Non-iterative Generalized Low Rank Approximation of Matrices

Jun Liu, Songcan Chen[*]

Dept of Computer Science & Engineering, Nanjing University of Aeronautics & Astronautics

Nanjing, 210016, P.R. China

Abstract: As an extension to 2DPCA, Generalized Low Rank Approximation of Matrices (GLRAM) applies two-sided (i.e., the left and right) rather than single-sided (i.e., the left or the right alone) linear projecting transform(s) to each 2D image for compression and feature extraction. Its advantages over 2DPCA include higher compression ratio and superior classification performance etc. However, GLRAM can *only* adopt an iterative rather than analytical approach to get the left and right projecting transforms and lacks a criterion to automatically determine the dimensionality of the projected matrix. In this paper, a novel non-iterative GLRAM (NIGLRAM) is proposed to overcome the above shortcomings. Experimental results on ORL and AR face datasets and COIL-20 object dataset show that NIGLRAM can get not only so-needed closed-form transforms but also comparable performance to GLRAM.

Keywords: Two-Dimensional Principal Component Analysis (2DPCA); Generalized Low Rank Approximation of Matrices (GLRAM); Non-Iterative GLRAM (NIGLRAM); Feature Extraction.

1. Introduction

As opposed to conventional Principal Component Analysis (PCA) (Turk and Pentland, 1991) manipulating on 1D vectors to extract features, Two-Dimensional Principal Component Analysis (2DPCA) (Yang et al, 2004) is directly based on 2D matrix patterns $A_i \in \Re^{r\times c}$, for $i = 1, 2, \cdots, n$, with n being the number of training images, and r and c their corresponding numbers of rows and columns. Thus 2DPCA has two important advantages over PCA (Yang et al, 2004): 1) easier to evaluate the image covariance matrix $G_t = \sum_{i=1}^{n} (A_i - \bar{A})^T (A_i - \bar{A})$ accurately, where $\bar{A} = \sum_{i=1}^{n} A_i$ is the sample mean image; and 2) less time is required to determine the corresponding eigenvectors

[*] Corresponding author: Tel: +86-25-84892852, Fax: +86-25-84498069. Email: s.chen@nuaa.edu.cn

$U \in \Re^{c \times l}$, whose columns are the l eigenvectors of G_t corresponding to the first l largest eigenvalues in a decreasing order. Wang et al (2005) pointed out that 2DPCA is related to an existing feature extraction method, block-based PCA (Kim et al, 2003) and more importantly is equivalent to its special case, line-based PCA. Such revealed equivalence relationship makes 2DPCA better to be understood from the viewpoint of the existing block-based PCA method. Furthermore, their experimental results on face dataset showed that 2DPCA (or equivalently line-based PCA) achieves higher classification accuracy than the typical rectangle block based PCA, which verifies the effectiveness of 2DPCA. However, 2DPCA applies linear transformation U only on the single (right here) side of each matrix data, i.e., $Y_i = A_i U$, to obtain its compressed version Y_i, leading to lower compression ratio (Yang et al, 2004; Ye, 2004). By simultaneously applying two-sided linear transformations $L \in \Re^{r \times l_1}$ and $R \in \Re^{c \times l_2}$ with orthonormal columns to each image data, i.e., $D_i = L^T A_i R$ to obtain its corresponding compressed version D_i, GLRAM improved both compression ratio and classification accuracy (Ye, 2004). However, GLRAM can *only* adopt an iterative rather than analytical approach to get the left and right (locally optimal) projecting transforms, L and R, and at the same time, lacks a criterion to automatically determine the dimensionalities of the projected matrices. In this paper, a non-iterative GLRAM (NIGLRAM) is proposed to overcome these shortcomings.

Here it is necessary for us to mention a more recent work by (Liang and Shi, 2005) in which the authors argued and claimed that they revealed the optimality property of GLRAM and accordingly found its analytical solution. However, as argued in (Liu and Chen, technical report), their arguments are problematic in that 1) the revealed optimality property of GLRAM is incorrect and 2) the accordingly found analytical solution is not a real analytical solution to GLRAM.

The rest of the paper is organized as follows. We review GLRAM in section 2, and then describe our NIGLRAM in section 3. In section 4, we present experimental results on ORL and AR face datasets, and in section 5 we draw a conclusion of this paper.

2. Review of GLRAM

GLRAM aims to compute two matrices $L \in \Re^{r \times l_1}$ and $R \in \Re^{c \times l_2}$ with orthonormal columns, such that $D_i = L^T A_i R$ is the compressed version of A_i, and $\tilde{A}_i = LD_iR^T$ its approximation. Mathematically, we can formulate this as the following minimization problem: computing optimal *L* and *R*, by

$$\min_{\substack{L\in R^{r\times l_1}:L^TL=I_{l_1}\\ R\in R^{c\times l_2}:R^TR=I_{l_2}}} \sum_{i=1}^{n} \left\| A_i - \tilde{A}_i \right\|_F^2 \tag{1}$$

where $\|.\|_F$ is the *Frobenius* norm. (1) is equivalent to the following maximization problem (Ye, 2004):

$$\max_{\substack{L\in R^{r\times l_1}:L^TL=I_{l_1}\\ R\in R^{c\times l_2}:R^TR=I_{l_2}}} \sum_{i=1}^{n} \left\| L^T A_i R \right\|_F^2 \tag{2}$$

Since there are no closed-form solutions to *L* and *R* for the maximization of (2), GLRAM, instead, employs an alternative iterative optimization way (Ye, 2004), concretely,

1) For a given *R*, *L* maximizes $\sum_{i=1}^{n} \left\| L^T A_i R \right\|_F^2 = \sum_{i=1}^{n} trace(L^T A_i RR^T A_i^T L) = trace(L^T M_L L)$ under the constraint of $L^T L = I_{l_1}$, where $M_L = \sum_{j=1}^{n} A_j RR^T A_j^T$. By means of Lagrangian multiplier method, the maximization of $trace(L^T M_L L)$ under the above constraint leads to the following eigenvalue problem

$$M_L L = L\Lambda \tag{3}$$

where $\Lambda = diag(\eta_1, \eta_2, \cdots, \eta_{l_1})$ is M_L non-negative eigenvalues due to the positive semi-definiteness of M_L and *L* is a transform matrix consisted of the corresponding eigenvectors. By (2) and (3), we have $trace(L^T M_L L) = trace(L^T L\Lambda) = \sum_{i=1}^{l_1} \eta_i$, thus for a given *R*, we can select the l_1 eigenvectors of M_L corresponding to the largest l_1 eigenvalues.

2) Similarly, for a given *L*, we can also obtain *R* which consists of the l_2 eigenvectors of $M_R = \sum_{j=1}^{n} A_j^T LL^T A_j$ corresponding to the largest l_2 eigenvalues.

3) Initialize *L* with a matrix L_0, and repeat 1) and 2) until *RMSRE* defined as $RMSRE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \left\| A_i - LD_iR^T \right\|}$ converges. It is easy to prove that the so-obtained *L* and *R* are just

local maxima of (2).

The pseudo-code for computing L and R by GLRAM is given in **Algorithm GLRAM**.

Algorithm GLRAM

Input: Matrices $\{A_i\}_{i=1}^{n}$, l_1, l_2

Output: Matrices L and R

```
1. Obtain initial $L_0$ and set $i$ to 1
2. While not convergent
3.   Form matrix $M_R = \sum_{j=1}^{n} A_j^T L_{i-1} L_{i-1}^T A_j$
4.   Compute $R_i$ which is the $l_2$ eigenvectors of $M_R$ corresponding to the largest $l_2$ eigenvalues
5.   Form matrix $M_L = \sum_{j=1}^{n} A_j R_i R_i^T A_j^T$
6.   Compute $L_i$ which is the $l_1$ eigenvectors of $M_L$ corresponding to the largest $l_1$ eigenvalues
7.   $i=i+1$
8. End while
9. Set $L$ to $L_{i-1}$, and $R$ to $R_{i-1}$
```

3. Proposed NIGLRAM

It has been shown that 2DPCA is better in classification performance and compression than PCA (Yang et al, 2004) and GLRAM is further better than 2DPCA (Ye, 2004). However, both 2DPCA and GLRAM have some disadvantages. 2DPCA applies linear transformation only on the single side of each matrix data, such a transformation yields lower compression ratio and high storage cost (Yang et al, 2004; Ye, 2004) and on the other hand, GLRAM can *only* adopt an iterative rather than analytical approach for computing L and R, and more importantly, lacks a optimization criterion to automatically determine l_1 of L and l_2 of R. In this paper, a non-iterative GLRAM (NIGLRAM) is proposed to overcome GLRAM's disadvantages and as a result, a closed-form solution is analytically obtained by optimizing alternative objective functions. By a closed form solution, we mean here that the projection matrices L and R can be separately obtained in a closed form in term of their respective reconstruction error criteria. The contributions of the proposed algorithm include 1) the speedup in computation not at the cost of classification accuracy and 2) proposing a criterion to automatically determine l_1 of L and l_2 of R, as done in the traditional PCA.

Now, we formulate our NIGLRAM as follows:

Step1: Use the left linear transformation $L \in \Re^{r\times l_1}$ with orthonormal columns to compress a given matrix A_i and get its compressed version $B_i = L^T A_i$. Let $\tilde{A}_i^L = LB_i$ be an approximation or reconstruction of A_i. Now we can obtain L by minimizing the reconstruction error as defined below:

$$\min_{L\in R^{r\times l_1}:L^T L=I_{l_1}} \sum_{i=1}^{n} \left\| A_i - \tilde{A}_i^L \right\|_F^2 \tag{4}$$

Following similar optimization procedure as that in GLRAM, $L \in \Re^{r\times l_1}$ consisting of the eigenvectors of the matrix $N_L = \sum_{i=1}^{n} A_i A_i^T$ corresponding to the first l_1 largest eigenvalues is the solution to (4). Here, l_1 is a user specified value or a value determined by parameter θ_1 according to the following inequality:

$$\sum_{i=1}^{l_1} \lambda_i \Big/ \sum_{j=1}^{r} \lambda_j \geq \theta_1 (>0) \tag{5}$$

where $\lambda_1, \lambda_2, \cdots, \lambda_r$ are N_L's nonnegative eigenvalues arranged in a decreasing order. The determination of l_1 here is reminiscent of what is done in PCA for selecting the number of the principal components, and as a result easy for both understanding and employment.

Step 2: On the basis of Step 1, we use the right linear transformation $R \in \Re^{c\times l_2}$ with orthonormal columns to compress B_i, and get its compressed matrix $D_i = B_i R$. Let $\tilde{B}_i = D_i R^T$ be an approximation or reconstruction of B_i. Similarly, we can obtain R by minimizing the reconstruction error as defined below:

$$\min_{R\in R^{c\times l_2}:R^T R=I_{l_2}} \sum_{i=1}^{n} \left\| B_i - \tilde{B}_i \right\|_F^2 \tag{6}$$

And R consisting of the l_2 eigenvectors of matrix $N_R = \sum_{i=1}^{n} B_i B_i^T = \sum_{i=1}^{n} A_i^T L L^T A_i$ corresponding to the first l_2 largest eigenvalues is the solution to (6). Hear, l_2 can also be a specified value or a value determined by parameter θ_2 according to the following inequality:

$$\sum_{i=1}^{l_2}\mu_i \Big/ \sum_{j=1}^{c}\mu_j \geq \theta_2 (>0) \quad (7)$$

where $\mu_1, \mu_2, \cdots, \mu_c$ are N_R's nonnegative eigenvalues arranged in a decreasing order. In addition, the determination of l_2 here is also similar to what is done in PCA for selecting the number of the principal components, and as a result easy for both understanding and employment.

Step 3: Combining the compression results in the previous two steps, we immediately get A_i's final compressed version $D_i = L^T A_i R$, and its reconstructed data is $\tilde{A}_i = LD_iR^T = LL^T A_i RR^T$.

The pseudo-code for computing L and R by NIGLRAM is given in **Algorithm NIGLRAM.**

Algorithm NIGLRAM

Input: Matrices $\{A_i\}_{i=1}^{n}$, l_1, l_2 or $\theta_1 = \theta_2 = \sqrt{1 - NMSE}$

Output: Matrices L and R

```
1. Form matrix $N_L = \sum_{i=1}^{n} A_i A_i^T$
2. Compute $L$ which is the $l_1$ eigenvectors of $N_L$ corresponding to the first largest $l_1$ eigenvalues, $l_1$ is a given value, or otherwise, $l_1$ is a value determined by (5)
3. Form matrix $N_R = \sum_{i=1}^{n} A_i^T LL^T A_i$
4. Compute $R$ which is the $l_2$ eigenvectors of $N_R$ corresponding to the first largest $l_2$ eigenvalues, $l_2$ is a given value, or otherwise, $l_2$ is a value determined by (7)
```

Our NIGLRAM is in fact a three-step analytical algorithm, namely, we compress A_i to B_i in the first step, then compress B_i to D_i in the second step, and finally we combine the compression results of the previous two steps to get A_i's final compressed data and the reconstructed data. Unlike GLRAM which directly optimizes (2), our NIGLRAM approximately optimizes it by minimizing (4) and (6) sequentially instead, and thus gets a non-iterative algorithm for solving L and R. For better comparing GLRAM and NIGLRAM from the theoretical point of view, let us look at the following two inequalities:

$$\sum_{i=1}^{n} \left\| L^T A_i R \right\|_F^2 = trace(R^T (\sum_{i=1}^{n} A_i^T L L^T A_i) R) \le \theta_2 trace(\sum_{i=1}^{n} A_i^T L L^T A_i) \quad (8)$$

$$trace(\sum_{i=1}^{n} A_i^T L L^T A_i) = trace(L^T (\sum_{i=1}^{n} A_i A_i^T) L) \le \theta_1 trace(\sum_{i=1}^{n} A_i A_i^T) \quad (9)$$

where θ_1 and θ_2 are defined in (5) and (7) respectively.

From (8), we know that maximizing (2) can be decomposed to two sequential steps: 1) calculate the L that maximizes $S = \theta_2 N_R$ (note that θ_2 is a function of L); and 2) calculate the R that makes the inequality in (8) be equality. The realization of the first step is relatively difficult in that we can not explicitly express θ_2 as a function of L. Different from GLRAM which employs iterative optimizing procedure, our NIGLRAM optimizes S approximately by calculating L that maximizes N_R instead, i.e., calculating L that makes the inequality in (9) meet equality. According to the matrix theory (Golub and Van Loan, 1996), the L and R calculated in our NIGLRAM obviously make the inequalities in (8) and (9) become equalities. Let L^* and L^{**} be the left transformation matrices respectively calculated by GLRAM and NIGLRAM, we have the following inequalities:

$$S(L^*) = \theta_2(L^*) N_R(L^*) \le N_R(L^*) \quad (10)$$

$$S(L^{**}) = \theta_2(L^{**}) N_R(L^{**}) \le S(L^*) \quad (11)$$

$$N_R(L^*) \le N_R(L^{**}) \quad (12)$$

from which, we can easily get:

$$\theta_2(L^{**}) N_R(L^*) \le S(L^{**}) \le S(L^*) \le N_R(L^*) \quad (13)$$

It is obvious that when $\theta_2(L^{**})$ is near 1, NIGLRAM will achieve near objective function value to GLRAM.

Mathematically, if L and R are the local optima of Eq (2), they should respectively satisfy the following two equations:

$$(\sum_{i=1}^{n} A_i R R^T A_i^T) L = L \Lambda_L \quad (14)$$

$$(\sum_{i=1}^{n} A_i^T L L^T A_i) R = R \Lambda_R \quad (15)$$

Therefore in terms of (14-15), R obtained by NIGLRAM is exactly a local optimum in the sense of

(2) while L is not.

The time complexity for performing NIGLRAM is $O(n(r \times c)^{3/2})$, while that for GLRAM is $O(I(r+c)^2 \max(l_1, l_2)n)$ (Ye, 2004), where I is its iteration times. As will be shown in section 4, NIGLRAM is far faster than GLRAM, thus much computation time can be saved, especially in determining their individual l_1 and l_2. In GLRAM, it is difficult to determine the optimal l_1 and l_2 theoretically, and thus extensive trial-and-error experiments have to be repeatedly performed to select appropriate l_1 and l_2. But in NIGLRAM, by first performing its step 1 and 2 *only once*, we can get all eigenvalues and their corresponding eigenvectors, and then construct so-needed Ls according to different specified l_1s or θ_1s, and in turn for a given l_1 or θ_1, Rs are similarly constructed according to specified l_2s or θ_2s. It is an analytical solution that leads NIGLRAM to time-saving.

Furthermore, as are shown in (5) and (7), θ_1 and θ_2 determine l_1 and l_2, respectively; and on the other hand, the parameters θ_1 and θ_2 are related to the normalized mean square error (*NMSE*) defined as

$$NMSE = \sum_{i=1}^{n} \left\| A_i - \tilde{A}_i \right\|_F^2 \Big/ \sum_{i=1}^{n} \left\| A_i \right\|_F^2 \tag{16}$$

Thus we can automatically determine l_1 and l_2 by *NMSE criterion*. *NMSE* is a commonly used criterion to measure the reconstruction quality, so using *NMSE* to automatically determine l_1 and l_2 makes the compression easier controllable than using l_1 and l_2 directly. We show the relationship between θ_1, θ_2 and *NMSE* as below:

Assume L and R to be the transform matrices computed in NIGLRAM as described before, we can get the following inequality according to the matrix theory (Golub and Van Loan, 1996).

$$\begin{aligned} \sum_{i=1}^{n} \left\| A_i - \tilde{A}_i \right\|_F^2 &= \sum_{i=1}^{n} \left\| A_i - LL^T A_i RR^T \right\|_F^2 = trace(\sum_{i=1}^{n} A_i^T A_i) - trace(R^T \ (\sum_{i=1}^{n} A_i^T LL^T A_i) \ R) \\ &\le \sum_{i=1}^{n} \left\| A_i \right\|_F^2 - \theta_2 \times trace(\sum_{i=1}^{n} A_i^T LL^T A_i) \\ &\le \sum_{i=1}^{n} \left\| A_i \right\|_F^2 - \theta_2 \theta_1 \times trace(\sum_{i=1}^{n} A_i A_i^T) = (1 - \theta_1 \theta_2) \sum_{i=1}^{n} \left\| A_i \right\|_F^2 \end{aligned} \tag{17}$$

Thus we obtain the inequality

$$NMSE \le (1 - \theta_1 \theta_2) \tag{18}$$

which clearly shows the relationship among θ_1, θ_2 and *NMSE*. For simplicity, we set $\theta_1 = \theta_2$ in the experiments here, and let $\theta_1 = \theta_2 = \sqrt{1-NMSE}$ for a given *NMSE*. Thus, we can use *NMSE* as a criterion to automatically determine l_1 and l_2, while GLRAM has no such criterion.

Incidentally, it is worthy pointing out that: 1) our methodology in NIGLRAM can also similarly be applied to extend GLRAM+SVD (Ye, 2004) which is an extension of GLRAM and exhibits good reconstruction performance. However, since the focus of this paper is on designing a non-iterative form of GLRAM, such extensions are beyond the topic of this paper and thus omitted here. 2) in NIGLRAM, we first determine *L* and then *R*, then a nature question is that "can we first determine *R* and then *L*?". The answer is positive. In fact, we can rewrite (8) and (9) as follows:

$$\sum_{i=1}^{n}\left\|L^T A_i R\right\|_F^2 = trace(L^T(\sum_{i=1}^{n} A_i RR^T A_i^T)L) \le \alpha_1 trace(\sum_{i=1}^{n} A_i RR^T A_i^T) \tag{19}$$

$$trace(\sum_{i=1}^{n} A_i RR^T A_i^T) = trace(R^T(\sum_{i=1}^{n} A_i^T A_i)R) \le \alpha_2 trace(\sum_{i=1}^{n} A_i^T A_i) \tag{20}$$

where α_1 and α_2 are respectively defined in the similar way to the θ_1 and θ_2. Then similar to NIGLRAM, *R* and *L* can be analytically obtained. Considering the fact that first determining *R* and then *L* is theoretically similar to NIGLRAM, we will only dwell on NIGLRAM in the following experiment parts.

4. Experimental results

4.1 Dataset description and experiment setting

ORL face dataset consists of 400 different grey scale images of 40 different persons, with a resolution of 112×92. The main challenge on this dataset is of pose and expression variation.

AR face dataset (Martinez and Benavente, 1998) is a large face dataset containing over 3200 165×120 color images of frontal images of faces of 126 subjects. In our experiment on AR, we use a subset of AR, which contains 700 face images of 100 persons with seven images each. The seven images are selected from the first session, mainly subject to both expression and light variations.

We use *m*-fold-cross-validation for experiments on ORL and AR. More specifically, we divide the given dataset into *m* non-overlapping subsets with equal size. Then we perform the training and testing *m* times, each time leaving out one of the subsets for testing, and using the rest subsets for training. As a consequence, the experimental results reported in this article are the average ones over the m times. Here, the value of m is 10 for ORL, and 7 for AR, respectively.

COIL-20 is an object database from the Columbia Object Image Library and contains 1440 images of 20 objects, with each object having 72 128×128 images that are numbered from 0 to 71. Images of the objects are taken at pose interval of 5 degree, which corresponds to 72 poses per object. In our experiments on COIL-20, we use those images numbered 0, 4, 8, …, 68 (at interval of 4) for training and the rest for testing.

When reporting classification performance, we employ two classifiers: 1-Nearest-Neighbor (1NN) and Nearest Mean (NM), and both classifiers are based on Euclidean distance.

4.2 On optimizing *S* approximately

In this subsection, we carry out experiment on ORL face dataset, and set $l_1= l_2=20$. We perform the NIGLRAM algorithm to calculate its left projection matrix L^{**}, and calculate the eigenvalues of matrix $\sum_{i=1}^{n} A_i^T LL^T A_i$. Experiments consistently show that the leading l_2 (=20) eigenvalues consistently account for over 99.9999% of the sum of all the eigenvalues (namely, $\theta_2(L^{**})$ is over 99.9999%). From (13), we know that NIGLRAM can get near objective function value to GLRAM, despite that it optimizes (2) approximately.

It is worthwhile to note that since NIGLRAM optimizes (2) approximately, it achieves the suboptimal result in the sense of (2), however, the difference between optimal and suboptimal results is not so distinct, which will be shown in the next subsection.

4.3 Comparison among GLRAM, 2DPCA and NIGLRAM

We try three different values of l_1 and l_2 for GLRAM and NIGLRAM, namely $l_1= l_2=15$, $l_1= l_2=20$, and $l_1= l_2=25$, and results are listed in Table 1. At the same time, results by 2DPCA under the same compression ratio are also reported in the corresponding columns. From Table 1, we can clearly see that, under the same compression ratio, both GLRAM and NIGLRAM yield comparable or higher classification accuracies and lower *RMSREs* and *NMSE*s than 2DPCA, which benefits from a two-sided rather than a single-sided transformation. Generally speaking, GLRAM produces slightly lower reconstruction errors (measured in terms of *RMSRE* or *NMSE*) than NIGLRAM, but such differences can almost be negligible. Further, NIGLRAM achieves almost identical classification performance to GLRAM despite the relatively higher reconstruction errors. This is in accord with such a phenomenon in the PCA related methods, namely the lower reconstruction errors may not definitely means higher classification performance. Comparing the classification performance by the 1NN classifier and the NM classifier, 1NN classifier consistently yields higher classification accuracy than NM classifier in all the experiments.

To give a concrete idea of the reconstruction abilities of the three methods, Fig. 1 shows images for 5 different persons from ORL dataset. The 5 images in the first row are the original images from this dataset, the second row are the ones compressed by our NIGLRAM with $l_1 = l_2 = 20$, and the third and the fourth row are the ones compressed under the same compression ratio by GLRAM and 2DPCA respectively. It is clear that when using the same compression ratio, the images compressed respectively by GLRAM and our NIGLRAM have better visual quality than those compressed by 2DPCA. And at the same time, the images compressed respectively by GLRAM and NIGLRAM almost have the same visual quality.

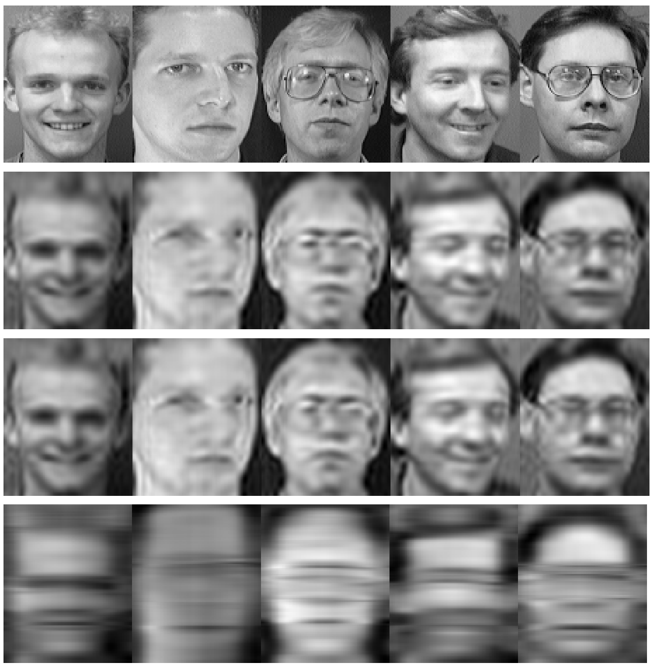


Fig. 1 First row: original images; Second row: images compressed by NIGLRAM; Third row: images compressed by GLRAM; Fourth row: images compressed 2DPCA.

Their CPU times (performed on the computer with CPU: Pentium(R) 4 1.7GHz; RAM: 256Mb) consumed for calculating corresponding projecting transforms are also reported in Table 1.

NIGLRAM consumes slightly more time than 2DPCA, but both NIGLRAM and 2DPCA consume much less time than GLRAM, which contributes to the existence of analytical solutions to the projecting transforms.

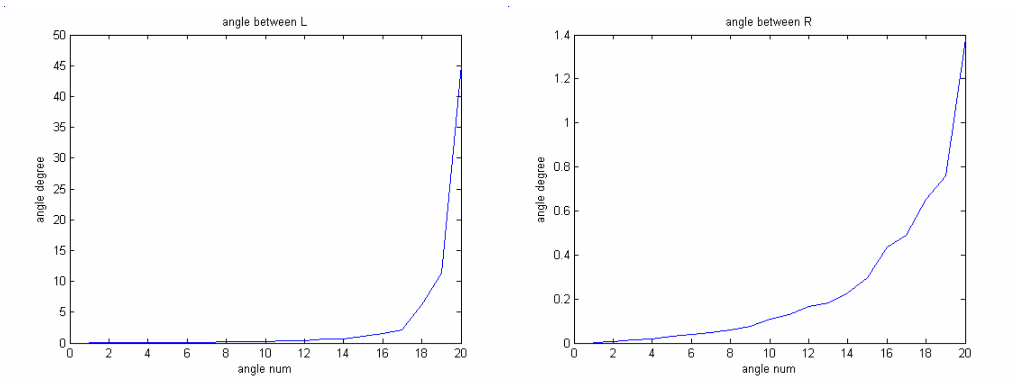
Table 1: Classification accuracy, *NMSE*, *RMSRE* and consumed time on ORL, AR and COIL-20

Dataset	Method	l_1=l_2=15	l_1=l_2=20	l_1=l_2=25
ORL	2DPCA	97.25[a] / 89.50[b] / **2.45**[c] 0.04690[d] / 3633.5[e]	97.75 / 93.50 / **2.45** 0.03190 / 2996.5	98.25 / 94.25 / **2.45** 0.02393 / 2790.6
	GLRAM	98.25 / 94.50 / **13.9** 0.01631 / 2142.8	98.25 / 94.50 /**14.8** 0.01176 / 1819.7	98.25 / 94.50 / **15.1** 0.008944 / 1586.7
	NIGLRAM	98.25 / 94.50 / **3.17** 0.01636 / 2146.1	98.25 / 94.50 / **3.24** 0.01181 / 1823.6	98.25 / 94.50 / **3.35** 0.008975 / 1589.5
AR	2DPCA	71.29 / 68.71 / **2.84** 0.0842 / 7144.5	71.14 / 64.57 / **2.84** 0.05937 / 5070.6	74.43 / 69.14 / **2.84** 0.04471 / 4400.3
	GLRAM	87.57 / 81.86 / **54.2** 0.02185 / 3075.8	89.14 / 83.29 / **57.3** 0.01475 / 2526.8	90.00 / 83.86 / **58.3** 0.01047 / 2129.1
	NIGLRAM	87.43 / 82.00 / **3.94** 0.02192 / 3080.9	89.29 / 83.29 / **3.97** 0.01478 / 2529.4	90.00 / 83.71 / **4.25** 0.01049 / 2131.0
COIL	2DPCA	99.86 / 88.75 / **2.63** 0.1232 / 5065.0	99.72 / 90.14 / **2.63** 0.06428 / 3659.4	99.44 / 90.14 / **2.63** 0.05121 / 3266.3
	GLRAM	99.07 / 91.67 / **27.9** 0.03189 / 2577.3	99.17 / 92.13 / **29.7** 0.02276 / 2177.2	98.98 / 92.13 / **30.5** 0.01683 / 1872.3
	NIGLRAM	99.17 / 91.67 / **3.42** 0.03191 / 2578.1	99.26 / 92.13 / **3.51** 0.02277 / 2177.8	98.98 / 92.13 / **3.65** 0.01683 / 1872.6

[a] Average classification accuracies (%) under given l_1 and l_2 by 1NN classifier
[b] Average classification accuracies (%) under given l_1 and l_2 by NM classifier
[c] Average time (seconds) consumed for computing transformation matrices
[d] Average *NMSE* under given l_1 and l_2
[e] Average *RMSRE* under given l_1 and l_2

4.4 Principal angles between the transformations of GLRAM and NIGLRAM

Principal angles (Golub and Van Loan, 1996) between two subspaces are usually used to measure to what extent the two subspaces intersect with each other, and the number of principal angles that have a degree of zero measures the rank of the intersection of subspaces. In order to explain why NIGLRAM can achieve comparable performance to GLRAM, we report in Fig. 2 the degrees of the principal angles between the transformations of both methods. The experiments are carried out on ORL dataset with l_1=l_2=20. From Fig. 2, we can see that most of degrees of the principal angles between *Ls* (or *Rs*) respectively calculated by GLRAM and NIGLRAM are very small (close to 0), which explains the reason why NIGLRAM achieves comparable performance to GLRAM. Moreover, comparing Fig. 2(a) and (b), we can also see that the degrees of principal angles yielded by the *R* are generally less than those by the *L*, which is due to such a fact that the right transformation *R* calculated by NIGLRAM is the local optimum in the sense of (2), while on the contrary, *L* calculated by NIGLRAM is not.



(a) (b)

Fig. 2 Degrees of principal angles between *Ls* (and *Rs*) calculated respectively by GLRAM and NIGLRAM.

4.5 Using *NMSE* to determine l_1 and l_2 automatically

In NIGLRAM, we can use *NMSE* as a selecting criterion to compute l_1 and l_2 automatically, while GLRAM has no such explicit criterion. Results on Table 2 show how *NMSE* serves as a criterion of computing l_1 and l_2 automatically. From the same table, we can see that, the change in *NMSE* does hardly affect the classification accuracies on ORL and COIL-20 datasets, but affect observably the classification accuracies on AR dataset. A possible explanation for this phenomenon is that ORL and COIL-20 are datasets relatively easier to be classified than AR.

Table 2: l_1 and l_2 determined automatically by *NMSE* criterion in NIGLRAM

NMSE	ORL	AR	COIL
0.005	98.25[a] / 38.00[b] / 37.00[b]	90.14 / 46.71 / 31.74	98.7 / 66.00 / 37.00
0.010	98.25 / 26.20 / 21.00	89.71 / 32.43 / 20.86	99.0 / 46.00 / 24.00
0.015	98.25 / 20.00 / 14.00	88.43 / 25.57 / 15.57	98.8 / 37.00 / 18.00
0.020	98.25 / 17.00 / 10.00	87.43 / 21.57 / 12.29	99.0 / 31.00 / 15.00

[a] Average classification accuracy (%) by 1NN classifier under given *NMSE*
[b] Average values of l_1 and l_2 automatically determined by given *NMSE*

5. Conclusion

In this article, NIGLRAM is proposed as an extension to GLRAM. Compared to GLRAM, the proposed method adopts an analytical rather than an iterative solution. And as a consequence, NIGLRAM can be performed more efficiently. Experimental results on ORL and AR dataset show that NIGLRAM not only is more computationally efficient than GLRAM, but also keeps comparable classification accuracy and reconstruction performance. Furthermore, we show that in NIGLRAM, *NMSE* can be used as a criterion to automatically determine l_1 and l_2, while GLRAM has no such explicit criterion.

As is revealed in (8), maximizing (2) is identical to optimizes *S*. Then future study can be done to directly optimize *S* analytically (note that GLRAM optimizes it iteratively) rather than optimizing *S* approximately as employed in our NIGLRAM.

Acknowledgements

We thank the associate editor and the anonymous referees for their valuable comments, thank Ye Jieping for kindly offering the GLRAM codes, and thank Natural Science Foundations of China under Grant No. 60473035, Jiangsu Natural Science Foundation under Grant No. BK2005122 and Jiangsu Natural Science Key Project under Grant No. BK2004001 for partial supports respectively.

Reference

Golub, G.H., Van Loan, C.F., Matrix Computations, 1996, The Johns Hopkins University Press, Baltimore, MD, USA, third edition.

Kim, M., Kim, D., Lee, S., 2003. Face recognition using the embedded HMM with second-order block-specific observations, Pattern Recognition, 36, 2723–2735.

Liang, Z., Shi, P., 2005. An analytical algorithm for generalized low-rank approximations of matrices, Pattern Recognition, 38(11), 2213-2216.

Liu, J., Chen, S. C., A comment on “An analytical algorithm for generalized low-rank approximations of matrices”, technical report, available at http://parnec.nuaa.edu.cn/paper/comment.pdf

Martinez, A.M., Benavente, R., 1998. The AR-face database, CVC Tech. Rep. #24.

Turk, M.A., Pentland, A.P., 1991. Face Recognition Using Eigenfaces, Proc. of IEEE Conf. on

Computer Vision and Pattern Recognition, pp. 586-591.

Wang, L., Wang, X. et al, 2005. The equivalence of two-dimensional PCA to line-based PCA, Pattern Recognition Letters, 26, 57-60.

Yang, J., Zhang, D. et al, 2004. Two-dimensional PCA: A new approach to appearance-based face representation and recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 26(1), 131-137.

Ye, J., 2004, Generalized low rank approximation of matrices. The Twenty-First International Conference on Machine Learning (ICML 2004), 887-894.