Received: 24 June 2019 | Revised: 15 March 2020 | Accepted: 31 August 2020

DOI: 10.1002/nla.2338



RESEARCH ARTICLE

WILEY

# A randomized generalized low rank approximations of matrices algorithm for high dimensionality reduction and image compression

**Ke Li** | **Gang Wu**

School of Mathematics, China University of Mining and Technology, Xuzhou, P.R. China

**Correspondence**
Gang Wu, School of Mathematics, China University of Mining and Technology, Xuzhou 221116, Jiangsu, P.R. China.
Email: gangwu@cumt.edu.cn, and gangwu76@126.com

**Summary**

High-dimensionality reduction techniques are very important tools in machine learning and data mining. The method of generalized low rank approximations of matrices (GLRAM) is a popular technique for dimensionality reduction and image compression. However, it suffers from heavily computational overhead in practice, especially for data with high dimension. In order to reduce the cost of this algorithm, we propose a randomized GLRAM algorithm based on randomized singular value decomposition (RSVD). The theoretical contribution of our work is threefold. First, we discuss the decaying property of singular values of the matrices during iterations of the GLRAM algorithm, and provide a target rank required in the RSVD process from a theoretical point of view. Second, we establish the relationship between the reconstruction errors generated by the standard GLRAM algorithm and the randomized GLRAM algorithm. It is shown that the reconstruction errors generated by the former and the latter are comparable, even if the solutions are computed inaccurately during iterations. Third, the convergence of the randomized GLRAM algorithm is investigated. Numerical experiments on some real-world data sets illustrate the superiority of our proposed algorithm over its original counterpart and some state-of-the-art GLRAM-type algorithms.

**KEYWORDS**

generalized low rank approximations of matrices, high dimensionality reduction, image compression, randomized singular value decomposition, singular value decomposition

## 1 | INTRODUCTION

In big data era, a large number of objects are represented in the form of high-dimensional data, such as images, microarray, videos, and so on. With the fast development of data science and technology, there has been an exponential increase in the availability and usage of large and high-dimensional data sets. Unfortunately, operating directly on large and high-dimensional data is inefficient and even infeasible for many classical methods. At present, the problem of dimensionality reduction has received broad attentions in areas such as machine learning, computer vision, and pattern recognition.[1-3] In essence, the aim of dimensionality reduction is to extract important information, and meanwhile to remove redundant information and noise as much as possible.

Principle component analysis (PCA) is a widely used technique for dimensionality reduction,[4] and it has been a baseline algorithm for face recognition. However, PCA-based techniques usually operate on vectors (1D arrays). That is, before applying PCA, the 2D image (matrices) should be mapped into pattern vectors by concatenating their columns or rows. This generally leads to a high-dimensional space. In order to deal with this problem, a two-dimensional PCA method (2DPCA) was proposed,[3] which is a more efficient technique for dealing with 2D images rather than on vectors. However, the disadvantage of 2DPCA is that it can only perform one-sided linear transform, resulting in a relatively low compression ratio. Thus, a $(2D)^2$PCA method based on the 2DPCA method was developed in Reference 5. The key idea is the quadratic feature extraction on the basis of 2DPCA, which makes the dimension of eigenspace extracted smaller and obtains a higher compression rate. Nevertheless, the disadvantage of the $(2D)^2$PCA method is that it is susceptible to changes in lighting and shooting angles.[6]

As a comparison, generalized low rank approximations of matrices (GLRAM) treats data as the native two-dimensional matrix patterns.[7] This method aims to achieve double-sided dimensionality reduction by minimizing the reconstruction error to obtain the optimal projection matrices. Benefited from the two-dimensional representation of data, GLRAM was reported to consume less computation time and yield higher compression ratio than singular value decomposition (SVD) in applications such as image compression and retrieval, and achieve competitive classification performance to SVD.[7,8] Compared with some two-dimensional methods such as 2DPCA, GLRAM employs two-sided transformations rather than single-sided one, and can yield higher compression ratio.

On the other hand, however, a disadvantage of the GLRAM algorithm is that its computational overhead is often large, especially for high-dimensional data. It is known that GLRAM does not admit a closed-form solution,[7] and an iterative procedure is employed to compute the two-sided transformations. In this procedure, one has to form two large-scale matrices explicitly, and to compute two singular value decompositions (SVD) or eigenvalue decompositions (EVD) of the two matrices in each outer iteration.[7] To overcome this difficulty, some extensions and variations of GLRAM were put forward to reduce the cost. For instance, two noniterative algorithms, the noniterative GLRAM (NIGLRAM) algorithm and the simplified GLRAM (SGLRAM) algorithm were presented in References 9 and 10, respectively. Two Krylov subspace methods, the bilinear Lanczos components (BLC) algorithm and the inexact thick-restarted bilinear Lanczos components algorithm (Inex-TRBLC) were presented in References 11 and 12, whose key is to use Lanczos vectors or Ritz vectors as approximations to some dominant eigenvectors, respectively. A robust version of the GLRAM algorithm was brought up in References 13, which effectively weakens the effects of noise and outliers. A symmetric GLRAM algorithm was proposed in References 14. Some theoretical results on the GLRAM algorithm were given in References 8,15.

In essence, the GLRAM algorithm can be understood as low-rank approximations to a collection of matrices.[7] Recently, the randomized singular value decomposition (RSVD) was proposed and it has been a competitive method for calculating low-rank approximations of matrices,[16-20] and randomized projection methods were applied to dimension reduction.[21-23] In order to reduce the computational overhead in GLRAM-type algorithms, we use the randomized singular value decomposition to take the place of the singular value decomposition in the GLRAM, SGLRAM, and NIGLRAM algorithms, and propose three corresponding randomized GLRAM-type algorithms for dimensionality reduction and image compression.

Theoretical results are established to show the validity and rationality of our strategy. More precisely, we first discuss the decaying property of singular values of the matrices during iterations of the GLRAM algorithm, and provide the target rank needed in RSVD from a theoretical point of view. We then consider the relationship between the reconstruction errors generated by the original GLRAM algorithm and our randomized algorithm. It is shown that the reconstruction errors generated by the original algorithm and our randomized algorithm are comparable, even if the solutions are computed inaccurately during iterations. Finally, we provide a sufficient condition for the convergence of the randomized GLRAM algorithm.

The structure of this article is organized as follows. In Section 2, we briefly review the original GLRAM algorithm and some of its properties. In order to reduce the computational complexity of the GLRAM algorithm, in Section 3 we propose a randomized GLRAM algorithm (RGLRAM), and show the rationale and the convergence of the new algorithm. The applications of RSVD to two variations of GLRAM, that is, SGLRAM and NIGLRAM, are also discussed briefly. Numerical experiments are performed in Section 4 to illustrate the superiority of the proposed algorithms over their original counterparts and some state-of-the-art algorithms, and to demonstrate the effectiveness of our theoretical results. Concluding remarks are drawn in Section 5. Some notations used in this article are listed in Table 1.

**TABLE 1** Some notations used in this article

| Notations | Descriptions |
|---|---|
| $A_i$ | The $i$th data point in matrix form |
| $r$ | Number of rows in $A_i$ |
| $c$ | Number of columns in $A_i$ |
| $N$ | Number of training data |
| $A_i^T$ | Transpose of $A_i$ |
| $L$ | Projection matrix on the left side |
| $R$ | Projection matrix on the right side |
| $L_t$ | Approximation of $L$ obtained from the $t$th iteration |
| $R_t$ | Approximation of $R$ obtained from the $t$th iteration |
| $M_i$ | $M_i = L^T A_i R$ is the compressed version of $A_i$ |
| $l_1$ | Number of rows in $M_i$ |
| $l_2$ | Number of columns in $M_i$ |
| $l$ | Common value for $l_1$ and $l_2$, that is, $l_1 = l_2 = l$ |
| $\sigma_i(A)$ | The $i$th largest singular value of matrix $A$ |
| $\|\cdot\|_F$, $\|\cdot\|_2$ | Frobenius norm and 2-norm of a matrix or vector |
| $\mathrm{tr}(A)$ | Trace of the matrix $A$ |
| NMSE | Normalized mean square error |

## 2 | GENERALIZED LOW RANK APPROXIMATIONS OF MATRICES (GLRAM)

In this section, we briefly introduce the method of generalized low rank approximations of matrices (GLRAM)[7] and review some of its properties. Let $\{A_i\}_{i=1}^N \in \mathbb{R}^{r\times c}$ be $N$ data matrices, the GLRAM algorithm aims to find low-rank approximations on a collection of matrices. More precisely, it seeks two orthonormal matrices $L \in \mathbb{R}^{r\times l_1}$ and $R \in \mathbb{R}^{c\times l_2}$ with $l_1 \ll r$ and $l_2 \ll c$, such that[7,8]

$$\min_{\substack{L\in\mathbb{R}^{r\times l_1}, L^TL=I_{l_1} \\ R\in\mathbb{R}^{c\times l_2}, R^TR=I_{l_2}}} \sum_{i=1}^{N} \|A_i - LM_iR^T\|_F^2, \tag{1}$$

where $M_i = L^T A_i R \in \mathbb{R}^{l_1\times l_2}$, and $\tilde{A}_i = LM_iR^T$ is called a reconstructed matrix (or compression) of $A_i, i = 1, 2, \ldots, N$. In order to make the above reconstruction error as small as possible, Ye[7] suggested that the values of $l_1$ and $l_2$ should be equal (denoted by $l$, a common value for $l_1$ and $l_2$ hereafter). Although there is no strict theoretical proof for this, Liu et al.[8] provided some theoretical support for such a choice.

Furthermore, it was shown that the above minimization problem is mathematically equivalent to the following optimization problem[7]

$$\max_{\substack{L\in\mathbb{R}^{r\times l}, L^TL=I_l \\ R\in\mathbb{R}^{c\times l}, R^TR=I_l}} \sum_{i=1}^{N} \|L^T A_i R\|_F^2. \tag{2}$$

Unfortunately, both (1) and (2) have no closed-form solutions in general,[7] and one has to solve them by using some iterative algorithms, as the following theorem indicates.

**Theorem 1** (7). *Let L and R be the optimal solutions to the problem in (1), then* (i) *Given matrix R, L constitutes the l eigenvectors of the matrix*

$$M_L = \sum_{i=1}^{N} A_i R R^T A_i^T \tag{3}$$

*corresponding to the largest l eigenvalues;* (ii) *Given matrix L, R constitutes the l eigenvectors of the matrix*

$$M_R = \sum_{i=1}^{N} A_i^T L L^T A_i \tag{4}$$

*corresponding to the largest l eigenvalues.*

That is, we achieve $L$ and $R$ by alternately solving $l$ dominant eigenvectors of $M_L$ and $M_R$, respectively. This yields an iterative algorithm for solving the projection matrices, and the resulting algorithm is described in Algorithm 1. Here the "normalized mean square error (NMSE)"[7,8] is used as error measurement for the outer iteration, that is,

$$\mathrm{NMSE}_G^{(t)} = 1 - \frac{\sum_{i=1}^{N} \|L_t^T A_i R_t\|_F^2}{\sum_{i=1}^{N} \|A_i\|_F^2} \quad \text{and} \quad \mathrm{Rerr} = \left| \frac{\mathrm{NMSE}_G^{(t)} - \mathrm{NMSE}_G^{(t-1)}}{\mathrm{NMSE}_G^{(t-1)}} \right|, \tag{5}$$

and $\mathrm{NMSE}_G^{(t)}$ and $\mathrm{NMSE}_G^{(t-1)}$ are the NMSE values in the $t$th and the $(t-1)$th iteration of the GLRAM algorithm, respectively, and $L_t$, $R_t$ are projection matrices obtained from the $t$th iteration of the GLRAM algorithm.

**Algorithm 1.** The GLRAM algorithm (GLRAM)[7]

**Input:** *Given the data matrices* $\{A_i\}_{i=1}^N \in \mathbb{R}^{r\times c}$, *the initial guess* $R_0$ *for* $R$, *the number of desired eigenvectors* $l$, *the convergence threshold tol, the maximal iteration number maxit, and set* $t = 1$, $\mathrm{NMSE}_G^{(0)} = Rerr = 1$;
**Output:** *The matrices* $L \in \mathbb{R}^{r\times l}$, $R \in \mathbb{R}^{c\times l}$.
**1.** *while* $\mathrm{NMSE}_G^{(t-1)} > tol$ *and* $\mathrm{Rerr} > tol$ *and* $t \le maxit$ *do:*
**2.** *Form the matrix* $M_L^{(t)} = \sum_{i=1}^N A_i R_{t-1} R_{t-1}^T A_i^T$;
**3.** *Compute the l eigenvectors* $\{\phi_i^L\}_{i=1}^l$ *of* $M_L^{(t)}$ *corresponding to the largest l eigenvalues;*
**4.** *Let* $L_t = [\phi_1^L, \phi_2^L, \ldots, \phi_l^L]$;
**5.** *Form the matrix* $M_R^{(t)} = \sum_{i=1}^N A_i^T L_t L_t^T A_i$;
**6.** *Compute the l eigenvectors* $\{\phi_i^R\}_{i=1}^l$ *of* $M_R^{(t)}$ *corresponding to the largest l eigenvalues;*
**7.** *Let* $R_t = [\phi_1^R, \phi_2^R, \ldots, \phi_l^R]$;
**8.** *Compute the* $\mathrm{NMSE}_G^{(t)}$ *and Rerr according* to (2.5);
**9.** $t = t + 1$;
**10.** *end while*
**11.** *Let* $L = L_{t-1}$ and $R = R_{t-1}$.

However, as the size of the $\{A_i\}$'s is very large in practice, the computational overhead of the GLRAM algorithm comes primarily from forming the two matrices $M_L^{(t)}$ and $M_R^{(t)}$ explicitly, and from solving the projection matrices $L_t$ and $R_t$ in each outer iteration. Ye* used the singular value decomposition (SVD) to compute $L_t$ and $R_t$ in GLRAM, which requires $\mathcal{O}(r^3)$ and $\mathcal{O}(c^3)$ flops in each outer iteration. For storage, GLRAM needs to store an $r\times r$ matrix $M_L^{(t)}$, a $c\times c$ matrix $M_R^{(t)}$, an $r\times l$ matrix $L_t$, and a $c\times l$ matrix $R_t$ in main memory. Therefore, the GLRAM algorithm will be quite expensive for large-scale problems.[11,12]

In order to reduce the computational cost of this algorithm, some variations were proposed. For instance, two noniterative GLRAM-type algorithms, the noniterative GLRAM algorithm (NIGLRAM) and the simplified GLRAM algorithm (SGLRAM) were proposed in References 9 and 10, respectively. In the noniterative GLRAM algorithm (NIGLRAM), the

*http://www-users.cs.umn.edu/\LY1\textbackslash~~jieping/GLRAM/

orthonormal matrices $L$ and $R$ are composed of the eigenvectors of

$$N_L = \sum_{i=1}^{N} A_i A_i^T \quad \text{and} \quad N_R = \sum_{i=1}^{N} A_i^T L L^T A_i \tag{6}$$

corresponding to the first $l$ largest eigenvalues in magnitude, respectively.

Let $W_i = A_i - \overline{A}$, $i = 1, 2, \ldots, N$, where $\overline{A} = \frac{1}{N}\sum_{i=1}^{N} A_i$ is the mean of the $N$ samples, and let

$$S_L = \sum_{i=1}^{N} W_i W_i^T \quad \text{and} \quad S_R = \sum_{i=1}^{N} W_i^T W_i, \tag{7}$$

then in the simplified GLRAM algorithm (SGLRAM),[10] the left and the right projection matrices $L$ and $R$ are composed of the $l$ eigenvectors corresponding to the dominant eigenvalues of $S_L$ and $S_R$, respectively.

Similarly, in the two variations of the GLRAM algorithm, we have to form two large-scale matrices and to solve two large-scale eigenvalue problems (or singular value problems), and the computational cost is prohibitive for large-scale problems. Recently, two Krylov subspace algorithms were presented in References 11,12, where the Lanczos algorithm[11] and the thick-restarted Lanczos algorithm[12,24] are explored for the computation of $L$ and $R$, respectively. However, the two Krylov subspace algorithms may still suffer from heavily computational overhead for large-scale problems when the number of the desired eigenpairs (or the reducing dimension) $l$ is large.[11,12] Thus, it is necessary to seek new technologies to speed up the GLRAM-type algorithms for high-dimensionality reduction.

## 3 | A RANDOMIZED GLRAM ALGORITHM

In this section, we use the randomized singular value decomposition (RSVD)[16-20] to take the place of the singular value decompositions utilized in GLRAM, and propose a randomized GLRAM for image compression and recognition. Theoretical results are given to show the feasibility and rationality of the proposed method.

### 3.1 | The proposed algorithm

A classical problem in large matrix computations is to compute a low-rank approximation to a given matrix.[7,18,19,25] The truncated singular value decomposition (TSVD) can be used to calculate the best low-rank approximation for a desired rank in terms of unitarily invariant norm;[25] however, it may suffer from large overhead in practice.[25] Recently, the randomized singular value decomposition (RSVD) has received great attention, and it provides a powerful tool for solving low-rank approximations to very large matrices.[16-20] To be specific, the idea of the randomized algorithm can be simply divided into two phases: (1) constructing a low-dimensional subspace capturing most of the information of a given matrix; (2) projecting the original matrix into the subspace and then computing standard decomposition of reduced matrix. In Algorithm 2, we present a randomized singular value decomposition algorithm with power iteration; for more details, refer to References 16,17.

**Algorithm 2.** A randomized singular value decomposition accelerated with power iteration (RSVD)[16]

**Input:** *The data matrix* $A \in \mathbb{R}^{m\times n}$, *a target rank* $k$, *the over-sampling parameter* $p$, *and a positive integer* $q$, let $s = k + p$;
**Output:** *The matrices* $Q$, $U_B$, $V_B$ *and* $S_B$.
**1.** *Generate a random matrix* $\Omega \in \mathbb{R}^{n\times s}$;
**2.** *Compute the matrix* $Y = (AA^T)^q A\Omega$, *by multiplying alternately with* $A$ *and* $A^T$;
**3.** *Compute an orthogonal basis* $Q$ *for* $Y$;
**4.** *Compute the matrix* $B = Q^T A$;
**5.** *Compute* $B_k = U_B S_B V_B^T$, *the rank-k truncated SVD of* $B$;
**6.** *Return* $QU_B$, $V_B$ *and* $S_B$, *with* $A \approx (QU_B)S_B V_B^T$.

Algorithm 2 requires $\mathcal{O}(mnk)$ flops[17] in contrast to $\mathcal{O}(mn^2)$ flops for SVD in finding the $k$ dominant components of the singular value decomposition of an $m \times n$ matrix. The following theorem gives a deviation bound for the low-rank approximation $QQ^TA$ obtained from Algorithm 2 in terms of 2-norm, which is an analog of Reference 17, theorem 10.7. It indicates that if the singular values of matrix $A$ decay sufficiently rapidly, the randomized singular value decomposition can provide a good approximation to the input matrix. For more bounds on the approximation obtained from the randomized power method, we refer to References 16,26,27.

**Theorem 2** (28). *Let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(m,n)}$ be the singular values of $A$, then under the above notations, for all $u, v \geq 1$ and $p \geq 4$, we have that*

$$\|A - QQ^TA\|_2 \leq \left[\left(1 + v\sqrt{\frac{3k}{p+1}} + uv\frac{e\sqrt{k+p}}{p+1}\right)\sigma_{k+1}^{2q+1} + v\frac{e\sqrt{k+p}}{p+1}(\Sigma_{j>k}\sigma_j^{2(2q+1)})^{\frac{1}{2}}\right]^{\frac{1}{2q+1}},$$

*with failure probability at most $2v^{-p} + e^{-u^2/2}$.*

*Remark* 1. The motivation of our proposed randomized GLRAM algorithm is twofold: First, the singular values of image matrices such as face images often decay quickly in practice;[29] see Figure 7 of Section 4. Second, the accuracy required in image compression and pattern recognition problems is often very low.[7,12,30] These motivate us to apply RSVD to the GLRAM algorithm.

We are ready to present the randomized GLRAM algorithm. Recall that one has to solve two large-scale eigenvalue problems in each iteration of the GLRAM algorithm. More precisely, in Step 3 of Algorithm 1, it needs to compute the $l$ dominant eigenvectors of the $r$-by-$r$ large matrix $M_L^{(t)} = \sum_{i=1}^N A_i R_{t-1} R_{t-1}^T A_i^T$, which is mathematically equivalent to solving the left singular vectors corresponding to the $l$ largest singular values of

$$G_L^{(t)} = [A_1 R_{t-1}, A_2 R_{t-1}, \ \ldots\ , A_N R_{t-1}] \in \mathbb{R}^{r \times Nl}. \tag{8}$$

Similarly, in Step 6, it needs to compute the $l$ dominant eigenvectors of the $c$-by-$c$ large matrix $M_R^{(t)} = \sum_{i=1}^N A_i^T L_t L_t^T A_i$, which is mathematically equivalent to solving the left singular vectors corresponding to the $l$ largest singular values of

$$G_R^{(t)} = [A_1^T L_t, A_2^T L_t, \ \ldots\ , A_N^T L_t] \in \mathbb{R}^{c \times Nl}. \tag{9}$$

The idea is to solve singular vectors of $G_L^{(t)}$ and $G_R^{(t)}$ by using RSVD instead of using the classical SVD algorithm, and the computational costs and storage requirements will be reduced substantially. One refers to Algorithm 3 for the proposed algorithm. Similar to Algorithm 1, we make use of NMSE values to check the convergence of this algorithm:

$$\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t)} = 1 - \frac{\sum_{i=1}^N \|\tilde{L}_t^T A_i \tilde{R}_t\|_F^2}{\sum_{i=1}^N \|A_i\|_F^2} \quad \text{and} \quad \widetilde{Rerr} = \left|\frac{\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t)} - \widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t-1)}}{\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t-1)}}\right|, \tag{10}$$

where $\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t)}$ and $\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t-1)}$ are the NMSE values in the $t$th and the $(t-1)$th iteration of the RGLRAM algorithm, respectively, and $\tilde{L}_t$, $\tilde{R}_t$ are projection matrices obtained from the $t$th iteration of the RGLRAM algorithm.

Using the same trick, one can propose the randomized simplified GLRAM algorithm (RSGLRAM) and randomized noniterative GLRAM algorithm (RNIGLRAM), based on the simplified GLRAM (SGLRAM)[9] and the noniterative GLRAM algorithm (NIGLRAM).[10] The key difference is that we make use of RSVD instead of the singular value decomposition (SVD) in the original counterparts. A comparison of the operation costs and memory requirements of the three GLRAM-type algorithms and their randomized version is given in Table 2, where we list the overhead required for per iteration in the two iterative algorithms GLRAM and RGLRAM. Based on the fact that $l, N \ll r, c$, it is seen that the three randomized algorithms require much less computational overhead than their original counterparts. One refers to Section 4 for a numerical comparison of these algorithms.

**TABLE 2** A comparison of the computational overhead of six algorithms

| Algorithms | Operation costs[a] | Memory requirements |
|---|---|---|
| GLRAM | $\mathcal{O}(r^3)+\mathcal{O}(c^3)+\mathcal{O}((r^2+c^2)Nl)+\mathcal{O}(rcNl)$ | $\max\{\mathcal{O}(r^2),\mathcal{O}(c^2)\}$ |
| RGLRAM | $\mathcal{O}(rcNl)$ | $\max\{\mathcal{O}(rNl),\mathcal{O}(cNl)\}$ |
| SGLRAM | $\mathcal{O}(r^2cN)+\mathcal{O}(c^2rN)+\mathcal{O}(r^3)+\mathcal{O}(c^3)$ | $\max\{\mathcal{O}(r^2),\mathcal{O}(c^2)\}$ |
| RSGLRAM | $\mathcal{O}(rcNl)$ | $\mathcal{O}(rc)$ |
| NIGLRAM | $\mathcal{O}(r^2cN)+\mathcal{O}(r^3)+\mathcal{O}(c^3)+\mathcal{O}(c^2Nl)+\mathcal{O}(rcNl)$ | $\max\{\mathcal{O}(r^2),\mathcal{O}(c^2)\}$ |
| RNIGLRAM | $\mathcal{O}(rcNl)$ | $\mathcal{O}(cNl)+\mathcal{O}(rl)$ |

[a]Operation costs refer to one iteration in the GLRAM and RGLRAM algorithms

**Algorithm 3.** A randomized GLRAM algorithm (RGLRAM)

**Input:** *Given the data matrices* $\{A_i\}_{i=1}^N \in \mathbb{R}^{r\times c}$, *the initial guess* $\tilde{R}_0$ *for* $\tilde{R}$, *the number of desired eigenvectors* $l$, *and the convergence threshold tol. Choose the maximal iteration number maxit, set* $t=1$ *and let* $\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(0)} = \widetilde{Rerr} = 1$;
**Output:** *The matrices* $\tilde{L}\in\mathbb{R}^{r\times l}$, $\tilde{R}\in\mathbb{R}^{c\times l}$.
**1.** *while* $\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t-1)} > tol$ *and* $\widetilde{Rerr} > tol$ *and* $t < maxit$ *do:*
**2.** *Form the matrix* $\tilde{G}_L^{(t)} = [A_1\tilde{R}_{t-1}, A_2\tilde{R}_{t-1}, \dots, A_N\tilde{R}_{t-1}]$;
**3.** *Compute the* $l$ *left singular vectors* $\{\phi_i^L\}_{i=1}^l$ *of* $\tilde{G}_L^{(t)}$ *corresponding to the largest* $l$ *singular values by using Algorithm 2, and let* $\tilde{L}_t = [\phi_1^L, \phi_2^L, \dots, \phi_l^L]$;
**4.** *Form the matrix* $\tilde{G}_R^{(t)} = [A_1^T\tilde{L}_t, A_2^T\tilde{L}_t, \dots, A_N^T\tilde{L}_t]$;
**5.** *Compute the* $l$ *left singular vectors* $\{\phi_i^R\}_{i=1}^l$ *of* $\tilde{G}_R^{(t)}$ *corresponding to the largest* $l$ *singular values by using Algorithm 2, and let* $\tilde{R}_t = [\phi_1^R, \phi_2^R, \dots, \phi_l^R]$;
**6.** *Compute* $\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t)}$ *and* $\widetilde{Rerr}$ *according to (10);*
**7.** $t = t+1$;
**8.** *end while*
**9.** *Let* $\tilde{L} = \tilde{L}_{t-1}$ *and* $\tilde{R} = \tilde{R}_{t-1}$.

## 3.2 | Theoretical analysis

The key difference between RGRLAM and GLRAM is that one exploits RSVD to take the place of SVD. Theorem 2 indicates that RSVD can provide good approximations if the matrix in question has decaying singular values. As was mentioned in Remark 1, the data matrices $\{A_i\}_{i=1}^N$ often have decaying singular values in practice. So *the key problem* is, "whether the matrices $G_L^{(t)}$ and $G_R^{(t)}$ have decaying singular values during iterations of the GLRAM algorithm, if $\{A_i\}_{i=1}^N$ do so?" The following theorem shows this is indeed the case.

Let the singular values of the matrices $\{A_i^TL_t\}_{i=1}^N$ be $\tilde{\sigma}_1^{(t)} \le \tilde{\sigma}_2^{(t)} \le \dots \le \tilde{\sigma}_{(N-1)l}^{(t)} \le \dots \le \tilde{\sigma}_{Nl}^{(t)}$, and let those of $\{A_iR_{t-1}\}_{i=1}^N$ be $\tilde{\tau}_1^{(t)} \le \tilde{\tau}_2^{(t)} \le \dots \le \tilde{\tau}_{(N-1)l}^{(t)} \le \dots \le \tilde{\tau}_{Nl}^{(t)}$. So there exist two positive numbers $f^{(t)}(1\le f^{(t)}\le l)$ and $d^{(t)}(1\le d^{(t)}\le N)$, such that $\tilde{\sigma}_{(N-1)l}^{(t)}$ is the $f^{(t)}$th largest singular value of the matrix $A_{d^{(t)}}^TL_t$. Similarly, there exist $g^{(t)}(1\le g^{(t)}\le l)$ and $h^{(t)}(1\le h^{(t)}\le N)$, such that $\tilde{\tau}_{(N-1)l}^{(t)}$ is the $g^{(t)}$th largest singular value of the matrix $A_{h^{(t)}}R_{t-1}$. The following theorem establishes upper bounds for the $(l+1)$th singular values of $G_R^{(t)}$ and $G_L^{(t)}$ during iterations of the GLRAM algorithm.

**Theorem 3.** *Under the above notations, let the* $f^{(t)}$*th largest singular value of* $A_{d^{(t)}}$ *be* $\breve{\sigma}_f^{(t)}$, *and let the* $g^{(t)}$*th largest singular value of* $A_{h^{(t)}}$ *be* $\breve{\tau}_g^{(t)}$. *Then we have*

$$\sigma_{l+1}(G_R^{(t)}) \le \sqrt{N}\cdot\breve{\sigma}_f^{(t)} \quad \text{and} \quad \sigma_{l+1}(G_L^{(t)}) \le \sqrt{N}\cdot\breve{\tau}_g^{(t)}.$$

*Proof.* It follows from the Courant–Fischer minimax theorem that[25]

$$\sigma_{l+1}(G_R^{(t)}) = \min_{\substack{\dim(\Pi)\\=(N-1)l}} \max_{\substack{x\in\Pi\\ \|x\|_2=1}} \|G_R^{(t)}\mathbf{x}\|_2. \tag{11}$$

Now we build a subspace $\Pi^{(t)}$ whose dimension is $Nl-(l+1)+1=(N-1)l$. Arrange all the $Nl$ singular values of $\{A_i^T L_t\}_{i=1}^N$ in ascending order, and consider the set $\Upsilon^{(t)} = \{\tilde{\sigma}_1^{(t)}, \tilde{\sigma}_2^{(t)}, \dots, \tilde{\sigma}_{(N-1)l}^{(t)}\}$ which is composed of the smallest $(N-1)l$ ones. Assume that there are $k_i^{(t)}$ singular values $\sigma_{i,1}^{(t)} \le \sigma_{i,2}^{(t)} \le \dots \le \sigma_{i,k_i^{(t)}}^{(t)}$ of $A_i^T L_t$ appeared in this set, and let $Y_i^{(t)} = [\mathbf{y}_{i,1}^{(t)}, \mathbf{y}_{i,2}^{(t)}, \dots, \mathbf{y}_{i,k_i^{(t)}}^{(t)}]$ be the matrix composed of the corresponding right singular vectors. Note that $\Sigma_{i=1}^N k_i^{(t)} = (N-1)l$, we define the subspace as follows

$$\Pi^{(t)} = \operatorname{span}\{\tilde{Y}_1^{(t)}, \tilde{Y}_2^{(t)}, \dots, \tilde{Y}_N^{(t)}\} = \operatorname{span}\left\{\begin{bmatrix} Y_1^{(t)} \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{0} \\ Y_2^{(t)} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}, \dots, \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ Y_N^{(t)} \end{bmatrix}\right\},$$

with $\tilde{Y}_i^{(t)} \in \mathbb{R}^{Nl\times k_i^{(t)}}$ and $\dim(\Pi^{(t)}) = (N-1)l$. Notice that

$$\|G_R^{(t)}\tilde{Y}_i^{(t)}\|_2 = \|[A_1^T L_t, A_2^T L_t, \dots, A_N^T L_t]\tilde{Y}_i^{(t)}\|_2 = \|A_i^T L_t Y_i^{(t)}\|_2 = \sigma_{i,k_i^{(t)}}^{(t)}, \quad i = 1, 2, \dots, N.$$

For any unit vector $\mathbf{x} \in \Pi^{(t)}$, we have

$$\mathbf{x} = \tilde{Y}_1^{(t)}\mathbf{z}_1^{(t)} + \tilde{Y}_2^{(t)}\mathbf{z}_2^{(t)} + \dots + \tilde{Y}_N^{(t)}\mathbf{z}_N^{(t)},$$

with $\|\mathbf{z}_1^{(t)}\|_2^2 + \|\mathbf{z}_2^{(t)}\|_2^2 + \dots + \|\mathbf{z}_N^{(t)}\|_2^2 = 1$. Thus,

$$\begin{aligned}
\|G_R^{(t)}\mathbf{x}\|_2 &= \|G_R^{(t)}\tilde{Y}_1^{(t)}\mathbf{z}_1^{(t)} + G_R^{(t)}\tilde{Y}_2^{(t)}\mathbf{z}_2^{(t)} + \dots + G_R^{(t)}\tilde{Y}_N^{(t)}\mathbf{z}_N^{(t)}\|_2 \\
&\le \|G_R^{(t)}\tilde{Y}_1^{(t)}\|_2 \cdot \|\mathbf{z}_1^{(t)}\|_2 + \|G_R^{(t)}\tilde{Y}_2^{(t)}\|_2 \cdot \|\mathbf{z}_2^{(t)}\|_2 + \dots + \|G_R^{(t)}\tilde{Y}_N^{(t)}\|_2 \cdot \|\mathbf{z}_N^{(t)}\|_2 \\
&= \sigma_{1,k_1^{(t)}}^{(t)} \cdot \|\mathbf{z}_1^{(t)}\|_2 + \sigma_{2,k_2^{(t)}}^{(t)} \cdot \|\mathbf{z}_2^{(t)}\|_2 + \dots + \sigma_{N,k_N^{(t)}}^{(t)} \cdot \|\mathbf{z}_N^{(t)}\|_2 \\
&\le \tilde{\sigma}_{(N-1)l}^{(t)} \cdot (\|\mathbf{z}_1^{(t)}\|_2 + \|\mathbf{z}_2^{(t)}\|_2 + \dots + \|\mathbf{z}_N^{(t)}\|_2) \le \sqrt{N} \cdot \tilde{\sigma}_{(N-1)l}^{(t)},
\end{aligned} \tag{12}$$

where we used $\tilde{\sigma}_{(N-1)l}^{(t)} \ge \sigma_{i,k_i^{(t)}}^{(t)}$, $i = 1, \dots, N$. From (11) and (12), we obtain

$$\sigma_{l+1}(G_R^{(t)}) \le \max_{\substack{x\in\Pi\\ \|x\|_2=1}} \|G_R^{(t)}\mathbf{x}\|_2 \le \sqrt{N} \cdot \tilde{\sigma}_{(N-1)l}^{(t)}.$$

Suppose that $\tilde{\sigma}_{(N-1)l}^{(t)}$ is the $f^{(t)}$th largest singular value of $A_{d^{(t)}}^T L_t$, with $1 \le f^{(t)} \le l$ and $1 \le d^{(t)} \le N$, then we have from the interlacing property of the singular values that[25] $\tilde{\sigma}_{(N-1)l}^{(t)} \le \breve{\sigma}_f^{(t)}$, where $\breve{\sigma}_f^{(t)}$ denotes the $f^{(t)}$th largest singular value of $A_{d^{(t)}}$. Therefore,

$$\sigma_{l+1}(G_R^{(t)}) \le \sqrt{N}\breve{\sigma}_f^{(t)}.$$

Using the same trick, we can prove that

$$\sigma_{l+1}(G_L^{(t)}) \le \sqrt{N}\breve{\tau}_g^{(t)},$$

and the proof is completed. ∎

*Remark* 2. Theorem 3 establishes a relationship between the $(l+1)$th singular values of $G_R^{(t)}$ and $G_L^{(t)}$ and those of $\{A_i\}_{i=1}^N$. More precisely, it indicates that the $(l+1)$th largest singular values of $G_R^{(t)}$ and $G_L^{(t)}$ can be small provided the singular values of the data matrices $\{A_i\}_{i=1}^N$ decay quickly; however, this may not be true for all the singular values.

Thus, the matrices $G_L^{(t)}$ and $G_R^{(t)}$ will have decaying singular values during iterations of the algorithm. As a result, we can choose $k=l$ as the target rank in Algorithm 2, where $l$ is dimension of the projection subspaces. Indeed, we can prove that the matrices $\tilde{G}_R^{(t)}$ and $\tilde{G}_L^{(t)}$ appeared in RGLRAM have similar properties. Moreover, this theorem also shows the rationality of knitting randomized SVD together with the GLRAM framework. One refers to Example 3 for tightness of Theorem 3.

Next, we give insight into the rationality of the randomized GLRAM algorithm from another perspective. Denote by $L$ and $R$ the solutions of the minimization problem (1), and by

$$G_L = [A_1R, A_2R, \dots, A_NR] \quad \text{and} \quad G_R = [A_1^TL, A_2^TL, \dots, A_N^TL].$$

Define the NMSE value with respect to $L$ and $R$ as follows

$$\text{NMSE}_G = 1 - \frac{\sum_{i=1}^N \|L^TA_iR\|_F^2}{\sum_{i=1}^N \|A_i\|_F^2}. \tag{13}$$

Recall that $\widetilde{\text{NMSE}}_{\text{G}}^{(t)}$ appeared in (10) is the NMSE value obtained from the RGLRAM algorithm in the $t$th iteration, in which $\tilde{L}_t$ and $\tilde{R}_t$ are approximate left singular vector matrices of

$$\tilde{G}_L^{(t)} = [A_1\tilde{R}_{t-1}, A_2\tilde{R}_{t-1}, \dots, A_N\tilde{R}_{t-1}] \tag{14}$$

and

$$\tilde{G}_R^{(t)} = [A_1^T\tilde{L}_t, A_2^T\tilde{L}_t, \dots, A_N^T\tilde{L}_t], \tag{15}$$

respectively.

Consider the four subspaces

$$\mathcal{L} = span\{L\}, \quad \mathcal{R} = span\{R\}, \quad \tilde{\mathcal{L}}^{(t)} = span\{\tilde{L}_t\}, \quad \text{and} \quad \tilde{\mathcal{R}}^{(t)} = span\{\tilde{R}_t\},$$

and let

$$P_L = LL^T, \quad P_R = RR^T, \quad \tilde{P}_L^{(t)} = \tilde{L}_t\tilde{L}_t^T, \quad \text{and} \quad \tilde{P}_R^{(t)} = \tilde{R}_t\tilde{R}_t^T$$

be the orthogonal projectors on the four subspaces, respectively. Then the distances between the subspaces $\mathcal{L}$ and $\tilde{\mathcal{L}}^{(t)}$, and $\mathcal{R}$ and $\tilde{\mathcal{R}}^{(t)}$ are defined as[31]

$$\|\sin\theta(\mathcal{L}, \tilde{\mathcal{L}}^{(t)})\|_2 = \|P_L - \tilde{P}_L^{(t)}\|_2, \quad \text{and} \quad \|\sin\theta(\tilde{\mathcal{R}}^{(t)}, \mathcal{R})\|_2 = \|P_R - \tilde{P}_R^{(t)}\|_2, \tag{16}$$

respectively. The following theorem shows that the difference between $\text{NMSE}_G$ and $\widetilde{\text{NMSE}}_{\text{G}}^{(t)}$ is dominated by the distances between the subspaces $\mathcal{L}, \tilde{\mathcal{L}}^{(t)}$ and $\mathcal{R}, \tilde{\mathcal{R}}^{(t)}$.

**Theorem 4.** *Under the above notations, we have that*

$$|\widetilde{\text{NMSE}}_G^{(t)} - \text{NMSE}_G| \le \|\sin\theta(\tilde{\mathcal{L}}^{(t)}, \mathcal{L})\|_2 + \|\sin\theta(\tilde{\mathcal{R}}^{(t)}, \mathcal{R})\|_2. \tag{17}$$

*Proof.* We notice that

$$\sum_{i=1}^{N}\|L^TA_iR\|_F^2=\sum_{i=1}^{N}\mathrm{tr}(R^TA_i^TLL^TA_iR)=\sum_{i=1}^{N}\mathrm{tr}(A_i^TP_LA_iP_R),\tag{18}$$

and

$$\sum_{i=1}^{N}\|\tilde{L}_t^TA_i\tilde{R}_t\|_F^2=\sum_{i=1}^{N}\mathrm{tr}(A_i^T\tilde{P}_L^{(t)}A_i\tilde{P}_R^{(t)}).$$

Indeed, (18) can be rewritten as

$$\begin{aligned}\sum_{i=1}^{N}\|L^TA_iR\|_F^2&=\sum_{i=1}^{N}\mathrm{tr}(A_i^T(\tilde{P}_L^{(t)}+P_L-\tilde{P}_L^{(t)})A_i(\tilde{P}_R^{(t)}+P_R-\tilde{P}_R^{(t)}))\\&=\sum_{i=1}^{N}\mathrm{tr}(A_i^T\tilde{P}_L^{(t)}A_i\tilde{P}_R^{(t)})+\sum_{i=1}^{N}\mathrm{tr}((A_i^T(P_L-\tilde{P}_L^{(t)})A_i\tilde{P}_R^{(t)})+(A_i^TP_LA_i(P_R-\tilde{P}_R^{(t)})))\\&=\sum_{i=1}^{N}\|\tilde{L}_t^TA_i\tilde{R}_t\|_F^2+\sum_{i=1}^{N}\mathrm{tr}(A_i^T(P_L-\tilde{P}_L^{(t)})A_i\tilde{P}_R^{(t)})+\sum_{i=1}^{N}\mathrm{tr}(A_i^TP_LA_i(P_R-\tilde{P}_R^{(t)})).\end{aligned}\tag{19}$$

So it follows that

$$\begin{aligned}\left|\sum_{i=1}^{N}\|L^TA_iR\|_F^2-\sum_{i=1}^{N}\|\tilde{L}_t^TA_i\tilde{R}_t\|_F^2\right|&=\left|\sum_{i=1}^{N}\mathrm{tr}(A_i^T(P_L-\tilde{P}_L^{(t)})A_i\tilde{P}_R^{(t)})+\sum_{i=1}^{N}\mathrm{tr}(A_i^TP_LA_i(P_R-\tilde{P}_R^{(t)}))\right|\\&\le\sum_{i=1}^{N}|\mathrm{tr}(A_i^T(P_L-\tilde{P}_L^{(t)})A_i\tilde{P}_R^{(t)})|+\sum_{i=1}^{N}|\mathrm{tr}(A_i^TP_LA_i(P_R-\tilde{P}_R^{(t)}))|\\&\le\sum_{i=1}^{N}\|A_i\|_F^2\|P_L-\tilde{P}_L^{(t)}\|_2\|\tilde{P}_R^{(t)}\|_2+\sum_{i=1}^{N}\|A_i\|_F^2\|P_R-\tilde{P}_R^{(t)}\|_2\|P_L\|_2\\&=\sum_{i=1}^{N}\|A_i\|_F^2\|P_L-\tilde{P}_L^{(t)}\|_2+\sum_{i=1}^{N}\|A_i\|_F^2\|P_R-\tilde{P}_R^{(t)}\|_2.\end{aligned}\tag{20}$$

From (10), (13), and (20), we obtain

$$\begin{aligned}|\widetilde{\mathrm{NMSE}}_G^{(t)}-\mathrm{NMSE}_G|&=\frac{\left|\sum_{i=1}^{N}\|L^TA_iR\|_F^2-\sum_{i=1}^{N}\|\tilde{L}_t^TA_i\tilde{R}_t\|_F^2\right|}{\sum_{i=1}^{N}\|A_i\|_F^2}\\&\le\|P_L-\tilde{P}_L^{(t)}\|_2+\|P_R-\tilde{P}_R^{(t)}\|_2\\&=\|\sin\theta(\tilde{\mathcal{L}}^{(t)},\mathcal{L})\|_2+\|\sin\theta(\tilde{\mathcal{R}}^{(t)}\mathcal{R})\|_2,\end{aligned}$$

which completes the proof. ∎

Notice that $\tilde{G}_L^{(t)}$ and $\tilde{G}_R^{(t)}$ generated in the $t$th iteration of RGLRAM can be viewed as some approximations to $G_L$ and $G_R$, respectively. It is interesting to consider the relationship between the difference $|\widetilde{\mathrm{NMSE}}_G^{(t)}-\mathrm{NMSE}_G|$ and the distances $\|\tilde{G}_L^{(t)}-G_L\|_2$ and $\|\tilde{G}_R^{(t)}-G_R\|_2$.

**Corollary 1.** *Denote by* $\hat{G}_L^{(t)}=\tilde{L}_t\tilde{L}_t^T\tilde{G}_L^{(t)}$ *and by* $\hat{G}_R^{(t)}=\tilde{R}_t\tilde{R}_t^T\tilde{G}_R^{(t)}$. *Consider the following four sets composed of singular values in descending order*

$$\Sigma_L=\{\sigma_1(G_L),\sigma_2(G_L),\ \dots\ ,\sigma_l(G_L)\},\quad \hat{\Sigma}_L^{(t)}=\{\sigma_{l+1}(\hat{G}_L^{(t)}),\sigma_{l+2}(\hat{G}_L^{(t)}\},\ \dots\ ,\sigma_{Nl}(\hat{G}_L^{(t)})\},$$

*and*

$$\Sigma_R = \{\sigma_1(G_R), \sigma_2(G_R), \ldots, \sigma_l(G_R)\}, \quad \hat{\Sigma}_R^{(t)} = \{\sigma_{l+1}(\hat{G}_R^{(t)}), \sigma_{l+2}(\hat{G}_R^{(t)}), \ldots, \sigma_{Nl}(\hat{G}_R^{(t)})\}.$$

*If*

$$\delta_L^{(t)} = \min_{\substack{1\le i\le l,\\ l+1\le j\le Nl}} \{|\sigma_i - \hat{\sigma}_j| : \sigma_i \in \Sigma_L, \hat{\sigma}_j \in \hat{\Sigma}_L^{(t)}\} > 0 \quad \text{and} \quad \delta_R^{(t)} = \min_{\substack{1\le i\le l,\\ l+1\le j\le Nl}} \{|\sigma_i - \hat{\sigma}_j| : \sigma_i \in \Sigma_R, \hat{\sigma}_j \in \hat{\Sigma}_R^{(t)}\} > 0, \tag{21}$$

*then*

$$|\widetilde{\mathrm{NMSE}}_G^{(t)} - \mathrm{NMSE}_G| \le \frac{\|\tilde{G}_L^{(t)} - G_L\|_2 + \mathcal{O}(\sigma_{l+1}(\tilde{G}_L^{(t)}))}{\delta_L^{(t)}} + \frac{\|\tilde{G}_R^{(t)} - G_R\|_2 + \mathcal{O}(\sigma_{l+1}(\tilde{G}_R^{(t)}))}{\delta_R^{(t)}}, \tag{22}$$

*with failure probability at most* $2v^{-p} + e^{-u^2/2}$ *for all* $u, v \ge 1$ *and* $p \ge 4$.

*Proof.* Recall that the columns of $L$ and $\tilde{L}_t$ are the left singular vectors of $G_L$ and $\hat{G}_L^{(t)} = \tilde{L}_t \tilde{L}_t^T \tilde{G}_L^{(t)}$, respectively. Let $E_L^{(t)} = \hat{G}_L^{(t)} - G_L$, $\Sigma_L = \{\sigma_1(G_L), \sigma_2(G_L), \ldots, \sigma_l(G_L)\}$, and let $\hat{\Sigma}_L^{(t)} = \{\sigma_{l+1}(\hat{G}_L^{(t)}), \sigma_{l+2}(\hat{G}_L^{(t)}\}, \ldots, \sigma_{Nl}(\hat{G}_L^{(t)})\}$. We obtain from the generalized $\sin\Theta$ theorem[32] that, if $\delta_L^{(t)} = \min_{i,j}\{|\sigma_i - \hat{\sigma}_j| : \sigma_i \in \Sigma_L, \hat{\sigma}_j \in \hat{\Sigma}_L^{(t)}\} > 0$, then

$$\|\sin\theta(\tilde{\mathcal{L}}^{(t)}, \mathcal{L})\|_2 \le \frac{\|E_L^{(t)}\|_2}{\delta_L^{(t)}}. \tag{23}$$

Similarly, let $\hat{G}_R^{(t)} = \tilde{R}_t \tilde{R}_t^T \tilde{G}_R^{(t)}$, $E_R^{(t)} = \hat{G}_R^{(t)} - G_R$, $\Sigma_R = \{\sigma_1(G_R), \sigma_2(G_R), \ldots, \sigma_l(G_R)\}$, and $\hat{\Sigma}_R^{(t)} = \{\sigma_{l+1}(\hat{G}_R^{(t)}), \sigma_{l+2}(\hat{G}_R^{(t)}), \ldots, \sigma_{Nl}(\hat{G}_R^{(t)})\}$, if $\delta_R^{(t)} = \min_{i,j}\{|\sigma_i - \hat{\sigma}_j| : \sigma_i \in \Sigma_R, \hat{\sigma}_j \in \hat{\Sigma}_R^{(t)}\} > 0$, then

$$\|\sin\theta(\tilde{\mathcal{R}}^{(t)}, \mathcal{R})\|_2 \le \frac{\|E_R^{(t)}\|_2}{\delta_R^{(t)}}. \tag{24}$$

A combination of (17), (23), and (24) yields

$$|\widetilde{\mathrm{NMSE}}_G^{(t)} - \mathrm{NMSE}_G| \le \frac{\|E_L^{(t)}\|_2}{\delta_L^{(t)}} + \frac{\|E_R^{(t)}\|_2}{\delta_R^{(t)}}. \tag{25}$$

On the other hand, we notice that

$$E_L^{(t)} = (\tilde{G}_L^{(t)} - G_L) - (I - \tilde{L}_t \tilde{L}_t^T)\tilde{G}_L^{(t)}, \quad E_R^{(t)} = (\tilde{G}_R^{(t)} - G_R) - (I - \tilde{R}_t \tilde{R}_t^T)\tilde{G}_R^{(t)},$$

and thus

$$\|E_L^{(t)}\|_2 \le \|\tilde{G}_L^{(t)} - G_L\|_2 + \|(I - \tilde{L}_t \tilde{L}_t^T)\tilde{G}_L^{(t)}\|_2, \quad \|E_R^{(t)}\|_2 \le \|\tilde{G}_R^{(t)} - G_R\|_2 + \|(I - \tilde{R}_t \tilde{R}_t^T)\tilde{G}_R^{(t)}\|_2. \tag{26}$$

It follows from Theorem 2 that

$$\|(I - \tilde{L}_t \tilde{L}_t^T)\tilde{G}_L^{(t)}\|_2 = \mathcal{O}(\sigma_{l+1}(\tilde{G}_L^{(t)})), \quad \|(I - \tilde{R}_t \tilde{R}_t^T)\tilde{G}_R^{(t)}\|_2 = \mathcal{O}(\sigma_{l+1}(\tilde{G}_R^{(t)})), \tag{27}$$

with failure probability at most $2v^{-p} + e^{-u^2/2}$ for all $u, v \ge 1$ and $p \ge 4$. A combination of (25)–(27) gives the proof. ∎

*Remark* 3. As $\tilde{G}_L^{(t)}$ and $\tilde{G}_R^{(t)}$ are approximations to $G_L^{(t)}$ and $G_R^{(t)}$, we have from Theorem 3 that $\sigma_{l+1}(\tilde{G}_L^{(t)})$ and $\sigma_{l+1}(\tilde{G}_R^{(t)})$ can be small provided the singular values of the data matrices $\{A_i\}_{i=1}^N$ decaying quickly. Although (22) may be not sharp in

practice, Corollary 1 indicates that the difference between $\mathrm{NMSE}_G$ and $\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t)}$ is closely related to the quality of the approximations $\tilde{G}_L^{(t)}$, $\tilde{G}_R^{(t)}$ to $G_L$, $G_R$, respectively, and the gaps $\delta_L^{(t)}$ and $\delta_R^{(t)}$.

Finally, we focus on the convergence of the RGLRAM algorithm. In the GLRAM algorithm, the NMSE values are nonincreasing and bounded from below;[7] however, this is not true for the RGLRAM algorithm in general. Note that $\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t)}$ is closely related to the stopping criterion in RGLRAM, and the larger the value is, the worse the compression performance will be, and vice versa;[7,8] refer to (10). Therefore, we only need to consider the behavior of $\sum_{i=1}^N \|\tilde{L}_t^T A_i \tilde{R}_t\|_F^2$ as $t \to \infty$. As it is bounded from above by $\sum_{i=1}^N \|A_i\|_F^2$, it is necessary to consider the monotonically nondecreasing condition. The following theorem gives a sufficient condition for the convergence of the RGLRAM algorithm.

**Theorem 5.** *Let $\hat{L}_t$ and $\hat{R}_t$ be composed of the exact $l$ dominant left singular vectors of $\tilde{G}_L^{(t)}$ and $\tilde{G}_R^{(t)}$, respectively. Denote by the four matrices $\hat{\mathcal{L}}^{(t)} = span\{\hat{L}_t\}, \tilde{\mathcal{L}}^{(t)} = span\{\tilde{L}_t\}, \hat{\mathcal{R}}^{(t)} = span\{\hat{R}_t\}$, and $\tilde{\mathcal{R}}^{(t)} = span\{\tilde{R}_t\}$. If*

$$\|\sin\theta(\hat{\mathcal{L}}^{(t)}, \tilde{\mathcal{L}}^{(t)})\|_2 + \|\sin\theta(\hat{\mathcal{R}}^{(t)}, \tilde{\mathcal{R}}^{(t)})\|_2 \leq \frac{(\|(\tilde{G}_R^{(t)})^T \hat{R}_t\|_F^2 - \|(\tilde{G}_R^{(t)})^T \tilde{R}_{t-1}\|_F^2) + (\|(\tilde{G}_L^{(t)})^T \hat{L}_t\|_F^2 - \|(\tilde{G}_L^{(t)})^T \tilde{L}_{t-1}\|_F^2)}{\sum_{i=1}^N \|A_i\|_F^2}, \tag{28}$$

*then the $\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t)}$ values are nonincreasing and the RGLRAM algorithm converges.*

*Proof.* If we denote by

$$\Delta_1 = \sum_{i=1}^N (\|\tilde{L}_t^T A_i \hat{R}_t\|_F^2 - \|\tilde{L}_t^T A_i \tilde{R}_{t-1}\|_F^2), \quad \Delta_2 = \sum_{i=1}^N (\|\hat{L}_t^T A_i \tilde{R}_{t-1}\|_F^2 - \|\tilde{L}_{t-1}^T A_i \tilde{R}_{t-1}\|_F^2),$$
$$\Delta_3 = \sum_{i=1}^N (\|\tilde{L}_t^T A_i \tilde{R}_t\|_F^2 - \|\tilde{L}_t^T A_i \hat{R}_t\|_F^2), \quad \Delta_4 = \sum_{i=1}^N (\|\tilde{L}_t^T A_i \tilde{R}_{t-1}\|_F^2 - \|\hat{L}_t^T A_i \tilde{R}_{t-1}\|_F^2),$$

then

$$\sum_{i=1}^N \|\tilde{L}_t^T A_i \tilde{R}_t\|_F^2 - \sum_{i=1}^N \|\tilde{L}_{t-1}^T A_i \tilde{R}_{t-1}\|_F^2 = \Delta_1 + \Delta_2 + \Delta_3 + \Delta_4. \tag{29}$$

Therefore, $\sum_{i=1}^N \|\tilde{L}_t^T A_i \tilde{R}_t\|_F^2$ is monotonically nondecreasing if and only if (29) is nonnegative.

Given $\tilde{R}_{t-1}$ and $\tilde{L}_t$, as $\hat{L}_t$ and $\hat{R}_t$ are composed of the $l$ dominant left singular vectors of $\tilde{G}_L^{(t)}$ and $\tilde{G}_R^{(t)}$, respectively, so we have $\Delta_1, \Delta_2 \geq 0$, and $\Delta_3, \Delta_4 \leq 0$. Let $\hat{P}_L^{(t)} = \hat{L}_t \hat{L}_t^T$, $\hat{P}_R^{(t)} = \hat{R}_t \hat{R}_t^T$, $\tilde{P}_L^{(t)} = \tilde{L}_t \tilde{L}_t^T$, and $\tilde{P}_R^{(t)} = \tilde{R}_t \tilde{R}_t^T$, similar to (20), we can prove that

$$|\Delta_3| \leq \sum_{i=1}^N \|A_i\|_F^2 \cdot \|\hat{P}_R^{(t)} - \tilde{P}_R^{(t)}\|_2, \tag{30}$$

and

$$|\Delta_4| \leq \sum_{i=1}^N \|A_i\|_F^2 \cdot \|\hat{P}_L^{(t)} - \tilde{P}_L^{(t)}\|_2. \tag{31}$$

Consider the four subspaces $\hat{\mathcal{L}}^{(t)} = span\{\hat{L}_t\}, \tilde{\mathcal{L}}^{(t)} = span\{\tilde{L}_t\}, \hat{\mathcal{R}}^{(t)} = span\{\hat{R}_t\}$, and $\tilde{\mathcal{R}}^{(t)} = span\{\tilde{R}_t\}$, we obtain from Reference 31 that

$$\|\hat{P}_L^{(t)} - \tilde{P}_L^{(t)}\|_2 = \|\sin\theta(\hat{\mathcal{L}}^{(t)}, \tilde{\mathcal{L}}^{(t)})\|_2, \quad \|\hat{P}_R^{(t)} - \tilde{P}_R^{(t)}\|_2 = \|\sin\theta(\hat{\mathcal{R}}^{(t)}, \tilde{\mathcal{R}}^{(t)})\|_2. \tag{32}$$

A combination of (30), (31), and (32) yields

$$|\Delta_3| + |\Delta_4| \le \sum_{i=1}^{N} \|A_i\|_F^2 \cdot (\|\sin\theta(\hat{\mathcal{L}}^{(t)}, \tilde{\mathcal{L}}^{(t)})\|_2 + \|\sin\theta(\hat{\mathcal{R}}^{(t)}, \tilde{\mathcal{R}}^{(t)})\|_2).$$

Therefore, if

$$\begin{aligned}&\|\sin\theta(\hat{\mathcal{L}}^{(t)}, \tilde{\mathcal{L}}^{(t)})\|_2 + \|\sin\theta(\hat{\mathcal{R}}^{(t)}, \tilde{\mathcal{R}}^{(t)})\|_2\\ &\quad\le \frac{\Delta_1 + \Delta_2}{\sum_{i=1}^{N} \|A_i\|_F^2} = \frac{(\|(\tilde{G}_R^{(t)})^T \hat{R}_t\|_F^2 - \|(\tilde{G}_R^{(t)})^T \tilde{R}_{t-1}\|_F^2) + (\|(\tilde{G}_L^{(t)})^T \hat{L}_t\|_F^2 - \|(\tilde{G}_L^{(t)})^T \tilde{L}_{t-1}\|_F^2)}{\sum_{i=1}^{N} \|A_i\|_F^2},\end{aligned}$$

then $|\Delta_3 + \Delta_4| = |\Delta_3| + |\Delta_4| \le \Delta_1 + \Delta_2$, where we used $\Delta_3, \Delta_4 \le 0$. That is,

$$\sum_{i=1}^{N} \|\tilde{L}_t^T A_i \tilde{R}_t\|_F^2 - \sum_{i=1}^{N} \|\tilde{L}_{t-1}^T A_i \tilde{R}_{t-1}\|_F^2 \ge 0,$$

and $\sum_{i=1}^{N} \|\tilde{L}_t^T A_i \tilde{R}_t\|_F^2$ is nondecreasing; moreover, it is bounded from above by $\sum_{i=1}^{N} \|A_i\|_F^2$. Thus, the RGLRAM algorithm converges as $t \to \infty$. ∎

## 4 | NUMERICAL EXPERIMENTS

In this section, we perform numerical experiments on some real-world data sets to show the numerical behavior of our randomized algorithms. All the numerical experiments are carried on a Hp workstation with 16 cores double Intel(R)Xeon(R) E5-2620 v4 processors, and with CPU 2.10 GHz and RAM 64 GB. The operation system is 64-bit Windows 10. All the numerical results are obtained from running the MATLAB R2016b software.

As was suggested in Reference 8, we set the reduced dimensions $l_1 = l_2 = l$ in all the numerical experiments. The stopping criterion in GLRAM and RGLRAM is chosen as

$$\mathrm{NMSE}^{(t)} < 10^{-3} \quad \text{or} \quad \left|\frac{\mathrm{NMSE}^{(t)} - \mathrm{NMSE}^{(t-1)}}{\mathrm{NMSE}^{(t-1)}}\right| < 10^{-3},$$

where $\mathrm{NMSE}^{(t)}$ and $\mathrm{NMSE}^{(t-1)}$ denote the NMSE values obtained from the $t$th and the $(t-1)$th iteration, respectively. In addition, the maximum number of iteration *maxit* is set to be 100 in all the numerical experiments. In view of Theorem 3, we set the target rank $k = l$, where $l$ is dimension of the projection subspaces, and choose $p = 0, q = 1$ in all the randomized algorithms. So as to compare the numerical performance of the algorithms, we list the NMSE value (NMSE), the CPU time in seconds (CPU time (s)), and the number of iterations (IT) in the tables below. Notice that NIGLRAM, SGLRAM, and their randomized versions RNIGLRAM and RSGLRAM are all noniterative algorithms, so the number of iterations in all the tables are filled with "—."

Five image databases, `Artificial filming images`, `NWPU VHR-10 remote sensing`[33†], `Japanese Female Facial Expression (JAFFE)`[34 ‡], `Images1`, and `Images2`[§] are used in Examples 1–3 for image compression problems; one refers to Figure 1 for some samples of the first three databases. As for the last two data sets Images1 (7 images) and Images2 (10 images), see Figures 2 and 3. In all the experiments, the data matrices are normalized so that $\|A_i\|_F = 1, i = 1, 2, \ldots, N$.

**Example 1.** In this example, we compare the randomized algorithms RGLRAM, RSGLRAM, and RNIGLRAM with their original counterparts GLRAM, NIGLRAM, and SGLRAM on the `NWPU VHR-10 remote sensing images` database and the `Artificial filming images` database, and illustrate the efficiency of our proposed algorithms.

The NWPU VHR-10 remote sensing images database is a publicly available geospatial object detection database used for research purposes.[33] This database contains 10 classes of objects including airplane, ship, storage tank, baseballdiamond, tennis court, basketball court, ground track field, harbor, bridge, and vehicle. It consists of very-high-resolution

[†]http://www.escience.cn/people/JunweiHan/NWPUVHR10dataset.html
[‡]http://www.kasrl.org/jaffe.html
[§]http://sc.chinaz.com/tupian

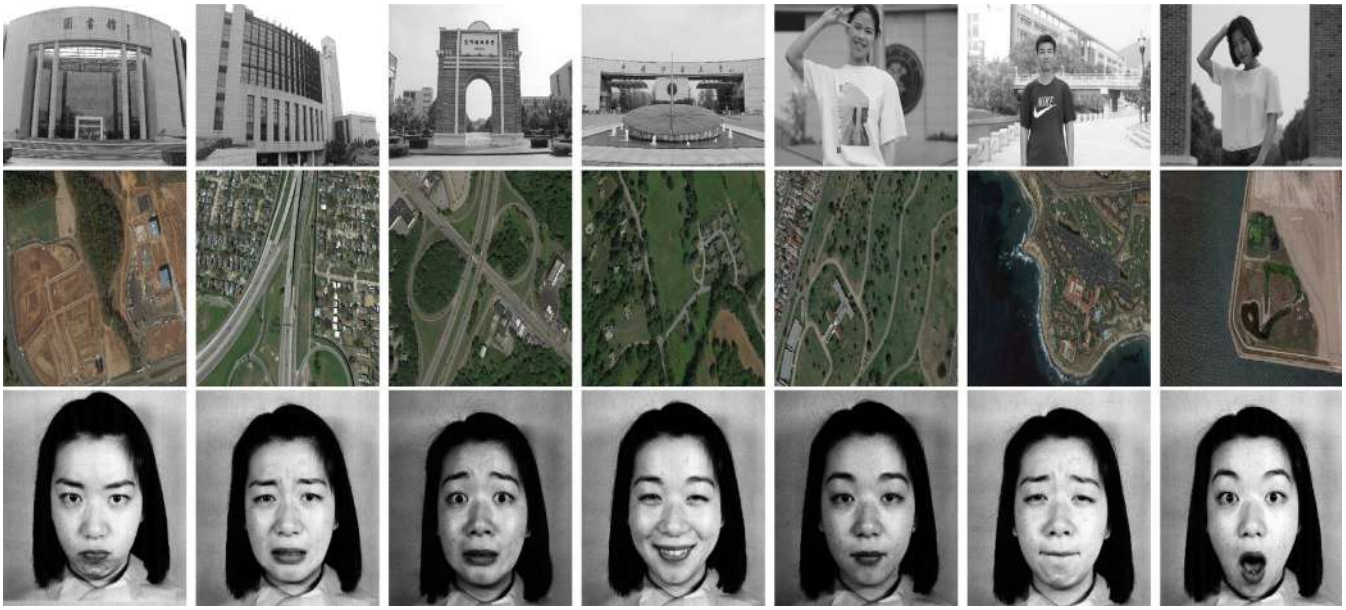


**FIGURE 1** Some samples of the Artificial filming images database (the first line, of size $5,472 \times 3,648$), the NWPU VHR-10 remote sensing database (the second line, of size $5,000 \times 3,000$), and the JAFFE database (the third line, of size $3,000 \times 3,000$)

**FIGURE 2** The images of Images1 (of size $3,609 \times 2,592$, $N = 7$)

(VHR) remote sensing images that were cropped from Google Earth and Vaihingen data set and then manually annotated by experts. We choose $N = 150$ images as sample from the database. As the pixel size of images is not uniform, we normalize them to $5000 \times 3000$ pixels by using the MATLAB command `imresize.m`. The Artificial filming images database contain $N = 120$ high-resolution photos of size $5472 \times 3648$; one refers to Figure 1 for some samples of these two data sets.

Tables 3 and 4 report the NMSE values, CPU time (in seconds), and iteration numbers of the six algorithms with reducing dimension $l = 60$, 90 on the Artificial filming images database with $N = 120$, and $l = 120$, 150 on the NWPU VHR-10 remote sensing images database with $N = 150$. Some remarks are in order. First, it seems that $Nl \ll c$ and $Nl \ll r$ are crucial for the success of the randomized algorithms; see Algorithm 3. However, we see from Tables 3 and 4 that our randomized algorithms can still run much faster than their original counterparts even if $Nl \geq r$ and $c$. This is because the reducing dimension $l$ is small in practice, and we can apply RSVD to the transpose of $\tilde{G}_L$ and $\tilde{G}_R$ if $Nl \geq r$ and $c$. Thus, the cost required by RSVD are much cheaper than those in SVD.

More precisely, as is seen from the numerical all the randomized algorithms run much faster than their original counterparts. For example, for the Artificial filming images database, RGLRAM uses 27.03 and 28.19 seconds as $l = 60$ and 90, while GLRAM uses 307.95 and 332.38 s, respectively. In other words, RGLRAM is about 10 times faster than GLRAM, and the two algorithms share about the same NMSE values. Similar cases occur in the other four algorithms. Among the three randomized algorithms, we see that RSGLRAM is the slowest one. This is because we have to explicitly form two

**FIGURE 3** The images of Images2 (of size $3,744 \times 5,616$, $N = 10$)

**FIGURE 4** Example 1 (artificial filming images): the original image (the first line) and the images reconstructed (from left to right) by the GLRAM, RGLRAM, NIGLRAM, RNIGLRAM, SGLRAM, RSGLRAM algorithms with $l = 60$ (the second line) and $l = 90$ (the third line)

matrices of size $r \times Nc$ and $c \times Nr$, respectively, as well as to compute their randomized singular value decompositions, however, RSGLRAM still runs much better than SGLRAM.

Second, we observe that the NMSE values (i.e., the reconstruction errors) of the randomized algorithms and the standard ones are comparable, which demonstrate the effectiveness of Theorems 3 and 4. To show this more precisely, we plot in Figures 4 and 5 the original images and the reconstructed ones obtained from the six algorithms. It is seen that the reconstruction quality of the randomized algorithms is favorable. Third, we also observe that the number of iterations required for the RGLRAM algorithm is generally a little larger than those of the GLRAM algorithm. This is because we solve the singular value problems "inexactly" in RGLRAM. Although our randomized algorithm may use more iterations than the GLRAM algorithm, the computational complexities required in RSVD are much cheaper than those in SVD. Thus, the RGLRAM can run much faster than GLRAM, with comparable reconstruction errors.

**Example 2.** In this example, we compare the three randomized algorithms RGLRAM, RNIGLRAM, and RSGLRAM with two GLRAM-type algorithms based on Krylov subspaces. One is the bilinear Lanczos components algorithm (BLC)[11] which exploits the classical Lanczos algorithm[35] as the inner iteration, instead of using the expensive singular value decomposition (SVD) or eigenvalue decomposition to compute the dominant eigenpairs. After the dominant eigenpairs are evaluated in the inner iteration, the GLRAM process is ran as the outer iteration to calculate the projection matrices $L$ and $R$. The other is the inexact thick-restarted Lanczos components algorithm (Inex-TRBLC) proposed recently,[12] in which the authors apply the inexact thick-restarting strategy[24] to the BLC algorithm, and make use of the Ritz vectors as some approximations to dominant eigenvectors instead of the Lanczos vectors; for more details on practical implementations, we refer to Reference 12.

The `Japanese Female Facial Expression (JAFFE)` database[34] contains $N = 213$ images of seven facial expressions (six basic facial expressions plus one neutral) posed by 10 Japanese female models. Each image has been rated

| Algorithms | $l$ | Time (s) | IT | NMSE |
|---|---|---|---|---|
| GLRAM | 60 | 307.95 | 4 | 0.0112 |
| **RGLRAM** | 60 | 27.03 | 5 | 0.0118 |
| NIGLRAM | 60 | 137.57 | — | 0.0113 |
| **RNIGLRAM** | 60 | 39.81 | — | 0.0119 |
| SGLRAM | 60 | 207.17 | — | 0.0114 |
| **RSGLRAM** | 60 | 117.40 | — | 0.0121 |
| GLRAM | 90 | 332.38 | 4 | 0.0077 |
| **RGLRAM** | 90 | 28.19 | 4 | 0.0081 |
| NIGLRAM | 90 | 142.80 | — | 0.0077 |
| **RNIGLRAM** | 90 | 53.30 | — | 0.0082 |
| SGLRAM | 90 | 208.66 | — | 0.0078 |
| **RSGLRAM** | 90 | 141.58 | — | 0.0083 |

**TABLE 3** Example 1: A comparison of the six algorithms on the Artificial filming images database (of size $5,472 \times 3,648$, $N = 120$)

| Algorithms | $l$ | CPU time (s) | IT | NMSE |
|---|---|---|---|---|
| GLRAM | 120 | 282.06 | 4 | 0.0233 |
| **RGLRAM** | 120 | 39.97 | 4 | 0.0245 |
| NIGLRAM | 120 | 135.53 | — | 0.0233 |
| **RNIGLRAM** | 120 | 53.85 | — | 0.0248 |
| SGLRAM | 120 | 182.13 | — | 0.0234 |
| **RSGLRAM** | 120 | 147.32 | — | 0.0252 |
| GLRAM | 150 | 290.15 | 4 | 0.0188 |
| **RGLRAM** | 150 | 42.96 | 4 | 0.0199 |
| NIGLRAM | 150 | 137.78 | — | 0.0188 |
| **RNIGLRAM** | 150 | 52.04 | — | 0.0201 |
| SGLRAM | 150 | 183.20 | — | 0.0188 |
| **RSGLRAM** | 150 | 136.99 | — | 0.0204 |

**TABLE 4** Example 1: A comparison of the six algorithms on the NWPU VHR-10 remote sensing database (of size $5,000 \times 3,000$, $N = 150$)

on six emotion adjectives by 60 Japanese subjects. In this example, we make use of the MATLAB function `imresize.m` to resize the images to $3000 \times 3000$ pixels; ones refer to Figure 1 for some samples of this database.

In the BLC and the Inex-TRBLC algorithms, we choose the number of Lanczos steps $m = 2l$ with $l = 60$ and 90, and the number of approximated eigenpairs is chosen as $l + 10$. The convergence threshold *tol* for the inner iteration is chosen as $10^{-2}$ in the Inex-TRBLC algorithm, moreover, we explicitly form $M_L$ and $M_R$ in both the BLC and the Inex-TRBLC algorithms. Table 5 lists the numerical results. As a by-product, we also present the results obtained from running the GLRAM algorithm.

We observe from Table 5 that our three proposed algorithms run much faster than BLC and the recently proposed algorithm Inex-TRBLC. Moreover, the NMSE values of the five algorithm are comparable. Therefore, the new algorithms are much more efficient than the two Krylov subspace algorithms. In Figure 6, we plot the original image and the reconstructed ones by using the six algorithms as $l = 60$ and 90. One sees that the images compression from the proposed algorithms are satisfactory.

**Example 3.** In this example, we try to show the validity of the proposed theoretical results. Figure 7 lists the first 100 largest singular values of $\{A_i\}_{i=1}^{N}$, where `Images1` is of size $3609 \times 2592$ with $N = 7$, and `Images2` is of size $3744 \times 5616$ with $N = 10$. It is observed that the singular values of these data matrices decay quickly to the order of $10^{-2}$.

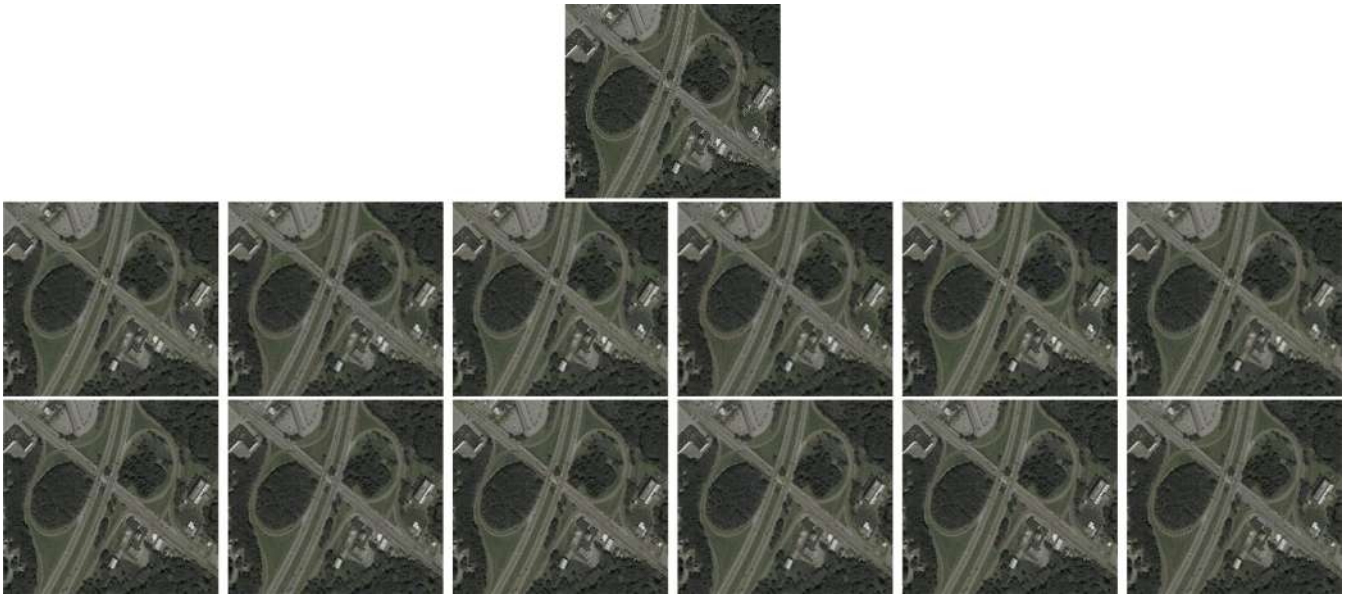


**FIGURE 5** Example 1 (`NWPU VHR-10 remote sensing images`): the original image (the first line) and the images reconstructed (from left to right) by the GLRAM, RGLRAM, NIGLRAM, RNIGLRAM, SGLRAM, RSGLRAM algorithms with $l = 120$ (the second line ) and $l = 150$ (the third line)

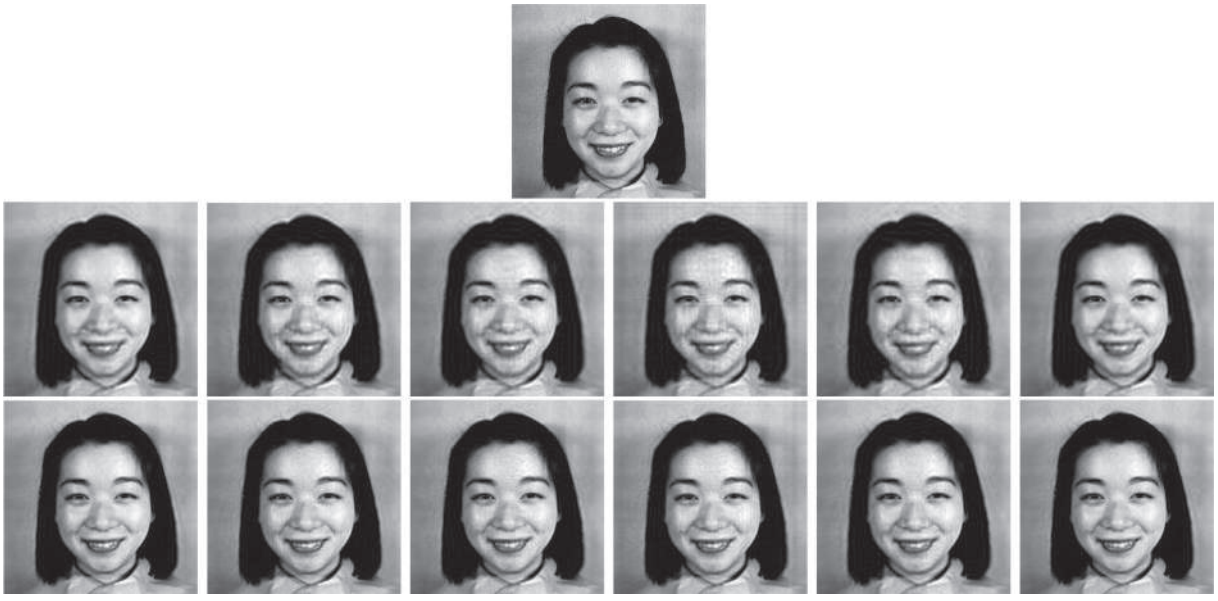


**FIGURE 6** Example 2 (`JAFFE`): The original image (the first line) and images reconstructed (from left to right) by the GLRAM, RGLRAM, RNIGLRAM, RSGLRAM, BLC, and Inex-TRBLC algorithms with $l = 60$ (the second line) and $l = 90$ (the third line)

First, we show the sharpness of Theorem 3. The test data sets are Images1 and a subset of 10 images of the Artificial filming images database, called `Sub-Artificial filming images`. Recall from Theorem 3 that, there exist two numbers $g$ and $f$, such that the $(l+1)$th singular values of $G_L^{(t)}$, $G_R^{(t)}$ are bounded from above by $\sqrt{N}\check{\tau}_g^{(t)}$ and $\sqrt{N}\check{\sigma}_f^{(t)}$, respectively. So as to determine the values of $\check{\tau}_g^{(t)}$ and $\check{\sigma}_f^{(t)}$ in each iteration of GLRAM, we compute the $(l+1)$th singular values of $G_L^{(t)}$ and $G_R^{(t)}$ as well as all the singular values of the matrices $\{A_i^T L_t\}_{i=1}^N$ and $\{A_i R_{t-1}\}_{i=1}^N$, and then sort them in ascending order. Tables 6 and 7 list the values of $\sigma_{l+1}(G_L^{(t)})$, $\sigma_{l+1}(G_R^{(t)})$, $\sqrt{N}\check{\tau}_g^{(t)}$, and $\sqrt{N}\check{\sigma}_f^{(t)}$ in each iteration of GLRAM with $l = 100$

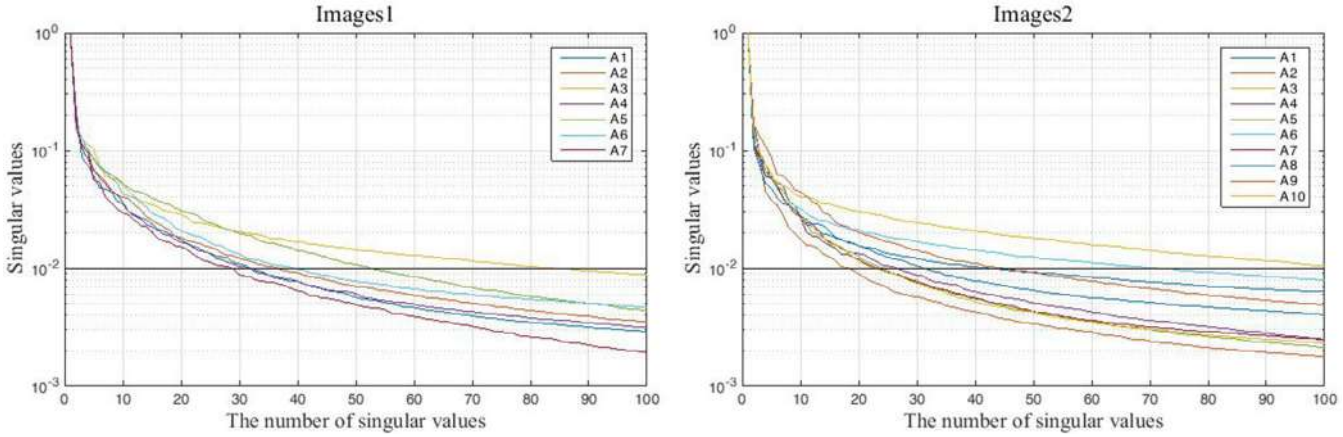



**FIGURE 7** The first 100 largest singular values of $\{A_i\}_{i=1}^N$ of Images1 (left) and Images2 (right)

**TABLE 5** Example 2: A comparison of the six algorithms on the JAFFE database (of size $3,000 \times 3,000$, $N = 213$)

| Algorithms | $l$ | CPU time (s) | IT | NMSE |
|---|---|---|---|---|
| GLRAM | 60 | 177.54 | 4 | 0.0051 |
| **RGLRAM** | 60 | 22.05 | 5 | 0.0054 |
| **RNIGLRAM** | 60 | 34.31 | — | 0.0055 |
| **RSGLRAM** | 60 | 97.37 | — | 0.0058 |
| BLC | 60 | 136.08 | 4 | 0.0063 |
| Inex-TRBLC | 60 | 132.14 | 4 | 0.0051 |
| GLRAM | 90 | 171.19 | 4 | 0.0034 |
| **RGLRAM** | 90 | 35.49 | 5 | 0.0036 |
| **RNIGLRAM** | 90 | 42.75 | — | 0.0036 |
| **RSGLRAM** | 90 | 115.04 | — | 0.0038 |
| BLC | 90 | 148.59 | 4 | 0.0040 |
| Inex-TRBLC | 90 | 143.09 | 4 | 0.0034 |

and 120. Notice that four iterations are needed for Images1, while only three iterations are required for sub-Artificial filming images. We see that the $(l+1)$th singular values of $G_L^{(t)}$ and $G_R^{(t)}$ and their upper bounds are in the same order, and our theoretical results are sharp.

Second, we show the effectiveness of Theorem 4, Corollary 1, and Theorem 5. The test set is Images2 of size $3744 \times 5616$ with $N = 10$. In this experiment, we choose the reduced dimensions $l = 10$ and set the oversampling parameter $p = 85$ in RGLRAM. Table 8 and Figure 8 depict the difference between the NMSE values obtained from GLRAM and RGLRAM algorithms during iterations, as well as its theoretical upper bound. Although the theoretical upper bound given in Theorem 4 may not be sharp in practice, we see that the difference is dominated by the distances between the subspaces $\mathcal{L}, \tilde{\mathcal{L}}^{(t)}$ and $\mathcal{R}, \tilde{\mathcal{R}}^{(t)}$, and they have the same decreasing pattern.

An interesting question is whether the assumptions (21) and (28) made in Corollary 1 and Theorem 5 are practical or not. In Table 8 and Figure 8 we present the gap values $\delta_L^{(t)}$ and $\delta_R^{(t)}$ involved in Corollary 1. It is obvious to see that they are greater than zero and our assumption (21) is practical. Table 9 and Figure 9 report the values of $\| \sin\theta(\hat{\mathcal{L}}^{(t)}, \tilde{\mathcal{L}}^{(t)})\|_2 + \| \sin\theta(\hat{\mathcal{R}}^{(t)}, \tilde{\mathcal{R}}^{(t)})\|_2$ and those of $(\Delta_1 + \Delta_2)/\sum_{i=1}^N \|A_i\|_F^2$ appeared in Theorem 5, as well as the NMSE values obtained from the RGLRAM algorithms during iterations. Note that the computation of $(\Delta_1 + \Delta_2)/\sum_{i=1}^N \|A_i\|_F^2$ involves the initial guess $\tilde{L}_0$ and $\tilde{R}_0$ as $t = 1$, and we generate the two matrices randomly by using the MATLAB command `rand.m`, and then orthogonalize them by using the MATLAB command `orth.m`. We observe from Table 9 and Figure 9 that the assumption (28) is practical, and the NMSE values of RGLRAM algorithm are nonincreasing if this assumption is satisfied.

**TABLE 6** Example 3: The values of the $(l+1)$th singular value of $G_L^{(t)}$ and the upper bound $\sqrt{N}\check{\tau}_g^{(t)}$ in each iteration of GLRAM

| | Images1 $(3,609\times 2,592, N=7)$ | | | | Sub-Artificial filming images $(5,472\times 3,648, N=10)$ | | | |
|---|---|---|---|---|---|---|---|---|
| | $l=100$ | | $l=120$ | | $l=100$ | | $l=120$ | |
| Iteration | $\sigma_{l+1}(G_L^{(t)})$ | $\sqrt{N}\check{\tau}_g^{(t)}$ | $\sigma_{l+1}(G_L^{(t)})$ | $\sqrt{N}\check{\tau}_g^{(t)}$ | $\sigma_{l+1}(G_L^{(t)})$ | $\sqrt{N}\check{\tau}_g^{(t)}$ | $\sigma_{l+1}(G_L^{(t)})$ | $\sqrt{N}\check{\tau}_g^{(t)}$ |
| 1 | 0.0027 | 0.0756 | 0.0025 | 0.0734 | 0.0030 | 0.0927 | 0.0029 | 0.0794 |
| 2 | 0.0139 | 0.0767 | 0.0115 | 0.0634 | 0.0182 | 0.0915 | 0.0164 | 0.0811 |
| 3 | 0.0137 | 0.0767 | 0.0114 | 0.0634 | 0.0182 | 0.0915 | 0.0163 | 0.0811 |
| 4 | 0.0136 | 0.0767 | 0.0114 | 0.0634 | — | — | — | — |

**TABLE 7** Example 3: The values of the $(l+1)$th singular value of $G_R^{(t)}$ and the upper bound $\sqrt{N}\check{\sigma}_f^{(t)}$ in each iteration of GLRAM

| | Images1 $(3,609\times 2,592, N=7)$ | | | | Sub-Artificial filming images $(5,472\times 3,648, N=10)$ | | | |
|---|---|---|---|---|---|---|---|---|
| | $l=100$ | | $l=120$ | | $l=100$ | | $l=120$ | |
| Iteration | $\sigma_{l+1}(G_R^{(t)})$ | $\sqrt{N}\check{\sigma}_f^{(t)}$ | $\sigma_{l+1}(G_R^{(t)})$ | $\sqrt{N}\check{\sigma}_f^{(t)}$ | $\sigma_{l+1}(G_R^{(t)})$ | $\sqrt{N}\check{\sigma}_f^{(t)}$ | $\sigma_{l+1}(G_R^{(t)})$ | $\sqrt{N}\check{\sigma}_f^{(t)}$ |
| 1 | 0.0124 | 0.0756 | 0.0105 | 0.0642 | 0.0165 | 0.0900 | 0.0142 | 0.0814 |
| 2 | 0.0123 | 0.0771 | 0.0103 | 0.0634 | 0.0163 | 0.0915 | 0.0142 | 0.0801 |
| 3 | 0.0122 | 0.0771 | 0.0101 | 0.0634 | 0.0163 | 0.0915 | 0.0141 | 0.0801 |
| 4 | 0.0123 | 0.0771 | 0.0101 | 0.0634 | — | — | — | — |

**TABLE 8** Example 3: The difference $|\widetilde{\text{NMSE}}_G^{(t)} - \text{NMSE}_G|$ between the NMSE values obtained from GLRAM and RGLRAM algorithms as well as the theoretical upper bound $\|\sin\theta(\tilde{\mathcal{L}}^{(t)},\mathcal{L})\|_2 + \|\sin\theta(\tilde{\mathcal{R}}^{(t)},\mathcal{R})\|_2$ established in Theorem 4, and the gap values $\delta_L^{(t)}$, $\delta_R^{(t)}$ involved in Corollary 1; Images2 with $l=10$, $N=10$

| Iteration | 1 | 2 | 3 |
|---|---|---|---|
| $\lvert\widetilde{\text{NMSE}}_G^{(t)} - \text{NMSE}_G\rvert$ | 0.0035 | $1.30\times 10^{-5}$ | $8.31\times 10^{-12}$ |
| $\|\sin\theta(\tilde{\mathcal{L}}^{(t)},\mathcal{L})\|_2 + \|\sin\theta(\tilde{\mathcal{R}}^{(t)},\mathcal{R})\|_2$ | 1.33 | 0.16 | $1.78\times 10^{-5}$ |
| $\delta_L^{(t)}$, $\delta_R^{(t)}$ | 0.094, 0.11 | 0.094, 0.11 | 0.094, 0.11 |

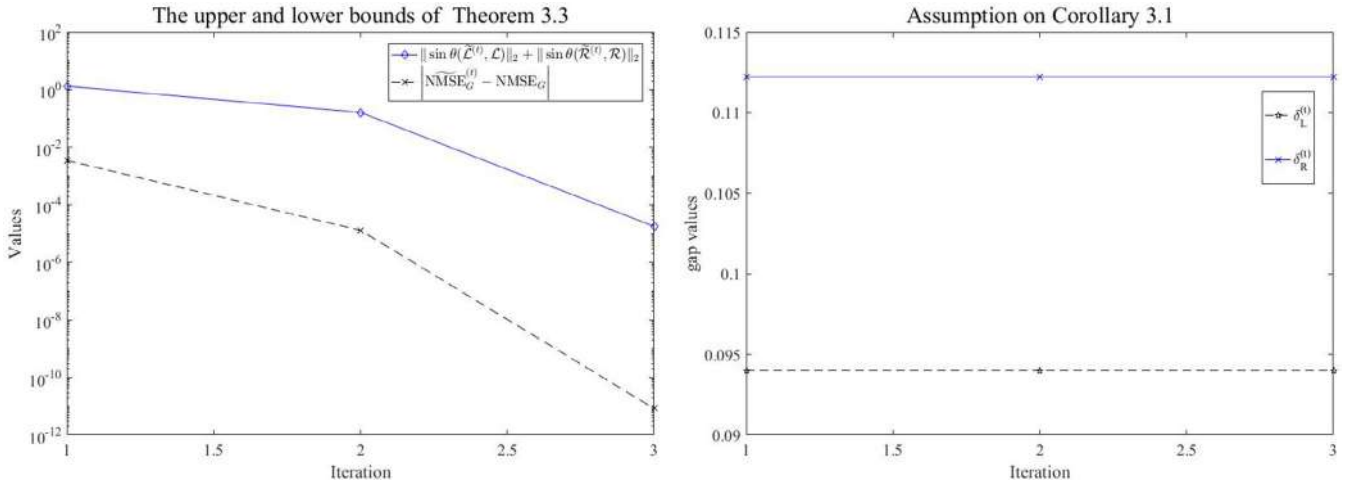



**FIGURE 8** Example 3: The difference $|\widetilde{\text{NMSE}}_G^{(t)} - \text{NMSE}_G|$ between the NMSE values obtained from GLRAM and RGLRAM algorithms as well as the theoretical upper bound $\|\sin\theta(\tilde{\mathcal{L}}^{(t)},\mathcal{L})\|_2 + \|\sin\theta(\tilde{\mathcal{R}}^{(t)},\mathcal{R})\|_2$ established in Theorem 4 (left), and the gap values $\delta_L^{(t)}$, $\delta_R^{(t)}$ involved in Corollary 1 (right); Images2 with $l=10$, $N=10$

| Iteration | 1 | 2 | 3 |
|---|---|---|---|
| $\|\sin\theta(\hat{\mathcal{L}}^{(t)},\tilde{\mathcal{L}}^{(t)})\|_2+\|\sin\theta(\hat{\mathcal{R}}^{(t)},\tilde{\mathcal{R}}^{(t)})\|_2$ | $2.03\times10^{-5}$ | $3.99\times10^{-6}$ | $2.23\times10^{-6}$ |
| $\frac{\Delta_1+\Delta_2}{\sum_{i=1}^{N}\|A_i\|_F^2}$ | 0.12 | 0.0035 | $1.32\times10^{-5}$ |
| $\widetilde{\mathrm{NMSE}}_{\mathrm{G}}^{(t)}$ | 0.0323 | 0.0288 | 0.0288 |

$\Delta_1=\|(\tilde{G}_R^{(t)})^T\hat{R}_t\|_F^2-\|(\tilde{G}_R^{(t)})^T\tilde{R}_{t-1}\|_F^2$, $\Delta_2=\|(\tilde{G}_L^{(t)})^T\hat{L}_t\|_F^2-\|(\tilde{G}_L^{(t)})^T\tilde{L}_{t-1}\|_F^2$.

**TABLE 9** Example 3: The values of $\|\sin\theta(\hat{\mathcal{L}}^{(t)},\tilde{\mathcal{L}}^{(t)})\|_2+\|\sin\theta(\hat{\mathcal{R}}^{(t)},\tilde{\mathcal{R}}^{(t)})\|_2$ and those of $(\Delta_1+\Delta_2)/\sum_{i=1}^{N}\|A_i\|_F^2$ involved in Theorem 5, as well as the NMSE values obtained from the RGLRAM algorithms during iterations; Images2 with $l=10$, $N=10$

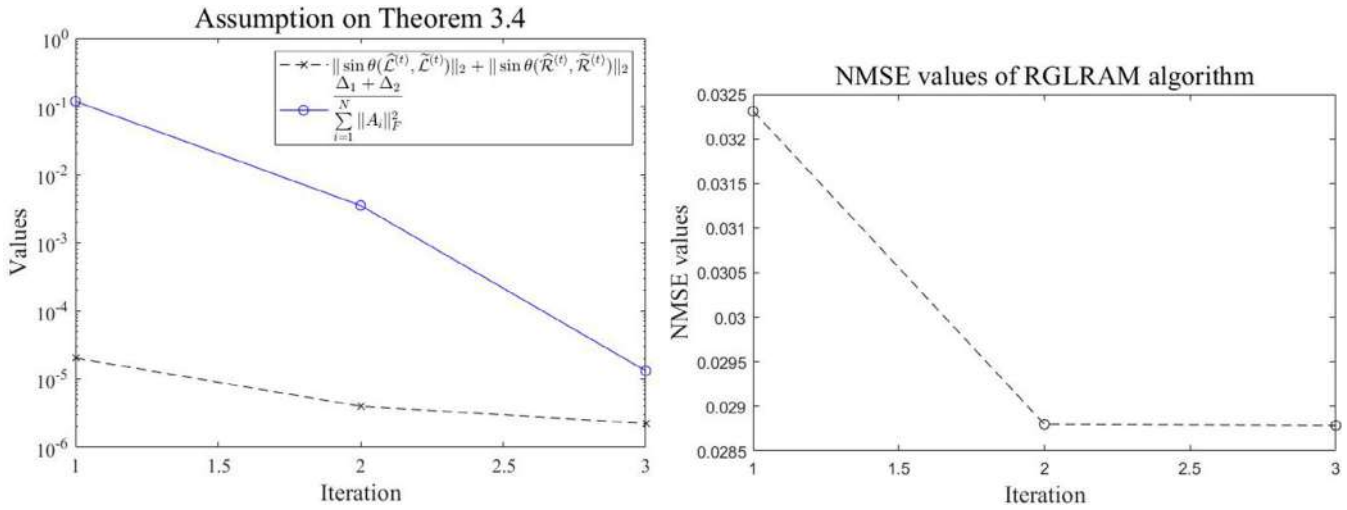



**FIGURE 9** Example 3: The values of $\|\sin\theta(\hat{\mathcal{L}}^{(t)},\tilde{\mathcal{L}}^{(t)})\|_2+\|\sin\theta(\hat{\mathcal{R}}^{(t)},\tilde{\mathcal{R}}^{(t)})\|_2$ and those of $(\Delta_1+\Delta_2)/\sum_{i=1}^{N}\|A_i\|_F^2$ involved in Theorem 5 (left), as well as the NMSE values obtained from the RGLRAM algorithms during iterations (right); Images2 with $l=10$, $N=10$

**Example 4.** In this example, we demonstrate the proposed approaches on other applications including face recognition and text recognition. There are two test sets in this example. The `CMU-PIE`[¶] database[36] contains more than 40,000 images of 68 subjects with more than 500 images for each. These face images are captured by 13 synchronized cameras and 21 flashes under varying pose, illumination, expression and lights. In this experiment, we choose 40 subjects and 60 images under different illuminations, lights, expressions and poses for each subject. Thus, the total number of images chosen from `CMU-PIE` database is 2400. The original size of these face images is $486\times640$ pixels; see Figure 10 for some samples of this data set. In this experiment, we randomly choose 30 images in each subject for training (i.e., $N=1200$), and the rest are used for testing.

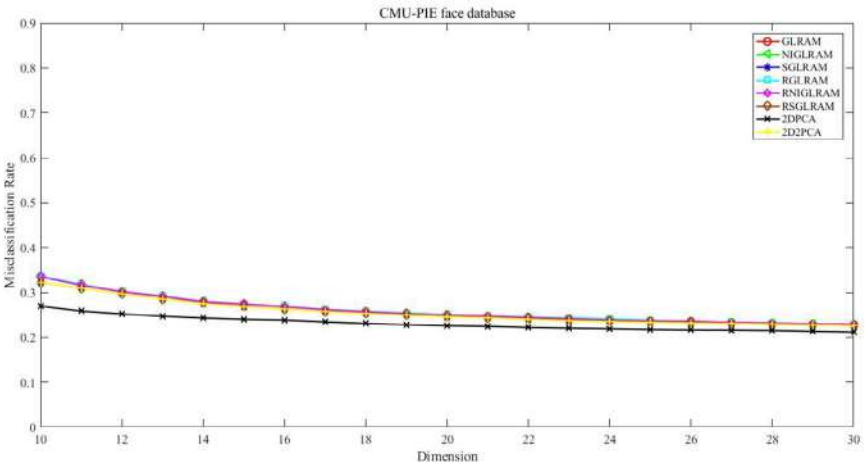
Eight algorithms including RGLRAM, RNIGLRAM, RSGLRAM, and their original counterparts, as well as 2DPCA[3] and (2D)$^2$PCA[5] are considered in this example. All the experiments are repeated for 10 times independently, and we make use of the nearest neighbor classifier (NN)[37] for classification, in which the distance is chosen as the Euclidean distance. Table 10 lists the misclassification ratios, standard deviations, and CPU time in seconds, and Figure 11 plots the average misclassification ratios obtained from these algorithms.

For the CMU-PIE face database, we observe from Table 10 and Figure 11 that our randomized algorithms are faster than their original counterparts, except for the RSGLRAM algorithm. Moreover, all of them run faster than 2DPCA and (2D)$^2$PCA, and the RNIGLRAM algorithm runs the best. The misclassification ratios of 2DPCA algorithm are the smallest, while those of the seven algorithms are comparable. However, as our randomized algorithms are more suitable for high-dimensional and small sample problems, the advantage in terms of CPU time is not obvious as before. We see that the RSGLRAM algorithm is even slower than its original counterpart, that is, the SGLRAM algorithm. Indeed, as was mentioned in Example 1, this is due to the fact that one has to explicitly form two matrices of size $r\times Nc$ and $c\times Nr$, respectively, as well as to compute their randomized singular value decompositions.

[¶]www.cs.cmu.edu/afs/cs/project/PIE/web/

**FIGURE 10** Example 4: Some samples of the CMU-PIE face database (of size $486 \times 640$)

**FIGURE 11** Example 4: Misclassification rates of the eight algorithms on the CMU-PIE face database




The `Nist Topic Detection and Tracking corpus`[38] [#] (TDT2 corpus text) consists of data collected during the first half of 1998 and taken from six sources, including two newswires (APW, NYT), two radio programs (VOA, PRI), and two television programs (CNN, ABC). It consists of 11,201 on-topic documents which are classified into 96 semantic categories. In this experiment, those documents appearing in two or more categories were removed, and only the largest 30 categories were kept, thus leaving us with 9,394 documents in total. In addition, this corpus contains 36,771 distinct terms. Under the premise that the operation object of this experiment is a matrix, those documents based on vectors are subsequently used in order to obtain matrices. By using the MATLAB command `reshape.m`, each document is reshaped to an image of $357 \times 103$ pixels in this experiment. Similar to the experiments performed on the `CMU-PIE` face database, we select 70% of each subject randomly for training (with $N = 6,589$) and the rest are used for testing. The numerical experiment is repeated for 10 times, and the numerical results are the mean of the ten runs. Table 10 and Figure 12 give the numerical results.

One observes from Table 10 and Figure 12 that the misclassification ratios of the three randomized algorithms are lower than those of 2DPCA and 2D$^2$PCA, and they are even lower than their original counterparts. One reason is that the randomized algorithms, just like the truncated singular value decomposition (TSVD),[25] have the effect of denoising. Moreover, our randomized algorithms run much faster than GLRAM, NIGLRAM, and SGLRAM, and they outperform 2DPCA and 2D$^2$PCA significantly. Specifically, RNIGLRAM performs the best, both in terms of CPU time and misclassification rates. Therefore, our proposed randomized GLRAM-type algorithms are competitive candidates for image compression and pattern recognition, especially for data sets with high dimension.

[#]http://www.nist.gov/speech/tests/tdt/tdt98/index.htm

**TABLE 10** Example 4: Face and text recognitions performances of eight algorithms on the CMU-PIE face database (of size $486 \times 640$, with 2,400 images) and the TDT2 text database (of size $357 \times 103$, with 9,394 documents); with $l = 30$

| | CMU-PIE face database ($486 \times 640$, $N = 1,200$) | | TDT2 text database ($357 \times 103$, $N = 6,589$) | |
|---|---|---|---|---|
| **Algorithms** | **(Misclassification ratio ± SD%)** | **Time (s)** | **(Misclassification ratio ± SD%)** | **Time (s)** |
| GLRAM | 26.00 ± 1.30% | 32.47 | 11.20 ± 0.39% | 18.55 |
| **RGLRAM** | 26.06 ± 1.31% | 18.36 | 10.28 ± 0.51% | 12.95 |
| NIGLRAM | 25.91 ± 1.29% | 11.15 | 11.20 ± 0.40% | 6.05 |
| **RNIGLRAM** | 25.95 ± 1.29% | 7.98 | 10.15 ± 0.46% | 5.97 |
| SGLRAM | 25.53 ± 1.27% | 16.79 | 11.16 ± 0.39% | 7.23 |
| **RSGLRAM** | 25.58 ± 1.29% | 18.84 | 10.12 ± 0.45% | 15.32 |
| 2DPCA | 23.06 ± 1.28% | 175.33 | 13.15 ± 1.75% | 74.70 |
| $(2D)^2$PCA | 25.53 ± 1.27% | 336.07 | 11.16 ± 0.39% | 213.69 |

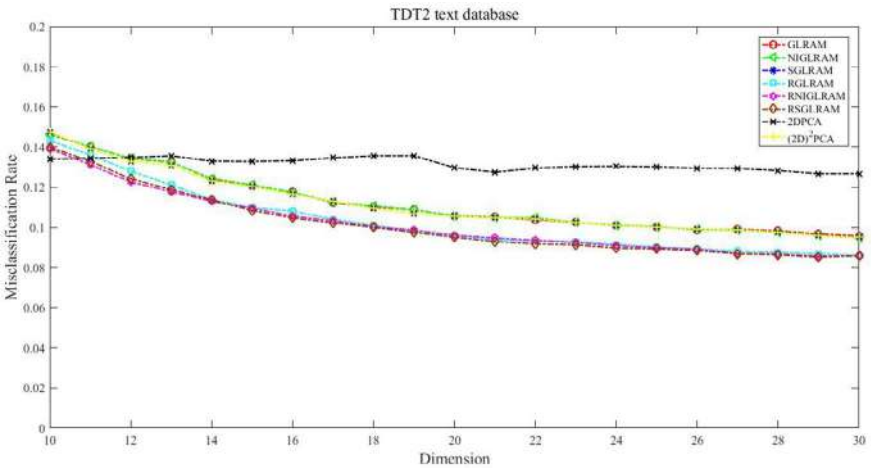



**FIGURE 12** Example 4: Misclassification rates of the eight algorithms on the TDT2 text database

## 5 | CONCLUDING REMARKS

In this article, we propose three randomized GLRAM-type algorithms with applications to image compression, face recognition, as well as text recognition, by knitting the randomized singular value decomposition (RSVD) together with three GLRAM-type algorithms. Theoretical results are established to show the validity and rationality of our proposed algorithms. Compared with their original counterparts, our randomized algorithms are much cheaper, and share similar reconstruction errors and recognition rates.

The main contribution of this article is to show why RSVD works for the GLRAM-type algorithms from a theoretical point of view. To do this, we first consider the decaying property of singular values of the matrices during iterations of the GLRAM algorithm, and determine the target rank $k$ in the RSVD process. At the first glance, it seems that $Nl \ll c$ and $Nl \ll r$ are crucial for the success of the randomized algorithms. However, we see from the numerical experiments that our randomized algorithms can still run much faster than their original counterparts as $Nl \geq r$ and $c$. This is due to the fact that the reducing dimension $l$ is small in practice, and we can apply RSVD to the transpose of $\tilde{G}_L$ and $\tilde{G}_R$ in this case. However, the new algorithms will not be so effective for large data sets with a huge number of images.

Second, we consider the relationship between the reconstruction errors generated by GLRAM and those by RGLRAM. It is shown that they are comparable, even if the solutions are computed inaccurately in the proposed algorithm. Third, we provide a sufficient condition for the convergence of the RGLRAM algorithm. Numerical experiments on some real-world data sets demonstrate the numerical behavior of the proposed algorithms, and show the effectiveness of our theoretical results.

In the era of big data, the size of the data is usually so huge that the data collections cannot be stored in main memory. In essence, the GLRAM-type algorithms can be viewed as a generalization to the singular value decomposition (SVD), the difference is that the former can provide low-rank approximations to a collection of matrices rather than a single matrix. From this standpoint, our proposed randomized GLRAM algorithms provide powerful alternatives to the popular SVD and RSVD algorithms.

## ACKNOWLEDGEMENT

Gang Wu is supported by the Fundamental Research Funds for the Central Universities under grant 2019XKQYMS89.

## ORCID

*Gang Wu* https://orcid.org/0000-0002-4936-437X

## REFERENCES

1. Fukunaga K. Introduction to statistical pattern classification. California: Academic Press, 1990.
2. Kirby M, Sirovich L. Application of the karhunen-loeve procedure for the characterization of human faces. IEEE Trans Pattern Anal Mach Intell. 1990;12:103–108.
3. Yang J, Zhang D, Frangi A, Yang JY. Two-dimensional PCA: a new approach to appearance based face representation and recognition. IEEE Trans Pattern Anal Mach Intell. 2004;26:131–137.
4. Turk M, Pentland A. Eigenfaces for recognition. J Cognit Neurosci. 1991;3:71–86.
5. Zhang D, Zhou Z. $(2D)^2$ PCA: 2-directional 2-dimensional PCA for efficient face representation and recognition. Neurocomputing. 2005;69:224–231.
6. Song H, Chen G, Wei H, Yang W. The improved $(2D)^2$PCA algorithm and its parallel implementation based on image block. Microprocess Microsyst. 2016;47:170–177.
7. Ye J. Generalized low rank approximations of matrices. Mach Learn. 2005;61:167–191.
8. Liu J, Chen S, Zhou ZH. Generalized low rank approximations of matrices revisited. IEEE Trans Neural Netw. 2010;21:621–632.
9. Liu J, Chen S. Non-iterative generalized low rank approximation of matrices. Pattern Recogn Lett. 2006;27:1002–1008.
10. Lu C, Liu W, An S. A simplified GLRAM algorithm for face recognition. Neurocomputing. 2008;72:212–217.
11. Ren C, Dai D. Bilinear Lanczos components for fast dimensionality reduction and feature extraction. Pattern Recogn. 2010;43:3742–3752.
12. Wu G, Xu W, Leng H. Inexact and incremental bilinear Lanczos components algorithm for high dimensionality reduction and image reconstruction. Pattern Recognit. 2015;48:244–263.
13. Nakouri H, Limam M. Robust generalized low rank approximation of matrices for image recognition. Paper presented at: Proceedings of the IEEE International Symposium on Signal Processing and Information Technology(ISSPIT); Limassol, Cyprus: 2016:203-207.
14. Inoue K, Hara K, Urahama K. Symmetric generalized low rank approximations of matrices. Paper presented at: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); Kyoto, Japan, 2012. p. 949-952.
15. Liang Z, Zhang D, Shi P. The theoretical analysis of GLRAM and its applications. Pattern Recognit. 2007;40:1032–1041.
16. Gu M. Subspace iteration randomization and singular value problems. SIAM J Sci Comput. 2015;37:A1139–A1173.
17. Halko N, Martinsson PG, Troop JA. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev. 2011;53:217–288.
18. Liberty E, Woolfe F, Martinsson P, Rokhlin V, Tygert M. Randomized algorithms for the low-rank approximation of matrices. Proceedings of the National Academy of Sciences of the United States of America; vol 104; 2007. p. 20167–20172.
19. Martinsson P. Randomized methods for matrix computations and analysis of high dimensional data; 2016. arXiv:1607.01649.
20. Martinsson P, Rokhlin V, Tygert M. A randomized algorithm for the decomposition of matrices. Appl Comput Harmonic Anal. 2011;30:47–68.
21. Bingham E, Mannila H. Random projection in dimensionality reduction: applications to image and text data. Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; San Francisco, California, 2001, p. 245–250.
22. Darnell G, Georgiev S, Mukherjee S, Engelhardt BE. Adaptive randomized dimension reduction on massive data. J Mach Learn Res. 2017;18:1–30.
23. Goel N, Bebis G, Nefian A. Face recognition experiments with random projection. Biometr Technol Human Identificat II. 2005;5779:426–437.
24. Wu K, Simon H. Thick-restart Lanczos mthod for large symmetric eigenvalue problems. SIAM J Matrix Anal Appl. 2000;22:602–616.
25. Golub GH, Van Loan CF. Matrix computations. 4th ed. Baltimore: Johns Hopkins University Press, 2014.
26. Musco C, Musco C. Randomized block Krylov methods for stronger and faster approximate singular value decomposition. Advances in neural information processing systems, Neural Information Processing Systems (NIPS), 10010 North Torrey Pines RD, LA Jolla, California 92037 USA: 2015; p. 1396–1404.
27. Woodruff D. Sketching as a tool for numerical linear algebra. Found Trends Theoret Comput Sci. 2014;10(1-2):1–157.
28. Shi W, Wu G. A randomized algorithm for trace-ratio problem with applications to high-dimension and large-sample data dimensionality reduction. Machine Learning, revised.
29. Wright J, Ganesh A, Yang A, Zhou Z, Ma Y. Sparsity and robustness in face recognition; 2011. arXiv:1111.1014v1.

30. Wu G, Feng T, Zhang L, Yang M. Inexact implementation using Krylov subspace methods for large scale exponential discriminant analysis with applications to high dimensionality reduction problems. Pattern Recognit. 2017;66:328–341.
31. Sun J. The perturbation bounds for eigenspaces of a definite matrix-pair. Numerische Mathematik. 1983;41:321–343.
32. Wedin P-Å. Perturbation bounds in connection with singular value decomposition. BIT. 1972;12:99–111.
33. Cheng G, Zhou P, Han J. Learningrotation-invariant convolutional neural networks for object detection in VHRoptical remote sensing images. IEEE Trans Geosci Remotesens. 2016;54:7405–7415.
34. Lyons M, Akamatsu S, Kamachi M, Gyoba J. Coding facial expressions with Gabor Wavelets. Proceedings of the 3rd IEEE International Conference on Automatic Face and Gesture Recognition; Nara, Japan: 1998. p. 200–205.
35. Saad Y. Numerical algorithms for large eigenvalus problems. 2nd ed. Philadelphia, PA: SIAM, 2011.
36. Sim T, Baker S, Bsat M. The CMU pose, illumination, and expression database. IEEE Trans Pattern Anal Mach Intell. 2003;25:1615–1618.
37. Cover T, Hart P. Nearest neighbor pattern classification. IEEE Trans Inf Theory. 1967;13:21–27.
38. Cai D, He X, Zhang W, Han J. Regularized locality preserving indexing via spectral regression. Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management (CIKM'07); Lisbon, Portugal: 2007. p. 741–750.

**How to cite this article:** Li K, Wu G. A randomized generalized low rank approximations of matrices algorithm for high dimensionality reduction and image compression. *Numer Linear Algebra Appl*. 2020;e2338. https://doi.org/10.1002/nla.2338