Supplemental materials for this article are available online
www.amstat.org/publications/JCGS

Kernel Sliced Inverse Regression with Applications to Classification

Han-Ming WU

Sliced inverse regression (SIR) was introduced by Li to find the effective dimension reduction directions for exploring the intrinsic structure of high-dimensional data. In this study, we propose a hybrid SIR method using a kernel machine which we call kernel SIR. The kernel mixtures result in the transformed data distribution being more Gaussian like and symmetric; providing more suitable conditions for performing SIR analysis. The proposed method can be regarded as a nonlinear extension of the SIR algorithm. We provide a theoretical description of the kernel SIR algorithm within the framework of reproducing kernel Hilbert space (RKHS). We also illustrate that kernel SIR performs better than several standard methods for discriminative, visualization, and regression purposes. We show how the features found with kernel SIR can be used for classification of microarray data and several other classification problems and compare the results with those obtained with several existing dimension reduction techniques. The results show that kernel SIR is a powerful nonlinear feature extractor for classification problems.

Key Words: Dimension reduction; Kernel machines; Reproducing kernel Hilbert space; Visualization.

1. INTRODUCTION

The sliced inverse regression (SIR) proposed by Li (1991) is a dimension reduction method that can be employed to find compact representations of data for exploring the intrinsic structure of high-dimensional observations. The effective dimension reduction (EDR) concept is introduced and estimated (see Section 2.1). SIR has been successfully extended and used in various applications (Chen and Li 1998; Cook and Yin 2001). One especially important application is the use of SIR as a tool for feature extraction in supervised learning for training classifiers and visualizing various aspects of the data structure (Bura and Pfeiffer 2003; Wu and Lu 2004). Unfortunately, the first moment-based SIR does not always function well in finding the entire EDR subspace. Also, as an EDR method designed for extracting prominent linear subspaces, it is unable to find important nonlinear

Han-Ming Wu is Assistant Professor, Department of Mathematics, Tamkang University, Taipei County 25137, Taiwan, R.O.C. (E-mail: *hmwu@mail.tku.edu.tw*).

Journal of Computational and Graphical Statistics, Volume 17, Number 3, Pages 590–610
DOI: 10.1198/106186008X345161

features. For example, SIR fails in the search for symmetric patterns about the responses variable y. These limitations motivate a nonlinear generalization of SIR. This generalization is the main topic of this article and we approach it using the so-called *kernel trick* (Aizerman, Braverman, and Rozonoer 1964; Vapnik 1995; Hastie, Tibshirani, and Friedman 2001; Schölkopf and Smola 2002).

The kernel trick presented by Aizerman et al. (1964) offers a nonlinear generalization of any algorithm that depends solely on the dot product of two vectors in a nonlinear fashion (Schölkopf, Smola, and Müller 1998). The underlying algorithm can be performed implicitly in the feature space associated with any kernel by replacing the dot product with a more general kernel. The nonlinear algorithm can be regarded as the underlying linear algorithm operating in the feature space. Its operation is analogous to the linear algorithm operating in the original input space. In this case, data are not represented individually anymore, but only through a set of pairwise comparisons, namely *kernel data*. Over the last decade, various kernelized algorithms such as kernel principal component analysis (KPCA) (Schölkopf et al. 1998, 1999), kernel Fisher's discriminant analysis (KFDA) (Mika, Rätsch, Weston, Schölkopf, and Müller 1999; Baudat and Anouar 2000; Roth and Steinhage 2000), kernel canonical correlation analysis (KCCA) (Lai and Fyfe 2000), and kernel partial least squares (KPLS) (Rosipal and Trejo 2001) have been presented in the machine learning and data mining literature (Bach and Jordan 2002; Schölkopf and Smola 2002).

Taking into account nonlinearities using the kernel trick is conducive to performing SIR for nonlinear dimension reduction and feature extraction. Kernel SIR (KSIR) provides a new tool which can be applied with little computational cost and possibly substantial performance gains. Using the first two or three KSIR variates, one can observe the structure of the set of points; such as the presence of clusters or outliers. The proposed approach can be used in the setting of supervised learning, in which a set of observations $\mathbf{x}$ and desired responses or class labels y are available. If the data come without the class labels y, we can apply a K-means clustering to $\mathbf{x}$ to produce generalized slices.

Among the many potential applications of KSIR, we focus on the classification problem. It is common to first extract features suitable for classification before applying the relevant classification algorithm. The KSIR approach extends the linear EDR subspace into nonlinear subspace by formulating it in a reproducing kernel Hilbert space (RKHS). KSIR involves the information contained by labels and kernel mixtures. It preserves the computational advantages of classical SIR. Our experimental results suggest that the directions found by KSIR are particularly suitable for discriminative purposes.

This article is structured as follows. Section 2 introduces the KSIR model and its properties with respect to RKHS. Section 3 discusses the connections between KSIR and other multivariate statistical techniques. Section 4 describes the behavior of the kernelized estimates of EDR directions and their nonlinear projection view for discriminative, visualization and regression purposes. Classification of microarray data and other real-world data using extracted features by principal component analysis (PCA), SIR, and their kernelized counterparts are reported in Section 5. We then conclude with our results in Section 6. A proof is given in the Appendix.

2. KERNEL SLICED INVERSE REGRESSION

2.1 SIR IN EUCLIDEAN SPACE

Li (1991) presented the sliced inverse regression (SIR) method as a prototypical framework for dimension reduction. The following regression model is considered

$$y = f(\boldsymbol{\beta}_1^t \mathbf{x}, \ldots, \boldsymbol{\beta}_B^t \mathbf{x}, \epsilon), \tag{2.1}$$

where y is an univariate random variable, $\mathbf{x}$ is a random vector with dimension $p \times 1$, $p \geq B$, $\boldsymbol{\beta}$'s are vectors with dimension $p \times 1$, ϵ is a random noise independent of $\mathbf{x}$, and f is an arbitrary function. The $\boldsymbol{\beta}$'s are referred to as effective dimension reduction (EDR) or projection directions. One can reduce the dimension of the independent variable $\mathbf{x}$ by projecting it along the EDR directions while maintaining most of the information conveyed in $\mathbf{x}$ about y. Sliced inverse regression is a method for estimating EDR directions based on y and $\mathbf{x}$. Under model assumption (2.1) and the linearity condition presented by Li (1991), it has been shown that the centered inverse regression curve $E(\mathbf{x}|y) - E(\mathbf{x})$ is contained in the linear subspace spanned by $\boldsymbol{\beta}_b^t \Sigma_{\mathbf{xx}}$ $(b = 1, \ldots, B)$, where $\Sigma_{\mathbf{xx}}$ denotes the covariance matrix of $\mathbf{x}$. Based on this property, estimates for $\boldsymbol{\beta}$'s can be obtained by standardizing $\mathbf{x}$, partitioning slices (or groups) according to the value of y, calculating the slice means of standardized $\mathbf{x}$'s, and performing the principal component analysis of the slice means with weights as sample proportion within each slice.

More precisely, SIR performs an eigenvalues decomposition of weighted covariance matrix $\Sigma_{E(\mathbf{x}|\tilde{y})}$ with respect to $\Sigma_{\mathbf{xx}}$, where $\tilde{y}$ is the discretized y formed by partitioning sorted y into slices $(I_h, h = 1, \ldots, H)$ and is constant within each slice. The weighted covariance matrix $\Sigma_{E(\mathbf{x}|\tilde{y})}$ is constructed in the following way

$$\Sigma_{E(\mathbf{x}|\tilde{y})} = \sum_{h=1}^{H} p_h (\hat{\mathbf{m}}_h - \bar{\mathbf{m}})(\hat{\mathbf{m}}_h - \bar{\mathbf{m}})^t, \tag{2.2}$$

where p_h is the proportion of all observed y_i's that fall into hth slice (I_h), that is,

$$p_h = \frac{\sum_{i=1}^{n} \delta_h(y_i)}{n} = \frac{n_h}{n}, \quad \delta_h(y_i) = 1, \quad \text{if} \quad y_i \in I_h, \ \delta_h(y_i) = 0, \ \text{otherwise},$$

and $\bar{\mathbf{m}}$ is the grand mean and $\hat{\mathbf{m}}_h$ is the sample mean for the hth slice . Then SIR solves the following eigen-problem

$$\Sigma_{E(\mathbf{x}|\tilde{y})} \boldsymbol{\beta}_j = \lambda_j \Sigma_{\mathbf{xx}} \boldsymbol{\beta}_j, \tag{2.3}$$

where $j = 1, \ldots, p$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$. Then, the leading B eigenvectors $\boldsymbol{\beta}_j$'s are used as projection directions. The reduced subspace considered by the classical SIR is a B-dimensional linear subspace spanned by $\{\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_B\}$. Cook (1994, 1996) named this reduced subspace the central dimension reduction subspace, $\mathcal{S}_{Y|X}$, for the regression of y on $\mathbf{x}$ in which $\{\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_B\}$ form a basis for $\mathcal{S}_{Y|X}$. However, in practice there are often cases where the data cloud of independent variables cannot be well characterized by projection into a low-dimensional linear subspace, but rather can be accurately approximated by a small number of nonlinear components. In the following we introduce a kernelized

version of the SIR algorithm, which extends the linear EDR into a nonlinear one using a RKHS.

2.2 KERNEL SIR IN A NONLINEAR FEATURE SPACE

RKHS (Girosi 1998; Schölkopf and Smola 2002) can be characterized in terms of kernels. Aronszajn (1950) linked kernels in terms of dot products using what is now known as the *kernel trick.*

Proposition 1. *(Kernel trick; Schölkopf, Tsuda, and Vert 2004, p. 44) Any algorithm for vectorial data that can be expressed only in terms of dot products between vectors can be performed implicitly in the feature space associated with any kernel by replacing each dot product by a kernel evaluation.*

For a given positive definite kernel κ and its spectrum

$$\kappa(\mathbf{x}, \mathbf{u}) = \sum_{q=1}^{d} \lambda_q \phi_q(\mathbf{x}) \phi_q(\mathbf{u}), \ d \leq \infty, \ \mathbf{x}, \mathbf{u} \in \mathcal{X}, \tag{2.4}$$

the main idea of KSIR is first to map the data in the input space $\mathcal{X} \subset \Re^p$ into the spectrum-based feature space $\mathcal{H}$ via the transformation Φ

$$\mathbf{x} \mapsto \mathbf{z} := \Phi(\mathbf{x}) := (\sqrt{\lambda}_1 \phi_1(\mathbf{x}), \sqrt{\lambda}_2 \phi_2(\mathbf{x}), \ldots)^t. \tag{2.5}$$

In practice, the kernel function κ is defined directly without explicit expression of its spectrum ϕ. The dot product in the feature space can be obtained via kernel value

$$\langle \Phi(\mathbf{x}), \Phi(\mathbf{u}) \rangle_{\mathcal{H}} = \sum_{q} \lambda_q \phi_q(\mathbf{x}) \phi_q(\mathbf{u}) = \kappa(\mathbf{x}, \mathbf{u}), \tag{2.6}$$

which allows us to work directly on the kernel values without explicitly knowing the spectrum-based transformation $\Phi : \mathcal{X} \to \mathcal{H}$.

Now consider the regression model analogous to model (2.1) in the feature space $\mathcal{H}$

$$y = f(\boldsymbol{\beta}_1^t \Phi(\mathbf{x}), \ldots, \boldsymbol{\beta}_B^t \Phi(\mathbf{x}), \epsilon) = f(\boldsymbol{\beta}_1^t \mathbf{z}, \ldots, \boldsymbol{\beta}_B^t \mathbf{z}, \epsilon), \text{ where } \boldsymbol{\beta}_b \in \Re^d, \ d \leq \infty. \tag{2.7}$$

Model (2.7) describes how the response variable y depends on variables $\mathbf{z}$ only through a low-dimensional subspace of $\mathcal{H}$ spanned by $\boldsymbol{\beta}_b^t \mathbf{z}$, $1 \leq b \leq B$. Assume the transformed data, $\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_n)$, in the feature space are centered, that is, $\sum_{i=1}^n \Phi(\mathbf{x}_i) = 0$ (if not, center them accordingly; see Section 2.3). Let $\Sigma_{\mathbf{zz}} = \frac{1}{n} \sum_{i=1}^n \Phi(\mathbf{x}_i) \Phi(\mathbf{x}_i)^t$ be the sample covariance matrix of $\mathbf{z} := \Phi(\mathbf{x})$. Let $\Sigma_{E(\mathbf{z}|\tilde{y})}$ be the sample between-slice covariance matrix given by

$$\Sigma_{E(\mathbf{z}|\tilde{y})} = \sum_{h=1}^{H} p_h \bar{\Phi}_h \bar{\Phi}_h^t,$$

where $\bar{\Phi}_h = \frac{1}{np_h} \sum_{i=1}^n \Phi(\mathbf{x}_i) \delta_h(y_i)$ denotes the sample mean of the hth slice. We must find eigenvalues $\lambda \geq 0$ and eigenvectors $\boldsymbol{v} \in \mathcal{H}$ satisfying

$$\Sigma_{E(\mathbf{z}|\tilde{y})} \boldsymbol{v} = \lambda \Sigma_{\mathbf{zz}} \boldsymbol{v}. \tag{2.8}$$

We can solve the problem by an equivalent system of equations

$$\langle \Phi(\mathbf{x}_i), \Sigma_{E(\mathbf{z}|\tilde{y})}\boldsymbol{v}\rangle_{\mathcal{H}} = \lambda\langle \Phi(\mathbf{x}_i), \Sigma_{\mathbf{zz}}\boldsymbol{v}\rangle_{\mathcal{H}}, \quad i = 1, \ldots, n. \tag{2.9}$$

The solution is of the form $\boldsymbol{v} = \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{x}_i)$ for some $\alpha_1, \ldots, \alpha_n$. Define the *kernel data* by $\mathbf{K} := \{\kappa_{ij} = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle_{\mathcal{H}}\}_{n\times n}$ whose ijth element is κ_{ij}. $\mathbf{K}$ is also called the *Gram matrix* of κ with respect to $\mathbf{x}_1, \ldots, \mathbf{x}_n$. Let $\mathcal{H}_\kappa$ be the reproducing kernel Hilbert space induced by κ. We can rephrase SIR in terms of kernel data by solving the following eigen-problem (see Appendix):

$$\mathbf{E}_H\mathbf{K}\boldsymbol{\alpha} = \lambda\mathbf{K}\boldsymbol{\alpha}, \tag{2.10}$$

where $\mathbf{E}_H = \sum_{h=1}^{H} n_h^{-1}\mathbf{1}_h\mathbf{1}_h^t$ with $\mathbf{1}_h = [\delta_h(y_1)\ldots\delta_h(y_n)]^t$, and $\boldsymbol{\alpha}$ is an n-vector whose jth element is the coefficient α_j.

Kernel SIR performs a spectrum decomposition of the weighted kernel matrix $\mathbf{E}_H\mathbf{K}$ with respect to the kernel matrix $\mathbf{K}$. Let $\lambda_1 \geq \cdots \geq \lambda_n$ denote the eigenvalues, and $\boldsymbol{\alpha}^1, \ldots, \boldsymbol{\alpha}^n$ be the corresponding complete set of eigenvectors. We assume $\boldsymbol{\alpha}^1, \ldots, \boldsymbol{\alpha}^n$ are normalized so that $\langle \boldsymbol{v}^k, \boldsymbol{v}^k\rangle_{\mathcal{H}} = 1$ for all $k = 1, \ldots, n$. Since each $\boldsymbol{v}^k$ is of the form $\boldsymbol{v}^k = \sum_{i=1}^{n} \alpha_i^k \Phi(\boldsymbol{x}_i)$, the normalization translates to

$$1 = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i^k\alpha_j^k\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle_{\mathcal{H}} = \langle \boldsymbol{\alpha}^k, \mathbf{K}\boldsymbol{\alpha}^k\rangle_{\Re^n} = \lambda_k\langle \boldsymbol{\alpha}^k, \boldsymbol{\alpha}^k\rangle_{\Re^n}, \quad k = 1, \ldots, n. \tag{2.11}$$

Let $\mathbf{x}$ be a test point, with an image $\Phi(\mathbf{x})$ in $\mathcal{H}$, then the projections of $\Phi(\mathbf{x})$ along the eigenvectors $\boldsymbol{\alpha}^k$, $k = 1, \ldots, n$, are given by

$$\langle \boldsymbol{\beta}_k, \Phi(\mathbf{x})\rangle_{\mathcal{H}} = \sum_{i=1}^{n} \alpha_i^k\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x})\rangle_{\mathcal{H}} = \sum_{i=1}^{n} \alpha_i^k\, \kappa(\mathbf{x}_i, \mathbf{x}). \tag{2.12}$$

Equations (2.10), (2.11), and (2.12) do not require the mapping of Φ in explicit form. Therefore, we are able to use kernel functions for computing these dot products without actually knowing the map Φ (Aizerman et al. 1964; Boser, Guyon, and Vapnik 1992). Also we will be interested only in the leading B eigen-components. Analogous to the steps in Li (1991), we set a condition and state a theorem.

Condition 1. *(Linear design condition) Assume for any $\boldsymbol{v} \in \mathcal{H}$, we have that $E(\boldsymbol{v}^t\Phi(\mathbf{x})|\boldsymbol{\beta}_1^t\Phi(\mathbf{x}), \ldots, \boldsymbol{\beta}_B^t\Phi(\mathbf{x}))$ is linear in $\boldsymbol{\beta}_1^t\Phi(\mathbf{x}), \ldots, \boldsymbol{\beta}_B^t\Phi(\mathbf{x})$, that is,*

$$E(\boldsymbol{v}^t\Phi(\mathbf{x})|\boldsymbol{\beta}_1^t\Phi(\mathbf{x}), \ldots, \boldsymbol{\beta}_B^t\Phi(\mathbf{x})) = c_0 + c_1\boldsymbol{\beta}_1^t\Phi(\mathbf{x}) + \cdots + c_B\boldsymbol{\beta}_B^t\Phi(\mathbf{x})$$

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

Theorem 1. *Under (2.7) and Condition 1, $E(\Phi(\mathbf{x})|\, y) - E(\Phi(\mathbf{x}))$ falls into the linear subspace spanned by $\boldsymbol{\beta}_b^t\Sigma_{\mathbf{zz}}$, $b = 1, \ldots, B$.*

Remark 1. When $\Phi(\mathbf{x})$ is elliptically symmetric, the above condition is fulfilled. However, the mapping Φ is unknown practically which results in the Condition 1 being difficult to verify. Nevertheless, one can evaluate the kernel matrix $\mathbf{K}$ instead since KSIR can be regarded as the classical SIR performed on the kernel matrix. When this condition is violated, the biases in estimating the projection directions are not large, as discussed by Li (1991).

Remark 2. By (2.12), model (2.7) becomes

$$y = f(\sum_{i=1}^{n} \alpha_i^1 \kappa(\mathbf{x}_i, \mathbf{x}), \ldots, \sum_{i=1}^{n} \alpha_i^B \kappa(\mathbf{x}_i, \mathbf{x}), \epsilon). \tag{2.13}$$

In other words, the regression surface supposedly can be characterized by the B components of kernel mixtures. Kernels are known to provide a class of versatile local basis for flexible function approximation. B components of kernel mixtures, with mixing coefficients $\boldsymbol{\alpha}$ adaptive to the data, form a rich body of smooth functions (Berlinet and Thomas-Agnan 2004). Let $g_b(\mathbf{x}) = \boldsymbol{\beta}_b^t \Phi(\mathbf{x}) = \sum_{i=1}^n \alpha_i^b \kappa(\mathbf{x}_i, \mathbf{x}) \in \mathcal{H}_\kappa$, $b = 1, \ldots, B$. The linear design condition means that for an arbitrary $h(\mathbf{x}) \in \mathcal{H}_\kappa$

$$E(h(\mathbf{x})|g_1(\mathbf{x}), \ldots, g_B(\mathbf{x})) = c_0 + c_1 g_1(\mathbf{x}) + \cdots + c_B g_B(\mathbf{x}) = c_0 + \sum_{b=1}^{B} \sum_{i=1}^{n} c_b \alpha_i^b \kappa(\mathbf{x}_i, \mathbf{x}). \tag{2.14}$$

As mentioned in Remark 1, even if the linear design condition is violated, the biases are not large for SIR. The biases caused by the violation of (2.14) are even smaller for introducing nonlinear kernel mixtures for KSIR modeling.

Remark 3. Huang and Hwang (2006) investigated some fundamental properties of data represented via kernel transformation into feature space. They suggested that under suitable conditions, most low-dimensional projections of kernel data are approximately Gaussian in the weak sense. Since our procedures function in a low-dimensional working subspace, namely the subspace spanned by $\{\sum_{i=1}^n \alpha_i^1 \kappa(\mathbf{x}_i, \mathbf{x}), \ldots, \sum_{i=1}^n \alpha_i^B \kappa(\mathbf{x}_i, \mathbf{x})\}$; their finding provides a justification for the Gaussian, or elliptical symmetric distribution assumption for kernel data. More information concerning the choice of the number of slices H and the SIR estimation procedure can be found in Li (1991) and Chen and Li (1998).

2.3 CENTERING AND REDUCED FEATURES

As outlined in the previous section, we have assumed the mapped data are centered in $\mathcal{H}$, that is, $\sum_{i=1}^n \Phi(\mathbf{x}_i) = 0$. Let the centered data be denoted by $\tilde{\Phi}(\mathbf{x}_i) := \Phi(\mathbf{x}_i) - \bar{\Phi}$. The corresponding kernel data matrix is defined by $\tilde{\mathbf{K}} := \langle \tilde{\Phi}(\mathbf{x}_i), \tilde{\Phi}(\mathbf{x}_i) \rangle_{\mathcal{H}}$. However, we cannot center $\Phi(\mathbf{x}_i)$'s directly, since Φ is not explicitly known. Instead, we show that $\tilde{\mathbf{K}}$ can be obtained by performing the desired algorithms without knowing Φ. Given a training dataset, for the supervised learning problem, $\{\mathbf{x}_i^r \in \Re^p\}_{i=1}^n$ with class labels $\{y_i^r\}_{i=1}^n$ and a test dataset $\{\mathbf{x}_t^e \in \Re^p\}_{t=1}^m$, we compute the kernel matrix, for the training data, $\mathbf{K}_{tr} = [\kappa_{ij}]_{n \times n}$, where $\kappa_{ij} = \kappa(\mathbf{x}_i^r, \mathbf{x}_j^r)$; and the kernel matrix, for the test data, $\mathbf{K}_{te} = [\kappa_{ti}]_{m \times n}$, where $\kappa_{ti} = \kappa(\mathbf{x}_t^e, \mathbf{x}_i^r)$. Note that $(\kappa_{ti}/\sqrt{\kappa_{ii}})\Phi(\mathbf{x}_i^r)$ is the projection of the test data $\Phi(\mathbf{x}_t^e)$ along the training data $\Phi(\mathbf{x}_i^r)$ in the high-dimensional feature space $\mathcal{H}$. We next centralize $\mathbf{K}_{tr}$ and $\mathbf{K}_{te}$ as follows

$$\tilde{\mathbf{K}}_{tr} = \left(I_n - \frac{1}{n}\mathbf{1}_n \mathbf{1}_n^t\right) \mathbf{K}_{tr} \left(I_n - \frac{1}{n}\mathbf{1}_n \mathbf{1}_n^t\right),$$

and

$$\tilde{\mathbf{K}}_{te} = \left(\mathbf{K}_{te} - \frac{1}{n}\mathbf{1}_m\mathbf{1}_n^t\mathbf{K}_{tr}\right)\left(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^t\right),$$

where I_n is the identity matrix with length n, and $\mathbf{1}_n$ is a column vector consisting of all ones.

The dimension of the kernel data equals at most the number of observations. However, working in such a space can pose numerical difficulties. To deal with this, one can use only a subset of the training data by randomly sampling $\{\mathbf{x}_i^r\}_{i=1}^{n'}$ with size $n' < n$, to compute the reduced kernel matrices, $\mathbf{K}_{tr}' = [\kappa_{ij}]_{n\times n'}$ and $\mathbf{K}_{te}' = [\kappa_{ti}]_{m\times n'}$, for the training data and the test data. For more sophisticated methods, one can use leading components by performing singular value decomposition (SVD) or use other forms of capacity control or regularization (Lee and Huang 2007). For additional discussion of this issue, such as issues surrounding the low-rank approximations of Gram matrices, see Smola and Schölkopf (2000) and Williams and Seeger (2001). The approaches they suggested often provide sufficient fidelity for the needs of kernel-based algorithms and are applicable to large problems. Notice that in the following description of the KSIR algorithm, Steps 1–3 are exactly the same as the classical SIR except for the data used.

KSIR algorithm:

0. Prepare the data in kernel form $\tilde{\mathbf{K}}$, where $\tilde{\mathbf{K}}$ is centered and possibly reduced by random sampling or by SVD.
1. Partition a range of y into H slices to get the discretized $\tilde{y}$.
2. Calculate within-slice means for each slice and the between-slice covariance matrix, denoted by $\Sigma_{E(\tilde{\mathbf{K}}|\tilde{y})}$, using weights similar to those of Equation (2.2). Also calculate the covariance matrix for the kernel data, denoted by $\Sigma_{\tilde{\mathbf{K}}}$.
3. Extract the leading eigenvalues and eigenvectors of the between-slice covariance $\Sigma_{E(\tilde{\mathbf{K}}|\tilde{y})}$ with respect to $\Sigma_{\tilde{\mathbf{K}}}$. This is equivalent to solving the eigen-problem in equation (2.10).
4. Normalize the eigenvectors and get the projection directions by Equations (2.11) and (2.12). The normalization does not affect the KSIR directions but only affects their scales.

3. RELATIONS TO OTHER METHODS

KSIR vs. KPCA. Similar to PCA (Jolliffe 1986), SIR is a method based on the projection of input variables onto the latent variables (components). However, in contrast to PCA, SIR creates the components by modeling the relationship between input and response variables while maintaining most of the information in the input variables. SIR can be seen as a PCA-like procedure performed on the random variable $E(\mathbf{x}|y)$ instead of on $\mathbf{x}$. That is, SIR looks for linear combinations of $\mathbf{x}$ which maximize $\text{var}(E(\boldsymbol{a}^t\mathbf{x}|y))/\text{var}(\boldsymbol{a}^t\mathbf{x})$ instead of just $\text{var}(\boldsymbol{a}^t\mathbf{x})$. While PCA leads to an eigen-problem, SIR leads to a generalized eigen-problem. The similar results obtained can be extended and applied to their kernelized approaches.

A semiparametric method. SIR is a spectrum analysis of $\text{var}(E(\mathbf{x}|y))$ with respect to $\text{var}(\mathbf{x})$. KSIR is a spectrum analysis of a generalized association measure $\text{var}(E(\Phi(\mathbf{x})|y))$ with respect to $\text{var}(\Phi(\mathbf{x}))$. SIR is described as a linear method of dimension reduction and fails to find direction in the null-column-space of $\text{var}(E(\mathbf{x}|y))$ where patterns are symmetric about the responses y. KSIR overcomes these limitations while offering a simple method that can be generalized as nonlinear algorithm. KSIR shares many properties with SIR in which no nonlinear optimization is involved. One only must solve a generalized eigen-problem as in the case of a standard SIR. In contrast to linear SIR, KSIR is capable of capturing part of the higher-order statistics which are particularly important for visualizing complex data structures.

KSIR-based variates versus KFDA- and KCCA-based variates. The leading canonical variates in Fisher linear discriminant analysis (LDA) are the linear combinations of $\mathbf{x}$ formed by the vector $\boldsymbol{a}$, which solves the maximization problem by $\max_a(\boldsymbol{a}^t \Sigma_B \boldsymbol{a}/\boldsymbol{a}^t \Sigma_W \boldsymbol{a})$, where Σ_B and Σ_W are the between-group and within-group covariances, respectively. The solutions are the leading eigenvalues and associated eigenvectors of the following eigenvalue decomposition

$$\Sigma_B \boldsymbol{a}_i = \gamma_i \Sigma_W \boldsymbol{a}_i, \quad \gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_p. \tag{3.1}$$

By the equality $\Sigma_{\mathbf{xx}} = \Sigma_B + \Sigma_W$, we obtain an equivalent system

$$\Sigma_B \boldsymbol{a}_i = \frac{\gamma_i}{1+\gamma_i} \Sigma_{\mathbf{xx}} \boldsymbol{a}_i. \tag{3.2}$$

Compared to the SIR-found dimension reduction directions $\boldsymbol{\beta}_i$ in (3), we have $\boldsymbol{a}_i \propto \boldsymbol{\beta}_i$ and $\lambda_i = \gamma_i/(1+\gamma_i)$. That is, the SIR variates are equivalent to the Fisher canonical variates except for possible difference in scale (Chen and Li 2001). See Kent (1991) for more discussion on the connection between SIR and LDA. Kernel tricks indicate that the kernelized algorithms of SIR and LDA are simply the corresponding linear algorithm operated on the kernel matrix. Therefore, the projection directions found by KSIR are equivalent theoretically to those of KFDA. In practice, the differences can arise depending on the methods used to estimate the between-group and within-group covariances. Kuss and Graepel (2003) addressed the connection between the kernel Fisher discriminant analysis with the kernel canonical correlation analysis. As a result, KSIR also has a link to the KCCA and KFDA variates.

4. KSIR FOR NONLINEAR DIMENSION REDUCTION AND DATA VISUALIZATION

KSIR as a dimension reduction tool adopts the response information y for estimating the nonlinear EDR subspace, a central subspace in $\mathcal{H}$. Taking into account nonlinear features can be beneficial in extracting interesting and important nonlinear data structures. A plot of y versus the first few KSIR variates can be informative. If the responses y's in the experiments are of a finite number of categories C, we have the natural slicing for $H = C$.

To demonstrate the visualization capability of KSIR for nonlinear dimension reduction, we have conducted studies on simulated data and real-world data using polynomial kernels and Gaussian kernels. Some of the results are reported in the following. For complete examples with figures (including 3D animation of the dimension reduction projections), please refer to our Web site (*http://www.hmwu.idv.tw/KSIR/*).

Three commonly used kernels in the study are

- linear kernel: $\kappa(\mathbf{x}, \mathbf{u}) = \langle \mathbf{x}, \mathbf{u} \rangle$;
- Gaussian radial basis function (RBF): $\kappa(\mathbf{x}, \mathbf{u}) = \exp\{-\text{scale} \cdot \|\mathbf{x} - \mathbf{u}\|^2\}$; and
- polynomial kernel: $\kappa(\mathbf{x}, \mathbf{u}) = (\text{scale} \cdot \langle \mathbf{x}, \mathbf{u} \rangle + \text{offset})^{\text{degree}}$,

where scale, degree, and offset are three parameters that must be determined beforehand. Although the choice of optimal kernel and its associated parameters have been investigated by the machine learning community (known as model selection problem), the optimal model parameters are generally domain-specific (Keerthi and Lin 2003; Duan, Keerthi and Poo 2003). The Gaussian kernel is most popularly used when there is no prior knowledge regarding the data. In our study, the parameters are selected by a limited empirical search of the parameter space. For example, the scales of 0.01, 0.1, 1, and 10 are used for Gaussian kernels and the degrees of 1, 2, 3, and 4 for polynomial kernels.

Before applying the dimension-reduction algorithm, we first standardize each variable of the data to have zero mean and unit variance. It is often effective to linearly scale each variable, and then to apply the Gaussian RBF kernel or polynomial kernel (Hsu, Chang,

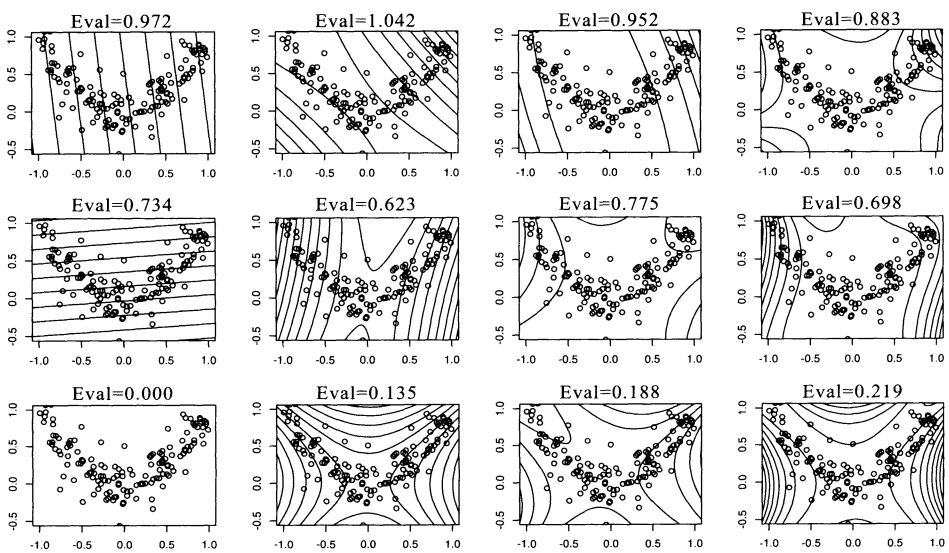

Figure 1. SIR vs. KSIR with polynomial kernels for the *square data*. From left to right, the polynomial degree of the kernel increases from 1 to 4 (degree = 1 for SIR and degree = 2, 3, 4 for KSIR). From top to bottom, contour lines of constant value of the first three eigenvectors with the corresponding eigenvalues are shown. Note that only two eigenvectors are available in linear SIR.

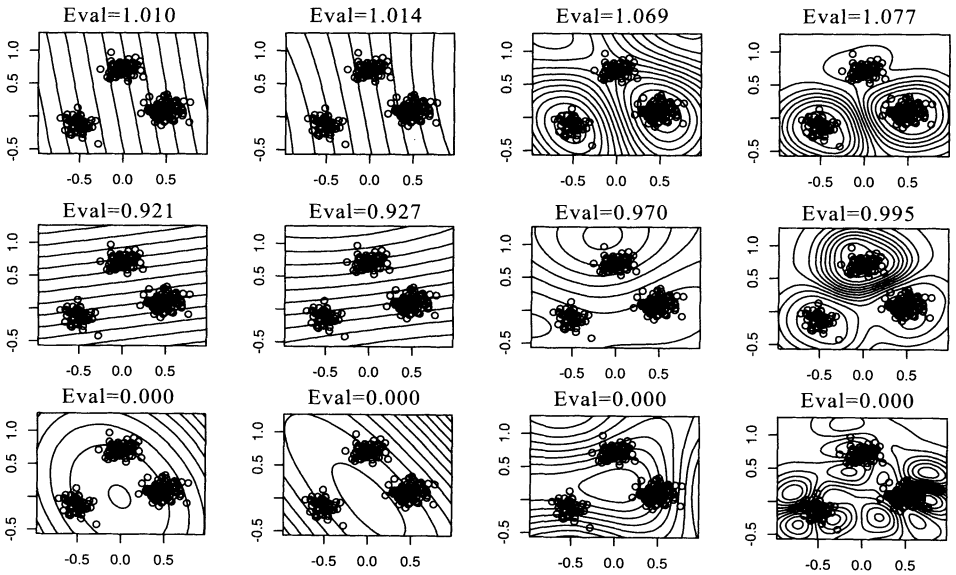

Figure 2. KSIR with Gaussian kernels for the *three clusters data*. From left to right, the scale of the kernel is set by 0.01, 0.1, 1, and 10. From top to bottom, contour lines of constant value of the first three eigenvectors with the corresponding eigenvalues are shown.

and Lin 2003). This standardization prevents variables with larger numeric ranges from dominating those with smaller ranges. The same setting is also employed in the classification experiments.

Example 1. *(square data)* To provide some intuition as to how the behavior of the KSIR directions in $\mathcal{H}$ can be traced back to the original input space, we first perform an experiment on a simple artificial 2D dataset. The square data of 150 points is generated in the following way: x_1-values are uniformly distributed in $[-1, 1]$, and x_2-values are generated as $x_2 = x_1^2 + 0.2\epsilon$, where ϵ is the standard normal noise. The response y is obtained by performing a K-means clustering with eight clusters labeled 1 through 8. Figure 1 shows the contour plots of the SIR feature extractors using polynomial kernels of various degrees. The lines of contour plots correspond to projections onto the first three eigenvectors in the feature space from top to bottom. The figures in the left-most columns have degree $d = 1$ and apply to classical SIR, while the other columns apply to KSIR having degrees ranging from 2 to 4. Contour lines with $d = 1$ are straight. This means that SIR linearly transforms the data and fails to find nonlinear directions. On the other hand, contour lines with $d = 2$ and 3 are curved lines which indicate that KSIR can extract nonlinear components from input data. In summary, KSIR ($d > 1$) extracts features which nicely increase along the curved directions of main variance structure in the data.

Example 2. *(three clusters data)* The three-clusters data are generated in a two-dimensional Euclidean space. We place three points, $(-0.5, -0.1)$, $(0, 0.7)$, and $(0.5, 0.1)$, in the two-dimensional space as cluster centers; and then Gaussian noises with standard deviation of 0.1 are sampled and added to the cluster centers to form our dataset. We have the

imbalanced cluster sizes 50, 70, and 100 here. Figure 2 shows the contour plots of performing KSIR with Gaussian kernels. From left to right, the scale of the kernel is set to 0.01, 0.1, 1, and 10. From top to bottom, contour lines of the first three eigenvectors are shown. The first two eigenvectors separate the three clusters in the case of scales 1 and 10. Large values of scale make the KSIR feature extractors increasingly nonlinear, which is useful for the identification of the clustering structure. Contour lines of the third component in the case of scale 0.1 are along the circle with an eigenvalue of zero. This means that the component picks up the variation caused by the noise.

Example 3. *(wine data)* To illustrate how KSIR projection onto a low-dimension space represents an effective description of real-world datasets, a wine recognition dataset from the UCI machine learning repository (Murphy and Aha 1994) is employed. The data consist of 178 samples with 13 constituent variables and a label variable for three types of wines. Figure 3 shows the projection onto the two-dimensional subspace estimated by PCA, SIR, KPCA, and KSIR. The Gaussian kernel with scale of 0.05 is used here. As we can see, the projected data are visually separable in the two-dimensional subspace estimated by PCA, SIR, and KPCA; while KSIR provides the patterns that are most distinctly separated.

Example 4. *(pendigit data)* In all examples above, we use the full dimension of kernel mixtures. To demonstrate the capability of KSIR to deal with reduced features, a pen-based recognition of handwritten digits dataset from the UCI machine learning repository is employed. The data consist 7,494 samples with 16 variables and a label variable for ten class digit codes. We use 200 reduced features by stratified random subset to perform KPCA and KSIR with a Gaussian kernel of scale 0.05. Figure 4 shows the results of the projection onto the two-dimensional subspace estimated by PCA, SIR, KPCA, and KSIR. Evidently, KSIR maintains the superior performance over others with respect to discriminative and visualization purposes.

Example 5. *(Li data model)* The Li data model (Li 1991, Equation (6.3)) is generated by

$$y = \frac{x_1}{0.5 + (x_2 + 1.5)^2} + \sigma\epsilon, \tag{4.1}$$

where $x_1, \ldots, x_p$ and ϵ follow iid standard normal. We take $p = 10$ and $\sigma = 0.5$. The sample size is set at $n = 400$. The purpose of this data example in Li (1992) is to use the first two components of SIR as estimates of EDR directions. The true EDR directions are the vectors $(1, 0, \ldots, 0)$ and $(0, 1, 0, \ldots, 0)$ in the 10-dimensional Euclidean space. However, in the KSIR scheme, the estimates of EDR directions in the feature space become nonlinear. We are interested in seeing how the nonlinear structure that KSIR can preserve. Figure 5(a) shows the 3D plot of y against x_1, x_2. Figure 5(b) shows the 3D plot of y against the first two projections of SIR with the number of slices $H = 13$. Clearly, SIR recovers the data structure in two directions. If the algorithm for estimating directions does not involve the information of y such as kernel PCA does, the data structure recovered in

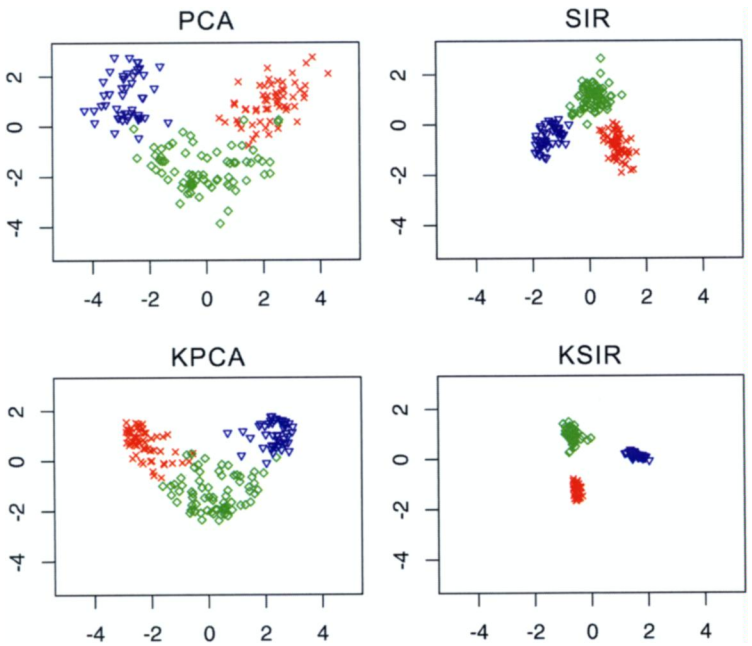

Figure 3. The projections for the *wine data* onto the estimated first two-dimensional subspace. The symbols represent the three classes. A Gaussian kernel with scale of 0.05 is used.

two dimensions is clouded, as shown in Figure 5(c). Figure 5(d) shows the results of KSIR with Gaussian kernel of scale 0.05. The spinning plot exhibits spiral structure along the y direction.

Example 6. *(curves and clusters)* This example has been employed for studying regression problems by SIR analysis (Chen and Li, 1998, p. 298). The data are simulated from the model

$$y = \text{sign}(\beta_1'\mathbf{x} + \sigma_1\epsilon_1)\log(|\beta_2'\mathbf{x} + \alpha + \sigma_2\epsilon_2|), \tag{4.2}$$

where $\mathbf{x} = (x_1, \ldots, x_{15})$ and ϵ_1, ϵ_2 are independent standard normal random variables. The data are generated by 300 cases with $\beta_1' = (1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)$, $\beta_2' = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)$, $\alpha = 5$, and $\sigma_1 = \sigma_2 = 1$. Using multiple linear regression (MLR) as an illustration, we get 52.2% of the R^2 value of the fit. Then we fit the MLR of y on the first two SIR-variates and KSIR-variates (the number of slices is equal to 15). The R^2 values are of 51.7% and 75.5%, respectively. The rotation plots for the response y against the first two KSIR projections are shown in Figure 6(a)–(d). We can see that the first projection (`beta(1)'X`) finds two curves spreading out symmetrically about the y axis and the second one (`beta(2)'X`) shows a pattern of two clusters. By investigating these low dimensional nonlinear confounding patterns, the nonmonotone transformation on y may increase the linear fit. We then applied MLR models and got 80.9% and 86.9% of the R^2 values of the fit when the linear model is of y^2 against first two SIR-variates and KSIR-variates.

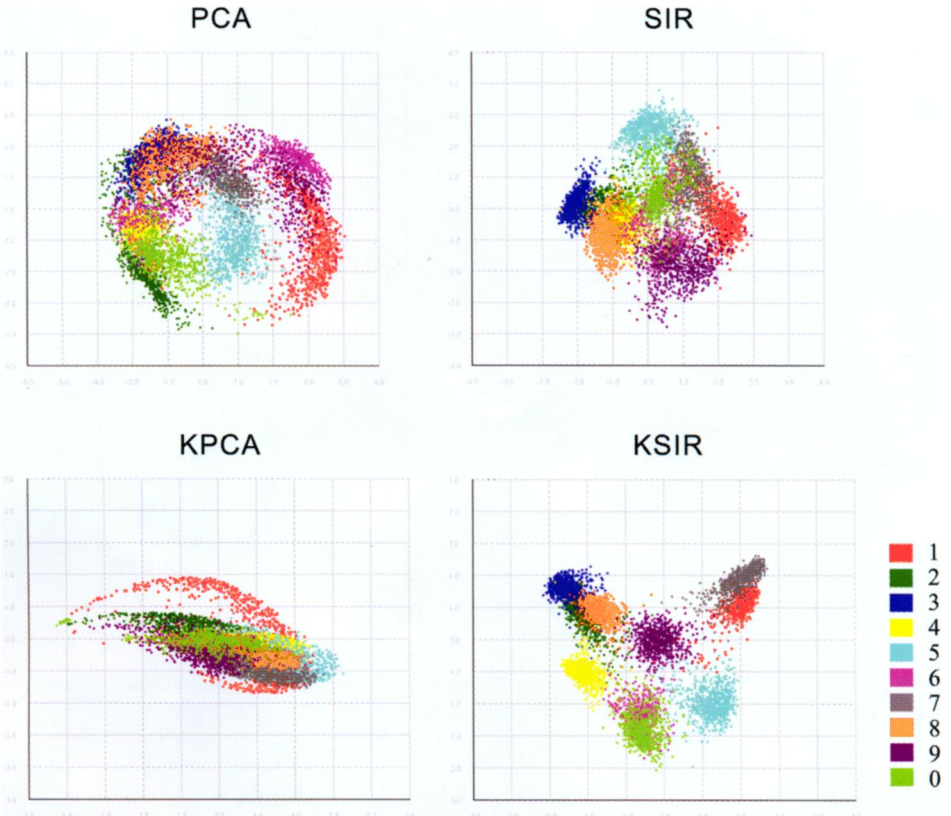

Figure 4. The projections for the *pendigit data* onto the estimated first two-dimensional subspace. The colors represent the ten classes. The 200 reduced features are used by stratified random subset for performing KPCA and KSIR with a Gaussian kernel of scale 0.05.

Our examples show the ability of KSIR to preserve information of y in functional subspaces. Particularly, in Examples 2–4, the directions obtained by KSIR exhibit the largest difference among group means relative to the within-group variance, and are superior for classification purposes. Examples 5 and 6 are used to show that KSIR as well as SIR can help to summarize the information to study the relationship between the response y and the regressor variables $\mathbf{x}$ in multiple regression problems.

5. EXPERIMENTS ON CLASSIFICATION

To compare the numerical performance of projection directions found by different approaches in classification problems, we apply PCA, SIR, KPCA, and KSIR as feature extractors on real world datasets and microarray data. We then use a *linear* support vector classifier to evaluate the discriminatory ability of these four feature extraction methods. We use a one-against-one approach for multiclass classification which is implemented in R package: `e1071` (`svm` routine). This approach constructs $C(C-1)/2$ classifiers, where C is the number of classes. A test sample is classified by majority votes. If two or more top ranking groups have identical numbers of votes, `svm` selects the one with the smallest in-

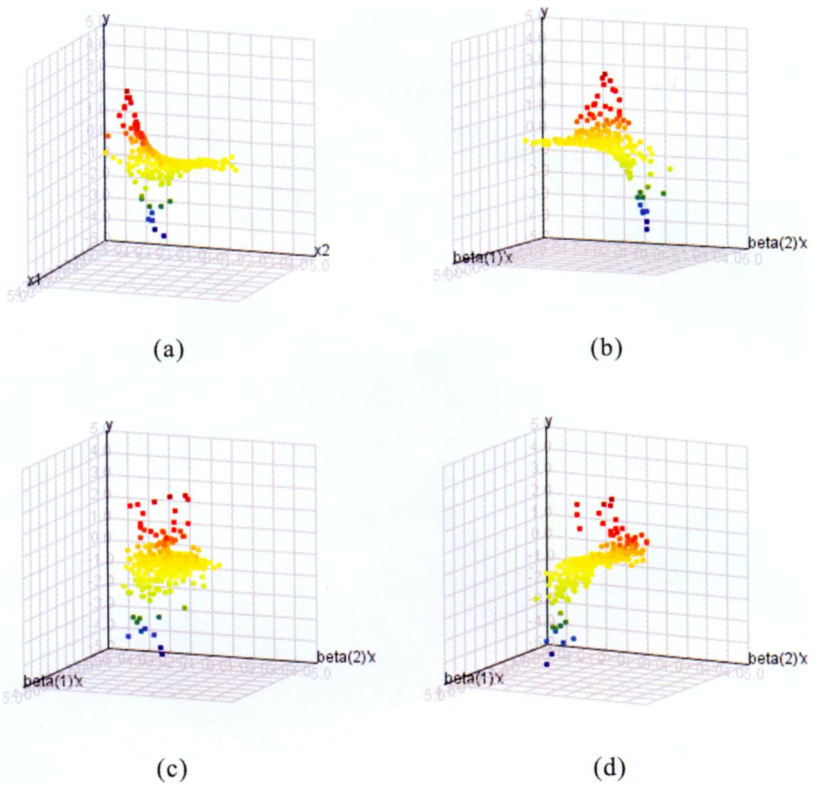

Figure 5. Projection view of the *Li data model (6.3)* with different rotations: (a) best view of the data, (b) SIR, (c) KPCA, and (d) KSIR. A Gaussian kernel with scale of 0.05 is used. The 3D animations for the projection view are available on our Web site (*http://www.hmwu.idv.tw/KSIR/*).

dex. Since a linear SVM finds an optimal linear separator between the two classes of data, it is suitable for use in the evaluation of the discriminatory power of different feature extraction methods. In addition to comparing different extracted features, we also compared performance based on the full dimensional vector **x**.

5.1 ON CLASSIFICATION OF UCI DATASETS

Table 1 shows a summary of characteristics of the datasets we have investigated; they are available in the UCI machine learning repository (Murphy and Aha 1994). All datasets used contain no missing observations. These datasets are commonly used for the conventional multivariate statistical analysis in which $n \gg p$. Before performing classification, the training set and test set are stratifiedly sampled, so that the distributions of the datasets are similar. For datasets with large n such as `seg` and `veh`, a stratified random subset of 200 reduced features is used. We reported 10-fold cross-validation error rates for various dimensionality (i.e., number of features used) from one to ten, as shown in Figure 7. A Gaussian kernel with scale 0.05 is used.

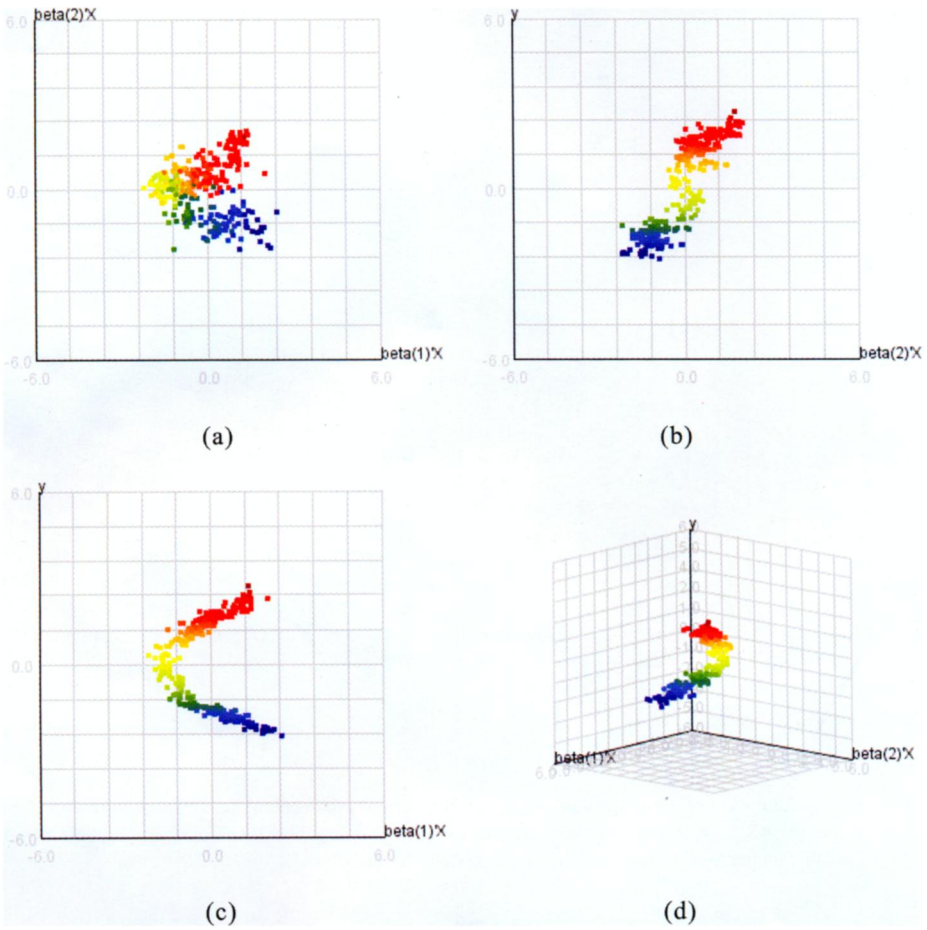

Figure 6. KSIR's view of the *curves and clusters data* generated from Equation (4.2). A Gaussian kernel with scale of 0.05 is used. The colors represent the values of the response y.

Table 1. Characteristics of the selected UCI datasets.

Dataset		n	p	C (n_h)
bcw	Wisconsin breast cancer	683	9	2 (444, 239)
gls	Glass identification	214	9	6 (70, 76, 17, 13, 9, 29)
ion	Ionosphere	351	33	2 (225, 126)
iri	Iris plants	150	4	3 (50×3)
liv	BUPA liver disorders (liv)	345	6	2 (145, 200)
pid	Pima Indians diabetes	768	8	2 (500, 268)
seg	StatLog image segmentation	2310	18	7 (330×7)
veh	StatLog vehicle silhouettes	846	18	4 (212, 217, 218, 199)
wav	Waveform database generator	600	21	3 (200×3)
win	Wine recognition data	178	13	3 (59, 71, 48)

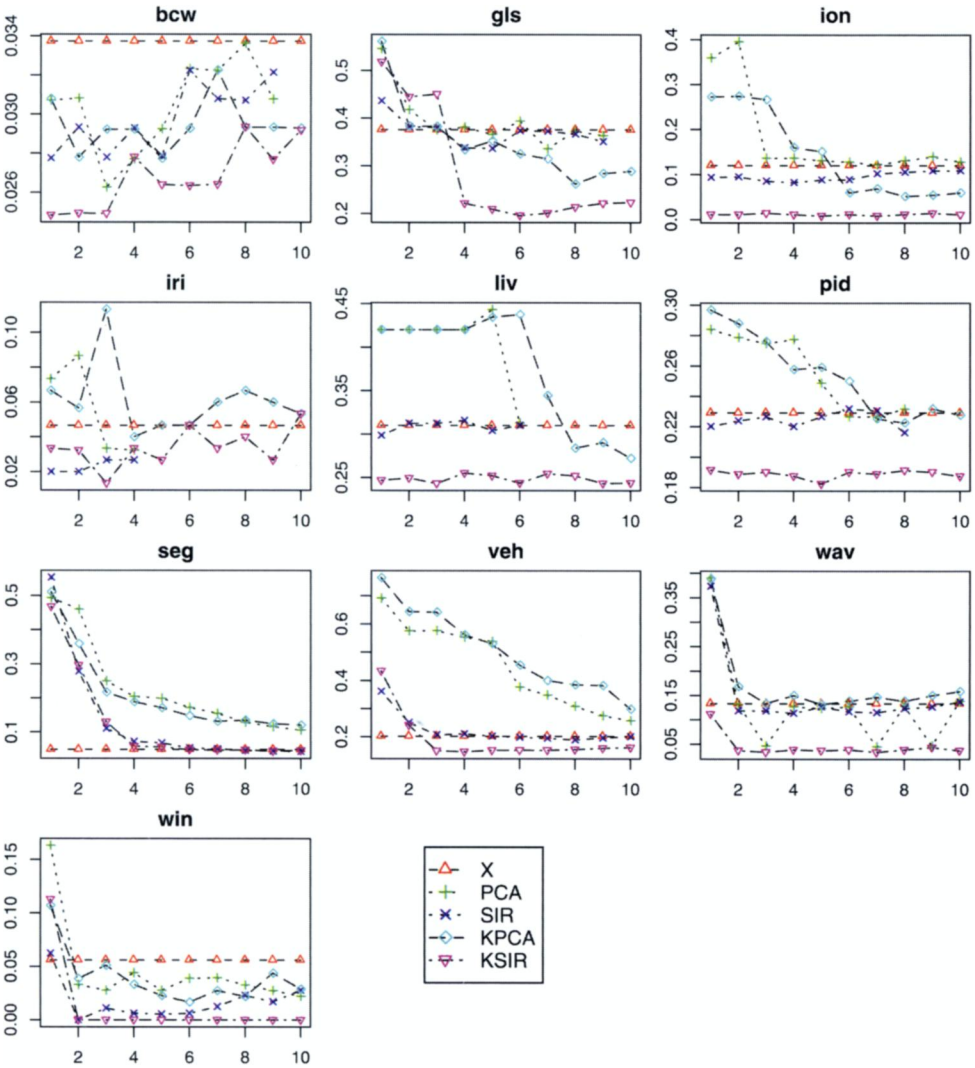

Figure 7. Classification error rates with 10-fold cross-validation against the dimensionality from one to ten based on dimension reduction variates and the full dimensional space vector **x** for ten UCI datasets. A Gaussian kernel with scale of 0.05 is used.

Based on a linear support vector classifier, Table 2 lists the number of top-rank performance counts in classification accuracy for the five methods among ten UCI datasets. For each dataset, mark one count for the method with the highest test accuracy. If there are ties between two methods, assign 1/2 to each of the two. As we can see, KSIR outperforms others in most datasets for various dimensionalities. In addition, the error rates of KSIR for all data used can be reduced significantly by using the first three or four dimensions. KSIR yields good separation even in low-dimensional subspace, while PCA and KPCA are much worse in low dimensions. For data `seg`, the classifier outperforms others in the full dimensional vector **x** with error rates of 0.0489. For data `bcw` and `iri`, fluctuations of the error rates appear in KSIR. This may happen when eigenvectors associated with zero

Table 2. Number of top-rank performance counts for five methods among ten UCI datasets.

	Dimensionality									
Methods	1	2	3	4	5	6	7	8	9	10
X (full dimension)	4	3	1	1	1	1	0	0	0	1
PCA	0	0	1	1	0	0	0	0	0	0
SIR	1	1.5*	0	1	0	0	0	1	0.5*	1
KPCA	0	0	0	0	0	0	0	1	0	0
KSIR	5	5.5*	8	7	9	9	10	8	9.5*	8

* Ties.

eigenvalues are involved in the prediction of y. These eigenvectors contribute little or not at all to the discriminatory power, and at times they behave as noises. These noises are irrelevant to the prediction of y. In summary, the KSIR features successfully find a subspace which preserves the class structure for classification in low-dimension set-up.

5.2 On Classification of Microarray Datasets

Gene expression data are characterized by many variables (genes) of only a few observations (subjects) (i.e., $n \ll p$). One important application of gene expression data is the classification of samples into different categories, such as types of tumors or diseases. In the case of $n \ll p$, the covariance matrix $\Sigma_{\mathbf{xx}}$ is singular and not invertible, since it has rank of at most $n-1$ and dimensions $p \times p$. However, we can take advantage of kernel matrices to reduce the computational cost by performing algorithms on these $n \times n$ matrices. The KPCA and KSIR features with a polynomial kernel of degree 3 and a Gaussian kernel of scale 0.05 are applied to six different gene expression datasets, shown in Table 3 (Dettling and Bühlmann 2002). For illustrative purposes, we did not consider the issue of gene selection in reducing the number of variables. Figure 8 shows the classification error rates for dimensionality from one to ten with a leave-one-out cross-validation. As we can see, KSIR provides better classification rates among the competitors studied. KSIR extracts the features which are most useful for classification purposes. On the other hand, KPCA using polynomial kernels outperforms those using Gaussian kernels.

Table 3. Six publicly available microarray datasets.

Dataset	Publication	n	p	$H(n_h)$	Response
Leukemia	Golub et al.(1999)	72	3571	2 (47, 25)	Subtypes of leukemia
Colon	Alon et al.(1999)	62	2000	2 (22, 40)	Tumor/normal tissue
Prostate	Singh et al.(2002)	102	6033	2 (50, 52)	Tumor/normal tissue
Lymphoma	Alizadeh et al.(2000)	62	4026	3 (42, 9, 11)	Subtypes of lymphoma
SRBCT	Khan et al. (2001)	63	2308	4 (23, 20, 12, 8)	Different tumor types
Brain	Pomeroy et al. (2002)	42	5597	5 (10, 10, 10, 4, 8)	Different tumor types

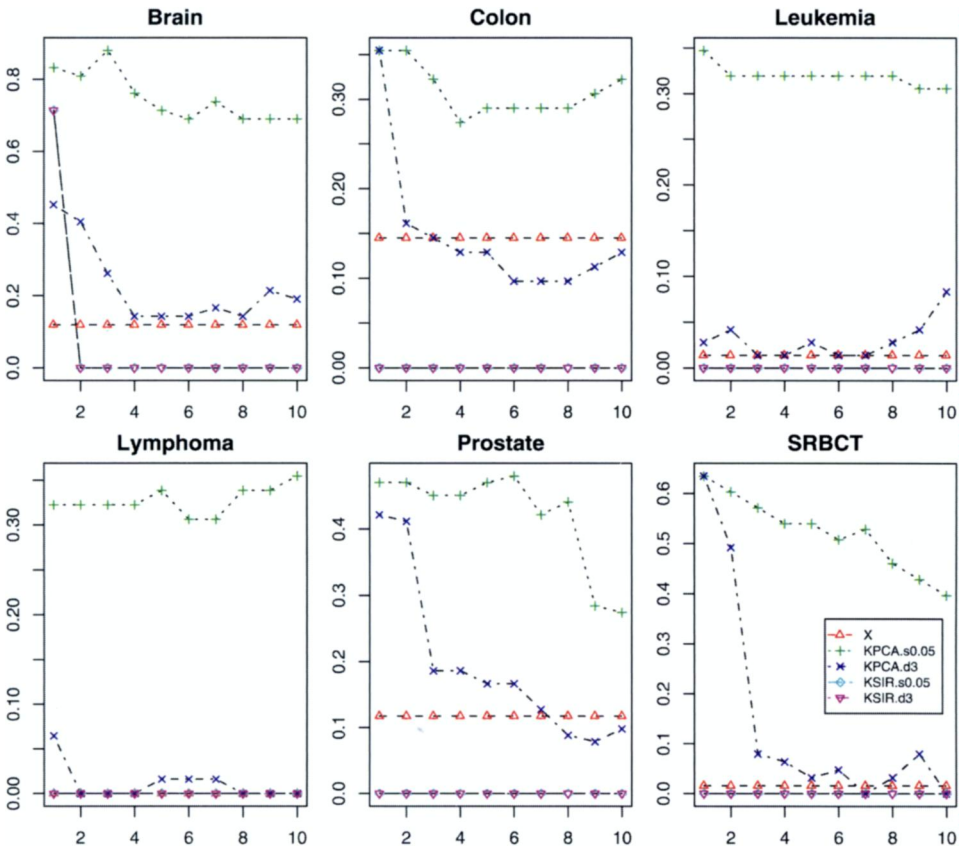

Figure 8. Classification error rates with a leave-one-out cross-validation against the dimensionality from one to ten based on dimension reduction variates and the full dimensional space vector $\mathbf{x}$ for six public microarray datasets. A Polynomial kernel with degree of 3 and a Gaussian kernel with scale of 0.05 are used.

6. CONCLUSION AND DISCUSSION

In this study, we have used the kernel trick to generalize the linear algorithm of SIR to a nonlinear feature space endowed with a kernel. KSIR as a feature extractor can be regarded as the equivalent of the classical SIR performed on a kernel data matrix. This approach has the ability of capturing the intrinsic nonlinear structure of the data via nonlinear dimension reduction from a $\Re^p$ viewpoint. KSIR has proven to be a powerful and promising feature extraction approach which achieves superior performance to PCA, SIR, and KPCA in classification problems. In addition, KSIR provides data visualization capabilities for the presence of clusters or outliers, yielding superior discriminatory power. In common with SIR, KSIR is helpful for exploring the global structure of data and can be used to suggest appropriate parametric models such as multiple linear regression. Other than classification and regression, it can potentially suit many other applications. Although, in recent years, KPCA has been suggested for various applications for use in the preprocessing step, the main strength of KSIR compared to the KPCA algorithm is that it allows the utilization of class information. We therefore suggest KSIR as a preprocessing step for classification

and regression problems. The general question of how to choose the best kernel and the number of KSIR-variates for a given problem requires further study.

APPENDIX

Here we present a detailed proof for the equivalence of (2.8) and (2.10). First calculate

$$\begin{aligned}\left\langle \Phi(\mathbf{x}_k), \sum_{h=1}^{H} p_h \bar{\Phi}(\mathbf{m}_h) \right\rangle_{\mathcal{H}} &= \sum_{h=1}^{H} p_h \langle \Phi(\mathbf{x}_k), \bar{\Phi}(\mathbf{m}_h) \rangle_{\mathcal{H}} \\ &= \sum_{h=1}^{H} p_h \left\langle \Phi(\mathbf{x}_k), \frac{\sum_{j=1}^{n} \Phi(\mathbf{x}_j)\delta_h(y_j)}{\sum_{j=1}^{n} \delta_h(y_j)} \right\rangle_{\mathcal{H}} \\ &= \sum_{h=1}^{H} \frac{\sum_{j=1}^{n} \mathbf{K}_{kj}\delta_h(y_j)}{n}, \quad \text{and}\end{aligned}$$

$$\begin{aligned}\langle \bar{\Phi}(\mathbf{m}_h), \Phi(\mathbf{x}_i) \rangle_{\mathcal{H}} &= \left\langle \frac{\sum_{j=1}^{n} \Phi(\mathbf{x}_j)\delta_h(y_j)}{\sum_{j=1}^{n} \delta_h(y_j)}, \Phi(\mathbf{x}_i) \right\rangle_{\mathcal{H}} \\ &= \frac{\sum_{j=1}^{n} \mathbf{K}_{ji}\delta_h(y_j)}{\sum_{j=1}^{n} \delta_h(y_j)}.\end{aligned}$$

For the left-hand side of Equation (2.9), we have

$$\begin{aligned}\langle \Phi(\mathbf{x}_k), \Sigma_{E(\mathbf{z}|\tilde{y})}\boldsymbol{v} \rangle_{\mathcal{H}} &= \left\langle \Phi(\mathbf{x}_k), \left\{ \sum_{h=1}^{H} p_h \bar{\Phi}(\mathbf{m}_h)\bar{\Phi}(\mathbf{m}_h)^t \right\} \left\{ \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{x}_i) \right\} \right\rangle_{\mathcal{H}} \\ &= \sum_{i=1}^{n} \alpha_i \left\langle \Phi(\mathbf{x}_k), \sum_{h=1}^{H} p_h \bar{\Phi}(\mathbf{m}_h) \right\rangle \langle \bar{\Phi}(\mathbf{m}_h), \Phi(\mathbf{x}_i) \rangle_{\mathcal{H}} \\ &= \sum_{i=1}^{n} \alpha_i \sum_{h=1}^{H} \frac{\sum_{j=1}^{n} \kappa_{kj}\delta_h(y_j)}{n} \frac{\sum_{j=1}^{n} \kappa_{ji}\delta_h(y_j)}{\sum_{j=1}^{n} \delta_h(y_j)} \\ &= \frac{1}{n} \sum_{i=1}^{n} \alpha_i \sum_{h=1}^{H} \frac{\sum_{j=1}^{n} \kappa_{kj}\delta_h(y_j)}{\sqrt{\sum_{j=1}^{n} \delta_h(y_j)}} \frac{\sum_{j=1}^{n} \kappa_{ji}\delta_h(y_j)}{\sqrt{\sum_{j=1}^{n} \delta_h(y_j)}}, \\ &\Rightarrow \frac{1}{n}\mathbf{K}\mathbf{E}_H\mathbf{K}\boldsymbol{\alpha},\end{aligned}$$

where $n_h = \sum_{j=1}^{n} \delta_h(y_j)$, $p_h = \frac{n_h}{n}$, $\mathbf{E}_H = \sum_{h=1}^{H} \frac{\mathbf{1}_h\mathbf{1}_h^t}{n_h}$, and $\mathbf{1}_h = [\delta_h(y_1) \ldots \delta_h(y_n)]^t$. For the right-hand side of Equation (2.9), we have

$$\begin{aligned}\lambda\langle \Phi(\mathbf{x}_k), \Sigma_{\mathbf{zz}}\boldsymbol{v} \rangle_{\mathcal{H}} &= \lambda\langle \Phi(\mathbf{x}_k), \left\{ \frac{1}{n} \sum_{j=1}^{n} \Phi(\mathbf{x}_j)\Phi(\mathbf{x}_j)^t \right\} \left\{ \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{x}_i) \right\} \rangle_{\mathcal{H}} \\ &= \lambda \frac{1}{n} \sum_{i=1}^{n} \alpha_i \left\langle \Phi(\mathbf{x}_k), \sum_{j=1}^{n} \Phi(\mathbf{x}_j) \right\rangle_{\mathcal{H}} \langle \Phi(\mathbf{x}_j), \Phi(\mathbf{x}_i) \rangle_{\mathcal{H}}\end{aligned}$$

$$= \lambda \frac{1}{n} \sum_{i=1}^{n} \alpha_i \sum_{j=1}^{n} \kappa_{kj} \kappa_{ji},$$

$$\Rightarrow \lambda \frac{1}{n} \mathbf{K}\mathbf{K}\boldsymbol{\alpha}$$

Therefore, solving the eigen-problem of $\lambda \Sigma_{\mathbf{zz}} \boldsymbol{\beta} = \Sigma_{E(\mathbf{z}|\bar{y})} \boldsymbol{\beta}$ is equivalent to solving $\lambda \mathbf{K}\mathbf{K}\boldsymbol{\alpha} = \mathbf{K}\mathbf{E}_H \mathbf{K}\boldsymbol{\alpha}$, and hence equivalent to solving (2.10).

ACKNOWLEDGMENTS

A suite of functions coded in R and Java for implementing the algorithms and additional examples are available at *http://www.hmwu.idv.tw/KSIR*. The author thanks Drs. Su-Yun Huang, Yuh-Jye Lee, and Chun-houh Chen and members of IE&SLR group at Institute of Statistical Science, Academia Sinica, for many helpful discussions and suggestions. The comments of the anonymous referees as well as the Associated Editor and Professor David van Dyk are gratefully acknowledged.

[Received August 2006. Revised January 2008.]

REFERENCES

Aizerman, M., Braverman, E., and Rozonoer, L. (1964), "Theoretical Foundations of The Potential Function Method in Pattern Recognition Learning," *Automation and Remote Control*, 25, 821–837.

Bach, F., and Jordan, M. (2002), "Kernel Independent Component Analysis," *Journal of Machine Learning Research*, 3, 1–48.

Baudat, G., and Anouar, F. (2000), "Generalized Discriminant Analysis Using a Kernel Approach," *Neural Computation*, 12, 2385–2404.

Berlinet, A., and Thomas-Agnan, C. (2004), *Reproducing Kernel Hilbert Spaces in Probability and Statistics*, Boston: Kluwer Academic Publishers.

Boser, B. E., Guyon, I. M., and Vapnik, V. (1992), "A Training Algorithm for Optimum Margin Classifiers," in *Fifth Annual Workshop on Computational Learning Theory*, Pittsburgh, ACM.

Bura, E., and Pfeiffer, R. M. (2003), "Graphical Methods for Class Prediction Using Dimension Reduction Techniques on DNA Microarray Data," *Bioinformatics*, 19, 1252–1258.

Chang, C. C., and Lin, C. J. (2004), "LIBSVM: a Library for Support Vector Machines," software available at *http://www.csie.ntu.edu.tw/~cjlin/libsvm*.

Chen, C. H., and Li, K. C. (1998), "Can SIR be as Popular as Multiple Linear Regression?" *Statistica Sinica*, 8, 289–316.

——— (2001), "Generalization of Fisher's Linear Discriminant Analysis via the Approach of Sliced Inverse Regression," *Journal of the Korean Statistical Society*, 30, 193–217.

Cook, R. D. (1994), "On the Interpretation of Regression Plots," *Journal of the American Statistical Association*, 89, 177–190.

Cook, R. D. (1996), "Graphics for Regressions With a Binary Response," *Journal of the American Statistical Association*, 91, 983–992.

Cook, R. D., and Yin, X. (2001), "Dimension-reduction and Visualization in Discriminant Analysis," *Australian and New Zealand Journal of Statistics*, 43, 147–200.

Cook, R. D., and Weisberg, S. (1991), Discussion of "Sliced Inverse Regression For Dimension Reduction," by Li, *Journal of the American Statistical Association*, 86, 328–332.

Cristianini, N., and Shawe-Taylor, J. (2000), *An Introduction to Support Vector Machines*, Cambridge, UK: Cambridge University Press.

Dettling, M., and Bühlmann, P. (2002), "Supervised Clustering of Genes," *Genome Biology*, 3(12), research0069.1-0069.15.

Duan, K., Keerthi, S. S., and Poo, A. N. (2003), "Evaluation of Simple Performance Measures for Tuning SVM Hyperparameters," *Neurocomputing*, 51, 41–59.

Fukumizu, K., Bach, F. R., and Jordan, M. I. (2004), "Dimensionality Reduction for Supervised Learning With Reproducing Kernel Hilbert Spaces," *Journal of Machine Learning Research*, 5, 73–99.

Girosi, F. (1998), "An Equivalence Between Sparse Approximation and Support Vector Machines," *Neural Computation*, 10, 1455–1480.

Hastie, T., Tibshirani, R., and Friedman, J. (eds.) (2001), *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, New York: Springer.

Hsu, C., Chang, C., and Lin, C. J. (2003), "A Practical Guide to Support Vector Classification," Dept. of Computer Science and Information Engineering, National Taiwan University. Available online at *http://www.csie.ntu.edu.tw/~cjlin/*.

Huang, S. Y., and Hwang, C. R. (2006), "Kernel Fisher Discriminant Analysis in Gaussian Reproducing Kernel Hilbert Space: Theory," unpublished manuscript, *http://www.stat.sinica.edu.tw/syhuang*.

Jolliffe, I. T. (ed.) (1986), *Principal Component Analysis*, New York: Springer.

Keerthi, S. S., and Lin, C.-J. (2003), "Asymptotic Behaviors of Support Vector Machines With Gaussian Kernel," *Neural Computation*, 15, 1667–1689.

Kent, J. T. (1991), Discussion of "Sliced Inverse Regression For Dimension Reduction," by Li, *Journal of the American Statistical Association*, 86, 336–337.

Kuss, M., and Graepel, T. (2003), "The Geometry of Kernel Canonical Correlation Analysis," Technical report, Max Planck Institute for Biological Cybernetics, Tübingen, Germany. Available online at *http://research.microsoft.com/~thoreg*.

Lai, P. L., and Fyfe, C. (2000), "Kernel and Nonlinear Canonical Correlation Analysis," *International Journal of Neural Systems*, 10(5), 365–377.

Lee, Y. J., and Huang, S. Y. (2007), "Reduced Support Vector Machines: a Statistical Theory," *IEEE Transactions on Neural Networks*, 18, 1–13.

Li, K. C. (1991), "Sliced Inverse Regression For Dimension Reduction," *Journal of The American Statistical Association*, 86, 316–342.

——— (1992), "On Principal Hessian Directions for Data Visualization and Dimension Reduction: Another Application of Stein's Lemma," *Journal of the American Statistical Association*, 87, 1025–1039.

Mika, S., Rätsch, G., Weston, J., Schölkopf, B., and Müller, K.-R. (1999), "Fisher Discriminant Analysis With Kernels," in *Proceedings of IEEE Neural Networks for Signal Processing Workshop*.

Murphy, P. M., and Aha, D. W. (1993), UCI Repository of Machine Learning Databases. University of California, Department of Information and Computer Science, Irvine, CA.

Pekalska, E., Paclik, P., and Duin, R. P. W. (2001), "A Generalized Kernel Approach to Dissimilarity-based Classification," *Journal of Machine Learning Research*, 2, 175–211.

Rosipal, R., and Trejo, L. T. (2001), "Kernel Partial Least Squares Regression in Reproducing Kernel Hilbert Space," *Journal of Machine Learning Research*, 2, 97–123.

Roth, V., and Steinhage, V.(2000), "Nonlinear Discriminant Analysis Using Kernel Functions," in *Advances in Neural Information Processing Systems*, Cambridge, MA: MIT Press, pp. 568–574.

Schölkopf, B., and Smola, A. J. (eds.) (2002), *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, Cambridge, MA: MIT Press.

Schölkopf, B., Smola, A., and Müller, K. R. (1998), "Nonlinear Component Analysis as a Kernel Eigenvalue Problem," *Neural Computation*, 10, 1299–1319.

——— (1999), "Kernel Principal Component Analysis," in *Advances in Kernel Methods: Support Vector Learning*, Cambridge, MA: MIT Press, pp. 327–352.

Schölkopf, B., Smola, A., Williamson, R. C., and Bartlett, P. L. (2000), "New Support Vector Algorithms," *Neural Computation*, 12, 1207–45.

Schölkopf, B., Tsuda, K., and Vert, J.-P. (eds.) (2004), *Kernel Methods in Computational Biology*, Cambridge, MA: MIT Press.

Smola, A. J., and Schölkopf B. (2000), "Sparse Greedy Matrix Approximation for Machine Learning," in *Proceedings of the 17th International Conference on Machine Learning*, Stanford University, CA, Morgan Kaufmann Publishers, pp. 911–918.

Tsuda, K. (1999), "Support Vector Classifier With Asymmetric Kernel Functions," in *Processing of the Seventh European Symposium on Artificial Neural Networks (ESANN)*, pp. 183–188.

Vapnik, V. (ed.) (1995), *The Nature of Statistical Learning Theory*, New York: Springer-Verlag.

Williams, C., and Seeger, M. (2001), "Using the Nyström Method to Speed up Kernel Machines," in *Advances in Neural Information Processing Systems*, 13, eds. T.K. Leen, T. G. Dietterich, and V. Tresp, Cambridge, MA: MIT Press, pp. 682–688.

Wu, H. M., and Lu, H. H.-S. (2004), "Supervised Motion Segmentation by Spatial-Frequential Analysis and Dynamic Sliced Inverse Regression," *Statistica Sinica*, 14, 413–430.