Application-Driven Dimension Reduction

Anuj Srivastava [1] Xiuwen Liu [2]

Abstract

Simplicity and efficiency of linear transformations make them a popular tool for reducing dimensions (of data) before or during statistical analysis. Examples of their applications include image compression and reconstruction, data clustering, pattern classification, and image or text retrieval. Linear transformations with natural orthogonality constraints can be represented as elements of Stiefel and Grassmann manifolds. We advocate that the choice of a transformation for dimension reduction is not standard; it is dictated by the application and the data set, and can be formulated as an optimization problem on these above-mentioned manifolds. We demonstrate this idea by deriving dimension-reducing transformations in several applications, including image-based recognition of objects and content-based retrieval of images.

Key words:
PACS:

1 Introduction

In many applications involving pattern recognition and image analysis, large dimensions of observed data prohibit efficient use of statistical approaches. It becomes imperative to use a dimension-reduction technique either before or during statistical analysis of data. Dimension reduction also provides compression and coding necessary to avoid excessive memory storage and computation. In some applications, such as face recognition using images, the underlying variability in observed data is known to result from a handful of physical variables, such as pose, shape, and illumination, and that provides a strong motivation for seeking low-dimension representations of data. Additionally, these low-dimension representations can provide a useful immunity to observation noise or clutter.

[1] Department of Statistics, Florida State University, Tallahassee, FL 32306
[2] Department of Computer Science, Florida State University, Tallahassee, FL 32306

One can devise several ways of generating low-dimensional representations of data; a common classification of these ideas is into linear and nonlinear methods. There is a growing realization that observed data points form low-dimensional nonlinear manifolds inside much larger Euclidean spaces, and several recent papers have focused on discovering these underlying manifolds using the observed points. Nonlinear mapping, isomaps [1], local-linear embedding [2], Laplacian eigenmaps [3], and their variations are some of the approaches to attempt to accomplish this task. Despite these advances in understanding of nonlinear structures, there remains a need for linear representations. Linear representations are attractive because of the simplicity and efficiency they provide, especially in real-time applications. From a geometric viewpoint, a linear dimension-reduction approach amounts to approximating a nonlinear manifold globally by a low-dimensional hyperplane. Although such strategy will fail in general, it has often been useful in special cases. For example, in situations involving proximity of data points and/or negligible curvature of underlying manifold in neighborhoods local to observed points, a case can be made for use of linear models.

For reducing data dimension from n to d, where $d << n$, using a linear transformation, a common approach is to multiply the data vector by a $d \times n$ matrix. It seems natural and efficient to restrict to matrices with linearly independent rows, or even further, to impose orthonormality constraint on the rows. For instance, let U be an $n \times d$ orthogonal matrix denoting an orthonormal basis of a d-dimensional subspace of $\mathbb{R}^n$ ($n >> d$), and let X be an $n \times 1$ observation vector. Then, the vector $Y(X, U) = U^T X \in \mathbb{R}^d$, also called the vector of coefficients, is a d-dimensional representation of X. In this setup, several standard transformations, such as those resulting from a principal component analysis (PCA) or a Fisher discriminant analysis (FDA), have widely been used. Although they satisfy some optimality criteria, they may not necessarily be optimal for a specific application at hand. Their optimality criterion is often not related to the application performance, or if related, is based on unrealistic assumptions (such as X being multivariate normal). We contend that the choice of U should be closely related to both application and data. In other words, a criterion dictated by the application should be used to find an optimal choice of U. Similar ideas have also been pursued in [4,5], albeit in different contexts. Guided by the application, one can derive or formulate a performance function F such that the optimal linear transformation is given by $\hat{U} = \text{argmax}_U F(U)$. The problem of finding a linear transformation now becomes that of optimization over an appropriate space of orthonormal bases. The main goal of this paper is to present a family of algorithms for finding *orthogonal linear transformations of data that are optimal for specific tasks and specific datasets*. Our search for optimal transformations is based on a stochastic optimization process that maximizes a given performance function (chosen for that application). To be computationally effective, the optimization process has been modified to account for the geometry of the underlying

space by formulating the problem on manifolds of orthogonal bases. In case one is interested in searching over bases, the required space is given by a Stiefel manifold. On the other hand, if one is interested in a subspace, and not in a particular basis, the search space becomes a Grassmann manifold. A important part of solving optimization problems on these manifolds is to understand and utilize their differential geometry in reaching intrinsic solutions.

The rest of this paper is organized as follows: Section 2 introduces the two manifolds of interest, Grassmann and Stiefel. Section 3 motivates this approach by listing several applications and performance functions associated with these applications. Section 4 introduces elements from different geometry to construct gradient-based solutions to optimization problems on these manifolds, and Section 5 presents an algorithm for solving such problems. A few experimental results illustrating the basic ideas are presented in Section 6. The papers ends with a brief summary in Section 7.

2 Representation of Linear Projections

We are interested in linear transformations that can be used for reducing dimensions of observations before their analysis. Such transformations can be denoted by $n \times d$, non-singular matrices, or elements of a generalized linear group $GL(n, d)$. If the columns are forced to be linearly independent, which seems natural for studying linear transformations, an efficient representation is obtained by further assuming that the columns are orthogonal and of unit length. Denoting such a linear transformation via a matrix $U \in \mathbb{R}^{n\times d}$, U satisfies the property that $U^T U = I_d$, where I_d is the $d \times d$ identity matrix. Let the set of all such matrices be $\mathcal{S}_{n,d}$,

$$\mathcal{S}_{n,d} = \{U \in \mathbb{R}^{n\times d} | U^T U = I_d\} \quad \subset GL(n, d). \tag{1}$$

$\mathcal{S}_{n,d}$ is called a **Stiefel** manifold, and each element of $\mathcal{S}_{n,d}$ provides an orthonormal basis for a d-dimensional subspace of $\mathbb{R}^n$. If one is interested only in the subspace spanned by the columns of U, and not in a particular basis, then the required space is reduced further, as follows. Define an equivalence relation between two elements of $\mathcal{S}_{n,d}$ according to:

$$U_1 \sim U_2 \;\text{ if and only if }\; U_1 = U_2 O, \;\text{ for some }\; O \in SO(d)\ .$$

Here, $SO(d) = \{O \in \mathbb{R}^{d\times d} | O^T O = I_d,\ \det(O) = 1\}$. This relation implies that all orthonormal bases of the same subspace are deemed equivalent. Define a new set $\mathcal{G}_{n,d}$ as the quotient space of $\mathcal{S}_{n,d}$ under this equivalence relation, i. e. $\mathcal{G}_{n,d} = \mathcal{S}_{n,d}/\sim$. $\mathcal{G}_{n,d}$ is called a **Grassmann manifold** and it forms the set

of all d-dimensional subspaces of $\mathbb{R}^n$. Another way of stating this equivalence relation is $\mathcal{G}_{n,d} = \mathcal{S}_{n,d}/SO(d)$.

Both $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$ can also be viewed as quotient spaces of a larger (special) orthogonal group according to:

$$\mathcal{S}_{n,d} = SO(n)/SO(n-d) \text{ and } \mathcal{G}_{n,d} = \mathcal{S}_{n,d}/SO(d) = SO(n)/(SO(n-d)\times SO(d)) .$$

To interpret these definitions, consider $SO(n)$ as the set of $n \times n$ rotation matrices. The subgroup $SO(d)$ consists of rotations within a d-dimensional subspace of $\mathbb{R}^n$ while the subgroup $SO(n-d)$ consists of rotations within the remaining $(n-d)$-dimensional subspace. The quotient space $SO(n)/SO(n-d)$ can be interpreted as considering those rotations in $\mathbb{R}^n$ where the rotations within an $(n-d)$-dimensional subspace are ignored. Similarly, in the case of $SO(n)/(SO(d) \times SO(n-d))$, rotations within a d-dimensional subspace, and the subspace orthogonal to it in $\mathbb{R}^n$ are ignored. Therefore, an element of $\mathcal{G}_{n,d}$ represents a subspace (rotations within the subspace, and rotations orthogonal to it are ignored), while an element of $\mathcal{S}_{n,d}$ represents a particular basis of a subspace (only rotations orthogonal to the subspace are ignored, the ones within the subspace are included).

Many properties of $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$ are inherited from $SO(n)$. $SO(n)$ is a matrix Lie group, i.e. it is a differentiable manifold and a group under matrix multiplication (see for example [6]). It is compact, smooth, and has dimensions given by $n(n-1)/2$. $\mathcal{S}_{n,d}$ is a manifold of dimension $(n(n-1)/2)-((n-d)(n-d-1)/2) = nd - d(d+1)/2$, and $\mathcal{G}_{n,d}$ has $d(n-d)$ dimensions. $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$ are also compact, and continuous functions defined on these manifolds attain maximum (or minimum) on them.

We emphasize the choice of orthonormal bases in representing linear transformations as it leads to a significant reduction in computational cost. Solving for an orthogonal basis on $\mathcal{S}_{n,d}$ or $\mathcal{G}_{n,d}$ leads to a smaller search space as compared to searching for optimal linear transformations on nd-dimensional space of $n \times d$ matrices. It also provides stability to iterative optimization algorithms by ensuring that the basis vectors remain unit length and the basis matrix is always full ranked.

We will use an orthogonal matrix $U \in \mathbb{R}^{n\times d}$ to denote an element of $\mathcal{S}_{n,d}$ and a set $[U]$ to denote an element of $\mathcal{G}_{n,d}$, where

$$[U] = \{UO \in \mathbb{R}^{n\times d} | O \in SO(d)\} .$$

$[U]$ denotes the equivalence class of all the bases spanning the same d-dimensional subspace of $\mathbb{R}^n$.

3 Application-Driven Dimension Reduction

We will use the following notation. Let $X \in \mathbb{R}^n$ be a vector of random variables under investigation, and for $U \in \mathcal{S}_{n,d}$ or $[U] \in \mathcal{G}_{n,d}$, let $Y = U^T X$ denote a random vector that represents a linear projection of X into $\mathbb{R}^d$. In situations where n is rather large, Y is used instead of X in generating statistical inferences. This step is often called *dimension reduction* or *component analysis*. The choice of basis U has a significant impact on the eventual performance and, thus, becomes a topic of research. Our strategy is that instead of selecting a standard U, disjoint from the application under investigation, we tie the choice of U to the application. In other words, the application, or a performance function in that application, dictates what U is optimal in that context. Next, we present some examples of these applications and their performance functions. In some cases, it is straightforward to solve for an optimal basis, while others require iterative strategies.

(1) **Image reconstruction under mean-squared error**: Let $X \in \mathbb{R}^n$ denote a vector form of an image, and the goal is to find a d-dimensional subspace of $\mathbb{R}^n$ such that the projection of X on that subspace minimizes expected squared error. That is, find a $[U] \in \mathcal{G}_{n,d}$ such that:

$$[\hat{U}] = \underset{[U]\in\mathcal{G}_{n,d}}{\operatorname{argmax}} F([U]) = \underset{[U]\in\mathcal{G}_{n,d}}{\operatorname{argmax}} E[\|U^T X - X\|^2] \ , \tag{2}$$

where E denotes the expectation over a joint probability density function of X. Let $\Sigma \in \mathbb{R}^{n\times n}$ be the covariance matrix of X, and $\Sigma = V\Lambda V^T$ be its singular value decomposition (SVD). Then, the first d columns of V provide $[\hat{U}]$. Here we are assuming that the entries of diagonal matrix Λ are non-increasing from top to bottom.

In practical situations, where the covariance Σ is not known, and only some observations of X are available, one uses the sample covariance matrix, $\hat{\Sigma} \in \mathbb{R}^{n\times n}$:

$$\hat{\Sigma} = \frac{1}{M-1}\sum_{m=1}^{M}(X_i - \bar{X})(X_i - \bar{X})^T \ , \quad \text{where} \quad \bar{X} = \frac{1}{M}\sum_{m=1}^{M} X_i \quad .$$

This calculation of $[\hat{U}]$ is called *principal component analysis*, and is the most commonly-used technique for statistical dimensional reduction [7,8].

(2) **Class separation for clustered data**: In this problem, the goal is to transform labelled data points in such a way that different clusters maximally separate from each other. Let $\{C_j, \ \ j = 1, \ldots, K\}$ denote different classes such that each data point $X_i \in C_j$ for some j. We simplify the notation by using X_i^j to imply $X_i \in C_j$. We want to find an optimal d-dimensional subspace, or find a $[U] \in \mathcal{G}_{n,d}$, that projects X_i^js into maximally separated clusters, assuming normal distributions with identical

covariances and different means across clusters.

Let $\bar{Y}_j(U) \in \mathbb{R}^d$, $j = 1, 2, \ldots, K$, be the average (over i) of projected vectors $U^T X_i^j$, in each class. The *between-class* scatter matrix is given by:

$$S_B(U) = \sum_{j=1}^{K} (\bar{Y}_j(U) - \bar{Y}(U))(\bar{Y}_j(U) - \bar{Y}(U))^T \in \mathbb{R}^{d \times d}, \text{ where } \bar{Y}(U) = \frac{1}{K} \sum_{j=1}^{K} \bar{Y}_j(U) .$$

Let the *within-class* scatter matrix be given by:

$$S_W(U) = \sum_{j=1}^{K} \left(\sum_{i=1} (X_i^j - \bar{Y}_j(U))(X_i^j - \bar{Y}_j(U))^T \right) \in \mathbb{R}^{d \times d} .$$

The required basis is now given by:

$$[\hat{U}] = \underset{[U] \in \mathcal{G}_{n,d}}{\operatorname{argmax}} F([U]) = \underset{[U] \in \mathcal{G}_{n,d}}{\operatorname{argmax}} \frac{\det(S_B(U))}{\det(S_W(U))} , \tag{3}$$

where det denotes matrix determinant. The solution is obtained using a generalized eigen-decomposition [9]. This analysis is also known as *Fisher's linear discriminant analysis* [10].

(3) **Signal extraction from a mixture using independence**: Consider the problem where observed signal is an (unknown) linear mixture of a number of (statistically independent) signals, and the goal is to extract individual signals from the observed data. Comon [11] and others have posed this problem as that of finding linear transformations of data such that the transformed components are statistically independent from each other. This approach, called *independent component analysis* has found applications in signal processing [11] and image analysis [12–14].

Let $Y(U) = U^T X$ be the vector of transformed random variables. Seeking independence of elements of $Y(U)$, define a cost function:

$$F(U) = -KL(P(Y_1(U), Y_2(U), \ldots, Y_n(U)) || P_1(Y_1(U)) P_2(Y_2(U)) \ldots P_n(Y_n(U))) , \tag{4}$$

where KL denotes the Kullback-Leibler divergence, and $Y_i(U)$ implies the i^{th} (scalar) component of the vector Y. In this definition, P denotes the joint probability, and P_is denote the marginal probabilities of the random variables $Y_i(U)$s. The desired transformation is obtained by solving for $\hat{U} = \operatorname{argmax}_{U \in \mathcal{S}_{n,d}} F(U)$. Since it is difficult to estimate mutual information using observations of random variables, several approximations of F have exploited to obtain ICA in the literature. Comon [11] proposed cost functions that approximate minimization of mutual information and Bell et al. [15] used a stochastic gradient technique to solve such optimization problem. It must be noted that some of these formulations do not require orthogonality of basis; in fact, they often use over-complete or non-orthogonal bases. Hyvärinen [16] proposed a FastICA algorithm for computing ICA using an approximation based on negative entropy function.

(4) **Recognition of objects from image data**: In this example, X denotes an image of an object of interest and the goal is to recognize that object using a statistical analysis of X. This application is called appearance-based recognition of objects. There are several approaches to solve this problem, and as an example, one can choose a nearest neighbor classifier to compare test images with the training images, and to label the test images according to their nearest training image. Using the Euclidean distance between the transformed image, define $d(X_1, X_2; U) = \|U^T X_1 - U^T X_2\|$, where $\|\cdot\|$ denotes the 2-norm of a vector. Let there be K objects (or classes) to be recognized from the images; each object has m_{train} training images (denoted by $X_1^j, \ldots, X_{m_{train}}^j$, $j = 1, \ldots, K$). Define $\rho(X_i^j, U)$ to be the ratio of the between-class-minimum distance and within-class minimum distance of a training image from class j indexed by i in the leave-one out sense, given by

$$\rho(X_i^j, U) = \frac{\min_{j' \neq j, k} d(X_i^j, X_k^{j'}; U)}{\min_{k \neq i} d(X_i^j, X_k^j; U) + \epsilon_0} ,$$

where $\epsilon_0 > 0$ small is added to avoid division by zero. Then, define F according to:

$$F([U]) = \frac{1}{K m_{train}} \sum_{j=1}^{K} \sum_{i=1}^{m_{train}} h(\rho(X_i^j, U) - 1), \tag{5}$$

where $h(\cdot)$ is a monotonically increasing and bounded function. In our experiments, we have used $h(x) = 1/(1 + \exp(-2\beta x))$, where β controls the smoothness of F. Note that X_i^j is classified correctly in the leave-one out sense according to the nearest neighbor rule under U if and only if $\rho(X_i^j, U) > 1$. It follows that F is precisely the leave-one out recognition performance of the nearest neighbor classifier when $\beta \to \infty$. Having defined F, the goal is to solve for $[\hat{U}] = \text{argmax}_{[U] \in \mathcal{G}_{n,d}} F([U])$, as described in [17].

(5) **Generalization of representations**: A key problem in appearance-based recognition is the issue of generalization, i.e. how to extend a good performance on the training data to the test data. Without explicit models of the underlying physical processes, one has to rely on heuristics to improve the performance on test datasets that are not known beforehand. We follow the idea of maximally-separated hyperplanes (used in support vector machines [18]) and define the generalization of a linear representation as separation between clusters formed by (training) images from different classes.

Define a generalization performance function as follows. Let there be K classes to be recognized from the images; each class has m_{train} training images (denoted by $X_1^j, \ldots, X_{m_{train}}^j$) and m_{cross} cross validation im-

ages[3] (denoted by $\tilde{X}_1^j, \ldots, \tilde{X}_{m_{cross}}^j$) to evaluate the recognition performance measure, for $j = 1, 2, \ldots, K$. To obtain a smooth F, first define $\rho(\tilde{X}_i^j, U)$ to be the ratio of the between-class-minimum distance and average within-class minimum distance of a validation image i from class j based on the r-nearest neighbor rule, given by

$$\rho(\tilde{X}_i^j, U) = \frac{\min_{j' \neq j,k} d(\tilde{X}_i^j, X_k^{j'}; U)}{1/|\tau(\tilde{X}_i^j, r)| \sum_{X \in \tau(\tilde{X}_i^j, r)} d(\tilde{X}_i^j, X; U) + \epsilon_0}, \tag{6}$$

where $\tau(\tilde{X}_i^j, r)$ is the set of r training images closest to $\tilde{X}_i^j$ in class j. If $\tau(\tilde{X}_i^j, r)$ does not contain r elements, the first term in the denominator is set to $\{\min_k d(\tilde{X}_i^j, X_k^j)\}$). Then, define $F(U)$ according to:

$$F([U]) = \frac{1}{K m_{cross}} \sum_{j=1}^{K} \sum_{i=1}^{m_{cross}} h(\rho(\tilde{X}_i^j, U) - 1), \quad 0 \leq F(U) \leq 1.0 \tag{7}$$

where $h(\cdot)$ is a monotonically increasing and bounded function, as earlier. Under this formulation, F achieves its maximum when clusters formed by images from one class are far way from clusters by images from different classes. Therefore, the optimization problem becomes $[\hat{U}] = \text{argmax}_{[U] \in \mathcal{G}_{n,d}} F([U])$.

(6) **Sparse representations of natural images**: An important principle for deriving components of natural images has been to maximize the sparseness of transformed images ([19,20]). Sparseness of the resulting coding on the given dataset can be specified in many ways. One example is:

$$F([U]) = -E[\sum_{i=1}^{d} \log(1 + Y_i(X, U)^2)] \ , \tag{8}$$

where E is the expectation over the probability distribution of X. One motivation for choosing sparseness comes from its emerging connection with neural coding of visual stimuli in the visual cortex. In practice, this expectation is computed by averaging over all training images. Note that this definition of $F(U)$ is the negative of the one given in ([20]). A related problem deals with finding data transformations that enhance non-Gaussianity of the observed data. As an example, let $\kappa(y)$ denote the kurtosis of a random variable y. Then, define total kurtosis of a random vector as

$$F(U) = \sum_{i=1}^{d} \kappa(Y_i(U)) \ ,$$

[3] This can be implemented in two ways: having a separate cross validation set or using the training set as the validation set in the leave-one-out sense.

and set the goal to find $\hat{U} = \text{argmax}_U F(U)$. In the context of spectral decomposition of images via linear filters, several papers have studied this problem [21–23].

(7) **Retrieval of images from a large database**: Another important application of linear components is in retrieval of images (or text files) from a large database using queries. Since images, or text files, have large dimensions, a dimension reduction step becomes important before retrieval, to improve implementation speed. How to setup a performance function in the context of content-based image retrieval? We use the retrieval precision as F with a fixed number of retrieved images [24]. To be more specific, let there are K classes in an image dataset; each class has m_{db} images (denoted by $X_1^j, \ldots, X_{m_{db}}^j$) to be retrieved and m_{qu} query images (denoted by $\tilde{X}_i^j, \ldots, \tilde{X}_{m_{qu}}^j$). Here for simplicity, we assume that each class has the same m_{db} and m_{qu}, although it can be modified easily to allow for different numbers of images in different classes. To evaluate the precision measure, let R denote the number of images to be retrieved for each query image, we define F as the average retrieval precision for all the query images, given by

$$F(U) = \frac{1}{Km_{qu}} \sum_{j=1}^{K} \sum_{i=1}^{m_{qu}} \frac{\text{No. of relevant images retrieved}}{R}. \tag{9}$$

Relevant images are pre-defined by the end user for the purpose of testing retrieval algorithms. As the total number of relevant images is known in this setting, for each $F(U)$, the corresponding average *recall* measure is given by $F(U)R/m_{db}$. Now, the goal is to find a basis that maximizes F, i.e. $\hat{U} = \text{argmax}_{\mathcal{S}_{n,d}} F(U)$. Note that the optimal performance of $F(U)$ is given by $\min\{1, m_{db}/R\}$, the goal is basically to find a U that achieves this performance.

These are some examples of applications that naturally specify a performance function under which a choice of dimension-reduction transformation must be made. It is easy to find such examples in all branches of computational sciences, in situations where high-dimensional observations are involved.

4 Differential Geometry of $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$

Our goal is to solve optimization problems on these two spaces and we will use a stochastic gradient procedure to solve this problem. In view of the nonlinear nature of these manifolds, we require two tools from differential geometry to solve optimization problems. One relates to defining tangent spaces on these manifolds, in order to define derivatives and gradients, and the second relates to computing flows generated by vector fields on these spaces, in order to

compute gradient flows. Next, we summarize these two ideas in the context of $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$.

4.1 Tangent Spaces

For a manifold M, and any point $p \in M$, we will use $T_p(M)$ to denote the set of vectors tangent to M at p. This set is actually a vector space with same dimension as M. Elements of $T_p(M)$, called the *tangent vectors* (even though they happen to be $n \times d$ matrices in case of $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$), are found by constructing paths on M passing through p and finding the velocities of these paths at p. The characterization of $T_p(M)$ is important because for any $v \in T_p(M)$ and any differentiable map $F : M \mapsto \mathbb{R}$, $(vF)(p)$ denotes the directional derivative of F in the direction v at point p. In order to perform gradient-based optimization, one has to be able to define and evaluate these directional derivatives at all points of M.

In the case $M = SO(n)$, the tangent spaces are simple to construct. At identity I_n, the tangent space $T_{I_n}(SO(n))$ is given by the set of $n \times n$ skew-symmetric matrices. This vector space is spanned by an orthogonal basis $\{E_{ij}\}$, where E_{ij} is an $n \times n$ skew-symmetric matrix such that: for $1 \leq i \leq n$ and $i < j \leq n$,

$$E_{ij}(k,l) = \begin{cases} 1/\sqrt{2} & \text{if } \; k = i, \; l = j \\ -1/\sqrt{2} & \text{if } \; k = j, \; l = i \\ 0 & \text{otherwise .} \end{cases} \tag{10}$$

Any element of $T_{I_n}(SO(n))$ can be written as a linear combination of E_{ij}s.

For any $Q \in SO(n)$ other than identity, the tangent space $T_Q(SO(n))$ is simply a rotated version of $T_{I_n}(SO(n))$. (This is a property of Lie groups where the tangent space at any point can be generated by "rotating" the tangent space at any other point). It can be shown that

$$T_Q(SO(n)) = \{QA | A \in T_{I_n}(SO(n))\} \; .$$

In fact, the set $\{QE_{ij}\}$ forms an orthogonal basis of the tangent space $T_Q(SO(n))$.

Using the tangent structure on $SO(n)$, we can derive tangent structures on the quotient spaces $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$ as follows. If M/H is a quotient space of M under the action of a subgroup $H \subset M$ (assuming H acts on M), then, for any point $p \in M$, a vector $v \in T_p(M)$ is also tangent to M/H as long as it is perpendicular to the tangent space $T_p(pH)$. Next, we use this idea to find tangent spaces on $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$.

(1) **Tangent Structure of $\mathcal{S}_{n,d}$**: Since $\mathcal{S}_{n,d} = SO(n)/SO(n-d)$, set $M = SO(n)$ and $H = SO(n-d)$, and the tangent spaces on $\mathcal{S}_{n,d}$ can be seen as subsets of tangent spaces on $SO(n)$. Let $J \in \mathbb{R}^{n\times d}$ be made up of the first d columns of I_n; J acts as "identity" element in $\mathcal{S}_{n,d}$. A vector in $T_{I_n}(SO(n))$ is also tangent to $\mathcal{S}_{n,d}$ at J as long as it is perpendicular to $T_{I_{n-d}}(SO(n-d))$. Therefore,

$$\begin{aligned} T_J(\mathcal{S}_{n,d}) &= \{AJ | A \in T_{I_n}(SO(n)), A \perp T_{I_{n-d}}(SO(n-d))\} \\ &= \{AJ | A \in \mathbb{R}^{n\times n}, A = -A^T, A \perp \begin{bmatrix} 0 & 0 \\ 0 & D \end{bmatrix}, D = -D^T, D \in \mathbb{R}^{(n-d)\times(n-d)}\} \\ &= \{\begin{bmatrix} C & B \\ -B^T & 0 \end{bmatrix} J | C = -C^T, C \in \mathbb{R}^{d\times d}, B \in \mathbb{R}^{d\times(n-d)}\} \ . \end{aligned} \tag{11}$$

An orthonormal basis of this tangent space is given by:

$$\{E_{ij}J, i = 1, \ldots, d; j = i+1, \ldots, d\} \cup \{E_{ij}J, \ i = 1, \ldots, d; j = d+1, \ldots, n\} \ . \tag{12}$$

The first set of vectors span the matrix C, while the second set span the matrix B.

For any other point $U \in \mathcal{S}_{n,d}$, let $Q^T \in SO(n)$ be a matrix that rotates the columns of U to align with the columns of J, i.e. let $U = Q^T J$. Note that the choice of Q is not unique and can be obtained efficiently using Householder decomposition [25]. Then, the tangent space at U is given by:

$$T_U(\mathcal{S}_{n,d}) = \{Q^T A | A \in T_J(\mathcal{S}_{n,d})\} \ .$$

An orthonormal basis of this space is obtained by multiplying the elements in Eqn. 12 by Q^T on the left.

(2) **Tangent Structure of $\mathcal{G}_{n,d}$**: In this case, set $M = SO(n)$ and $H = SO(d)\times SO(n-d)$. Now the vectors tangent to $SO(n)$ will also be tangent to $\mathcal{G}_{n,d}$ as long as they are perpendicular to the space $(T_{I_d}(SO(d)) \times T_{I_{n-d}}(SO(n-d)))$. The tangent space at $[J] \in \mathcal{G}_{n,d}$ is given by:

$$\begin{aligned} T_{[J]}(\mathcal{G}_{n,d}) = \{AJ | A \in T_{I_n}(SO(n)), A \perp \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix}, \ \ A_1 \in T_{I_d}(SO(d)), \\ A_2 \in T_{I_{n-d}}(SO(n-d))\} \\ = \{\begin{bmatrix} 0 & B \\ -B^T & 0 \end{bmatrix} J \ | \ \ B \in \mathbb{R}^{d\times(n-d)}\} \end{aligned} \tag{13}$$

This space is spanned by orthonormal basis vectors $\{E_{ij}J, \ i = 1, \ldots, d; j = d+1, \ldots, n\}$. For any other point $[U] \in \mathcal{G}_{n,d}$, let $Q \in SO(n)$ be a matrix

such that $U = Q^T J$. Then, the tangent space at $[U]$ is given by:

$$T_{[U]}(\mathcal{G}_{n,d}) = \{Q^T A | A \in T_{[J]}(\mathcal{G}_{n,d})\} \ .$$

An orthonormal basis of this space is given by $\{Q^T E_{ij} J, \ i = 1, \ldots, d; j = d+1, \ldots, n\}$.

Now that we have defined tangent spaces at all points on $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$, we consider (tangent) vector fields on these manifolds. V is called a vector field on M, if it assigns a tangent vector $V_p \in T_P(M)$ for all $p \in M$. In case the vector V_p changes smoothly with p, the vector field V is called a *smooth vector field*. Let $F : M \mapsto \mathbb{R}$ be a smooth map, and $\{e_{i,p}\}$ form an orthonormal basis for $T_p(M)$ for all p. Then, $(e_{i,p}F)$ is the directional derivative of F at p in the direction of $e_{i,p}$, and the vector field $V_p = \sum_i (e_{i,p}F)e_{i,p}$ is called the *gradient vector field* of F on M. In the case of $\mathcal{S}_{n,d}$ or $\mathcal{G}_{n,d}$, a gradient vector field takes the form: for $U \in \mathcal{S}_{n,d}$ or $[U] \in \mathcal{G}_{n,d}$,

$$V_U = Q^T (\sum_{i,j} (E_{ij}F) E_{ij}) J \ , \tag{14}$$

where E_{ij} are as defined earlier, and the ranges of i, j for summation depend on whether the manifold is $\mathcal{S}_{n,d}$ or $\mathcal{G}_{n,d}$.

4.2 Gradient Flows on $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$

Given a gradient vector field V on a manifold $\mathcal{S}_{n,d}$ or $\mathcal{G}_{n,d}$, we seek a process $X(t)$ such that

$$\frac{dX(t)}{dt} = V_{X(t)} \ . \tag{15}$$

$X(t)$ is called a *gradient flow*, generated by the gradient vector field V (Eqn. 14. The next issue is: How to solve for the flow $X(t)$? On a computer, one can approximate the solution using the following discretization: for a small step size $\delta > 0$, we will generate a discrete-time process $\{X(s\delta), s = 1, 2, \ldots, \}$ that will approximate the solution of Eqn. 15 as δ gets smaller. This discrete process is computed according to:

$$X_{(s+1)\delta} = \psi_{X(s\delta)}(V_{X(s\delta)}, \delta) \ ,$$

where ψ is a geodesic path (on the manifold) that is defined next. The notation $\psi_p(v, t)$ for a point $p \in M$, and a vector $v \in T_p(M)$ implies a geodesic flow starting at p in the direction v, and parameterized by time t. In other words, $\psi_p(v, 0) = p$ and $\dot{\psi}_p(v, 0) = v$.

For any $Q \in SO(n)$, a geodesic flow in a tangent direction (say, QA) is given by $\psi_Q(A,t) = Q\exp(tA)$, where exp is the matrix exponential. Since A is a skew-symmetric matrix, and $\exp(A)$ is always orthogonal, this flow always stays on $SO(n)$. Note that the velocity vector of this flow at Q is actually QA, the given tangent vector. Similarly, in case of $\mathcal{S}_{n,d}$ a geodesic flow starting from a point $U \in \mathcal{S}_{n,d}$ in a direction $Q^TAJ \in T_U(\mathcal{S}_{n,d})$ is given by:

$$\psi_U(Q^TAJ, \cdot) : t \mapsto Q^T \exp(tA)J \quad \in \mathcal{S}_{n,d} \ , \tag{16}$$

and for $\mathcal{G}_{n,d}$ by:

$$\psi_{[U]}(Q^TAJ, \cdot) : t \mapsto Q^T \exp(tA)J \quad \in \mathcal{G}_{n,d} \ . \tag{17}$$

Recall that in case of $\mathcal{S}_{n,d}$, the skew-symmetric matrix A is of the type $\begin{bmatrix} C & B \\ -B^T & 0 \end{bmatrix}$, whereas for $\mathcal{G}_{n,d}$ it is of the type $\begin{bmatrix} 0 & B \\ -B^T & 0 \end{bmatrix}$. The cost of computing these flows depends heavily on evaluation of $\exp(tA)$; the paper [25] utilizes the structure of A in these cases and describes an efficient way to compute $\exp(tA)$ in the case of $\mathcal{G}_{n,d}$.

Combining these ideas, a gradient vector flow on $\mathcal{S}_{n,d}$ or $\mathcal{G}_{n,d}$ is given by:

$$X_{(s+1)\delta} = Q_s^T \exp(\delta W_s)J \ , \quad \text{where} \quad X_{s\delta} = Q_s^T J \quad \text{and} \quad W_s = \sum_{ij}(E_{ij}F)E_{ij} \ . \tag{18}$$

Note that W_s is slightly different from V_s in Eqn. 14 in that Q^T is missing on the left side. Also, the gradient flows on $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$ differ only in the terms include in summation over E_ij.

The notion of geodesic flow also helps in approximating directional derivatives of functions in cases where analytical expressions are not available. For a function F on $\mathcal{S}_{n,d}$, the directional derivative $E_{ij}F$ at $U \in \mathcal{S}_{n,d}$ is defined as:

$$(E_{ij}F) = \lim_{\epsilon \downarrow 0} \frac{1}{\epsilon} \Big(F(Q^T \exp(\epsilon E_{ij})J) - F(U) \Big) \ .$$

In cases, where $(E_{ij}F)$ is difficult to derive analytically, one can use the approximation:

$$(E_{ij}F) \sim \frac{1}{\epsilon} \Big(F(Q^T \exp(\epsilon E_{ij})J) - F(U) \Big) \ , \tag{19}$$

for a small $\epsilon > 0$.

5 Optimization Procedures

In each of the application mentioned earlier, the goal of finding an optimal dimension-reduction transformation reduces to solving an optimization problem on a Stiefel or a Grassmann manifold. In this section, we look at a general stochastic gradient type approach to solving such problems.

For $M = \mathcal{S}_{n,d}$ or $\mathcal{G}_{n,d}$, let $F : M \mapsto \mathbb{R}_+$ be a performance function such that we seek

$$\hat{U} = \underset{U \in \mathcal{S}_{n,d}}{\operatorname{argmax}} F(U) \quad \text{or} \quad [\hat{U}] = \underset{[U] \in \mathcal{G}_{n,d}}{\operatorname{argmax}} F([U]) \ . \tag{20}$$

Since F is a C^1-function, one can define a continuous vector field on M according to gradient of F as in Eqn. 14. The gradient flow is approximated by Eqn. 18 , and has the property that it converges to a local maximum of the function F. To seek global optimal, one often adds a stochastic component to the gradient according to $\tilde{W}_s = W_s + \sum_{i,j} z_{ij} E_{ij}$, where z_{ij} is a normal random variable with mean zero and variance $1/\delta$. Another useful idea seems to be use a Markov chain framework. The basic idea is to construct a stochastic gradient, governed by Markov chain dynamics, in such a way that the process converges to a global optimum in the limit. This idea has been pursued earlier in [17,26] for optimization on Grassmann manifolds. In [17], we have adopted a Metropolis-Hastings type of idea to stochastic gradient search; stochastic gradient provides candidates for updating estimates, but they are accepted or rejected according to a probability density function that depends upon F. In other words, it uses randomly-perturbed versions of the gradient directions to find candidates for updating the chain; these candidates are accepted and rejected according to a probability proportional to $\exp(F(p))$. The algorithm is repeated here for convenience of readers.

Algorithm 1: Let $X_0 = U_0 \in \mathcal{S}_{n,d}$ or $[U_0] \in \mathcal{G}_{n,d}$ be any initial condition. Set $s = 0$.

(1) Calculate the directional derivative of F at $X_{s\delta}$ according to:

$$W_s = \sum_{i,j} (E_{ij} F) E_{ij} \quad \in \ \mathbb{R}^{n \times n} \ , \tag{21}$$

where $(E_{ij}F)$ can be approximated numerically using Eqn. 19.

(2) Generate independent and identical realizations z_{ij}s, from $N(0, 1/\delta)$. Using the value of $X_{s\delta}$, calculate a candidate value Y according to:

$$\tilde{W}_s = W_s + \sqrt{2T_s} \sum_{i,j} z_{ij} E_{ij},$$

$$Y = Q_s^T \exp(\delta \tilde{W}_s) J, \tag{22}$$

The operation $\exp(\delta \tilde{V}_s)$ can be performed in $O(nk^2)$ operations by exploiting the structure of sparse skew-symmetric matrix $\tilde{V}_s$, as described in [25].

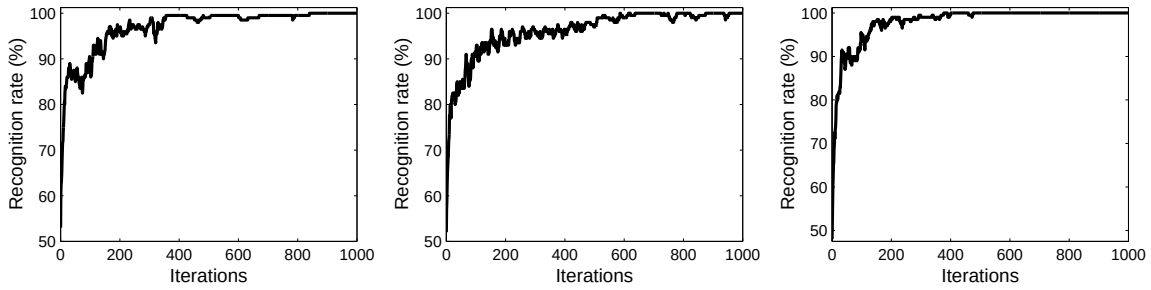
(3) Compute $F(Y)$, $F(X_{s\delta})$, and set $dF = F(Y) - F(X_{s\delta})$.
(4) Set $X_{(s+1)\delta} = Y$ with probability $\min\{\exp(dF/T_s), 1\}$, else set $X_{(s+1)\delta} = X_{s\delta}$.
(5) Set $T_{s+1} = T_s/\gamma$, $s = s + 1$, and go to Step 1.

Here $\gamma > 1$ is the cooling ratio for simulated annealing with a typical value of 1.0025. We must note that there is a larger family of algorithms that perform optimization over manifolds with constraints, of which this algorithm is an example.

6 Experimental Results

In this section we illustrate Algorithm 1 using a number of examples and databases. In each case, we will optimize over a performance function F specified in Section 3 and show improvements over standard choices of dimension-reduction transformations.

(1) **Optimal Subspace for Image-Based Recognition**: The data used in this experiment is taken from the ORL face recognition dataset [4]. This database consists of 40 subjects, each imaged at 10 different facial expressions. In this experiment we used $n = 112 \times 92 = 10304$, and we seek optimal transformations onto space of dimension $d = 5$, using $m_{train} = 5$, and $m_{test} = 5$. Shown in Figure 1 are three example evolutions of $F(X_{s\delta})$ versus s starting from random initial points in $\mathcal{G}_{n,d}$. In each of these case, the algorithm starts from a low performance but reaches optimal performance (100%) eventually.

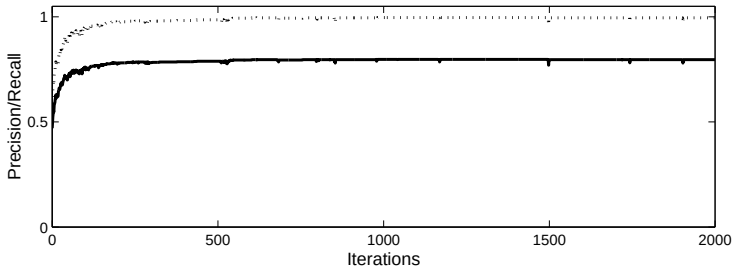


(a) (b) (c)

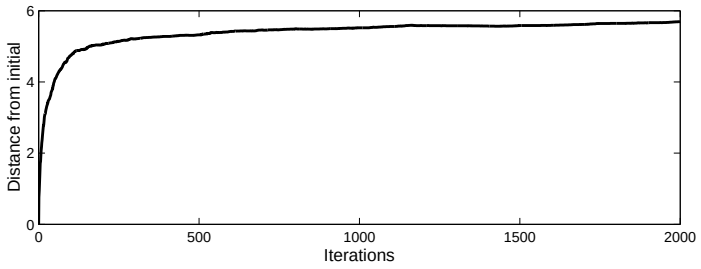
Fig. 1. Performance of $X_{s\delta}$ versus s for three random initial conditions.

[4] http://www.uk.research.att.com/facedatabase.html

(2) **Content-Based Image Retrieval**: We describe a small experiment on ORL database for image retrieval. In this experiment, two images of each person were used as query images, i.e. $m_{qu} = 2$ and $m_{db} = 8$, and the remaining eight images of that were all considered relevant images. Our goal is to search for a $[U] \in \mathcal{G}_{n,d}$ that maximizes F specified in Eqn. 9. Shown in Figure 2, is an example of Algorithm 1 applied to this problem. The top figure plots evolution of $F(X_{s\delta})$ (dotted line) and the average recall measure (solid line) versus s, and shows a steady increase in F till it reaches its maximum value. For this experiment, $d = 20$ and $R = 10$. The lower figure shows the displacement from initial condition by plotting $\|X_{s\delta} - X_0\|$ versus s to emphasize a non-local nature of this search.



(a)



(b)

Fig. 2. Temporal evolution of the optimization algorithm. (a) Plots of retrieval precision (solid line) and the corresponding recall (dotted line). (b) Distance of X_t from X_0.

(3) **Sparse Representations of Images**: In this experiment, we used Algorithm 1 to find sparse representations of images in ORL data, by maximizing F as defined in Eqn. 8. Shown in Figure 3 three examples of this optimization. In each row, the right panel shows evolution of sparseness performance versus the iteration index showing a steady improvement until convergence. Several papers have noted that sparseness by itself is not an interesting criterion; it becomes interesting when combined with an application such as reconstruction or classification. To underscore this point, we have drawn the corresponding recognition performance of the transformations that result from maximization of sparseness performance. These plots point out that recognition performance of sparse representations can be very low. This suggests that a better option is to seek an optimal transformation U under a joint cost function, a function that is a

weighted combination of recognition and sparseness performances. Shown in Figure 4 are some examples of joint optimization over recognition and sparsity performance. Here λ_1 denotes the weight on recognition performance and λ_2 weights the sparsity performance. The three rows present evolutions of performance function (left panel) and the sparsity function (right panel) for three different set of weights.

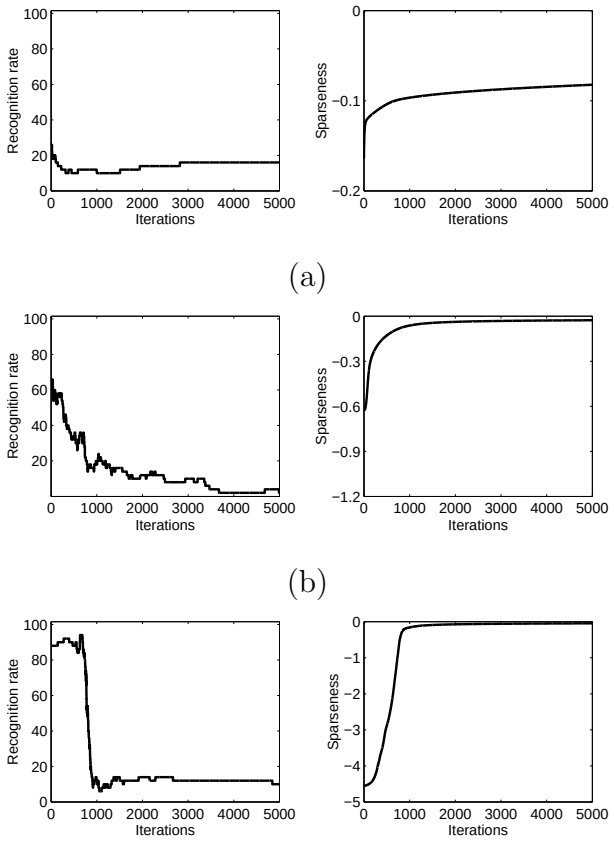


Fig. 3. Evolution of sparseness performance (right panels in each row) on ORL dataset starting at three different initial conditions. (a)X_0=$U_{FastICA}$.(b)X_0=$U_{InfoMaxICA}$.(c)X_0=U_{PCA}. Left panels show the corresponding recognition performance for the same process.

(4) **Generalization to test data**: In this case, we seek those transformations that extend a good performance obtained on the training data to the test data. The idea is to separate the classes maximally in order to improve the generalization performance stated earlier. Shown in Figure 5 is an example of this idea. Algorithm 1 was used to maximally separate the transformed classes (in the training data) according to F given in Eqn. 7. Top left panel shows the evolution of F over the training data (solid line) and the corresponding evaluation of F on a separate test data. Bottom left panel plots the difference between training and test performance; this difference gets smaller as the algorithm proceeds. In the next two

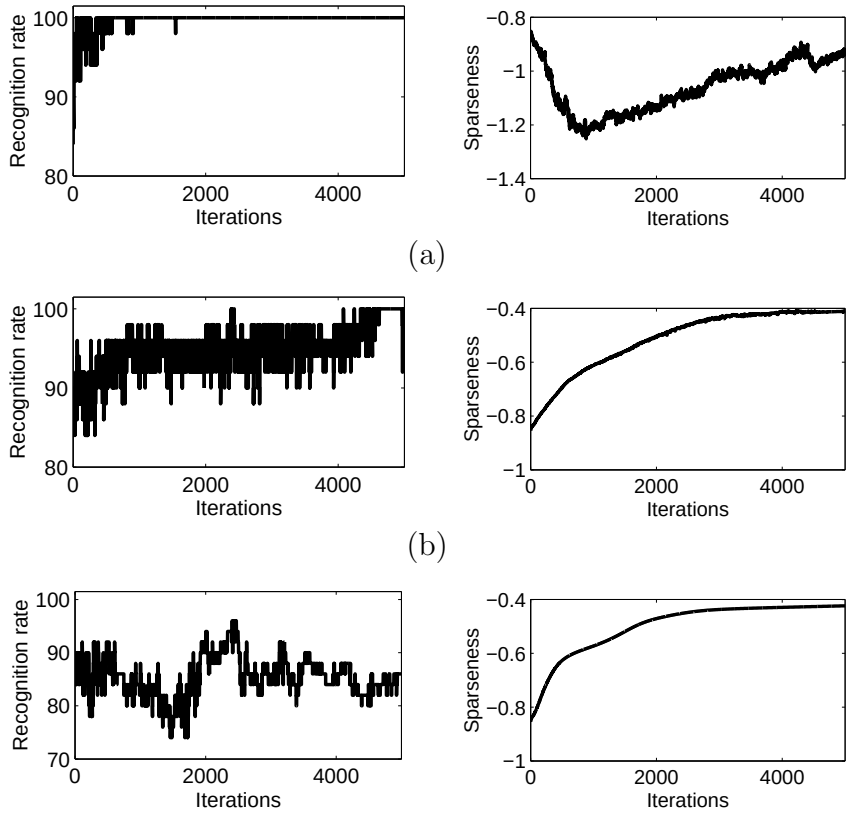
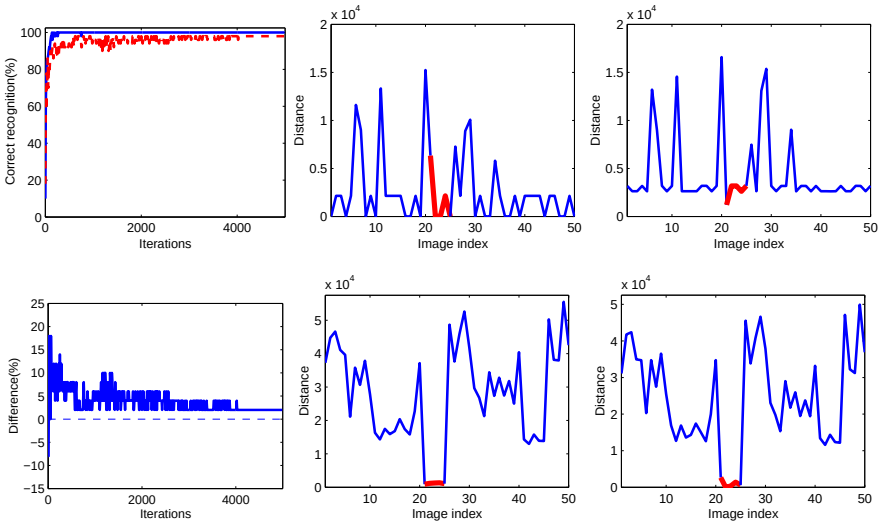


Fig. 4. Joint optimization over recognition and sparsity functions. Left panels plot the recognition performance and right panels display sparsity function. The three rows show results three different weights on the performance functions. (a) $\lambda_1 = 1.0$ and $\lambda_2 = 0.0$. (b) $\lambda_1 = 0.2$ and $\lambda_2 = 0.8$. (c) $\lambda_1 = 0.0$ and $\lambda_2 = 1.0$.

columns, the idea of maximally separating the classes is demonstrated. To middle shows the distance $d(X_1, X_2; U)$ between a training image and the remaining training images for $U = X_0$, a random initial condition. The bottom middle panel shows a similar plot for $X_{5000\delta}$. These plots show that images of the same class came closer together, the images from other classes drifted further apart, during the evolution of $X_{s\delta}$. The right panels plot the same two sets of distances, this time using the test images.

7 Summary

In this paper we have presented an optimization approach for finding a dimension-reduction transformation that is optimal for the application of interest. The basic idea is to define a performance function for that application, that relates an orthonormal transformation to the desired performance, and then to search over the appropriate manifold of orthonormal bases to obtain an op-



(a) (b) (c)

Fig. 5. The search process of a linear representation with better generalization. (a) Top: $F(X_{s\delta})$ on the training set (solid line) and on a separate test set (dotted line) using the nearest neighbor classifier; bottom: the difference between training and test performance (solid line). Here X_0 is obtained using the FastICA algorithm [16]. (b) The distances between a training image and all the training ones using X_0 (top) and $X_{5000\delta}$ (bottom). The thick segment corresponds to the correct class. (c) As in (b) but for a test image from the same class.

timal transformation. Orthonormal bases are elements of a Grassmann or a Stifel manifold, and we utilize differential geometry of these manifolds construct a stochastic gradient search for solving the optimization problem.

Acknowledgements

This research was supported in part by grants NSF DMS-0101429, NMA 201-01-2010, and NSF IIS-0307998.

References

[1] J. B. Tenenbaum, V. Silva, , J. C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (2000) 2319–2323.

[2] S. T. Roweis, L. K. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (2000) 2323–2326.

[3] M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, Neural Computation 15 (6) (2003) 1373–1396.

[4] S. Amari, Natural gradient works efficiently in learning, Neural Computation 10 (1998) 251–276.

[5] S. Fiori, A theory for learning by weight flow on stiefel-grassmann manifold, Neural Computation 13 (2001) 1625–1647.

[6] W. M. Boothby, An Introduction to Differential Manifolds and Riemannian Geometry, Academic Press, Inc., 1986.

[7] M. Kirby, L. Sirovich, Application of the karhunen-loeve procedure for the characterization of human faces, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (1) (1990) 103–108.

[8] M. Turk, A. Pentland, Eigenfaces for recognition, Journal of Cognitive Neuroscience 3 (1991) 71–86.

[9] G. H. Golub, C. F. V. Loan, Matrix Computations, Johns Hopkins University Press, Baltimore, 1989.

[10] P. N. Belhumeur, J. P. Hepanha, D. J. Kriegman, Eigenfaces vs. fisherfaces: Recognition using class specific linear projection, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (7) (1997) 711–720.

[11] P. Comon, Independent component analysis, a new concept?, Signal Processing, Special issue on higher-order statistics 36 (3) (1994) 287–314.

[12] J. A. van Hateren, Independent component filters of natural images compared with simple cells in primary visual cortex, Proc. Royal Statistical Society of London (B) 265 (1998) 359:366.

[13] X. Liu, L. Cheng, Independent spectral representations of images for recognition, Journal of Optical Society of America A 20 (7).

[14] M. S. Bartlett, J. R. Movellan, T. J. Sejnowski, Face recognition by independent component analysis, IEEE Transactions on neural networks 13 (6) (2002) 1450–1464.

[15] A. J. Bell, T. J. Sejnowski, "An information maximization approach to blind separation and blind deconvolution", Neural Computation 7 (1995) 1129–1159.

[16] A. Hyvarinen, Fast and robust fixed-point algorithm for independent component analysis, IEEE Transactions on Neural Networks 10 (1999) 626–634.

[17] X. Liu, A. Srivastava, K. Gallivan, Optimal linear representations of images for object recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence to appear in May.

[18] V. N. Vapnik, The Nature of Statistical Learning Theory, 2nd ed., Springer-Verlag, 2000.

[19] D. J. Field, What is the goal of sensory coding?, Neural Computation 6 (4) (1994) 559–601.

[20] B. A. Olshausen, D. J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, Nature 381 (1996) 607–609.

[21] J. Daugman, Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters, Journal of the Optical Society of America A 2 (7) (1985) 23–26.

[22] A. B. Watson, The cortex transform: rapid computation of simulated neural images, Comp. Vis. Graphics Image Proc. 39 (1987a) 311–327.

[23] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, D. J. Heeger, Shiftable multi-scale transforms, IEEE Trans Information Theory 38 (2) (1992) 587–607.

[24] H. Muller, W. Muller, D. M. Squire, S. Marchand-Maillet, T. Pun, Performance evaluation in content-based image retrieval: overview and proposals, Pattern Recognition Letters 22 (2001) 93–601.

[25] K. Gallivan, A. Srivastava, X. Liu, P. V. Dooren, Efficient algorithms for inferences on grassmann manifolds, in: 12th IEEE Workshop on Statistical Signal Processing, St. Louis, MO, 2003, pp. 301–304.

[26] A. Srivastava, A Bayesian approach to geometric subspace estimation, IEEE Transactions on Signal Processing 48 (5) (2000) 1390–1400.