methods," *Journal of the American Statistical Association*, vol. 93, pp. 132–140, 1998.
[25] J. R. Schott, "Determining the dimensionality in sliced inverse regression," *Journal of the American Statistical Association*, vol. 89, pp. 141–148, 1994.
[26] S. Velilla, "Assessing the number of linear components in a general regression problem," *Journal of the American Statistical Association*, vol. 93, pp. 1088–1098, 1998.
[27] Z. Ye and R. E. Weiss, "Using the bootstrap to select one of a new class of dimension reduction methods," *Journal of the American Statistical Association*, vol. 98, pp. 968–979, 1998.
[28] I. P. Tu, H. Chen, H. P. Wu, and X. Chen, "An eigenvector variability plot," *Statistica Sinica*, to appear.
[29] R. B. Cattell, "The scree test for the number of factors," *Multivariate Behavioral Research*, vol. 1, pp. 245–76, 1966.
[30] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.
[31] C. M. Huang, Y. J. Lee, D. K. J. Lin, and S. Y. Huang, "Model selection for support vector machines via uniform design," *A special issue on Machine Learning and Robust Data Mining of Computational Statistics and Data Analysis*, vol. 52, pp. 335–346, 2007.

Yi-Ren Yeh received the M.S. degree from the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taiwan in 2006. He is currently working toward the PhD degree in the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology. His research interests include machine learning, data mining, and information security.

Su-Yun Huang received the B.S. and M.S. degrees from Department of Mathematics, National Taiwan University, in 1983 and 1985, respectively, and the Ph.D. degree from Department of Statistics, Purdue University in 1990. She is currently a Research Fellow in the Institute of Statistical Science, Academia Sinica, Taiwan. Her research interests include mathematical statistics and statistical learning theory.

Yuh-Jye Lee received his Master degree in Applied Mathematics from the National Tsing Hua University, Taiwan in 1992 and PhD degree in computer sciences from the University of Wisconsin-Madison in 2001. He is an associate professor in the Dept. of Computer Science and Information Engineering, National Taiwan University of Science and Technology. His research interests are in machine learning, data mining, optimization, information security and operations research. Dr. Lee developed new algorithms for large data mining problems such as classification and regression problem, abnormal detection and dimension reduction. Using the methodologies such as support vector machines, reduced kernel method, chunking and smoothing techniques allow us to get a very robust solution (prediction) for a large dataset. These methods have been applied to solve many real world problems such as intrusion detection system (IDS), face detection, microarray gene expression analysis and breast cancer diagnosis and prognosis.