Supervised Principal Component Analysis: Visualization, Classification and Regression on Subspaces and Submanifolds

Elnaz Barshan[a], Ali Ghodsi[b], Zohreh Azimifar[a,*], Mansoor Zolghadri Jahromi[a]

[a]*Department of IT and Computer Engineering, School of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran*
[b]*Department of Statistics and Actuarial Science, School of Computer Science, University of Waterloo, Canada*

Abstract

We propose "Supervised Principal Component Analysis (Supervised PCA)", a generalization of PCA that is uniquely effective for regression and classification problems with high-dimensional input data. It works by estimating a sequence of principal components that have maximal dependence on the response variable. The proposed Supervised PCA is solvable in closed-form, and has a dual formulation that significantly reduces the computational complexity of problems in which the number of predictors greatly exceeds the number of observations (such as DNA microarray experiments). Furthermore, we show how the algorithm can be kernelized, which makes it applicable to non-linear dimensionality reduction tasks. Experimental results on various visualization, classification and regression problems show significant improvement over other supervised approaches both in accuracy and computational efficiency.

Keywords: Dimensionality reduction, Principal component analysis (PCA), Kernel methods, Supervised learning, Visualization, Classification, Regression

*Corresponding author: Tel.: +98 (711) 613 3036; fax: +98 (711) 647 4605.
Email addresses: barshan@cse.shirazu.ac.ir (Elnaz Barshan), aghodsib@uwaterloo.ca (Ali Ghodsi), azimifar@cse.shirazu.ac.ir (Zohreh Azimifar), zjahromi@cse.shirazu.ac.ir (Mansoor Zolghadri Jahromi)

1. Introduction

Principal Component Analysis (PCA) [1] is a classical data analysis method that provides a sequence of the best linear approximations to a given high-dimensional data set. It is one of the most popular techniques for dimensionality reduction. The subspace modeled by PCA captures the maximum variability in the data, and can be viewed as modeling the covariance structure of the data. However, its effectiveness is limited to unsupervised problems.

Consider the supervised task of predicting a dependent response random variable from an independent high-dimensional explanatory random variable. Conventional classification and regression methods usually generate unsatisfactory results due to the "curse of dimensionality" [2], whereby the number of data points required for learning grows exponentially with the dimensionality of the data. This problem will be intensified when the number of predictors greatly exceeds the number of observations.

A canonical example of this scenario is in DNA microarray experiments, which usually consist of expression values for thousands of genes while the number of observations (e.g., individual tumor samples) is relatively small. In other words, each observation is expressed by thousands of features. These situations are especially prone to the curse of dimensionality. So some form of dimensionality reduction as a pre-processing step is of critical importance.

On the other hand, the prominent dimensionality reduction techniques, such as PCA, are unsupervised. Therefore, it is not possible to guide the algorithm toward the modes of variability that are of particular interest. Instead, conventional PCA will ignore the response variable and discover a sequence of directions that correspond to the maximum variation of the covariate data. When the task is regression or classification, it would be preferable to project the explanatory variables along directions that are related to the response variable. That is, we are interested in certain modes of variability that are dependent on the response variable; this goal is not necessarily achieved by using the directions that have maximum variation.

In this paper we propose supervised dimensionality reduction technique called "Supervised Principal Component Analysis (Supervised PCA)". It is a generalization of PCA which aims at finding the principal components with maximum dependence on the response variables. In other words, we seek for a subspace in which the dependency between predictors and response variable is maximized. We show that conventional PCA is a special form of Supervised PCA as a general framework. Similar to PCA, Supervised PCA can be solved in closed-form and does not suffer from the high computational complexity of iterative optimization procedures. Our proposed Supervised PCA also investigates the quantitative value of target variable and thus it is applicable on regression problems. This property is in contrast with a large number of supervised dimensionality reduction techniques which consider only similarities and dissimilarities along the labels, a fact which causes these methods to be limited on classification problems only. Besides, we derive a dual formulation for Supervised PCA which significantly reduces the computational complexity of problems in which the number of predictors greatly exceeds the number of observations. In order for our method to be applicable for estimating nonlinear transformation of data, a kernel version of Supervised PCA is also proposed.

The rest of this paper is organized as follows: Section 2 precisely describes the mathematical framework of the problem of supervised dimensionality reduction. A brief overview on different categories of prominent methods of supervised dimensionality reduction is given in Section 3. Section 4 describes the dependence measurement criterion which our proposed method relies on. In Section 5, we introduce our new algorithm, Supervised PCA, and its extensions as well as its connection to conventional PCA. We examine the performance of the proposed method on various visualization, classification and regression problems in comparison with other methods in Section 6. Finally we conclude in Section 8.

2. Problem Definition

Suppose $(\mathcal{X}, \mathcal{Y})$ are drawn from distribution $P_{\mathcal{X},\mathcal{Y}}$, where $\mathcal{X} \in \mathbb{R}^p$ is a p-dimensional explanatory variable and $\mathcal{Y} \in \mathbb{R}^\ell$ is an ℓ-dimensional response variable[1]. Given i.i.d. samples $\{(\mathbf{x}_1, \mathbf{y}_1), \cdots, (\mathbf{x}_n, \mathbf{y}_n)\}$ as n realization of $(\mathcal{X}, \mathcal{Y})$, we are looking for an orthogonal projection U of $\mathcal{X}$ onto $\mathcal{S}$ such that $\mathcal{S} = U^T\mathcal{X}$ and $\mathcal{Y}$ depends mainly on $U^T\mathcal{X}$.

[1]$\mathcal{X}$ is called explanatory variable, covariate or feature. $\mathcal{Y}$ is called response variable, label, outcome or target variable. Note that $\mathcal{Y}$ could be continuous (regression) or discrete (classification). It could also be either single or multivariate.

3. Related Works

Most of the research in supervised dimensionality reduction is focused on learning a linear subspace. This body of research includes various approaches such as classical Fisher's Discriminant Analysis (FDA) [3], the large family of methods known as Metric Learning (ML) [4–12], the family of Sufficient Dimension Reduction (SDR) algorithms [13–20], and the method proposed by Bair et al. [21] known as supervised principal components (in this paper, we refer to this method as Bair's SPC).

3.1. Fisher's Discriminant Analysis (FDA)

Fisher's Discriminant Analysis (FDA) is a traditional method of supervised dimensionality reduction and continues to be practically useful. In brief, for a general C-class problem, FDA maps the data into a $(C-1)$-dimensional space such that the distance between means of the projected classes is maximized while the within-class variance is minimized.

3.2. Metric Learning

The Metric Learning (ML) family of techniques attempts to construct a Mahalanobis distance over the input space, which could then be used instead of Euclidean distances. This approach can be equivalently interpreted as learning a linear transformation of the original inputs, followed by using the Euclidean distance in the projected space. ML methods search for a metric (or equivalently, a linear transformation) under which points in the same class (similar pairs) are near each other and simultaneously far from points in the other classes.

3.3. Sufficient Dimensionality Reduction

Sufficient Dimensionality Reduction (SDR) finds an orthogonal transformation U such that $\mathcal{Y}$ and $\mathcal{X}$ are conditionally independent given $U^T\mathcal{X}$. That is, SDR tries to preserve the conditional probability density function such that $P_{\mathcal{Y}|\mathcal{X}}(y|x) = P_{\mathcal{Y}|U^T\mathcal{X}}(y|U^Tx)$. It has been shown that U always exists[2], and in many cases U is not unique. Therefore, SDR searches for the "central subspace", which is the intersection of all such subspaces.

Most of the SDR algorithms are based on the idea proposed by Li [13, 15], which considers the SDR problem as an *inverse regression* problem. The

[2]The identity matrix is always a trivial solution.

main intuition here is to find $\mathbf{E}[\mathcal{X}|\mathcal{Y}]$ due to the fact that if the conditional distribution $P(\mathcal{Y}|\mathcal{X})$ varies along a subspace of $\mathcal{X}$, then the inverse regression $\mathbf{E}[\mathcal{X}|\mathcal{Y}]$ should also lie in $\mathcal{X}$ (See [13] for more details). Unfortunately, for this approach to be successful strong assumptions should be made about the marginal distribution $P_{\mathcal{X}}(x)$ (e.g., the distribution should be elliptical). Inverse regression methods can be effective if the assumptions that they embody are met, but they fail if the assumptions are not satisfied.

In order to overcome the above problem, Kernel Dimensionality Reduction (KDR) [20] has been proposed as an alternative approach. KDR makes no strong assumptions about either the conditional distribution $P_{\mathcal{Y}|U^T\mathcal{X}}(y|U^Tx)$ or the marginal distribution $P_{\mathcal{X}}(x)$. It tries to quantify the notion of conditional dependency by making use of the conditional covariance operators defined on reproducing kernel Hilbert spaces (RKHS's) [22]. The conditional independence is then imposed by minimizing the conditional covariance operator in a RKHS. A nonlinear extension of KDR, called manifold KDR (mKDR), is proposed in [23], in which the original input space is mapped to a nonlinear manifold, and KDR is applied to find a linear subspace of the manifold. This algorithm provides a nonlinear projection of the given training data, but has no straightforward extension for the new test points. That is, for a new data point, one has to rebuild the entire manifold including the new input data. None of these SDR methods have a closed-form solution.

3.4. Bair's Supervised Principal Components

Bair's Supervised Principal Components (Bair's SPC) is an effective heuristic for supervised dimensionality reduction, especially in the case of regression problems. This method is similar to conventional PCA, except that it uses the subgroup of features with the highest dependence on the outcome Y rather than using all of the features. BSPC can be viewed as a preprocessing step for the conventional PCA, in which the irrelevant sources of variation are removed based on their scores. The BSPC procedure is summarized as follows.

Assume we have a set of n data points $\{\mathbf{x}_i\}_{i=1}^n$ each consisting of p features, stacked in the $p \times n$ matrix X, and denote the jth feature (jth row of X) by $X_{j:}$.

1. Compute standard regression coefficients for each feature j as:

$$w_j = \frac{X_{j:}^T Y}{\sqrt{X_{j:}^T X_{j:}}} \tag{1}$$

2. Reduce the data matrix X to include only those features whose coefficients exceed a threshold θ in absolute value.
3. Compute the first few principal components of the reduced data matrix.
4. Use the principal components calculated in step 3 in a regression model or a classification algorithm to predict the outcome.

4. Hilbert-Schmidt Independence Criterion

Gretton et al. [24] proposed an independence criterion in RKHSs. This measure, referred to as the Hilbert-Schmidt Independence Criterion (HSIC), measures the dependence between two random variables, $\mathcal{X}$ and $\mathcal{Y}$, by computing the Hilbert-Schmidt norm of the cross-covariance operator[3] associated with their RKHSs. The HSIC has been widely used in many practical applications such as feature selection [25], feature extraction [26], and clustering algorithms [27].

HSIC uses the fact that two random variables, $\mathcal{X}$ and $\mathcal{Y}$, are independent if and only if any bounded continuous function of the two random variables is uncorrelated. That is, HSIC deals with cross-covariance operators which map from one space to another. Let us define $\mathcal{F}$ as a separable RKHS containing all continuous bounded real-valued functions of x from $\mathcal{X}$ to $\mathbb{R}$. Likewise, let $\mathcal{G}$ be a separable RKHS containing all continuous bounded real-valued functions of y from $\mathcal{Y}$ to $\mathbb{R}$. Then the cross-covariance between elements of $\mathcal{F}$ and $\mathcal{G}$ is:

$$Cov(f(x), g(y)) = \mathbf{E}_{x,y}[f(x)g(y)] - \mathbf{E}_x[f(x)]\mathbf{E}_y[g(y)] \tag{2}$$

It can be shown that there exists a unique linear operator $C_{x,y} : \mathcal{G} \rightarrow \mathcal{F}$ mapping elements of $\mathcal{G}$ to the elements of $\mathcal{F}$ such that:

[3]In the terminology of functional analysis, an operator is a mapping which maps elements from one Hilbert space to elements of another.

$$\langle f, C_{x,y} g \rangle_{\mathcal{F}} = Cov(f(x), g(y)) \qquad \forall f \in \mathcal{F}, \forall g \in \mathcal{G} \tag{3}$$

According to [28] and [20], this operator can be defined as:

$$C_{x,y} := \mathbf{E}_{x,y}[(\Phi(x) - \mu_x) \otimes (\Psi(y) - \mu_y)] \tag{4}$$

where $\mu_x = \mathbf{E}[\Phi(x)]$, $\mu_y = \mathbf{E}[\Psi(y)]$, $\otimes$ is the tensor product and Φ and Ψ are the associated feature maps of $\mathcal{F}$ and $\mathcal{G}$, respectively. As with any operator, we may define a norm for the cross-covariance operator. Indeed, several norms have been defined in the literature. A particularly useful form is the Hilbert-Schmidt (HS) norm. For a linear operator $C : \mathcal{G} \to \mathcal{F}$, provided the sum converges, the HS norm is defined as:

$$\|C\|_{HS}^2 := \sum_{i,j} \langle C v_i, u_j \rangle_{\mathcal{F}}^2 \tag{5}$$

where u_j and v_i are orthogonal bases of $\mathcal{F}$ and $\mathcal{G}$, respectively.

The square of the Hilbert-Schmidt norm of the cross-covariance operator, or HSIC, is then used as a measure of the dependence of two random variables. It can be shown that $\|C_{xy}\|_{HS}^2 = 0$ if and only if x and y are independent as long as their RKHSs are universal. The HSIC can be expressed in terms of kernel functions via the following identity:

$$\begin{aligned} HSIC(P_{\mathcal{X},\mathcal{Y}}, \mathcal{F}, \mathcal{G}) \;&=\; \mathbf{E}_{x,x',y,y'}[k(x,x')l(y,y')] + \mathbf{E}_{x,x'}[k(x,x')]\mathbf{E}_{y,y'}[l(y,y')] \\ &\quad -2\mathbf{E}_{x,y}[\mathbf{E}_{x'}[k(x,x')]\mathbf{E}_{y'}[l(y,y')]] \end{aligned} \tag{6}$$

where k and l are the associated kernels of $\mathcal{F}$ and $\mathcal{G}$, respectively. Here $\mathbf{E}_{x,x',y,y'}$ stands for the expectation over independent pairs of (x,y) and (x',y') drawn from $P_{\mathcal{X},\mathcal{Y}}$.

4.1. Empirical HSIC

In order to make HSIC a practical criterion for testing independence, it has to be approximated given a finite number of observations. Let $\mathcal{Z} :=$

$\{(\mathbf{x}_1, \mathbf{y}_1), \cdots, (\mathbf{x}_n, \mathbf{y}_n)\} \subseteq \mathcal{X} \times \mathcal{Y}$ be a series of n independent observations drawn from $P_{\mathcal{X},\mathcal{Y}}$. An empirical estimate of HSIC is:

$$HSIC(\mathcal{Z}, \mathcal{F}, \mathcal{G}) := (n-1)^{-2}\mathbf{tr}(KHLH) \tag{7}$$

where $H, K, L \in \mathbb{R}^{n \times n}, K_{ij} := k(\mathbf{x}_i, \mathbf{x}_j), L_{ij} := l(\mathbf{y}_i, \mathbf{y}_j)$, and $H_{ij} := I - n^{-1}\mathbf{e}\mathbf{e}^T$ (the centering matrix)[4]. Based on this result, we can conclude that in order to maximize the dependence between two kernels we need to increase the value of the empirical estimate, i.e., $\mathbf{tr}(KHLH)$. It is interesting to note that if one of the kernel matrices K or L is already centered, say L, then $HLH = L$ and thus we may simply use the objective function $\mathbf{tr}(KL)$ which no longer includes the centering matrix H. Similarly, if $HKH = K$ we may rewrite $\mathbf{tr}(KHLH) = \mathbf{tr}(HKHL)$ and arrive at identical results.

5. Supervised Principal Component Analysis

Assume we have a set of n data points $\{\mathbf{x}_i\}_{i=1}^n$ each consisting of p features, stacked in the $p \times n$ matrix X. In addition, assume that Y is the $\ell \times n$ matrix of outcome measurements. We address the problem of finding the subspace $U^T X$ such that the dependency between the projected data $U^T X$ and the outcome Y is maximized. In order to measure the dependence between $U^T X$ and the output variable Y, we use the Hilbert-Schmidt Independence Criterion.

We need to maximize $\mathbf{tr}(HKHL)$ where K is a kernel of $U^T X$ (e.g. $X^T U U^T X$), L is a kernel of Y (e.g. $Y^T Y$), and $H_{ij} := I - n^{-1}\mathbf{e}\mathbf{e}^T$. This objective can be formulated as

$$\begin{aligned} \mathbf{tr}(HKHL) &= \mathbf{tr}(HX^T U U^T XHL) \\ &= \mathbf{tr}(U^T XHLHX^T U) \end{aligned} \tag{8}$$

We are looking for an orthogonal transformation matrix U which maps data points to a space where the features are uncorrelated. Thus, our optimization problem is constrained and becomes:

[4] "$\mathbf{e}$" is a vector of all ones.

$$\begin{array}{ll} \arg\max_{U} & \mathbf{tr}(U^T XHLHX^T U) \\ \textit{subject to} & U^T U = I \end{array} \tag{9}$$

It is evident this optimization problem can be solved in closed-form. If the symmetric and real matrix $Q = XHLHX^T$ has eigenvalues $\lambda_1 \leq \ldots \leq \lambda_p$ and corresponding eigenvectors $v_1, \ldots, v_p$, then, the maximum value of the cost function satisfying the constraint is $\lambda_p + \lambda_{p-1} + \ldots + \lambda_{p-d+1}$ and the optimal solution is $U = [v_p, v_{p-1}, \ldots, v_{p-d+1}]$ [29]. Here, d denotes the dimension of the output space $\mathcal{S}$.

The Supervised PCA procedure is summarized in Algorithm 1.

```
Alg. 1 Supervised PCA
Input: training data matrix, X, testing data example, x, kernel matrix of target variable, L, and training data size, n.
Onput: Dimension reduced training and testing data, Z and z.
1: $H \leftarrow I - n^{-1}\mathbf{e}\mathbf{e}^T$
2: $Q \leftarrow XHLHX^T$
3: Compute basis: $U \leftarrow$ eigenvectors of $Q$ corresponding to the top $d$ eigenvalues.
4: Encode training data: $Z \leftarrow U^T X$
5: Encode test example: $\mathbf{z} \leftarrow U^T \mathbf{x}$
```

5.1. Connection to Principal Component Analysis

Consider the unsupervised problem, where the response variable is unknown; then L can be set equal to an identity matrix. The identity matrix is a kernel which only captures the similarity between a point and itself. Maximizing the dependence between K and the identity matrix corresponds to retaining the maximal diversity across all observations, and is thus equivalent to classical PCA.

It is easy to verify that when L is equal to the identity matrix I, then $XHLHX^T$ becomes the covariance matrix of X:

$$XHIHX^T = (XH)(XH)^T \tag{10}$$

$$
\begin{aligned}
&= [X(I - n^{-1}\mathbf{e}\mathbf{e}^T)][X(I - n^{-1}\mathbf{e}\mathbf{e}^T)]^T \\
&= (X - \mu_x)(X - \mu_x)^T \\
&= Cov(X)
\end{aligned}
$$

where $\mathbf{ee}^T$ is a square matrix of size n with all elements being equal to *one*, and μ_x denotes the mean of data presented in X.

This means that finding the top d eigenvalues of the covariance matrix of the data is equivalent to finding the top d eigenvalues of $XHIHX^T$ and consequently maximizing $\mathbf{tr}(U^T XHIHX^T U)$. In other words, PCA is a special form of the more general framework proposed by Supervised PCA; we set $L = I$ which means that we retain the maximal diversity between observations.

5.2. Dual Supervised Principal Component Analysis

In many cases the dimensionality p of the $p \times n$ data matrix X is much larger than the number of data points (i.e. $p >> n$). Such problems are of particular interest, especially in genomics and computational biology. In this case applying the direct form of Supervised PCA proposed in Algorithm 1 is impractical because of the need to calculate the eigenvectors of the very large $p \times p$ matrix Q. We would prefer a run time that depends only on the number of training examples n, or it has at least a reduced dependence on p.

Note that both $Q = XHLHX^T$ and L are positive semidefinite matrices. Thus, we can apply the following definitions:

$$
\begin{aligned}
Q &= XHLHX^T = \Psi\Psi^T \qquad (11) \\
L &= \Delta^T\Delta \\
\Rightarrow \Psi &= XH\Delta^T
\end{aligned}
$$

Clearly the solution for U can be expressed as the singular value decomposition (SVD) of Ψ:

$$
\Psi = U\Sigma V^T \qquad (12)
$$

since the columns of U in the SVD contain the eigenvectors of $\Psi\Psi^T$.

The singular value decomposition also allows us to formulate the principle component algorithm entirely in terms of dot products between data points, which limits the direct dependence on the original dimensionality n.

Note that in the SVD factorization, the eigenvectors in U that correspond to non-zero singular values in Σ (square roots of the eigenvalues) are in a one-to-one correspondence with the eigenvectors in V. Now suppose that we perform a dimensionality reduction on U; we keep only the first d eigenvectors corresponding to the top d non-zero singular values in Σ. These eigenvectors will still be in a one-to-one correspondence with the first d eigenvectors in V and we have $\Psi\hat{V} = \hat{U}\hat{\Sigma}$, where the dimensionality of these matrices are:

$$\Psi_{p\times n} \quad \hat{U}_{p\times d} \quad \hat{\Sigma}_{d\times d} \quad \hat{V}_{n\times d}.$$

Here, notation "ˆ" denotes the matrices of the reduced SVD. Now, $\hat{\Sigma}$ is square and invertible, because its diagonal has non-zero entries. Thus, the following conversion between the top d eigenvectors can be derived:

$$\hat{U} \;=\; \Psi\,\hat{V}\,\hat{\Sigma}^{-1} \tag{13}$$

Replacing all deployments of U in Algorithm 1 with $\Psi\hat{V}\hat{\Sigma}^{-1}$ generates the dual form of Supervised PCA which is summarized in Algorithm 2.

```
Alg. 2 Dual Supervised PCA
Input: training data matrix, X, testing data example, x, kernel matrix of target variable, L, training data size, n.
Onput: Dimension reduced training and testing data, Z and z .
1: Decompose $L$ such that $L = \Delta^T\Delta$.
2: $H \leftarrow I - n^{-1}\mathbf{e}\mathbf{e}^T$
3: $\Psi \leftarrow XH\Delta^T$
4: Compute basis:
    $V \leftarrow$ eigenvectors of $\Psi^T\Psi = \Delta H[X^TX]H\Delta^T$ corresponding to the top $d$ eigenvalues.
    $\Sigma \leftarrow$ diagonal matrix of square roots of the top $d$ eigenvalues of $\Psi^T\Psi$.
    $U \leftarrow \Psi V\Sigma^{-1}$
5: Encode training data: $Z \leftarrow U^TX = \Sigma^{-1}V^T\Delta H[X^TX]$
6: Encode test example: $\mathbf{z} \leftarrow U^T\mathbf{x} = \Sigma^{-1}V^T\Delta H[X^T\mathbf{x}]$
```

5.3. Kernel Supervised Principal Component Analysis

In many cases, nonlinear transformations of the data are required to successfully apply learning algorithms. One efficient method for doing this is to use a kernel that computes a similarity measure between any two data points. In this section, we show how to perform Supervised Principal Component Analysis in the feature space implied by a kernel, which allows our method to be extended to nonlinear mappings of the data.

5.3.1. Kernel Supervised PCA based on the Dual formulation

Consider a feature space $\mathcal{H}$ such that:

$$\begin{aligned} \Phi : x &\rightarrow \mathcal{H} \\ x &\mapsto \Phi(x) \end{aligned} \tag{14}$$

This allows us to formulate the kernel supervised PCA objective as follows:

$$\begin{aligned} \arg\max_{U} \quad & \mathbf{tr}(U^T \Phi(X) HLH \Phi(X)^T U) \\ \textit{subject to} \quad & U^T U = I \end{aligned} \tag{15}$$

By the same argument used for Supervised PCA, the solution can be found by SVD:

$$\Psi = \Phi(X) H \Delta^T = U \Sigma V^T \tag{16}$$

where U contains the eigenvectors of $\Psi(X)\Psi(X)^T$.

Now assume that we have a kernel $K(\cdot,\cdot)$ that allows us to compute $K(x,y) = \Phi(x)^T\Phi(y)$. Given such a function, we can then compute the matrix $\Phi(X)^T\Phi(X) = K$ efficiently, without computing $\Phi(X)$ explicitly. Crucially, K is $n \times n$ here and does not depend on the dimensionality of the feature space. Therefore it can be computed with a run time that depends only on n. Also, note that Supervised PCA can be formulated entirely in terms of dot products between data points (Algorithm 2). Replacing the dot products in Algorithm 2 by the kernel function K (which is in fact equivalent to the inner product of a Hilbert space) yields the Kernel Supervised PCA algorithm.

5.3.2. Direct formulation of Kernel Supervised PCA

Kernel Supervised PCA can be formulated directly, without use of the dual formulation. The key idea is to express the transformation matrix U as a linear combination of the projected data points, $U = \Phi(X)\beta$, via representation theory [30]. Thus we can rewrite the objective function as:

$$\begin{aligned} \mathbf{tr}(U^T\Phi(X)HLH\Phi(X)^TU) &= \mathbf{tr}(\beta^T\Phi(X)^T\Phi(X)HLH\Phi(X)^T\Phi(X)\beta) \\ &= \mathbf{tr}(\beta^TKHLHK\beta) \qquad (17) \end{aligned}$$

with the constraint:

$$\begin{aligned} U^TU &= \beta^T\Phi(X)^T\Phi(X)\beta \qquad (18) \\ &= \beta^TK\beta \end{aligned}$$

where K is a kernel function. We have now expressed the objective function and the constraint in terms of inner products between data points, which can be computed via the kernel. The new optimization problem has the following form:

$$\begin{aligned} \arg\max_{U} \quad & \mathbf{tr}(\beta^TKHLHK\beta) \qquad (19) \\ \textit{subject to} \quad & \beta^TK\beta = I \end{aligned}$$

This is a generalized eigenvector problem. β can be computed as the top d generalized eigenvectors of $(KHLHK, K)$. Our Kernel Supervised PCA procedure is summarized in Algorithm 3.

Our experiments suggest that both variations of Kernel Supervised PCA produce very similar results in practice. However, the direct formulation (Algorithm 3) has slightly less computational complexity, as there is no need to decompose the kernel matrix L and compute the matrix Δ. In the rest of this paper, all experiments with Kernel Supervised PCA are reported using Algorithm 3.

```
Alg. 3 Kernel Supervised PCA
Input: Kernel matrix of training data, K,kernel matrix of testing data, K_{test}, kernel matrix of target variable, L, testing data example, x, training data size, n.
Onput: Dimension reduced training and testing data, Z and z.
1: $H \leftarrow I - n^{-1}\mathbf{e}\mathbf{e}^T$
2: $Q \leftarrow KHLHK$
3: Compute basis: $\beta \leftarrow$ generalized eigenvectors of (Q, K) corresponding to the top $d$ eigenvalues.
4: Encode training data: $Z \leftarrow \beta^T[\Phi(X)^T\Phi(X)] = \beta^T K$
5: Encode test example: $\mathbf{z} \leftarrow \beta^T[\Phi(X)^T\Phi(\mathbf{x})] = \beta^T K_{test}$
```

6. Experimental Results

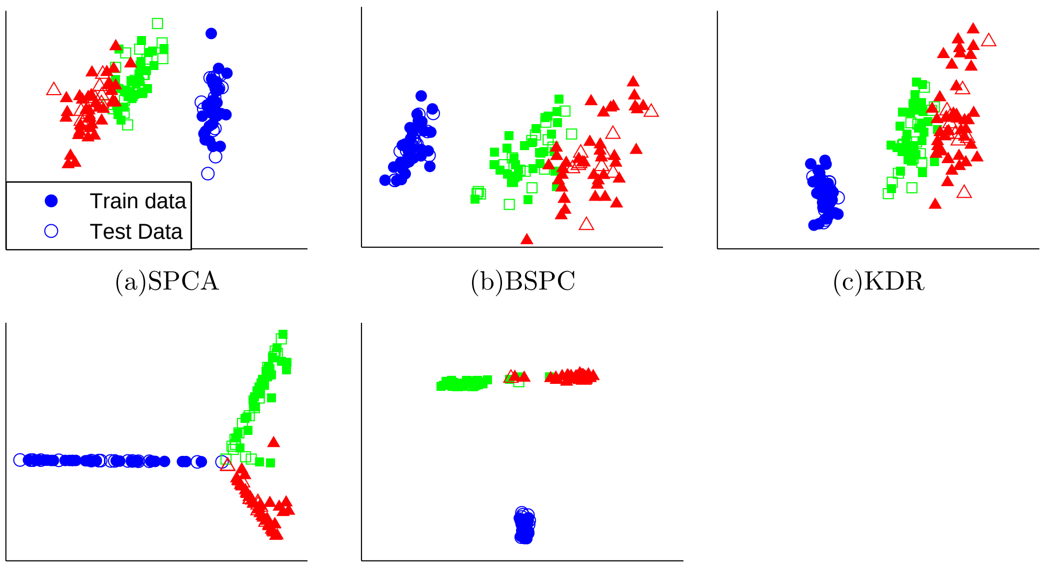
In this section we study the effectiveness of the proposed Supervised PCA method in comparison with some of the state-of-the-art algorithms for supervised dimensionality reduction. The performance of the methods is evaluated on a number of visualization, classification, and regression problems. In all of the following experiments, the input features are first normalized to the range [0, 1], and the kernel parameter is obtained using 10-fold cross-validation.

6.1. Visualization

First, the applicability of our method on a data visualization task is examined. We evaluate the performance of Supervised PCA and Kernel Supervised PCA and compare them with Bair's SPC, KDR and mKDR (as described in Section 3). In the case of KDR and mKDR, the MATLAB source code provided us by the authors was used.

When the response variable is provided in the form of labels, we apply a delta kernel $L(y, y') = \delta(y, y')$ to compute L. For example, in a 2-class problem, if there are 5 data points such that the first 3 data points belong to class 1 and the fourth and the fifth data points are from class 2, $L(y, y')$ can be formed as follows:

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$



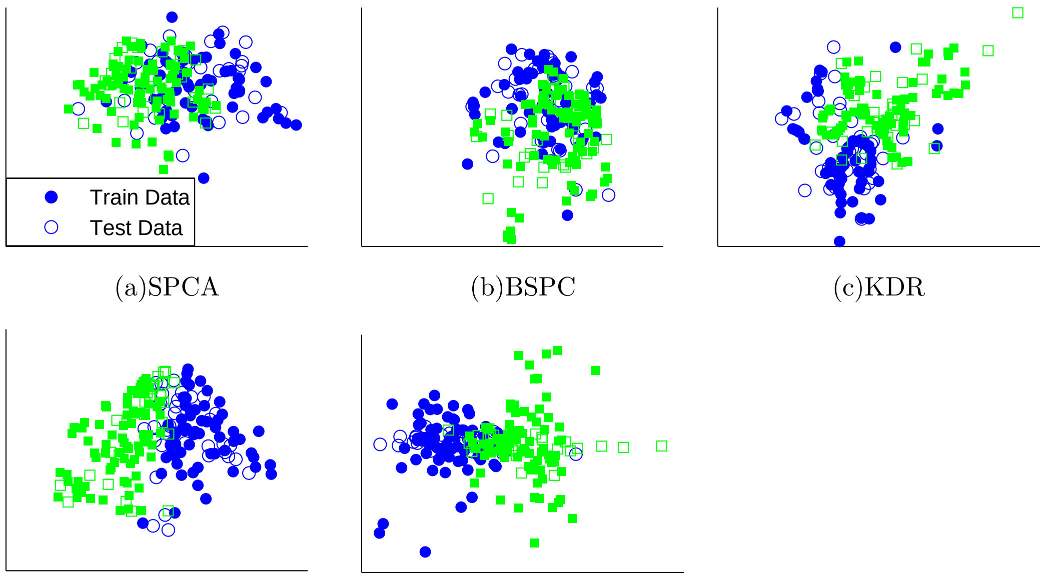
(a)SPCA (b)BSPC (c)KDR

(d)KSPCA (e)mKDR

Figure 1: The 2-dimensional projection of Iris data set, as produced by Supervised PCA (SPCA), Bair's SPC (BSPC), KDR, Kernel Supervised PCA (KSPCA), and mKDR, respectively. Different symbols are used to denote data instances from different classes. Solid symbols denote the training set, whereas 'hollow' symbols denote the test set.

This kernel leads the algorithm toward an embedding where instances from the same class are grouped tighter. Besides, an RBF kernel has been used as the data kernel in our nonlinear Supervised PCA.

We first present the effectiveness of our method on some real-world data sets, and later we explore its capability to detect data nonlinearity using some artificial data. A brief description of these data sets is given in Table 1. For the visualization experiments, we performed a random selection of 70% training set and 30% testing set. Note that in the case of mKDR, due to the lack of a straightforward extension for new test data, we assume that input features of both training and test sets are provided at the learning time. But the value of response variable is only provided for training data. This assumption holds for all of the visualization, classification and regression experiments with mKDR. In the case of real life data sets, we used the *Iris* and *Sonar* data sets available from the UCI machine learning repository [31]. The 2-dimensional projection results for the *Iris* data set using various methods are presented in Figure 1. Evidently, all of these methods demonstrate a



(a)SPCA (b)BSPC (c)KDR

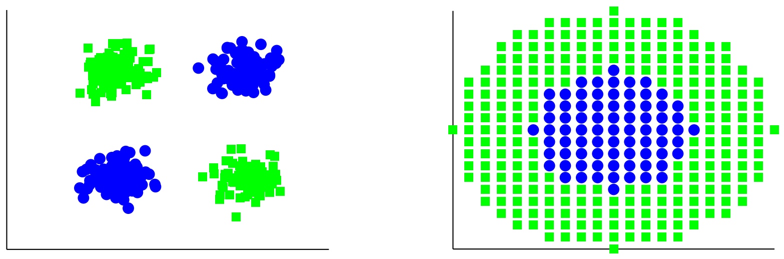
(d)KSPCA (e)mKDR

Figure 2: The 2-dimensional projection of Sonar data set, as produced by Supervised PCA (SPCA), Bair's SPC (BSPC), KDR, Kernel Supervised PCA (KSPCA), and mKDR, respectively. Different symbols are used to denote data instances from different classes. Solid symbols denote the training set, whereas 'hollow' symbols denote the test set.

reasonable separation between the three different classes. However, the projection produced by Kernel Supervised PCA is able to separate the classes shown by "□" and "△" symbols more clearly. To consider the linear methods, Supervised PCA generates a projection very close to the ones produced by KDR and Bair's SPC in which the data points at the class "◯" are separated perfectly. This is important to notice that the proposed Supervised PCA can be computed in closed-form and its computational complexity is much less than KDR. The reader is referred to Section 6.2 for further discussion.

Figure 2 shows the projection of the *Sonar* data set into a 2-dimensional space, as estimated by each method. Kernel supervised PCA yields a better embedding in this experiment as well and mKDR produces the next best separation. This is partly due to the fact that Kernel supervised PCA and mKDR provide a nonlinear projection while the other methods can only produce a linear embedding.

The poor performance of the linear methods on nonlinear problems be-



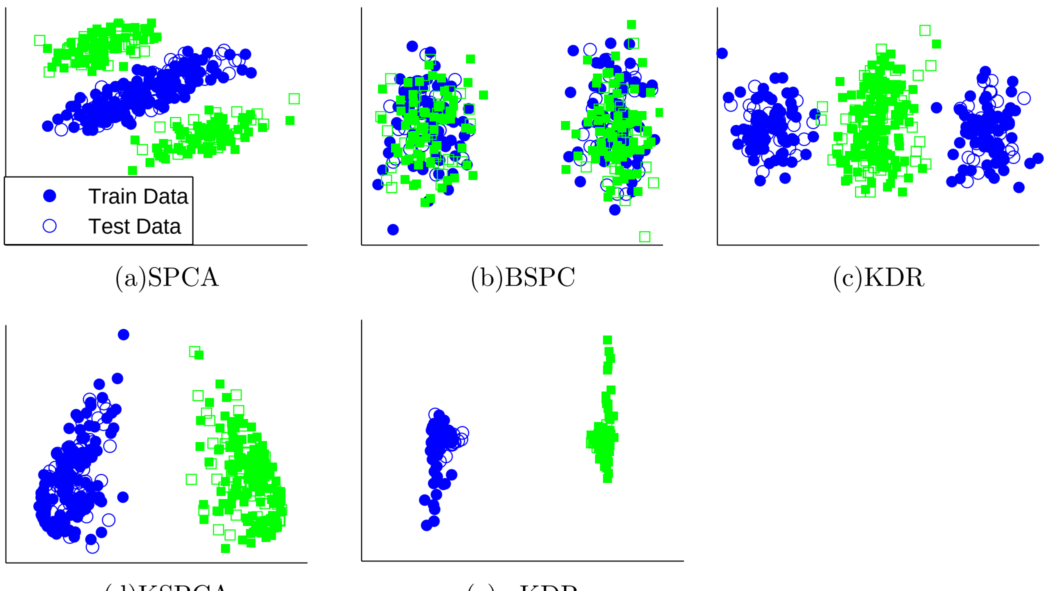
(a)Binary XOR (b)Concentric Rings

Figure 3: The original form of two artificial data sets used in data visualization experiments. Different symbols are used to denote data instances from different classes.

comes more evident in the following two experiments, in which we use the *Binary XOR* and *Concentric Rings* data sets. These two artificial data sets, depicted in Figure 3, are highly nonlinear. Figure 4 shows the 2-dimensional projection produced by the above methods on the *Binary XOR* data added with a 1-dimensional Gaussian noise attribute ($\sigma_{Noise} = 1$). The results indicate that Bair's SPC essentially fails at class separation, as opposed to Kernel Supervised PCA and mKDR which make the two classes linearly separable. Although KDR and Supervised PCA show better discrimination than Bair's SPC, they are also unable to unify the two clusters of both classes simultaneously. A similar experiment for the *Concentric Rings* data set is considered in Figure 5, and the nonlinearity is much more obvious. In this case, none of the linear methods could unfold the data and simply left the original structure of the data unchanged. It is worth noting that in all of these experiments, in addition to a good visualization of the training data, Supervised PCA and Kernel Supervised PCA were also able to provide good generalization to the testing data.

Most prominent dimensionality reduction algorithms, either supervised or unsupervised, provide an embedding only for the given training data, with no straightforward extension for new test points. In other words, they are merely used as a data visualization tool. Colored Maximum Unfolding (CMVU) [26] is one of the most effective techniques of this category.

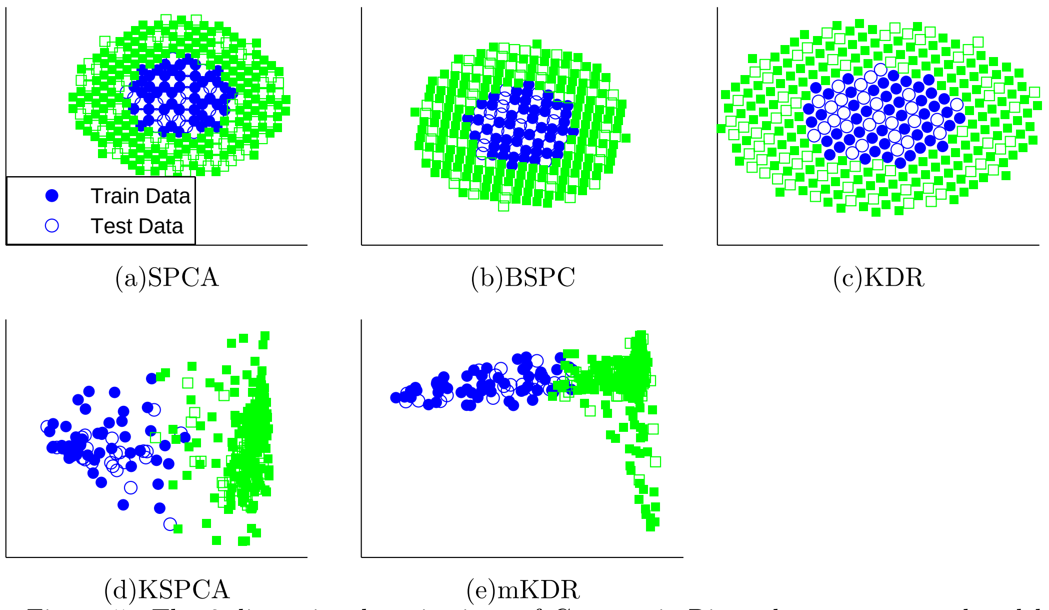
Figure 6 shows a final visualization experiment, in which a 1000-sample subset of the USPS hand-written digits data set is embedded into 2 dimensions by Kernel Supervised PCA and CMVU. Kernel Supervised PCA demonstrably provides more separability between the classes.



(a)SPCA (b)BSPC (c)KDR

(d)KSPCA (e)mKDR

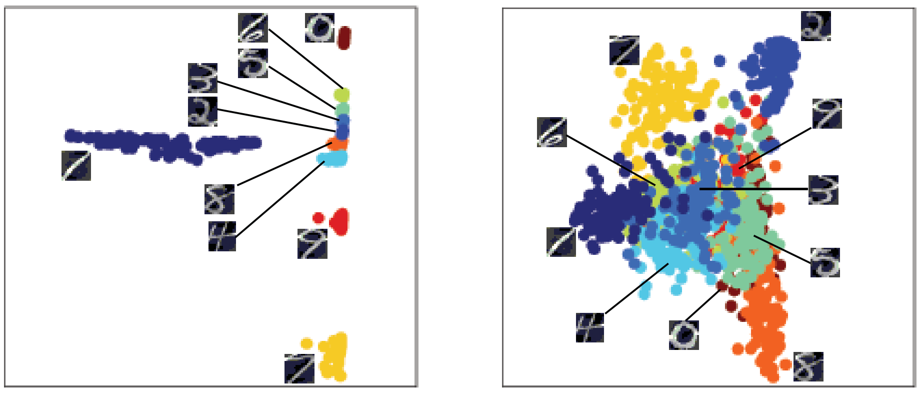
Figure 4: The 2-dimensional projections of Binary XOR data set added with a 1-dimensional Gaussian noise ($\sigma_{Noise} = 1$), as produced by Supervised PCA (SPCA), Bair's SPC (BSPC), KDR, Kernel Supervised PCA (KSPCA), and mKDR, respectively. Different symbols are used to denote data instances from different classes.



(a)SPCA (b)BSPC (c)KDR

(d)KSPCA (e)mKDR

Figure 5: The 2-dimensional projections of Concentric Rings data set, as produced by produced by Supervised PCA (SPCA), Bair's SPC (BSPC), KDR, Kernel Supervised PCA (KSPCA), and mKDR, respectively. Different symbols are used to denote data instances from different classes.



(a)KSPCA (b)CMVU

Figure 6: The 2-dimensional projections of USPS data set, as produced by Kernel Supervised PCA (KSPCA) and Colored Maximum Variance Unfolding (CMVU), respectively.

6.2. Classification

In this section we focus on classification problems and study the behavior of Supervised PCA and Kernel Supervised PCA in comparison with some other methods of supervised dimensionality reduction. We compare Supervised PCA with FDA, KDR, mKDR, CFML[5] and Bair's SPC. To the best of our knowledge, the above selection shows a well-known representative from each category of supervised dimensionality reduction techniques. In classification experiments, we performed 40 random splits of the data, taking 70% for training and 30% for testing. For each of the algorithms, we have calculated the transformation matrix using the training set, and after computing the low dimensional projection of the data, a one-nearest-neighbor classifier has been used to classify the test data. As with the visualization experiments, in the case of Supervised PCA and Kernel Supervised PCA a delta kernel $L(y, y') = \delta(y, y')$ was applied on the labels, and for Kernel Supervised PCA we used an RBF kernel for the data[6].

The data sets used in the first part of our comparative study are taken

[5]CFML has two versions. The one that is used here is the second version which uses the constraint $W^T M_S W = \mathbf{I}$.

[6]In the case of the labels' kernel matrix, we add an identity matrix of the same size to it to avoid the problem of rank deficiency.

Table 1: Description of data sets used in the visualization and classification experiments.

Data sets	No. of Data Points	No. of Dimensions	No. of Classes
Balance	625	4	3
Binary XOR	400	2	2
Colon Cancer	62	2000	2
Concentric Rings	313	2	2
Heart Disease	297	13	5
Ionosphere	351	34	2
Iris	150	4	3
Lymphoma	96	4026	2
Parkinsons	197	23	2
Sonar	208	60	2
SRBCT	83	2308	4
USPS	1000	256	10

from the UCI repository [31]. We conducted a number of experiments on the *Balance*, *Heart Disease*, *Ionosphere*, *Parkinsons* and *Sonar* data sets; their descriptions are presented in Table 1.

Figure 7 shows the testing data classification error rate along different projection dimensions. Note that since the solution of FDA is of rank $C-1$ (where C is the number of classes), this solution remains constant for any m-dimensional projection if $m \geq C-1$. As can be seen, Supervised PCA is significantly competitive with other linear methods; in 4 of 5 cases, it shows the lowest error rate. Furthermore, Kernel Supervised PCA outperforms all of the linear methods except in the cases of the *Sonar* and *Parkinsons* data, in which our Supervised PCA stays superior. In the case of nonlinear methods, Kernel Supervised PCA outperforms mKDR in all of the five cases. This is partly due to the fact that the optimization procedure of mKDR often results in a low rank transformation matrix which limits the efficiency of this method on classification problems. It is remarkable that in lower dimensional spaces (which are most suited for the task of data visualization), Kernel Supervised PCA performs significantly better than Supervised PCA. It can be seen that for higher dimensional data sets like *Sonar* and *Ionosphere*, FDA operates poorly due to its limitation on the maximum number of projection dimensions.

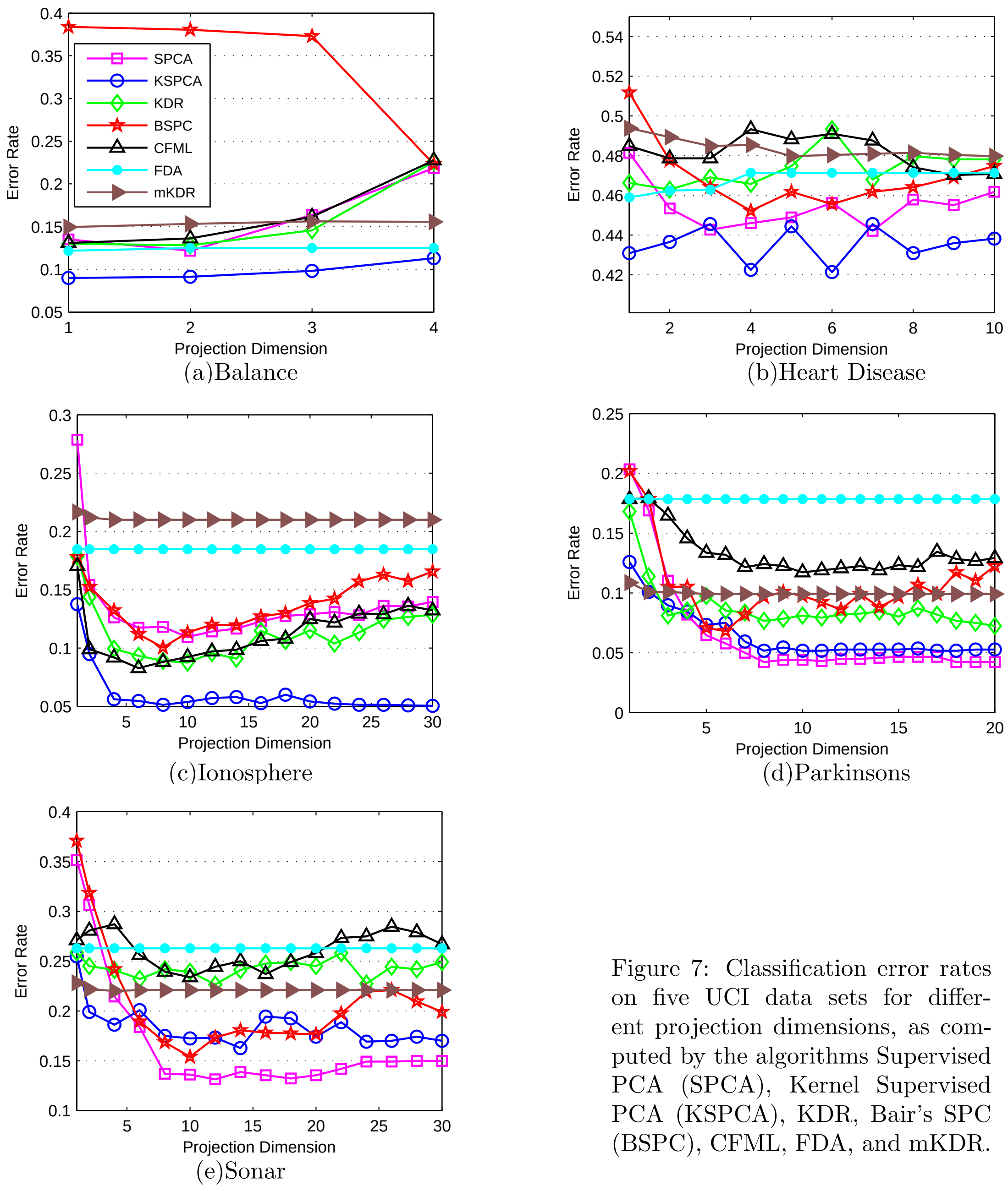


Figure 7: Classification error rates on five UCI data sets for different projection dimensions, as computed by the algorithms Supervised PCA (SPCA), Kernel Supervised PCA (KSPCA), KDR, Bair's SPC (BSPC), CFML, FDA, and mKDR.

Table 2: Average runtime of various algorithms in seconds.

Data sets	SPCA	KSPCA	BSPC	CFML	KDR	mKDR	FDA
Balance	0.012	0.480	0.009	5.216	454.105	0.679	0.001
Heart Disease	0.002	0.072	3.534	1.185	79.708	0.383	0.002
Ionosphere	0.007	0.146	0.259	1.961	349.853	0.978	0.007
Parkinsons	0.002	0.038	0.570	0.580	57.451	1.781	0.002
Sonar	0.015	0.067	1.435	0.996	156.428	1.199	0.084

The average runtimes of the algorithms is shown in Table 2.[7] More precisely, it is the amount of time each of these algorithms required in order to compute the projection matrix for the given data and label matrices. In the case of KDR, the time required to produce the projection matrix for a d-dimensional subspace depends on the value of d; so the reported time is the average runtime across the different values of d. All the other algorithms' runtimes are independent of d. In the case of Bair's SPC, since the threshold is approximated for each training set separately and it is all the supervision Bair's SPC performs on PCA, the time to find the appropriate threshold have been added to Bair's SPC runtime. Although KDR as a linear method performs well especially in low dimensional spaces (which is suitable for a data visualization task), its running time is dramatically larger than the other methods[8]. By comparison, Supervised PCA is computationally more efficient because it does not need any threshold tuning like Bair's SPC, nor does it have CFML's requirement on the identification of similar and dissimilar pairings of the data. This table illustrates that the computational complexity of Kernel Supervised PCA is much less than that of mKDR as a nonlinear method. Moreover, our proposed Kernel Supervised PCA is even faster than the linear methods CFML, Bair's SPC and KDR.

In the second part of our comparative study, we present the experimental results on a number of well-known DNA microarray dataset. We perform experiments on three data sets: Colon cancer [32], Lymphoma [33], and SR-

[7]All algorithms were implemented in MATLAB running on a system with Intel(R) Core(TM)2 CPU T7200 @ 2.00GHz and 1GB of RAM.

[8]We have used the KDR and mKDR codes provided us by the authors of the KDR and mKDR papers.

Table 3: Comparison of the different methods on three different microarray datasets. The methods are Bair's SPC (BSPC), Supervised PCA (SPCA), CFML, KDR, mKDR and Kernel Supervised PCA (KSPCA). Table lists the classification error rate for the test set predictions as well as the number of projection dimensions used by each method to achieve its best performance (the reported performance).

Method	Colon Cancer	Lymphoma	SRBCT
PCA	0.297±0.077 (9)	0.186±0.054 (9)	0.093±0.059 (8)
BSPC	0.242±0.086 (5)	0.102±0.065 (8)	0.132±0.072 (9)
SPCA	0.221±0.094 (2)	0.076±0.053 (7)	0.078±0.054 (5)
CFML	0.311±0.096 (1)	0.113±0.063 (4)	0.396±0.110 (3)
KDR	0.255±0.112 (5)	0.217±0.082 (8)	0.440±0.106 (9)
mKDR	0.329±0.092 (1)	0.228±0.072 (1)	0.306±0.082 (3)
KSPCA	0.237±0.100 (2)	0.081±0.060 (9)	0.092±0.060 (3)

BCT [34]; their descriptions are presented in Table 1. In the case of these data sets, since the dimensionality of the input space is much greater than the number of observations, we can benefit from the dual form of Supervised PCA. Due to this fact, the method FDA is not suitable for our comparative study in this part. Because applying FDA on data sets in which the number of samples is less than the number of input features, results in singular within-class covariance matrix and consequently sub-optimal performance. Instead, we examine the performance of PCA as an unsupervised method to understand the effect of supervsion. Table 3 shows the average classification error rate over 40 random splits of data taking 70% for training and 30% for testing. Error rate of test set computed for different projection dimensions (up to ten dimensions) and the reported result corresponds to the projection dimension in which each method achieves its best performance. Evidently, for all of these three data sets, the proposed Supervised PCA produces the lowest error rate and outperforms the other methods. Furthermore, the results indicate that the next best estimation is also produced by Kernel Supervised PCA in all of these three experiments. Superiority of Supervised PCA over its nonlinear version is in alignment with various reports in the literature of microarray gene analysis that claim better performance is realized with linear kernels [35]. It is worth to note that in all of these three data sets, Supervised PCA outperforms PCA and enhances class separability by considering the effect of response variable in the produced projection directions.

6.3. Regression

In the third and final experiment, we evaluate the performance of our proposed method on some regression problems. Most of conventional supervised dimensionality reduction methods do not address the problem of continuous target variables, and focuses instead on the discrete case. Therefore, these techniques are not applicable to the family of regression problems. FDA and CFML are in this category, and thus are unsuitable for comparison in this section.Instead, we compare Prinicipal Component Regression (PCR), KDR, Bair's SPC, MORP, Kernel MORP, Supervised PCA, Kernel Supervised PCA and manifold KDR at regression. In the case of Supervised PCA, MORP and their nonlinear versions, an RBF kernel is applied on the target variables. For each of these methods, after estimating the transformation matrix using the training set, we fit a linear regression model to the response variable Y and the dimension-reduced data Z.

Our regression experiments use three sets of synthetic data first introduced by Li et al. [36] in the context of dimensionality reduction in regression. In all of these experiments, it is assumed we have a set of n data points $\{\mathbf{x}_i\}_{i=1}^n$ each consisting of p features, stacked in the $p \times n$ matrix X. We construct a univariate response variable $\mathbf{y}$ which depends only on a specific subset of the features. We denote the jth feature (jth row of X) by $X_{j:}$, and the ith data point (ith column of X) by $X_{:i}$.

Regression A:

This regression is defined as:

$$\mathbf{y} = \frac{X_{1:}}{0.5 + (X_{2:} + 1.5)^2} + (1 + X_{2:})^2 + 0.5\epsilon, \tag{20}$$

where $X_{:i} \sim N(0, I_4)$ is a 4-dimensional input vector and $\epsilon \sim N(0,1)$ is normal additive Gaussian noise. Note that only the first two dimensions are relevant to the response variable $\mathbf{y}$.

Regression B:

The second regression definition is given by:

$$\mathbf{y} = \sin^2(\pi X_{2:} + 1) + 0.5\epsilon, \tag{21}$$

with predictor $X_{:i} \in \mathbb{R}^4$ uniformly distributed on the set

Table 4: Average Root Mean Square (RMS) error for the test set predictions on artificial regression data A, B, and C using PCR, KDR, Bair's SPC (BSPC), Supervised PCA (SPCA), MORP, Kernel Supervised PCA (KSPCA), Kernel MORP (KMORP), and mKDR.

Method	Regression(A)	Regression(B)	Regression(C)
PCR	2.2858±0.5363	0.6014±0.0738	0.8405±0.2705
KDR	1.6422±0.3709	0.5771±0.0769	0.8664±0.2704
BSPC	1.6420±0.3670	0.5766±0.0748	0.9348±0.2482
SPCA	1.6723±0.3800	0.5754±0.0763	0.8318 ±0.2734
MORP	2.0617±0.5188	0.6184±0.0673	0.8422±0.2697
KSPCA	1.5113±0.3263	0.5670±0.0701	0.8224±0.2767
KMORP	1.6468±0.5282	0.6151±0.0670	0.8465±0.2587
mKDR	1.6420±0.3416	0.5815±0.0757	0.8454±0.2741

$$[0,1]^4 \backslash \{\mathbf{x} \in \mathbb{R}^4 | \mathbf{x}_{j:} \leq 0.7 \ \forall j \in \{1,2,3,4\}\} \tag{22}$$

and ϵ is as defined in regression (A). Here, the true subspace is of dimension 1. Unlike the previous case, the distribution of X in regression (B) is not elliptical.

Regression C:

The final regression (C) is as follows:

$$\mathbf{y} = \frac{1}{2}(X_{1:})^2 \epsilon \tag{23}$$

where $X_{:i} \sim N(0, I_{10})$ and taking again $\epsilon \sim N(0,1)$ the independent noise. In this case the noise is multiplicative rather than additive, and the dimension of the true subspace is 1.

We draw 50 samples of size $n = 100$ out of each regression distribution, and in each set we examine 70% for training and 30% for testing. Table 4 shows the average Root Mean Square (RMS) error of estimating the test data response variable produced by each method. This table demonstrates that PCR as an unsupervised method is not good at prediction and generates

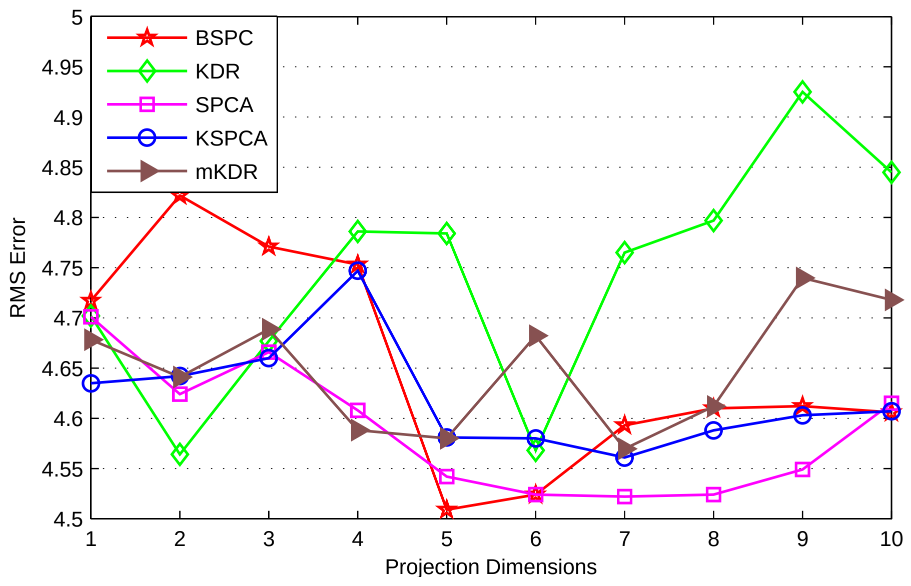


Figure 8: RMS error on DLBCL data set for different projection dimensions using Bair's SPC (BSPC), KDR, Supervised PCA (SPCA), Kernel Supervised PCA (KSPCA), and mKDR.

the highest error rate. For all of these three data sets, Kernel Supervised PCA produces the lowest error rate and outperforms the other methods. To consider only the linear methods, in regression (B) and (C) our Supervised PCA results in better performance than KDR, Bair's SPC and MORP. In the case of regression (A), among linear methods, Bair's SPC produces the best estimation.

As justified by the above artificial experiments, it would be more interesting to evaluate the algorithms with a real world data set. The one we study here is from a microarray data experiment, which consists of the survival times of patients with diffuse large B-cell lymphoma (DLBCL) [37]. This data set has been used previously in studies of dimension reduction methods [21]. It includes gene expression measurements of 7399 genes obtained from 240 patients; we consider 160 patients as a training set and 80 patients as the validation set. Since the dimensionality of the input space is much greater than the number of training samples ($7399 \gg 160$), we can benefit from the dual form of Supervised PCA. Figure 8 depicts the RMS error of testing data target variables, estimated along different projection dimensions.

The simple linear Supervised PCA has a better generalization performance than its nonlinear kernelized version. This result is in alignment with various reports in the literature of microarray gene analysis that claim superior performance is realized with linear kernels [35].

As shown in the figure, the best performance is achieved by Bair's SPC in dimension $d = 5$; Supervised PCA produces the next best estimation at $d = 7$. In the case of the nonlinear methods, the lowest prediction errors produced by mKDR and Kernel Supervised PCA are very close to each other and both approaches reach their best performance at $d = 7$. It is noticeable that although KDR does not generate the lowest error, it reaches its best performance in lower dimensional spaces ($d = 2$) in contrast with the other methods. This is in line with our observations from the classification experiments. It is important to note that, in addition to the good performance of our proposed technique, it is considerably more computationally efficient than the other approaches.

7. Discussion

The experimental results in the previous section demonstrate the performance of Supervised PCA on a variety of data sets. In this section, we discuss two other related approaches to PCA and the difference between these methods and Supervised PCA.

7.1. Partial Least Squares

The experimental results in the previous section demonstrate the performance of Supervised PCA on a variety of data sets. In this section, we discuss two other related approaches to PCA and the difference between these methods and Supervised PCA.

7.2. Partial Least Squares

Partial Least Squares (PLS) [38] is a family of techniques for analyzing the relationship between blocks of data by means of latent variables. Consider we have n observations from explanatory variable $\mathcal{X} \in \mathbb{R}^p$ and target variable $\mathcal{Y} \in \mathbb{R}^l$ stored in zero-mean matrices $X_{p\times n}$ and $Y_{l\times n}$. PLS decomposes these matrices into:

$$\begin{aligned} X &= PT^T + E \\ Y &= QW^T + F \end{aligned} \tag{24}$$

where T and W are $n \times d$ matrices of d extracted latent vectors, $P_{p\times d}$ and $Q_{l\times d}$ stands for loading matrices and $E_{n\times p}$ and $F_{n\times l}$ are matrices of residuals. PLS finds a set of weight vectors $U = [u_1, ..., u_d]$ and $C = [c_1, ..., c_d]$ such that:

$$[Cov(t_i, w_i)]^2 = \max_{u_i, c_i} \quad [Cov(X^T u_i, Y^T c_i)]^2 \tag{25}$$

where t_i and w_i are the i-th columns of matrices T and W respectively. PLS is an iterative procedure. In each iteration, after extraction of latent vectors t and w (through solving an eigenvalue problem), matrices X and Y are *deflated* by subtracting the information contained in the derived vectors t and w.

According to the above description, Supervised PCA and PLS are different in four major aspects. The first difference is in the definition of their objective functions. That is, PLS aims at maximizing the covariance between the two random variables while Supervised PCA maximizes the dependence between the twos. Therefore, PLS can only detect linear dependence between the two variables. In contrast, Supervised PCA is capable of capturing any kind of dependence (linear or nonlinear) with high probability. This difference will be discussed in more details in the following paragraphs. As the second dissimilarity, note that, unlike PLS which produces the score vectors for both predictors and target variable, Supervised PCA only extracts the projection of the input features. This comes from the fact that PLS originally designed as a regression model and not as a tool for dimensionality reduction. However, Supervised PCA, as a generalization for PCA, is essentially a tool for dimension reduction in which the projection of target variable is not beneficial. The third difference relates to the way of extracting the projection directions by each method. As mentioned earlier, PLS is an iterative procedure since it requires to deflate the random variables before deriving each columns of matrices T and U. On the contrary, Supervised PCA extracts all of the projection directions at once and has closed form solution. After discussing these major structural differences between PLS and Supervised PCA, we study one of the closest variants of PLS to the proposed approach and demonstrate their differences empirically.

Depending on the form of deflation, several variations of PLS have been defined. In most of the proposed variations, matrices X and Y are deflated separately. However, there is a variant of PLS called PLS-SB [39] (in accordance with [40]) which deflates the cross-product matrix XY^T instead of separate deflation of X and Y. This scheme leads to extraction of all latent vectors at once through solving an eignevalue problem of the form:

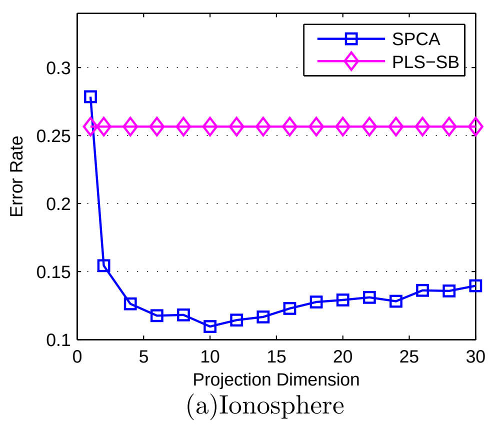
$$XY^TYX^TU = \lambda U \tag{26}$$

and calculating T and W as:

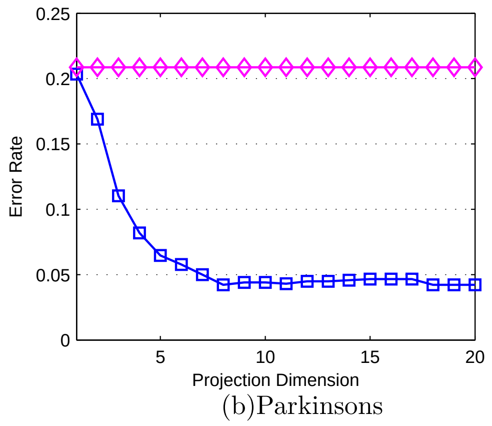
$$\begin{aligned} T &= X^TU \\ C &= YT \\ W &= Y^TC. \end{aligned} \tag{27}$$

Considering this formulation, PLS-SB is a special case of Supervised PCA, as a general framework, when a linear kernel applied on the target variable (i.e. $L = Y^TY$). However, there are problems in which applying other types of kernel is more beneficial. In this case, the general framework proposed by Supervised PCA gives a user the freedom to choose an appropriate type of kernel. In contrast, the definition of the objective function of PLS-SB does not provide us with such a possibility. For instance, consider the problem of data classification. In a $C-$class problem, the maximum number of projection dimensions that PLS-SB can extract is $C - 1$. In contrast, Supervised PCA can avoid this problem through applying a full-rank kernel matrix for the response variable Y. Figure 9 illustrates the effect of this limitation on the classification performance of PLS-SB compared with that of Supervised PCA. By adding an identity matrix to the linear kernel used in PLS-SB, a valid full-rank kernel is constructed and used as L in Supervised PCA.

In addition to the problem of rank deficiency, applying a linear kernel could affect the performance of PLS-SB in a more significant way. As discussed before, PLS can only detect linear dependence between the two variables while Supervised PCA is able to capture nonlinear dependence, too. That is, by applying a nonlinear kernel on the response variable Y, Supervised PCA, as a linear method, can produce a projection in which the nonlinear relationships between X and Y is captured. It is important to note that, Supervised PCA reflects the *nonlinear* properties of data through a *linear* projection which is completely interpretable. For example, consider



(a)Ionosphere



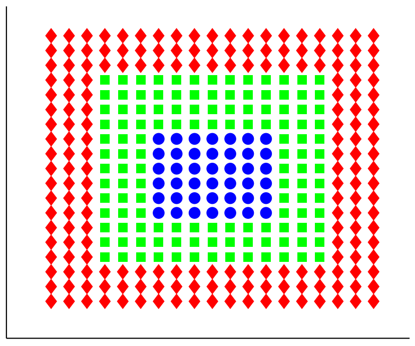
(b)Parkinsons

Figure 9: Classification error rates on two UCI data sets for different projection dimensions, as computed by the algorithms Supervised PCA (SPCA) and PLS-SB.

the regression problem (C) previously defined in Section 6.3 in which Y is nonlinearly dependent to X ($Y = \frac{1}{2}(X_{1:})^2\epsilon$). The average RMS error of predicting the response variable for testing data using PLS-SB and Supervised PCA is computed. In the case of Supervised PCA, an RBF kernel is applied on the dependent variable Y. The RMS errors produced by PLS-SB and Supervised PCA are 0.9011 ± 0.2504 and 0.8318 ± 0.2734, respectively. Thus, applying an RBF kernel on the response variable Y results in less prediction error compared with that of a linear kernel. In conclusion, unlike PLS-SB, whose application is limited to some specific problems, Supervised PCA is more general and can be applied on a variety of problems by appropriately choosing the kernel matrix L.

In order to model the nonlinear relationship between sets of data, kernelized variant of PLS was proposed firstly by Rosipal and Trejo [41]. The main idea is to reformulate the PLS procedure in terms of dot products between data points and then using the kernel trick. That is, the inner products are replaced with kernel matrices which contain the inner products of a nonlinear transformation of data. This procedure can be applied on different variations of PLS. For PLS-SB, if we multiply both sides of (26) by X^T and merge the last two steps of (27), we have [42]:

$$X^TXY^TYT = \lambda T \tag{28}$$



Concentric Rectangles

Figure 10: The original form of the artificial data set used in data visualization experiment. Different symbols are used to denote data instances from different classes.

$$W = Y^T Y T$$

By replacing the inner products matrices $X^T X$ and $Y^T Y$ with the centered kernel matrices K_x and K_y respectively, the nonlinear variant of PLS-SB is formulated as:

$$\begin{aligned} K_x K_y T &= \lambda T \\ W &= K_y T \end{aligned} \tag{29}$$

There is a key difference between the kernelized variants of PLS-SB and Supervised PCA. Kernel Supervised PCA solves the eigenvalue problem in order to extract the projection directions (i.e. U). However, in Kernel PLS-SB, the produced eigenvectors are the embedding directions (i.e. T). Therefore, In Kernel Supervised PCA, the projection directions are orthonormal (i.e. $U^T U = I$) while in Kernel PLS-SB, the embedding coordinates are orthonormal (i.e. $T^T T = I$). An immediate consequence of this fact is that, all of the extracted feature vectors using PLS-SB have unit variance. Thus, it is not suitable for applications in which the difference in the variance of the directions is important. On the other hand, since Kernel PLS-SB directly extracts the score vectors T, the transformation matrix U is not available.

Therefore, the way of projecting testing data should be changed and computed based on the matrix of latent vectors T (and not the weight matrix U). The formulation of Kernel PLS-SB for projecting the test set is as follows:

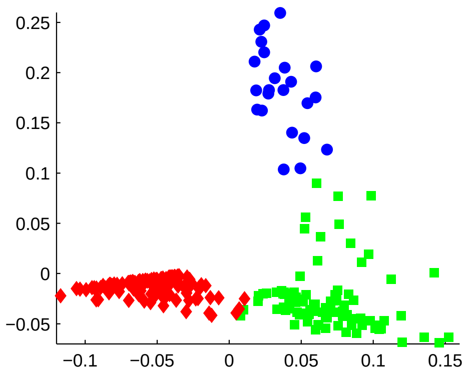
$$T_{test} = K_{test}W(T^T K_x W)^{-}1 \tag{30}$$

where K_{test} is the kernel matrix of testing data. In order to understand the effect of this difference emprically, the performance of Kernel PLS-SB and Kernel Supervised PCA is examined on visualizing an artificial data set shown in Figure 10. The data is firstly added with a 1-dimensional Gaussian noise attribute ($\sigma_{Noise} = 1$) and then its 2-dimensional projection is computed using Kernel Supervised PCA and Kernel PLS-SB. As the first row of Figure 11 indicates, the nonlinear version of PLS-SB provides a clearer separation. However, the projection of testing data in the second row, illustrates that Kernel Supervised PCA has better generalization in comparison with Kernel PLS-SB. This observation is supported by our experimental results on classification performance of Kernel Supervised PCA and Kernel PLS supports this observation. Figure 12 shows the testing data classification error rate along different projection directions using these two methods on five UCI data sets[9]. It is evident that Kernel Supervised PCA outperforms Kernel PLS-SB in all of the five cases. In addition, the kernelized variant of Supervised PCA, compared with PLS-SB, reaches its best performance in lower dimensional spaces. This observation is more obvious for higher dimensional data sets like *Parkinsons*, *Ionosphere* and *Sonar*.

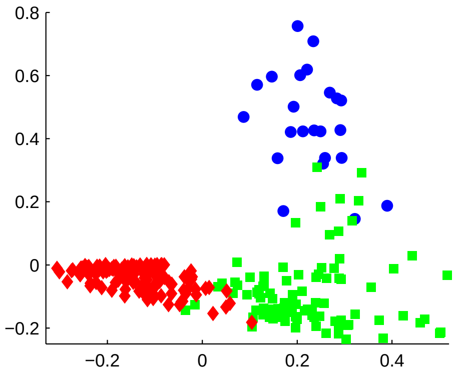
7.3. Canonical Correlation Analysis

Similar to PLS, Canonical Correlation Analysis (CCA) [43] is a technique for modeling the association between two variables. CCA searches for basis vectors for two blocks of variables such that the projection of variables onto these basis vectors are maximally correlated. Consider we have n observations from explanatory variable $\mathcal{X} \in \mathbb{R}^p$ and target variable $\mathcal{Y} \in \mathbb{R}^l$ stored in zero-mean matrices $X_{p\times n}$ and $Y_{l\times n}$. CCA seeks for linear transformation matrices $W_x \in \mathbb{R}^d$ and $W_y \in \mathbb{R}^q$ such that the correlation coefficient

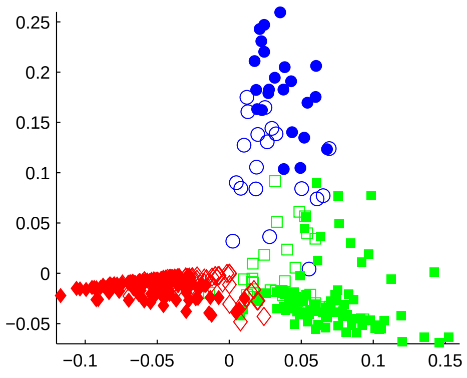
[9]The setup of the experiment is the same as the classification experiments in Section 6.2



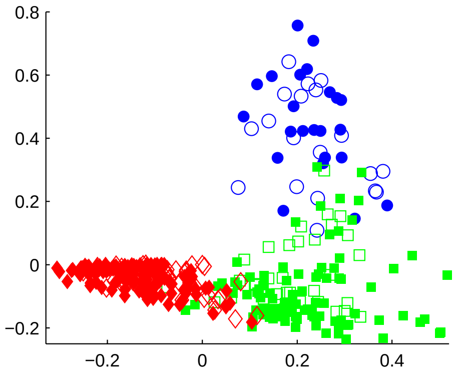
(a)KPLS-SB



(b)KSPCA

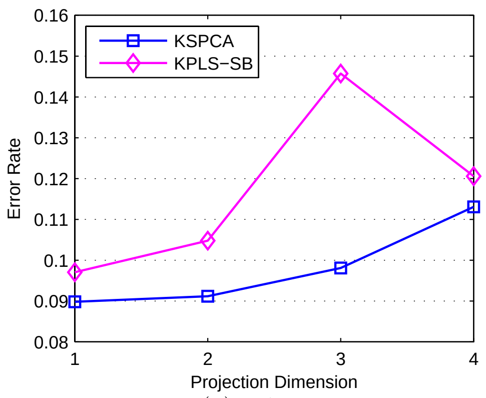


(c)KPLS-SB

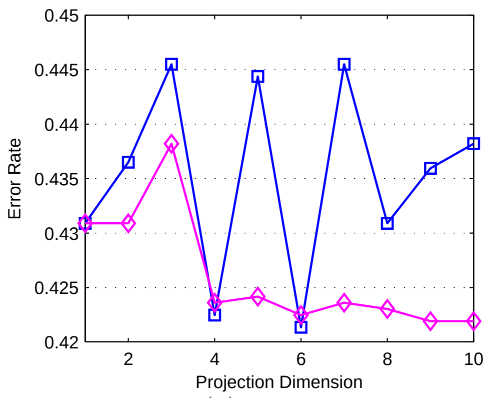


(d)KSPCA

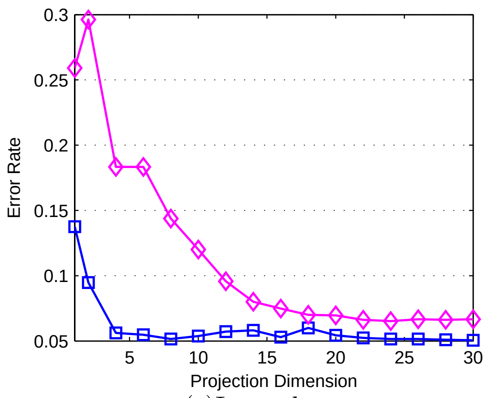
Figure 11: The 2-dimensional projection of Concentric Rectangles data set, as produced by Supervised Kernel Supervised PCA (KSPCA) and Kernel PLS-SB (KPLS-SB). Different symbols are used to denote data instances from different classes. Top panels show the projection of training set and bottom panels show the projection of testing and training data together. Solid symbols denote the training set, whereas 'hollow' symbols denote the test set.



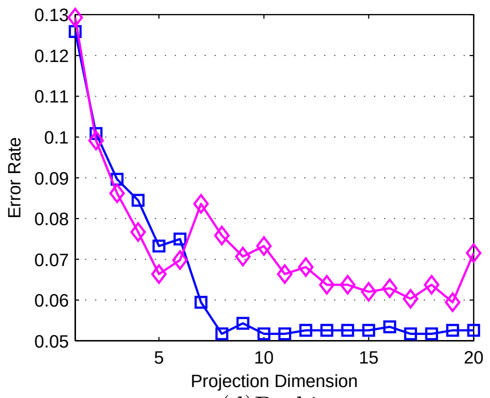
(a)Balance



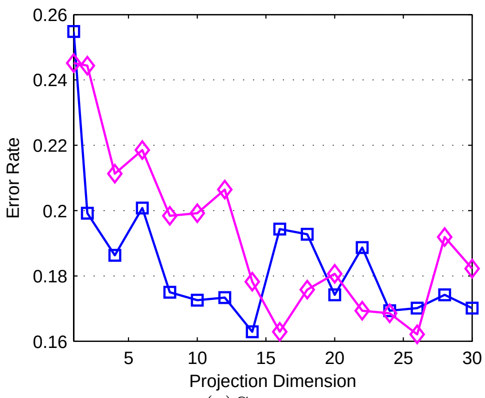
(b)Heart Disease



(c)Ionosphere



(d)Parkinsons



(e)Sonar

Figure 12: Classification error rates on five UCI data sets for different projection dimensions, as computed by the algorithms Kernel Supervised PCA (KSPCA) and Kernel PLS-SB (KPLS-SB).

$$\rho = \frac{Cov({W_x}^T X, {W_y}^T Y)}{\sqrt{Var({W_x}^T X) Var({W_y}^T Y)}} \tag{31}$$
$$= \frac{{W_x}^T X Y^T W_y}{\sqrt{({W_x}^T X X^T W_x)({W_y}^T Y Y^T W_y)}}$$

is maximized. Maximizing ρ is equivalent to optimizing the following objective function:

$$\begin{aligned} &\arg\max_{W_x, W_y} \quad {W_x}^T X Y^T W_y \\ &\textit{subject to}: \\ &\qquad {W_x}^T X X^T W_x = I \\ &\qquad {W_y}^T Y Y^T W_y = I. \end{aligned} \tag{32}$$

It can be shown that the optimum value of W_x for the above objective function is obtained by solving the generalized eigenvalue problem of the form [44]:

$$XY^T(YY^T)^{-1}YX^TW_x = \lambda^2 XX^TW_x. \tag{33}$$

After estimating columns of W_x as the top d eigenvectors of problem (33), the corresponding W_y can be obtained as $W_y = \frac{(YY^T)^{-1}YX^TW_x}{\lambda}$.

Considering the above formulation, Supervised PCA has two key advantages over CCA. The first one comes from the fact that CCA optimization procedure involves computation of two inverses, that of YY^T and that of XX^T. Thus, it is only applicable to problems in which the number of features does not exceed the number of observations. In contrast, Supervised PCA avoids any costly matrix inversion while its dual formulation significantly reduces the computational complexity of problems in which the number of predictors greatly exceeds the number of observations. As the second difference, note that, in discrimination, CCA and FDA performs identically.

That is, the directions given by CCA (using the dummy matrix Y for group membership) are equivalent to Fisher's discrimination directions [45]. Therefore, similar to FDA, CCA suffers from the problem of rank deficiency. Our experimental results in Section 6.2 previously showed the drastic effect of this limitation on the classification performance of FDA (CCA).

CCA may not extract useful features of data when the correlation exists in some nonlinear relationships. Kernel CCA [46] addresses this problem by first projecting the data into a higher dimensional feature space and then applying CCA in this new feature space. Let the kernel matrices K_x and K_y contain the inner products of the new projection of data $\Phi(X)$ and $\Psi(Y)$, respectively. The weights W_x and W_y can be rewritten as the linear combination of the projected data, $W_x = \Phi(X)\alpha$ and $W_y = \Psi(Y)\beta$. Substitution in equation (32) and replacing the inner products as $K_x = \Phi(X)^T\Phi(X)$ and $K_y = \Psi(Y)^T\Psi(Y)$ results in the following:

$$\begin{aligned} \arg\max_{\alpha,\beta} \quad & \alpha^T K_x K_y \beta \\ \mathit{subject\ to}: & \\ & \alpha^T {K_x}^2 \alpha = I \\ & \beta^T {K_y}^2 \beta = I \end{aligned} \tag{34}$$

It is important to note that when the kernel matrices are invertible, the naive kernelization of CCA gives trivial solution $|\rho = 1|$ and does not provide useful information. In order to avoid this problem the constraints in (34) should be regularized. Maximizing (34) with the regularized constraints leads to solving the generalized eigenvalue problem of the form:

$$\Rightarrow K_y(K_y + \kappa I)^{-1} K_x \alpha \quad = \quad \lambda^2 (K_x + \kappa I)\alpha \tag{35}$$

In contrast with the Kernel CCA, none of the two proposed algorithms for Kernel Supervised PCA suffer from overfitting problem and consequently no extra regularization is required. In conclusion, maximizing the dependency using HSIC leads to a more general and well-posed objective function in comparison with maximizing the correlation coefficient ρ.

8. Conclusion

This paper presented Supervised PCA, a new approach to supervised dimensionality reduction that is based on extracting the principal components of the data that have maximal dependence on the target variable. We demonstrated that conventional PCA is a special case of this general framework. Supervised PCA considers the quantitative value of the target variable, and is thus applicable to both classification and regression problems. It solves for a full-rank matrix with a closed-form solution. Therefore, it is computationally efficient without imposing any limitation on the dimensionality of the output feature space. We have also derived a dual form of Supervised PCA which reduces the computational complexity of the problems with the input space dimension much larger than the number of samples. Moreover, by kernelizing Supervised PCA, we have extended our method to efficiently model the nonlinear variability of the data and to perform nonlinear dimensionality reduction.

Our experiments on a variety of visualization, classification and regression problems illustrate the efficiency of the proposed method and show that in many cases, Supervised PCA outperforms the existing state-of-the-art techniques both in accuracy and computational efficiency.

This study can be extended in different directions. Supervised PCA focuses on producing a projection with maximum dependence on the response variable and does not consider local structure of the data. Consequently, it cannot faithfully extract the intrinsic dimensionality of the data. This is a problem which could be addressed in future research on Supervised PCA. Additionally, this general framework could be investigated to derive supervised versions of many other conventional unsupervised techniques.

References

[1] I. T. Jolliffe, Principal Component Analysis, Springer-Verlag, New York, 1986.

[2] R. Bellman, Adaptive control process: A guided tour, Princeton. University Press.

[3] R. A. Fisher, The use of multiple measurements in taxonomic problems, Annals Eugen. 7 (1936) 179–188.

[4] E. P. Xing, A. Y. Ng, M. I. Jordan, S. Russell, Distance metric learning with application to clustering with side-information, Advances in Neural Information Processing Systems (NIPS) 15 (2002) 505–512.

[5] M. Bilenko, S. Basu, R. J. Mooney, Integrating constraints and metric learning in semi-supervised clustering, ICML 69 (2004) 11.

[6] H. Chang, D.-Y. Yeung, Locally linear metric adaptation for semi-supervised clustering, ICML 69 (2004) 153–160.

[7] H. Chang, D.-Y. Yeung, Locally linear metric adaptation with application to semi-supervised clustering and image retrieval, Pattern Recognition 39 (7) (2006) 1253–1264.

[8] D.-Y. Yeung, H. Chang, Extending the relevant component analysis algorithm for metric learning using both positive and negative equivalence constraints, Pattern Recognition 39 (5) (2006) 1007–1010.

[9] S. Basu, M. Bilenko, R. J. Mooney, A probabilistic framework for semi-supervised clustering, KDD (2004) 59–68.

[10] K. Q. Weinberger, J. Blitzer, L. K. Saul, Distance metric learning for large margin nearest neighbor classification, Advances in Neural Information Processing Systems 18 (2006) 1473–1480.

[11] A. Globerson, S. T. Roweis, Metric learning by collapsing classes, Advances in Neural Information Processing Systems 18 (2006) 451–458.

[12] B.Alipanahi, M. Biggs, A. Ghodsi, Distance metric learning versus fisher discriminant analysis, Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (2008) 598–603.

[13] K. Li, Sliced inverse regression for dimension reduction (with discussion), Journal of the American Statistical Association 86 (1991) 316–342.

[14] R. D. Cook, S. Weisberg, Discussion of li (1991), Journal of the American Statistical Association 86 (1991) 328–332.

[15] K. Li, On principal hessian directions for data visualization and dimension reduction: Another application of stein's lemma, Journal of the American Statistical Association 87 (1992) 1025–1039.

[16] A. M. Samarov, Exploring regression structure using nonparametric functional estimation, Journal of the American Statistical Association 88 (1993) 836–847.

[17] R. D. Cook, X. Yin, Dimension reduction and visualization in discriminant analysis (with discussion), Australian & New-Zealand Journal of Statistics 43 (2001) 147–199.

[18] M. Hristache, A. Juditsky, J. Polzehl, V. Spokoiny, Structure adaptive approach for dimension reduction, The Annals of Statistics 29 (2001) 1537–1566.

[19] K. Torkkola, Feature extraction by non-parametric mutual information maximization, Journal of Machine Learning Research 3 (2003) 1415–1438.

[20] K. Fukumizu, F. R. Bach, M. I. Jordan, Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces, Journal of Machine Learning Research 5 (2004) 73–99.

[21] E. Bair, T. Hastie, D. Paul, R. Tibshirani, Prediction by supervised principal components, Journal of the American Statistical Association 101 (2006) 119–137.

[22] N. Aronszajn, Theory of reproducing kernels, Transactions of the American Mathematical Society 68 (3) (1950) 337–404.

[23] J. Nilsson, F. Sha, M. I. Jordan, Regression on manifolds using kernel dimension reduction, ICML 227 (2007) 697–704.

[24] A. Gretton, O. Bousquet, A. J. Smola, B. Scholkopf, Measuring statistical dependence with hilbert-schmidt norms., Proceedings Algorithmic Learning Theory (ALT) 3734 (2005) 63–77.

[25] L. Song, A. J. Smola, A. Gretton, K. M. Borgwardt, J. Bedo, Supervised feature selection via dependence estimation, ICML 227 (2007) 823–830.

[26] L. Song, A. J. Smola, K. M. Borgwardt, A. Gretton, Colored maximum variance unfolding, NIPS.

[27] L. Song, A. J. Smola, A. Gretton, K. M. Borgwardt, A dependence maximization view of clustering, ICML 227 (2007) 815–822.

[28] C. R. Baker, Joint measures and cross-covariance operators, Transactions of the American Mathematical Society 186 (1973) 273–289.

[29] H. Lutkepohl, Handbook of Matrices, John Wiley and Sons, 1997.

[30] J. L. Alperin, Local Representation Theory: Modular Representations as an Introduction to the Local Representation Theory of Finite Groups, Cambridge University Press, 1986.

[31] A. Asuncion, D. Newman, Uci machine learning repository (2007).

[32] U. Alon, N. Barkai, D. A. Notterman, K. Gishdagger, S. Ybarradagger, D. Mackdagger, A. J. Levine, Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays, Proceedings of the National Academy of Sciences of the United States of America 96 (12) (1999) 6745–6750.

[33] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, R. Levy, W. Wilson, M. R. Grever, J. C. Byrd, D. Botstein, P. O. Brown, L. M. Staudt, Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling., Nature 403 (6769) (2000) 503–511.

[34] J. Khan, J. S. Wei, M. Ringnr, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, P. S. Meltzer, Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks., Nature Medicine 7 (6) (2001) 673–9.

[35] L. Song, J. Bedo, K. M. Borgwardt, A. Gretton, A. J. Smola, Gene selection via the bahsic family of algorithms, Bioinformatics 23 (2007) 490–498.

[36] B. Li, H. Zha, F. Chiaromonte, Contour regression: A general approach to dimension reduction, ICML 33 (2005) 1580–1616.

[37] A. Rosenwald, G. Wright, W. C. Chan, J. M. Connors, E. Campo, R. I. Fisher, R. D. Gascoyne, H. K. Muller-Hermelink, E. B. Smeland, L. M.Staudt, The use of molecular profiling to predict survival after chemotherapy for diffuse large b-cell lymphoma, Annals of Statistics 346 (4) (2002) 1937–1947.

[38] S. Wold, C. Albano, W. J. Dunn, U. Edlund, K. Esbensen, P. Geladi, S. Hellberg, E. Johansson, W. Lindberg, M. Sjostrom, Mathematics and statistics in chemistry, chapter multivariate data analysis in chemistry, Chemometrics (1984) 17.

[39] P. D. Sampson, A. P. Streissguth, H. M. Barr, F. L. Bookstein, Neurobehavioral effects of prenatal alcohol: Part ii. partial least squares analysis, Neurotoxicology 11 (1989) 477–491.

[40] J. A. Wegelin, A survey of partial least squares (pls) methods, with emphasis on the two-block case, Technical report, University of Washington, 2000.

[41] R. Rosipal, L. J. Trejo, Kernel partial least squares regression in reproducing kernel hilbert space, Journal of Machine Learning Research 2 (2001) 97–123.

[42] R. Rosipal, Kernel partial least squares for nonlinear regression and discrimination, Neural Network World 13 (3) (2003) 291–300.

[43] H. Hotelling, Relations between two sets of variables, Biometrika 28 (1936) 312–377.

[44] D. Hardoon, S. Szedmak, J. Shawe-taylor, Canonical correlation analysis: An overview with application to learning methods, Neural Computation 16 (12) (2004) 2639–2664.

[45] M. S. Bartlett, Further aspects of the theory of multiple regression, In Proceedings of the Cambridge Philosophical Society 34 (1938) 33–40.

[46] C. Fyfe, P. Lai, Kernel and nonlinear canonical correlation analysis, In IEEE-INNS-ENNS International Joint Conference on Neural Networks 4 (2000) 4614.