Learning Incoherent Subspaces: Classification via Incoherent Dictionary Learning.

Daniele Barchiesi and Mark D. Plumbley*

Centre for Digital Music, Queen Mary University of London

E1 4NS, London, United Kingdom

May 30, 2014

Abstract

In this article we present the supervised iterative projections and rotations (S-IPR) algorithm, a method for learning discriminative incoherent subspaces from data. We derive S-IPR as a supervised extension of our previously proposed iterative projections and rotations (IPR) algorithm for incoherent dictionary learning, and we employ it to learn incoherent sub-spaces that model signals belonging to different classes. We test our method as a feature transform for supervised classification, first by visualising transformed features from a synthetic dataset and from the 'iris' dataset, then by using the resulting features in a classification experiment.

*This work has been supported by the Platform Grant EP/K009559/1 and the Leadership Fellowship EP/G007144/1, both from the UK Engineering and Physical Sciences Research Council (EPSRC).

1 Introduction: Classification And Feature Transforms

Supervised classification is one of the classic problems in machine learning where a system is designed to discriminate the class of an observed signal, having previously observed representative examples from the considered classes [1].

Typically, a classification algorithm consists of a training phase where class-specific models are learned from labelled samples, followed by a testing phase where unlabelled data are classified by comparison with the learned models. Both training and testing comprise various stages. Firstly, we observe a signal that measures a process of interest, such as the recording of a sound or image, or a log of the temperatures in a particular geographic area. Then, a set of features are extracted from the raw signals using signal processing techniques. This step is performed in order to reduce the dimensionality of the data and provide a new signal that allows generalisation among examples of the same class, while retaining enough information to discriminate between different classes.

Following the features extraction step, a feature transform can be employed to further reduce the dimensionality of the data and to enhance discrimination between classes. Thus classification benefits from feature transforms especially when features are not separable, that is, when it is not possible to optimise a simple function that maps features belonging to signals of a given class to the corresponding class. A further dimensionalty reduction may be performed when dealing with high dimensional signals (such as audio or high resolution images) by fitting the parameters of global statistical distributions with features learned on portions of the signal. Models learned on different classes are finally compared using a distance metric to the model learned from an unlabelled signal, which is typically assigned to the nearest class.

1.1 Traditional Algorithms For Feature Transform

Two of the main feature transform techniques include principal component analysis (PCA) [2] and Fisher's linear discriminant analysis (LDA) [1].

1.1.1 PCA

Let $\left\{\boldsymbol{x}_m \in \mathbb{R}^N\right\}_{m=1}^M$ be a set of vectors containing features extracted from M training signals. The goal of PCA is to learn an orthonormal set of basis functions $\left\{\boldsymbol{\phi}_k \in \mathbb{R}^N\right\}_{k=1}^N$ such that $||\boldsymbol{\phi}_k||_2 = 1$ and $\langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle = 0 \ \forall i \neq j$ that are placed along the columns of a so-called *dictionary* $\boldsymbol{\Phi} \in \mathbb{R}^{N \times N}$. The bases are optimised from the data to identify their principal components, that is, the sub-spaces that retain the maximum variance of the features.

To compute the dictionary, the eigenvalue decomposition of the outer product

$$\boldsymbol{X}\boldsymbol{X}^T = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T \tag{1}$$

is first calculated, where $\boldsymbol{X}$ contains the features $\boldsymbol{x}_m$ along its columns. Then, the L eigenvectors corresponding to the L largest eigenvalues are selected from the matrix $\boldsymbol{Q}$, and scaled to unit ℓ_2 norm to form the dictionary $\boldsymbol{\Phi} \in \mathbb{R}^{N \times L}$. A new set of transformed features $\boldsymbol{y}_{\text{PCA}} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{x}$ is computed by projecting the data onto the sub-space spanned by the columns of $\boldsymbol{\Phi}$ (that is, onto the L-dimensional principal sub-space). This operation reduces the dimensionality of the features by projecting them onto a linear subspace embedded in $\mathbb{R}^N$. It is an unsupervised technique that does not exploit knowledge about the classes associated with the training set, but implicitly relies in the assumption that the principal component directions encode relevant differences between classes.

1.1.2 LDA

In contrast, LDA is a supervised method for feature transform whose objective is to explicitly maximise the separability of classes in the transformed domain.

Let Γ_p be a set indexing features extracted from data belonging to the p-th class, and $|\Gamma_p|$ be its cardinality. Let

$$\bar{\boldsymbol{x}}_p \stackrel{\text{def}}{=} \frac{1}{|\Gamma_p|} \sum_{m \in \Gamma_p} \boldsymbol{x}_m \tag{2}$$

be the p-th class feature centroid, and $\bar{\boldsymbol{x}} \stackrel{\text{def}}{=} \sum_{m=1}^{M} \boldsymbol{x}_m$ the centroid of the features extracted from the entire training dataset. The between-classes scatter matrix

$$\boldsymbol{S_b} \stackrel{\text{def}}{=} \sum_{p=1}^{P} |\Gamma_p| \left(\bar{\boldsymbol{x}}_p - \bar{\boldsymbol{x}}\right) \left(\bar{\boldsymbol{x}}_p - \bar{\boldsymbol{x}}\right)^T \tag{3}$$

is defined to measure the mutual distances between the centroids of different classes, while the within-classes scatter matrix

$$\boldsymbol{S}_w \stackrel{\text{def}}{=} \sum_{p=1}^{P} \sum_{m \in \Gamma_p} \left(\boldsymbol{x}_m - \bar{\boldsymbol{x}}_p\right) \left(\boldsymbol{x}_m - \bar{\boldsymbol{x}}_p\right)^T \tag{4}$$

quantifies the distances between features belonging to the same class. Let $\boldsymbol{W}$ be a linear transform matrix. To maximise an objective function $\mathcal{J}(\boldsymbol{W}) \stackrel{\text{def}}{=} \frac{|\boldsymbol{W}^T \boldsymbol{S}_b \boldsymbol{W}|}{|\boldsymbol{W}^T \boldsymbol{S}_w \boldsymbol{W}|}$ that promotes features belonging to the same class to be near each other and far away from features belonging to other classes, the eigenvalue decomposition of the matrix

$$\boldsymbol{S}_w^{\dagger} \boldsymbol{S}_b = \boldsymbol{Q} \boldsymbol{\Lambda} \boldsymbol{Q}^T \tag{5}$$

is computed, and the features $\boldsymbol{x}$ are projected onto the space spanned by its $(P-1)$ eigenvectors corresponding to the largest $(P-1)$ eigenvalues (that is,

onto a space of dimensionality equal to the number of classes minus one).

LDA explicitly seeks to enhance the discriminative power of features by optimising the objective $\mathcal{J}$.

1.2 Supervised PCA

Related works that extend PCA include the supervised PCA (S-PCA) proposed by Barshan et al. [3]. S-PCA is based on the theory of reproducing kernel Hilbert spaces (RKHS) (that are spaces of functions which satisfy certain properties and map elements from an arbitrary set to the set of complex numbers) [4], and on the so-called Hilbert-Schmidt independence criterion (HSIC)[5]. The HSIC is used to estimate the statistical dependence of two random variables based on the fact that this quantity is related to the correlation of functions belonging to their respective RKHS. While HSIC is defined in terms of the probability density function of the two random variables, empirical estimates of HSIC can be obtained from finite sequences of their realisations. The empirical HSIC can be used in turn to construct an objective function that maximises the dependence between the two variables. Hence, this strategy is adopted within the context of classification to maximise the statistical dependence between a transformed feature $\boldsymbol{y}_{\text{S-PCA}}$ and its corresponding class c.

In practice, S-PCA differs from PCA in that it calculates the eigenvalue decomposition of a matrix $\boldsymbol{R}$ defined as follows:

$$\boldsymbol{R} \stackrel{\text{def}}{=} \boldsymbol{X}\boldsymbol{H}\boldsymbol{L}\boldsymbol{H}\boldsymbol{X}^T \tag{6}$$

were $\boldsymbol{H} \stackrel{\text{def}}{=} \boldsymbol{I} - \boldsymbol{e}\boldsymbol{e}^T$ is a so-called *centring* matrix[1] and $\boldsymbol{L} \stackrel{\text{def}}{=} \boldsymbol{c}\boldsymbol{c}^T$ is the kernel matrix of the class variable that is constructed by computing the outer product of the vectors resulting from assigning different numerical values to each class.

[1] Here $\boldsymbol{e}$ is a vector of ones.

1.3 Other related work

The union of incoherent sub-spaces model proposed by Schnass and Vandergheynst [6] employs a very similar intuition to the one that inspired our proposed method, and models features belonging to different classes using incoherent subspaces. Other methods for supervised dimensionality reduction include metric learning algorithms [7], sufficient dimensionality reduction [8] and Bair's supervised principal components [9].

Manifold learning techniques are used to model nonlinear data and reviewed by Van Der Maaten et al. [10]. Finally, the sparse sub-space clustering technique developed by Elhamifar and Vidal [11] is aimed at identifying vectors that belong to an union of sub-spaces, and hence apply concepts form sparse approximation to clustering.

1.4 Paper organisation

The method proposed in this paper is aimed at learning discriminative subspaces that allow dimensionality reduction, while at the same time enhancing the separability between classes. It is derived from our previous work on learning incoherent dictionaries for sparse approximation [12].

The incoherent dictionary learning problem will be introduced in Section 2, while Section 3 will contain the main contribution of this paper consisting in learning incoherent subspaces for classification. Numerical experiments are presented in Section 4, and conclusions are drawn in Section 5.

2 Incoherent Dictionary Learning

A sparse approximation of a signal $\boldsymbol{x} \in \mathbb{R}^N$ is a linear combination of $K \geq N$ basis functions $\left\{\boldsymbol{\phi}_k \in \mathbb{R}^N\right\}_{k=1}^K$ called *atoms* described by:

$$\boldsymbol{x} \approx \tilde{\boldsymbol{x}} = \sum_{k=1}^{K} \alpha_k \boldsymbol{\phi}_k \tag{7}$$

where the vector of coefficients $\boldsymbol{\alpha}$ contains a *small* number of non-zero components, corresponding to a small number of atoms actively contributing to the approximation $\tilde{\boldsymbol{x}}$. Given a signal $\boldsymbol{x}$ and a dictionary, various algorithms have been proposed to find a sparse approximation that minimises the residual error $||\boldsymbol{x} - \tilde{\boldsymbol{x}}||_2$[13].

Dictionary learning aims at optimising a dictionary $\boldsymbol{\Phi}$ for sparse approximation given a set of training data. It is an unsupervised technique that can be thought as being a generalisation of PCA, as both methods learn linear subspaces that minimise the approximation error of the signals. Dictionary learning, however, is generally more flexible than PCA because it can be employed to learn more general non-orthogonal over-complete dictionaries [14].

2.1 The incoherent dictionary learning problem

Dictionaries for sparse approximation have important intrinsic properties that describe the relations between their atoms, like the mutual coherence $\mu(\boldsymbol{\Phi}) = \max_{i \neq j} \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle$ that is defined as the maximum inner product between any two different atoms. The goal of incoherent dictionary learning is to learn atoms that are well adapted to sparsely approximate a set of training signals, and that are at the same time mutually incoherent [12].

Given a set of M training signals contained in the columns of the matrix $\boldsymbol{X} \in \mathbb{R}^{N \times M}$, the incoherent dictionary learning problem can be expressed as:

$$\boldsymbol{\Phi}^{\star}, \boldsymbol{A}^{\star} = \underset{\boldsymbol{\Phi}, \boldsymbol{A}}{\arg\min} \; ||\boldsymbol{X} - \boldsymbol{\Phi}\boldsymbol{A}||_{\mathrm{F}} \tag{8}$$

$$\text{such that } \mu(\boldsymbol{\Phi}) \leq \mu_0$$

$$||\boldsymbol{\alpha}_m||_0 \leq S \quad \forall m$$

where μ_0 is a fixed mutual coherence constraint, the ℓ_0 pseudo-norm $||\cdot||_0$ counts the number of non-zero components of its argument, S is a fixed number of active atoms, and $\boldsymbol{\alpha}_m$ denotes the m-th column of $\boldsymbol{A}$. Setting a specific value of the parameter S indicates that signals live on (or near) a union of subspaces of dimension S, and hence depends on the type of observed signals and on the number of atoms K (a large number of atoms that cover different directions in an N-dimensional space is likely to widen the set of signals that may be approximated using such union of sub-spaces). To obtain a dictionary with minimal mutual coherence, the parameter μ_0 can be set to the lower bound on the mutual coherence of a $N \times K$ dictionary $\mu_0 = \sqrt{(K-N)/N(K-1)}$[12]. This implies that less coherent dictionaries necessarily have a smaller number of atoms, determining a tradeoff between the parameters S and μ_0 which ultimately needs to be investigated according to the problem at hand.

Algorithms for (incoherent) dictionary learning generally follow an alternate optimisation heuristic, iteratively updating $\boldsymbol{\Phi}$ and $\boldsymbol{A}$ until a stopping criterion is met. In the case of the iterative projections and rotations algorithm (IPR) algorithm [12], a dictionary de-correlation step is added after updating the dictionary in order to satisfy the mutual coherence constraint.

Given $\boldsymbol{X}$, fixed μ_0, S and a stopping criterion (such as a maximum number of iterations), the optimisation of (8) is tackled by iteratively performing the following steps:

- *Sparse coding*: fix $\boldsymbol{\Phi}$ and compute the matrix $\boldsymbol{A}$ using a suitable sparse approximation method.
- *Dictionary update*: fix $\boldsymbol{A}$ and update $\boldsymbol{\Phi}$ using a suitable method for dictionary learning.
- *Dictionary de-correlation*: given $\boldsymbol{X}$, $\boldsymbol{\Phi}$ and $\boldsymbol{A}$ update the dictionary $\boldsymbol{\Phi}$ to reduce its mutual coherence under the level μ_0.

2.2 The iterative projections and rotations algorithm

The IPR algorithm has been proposed in order to solve the dictionary de-correlation step, while ensuring that the updated dictionary provides a sparse approximation with low residual norm, as indicated by the objective function (8) [12].

The IPR algorithm requires the calculation of the Gram matrix $\boldsymbol{G} = \boldsymbol{\Phi}^T\boldsymbol{\Phi}$ which contains the inner products between any two atoms in the dictionary. $\boldsymbol{G}$ is iteratively projected onto two constraint sets, namely the structural constraint set $\mathcal{K}_{\mu_0}$ and the spectral constraint set $\mathcal{F}$. The former is the set of symmetric square matrices with unit diagonal values and off-diagonal values with magnitude smaller or equal than μ_0:

$$\mathcal{K}_{\mu_0} \stackrel{\text{def}}{=} \left\{ \boldsymbol{K} \in \mathbb{R}^{K\times K} : \boldsymbol{K} = \boldsymbol{K}^T, k_{i,i} = 1, \max_{i>j} |k_{i,j}| \leq \mu_0 \right\}.$$

The latter is the set of symmetric positive semidefinite square matrices with rank smaller than or equal to N:

$$\mathcal{F} \stackrel{\text{def}}{=} \left\{ \boldsymbol{F} \in \mathbb{R}^{K\times K} : \boldsymbol{F} = \boldsymbol{F}^T, \text{eig}(\boldsymbol{F}) \geq \boldsymbol{0}, \text{rank}(\boldsymbol{F}) \leq N \right\}$$

where the operator $\text{eig}(\cdot)$ returns the vector of eigenvalues of its argument.

Starting from the Gram matrix of an initial dictionary $\boldsymbol{\Phi}$, the IPR method

iteratively performs the following operations.

- *Projection onto the structural constraint set.* The projection $\boldsymbol{K} = \mathcal{P}_{\mathcal{K}_{\mu_0}}(\boldsymbol{G})$ can be obtained by:

  1. setting $k_{i,i} = 1$,
  2. limiting the off-diagonal elements so that, for $i \neq j$,

$$k_{i,j} = \text{Limit}(g_{i,j}, \mu_0) = \begin{cases} g_{i,j} & \text{if} \quad |g_{i,j}| \leq \mu_0 \\ \text{sgn}(g_{i,j})\mu_0 & \text{if} \quad |g_{i,j}| > \mu_0 \end{cases} \tag{9}$$

- *Projection onto the spectral constraint set and factorization.* The projection $\boldsymbol{F} = \mathcal{P}_{\mathcal{F}}(\boldsymbol{G})$ and subsequent factorisation are obtained by:

  1. calculating the eigenvalue decomposition (EVD) $\boldsymbol{G} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$,
  2. thresholding the eigenvalues by keeping only the N largest positive ones.

$$[\text{Thresh}(\boldsymbol{\Lambda}, N)]_{i,i} = \begin{cases} \lambda_{i,i} & \text{if} \quad i \leq N \text{ and } \lambda_{i,i} > 0 \\ 0 & \text{if} \quad i > N \text{ or } \lambda_{i,i} \leq 0 \end{cases}$$

     where the eigenvalues in $\boldsymbol{\Lambda}$ are ordered from the largest to the smallest. Following this step, at most N eigenvalues of the Gram matrix are different from zero,

  3. factorizing the projected Gram matrix into the product $\boldsymbol{G} = \boldsymbol{\Phi}^T\boldsymbol{\Phi}$ by setting:

$$\boldsymbol{\Phi} = \tilde{\boldsymbol{\Lambda}}^{1/2}\boldsymbol{Q}^T \tag{10}$$

     where $\tilde{\boldsymbol{\Lambda}} \in \mathbb{R}^{N \times K}$ is the eigenvalues matrix restricted to the first N rows.

- *Dictionary rotation.* Rotate the dictionary $\boldsymbol{\Phi}$ to align it to the training

set by solving the problem:

$$\boldsymbol{W}^{\star} = \underset{\boldsymbol{W}\boldsymbol{W}^T=\boldsymbol{I}}{\arg\min} \; ||\boldsymbol{X} - \boldsymbol{W}\boldsymbol{\Phi}\boldsymbol{A}||_{\mathrm{F}} \,. \tag{11}$$

The optimal rotation matrix can be calculated by:

1. computing the sample covariance between the observed signals and their approximations $\boldsymbol{C} \stackrel{\text{def}}{=} (\boldsymbol{\Phi}\boldsymbol{A})\boldsymbol{X}^T$,
2. calculating the SVD of the covariance $\boldsymbol{C} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$,
3. setting the optimal rotation matrix to $\boldsymbol{W}^{\star} = \boldsymbol{V}\boldsymbol{U}^T$,
4. rotating the dictionary $\boldsymbol{\Phi} \leftarrow \boldsymbol{W}^{\star}\boldsymbol{\Phi}$.

More details about the IPR algorithm can be found in [12], including details of its computational cost.

3 Learning Incoherent Subspaces

The IPR algorithm learns a dictionary where all the atoms are mutually incoherent. Therefore, given any two disjoint sets $\Lambda \bigcap \Gamma = \emptyset$ that identify non-overlapping collections of atoms, the sub-dictionaries $\boldsymbol{\Phi}_{\Lambda}, \boldsymbol{\Phi}_{\Gamma}$ are also mutually incoherent.

Starting from this observation, the main intuition driving the development of a supervised IPR (S-IPR) algorithm for classification is to learn mutually incoherent sub-dictionaries that approximate features from different classes of signals. The sub-dictionaries are in turn used to define incoherent sub-spaces, and features are projected onto these sub-spaces yielding discriminative dimensionality reduction.

3.1 The supervised IPR algorithm

Let $\{c_m \in \mathcal{C}\}_{m=1}^{M}$, $\mathcal{C} = \{C_1, C_2, \ldots, C_P\}$ be a set of labels that identify the class of the vectors of features $\boldsymbol{x}_m$, whose elements belong to a set $\mathcal{C}$ of P possible classes. The columns of the matrix $\boldsymbol{X}_p$ contain a selection of the features extracted from signals belonging to the p-th class.

To learn incoherent sub-dictionaries from the entire set of features, we must first cluster the atoms to different classes[2], and then only proceed with their de-correlation if they are assigned to different classes (while allowing coherent atoms to approximate features from the same class). To this aim, we employ the matrix $\boldsymbol{A}$ to measure the contribution of every atom to the approximation of features belonging to each class.

Let $\boldsymbol{\alpha}_p^k$ indicate the k-th row of the matrix $\boldsymbol{A}_p$ containing the coefficients that contribute to the approximation of $\boldsymbol{X}_p$, and N_p indicate the number of signals belonging to the p-th class. A coefficient $\gamma_{k,p}$ is defined as:

$$\gamma_{k,p} \stackrel{\text{def}}{=} \frac{1}{N_p} \left|\left|\boldsymbol{\alpha}_p^k\right|\right|_1, \tag{12}$$

and every atom $\boldsymbol{\phi}_k$ is associated with the class to which it maximally contributes $p_k^\star = \arg\max_p \{\gamma_{k,p}\}$. Figure 1 depicts the structure of the matrices involved in Equation (8), highlighting the vector used to calculate the coefficients $\gamma_{k,p}$.

Grouping together atoms that have been assigned to the same class leads to a set of sub-dictionaries whose size and rank depends on the number of atoms for each class, and to their linear dependence. As a general heuristic, if features corresponding to different classes do not occupy the same sub-space (according to the active elements in $\boldsymbol{A}$), a full-rank dictionary $\boldsymbol{\Phi}$ with $K \geq N \gg P$ ensures that $p_k^\star$ identify P non-empty and disjoint sub-dictionaries $\{\boldsymbol{\Phi}_p\}_{p=1}^{P}$.

[2]Note that the term *cluster* implies that a this stage the algorithm needs to make an unsupervised decision, since there is no any a-priori reason to assign a given atom to any particular class.

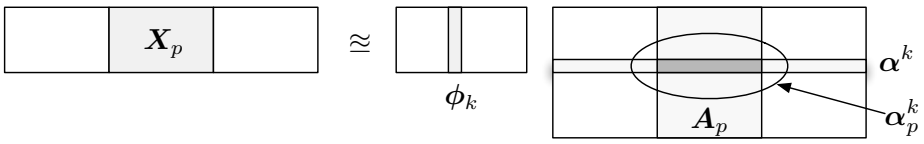


Figure 1: Sketch of the matrix factorisation introduced in Equation (8). The features $\boldsymbol{X}_p$ belonging to class p are approximated using the coefficients $\boldsymbol{A}_p$. The vector $\boldsymbol{\alpha}_p^k$ contains the coefficients that determine the contribution of the atom $\boldsymbol{\phi}_k$ to the approximation of features belonging to the p-th class.

Once the atoms have been clustered, the Gram matrix $\boldsymbol{G}$ is computed and iteratively projected as in the method described in Section 2.2, with the difference that equation (9) is modified in order to only constrain the mutual coherence between atoms assigned to different classes

$$\text{Limit}(g_{i,j}, \mu_0, \boldsymbol{p}^\star) = \begin{cases} g_{i,j} & \text{if} \quad |g_{i,j}| \leq \mu_0 \,\text{or}\, p_i^\star = p_j^\star \\ \text{sgn}(g_{i,j})\mu_0 & \text{if} \quad |g_{i,j}| > \mu_0 \,\text{and}\, p_i^\star \neq p_j^\star \end{cases} \tag{13}$$

A further modification of the standard IPR algorithm presented in [12] consists in the update of the Gram matrix, performed by computing its element-wise average with the projection $\boldsymbol{K} = \mathcal{P}_{\mathcal{K}_{\mu_0}}(\boldsymbol{G})$ (rather than by using the projection alone). This heuristic has led to improved empirical results by preventing $\boldsymbol{G}$ from changing too abruptly.

The complete supervised S-IPR method is summarised in Algorithm 1. Note that the mutual coherence $\mu_{p^\star}(\boldsymbol{\Phi}) = \underset{p_i^\star \neq p_j^\star}{\arg\max} \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle$ indicated in this algorithm measures the inner product between any two atoms assigned to different classes since atoms assigned to the same class are allowed to be mutually coherent.

3.2 Classification via incoherent subspaces

The S-IPR algorithm allows to learn a set of sub-dictionaries $\{\boldsymbol{\Phi}_p\}$ that contain mutually incoherent atoms. These cannot be directly used to define discriminative subspaces because, depending on N and on the rank of each sub-dictionary,

Algorithm 1: Supervised IPR

Input: $\boldsymbol{X}, \boldsymbol{\Phi}, \boldsymbol{A}, \mu_0, \boldsymbol{c}, I$
Output: $\boldsymbol{\Phi}^\star$

$i \leftarrow 1$;
// Cluster atoms
$\boldsymbol{A}_p \leftarrow [\boldsymbol{\alpha}_j] \, \forall j \in C_p$;
$\gamma_{k,p} \leftarrow \left|\left|\boldsymbol{\alpha}_p^k\right|\right|_1 / N_p$;
$p_k^\star = \arg\max_p \{\gamma_{k,p}\}$;
while $i \leq I$ *and* $\mu_{p^\star}(\boldsymbol{\Phi}) > \mu_0$ **do**
    // Calculate Gram matrix
    $\boldsymbol{G} \leftarrow \boldsymbol{\Phi}^T \boldsymbol{\Phi}$;
    // Project onto structural c.s.
    $\mathrm{diag}(\boldsymbol{K}) \leftarrow \boldsymbol{1}$;
    $\boldsymbol{K} \leftarrow \mathrm{Limit}(\boldsymbol{G}, \mu_0, \boldsymbol{p}^\star)$;
    $\boldsymbol{G} \leftarrow \frac{1}{2}\boldsymbol{G} + \frac{1}{2}\boldsymbol{K}$;
    // Project onto spectral c.s. and factorize
    $[\boldsymbol{Q}, \boldsymbol{\Lambda}] \leftarrow \mathrm{EVD}(\boldsymbol{G})$;
    $\boldsymbol{\Lambda} \leftarrow \mathrm{Thresh}(\boldsymbol{\Lambda}, N)$;
    $\boldsymbol{\Phi} \leftarrow \boldsymbol{\Lambda}^{1/2} \boldsymbol{Q}^T$;
    // Rotate dictionary
    $\boldsymbol{C} \leftarrow \boldsymbol{X}(\boldsymbol{\Phi}\boldsymbol{A})^T$;
    $[\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}] \leftarrow \mathrm{SVD}(\boldsymbol{C})$;
    $\boldsymbol{W} \leftarrow \boldsymbol{V}\boldsymbol{U}^T$;
    $\boldsymbol{\Phi} \leftarrow \boldsymbol{W}\boldsymbol{\Phi}$;
    $i \leftarrow i + 1$;
end

atoms belonging to disjoint sub-dictionaries might span identical subspaces. Instead, we fix a rank $Q \leq \lfloor N/P \rfloor$ (i.e., the integer part of the ratio N/P) and choose a collection of Q linearly independent atoms from each sub-dictionary $\boldsymbol{\Phi}_p$, using the largest values of $\gamma_{k,p}$ to define a picking order. Thus, we obtain a set $\{\boldsymbol{\Psi}_p\}_{p=1}^P$ of incoherent sub-spaces of rank Q embedded in the space $\mathbb{R}^N$, and use them to derive a feature transform for classification.

Each feature vector $\boldsymbol{x}_m$ that belongs to the class c_m is projected onto the relative subspace, yielding a set of transformed features $\{\boldsymbol{y}_m\}_{m=1}^M$.

$$\boldsymbol{y}_m = \boldsymbol{\Psi}_{c_m} \boldsymbol{\Psi}_{c_m}^\dagger \boldsymbol{x}_m \tag{14}$$

where $\mathbf{\Psi}^\dagger$ denotes the Moore-Penrose pseudo-inverse of the matrix $\mathbf{\Psi}$ and needs to be used in place of the transposition operator because the columns of $\mathbf{\Psi}$ are in general not orthogonal.

When an unlabelled signal is presented to the classifier, the corresponding vector of features $\boldsymbol{x}$ is projected onto all the learned sub-spaces. Then, the nearest sub-space is chosen using an Euclidean distance measure, and the corresponding projection $\boldsymbol{y}$ used as the transformed feature.

$$p^\star = \arg\min_p \left|\left|\boldsymbol{x} - \mathbf{\Psi}_p \mathbf{\Psi}_p^\dagger \boldsymbol{x}\right|\right|_2 \tag{15}$$

$$\boldsymbol{y} = \mathbf{\Psi}_{p^\star} \mathbf{\Psi}_{p^\star}^\dagger \boldsymbol{x} \tag{16}$$

The subspace $p^\star$ can be directly used as an estimator of the class of the signal $c^\star$. Alternatively, a simple *k-neaerst neighbour* classifier can be employed on the transformed features, and a class can be inferred as:

$$c^\star = \texttt{knn}(\boldsymbol{y}, \boldsymbol{Y}, \boldsymbol{c}) \tag{17}$$

where $\boldsymbol{Y}$ represents the matrix of training features after the transform stage. This latter approach is especially suitable when working with a large number of classes in a space of relatively small dimension, as in this case multiple classes might be assigned to the same subspace.

4 Numerical Experiments

4.1 Feature visualisation

To illustrate the S-IPR algorithm for feature transform, we first run visualisation experiments depicting how different feature transform methods act on training

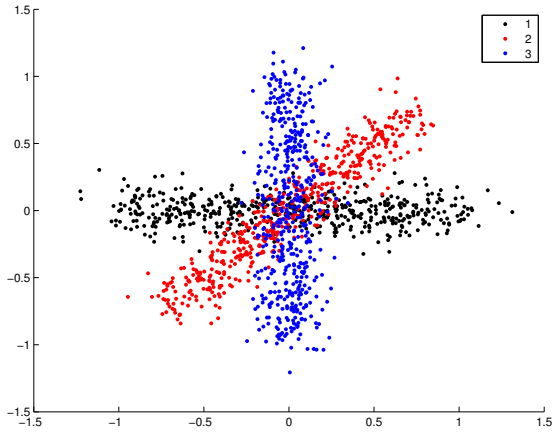


Figure 2: Synthetic data generated along one-dimensional subspaces of $\mathbb{R}^2$.

and test data[3].

4.1.1 Synthetic data

Figure 2 displays a total of 1500 synthetic features in $\mathbb{R}^2$ belonging to 3 different classes that we generated for this experiment. For each class, first we draw values distributed uniformly in the interval $[-1, 1]$ and assign them to the first component of the features (the x coordinate). Then, we add Gaussian noise with variance 0.1 to the second component (the y coordinate), and we rotate the resulting data by the angles $\theta_0 = 0$, $\theta_1 = \pi/4$ and $\theta_3 = \pi/2$ for the 3 classes respectively. This way, features belonging to different classes are clustered along different one-dimensional sub-spaces of $\mathbb{R}^2$.

Figure 3 displays the result of the application of feature transforms to the data depicted in Figure 2 using subspaces of dimension 1 (with the exception of LDA that projects the data onto a space of dimension $P-1 = 2$). To generate the plots, we divided the data into a training set (displayed using the '+' marker) and a test set (displayed using the 'o' marker). Samples were drawn in random

[3]The Matlab code used to generate the results in this Section is available from https://github.com/danieleb/2014-SJSPS

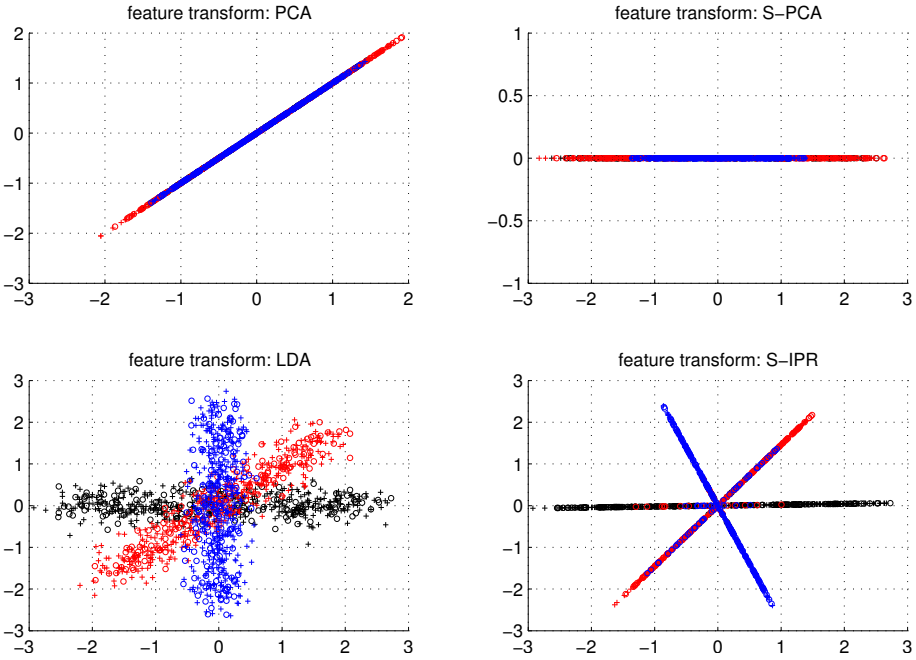


Figure 3: Feature transform applied to the synthetic data in Figure 2. Different colours correspond to different classes, '+' and 'o' markers represent samples taken from the training and test set respectively.

order from the dataset and assigned to either the training set or the test set, with the former containing 70% of the total data and the latter containing the remaining 30%. Then, we applied feature transforms on the training set, thereby learning the transform operators, and applied them to the test set.

Starting from the top-left plot, we can observe that PCA identified the direction $x = y$ as the one-dimensional subspace that contains most of the variance of the training set. However, given the type of dataset and the dimensionality reduction caused by PCA, features from all classes are overlapping, making this transform a poor choice for classification. Similar observations can be drawn from analysing the result of S-PCA, although this transform identifies the direction $y = 0$ as the one that leads to statistical dependence between the value of the transformed features in the training set and the relative class. LDA does not introduce any dimensionality reduction in this case, as it projects the features onto a space of dimension $P - 1 = 2$, leaving the original features unaltered. However, in the LDA plot we can appreciate the separation between training set and test set that is difficult to notice in the other plots.

Finally, the plot at the right-bottom corner of Figure 3 displays the results of the S-IPR algorithm. In setting the parameters of S-IPR, we chose a 2 times overcomplete dictionary, a number of active atoms equal to half the dimension of the data, and minimal mutual coherence. In the case considered here, this means $K = 4$, $S = 1$ and $\mu = \sqrt{(K - N)/N(K - 1)} \approx 0.33$. As discussed in Section 3.1, S-IPR does not project whole sets of features onto a unique sub-space, but rather learns one sub-space for each class, and projects features onto the nearest sub-space. The result depicted here shows that three directions were identified containing data from mostly one class each. Since the incoherent dictionary learning is designed to learn atoms with minimal mutual coherence, the angles between the directions of the sub-spaces learned by S-IPR are approximately

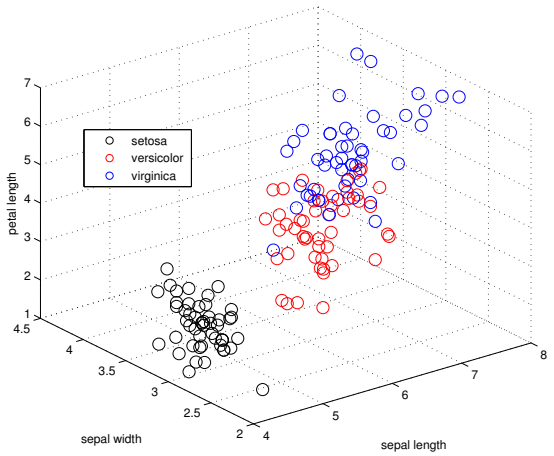


Figure 4: First three features of the 'iris' dataset depicting measurements of sepal length, sepal width and petal length of three iris species.

equal. Prior information regarding the directions of the data would allow to relax the parameter μ, and track more closely the directions of the three data classes.

4.1.2 Iris dataset

Figure 4 displays a subset of the 'iris' dataset, a popular database that has been used extensively to test and benchmark classification algorithms. The original dataset contains measurements of the sepal length, sepal width, petal length and petal width of three species of iris, namely 'setosa', 'versicolor' and 'virginica'. In this visualisation experiment we selected the first 3 features to be able to depict the data using three dimensional scatter plots. From observing the distribution of the data in the feature space, we see that 'setosa' is relatively separated from the other two classes, while the features relative to 'virginica' and 'versicolor' substantially overlap, with only a few exemplars of 'virginica' being distinguishable due to large sepal length and petal length.

The results of feature transforms are depicted in Figure 5. This time, we

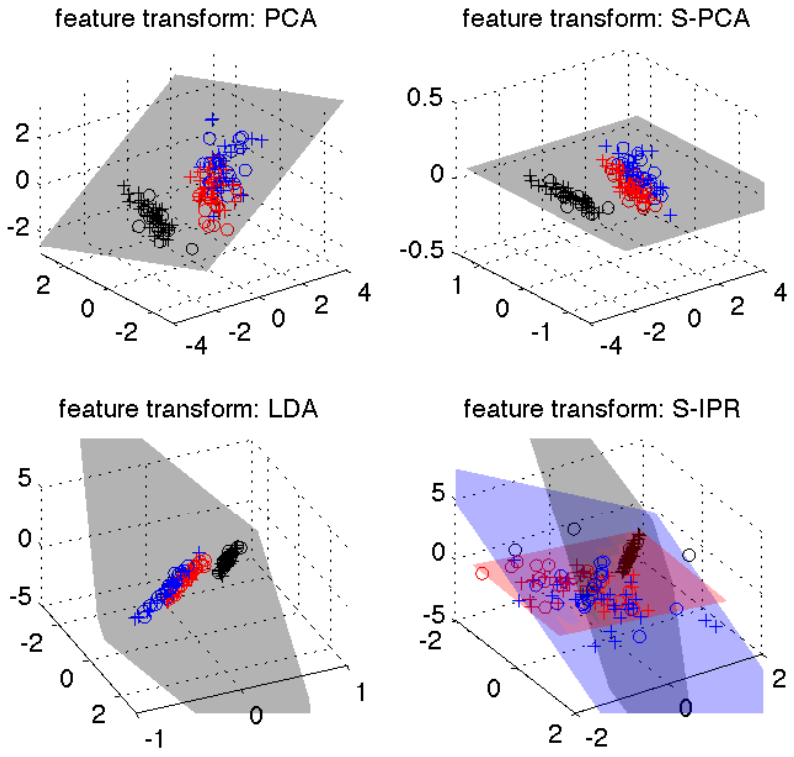


Figure 5: Feature transform applied to the iris data in Figure 4. Different colours correspond to different classes, '+' and 'o' markers represent samples taken from the training and test set respectively.

Name	N	P	M
Iris	4	3	150
Balance	4	3	625
Parkinsons	23	2	197
Sonar	60	2	208
USPS	256	3	1405

Table 1: Dataset used in the classification evaluation of feature transform algorithms. All the datasets can be downloaded from http://archive.ics.uci.edu/ml/datasets.html. Note that we only use a subset of the USPS dataset containing the digits 1, 3 and 8.

learn 2 dimensional subspaces from the 3 dimensional data points and plot the transformed features, along with the learned planes. We observe that PCA identifies a direction along a diagonal axis that follows the distribution of features displayed in Figure 4. S-PCA , on the other hand, projects the features onto a horizontal plane that slightly enhances the separation between 'versicolor' and 'virginica' samples. LDA results in a projection where features belonging to the same class are closely clustered together, but fails to separate the classes 'versicolor' and 'virginica'. Finally, the output of S-IPR displays three distinct sub-spaces associated with the three classes. As in the other plots, the separation between 'versicolor' and 'virginica' is far from perfect, however features from the two classes are mostly projected onto the respective sub-spaces. Features belonging to the 'setosa' class are mostly clustered together as a result of their projection onto the black subspace, however we can note a few test samples that have been associated by the algorithm to the blue sub-space.

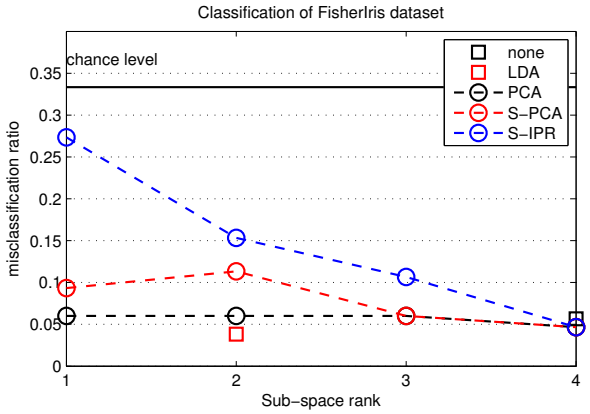
4.2 Classification

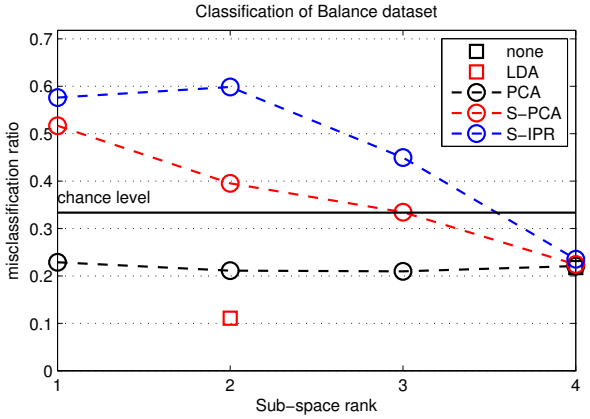
In the previous section, we have illustrated how the S-IPR algorithm is able to learn incoherent sub-spaces that model the distribution of features belonging to different classes. Here we evaluate S-IPR and the other feature transform algorithms in the context of supervised classification. To perform the classification,

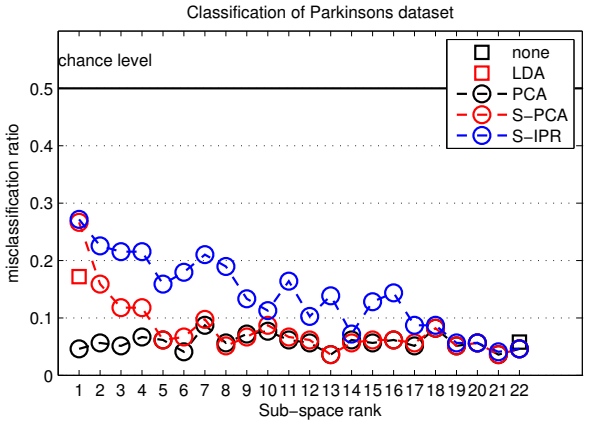
features are transformed using the methods already used for comparison in Section 4.1 by learning a transform operator on the training set and applying it to the test set. We use a 5-fold stratified cross-validation to classify all the features in a dataset during the test stage. This method produces 5 independent classification problems with a ratio between the number of training and test samples equal to $8:2$. Once the features have been transformed, a k-nearest neighbour classifier with $k = 5$ is used to estimate a class.

We employ the datasets detailed in Table 1, and for each of them we evaluate the misclassification ratio, that is defined as the fraction of misclassified samples as a proportion of the total number of samples in the test set, averaged over the 5 independent classification problems created by the stratified cross-validation protocol.

Figure 6 displays for each dataset the misclassification ratio as a function of the rank of the subspace learned by the algorithms. In the plots 'none' indicates that no feature transform was applied (hence resulting in a sub-space rank equal to the dimension of the original features). In general we can see that S-IPR does not perform as well as the other techniques, and is only comparable at high ranks that do not achieve an overall better classification ratio. Starting from the 'iris' dataset, LDA achieves the best performance followed by one-dimensional subspaces learned using PCA. Both S-PCA and S-IPR work better when learning subspaces of high rank. Note that, at rank $N = 4$ all the methods are equivalent because they are not performing dimensionality reduction. The results relative to the balance dataset are similar, with again LDA achieving the best misclassification ratio. Although the results on the 'Parkinsons' and 'sonar' datasets present similar trends regarding S-IPR, here LDA does not prove to be as successful as PCA and S-PCA in separating features belonging to different classes. Finally, for the 'USPS' digits dataset, it appears that a low-rank PCA


Classification of FisherIris dataset
chance level
none
LDA
PCA
S−PCA
S−IPR
misclassification ratio
Sub−space rank


Classification of Balance dataset
chance level
none
LDA
PCA
S−PCA
S−IPR
misclassification ratio
Sub−space rank


Classification of Parkinsons dataset
chance level
none
LDA
PCA
S−PCA
S−IPR
misclassification ratio
Sub−space rank

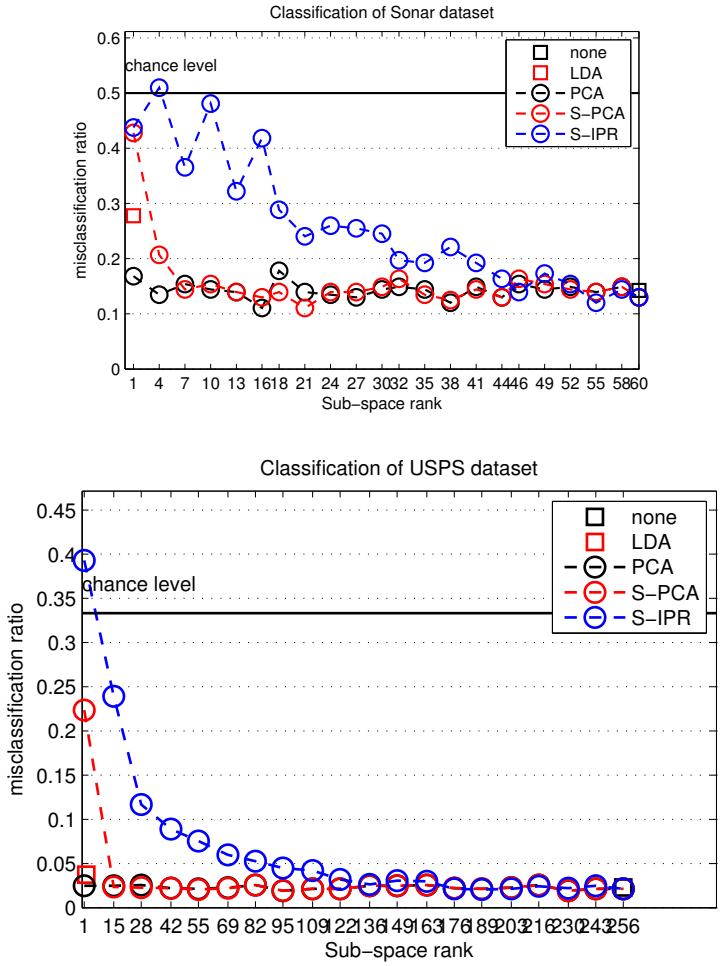


Figure 6: Misclassification ratio as a function of the rank of the subspace employed during feature transforms for the datasets 'iris','balance','Parkinsons', 'sonar' and 'USPS'.

is sufficient to reach the best classification performance.

5 Conclusion

5.1 Summary

We have presented the S-IPR algorithm for learning incoherent subspaces from data belonging to different classes. The encouraging experimental results obtained on the visualisation of the synthetic dataset and of a subset of features taken from the 'iris' dataset motivated us to test S-IPR as a general method for feature transform to be used in classification problems. Unfortunately, we found that the performance of our proposed method on a group of datasets commonly used to benchmark classification algorithms is only competitive compared to traditional and state-of-the-art methods for feature transform at high sub-space ranks.

The negative results presented in Section 4.2 do not imply that S-IPR is completely unsuitable as a tool for modelling data for classification, but they rather open a few important areas of future research that should be investigated to better understand the strengths and limitations of the proposed method.

5.2 Future work

The main assumption made when using incoherent dictionary learning for classification is that high dimensional features are arranged onto lower-dimensional sub-spaces, and that features belonging to different classes can be modelled using different subspaces that are mutually incoherent. This assumption might be met by some datasets, but might not generally be satisfied by others. Understanding the general distribution of the features in a dataset might be a necessary first step to inform a subsequent choice of algorithm, so that S-IPR

can be used in cases where its premise about the feature distribution is valid. This same argument holds for the whole class of linear models that comprises the dictionary learning model. Indeed, many feature transform techniques have equivalent *kernelized* versions to model non-linear data.

Other substantial improvements can be made on the algorithm itself. The present implementation of S-IPR contains a fixed parameter μ that promotes minimal mutual coherence between the sub-spaces used to approximate different data classes. Knowledge about the distribution of the features might lead to relaxing this parameter, learning sub-spaces that are closer to the true distribution of the features and in turn improving class separation. Moreover, different values of mutual coherence for different pairs of subspaces can be easily included in the optimisation, greatly enhancing the flexibility of S-IPR as a modelling tool.

References

[1] R. Duda and P. E. Hart, *Pattern classification and scene analysis*, Wiley and Sons, 1973.

[2] K. Pearson, "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science, Sixth Series*, vol. 2, pp. 559–572, 1901.

[3] E. Barshan, A. Ghodsi, Z. Azimifar, and M. Z. Jahromi, "Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds," *Pattern Recognition*, vol. 44, no. 7, pp. 1357–1371, 2011.

[4] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American Mathematical Society*, vol. 68, no. 3, pp. 337–404, 1950.

[5] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf, "Measuring statistical dependence with hilbert-schmidt norms," in *Algorithmic learning theory*. Springer, 2005, pp. 63–77.

[6] Karin Schnass and Pierre Vandergheynst, "A union of incoherent spaces model for classification," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2010, pp. 5490–5493.

[7] Eric P Xing, Michael I Jordan, Stuart Russell, and Andrew Ng, "Distance metric learning with application to clustering with side-information," in *Advances in neural information processing systems*, 2002, pp. 505–512.

[8] Ker-Chau Li, "Sliced inverse regression for dimension reduction," *Journal of the American Statistical Association*, vol. 86, no. 414, pp. 316–327, 1991.

[9] Eric Bair, Debashis Paul, and Robert Tibshirani, "Prediction by supervised principal components," *Journal of the American Statistical Association*, vol. 101, pp. 119–137, 2006.

[10] L. Van Der Maaten, E. Postma, and J. Van Den Herik, "Dimensionality reduction: A comparative review," Tech. Rep., TiCC, Tilburg University, 2009.

[11] E. Elhamifar and R. Vidal, "Sparse subspace clustering: algorithm, theory, and applications," available at http://arxiv.org/abs/1203.1005, 2013.

[12] D. Barchiesi and M. D. Plumbley, "Learning incoherent dictionaries for sparse approximation using iterative projections and rotations," *IEEE Trans. on Signal Processing*, vol. 61, no. 8, pp. 2055–2065, Apr. 2013.

[13] M. Elad, *Sparse and redundant representations*, Springer, 2010.

[14] R. Rubinstein, A. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1045–1057, Jun. 2010.