Spectral Regression for Efficient Regularized Subspace Learning*

Deng Cai
UIUC
dengcai2@cs.uiuc.edu

Xiaofei He
Yahoo!
hex@yahoo-inc.com

Jiawei Han
UIUC
hanj@cs.uiuc.edu

Abstract

Subspace learning based face recognition methods have attracted considerable interests in recent years, including Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Locality Preserving Projection (LPP), Neighborhood Preserving Embedding (NPE) and Marginal Fisher Analysis (MFA). However, a disadvantage of all these approaches is that their computations involve eigen-decomposition of dense matrices which is expensive in both time and memory. In this paper, we propose a novel dimensionality reduction framework, called **Spectral Regression** *(SR), for efficient regularized subspace learning. SR casts the problem of learning the projective functions into a regression framework, which avoids eigen-decomposition of dense matrices. Also, with the regression based framework, different kinds of regularizers can be naturally incorporated into our algorithm which makes it more flexible. Computational analysis shows that SR has only* **linear***-time complexity which is a huge speed up comparing to the* **cubic***-time complexity of the ordinary approaches. Experimental results on face recognition demonstrate the effectiveness and efficiency of our method.*

1. Introduction

Many face recognition techniques have been developed over the past few decades. One of the most successful and well-studied techniques to face recognition is the appearance-based method [15, 22]. When using appearance-based methods, an image of size $n_1 \times n_2$ pixels is usually represented by a vector in an $n_1 \times n_2$-dimensional space. In practice, however, these $n_1 \times n_2$-dimensional spaces are too large to allow robust and fast face recognition. Previous works have demonstrated that the face recognition performance can be improved significantly in lower dimensional linear subspaces [1, 13, 22]. Two of the most popular appearance-based face recognition methods include *Eigenface* [22] and *Fisherface* [1]. Eigenface is based on Principal Component Analysis (PCA) [7]. PCA projects the face images along the directions of maximal variances. It also aims to preserve the Euclidean distances between face images. Fisherface is based on Linear Discriminant Analysis (LDA) [7]. Unlike PCA which is unsupervised, LDA is supervised. When the class information is available, LDA can be used to find a linear subspace which is optimal for discrimination.

Recently there are considerable interest in geometrically motivated approaches to visual analysis. Various researchers (see [2, 18, 21]) have considered the case when the data lives on or close to a low dimensional sub-manifold of the high dimensional ambient space. One hopes then to estimate geometrical and topological properties of the sub-manifold from random points ("scattered data") lying on this unknown sub-manifold. Along this direction, many subspace learning algorithms have been proposed for face recognition. Some popular ones include Locality Preserving Projection (LPP) [13], Neighborhood Preserving Embedding (NPE) [12] and Marginal Fisher Analysis (MFA) [23]. Despite the different motivations of these algorithms, they can be nicely interpreted in a general graph embedding framework [3, 13, 23]. One of the major limitations of these approaches is that their computations involve dense matrices eigen-decomposition which is expensive in both time and memory. Moreover, when the number of features is larger than the number of samples, some additional preprocessing steps (*e.g.*, PCA, SVD) are required to obtain the stable solution of the optimization problem. These preprocessing steps further increase the time cost. Thus, it is difficult to apply these approaches to very high dimensional data of large size.

In this paper, we propose a novel dimensionality reduction framework, called *Spectral Regression* (SR), for efficient regularized subspace learning. The proposed approach is fundamentally based on regression and spectral graph analysis [6]. Specifically, SR decomposes the sub-

*The work was supported in part by the U.S. National Science Foundation NSF IIS-05-13678, NSF BDI-05-15813 and MIAS (a DHS Institute of Discrete Science Center for Multimodal Information Access and Synthesis). Any opinions, findings, and conclusions or recommendations expressed here are those of the authors and do not necessarily reflect the views of the funding agencies.

space learning as a two-step approach: graph embedding for responses learning and regression for projective functions learning. The theoretical analysis shows that when the sample vectors are linearly independent, which is usually the case for small sample size problem, SR can give exactly the same solutions with ordinary subspace learning approaches. While dense matrices eigen-decomposition is avoided in SR which is a huge save of both memory and time.

The specific contributions of this paper include:

- It reviews and provides a unified graph embedding analysis (as well as the computational complexity analysis) of many existing subspace learning algorithms, *e.g.*, LDA, LPP and NPE (Section 2).
- It proposes a novel spectral regression approach to solve the optimization problem of the linear graph embedding, which reduces the cubic-time complexity to linear-time complexity (Section 3).
- We have performed extensive experimental comparisons of our approach and the state-of-the-art approaches, which demonstrate the effectiveness and efficiency of our method. (Section 4).

We summarize our findings and discuss extensions to the current work in Section 5, which concludes the paper.

2. Graph Embedding View of Subspace Learning

In this Section, we provide a general framework of analysis for the existing subspace learning algorithms from the graph embedding viewpoint. Particularly, the computational complexities of these algorithms can be well studied within this framework.

2.1. Graph based Subspace Learning

Suppose we have m face images $\{\mathbf{x}_i\}_{i=1}^m \subset \mathbb{R}^n$, $X = [\mathbf{x}_1, \cdots, \mathbf{x}_m]$. In the past decades, many dimensionality reduction algorithms have been proposed to find a low dimensional representation of $\mathbf{x}_i$. Despite the different motivations of these algorithms, they can be nicely interpreted in a general *graph embedding* framework [3, 13, 23].

Given a graph G with m vertices, each vertex represents a data point. Let W be a symmetric $m \times m$ matrix with W_{ij} having the weight of the edge joining vertices i and j. The G and W can be defined to characterize certain statistical or geometric properties of the data set. The purpose of graph embedding is to represent each vertex of the graph as a low dimensional vector that preserves similarities between the vertex pairs, where similarity is measured by the edge weight.

Let $\mathbf{y} = [y_1, y_2, \cdots, y_m]^T$ be the map from the graph to the real line. The optimal $\mathbf{y}$ is given by minimizing

$$\sum_{i,j}(y_i - y_j)^2 W_{ij}$$

under appropriate constraint. This objective function incurs a heavy penalty if neighboring vertices i and j are mapped far apart. Therefore, minimizing it is an attempt to ensure that if vertices i and j are "close" then y_i and y_j are close as well [10]. With some simple algebraic formulations, we have

$$\sum_{i,j}(y_i - y_j)^2 W_{ij} = 2\mathbf{y}^T L \mathbf{y},$$

where $L = D - W$ is the *graph Laplacian* [6] and D is a diagonal matrix whose entries are column (or row, since W is symmetric) sums of W, $D_{ii} = \sum_j W_{ji}$. Finally, the minimization problem reduces to find

$$\mathbf{y}^* = \arg\min_{\mathbf{y}^T D\mathbf{y}=1} \mathbf{y}^T L \mathbf{y} = \arg\min \frac{\mathbf{y}^T L \mathbf{y}}{\mathbf{y}^T D \mathbf{y}}.$$

The constraint $\mathbf{y}^T D\mathbf{y} = 1$ removes an arbitrary scaling factor in the embedding. Notice that $L = D - W$, it is easy to see that the above optimization problem has the following equivalent variation:

$$\mathbf{y}^* = \arg\max_{\mathbf{y}^T D\mathbf{y}=1} \mathbf{y}^T W \mathbf{y} = \arg\max \frac{\mathbf{y}^T W \mathbf{y}}{\mathbf{y}^T D \mathbf{y}}. \quad (1)$$

The optimal $\mathbf{y}$'s can be obtained by solving the maximum eigenvalue eigen-problem:

$$W\mathbf{y} = \lambda D\mathbf{y}. \quad (2)$$

Many recently proposed manifold learning algorithms, like ISOAMP [21], Laplacian Eigenmap [2], Locally Linear Embedding [18], can be interpreted in this framework with different choice of W.

The graph embedding approach described above only provides the mappings for the graph vertices in the training set. For classification purpose (*e.g.*, face recognition), a mapping for all samples, including new test samples, is required. If we choose a linear function, *i.e.*, $y_i = f(\mathbf{x}_i) = \mathbf{a}^T\mathbf{x}_i$, we have $\mathbf{y} = X^T\mathbf{a}$. Eq. (1) can be rewritten as:

$$\mathbf{a}^* = \arg\max \frac{\mathbf{y}^T W \mathbf{y}}{\mathbf{y}^T D \mathbf{y}} = \arg\max \frac{\mathbf{a}^T X W X^T \mathbf{a}}{\mathbf{a}^T X D X^T \mathbf{a}}. \quad (3)$$

The optimal $\mathbf{a}$'s are the eigenvectors corresponding to the maximum eigenvalue of eigen-problem:

$$XWX^T\mathbf{a} = \lambda XDX^T\mathbf{a}. \quad (4)$$

This approach is called *Linear extension of Graph Embedding* (LGE). With different choices of W, the LGE framework will lead to many popular linear dimensionality reduction algorithms, *e.g.*, LDA, LPP and NPE. We will briefly list the choices of W for these algorithms as follows.

LDA:

Suppose we have c classes and the t-th class have m_t samples, $m_1 + \cdots + m_c = m$. Define

$$W_{ij} = \begin{cases} 1/m_t, & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ both belong to} \\ & \text{the } t\text{-th class;} \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

With such W, it is easy to check that $D = I$. Please see [13], [5] for the detailed derivation.

LPP:

Let $N_k(\mathbf{x}_i)$ denote the set of k nearest neighbors of $\mathbf{x}_i$.

$$W_{ij} = \begin{cases} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}, & \text{if } \mathbf{x}_i \in N_k(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in N_k(\mathbf{x}_i) \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

For supervised case, one can also integrate the label information into W by searching the k nearest neighbors of $\mathbf{x}_i$ among the points sharing the same label with $\mathbf{x}_i$. Please see [13] for the details.

NPE:

Let $N_k(\mathbf{x}_i)$ denote the set of k nearest neighbors of $\mathbf{x}_i$ and M be a $m \times m$ local reconstruction coefficient matrix. M is defined as follows:

For i-th row of M, $M_{ij} = 0$ if $\mathbf{x}_j \notin N_k(\mathbf{x}_i)$. The other M_{ij} can be computed by minimizing the following objective function,

$$\min \|\mathbf{x}_i - \sum_{j \in N_k(\mathbf{x}_i)} M_{ij}\mathbf{x}_j\|^2, \quad \sum_{j \in N_k(\mathbf{x}_i)} M_{ij} = 1.$$

Define

$$W = M + M^T - M^T M \quad (7)$$

and it is easy to check that $D = I$. Please see [12], [23] for the detailed derivation. The label information can also be integrated into W by searching the k nearest neighbors of $\mathbf{x}_i$ among the points share the same label with $\mathbf{x}_i$.

2.2. Computational Analysis

All the above mentioned linear subspace learning algorithms need to solve the eigen-problem in Eqn. (4). To get a stable solution of this eigen-problem, the matrices XDX^T is required to be non-singular [9] which is not true when the number of features is larger than the number of samples. The Singular Value Decomposition (SVD) can be used to solve this problem. Suppose $rank(X) = r$, the SVD decomposition of X is

$$X = U\Sigma V^T$$

where $\Sigma = diag(\sigma_1, \cdots, \sigma_r)$ and $\sigma_1 \geq \cdots \geq \sigma_r > 0$ are the singular values of X, $U \in \mathbb{R}^{n \times r}$, $V =\in \mathbb{R}^{m \times r}$ and $U^TU = V^TV = I$. Let $\widetilde{X} = U^TX = \Sigma V^T$ and $\mathbf{b} = U^T\mathbf{a}$, we have

$$\mathbf{a}^T X W X^T \mathbf{a} = \mathbf{a}^T U\Sigma V^T W V \Sigma U^T \mathbf{a} = \mathbf{b}^T \widetilde{X} W \widetilde{X}^T \mathbf{b}$$

and

$$\mathbf{a}^T X D X^T \mathbf{a} = \mathbf{a}^T U\Sigma V^T D V \Sigma U^T \mathbf{a} = \mathbf{b}^T \widetilde{X} D \widetilde{X}^T \mathbf{b}$$

Now, the objective function in (3) can be rewritten as:

$$\mathbf{b}^* = \arg\max \frac{\mathbf{b}^T \widetilde{X} W \widetilde{X}^T \mathbf{b}}{\mathbf{b}^T \widetilde{X} D \widetilde{X}^T \mathbf{b}},$$

and the optimal $\mathbf{b}$'s are the eigenvectors corresponding to the maximum eigenvalues of eigen-problem:

$$\widetilde{X} W \widetilde{X}^T \mathbf{b} = \lambda \widetilde{X} D \widetilde{X}^T \mathbf{b}. \quad (8)$$

It is clear that $\widetilde{X} D \widetilde{X}^T$ is nonsingular and the above eigen-problem can be stably solved. After we get $\mathbf{b}^*$, the $\mathbf{a}^*$ can be obtained by

$$\mathbf{a}^* = U\mathbf{b}^*. \quad (9)$$

When the sample vectors are centered, SVD is essentially the same to PCA. This approach has been widely used in many subspace learning algorithms (*e.g.*, LDA [1], LPP [13] and NPE [12]) to solve the singularity problem. For clarity, we name this computational approach as SVD+LGE (Linear Graph Embedding).

Now let us analyze the computational complexity of SVD+LGE. We consider the case that the number of features (n) is larger than the number of samples (m), which is usually the case for face recognition. The term *flam* [19], a compound operation consisting of one addition and one multiplication, is used to present operation counts.

The most efficient algorithm to calculate the SVD decomposition requires $\frac{3}{2}m^2n + \frac{9}{2}m^3$ flam [20]. Thus, the time complexity of the SVD+LGE approach measured by flam is at least $\frac{3}{2}m^2n + \frac{9}{2}m^3$, which is cubic-time complexity with respect to m. For large scale high dimensional data, this approach is unlikely to be applied.

Another way to deal with the singularity of XDX^T is to apply the idea of regularization, by adding some constant values to the diagonal elements of XDX^T, as $XDX^T + \alpha I$, for any $\alpha > 0$. It is easy to see that $XDX^T + \alpha I$ is non-singular. This approach is used in LDA which leads to Regularized Discriminant Analysis (RDA) [8]. $XDX^T + \lambda I$ will be a $n \times n$ dense matrix and solving the eigen-problem

in Eqn. (4) requires at least $\frac{9}{2}n^3$ flam. Moreover, the calculation of matrices XWX^T and XDX^T requires at least $2mn^2$ flam. This high computational complexity restricts the regularized subspace learning approach to be applied on high dimensional data.

3. Spectral Regression Framework for Subspace Learning

The high computational cost restricts those popular subspace learning algorithms to be applied to large scale high dimensional data sets. In this section, we describe our approach which can overcome this difficulty.

3.1. Spectral Regression

In order to solve the eigen-problem in Eqn. (4) efficiently, we use the following theorem:

Theorem 1 *Let* $\mathbf{y}$ *be the eigenvector of eigen-problem in Eqn. (2) with eigenvalue* λ*. If* $X^T\boldsymbol{a} = \mathbf{y}$*, then* $\boldsymbol{a}$ *is the eigenvector of eigen-problem in Eqn. (4) with the same eigenvalue* λ*.*

Proof We have $W\mathbf{y} = \lambda D\mathbf{y}$. At the left side of Eqn. (4), replace $X^T\mathbf{a}$ by $\mathbf{y}$, we have

$$XWX^T\mathbf{a} = XW\mathbf{y} = X\lambda D\mathbf{y} = \lambda XD\mathbf{y} = \lambda XDX^T\mathbf{a}$$

Thus, $\mathbf{a}$ is the eigenvector of eigen-problem Eqn. (4) with the same eigenvalue λ. ∎

Theorem (1) shows that instead of solving the eigen-problem in Eqn. (4), the linear projective functions can be obtained through two steps:

1. Solve the eigen-problem in Eqn. (2) to get $\mathbf{y}$.
2. Find $\mathbf{a}$ which satisfies $X^T\mathbf{a} = \mathbf{y}$. In reality, such $\mathbf{a}$ might not exist. A possible way is to find $\mathbf{a}$ which can best fit the equation in the least squares sense:

$$\mathbf{a} = \arg\min_{\mathbf{a}} \sum_{i=1}^{m} (\mathbf{a}^T\mathbf{x}_i - y_i)^2 \tag{10}$$

   where y_i is the i-th element of $\mathbf{y}$.

The advantages of this two-step approach are as follows:

1. The matrix D is guaranteed to be positive definite and therefor the eigen-problem in Eqn. (2) can be stably solved. Moreover, we will show later how this eigen-problem is *trivial* and the eigenvectors $\mathbf{y}$ can be directly obtained with a supervised graph matrix W.
2. The technique to solve the least square problem is already matured [9] and there exist many efficient iterative algorithms (*e.g.*, LSQR [16]) that can handle very large scale least square problems.

In the situation that the number of samples is smaller than the number of features, the minimization problem (10) is *ill posed*. We may have infinitely many solutions to the linear equations system $X^T\mathbf{a} = \mathbf{y}$ (the system is underdetermined). The most popular way to solve this problem is to impose a penalty on the norm of $\mathbf{a}$:

$$\mathbf{a} = \arg\min_{\mathbf{a}} \left(\sum_{i=1}^{m} \left(\mathbf{a}^T\mathbf{x}_i - y_i\right)^2 + \alpha\|\mathbf{a}\|^2 \right) \tag{11}$$

This is so called regularization and is well studied in statistics. The regularized least square is also called ridge regression [11]. The $\alpha \geq 0$ is a parameter to control the amounts of shrinkage. Now we can see the third advantage of the two-step approach:

3 Since the regression is used as a building block, the regularization techniques can be easily incorporated and produce more stable and meaningful solutions, especially when there exist a large number of features [11].

The regularized least squares in Eqn. (11) can be rewritten in the matrix form as:

$$\mathbf{a} = \arg\min_{\mathbf{a}} \left((X^T\mathbf{a} - \mathbf{y})^T(X^T\mathbf{a} - \mathbf{y}) + \alpha\mathbf{a}^T\mathbf{a}\right). \tag{12}$$

Requiring the derivative of right side with respect to $\mathbf{a}$ vanish, we get

$$\begin{aligned} &(XX^T + \alpha I)\mathbf{a} = X\mathbf{y} \\ \Rightarrow\ &\mathbf{a} = (XX^T + \alpha I)^{-1}X\mathbf{y} \end{aligned} \tag{13}$$

When $\alpha > 0$, this regularized solution will not satisfy the linear equations system $X^T\mathbf{a} = \mathbf{y}$ and $\mathbf{a}$ will not be the eigenvector of eigen-problem in Eqn. (4). It is interesting and important to see when (13) gives the exact solutions of eigen-problem (4). Specifically, we have the following theorem:

Theorem 2 *Suppose* $\mathbf{y}$ *is the eigenvector of eigen-problem in Eqn. (2), if* $\mathbf{y}$ *is in the space spanned by row vectors of* X*, the corresponding projective function* $\boldsymbol{a}$ *calculated in Eqn. (13) will be the eigenvector of eigen-problem in Eqn. (4) as* α *deceases to zero.*

Proof See Appendix A. ∎

When the the number of features is larger than the number of samples, the sample vectors are usually linearly independent, *i.e.*, $rank(X) = m$. In this case, we will have a stronger conclusion which is shown in the following Corollary.

Corollary 3 *If the sample vectors are linearly independent,* i.e.*,* $rank(X) = m$*, all the projective functions calculated by Eqn. (13) are the eigenvectors of eigen-problem in Eqn. (4) as* α *deceases to zero. These solutions are identical to those of SVD+LGE in Eqn. (9).*

Proof See Appendix B. ∎

Our above two-step approach essentially performs regression after the spectral analysis of the graph, we called it *Spectral Regression* (SR) [4].

3.2. Eigenvectors of Eigen-problem in Eqn. (2)

Now let us study the eigenvectors of eigen-problem in Eqn. (2). We consider the case that both LPP and NPE construct their graph by incorporating the label information, *i.e.*, searching the k nearest neighbors of $\mathbf{x}_i$ among the points share the same label with $\mathbf{x}_i$.

Without loss of generality, we assume that the data points in $\{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$ are ordered according to their labels. It is easy to check that the matrix W in these three algorithms has a block-diagonal structure

$$W = \begin{bmatrix} W^{(1)} & 0 & \cdots & 0 \\ 0 & W^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & W^{(c)} \end{bmatrix} \tag{14}$$

where c is the number of classes, $W^{(t)} \in \mathbb{R}^{m_t \times m_t}$ and m_t is the number of samples in t-th class. We also have the D as the diagonal matrix. Thus, the eigenvalues and eigenvectors of $W\mathbf{y} = \lambda D\mathbf{y}$ are the union of the eigenvalues and eigenvectors of its blocks (the latter padded appropriately with zeros) [9]:

$$W^{(t)}\mathbf{y}^{(t)} = \lambda D^{(t)}\mathbf{y}^{(t)}.$$

It is straightforward to show that the above eigen-problem has an eigenvector $\mathbf{e}^{(t)} \in \mathbb{R}^{m_t}$ associated with the largest eigenvalue 1, where $\mathbf{e}^{(t)} = [1, 1, \cdots, 1]^T$ [6]. Thus the top c eigenvectors of eigen-problem in Eqn. (2) are

$$\mathbf{y}_t = [\underbrace{0, \cdots, 0}_{\sum_{i=1}^{t-1} m_i}, \underbrace{1, \cdots, 1}_{m_t}, \underbrace{0, \cdots, 0}_{\sum_{i=t+1}^{c} m_i}]^T. \tag{15}$$

These eigenvectors correspond to the same largest eigenvalue 1. Since 1 is a repeated eigenvalue, we could just pick any other c orthogonal vectors in the space spanned by $\{\mathbf{y}_t\}$ in Eqn. (15), and define them to be our c eigenvectors [9]. The vector of all ones is naturally in the spanned space. This vector is useless since the responses of all the data points are the same. In reality, we can pick the vector of all ones as our first eigenvector and use Gram-Schmidt process to get the remaining $c-1$ orthogonal eigenvectors. The vector of all ones can then be removed.

For the W in LDA, we can easily see that all the elements of $W^{(t)}$ are equal to $1/m_t$. Thus the rank of $W^{(t)}$ is 1 and there is only one non-zero eigenvalue which is exactly 1. We have exactly c eigenvectors (or $c-1$ useful eigenvectors after Gram-Schmidt process) with respect to non-zero eigenvalue for eigen-problem in Eqn. (2). For the W in LPP and NPE, we can get more eigenvectors since the rank of $W^{(t)}$ is usually larger than 1. For a c class problem, previous studies [1][12] show that $c-1$ projective functions are usually enough.

Our above analysis shows that when W is constructed by integrating label information, the top $c-1$ eigenvectors of eigen-problem in Eqn. (2) can be directly obtained. Moreover, although the graphs used in LDA, LPP and NPE are different, the top $c-1$ eigenvectors of their graph matrices are the same. Thus the projective functions calculated in SR are the same. By Theorem 2 and Corollary 3, these projective functions are identical to those of SVD+LGE approach in Eqn. (9) when the sample vectors are linearly independent. Our analysis here gives the reason why the three algorithms LDA [1], LPP [13] and NPE [12] achieve similar performance for high-dimensional low sample size problems.

It is easy to check that the values of the i-th and j-th entries of any vector $\mathbf{y}$ in the space spanned by $\{\mathbf{y}_t\}$ in Eqn. (15) are the same as long as $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same class. Thus the i-th and j-th rows of Y are the same, where $Y = [\mathbf{y}_1, \cdots, \mathbf{y}_{c-1}]$. Corollary (3) shows that when the sample vectors are linearly independent, the $c-1$ projective functions of LDA (LPP, NPE) are exactly the solutions of the $c-1$ linear equations systems $X^T\mathbf{a}_t = \mathbf{y}_t$. Let $A = [\mathbf{a}_1, \cdots, \mathbf{a}_{c-1}]$ be the transformation matrix which embeds the data points into the LDA (LPP, NPE) subspace as:

$$A^T X = Y^T.$$

The columns of matrix Y^T are the embedding results of samples in the LDA (LPP, NPE) subspace. Thus, the data points with the same label are corresponding to the same point in the LDA (LPP, NPE) subspace when the sample vectors are linearly independent.

These projective functions are optimal in the sense of separating training samples with different labels. However, they usually overfit the training set thus may not be able to perform well for the test samples, thus the regularization is necessary.

3.3. Computational Complexity Analysis

The SR computation involves two steps: responses generation (calculate the eigenvectors of eigen-problem in Eqn. (2)) and regularized least squares. The cost of the first step is mainly the cost of Gram-Schmidt method, which requires

Table 2. Performance comparisons on PIE

	Error rates (mean±std-dev%)			Computational time (s)		
	LDA	RDA	SR	LDA	RDA	SR
G30/P140	8.8±0.3	5.9±0.3	6.1±0.2	59.37	396.2	17.39
G40/P130	8.6±0.2	5.0±0.2	5.2±0.2	131.2	404.5	20.11
G50/P120	9.3±0.4	4.6±0.3	4.8±0.3	241.3	413.1	22.71
G60/P110	10.1±1.2	4.2±0.2	4.5±0.2	394.9	421.8	25.49
G80/P90	7.5±0.2	3.9±0.2	4.2±0.2	442.1	442.1	31.13
G100/P70	6.2±0.2	3.7±0.2	4.0±0.2	455.4	455.4	35.98
G120/P50	5.6±0.3	3.5±0.2	3.8±0.2	471.6	471.6	41.57

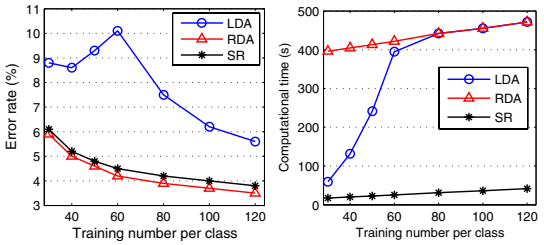


Figure 1. Recognition error rates and computational time of each algorithm on PIE.

Table 1. Computational complexity (operation counts, *flam* [19])

SVD+LGE	$\frac{3}{2}m^2n + \frac{9}{2}m^3$
RLGE	$2mn^2 + \frac{9}{2}n^3$
SR	$2csmn$

m: the number of data samples.
n: the number of features.
We consider the case that $n > m$
c: the number of classes.
s: the number of iterations in LSQR.

$(mc^2 - \frac{1}{3}c^3)$ flam [19]. The regularized least squares problem in Eqn. (11) can be efficiently solved by the iterative algorithm LSQR [16]. In each iteration, LSQR needs to compute two matrix-vector products in the form of $X\mathbf{p}$ and $X^T\mathbf{q}$. The remaining work load of LSQR in each iteration is $3m + 5n$ flam [16]. Thus, the time cost of LSQR in each iteration is $2mn + 3m + 5n$. If LSQR stops after s iterations[1], the time cost is $s(2mn+3m+5n)$. Finally, the total time cost for c projective functions is $cs(2mn + 3m + 5n)$.

We summarize our complexity analysis of SR in Table 1, together with SVD+LGE and Regularized LGE (RLGE). We only show the dominant part of the time cost for simplicity (we assume $m \gg c$). It is clear to see the computational advantage of SR over traditional SVD+LGE and RLGE, especially for the large scale high dimensional data (with large m and n). Please refer our technical report [4, 5] for more detailed analysis.

4. Experimental Results

In this section, we investigate the performance of our proposed SR approach for face recognition. The face recognition task is handled as a multi-class classification problem – we map each test image to a low-dimensional subspace via the embedding learned from training data, and then classify the test data by the nearest centroid criterion.

4.1. Datasets and Compared Algorithms

The CMU PIE and Extended Yale-B face databases are used in our experiments. The CMU PIE face database[2] contains 68 human subjects with 41,368 face images as a whole. The face images were captured by 13 synchronized cameras and 21 flashes, under varying pose, illumination and expression. We choose the five near frontal poses (C05, C07, C09, C27, C29) and use all the images under different illuminations and expressions, thus we get 170 images for each individual.

The Extended Yale-B face database[3] contains 16128 images of 38 human subjects under 9 poses and 64 illumination conditions [14]. In this experiment, we choose the frontal pose and use all the images under different illumination, thus we get 64 images for each person.

All the face images are manually aligned and cropped. The size of each cropped image is 64×64 pixels for PIE and 40×40 pixels for Extended Yale-B, with 256 gray levels per pixel. The features (pixel values) are then scaled to [0,1] (divided by 256).

The image set is then partitioned into the gallery and probe set with different numbers. For ease of representation, Gp/Pq means p images per person are randomly selected for training and the remaining q images are for testing.

Our analysis showed that when the sample vectors are linearly independent, which is usually the case when the number of features is larger than the number of samples, the top $c - 1$ projective functions of LDA, LPP and NPE are essentially the same. Thus, we choose LDA and RDA as the representatives of SVD+LGE and RLGE approaches for experimental comparison due to the space limit.

4.2. Face recognition results

The recognition error rates, as well as the computational time, of different algorithms on PIE and Yale-B databases are reported on the Table 2 and 3 respectively. For each Gp/Pq, we average the results over 20 random splits and report the mean as well as the standard deviation. For both RDA and SR, the regularization parameter α is set to be 0.01 empirically.

The main observations from the performance comparisons include:

[1] LSRQ converges very fast [16]. In our experiments, 30 iterations are enough.

[2] http://www.ri.cmu.edu/projects/project_418.html

[3] http://vision.ucsd.edu/∼leekc/ExtYaleDatabase/ExtYaleB.html

Table 3. Performance comparisons on Extended Yale-B

	Error rates (mean±std-dev%)			Computational time (s)		
	LDA	RDA	SR	LDA	RDA	SR
G10/P54	12.7±1.1	11.6±1.0	12.0±1.2	0.600	22.13	1.029
G20/P44	6.8±0.7	4.2±0.8	4.7±0.9	3.340	23.20	1.269
G30/P34	5.4±0.6	1.8±0.4	2.0±0.5	10.24	23.95	1.509
G40/P24	10.4±1.3	0.9±0.3	1.0±0.4	21.55	24.60	1.751
G45/P19	7.5±1.3	0.6±0.3	0.7±0.3	25.21	25.21	1.876
G50/P14	5.0±0.8	0.4±0.3	0.5±0.3	25.81	25.81	2.008

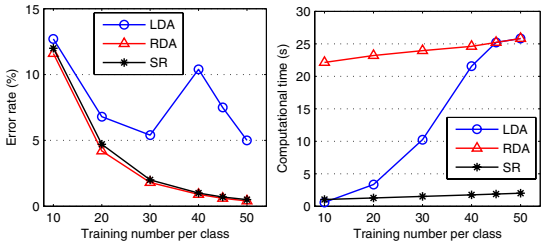


Figure 2. Recognition error rates and computational time of each algorithm on Yale-B.

- The LDA (SVD+LGE approach) seeks the projective functions which are optimal on the training set. It does not consider the possible over-fitting. RDA and SR are regularized versions of LDA. The Tikhonov regularizer is used to control the model complexity. In all the test cases, RDA and SR are significantly better than LDA, which suggests that over-fitting is a very crucial problem which should be addressed in subspace learning approach. It is interesting to find that the over-fitting problem is especially severe when the number of samples and the number of features are close to each other.

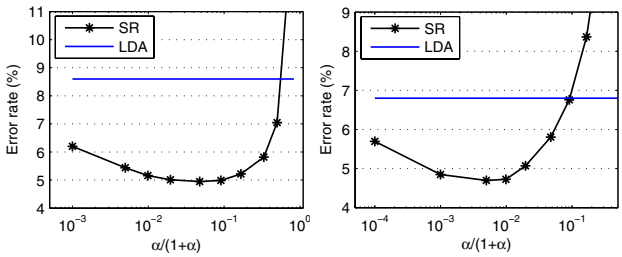
- When the number of samples is smaller than the number of features, LDA (SVD+LGE) uses SVD to solve the singularity problem. This leads to $O(m^3)$ time complexity. RDA (RLGE approach) directly solves the eigen-problem of $n \times n$ dense matrix, which leads to $O(n^3)$ time complexity. SR only needs to solve $c-1$ regularized least squares which is very efficient (with $O(mn)$ time complexity).

- Considering both accuracy and efficiency, SR is the best choice among three of the compared approaches. It provides an efficient and effective regularized subspace learning solution for large scale data sets.

4.3. Model selection for SR

The $\alpha \geq 0$ is an essential parameter in our SR approach (as well as the regularized LGE approaches) which controls the smoothness of the estimator. Our theoretical analysis showed that when the sample vectors are linearly independent, SR provides the same solution as the SVD+LGE approach as α decreases to zero. We empirically set it to be 0.01 in the previous experiments. In this subsection, we try to examine the impact of parameter α on the performance of SR.

Figure (3) shows the performance of SR as a function of the parameter α. For convenience, the X-axis is plotted as $\alpha/(1+\alpha)$ which is strictly in the interval $[0, 1]$. It is easy to see that SR can achieve significantly better performance than LDA over a large range of α. Thus, the parameter selection is not a very crucial problem in SR algorithm.



(a) PIE 40 Train (b) Yale-B 20 Train

Figure 3. Model selection for SR. The curve shows the error rates of SR with respect to $\alpha/(1+\alpha)$. The solid line shows the error rate of LDA.

5. Conclusions

In this paper, we propose a new regularized subspace learning framework called *Spectral Regression* (SR). Our framework is developed from a graph embedding viewpoint of dimensionality reduction algorithms. It combines the spectral graph analysis and regression to provide an efficient and effective approach for regularized subspace learning problem. Specifically, SR only needs to solve a set of regularized least squares problems and there is no eigenvector computation involved, which is a huge save of both time and memory (from **cubic** time complexity to **linear** time complexity). Many recently proposed popular linear subspace learning algorithms, *e.g.*, LDA [1], LPP [13], and NPE [12] can be interpreted as the linear extensions of specific graph embedding. Thus, all these algorithms can be fit into SR framework and their optimization problems can be efficiently solved. Extensive experimental results show that our method consistently outperforms the ordinary SVD+LGE (linear graph embedding) and regularized LGE approaches considering both effectiveness and efficiency.

References

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

[2] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14*. 2001.

[3] M. Brand. Continuous nonlinear dimensionality reduction by kernel eigenmaps. In *International Joint Conference on Artificial Intelligence*, 2003.

[4] D. Cai, X. He, and J. Han. Spectral regression for dimensionality reduction. Technical report, Computer Science Department, UIUC, UIUCDCS-R-2007-2856, May 2007.

[5] D. Cai, X. He, and J. Han. SRDA: An efficient algorithm for large scale discriminant analysis. Technical report, Computer Science Department, UIUC, UIUCDCS-R-2007-2857, May 2007.

[6] F. R. K. Chung. *Spectral Graph Theory*, volume 92 of *Regional Conference Series in Mathematics*. AMS, 1997.

[7] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, Hoboken, NJ, 2nd edition, 2000.

[8] J. H. Friedman. Regularized discriminant analysis. *Journal of the American Statistical Association*, 84(405):165–175, 1989.

[9] G. H. Golub and C. F. V. Loan. *Matrix computations*. Johns Hopkins University Press, 3rd edition, 1996.

[10] S. Guattery and G. L. Miller. Graph embeddings and laplacian eigenvalues. *SIAM Journal on Matrix Analysis and Applications*, 21(3):703–723, 2000.

[11] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag, 2001.

[12] X. He, D. Cai, S. Yan, and H.-J. Zhang. Neighborhood preserving embedding. In *Proc. Int. Conf. Computer Vision (ICCV'05)*, 2005.

[13] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

[14] K. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on PAMI*, 27(5):684–698, 2005.

[15] H. Murase and S. K. Nayar. Visual learning and recognition of 3-d objects from appearance. *International Journal of Computer Vision*, 14(1):5–24, 1995.

[16] C. C. Paige and M. A. Saunders. Algorithm 583 LSQR: Sparse linear equations and least squares problems. *ACM Transactions on Mathematical Software*, 8(2):195–209, June 1982.

[17] R. Penrose. A generalized inverse for matrices. In *Proceedings of the Cambridge Philosophical Society*, volume 51, pages 406–413, 1955.

[18] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[19] G. W. Stewart. *Matrix Algorithms Volume I: Basic Decompositions*. SIAM, 1998.

[20] G. W. Stewart. *Matrix Algorithms Volume II: Eigensystems*. SIAM, 2001.

[21] J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[22] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[23] S. Yan, D. Xu, B. Zhang, and H.-J. Zhang. Graph embedding: A general framework for dimensionality reduction. In *Proc. CVPR 2005*, 2005.

Appendix

A. Proof of Theorem 2

Proof Suppose $rank(X) = r$, the SVD decomposition of X is

$$X = U\Sigma V^T$$

where $\Sigma = diag(\sigma_1, \cdots, \sigma_r)$, $U \in \mathbb{R}^{n\times r}$, $V \in \mathbb{R}^{m\times r}$ and we have $U^TU = V^TV = I$. The $\mathbf{y}$ is in the space spanned by row vectors of X, therefor, $\mathbf{y}$ is in the space spanned by column vectors of V. Thus, $\mathbf{y}$ can be represented as the linear combination of the column vectors of V. Moreover, the combination is unique because the column vectors of V are linear independent. Suppose the combination coefficients are $b_1, \cdots, b_r$. Let $\mathbf{b} = [b_1, \cdots, b_r]^T$, we have:

$$V\mathbf{b}=\mathbf{y} \;\Rightarrow\; V^TV\mathbf{b}=V^T\mathbf{y} \;\Rightarrow\; \mathbf{b}=V^T\mathbf{y} \;\Rightarrow VV^T\mathbf{y}=\mathbf{y} \quad (16)$$

To continue our proof, we need introduce the concept of pseudo inverse of a matrix [17], which we denote as $(\cdot)^+$. Specifically, pseudo inverse of the matrix X can be computed by the following two ways:

$$X^+ = V\Sigma^{-1}U^T$$

and

$$X^+ = \lim_{\alpha\to 0}(X^TX + \alpha I)^{-1}X^T$$

The above limit exists even if X^TX is singular and $(X^TX)^{-1}$ does not exist [17]. Thus, the regularized least squares solution in Eqn. (13)

$$\mathbf{a} = \left(XX^T + \alpha I\right)^{-1}X\mathbf{y} \stackrel{\alpha\to 0}{=} (X^T)^+\mathbf{y} = U\Sigma^{-1}V^T\bar{\mathbf{y}}$$

Combine with the equation in Eqn. (16), we have

$$X^T\mathbf{a} = V\Sigma U^T\mathbf{a} = V\Sigma U^TU\Sigma^{-1}V^T\mathbf{y} = VV^T\mathbf{y} = \mathbf{y}$$

By Theorem (1), $\mathbf{a}$ is the eigenvector of eigen-problem in Eqn. (4). ∎

B. Proof of Corollary 3

Proof The matrices W and D are of size $m \times m$ and there are m eigenvectors $\{\mathbf{y}_j\}_{j=1}^m$ of eigen-problem (2). Since $rank(X) = m$, all these m eigenvectors $\mathbf{y}_j$ are in the space spanned by row vectors of X. By Theorem (2), all m corresponding $\mathbf{a}_j$ of SR in Eqn (13) are eigenvectors of eigen-problem in Eqn. (4) as α decreases to zero. They are

$$\mathbf{a}_j^{SR} = U\Sigma^{-1}V^T\mathbf{y}_j.$$

Consider the eigen-problem in Eqn. (8), since the m eigenvectors $\mathbf{y}_j$ are also in the space spanned by row vectors of $\widetilde{X} = U^TX = \Sigma V^T$, eigenvector $\mathbf{b}_j$ will be the solution of linear equations system $\widetilde{X}^T\mathbf{b}_j = \mathbf{y}_j$. The row vectors of $\widetilde{X} = \Sigma V^T$ are linearly independent, thus $\mathbf{b}_j$ is unique and

$$\mathbf{b}_j = \Sigma^{-1}V^T\mathbf{y}_j.$$

Thus, the projective functions of SVD+LGE

$$\mathbf{a}_j^{SVD+LGE} = U\mathbf{b}_j = U\Sigma^{-1}V^T\mathbf{y}_k = \mathbf{a}_j^{SR}$$

∎