1. INTRODUCTION

Principal component analysis (PCA) and linear discriminant analysis (LDA) are well-known techniques for dimensionality reduction. Since they are based on vectors, matrices such as 2D face images must be transformed into 1D image vectors in advance. However, the resultant vectors usually lead to a high-dimensional vector space, where it is difficult to solve the (generalized) eigenvalue problems for PCA and LDA.

Recently, Yang et al. [1] have proposed 2DPCA, and Ye [2] has proposed generalized low rank approximations of matrices (GLRAM). These methods can handle matrices directly without vectorizing them. Ye [2] proposed an iterative algorithm for GLRAM, which will be summarized in the next section. In GLRAM [2], a matrix \( A_i \) is approximated by the low rank matrix \( M_i = L^T A_i R \), and Ye’s iterative algorithm [2] renews two matrices \( L \) and \( R \) alternately. On the other hand, Liang and Shi [3] and Liang et al. [4] proposed an analytical algorithm which does not need to iterate the renewal procedure. Liang’s analytical algorithm [3, 4] selects the better one from two cases: \( L \) calculated with an initialized \( L \) and \( R \) calculated with an initialized \( R \). However, Hu et al. [5] and Inoue and Urahama [6] showed that Liang’s analytical algorithm [3, 4] does not necessarily give the optimal solution of GLRAM. Liu and Chen [7] also proposed a non-iterative algorithm for GLRAM. However, Liu’s non-iterative algorithm [7] does not select the better one from the two cases in Liang’s analytical algorithm [3, 4] but always outputs the former case. Therefore, Liu’s non-iterative algorithm [7] cannot outperform Liang’s analytical algorithm [3, 4]. Lu et al. [8] proposed another non-iterative algorithm which calculates \( L \) and \( R \) independently. However, the same algorithm as Liu’s one [8] has been shown in the paper [6] already.

In GLRAM [2], it is necessary for users to specify the number of rows \( l_1 \) and that of columns \( l_2 \) in the low rank matrix \( M_i \). Ye [2] experimentally showed that the good results are obtained when \( l_1 = l_2 \). Additionally, Liu et al. [9] derived a lower bound of the objective function for GLRAM and showed that the minimization of the lower bound results in \( l_1 = l_2 \). Ding and Ye [10] have also shown the same lower bound as Liu’s one.

In this paper, we propose a method for determining \( l_1 \) and \( l_2 \) semiautomatically by symmetrizing GLRAM [2]. Although the matrices handled in GLRAM [2] are asymmetric generally, in the proposed method, we construct symmetric matrices from the asymmetric ones to derive symmetric GLRAM. In the proposed method, \( l_1 \) and \( l_2 \) are semiautomatically determined from the sum \( l = l_1 + l_2 \), therefore, the users do not need to specify them. Experimental results show that the proposed method achieves better objective function values than the conventional method when \( l \) is fixed to a constant.

The rest of this paper is organized as follows: Section 2 summarizes GLRAM [2]. Section 3 proposes symmetric GLRAM, Section 4 shows experimental results, and Section 5 concludes this paper.

2. GENERALIZED LOW RANK APPROXIMATIONS OF MATRICES

Let \( A_i \in \mathbb{R}^{r \times c} \), \( i = 1, \ldots, n \) where \( \mathbb{R} \) denotes the set of real numbers. Then the generalized low rank approximations of matrices \( \{ A_i \}_{i=1}^n \) (GLRAM) are formulated as follows [2]:

\[
\begin{align*}
\min_{L, R, \{ M_i \}_{i=1}^n} \quad & \sum_{i=1}^n \| A_i - L M_i R^T \|_F^2 \\
\text{subj. to} \quad & L^T L = I_{l_1}, \quad R^T R = I_{l_2},
\end{align*}
\]

where \( L \in \mathbb{R}^{r \times l_1} \), \( R \in \mathbb{R}^{c \times l_2} \) for \( l_1 < r \), \( l_2 < c \), \( I_{l_1} \), and \( I_{l_2} \) denote the identity matrices of orders \( l_1 \) and \( l_2 \), and \( \| \cdot \|_F \)
Table 1. Ye’s algorithm [2].

<table>
<thead>
<tr>
<th>Algorithm GLRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong></td>
</tr>
<tr>
<td><strong>Output:</strong></td>
</tr>
</tbody>
</table>

1. Obtain initial $L_0$ for $L$ and set $t \leftarrow 1$;
2. While not convergent
   3. form the matrix $M_n = \sum_{i=1}^n A_i^T L_{t-1} L_{t-1}^T A_i$;
   4. compute the $l_2$ eigenvectors $\{\phi_{n,i}\}_{i=1}^{l_2}$ of $M_n$
      corresponding to the largest $l_2$ eigenvalues; 
   5. $R_t \leftarrow [\phi_{n,1}, \ldots, \phi_{n,l_2}]$;
   6. form the matrix $M_L = \sum_{i=1}^n A_i R_t R_t^T A_i$;
   7. compute the $l_1$ eigenvectors $\{\phi_{L,i}\}_{i=1}^{l_1}$ of $M_L$
      corresponding to the largest $l_1$ eigenvalues;
   8. $L_t \leftarrow [\phi_{L,1}, \ldots, \phi_{L,l_1}]$;
   9. $t \leftarrow t + 1$;
10. EndWhile
11. $L \leftarrow L_{t-1}$;
12. $R \leftarrow R_{t-1}$;
13. For $i$ from 1 to $n$
14. $M_i \leftarrow L^T A_i R$;
15. EndFor

denotes the Frobenius norm. If $L$ and $R$ are given, then the optimal $M_i$ is obtained by $M_i = L^T A_i R$. From

$$\sum_{i=1}^n \|A_i - L M_i R^T\|_F^2 = \sum_{i=1}^n \|A_i\|_F^2 - \sum_{i=1}^n \|L^T A_i R\|_F^2,$$

and that $\sum_{i=1}^n \|A_i\|_F^2$ is a constant with respect to $L$ and $R$, the above minimization problem (1) may be rewritten as

$$\max_{L,R} \sum_{i=1}^n \|L^T A_i R\|_F^2.$$  

Ye’s algorithm [2] for this problem is summarized in Table 1, in which $l_1$ and $l_2$ need to be specified by hand. Ye [2] experimentally showed that the good results are obtained when $l_1 = l_2$. Liu et al. [9] also derived the same result as Ye’s one [2] from the minimization of a lower bound of the objective function of GLRAM.

3. SYMMETRIC GLRAM

In the above GLRAM [2], given matrices $\{A_i\}_{i=1}^n$ are symmetric generally. In this section, we construct symmetric matrices from the asymmetric ones $\{A_i\}_{i=1}^n$ as follows:

$$S_i = \begin{pmatrix} O_{c,c} & A_i^T \\ A_i & O_{r,r} \end{pmatrix}, \quad i = 1, \ldots, n,$$

and then propose a low rank approximation method for symmetric matrices $\{S_i\}_{i=1}^n$, where $O_{c,c}$ denotes a $c \times c$ zero matrix. The symmetric GLRAM for $\{S_i\}_{i=1}^n$ becomes

$$\max_U \sum_{i=1}^n \|U^T S_i U\|_F^2$$

subject to

$$U^T U = I_1,$$

where $U \in \mathbb{R}^{(r+c) \times l}$, $I_1$ denotes the identity matrix of order $l_1$ and $l < r + c$. Let $F(U)$ be the objective function in (6). Then we find that

$$F(U) = \sum_{i=1}^n \text{tr} \left( (U^T S_i U)^2 \right) = \sum_{i=1}^n \text{tr} \left( (U^T S_i)^2 \right),$$

from which the Lagrange function for (6) with (7) is given by

$$\mathcal{L}(U, \Lambda) = \sum_{i=1}^n \text{tr} \left( (U^T S_i)^2 \right) - 2 \text{tr} \left[ \Lambda (U^T U - I_1) \right],$$

where $\Lambda \in \mathbb{R}^{(r+c) \times l}$ is a symmetric matrix of which the elements are the Lagrange multipliers and tr denotes the matrix trace. Then we have the necessary conditions for optimality:

$$\frac{\partial \mathcal{L}}{\partial U} = \sum_{i=1}^n S_i U^T S_i U - \Lambda = O_{(r+c)l}, \quad (10)$$

$$\frac{\partial \mathcal{L}}{\partial \Lambda} = U^T U - I_1 = O_{l \times l}, \quad (11)$$

From (10), we have

$$\sum_{i=1}^n S_i U^T S_i U = U \Lambda,$$  

and, from (11), we have $U^T U = I_1$, which is no less than the constraint in (7). Based on (12), we propose an algorithm in Table 2, where input data are matrices $\{A_i\}_{i=1}^n$ and the rank $l$ or the number of columns in $U$. While Ye’s algorithm [2] in Table 1 needs both $l_1$ and $l_2$ for $L$ and $R$ respectively, the proposed algorithm in Table 2 needs only $l$ for $U$.

The details of the algorithm in Table 2 are as follows: First we form symmetric matrices $\{S_i\}_{i=1}^n$ defined by (5) (Line 1). Next we compute the $l$ eigenvectors $\{\tilde{\phi}_{j,i}\}_{j=1}^{l_2}$ corresponding to the largest $l_2$ eigenvalues of $\sum_{i=1}^n S_i^2$ and then initialize $U$ as $U_0 = [\phi_1, \ldots, \phi_t]$, and initialize the number of iterations, $t$, as $t = 1$ (Line 2). Then, for example, $U$ after $t$ iterations is expressed as $U_t$. In the iterative procedure, we first form $M = \sum_{i=1}^n S_i U_{t-1} U_{t-1}^T S_i$ and then compute the $l$ eigenvectors $\{\phi_{j,i}\}_{j=1}^{l_1}$ corresponding to the largest $l_1$ eigenvalues of $M$ to form $U_t = [\phi_{1}, \ldots, \phi_t]$. We repeat this procedure until the convergence condition described below is satisfied (Lines 3-8). We used the convergence condition as

$$\text{RMSE}^{(t)} = \sqrt{\frac{1}{n} \sum_{i=1}^n \|S_i - U_{t-1} U_{t-1}^T S_i U_t U_t^T\|_F^2} < \epsilon$$

for $t = 1, 2, \ldots$, where $\text{RMSE}^{(t)}$ denotes the root mean square error RMSE and $\text{RMSE}^{(t)}$ for $t$ iterations
Table 2. The proposed algorithm

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Symmetric GLRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong></td>
<td>matrices ( { S_i }_{i=1}^n ) and ( l )</td>
</tr>
<tr>
<td><strong>Output:</strong></td>
<td>matrices ( L, R ) and ( { M_i }_{i=1}^n )</td>
</tr>
<tr>
<td>1. Form symmetric matrices ( { S_i }_{i=1}^n ).</td>
<td></td>
</tr>
<tr>
<td>2. Obtain initial ( U_0 ) for ( U ) and set ( t \leftarrow 1 );</td>
<td></td>
</tr>
<tr>
<td>3. While not convergent</td>
<td></td>
</tr>
<tr>
<td>4. Form the matrix ( M = \sum_{i=1}^n S_i U_{t-1} U_{t-1}^T S_i );</td>
<td></td>
</tr>
<tr>
<td>5. Compute the ( l ) eigenvectors ( { \phi_j }_{j=1}^l ) of ( M ) corresponding to the largest ( l ) eigenvalues;</td>
<td></td>
</tr>
<tr>
<td>6. ( U_t \leftarrow [\phi_1, \ldots, \phi_l] );</td>
<td></td>
</tr>
<tr>
<td>7. ( t \leftarrow t + 1 );</td>
<td></td>
</tr>
<tr>
<td>8. EndWhile</td>
<td></td>
</tr>
<tr>
<td>9. ( U \leftarrow U_{t-1} );</td>
<td></td>
</tr>
<tr>
<td>10. ( L = [] );</td>
<td></td>
</tr>
<tr>
<td>11. ( R = [] );</td>
<td></td>
</tr>
<tr>
<td>12. For ( j ) from 1 to ( l )</td>
<td></td>
</tr>
<tr>
<td>13. ( u \leftarrow U(1 \cdot c, j) );</td>
<td></td>
</tr>
<tr>
<td>14. ( v \leftarrow U(c+1 \cdot r, c, j) );</td>
<td></td>
</tr>
<tr>
<td>15. If ( | u | \geq | v | )</td>
<td></td>
</tr>
<tr>
<td>16. ( R \leftarrow [R, u] );</td>
<td></td>
</tr>
<tr>
<td>17. Else</td>
<td></td>
</tr>
<tr>
<td>18. ( L \leftarrow [L, v] );</td>
<td></td>
</tr>
<tr>
<td>19. EndIf</td>
<td></td>
</tr>
<tr>
<td>20. EndFor</td>
<td></td>
</tr>
<tr>
<td>21. For ( i ) from 1 to ( n )</td>
<td></td>
</tr>
<tr>
<td>22. ( M_i \leftarrow L^T A_i R );</td>
<td></td>
</tr>
<tr>
<td>23. EndFor</td>
<td></td>
</tr>
</tbody>
</table>

4. EXPERIMENTAL RESULTS

In this section, we show experimental results on the ORL face image database [11]. Fig. 1 shows face images in the ORL database [11]. The ORL database [11] contains face images of 40 persons. For each person, there are 10 different face images. In our experiments, we used the first 5 images per person, i.e., \( n = 5 \times 40 = 200 \). The height and width of an image are \( r = 112 \) and \( c = 92 \) pixels, respectively.

In Ye’s GLRAM [2], it is shown that the good results are obtained when

\[
l_1 = l_2 \quad (13)
\]

is satisfied [2, 9]. Thus, we call the GLRAM with the constraint (13) the constrained GLRAM (CGLRAM), and compare it with the proposed method.

Let \( L_Y \in \mathbb{R}^{r \times k} \) and \( R_Y \in \mathbb{R}^{c \times h} \) be the matrices \( L \) and \( R \) obtained by CGLRAM, where \( h = \frac{r}{4} \), and let \( L_O \in \mathbb{R}^{r \times l_1} \) and \( R_O \in \mathbb{R}^{c \times l_2} \) be that by the proposed method.

Then we evaluate the value of \( D = \sum_{i=1}^n \left( \left\| L_{i}^T A_i R_O \right\|_F^2 - \left\| L_{i}^T A_i R_Y \right\|_F^2 \right) \), that is, the difference between the two objective function values. If \( D > 0 \), then the objective function value obtained by the proposed method is larger than that by CGLRAM. The value of \( D \) is shown in Fig. 2, where the vertical axis denotes \( D \) and the horizontal axis denotes \( l = 2h = l_1 + l_2 \). In this figure, \( D \) is positive in almost all range of \( l \), and therefore the objective function value by the proposed method is larger than or equal to that by CGLRAM. Since the proposed method accepts different values for \( l_1 \) and \( l_2 \), the objective function value may be different from that by CGLRAM. The values of \( l_1 \) and \( l_2 \) is shown in Fig. 3, where the proposed method and CGLRAM are denoted by the solid and the broken lines, respectively.

Additionally, in CGLRAM, the value of \( l = l_1 + l_2 = 2h \) is restricted to even numbers, and therefore we cannot select odd numbers for \( l \). On the other hand, in the proposed method, we can select both even and odd numbers for \( l \). The
objective function value for the proposed method is shown in Fig. 4, where the solid and the broken lines correspond to the parity of $l$, i.e., odd and even numbers, respectively. The overlap between the solid and the broken lines in this figure shows that the proposed method achieves comparable performance when $l$ is an odd number, with that when $l$ is an even number. Finally, the reconstructed images $A_i = LM_iR_i^T$ are shown in Fig. 5, where the leftmost images are the original ones and the corresponding reconstructed images for $l = 5, 10, 15, \ldots, 45$ are arranged to their right.

Thus, in the proposed method, only $l$ is needed to compute the low rank approximations of matrices instead of $l_1$ and $l_2$ for GLRAM [2]. Furthermore, while $l = l_1 + l_2$ in CGLRAM is restricted to even numbers, the proposed method accepts both even and odd numbers for $l$.

### 5. CONCLUSION

In this paper, we proposed a method for determining semiautomatically the numbers of rows and columns in low rank matrices in the generalized low rank approximations of matrices (GLRAM) by symmetrizing GLRAM, and experimentally showed that the proposed method achieves larger objective function value than the conventional GLRAM (CGLRAM) which uses the same numbers of rows and columns. Additionally, while the total number of rows and columns in CGLRAM is restricted to even numbers, the proposed method accepts both even and odd numbers of rows and columns of low rank matrices.

### 6. REFERENCES


