Efficient Dimension Reduction for High-Dimensional Matrix-valued Data

Dong Wang^{a,b,*}, Haipeng Shen^c, Young Truong^d

^a*Department of Statistics and Biostatistics, Rutgers University, New Brunswick, NJ 08854*

^b*Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08544*

^c*Innovation and Information Management, School of Business, Faculty of Business and Economics, The University of Hong Kong, Hong Kong, China*

^d*Department of Biostatistics, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599*

Abstract

Collection of groups of high-dimensional matrix-valued data is becoming increasingly common in many modern applications such as imaging analysis. The massive size of such data creates challenges in terms of computing speed and computer memory. Numerical techniques developed for small or moderate-sized datasets simply do not translate to such massive datasets. The need to analyze such data effectively calls for the development of efficient dimension reduction techniques. We propose a novel dimension reduction approach that has nice approximation property, computes fast for high dimensionality, and also explicitly incorporates the intrinsic two-dimensional structure of the matrices. We approximate each matrix as the product of a group-level left basis matrix, a group-level right basis matrix, and an

*Corresponding author.

Email addresses: `dong.v.wang@rutgers.edu` (Dong Wang), `haipeng@hku.hk` (Haipeng Shen), `truong@bios.unc.edu` (Young Truong)

individual-level coefficient matrix, which are estimated through a two-stage singular value decomposition. We discuss the connection of our proposal with existing approaches, and compare them both numerically and theoretically. We also obtain theoretical upper bounds on the approximation error of our method. In the numerical studies, ours is much faster than the most accurate one, comparable to the near-optimal one both computationally and theoretically, and more precise than the one that requires the same amount of memory.

Keywords: Dimension reduction, Matrix-valued data, Singular value decomposition

1. Introduction

As the technology advances, matrix-valued data are more and more common. For instance, a typical *functional magnetic resonance imaging* (fMRI) data set is usually represented as a group of matrices of the same size, where each matrix is the measurement of the blood oxygen level dependent contrast for one subject, with each column corresponding to a vectorized three-dimensional image at a certain time point, and each row being a sequence of temporal observations for a particular brain voxel.

These matrices are often of high or even ultra-high dimension that needs a large amount of memory. For instance, a collection of fMRI data for 100 subjects may consist of 100 matrices with the spatial dimension corresponding to as many as 200,000 voxels and the temporal dimension consisting of around 200 time points, which altogether requires about 30 *gigabytes* (GB) memory in double precision. Hence, it is crucial to develop a group-wise dimension

15 reduction technique that is precise and scales well for high-dimensional data,
16 which is the goal of the current paper.

17 Most conventional dimension reduction techniques were developed for
18 groups of vector-valued data, such as the popular principal component anal-
19 ysis (PCA) [1]. To apply these approaches directly to matrix-valued data, we
20 need to vectorize each matrix. The conventional one dimensional (1D) PCA
21 then projects vector-valued observations onto a set of orthogonal directions
22 that preserve the maximum amount of variation in the data. These directions
23 are characterized by the leading eigenvectors of the sample covariance matrix.
24 However, the vectorization ignores the intrinsic two-dimensional (2D) struc-
25 ture embedded in the matrices, and creates high-dimensional vectors that
26 increase computational/memory burden. This usually makes the follow-up
27 dimension reduction not efficient.

28 Several dimension reduction methods have been developed that incorpo-
29 rate the 2D structure of matrices. Motivated by 1DPCA, 2DPCA of Yang
30 et al. [2] projects each matrix onto the principal eigen-space of the row-
31 row covariance matrix without vectorization. 2DPCA can also be under-
32 stood through the perspective of a one-sided-type low rank approximation
33 to matrices. However, 2DPCA only takes into consideration the row-row
34 covariance matrix. To fully capture both the row-row and column-column
35 correlations, Ye [3] proposed the *generalized low rank approximations of ma-*
36 *trices* (GLRAM) approach which is a two-sided-type low rank approxima-
37 tion. The idea of GLRAM originates from the minimization of the sum of
38 squared residuals. The optimization criterion has no closed form solution
39 and naturally leads to an iterative algorithm that can be slow. To achieve

40 better computational efficiency, Ding and Ye [4] proposed a non-iterative
 41 algorithm named *two-dimensional singular value decomposition* (2DSVD)
 42 which only implements eigen-decomposition on the row-row and column-
 43 column covariance matrices. Zhang and Zhou [5] independently proposed
 44 *two-directional two-dimensional principal component analysis* ((2D)²PCA)
 45 that is intrinsically equivalent to 2DSVD. The reduction of the computation
 46 cost inevitably makes the reconstruction error of 2DSVD and (2D)²PCA
 47 larger than GLRAM, the optimal iterative procedure.

48 Besides computational speed, the limitation of the computer memory is
 49 another major hurdle that one has to tackle when analyzing massive data.
 50 Take the aforementioned fMRI data for example. The large amount of mem-
 51 ory needed is beyond general computer capacity and the various algorithms
 52 discussed above are hence not implementable.

53 To cope with memory-demanding data and further speed up the compu-
 54 tation, recently Crainiceanu et al. [6] proposed the *population value decom-*
 55 *position* (PVD) approach that essentially boils down to a two-step *singular*
 56 *value decomposition* (SVD) algorithm. In the first step, SVDs are applied
 57 separately to individual matrices that are of relatively small size, and the
 58 leading left and right singular vectors are retained. This can be performed
 59 either in parallel or sequentially and often requires much less memory. In
 60 the second step, the leading left and right singular vectors obtained in the
 61 first step are concatenated column-wise, respectively; and SVD is applied
 62 again to each concatenated matrix. These aggregated matrices are substan-
 63 tially smaller than the raw data matrices if one only keeps the few leading
 64 singular vectors. The resulting left singular vectors in the second step are

65 used to obtain the final approximation for the original matrices. Obviously,
 66 ignoring the higher-order singular vectors in the first step results in less accu-
 67 racy for PVD. But PVD effectively reduces the computational burden, and
 68 is applicable for high-dimensional matrices. Recently Eloyan et al. [7] incor-
 69 porated PVD nicely into a likelihood-based independent component analysis
 70 framework.

71 One drawback of PVD though is that the computational efficiency does
 72 come at the price of reduced approximation accuracy. In this article, we
 73 further improve PVD and develop an *adjusted PVD* (APVD) algorithm that
 74 has the same computational cost and requires the same amount of memory
 75 as PVD, but produces more precise results. In fact, APVD often performs
 76 as accurate as GLRAM and 2DSVD for matrices of small to moderate sizes
 77 when they can be computed.

78 The key idea of the APVD modification arises from the observation that
 79 PVD assigns equal weights in the group-level SVD to those leading singular
 80 vectors obtained in the first SVD step. We all know that the singular vectors
 81 have a natural order of relative importance as reflected by the correspond-
 82 ing singular values. Hence, we adjust PVD by incorporating the relative
 83 importance of the singular vectors, which indeed results in a more accurate
 84 estimation of the group components. The first step of APVD is the same as
 85 PVD. While in the second step, our APVD procedure concatenates the scaled
 86 singular vectors, i.e. the product of the singular vectors and their correspond-
 87 ing singular values from each individual matrix, instead of concatenating just
 88 the singular vectors. Furthermore, we establish theoretical justification for
 89 APVD in terms of upper bound on the normalized reconstruction errors.

90 The rest of this article is organized as follows. In Section 2, we state
91 the model and give a brief review of the GLRAM, 2DSVD, and PVD pro-
92 cedures. In Section 3, we then describe the APVD algorithm, and compare
93 the computational complexities of the various approaches along with their
94 connections. The theoretical properties of APVD are studied in Section 4.
95 Numerical comparisons through simulation studies and a classical face image
96 data set are presented in Section 5, to show that APVD performs comparable
97 to GLRAM and 2DSVD and better than PVD. We conclude in Section 6,
98 and relegate all proofs to Appendix A.

99 2. Preliminaries

100 2.1. The model

101 Consider there are I matrices of dimension $m \times n$, denoted as X_i , $i =$
102 $1, \dots, I$. To achieve group dimension reduction for the matrices, a reasonable
103 model can be written as

$$X_i = LW_iR^T + E_i, \quad (1)$$

104 where $L \in \mathbb{R}^{m \times r_L}$ and $R \in \mathbb{R}^{n \times r_R}$ are orthonormal matrices representing the
105 left and right group components respectively, $W_i \in \mathbb{R}^{r_L \times r_R}$ is the individual
106 coefficient matrix, and the error matrix E_i contains the individual approxi-
107 mation errors. Throughout this article, we assume that $\sum_{i=1}^I X_i = 0$. If we
108 choose $r_L \ll n$ and $r_R \ll m$, the size of the individual component W_i ($r_L r_R$)
109 is much smaller than the size of the original data (mn), which achieves the
110 goal of dimension reduction.

111 The decomposition of X_i as in (1) is closely related to the SVD of a single
112 matrix. Suppose $I = 1$, that is, there is only one matrix. Then the optimal

113 L and R that minimize the sum of squares of the approximation errors in
 114 E_i are the $r = \min(r_L, r_R)$ leading left and right singular vectors of X_1 ,
 115 and W_1 can always be required to be a diagonal matrix with the r leading
 116 singular values. When $I > 1$, Model (1) relaxes the requirement that all of
 117 the subject-specific terms W_i should be diagonal matrices and only keep the
 118 orthonormal constraints of the group components. The reason is that the
 119 subspace spanned by the columns of L (or R) can be thought of as the best
 120 rank r_L (or r_R) subspace that spans the column (or row) subspace of all the
 121 X_i 's; the W_i 's are the coefficients when projecting X_i onto L and R , which
 122 are not necessarily diagonal matrices.

123 2.2. Review of existing methods

124 The GLRAM, 2DSVD, PVD (and APVD) procedures offer different ways
 125 of estimating Model (1), as we shall review below. Least squares offers a nat-
 126 ural criterion for model estimation. It can be shown that the least squares
 127 estimator of W_i is given by $\widehat{W}_i = \hat{L}^T X_i \hat{R}$, once we obtain the group compo-
 128 nent estimates, $\hat{L}$ and $\hat{R}$. Therefore, for the rest of this article, we focus on
 129 the estimation of L and R . Moreover, for simplicity, we describe how each
 130 approach can be used to estimate the left component L ; R can be estimated
 131 in the same way using the transpose of X_i .

The GLRAM of Ye [3] borrows the minimum reconstruction error prop-
 erty of SVD and seeks L , R and W_i to minimize the reconstruction error in

the least squares sense:

$$\begin{aligned}
& \min_{L, R, W_i} \sum_{i=1}^I \|X_i - LW_i R^T\|_F^2, \\
& \text{s.t. } R^T R = I_{r_R}, \quad L^T L = I_{r_L}, \\
& L \in \mathbb{R}^{m \times r_L}, \quad R \in \mathbb{R}^{n \times r_R} \text{ and } W_i \in \mathbb{R}^{r_L \times r_R},
\end{aligned} \tag{2}$$

132 where $\|\cdot\|_F$ is the matrix Frobenius norm.

133 Ye [3] pointed out that the optimization problem (2) has no closed form
134 solutions for L and R . Hence, GLRAM solves the problem in an iterative
135 fashion. In each iteration, it alternates the updating of L (or R) as the leading
136 r_L (or r_R) eigenvectors of $\sum_{i=1}^I X_i R R^T X_i^T$ (or $\sum_{i=1}^I X_i^T L L^T X_i$), by fixing R
137 (or L) as the corresponding estimate obtained in the previous iteration. The
138 algorithm terminates until it reaches a certain convergence criterion.

139 The 2DSVD of Ding and Ye [4] offers an alternative estimation approach
140 that is non-iterative. It estimates L by either SVD of the concatenated data
141 matrix or eigen-decomposition of the column-column covariance matrix. One
142 can concatenate the matrices as $A = [X_1, X_2, \dots, X_I]$, and then perform SVD
143 on A to obtain the leading r_L left singular vectors as the estimate for L . Al-
144 ternatively, the relationship between eigen-decomposition and SVD suggests
145 that the left singular vectors of A are the same as the eigenvectors of the
146 matrix $A A^T$, which are equal to the eigenvectors of the column-column co-
147 variance matrix $\sum_{i=1}^I X_i X_i^T$. Note that the column-column covariance matrix
148 plays a role similar to the sample covariance matrix in the single-matrix case.
149 We comment that 2DSVD reduces the computational cost of the GLRAM
150 procedure but does not directly minimize the objective function in (2). Hence
151 it offers an approximation solution to the optimization problem (2).

152 Unfortunately, neither GLRAM nor 2DSVD is applicable for high-dimensional
 153 matrices. For example, in many neuroimaging studies, the dimension of each
 154 matrix and the number of subjects are usually very large. For the fMRI
 155 data example in Section 1, we have $m = 200,000$, $n = 200$ and $I = 100$. It
 156 follows that the sizes of the concatenated matrix A and the column-column
 157 covariance matrix $\sum_{i=1}^I X_i X_i^T$ are $200,000 \times 20,000$ and $200,000 \times 200,000$,
 158 which respectively require about 30 and 300 GB to store in double precision.
 159 Apparently, the enormous sizes of the matrices make their storage nearly
 160 impossible in general computers, and their SVD and eigen-decomposition
 161 computationally prohibitive.

162 The PVD approach of Crainiceanu et al. [6] aims at overcoming these
 163 physical computer memory and computation limitations when dealing with
 164 massive data. Recall that 2DSVD computes the left singular vectors of the
 165 concatenated matrix A . Instead of combining all the raw data matrices,
 166 the idea of PVD is to only concatenate the dominating singular vectors of
 167 the individual matrices. This idea naturally makes PVD a two-step SVD
 168 approach. Formally, PVD first performs SVD on each individual matrix,
 169 denoted as $X_i = U_i D_i V_i^T$ where U_i is the $m \times r_i$ orthonormal left singular
 170 vector matrix, D_i is the $r_i \times r_i$ diagonal singular value matrix with the diag-
 171 onal entries being the positive singular values in descending order, V_i is the
 172 $n \times r_i$ orthonormal right singular vector matrix, and r_i denotes the rank of X_i .
 173 Given the SVDs, only the first k_i^u and k_i^v columns of U_i and V_i are retained
 174 and denoted as $\tilde{U}_i$ and $\tilde{V}_i$ respectively. The matrices $\tilde{U}_i$, $i = 1, \dots, I$, are then
 175 concatenated as $P^* \equiv [\tilde{U}_1, \tilde{U}_2, \dots, \tilde{U}_I]$. In the second step, PVD estimates
 176 L with the first r_L left singular vectors of P^* . Similarly, PVD obtains the

177 estimate for R with the first r_R left singular vectors of $Q^* \equiv [\tilde{V}_1, \tilde{V}_2, \dots, \tilde{V}_I]$.

178 PVD replaces one SVD on a single large matrix with many SVDs on
179 relatively small matrices, and can efficiently reduce the computational cost.
180 Coming back to the previous fMRI data example, the size of each individual
181 matrix is $200,000 \times 200$ which requires about 0.3 GB memory and the corre-
182 sponding SVD can be easily computed. If we retain the first 10 left singular
183 vectors from each matrix, the aggregated matrix of the singular vectors is of
184 size $200,000 \times 1,000$ which requires about 1.5 GB, and it is still feasible to
185 obtain its SVD.

186 3. Adjusted population value decomposition

187 We propose to further improve PVD with one adjustment – proper incor-
188 poration of the singular values in D_i . Our motivation of the adjustment for
189 PVD is from the fact that PVD gives equal weights to all singular vectors. As
190 mentioned in Section 1, one important property of SVD is that the singular
191 vectors of a matrix has a relative importance order which is reflected by their
192 singular values. Taking this order information into consideration by scaling
193 each singular vector with its corresponding singular value, it can potentially
194 improve the estimation accuracy on the estimates of group components.

195 We present our adjusted population value decomposition (APVD) ap-
196 proach in this section. We first present the APVD algorithm in Section 3.1,
197 and then discuss its memory and computation complexities in Sections 3.2
198 and 3.3, and its connections with PVD and 2DSVD in Section 3.4.

199 *3.1. The APVD algorithm*

200 Our APVD procedure modifies the PVD approach by taking into account
 201 the relative importance of the singular vectors in the group-level SVD step
 202 of PVD, while keeping the first SVD step intact.

203 The complete APVD procedure is presented below in Algorithm 1.

Algorithm 1 The APVD algorithm

1. Perform SVD on each X_i and obtain $X_i = U_i D_i V_i^T$.

Let $\tilde{U}_i$ and $\tilde{V}_i$ denote the first k_i^u and k_i^v columns of U_i and V_i respectively.

Let $\tilde{D}_i^u$ and $\tilde{D}_i^v$ be the corresponding diagonal matrices with the diagonal elements being the first k_i^u and k_i^v singular values of X_i respectively.

Define $P \equiv [\tilde{U}_1 \tilde{D}_1^u, \tilde{U}_2 \tilde{D}_2^u, \dots, \tilde{U}_I \tilde{D}_I^u]$ and $Q \equiv [\tilde{V}_1 \tilde{D}_1^v, \tilde{V}_2 \tilde{D}_2^v, \dots, \tilde{V}_I \tilde{D}_I^v]$.

2. Perform SVD on P and Q .

Obtain the APVD estimates of L and R as the first r_L and r_R left singular vectors of P and Q respectively.

204 **Selection of k_i^u and k_i^v .** For both PVD and APVD, one needs to choose
 205 k_i^u and k_i^v in the individual-level SVD step. Our numerical experience sug-
 206 gests that as long as $k_i^u \geq r_L, k_i^v \geq r_R$, the specific choices of k_i^u and k_i^v do
 207 not matter that much. The condition that the number of the components
 208 preserved in the first step should be no smaller than the rank of the final es-
 209 timator is intuitive, in that we must not throw away any *useful* information
 210 in each individual matrix. Since the smaller these two quantities k_i^u and k_i^v
 211 are, the less time-consuming the algorithm is, we recommend to set $k_i^u = r_L$
 212 and $k_i^v = r_R$ in practice.

213 3.2. Memory complexity

214 We want to compare the amount of memory needed by each of the four
 215 methods: GLRAM, 2DSVD, PVD, and APVD. The four methods do not
 216 necessarily need to load all data matrices into memory at the same time
 217 except the SVD algorithm for 2DSVD, i.e. one can load one data matrix
 218 into memory at a time, perform the calculation and then remove it. Hence,
 219 we compare the memory requirement using the largest matrix size needed to
 220 do SVD or eigen-decomposition for each approach.

221 For ease of illustration, we assume that, for PVD and APVD, the num-
 222 bers of components kept in the second SVD step are the same across different
 223 subjects and equal the desired ranks, $k_i^u = r_L$, $k_i^v = r_R$, $i = 1, \dots, I$. Further-
 224 more, the ranks are assumed to be significantly smaller than the size of the
 225 original matrix, i.e. $r_L \ll n$, $r_R \ll m$, which is usually the case for dimension
 226 reduction to be useful. Without loss of generality, we assume that $m \geq n$.

227 GLRAM needs to do an eigen-decomposition of size $m \times m$ to obtain
 228 the estimate of L at each iteration. 2DSVD has two versions: the one that
 229 computes the SVDs of the concatenated matrix needs to do SVD on a matrix
 230 of size $m \times nI$ or $n \times mI$; the other one that computes the eigen-decomposition
 231 of the covariance matrices needs to calculate eigen-decomposition on a matrix
 232 of maximum size $m \times m$. As for PVD and APVD, the algorithms need to do
 233 SVDs on matrices of size $m \times n$ in the first step and of maximum size $m \times r_L I$
 234 during the second step. By introducing the notations $a \vee b = \max(a, b)$ and
 235 $a \wedge b = \min(a, b)$, the maximum matrix size to do SVD for PVD and APVD
 236 is $m \times (n \vee (r_L I))$. The above comparison results are summarized in Table
 237 1.

Maximum matrix size	APVD	PVD	2DSVD	GLRAM
for SVD	$m \times (n \vee (r_L I))$	$m \times (n \vee (r_L I))$	$m \times nI$ or $n \times mI$	
for eigen-decomposition			$m \times m$	$m \times m$

Table 1: The maximum matrix sizes for the four estimation approaches.

Back to the fMRI data example with $m = 200,000$, $n = 200$, $I = 100$, and $r_L = r_R = 10$ (for example). The matrices of size $m \times nI$ and $m \times m$ require 30 GB and 300 GB computer memory respectively. On the other hand, the matrix of size $m \times (n \vee (r_L I))$ only needs 1.5 GB memory. Hence, in this case, the 2DSVD and GLRAM approaches need much more memory than APVD and PVD.

3.3. Computation complexity

We compare the computational time complexities of these four algorithms as follows, which are summarized in Table 2.

- Suppose GLRAM converges after K iterations. During each iteration, one needs to compute $\sum_{i=1}^I X_i R R^T X_i^T$ and $\sum_{i=1}^I X_i^T L L^T X_i$, which takes $\mathcal{O}((m+n)(mr_R + nr_L)I)$ flops; the follow-up eigen-decomposition requires $\mathcal{O}(m^3 + n^3)$ flops. Hence, GLRAM can be computed in $\mathcal{O}(K(m+n)(mr_R + nr_L)I + Km^3 + Kn^3)$ flops.
- When 2DSVD computes the SVDs of $[X_1, X_2, \dots, X_I]$ and $[X_1^T, X_2^T, \dots, X_I^T]$ which are of size $m \times nI$ and $n \times mI$, the calculation takes $\mathcal{O}(mnI(m \wedge (nI) + n \wedge (mI)))$ flops. When 2DSVD computes the eigen-decompositions of $\sum_{i=1}^I X_i X_i^T$ and $\sum_{i=1}^I X_i^T X_i$, the matrix multiplication consumes $\mathcal{O}(m^2 nI + mn^2 I)$ and the eigen-decompositions takes $\mathcal{O}(m^3 + n^3)$ flops.

- 258 • The first step of PVD and APVD consists of I individual SVDs of
 259 matrices of size $m \times n$ and can be implemented in $\mathcal{O}(mn^2I)$ flops. The
 260 second step involves two SVDs of matrices of size $m \times r_L I$ and $n \times r_R I$
 261 and needs $\mathcal{O}(mr_L I(m \wedge (r_L I)) + nr_R I(n \wedge (r_R I)))$ flops.

	Computation Complexity
APVD	$\mathcal{O}(mn^2I + mr_L I(m \wedge (r_L I)) + nr_R I(n \wedge (r_R I)))$
PVD	$\mathcal{O}(mn^2I + mr_L I(m \wedge (r_L I)) + nr_R I(n \wedge (r_R I)))$
2DSVD	$\mathcal{O}(m^2nI + mn^2I + m^3 + n^3)$ or $\mathcal{O}(mnI(m \wedge (nI)) + n \wedge (mI))$
GLRAM	$\mathcal{O}(K(m+n)(mr_R + nr_L)I + Km^3 + Kn^3)$

Table 2: Comparison of computation complexity for the four approaches.

262 3.4. Connections of APVD with PVD and 2DSVD

263 GLRAM optimizes the least squares criterion but is computationally the
 264 most expensive approach. 2DSVD is less costly to compute and has been
 265 shown to be near-optimal [4]. Both PVD and APVD attempt to overcome
 266 the memory limitation of GLRAM and 2DSVD through individual dimension
 267 reduction prior to SVD of the concatenated matrix.

268 In this section, we further investigate the connection between APVD and
 269 2DSVD, as well as between APVD and PVD. We first show that 2DSVD and
 270 APVD produce similar estimates. Given the near-optimality of 2DSVD, it
 271 follows that APVD also possesses nice estimation property. We then prove
 272 that, under certain conditions, PVD and APVD recover the same subspace,
 273 although in most scenarios APVD estimates better.

274 *APVD and 2DSVD.* Consider the SVD of X_i as $X_i = U_i D_i V_i^T$. Note that
 275 APVD concatenates the leading components of the scaled singular vectors
 276 $\{U_i D_i\}$, while 2DSVD concatenates the original data matrices $\{X_i\}$. The
 277 following Proposition 1 suggests that if APVD concatenates the full set of
 278 the scaled singular vectors from each subject, then APVD and 2DSVD give
 279 the same estimates for L and R in Model (1). The proof will be provided in
 280 the appendix.

281 **Proposition 1.** *The concatenated data matrix $[X_1, X_2, \dots, X_I]$ and the con-*
 282 *catenated scaled singular vector matrix $[U_1 D_1, U_2 D_2, \dots, U_I D_I]$ have the same*
 283 *set of left singular vectors and the singular values. Similarly, $[X_1^T, X_2^T, \dots, X_I^T]$*
 284 *and $[V_1 D_1, V_2 D_2, \dots, V_I D_I]$ have the same set of left singular vectors and sin-*
 285 *gular values.*

286 Given the above equivalence, when APVD only concatenates the first k_i^u
 287 scaled singular vectors, instead of the full set, the estimates from APVD are
 288 not exactly the same as those from 2DSVD, but they are close as we show
 289 below. 2DSVD computes the left singular vectors of $[X_1, X_2, \dots, X_I]$, which
 290 are the eigenvectors of $\sum_{i=1}^I X_i X_i^T$. On the other hand, APVD calculates the
 291 left singular vectors of $[\tilde{U}_1 \tilde{D}_1^u, \tilde{U}_2 \tilde{D}_2^u, \dots, \tilde{U}_I \tilde{D}_I^u]$, which are the eigenvectors of
 292 $\sum_{i=1}^I \tilde{U}_i (\tilde{D}_i^u)^2 \tilde{U}_i^T$. Since $X_i X_i^T = U_i D_i^2 U_i^T \approx \tilde{U}_i (\tilde{D}_i^u)^2 \tilde{U}_i^T$, we obtain $\widehat{L^{APVD}} \approx$
 293 $\widehat{L^{2DSVD}}$.

294 *APVD and PVD.* We remind that $P^* \equiv [\tilde{U}_1, \tilde{U}_2, \dots, \tilde{U}_I]$, $Q^* \equiv [\tilde{V}_1, \tilde{V}_2, \dots, \tilde{V}_I]$,
 295 $P \equiv [\tilde{U}_1 \tilde{D}_1^u, \tilde{U}_2 \tilde{D}_2^u, \dots, \tilde{U}_I \tilde{D}_I^u]$, and $Q \equiv [\tilde{V}_1 \tilde{D}_1^v, \tilde{V}_2 \tilde{D}_2^v, \dots, \tilde{V}_I \tilde{D}_I^v]$. The second
 296 step of PVD calculates SVD of P^* and Q^* , while APVD does SVD on P and
 297 Q .

298 The PVD procedure concatenates the singular vectors while APVD con-
 299 catenates the scaled singular vectors. If every singular vector in the ag-
 300 gregated matrix of PVD has a corresponding scaled vector in APVD, the
 301 column spaces of the two aggregated matrices would be the same. On the
 302 other hand, a fundamental property of SVD is that the left singular vectors
 303 corresponding to the non-zero singular values represent an orthonormal basis
 304 of the matrix column space. Therefore, the full set of the left singular vectors
 305 with non-zero singular values of the two aggregated matrices are basis repre-
 306 senting the same subspace. Hence, there exists an orthogonal transformation
 307 relationship between the two full sets of singular vectors with non-zero sin-
 308 gular values. We summarize the relationship between APVD and PVD in
 309 the following proposition.

310 **Proposition 2.** *The ranks of P and P^* are the same and we denote it by*
 311 *r_P . Moreover, the first r_P left singular vectors of P^* are orthogonal rotations*
 312 *of the first r_P left singular vectors of P . Similar results hold for Q and Q^* .*

313 Proposition 2 shows the orthogonal transformation relationship between
 314 APVD and PVD only if $r_L = r_P$. In practice, r_L is usually smaller than
 315 r_P since we take k_i^u greater than or equal to r_L . Hence, in most cases,
 316 there does not exist an orthogonal transformation relationship between the
 317 estimates by PVD and APVD. Simulation results in Section 5 show that
 318 APVD outperforms PVD measured by subspace recovery and normalized
 319 reconstruction errors.

320 3.5. Connections of APVD with tensor decomposition

321 Consider the group of $m \times n$ matrices $\{X_i, i = 1, \dots, I\}$. One can orga-
 322 nize them as a three-way tensor of size $m \times n \times I$. Lock et al. [8] showed that
 323 the Tucker decomposition for a three-way tensor can be expressed equiva-
 324 lently as a PVD model. However, approaching the problem from the tensor
 325 perspective demands a large amount of memory for the following reason. The
 326 Tucker decomposition can be obtained efficiently via the higher-order SVD
 327 (HOSVD) algorithm [9], which intrinsically applies 2DSVD three times along
 328 the different tensor modes. As we already discussed in Section 1, when the
 329 dimensions of the matrices are very high, the 2DSVD algorithm is expensive
 330 in memory. On the other hand, in these cases, the PVD/APVD procedures
 331 can efficiently reduce the memory cost.

332 Recently, Ospina et al. [10] extended the PVD approach to a group of
 333 three-way tensors. The extension first performs HOSVD on each tensor,
 334 then concatenates the k th factor matrices across subjects horizontally for $k =$
 335 $1, 2, 3$, and finally applies SVD on the k th aggregated matrix to obtain the k th
 336 group tensor factor matrix. We comment that it is more challenging to extend
 337 APVD to a collection of tensors. Unlike in PVD that only involves the factor
 338 matrices, APVD also needs singular values to appropriately incorporate the
 339 relative importance of the factor matrices. The core tensor obtained from
 340 HOSVD for each tensor is usually not diagonal and hence can not be directly
 341 incorporated into APVD. We leave this interesting extension of APVD for
 342 future research.

343 4. Theoretical properties

344 In this section, we study some theoretical properties of APVD regarding
 345 normalized reconstruction error which is a measure of accuracy of the pro-
 346 posed procedure. We will show that the normalized reconstruction error is
 347 bounded from above by the normalized information lost of the two-step SVD
 348 for both one-sided and two-sided type approximations. Before we proceed to
 349 the main results, we first introduce some notations.

350 In both SVD steps of the APVD approach, we only retain the first few
 351 singular vectors and singular values which can explain most of the variance.
 352 The amount of information kept or lost in each step can be characterized by
 353 the singular values or eigenvalues. For the first SVD step, let

$$\theta^u = \min_{i \in \{1, \dots, I\}} \frac{\sum_{j=1}^{k_i^u} d_{ij}^2}{\sum_{j=1}^{r_i} d_{ij}^2}, \quad \text{and} \quad \theta^v = \min_{i \in \{1, \dots, I\}} \frac{\sum_{j=1}^{k_i^v} d_{ij}^2}{\sum_{j=1}^{r_i} d_{ij}^2}, \quad (3)$$

354 where d_{ij} is the j th singular value of X_i in descending order. Then θ^u and
 355 θ^v denote the minimum fraction of variances kept from individual matrices.

356 Recall that P and Q are defined as

$$P \equiv [\tilde{U}_1 \tilde{D}_1^u, \tilde{U}_2 \tilde{D}_2^u, \dots, \tilde{U}_I \tilde{D}_I^u], \quad \text{and} \quad Q \equiv [\tilde{V}_1 \tilde{D}_1^v, \tilde{V}_2 \tilde{D}_2^v, \dots, \tilde{V}_I \tilde{D}_I^v]$$

357 which are the aggregation matrices of the scaled singular vectors. Let d_{Pj} and
 358 d_{Qj} be the j th singular values of P and Q in descending order, respectively.
 359 Similar to θ^u and θ^v , we define

$$\theta^P = \frac{\sum_{j=1}^{r_P} d_{Pj}^2}{\sum_{j=1}^{r_P} d_{Pj}^2}, \quad \text{and} \quad \theta^Q = \frac{\sum_{j=1}^{r_Q} d_{Qj}^2}{\sum_{j=1}^{r_Q} d_{Qj}^2}, \quad (4)$$

360 which represent the information we retain in the second SVD step.

361 The normalized reconstruction error r is a commonly used metric to mea-
 362 sure and compare the performances of various approaches. It is defined as

$$r = \frac{\sum_{i=1}^I \|X_i - \hat{X}_i\|_F^2}{\sum_{i=1}^I \|X_i\|_F^2}, \quad (5)$$

363 where $\hat{X}_i$ denotes the reconstruction of X_i . We present the upper bound
 364 of r for the APVD approach in terms of θ^P , θ^Q , θ^u and θ^v in the following
 365 theorems.

366 4.1. Upper bound for the two-sided type approximation

367 The following theorem specifies how the normalized reconstruction error
 368 of the proposed APVD approach is bounded from above by the normalized
 369 information lost in the two SVD steps.

Theorem 1. *Consider Model (1) and assume that the estimates of L and R are given by the APVD procedure. Then the upper bound of the normalized reconstruction error r is*

$$r \leq (1 - \theta^u \theta^P) + (1 - \theta^v \theta^Q).$$

370 From the definitions, we know that θ^u denotes the minimum fraction
 371 of variances retained in the first SVD step, and θ^P represents the fraction
 372 of information kept in the second SVD step for estimating L . Hence, the
 373 fraction of variances kept in the two SVD steps is at least $\theta^u \theta^P$. It follows
 374 that the normalized information lost is at most $1 - \theta^u \theta^P$ for estimating L .
 375 Similarly, $1 - \theta^v \theta^Q$ is the largest normalized information lost for estimating R .
 376 Theorem 1 shows that the normalized reconstruction error is bounded from

above by the sum of the maximum fraction of variances lost while estimating the left and right components.

As one potential direction for future research, the above theoretical upper bound can be leveraged for outlier detection, i.e. data points that significantly decrease either θ^u or θ^v , and hence increase the upper bound.

4.2. Upper bound for the one-sided type approximation

We comment here that our APVD procedure can also be used to estimate group components of the 2DPCA model [2] which is a one-sided type low rank approximation of matrices. To be more specific, 2DPCA approximates each $m \times n$ matrix X_i , $i = 1, \dots, I$, by a product of one group-specific right component R^{2DPCA} of size $n \times r_R$ and one subject-specific term W_i^{2DPCA} of size $m \times r_R$, i.e.,

$$X_i = W_i^{2DPCA}(R^{2DPCA})^T + E_i, \quad (6)$$

where E_i of size $m \times n$ is the error matrix.

The APVD estimation procedures of L and R in the two-sided type model (1) are separate processes. Hence, the estimate of R obtained by APVD can also be used as an estimate of R^{2DPCA} in Model (6). The subject-specific term W_i^{2DPCA} can be obtained via $\widehat{W}_i^{2DPCA} = X_i \widehat{R^{2DPCA}}$. It follows that X_i can be reconstructed by $\widehat{X}_i = \widehat{W}_i^{2DPCA}(\widehat{R^{2DPCA}})^T$. The upper bound of the normalized reconstruction error under the 2DPCA model (6) is given by the following theorem.

Theorem 2. *Consider model (6) and assume the estimate of R^{2DPCA} is given by the APVD procedure. Then the upper bound of the normalized re-*

construction error r is

$$r \leq 1 - \theta^v \theta^Q.$$

Similar to the interpretation of Theorem 1, Theorem 2 shows that the normalized reconstruction error is bounded from above by the maximum fraction of variances lost when estimating the right group component.

5. Numerical studies

We evaluate the performance of our proposed APVD approach through simulations in Sections 5.1 and 5.2. The simulation results show that APVD performs comparable to GLRAM and 2DSVD in terms of estimation accuracy and all three methods are more accurate than PVD. We then apply the methods to a well-known face image database in Section 5.3.

We compare the performance of the methods in terms of subspace recovery, normalized reconstruction errors, and computational time. Since the columns of L and $\hat{L}$ form two sets of orthonormal basis, we use the following metric to measure the discrepancy between the corresponding subspaces:

$$D(\hat{L}, L) = \|\hat{L}\hat{L}^T - LL^T\|_2, \quad (7)$$

where $\|\cdot\|_2$ denotes the matrix spectral norm. This distance metric equals the sine of the largest canonical angle between two subspaces, and has been used by Golub and Van Loan [11] among others in the dimension reduction literature. We note that $0 \leq D(\hat{L}, L) \leq 1$ with a smaller value corresponding to a better estimate. Similarly, we can define the distance for the right group component as

$$D(\hat{R}, R) = \|\hat{R}\hat{R}^T - RR^T\|_2. \quad (8)$$

416 *5.1. Simulation: subspace recovery and normalized reconstruction errors*

Data are simulated according to Model (1):

$$X_i = LW_iR^T + E_i,$$

417 for $i = 1, 2, \dots, I$ with $I = 10$, $r_L = 10$ and $r_R = 6$. The j th column of L
 418 (and R) is $(0, 0, \dots, 0, 1, 0, \dots, 0)^T$ with the j th entry being 1 and the others
 419 being 0. Each entry of the individual component W_i is i.i.d. $N(0, 1)$, and
 420 each entry of the error matrix E_i is i.i.d. $N(0, \sigma^2)$ where σ is the noise level
 421 determined as follows. We use $r_L r_R / (mn\sigma^2)$ to approximate the signal to
 422 noise ratio (SNR). Then σ is given by $\sqrt{r_L r_R / (mn \cdot \text{SNR})}$. In all simulations,
 423 we consider SNR as 2 to calculate the corresponding value of σ .

424 For 2DSVD and GLRAM, we take the first $r_L = 10$ and $r_R = 6$ leading
 425 eigenvectors of the corresponding matrices as the estimates of L and R . For
 426 APVD and PVD, we take $k_i^u = 10$ and $k_i^v = 6$ in the first-step SVD and
 427 $r_L = 10$ and $r_R = 6$ in the second-step SVD.

428 Figure 1 shows the results for $m = 100$ and $n = 50$ over 100 simulation
 429 replications. Panels (a) and (b) depict the boxplots of the distances between
 430 the estimated subspaces and the underlying truth for L and R , respectively.
 431 Panel (c) compares the boxplots of the normalized reconstruction errors r (5).
 432 According to all three measures, APVD achieves comparable performance as
 433 the near-optimal 2DSVD and the optimal GLRAM, all of which outperform
 434 PVD.

435 Furthermore, Table 3 compares the average results for four (m, n) pairs
 436 over 100 runs: $(100, 20)$, $(100, 50)$, $(500, 100)$ and $(500, 250)$. We can see
 437 that GLRAM achieves the best performance in all three measures, APVD

Figure 1: Boxplots comparing four approaches for subspace recovery and normalized reconstruction errors over 100 simulation runs for $m = 100$ and $n = 50$. (a) Distances between the subspaces spanned by $\hat{L}$ and L defined in (7). (b) Distances between the subspaces spanned by $\hat{R}$ and R defined in equation (8). (c) Normalized reconstruction errors r defined in (5).

438 performs comparable to 2DSVD and outperforms PVD. In terms of comput-
 439 ing time, 2DSVD is the fastest one, while APVD and PVD have nearly the
 440 same speed, both of which are faster than the iterative GLRAM.

441 The above simulation results offer great support for APVD (over PVD). It
 442 performs comparable with 2DSVD and GLRAM in these small to moderate
 443 simulation studies. When GLRAM and 2DSVD can not be computed for
 444 high-dimensional matrices, we expect that APVD still outperforms PVD.

445 5.2. Simulation: choices of r_L , r_R , k_i^u and k_i^v

446 In this simulation, we study how different choices of r_L , r_R , k_i^u and k_i^v can
 447 affect the estimation of L and R . Data are simulated according to Model (1)
 448 with $I = 10$, $m = 100$ and $n = 50$. The true rank of L and R are chosen to
 449 be 15. The matrices L , R , W_i and E_i are simulated in the same way as in
 450 Section 5.1. We replicate the simulation 100 times.

m	n	Approach	$D(\hat{L}, L)$	$D(\hat{R}, R)$	r	Time
100	20	APVD	0.276 (0.030)	0.086 (0.012)	0.306 (0.012)	0.009 (0.001)
		PVD	0.502 (0.094)	0.147 (0.023)	0.335 (0.014)	0.009 (0.003)
		2DSVD	0.278 (0.030)	0.083 (0.011)	0.306 (0.012)	0.003 (0.000)
		GLRAM	0.267 (0.028)	0.078 (0.010)	0.305 (0.012)	0.038 (0.001)
100	50	APVD	0.177 (0.017)	0.080 (0.007)	0.322 (0.014)	0.020 (0.001)
		PVD	0.380 (0.063)	0.129 (0.014)	0.342 (0.014)	0.019 (0.000)
		2DSVD	0.179 (0.018)	0.079 (0.007)	0.322 (0.014)	0.007 (0.015)
		GLRAM	0.171 (0.015)	0.076 (0.007)	0.322 (0.014)	0.054 (0.002)
500	100	APVD	0.120 (0.010)	0.034 (0.003)	0.328 (0.013)	0.216 (0.009)
		PVD	0.213 (0.025)	0.067 (0.010)	0.334 (0.013)	0.215 (0.009)
		2DSVD	0.120 (0.011)	0.034 (0.003)	0.328 (0.013)	0.199 (0.007)
		GLRAM	0.119 (0.010)	0.034 (0.003)	0.328 (0.013)	1.750 (0.077)
500	250	APVD	0.076 (0.007)	0.033 (0.002)	0.333 (0.013)	0.737 (0.026)
		PVD	0.162 (0.020)	0.063 (0.009)	0.337 (0.013)	0.738 (0.028)
		2DSVD	0.076 (0.007)	0.033 (0.002)	0.333 (0.013)	0.343 (0.013)
		GLRAM	0.075 (0.007)	0.033 (0.002)	0.333 (0.013)	2.613 (0.106)

Table 3: Average results for four (m, n) pairs based on 100 simulation runs. Standard deviations are shown in parentheses.

451 For APVD and PVD, we choose the same number of singular vectors
452 in the first SVD step for each individual matrix, i.e. $k_i^u = k^u$ and $k_i^v =$
453 k^v for all i . Note that r_L and r_R represent the numbers of columns of $\hat{L}$
454 and $\hat{R}$ respectively. We study how the number of components affects the
455 normalized reconstruction error by letting (1) the four parameters equal and
456 vary together; (2) k^u and k^v are fixed at 20 while r_L and r_R change; and
457 (3) r_L and r_R are fixed at 5 while k^u and k^v vary. The results are shown in
458 Figure 2.

459 Figure 2(a) studies the question of how many components we need to
460 choose. We take $k^u = k^v = r_L = r_R$ and vary them together from 1 to 20.
461 Recall that the true ranks of L and R are 15. As the numbers of columns of
462 $\hat{L}$ and $\hat{R}$ (r_L and r_R) increase, the more components are used to approximate
463 the data matrices and hence the normalized reconstruction errors decrease
464 for all four methods. We note that the errors decrease quickly towards the

Figure 2: The choices of the numbers of singular vectors. (a) $k^u = k^v = r_L = r_R$ and vary from 1 to 20. (b) $k^u = k^v = 20$ and $r_L = r_R$ vary from 1 to 20. (c) $r_L = r_R = 5$ and $k^u = k^v$ vary from 5 to 20. The dotted vertical lines represent the locations of the true ranks of L and R .

465 true rank 15 and slowly after 15. Moreover, APVD, 2DSVD and GLRAM
 466 have comparable normalized reconstruction errors and are smaller than the
 467 PVD approach for all cases.

468 Figure 2(b) investigates how the number of components chosen in the
 469 second SVD step will affect the final estimates. We set $k^u = k^v = 20$ and
 470 let $r_L = r_R$ vary from 1 to 20. As the number of components increases, the
 471 errors decreases. As in Figure 2(a), there is a change of decreasing rate at
 472 the true rank 15. We can see that APVD has comparable performance as
 473 2DSVD and GLRAM and outperforms PVD.

474 Figure 2(c) studies the relation between the number of components chosen
 475 in the first SVD step and the final estimates. We set $r_L = r_R = 5$ and let
 476 $k^u = k^v$ change from 5 to 20. Here the numbers of the first SVD step (k^u and
 477 k^v) are chosen to be greater than or equal to the numbers of the second SVD

478 step (r_L and r_R). This is reasonable since we should preserve information in
 479 each matrix. We note the following observations: (1) the error of the PVD
 480 procedure goes up ; (2) the normalized error of APVD changes little as k^u and
 481 k^v increase; and (3) APVD is very close to 2DSVD. These observations show
 482 the advantages of incorporating the singular values in the analysis. Below
 483 we provide intuitive explanations for the observations.

484 We illustrate (1) and (2) using the left group component L . As k^u
 485 increases, new singular vectors or scaled singular vectors from individual
 486 subjects are gradually added into the aggregation matrices. For the PVD
 487 method, since the newly added singular vectors are treated equally to the
 488 previous ones, the leading eigenspace of the aggregation matrix will be af-
 489 fected by these new unit vectors. However, if the new vectors are scaled and
 490 the scales are relatively small compared to the previous ones, the principal
 491 eigenspace is still dominated by the previous scaled singular vectors. Hence,
 492 the eigenspace will remain stable as k^u increases. These explain why the nor-
 493 malized errors of the PVD procedure go up but those of the APVD algorithm
 494 only vary slightly. As for (3), recall from Section 3.4 and Proposition 1, the
 495 estimates from 2DSVD are the same as the APVD ones when we take the
 496 full sets of individual scaled singular vectors. From (2), we know that adding
 497 more individual scaled singular vectors will only slightly vary the leading
 498 eigenspace. Hence, the normalized reconstruction errors of APVD are very
 499 close to those of 2DSVD.

500 5.3. Application to the AT&T Database of Faces

501 We apply the four methods (GLRAM, 2DSVD, PVD, APVD) to the
 502 well-known AT&T Database of Faces [12], and demonstrate their real ap-

503 plicabilities. The database contains 400 gray-scale images of faces of size
 504 92×112 from 40 individuals. There are 10 face images per individual.

For each individual subject, let $X_i \in \mathbb{R}^{92 \times 112}$ denote the i th image, $i = 1, 2, \dots, 10$. We first subtract the mean image $\bar{X} = \sum_{i=1}^{10} X_i / 10$ from each image, and consider the following model for each individual:

$$X_i - \bar{X} = LW_i R^T + E_i.$$

We then apply the four methods to estimate the model, and obtain the reconstruction for each image as

$$\widehat{X}_i = \widehat{L}\widehat{L}^T(X_i - \bar{X})\widehat{R}\widehat{R}^T + \bar{X}.$$

505 Here, we take $k^u = k^v = r_L = r_R = 20$. We perform the model fitting and
 506 reconstruction separately for each of the 40 subjects.

507 For a particular subject, Figure 3 shows the original face images, and
 508 the reconstruction results from each method. The first row contains the
 509 original images, while the second to the fifth rows are the reconstructed
 510 images given by APVD, PVD, 2DSVD and GLRAM, respectively. As one
 511 can see, the reconstructed images given by APVD are visually very similar
 512 to the ones from 2DSVD and GLRAM, all of which are better than the PVD
 513 reconstructions and capture more finer details.

514 We then randomly choose 10 representative subjects and select 1 repre-
 515 sentative image for each subject. The images and the reconstructions are
 516 shown in Figure 4. We can make the same observations as in Figure 3 that
 517 APVD gives finer reconstructions than PVD, and works comparably with
 518 2DSVD and GLRAM.

Figure 3: One particular subject: the true images (first row) and the reconstructed images by APVD (second row), PVD (third row), 2DSVD (fourth row), and GLRAM (fifth row).

519 6. Conclusions

520 We consider the problem of dimension reduction for groups of matrix-
521 valued data, motivated by analysis of high-dimensional neuroimaging data.
522 We develop a computationally efficient method - *adjusted population value*
523 *decomposition* (APVD) that requires significantly less memory than most of
524 the existing methods. Our method performs comparably with the state of the
525 art algorithms such as GLRAM and 2DSVD when they can be computed for
526 matrices of small-to-moderate sizes, and improves the performance of PVD
527 by a considerable amount but maintains the nice property of PVD that it
528 requires little storage space. Furthermore, we establish the error bound of
529 APVD theoretically and demonstrate its superior performance numerically.

Figure 4: Ten representative subjects with one representative image per subject.

530 Acknowledgments

531 The authors would like to thank the anonymous reviewers for their helpful
532 comments. Haipeng Shen’s research was partially supported by US NSF
533 grants DMS-1106912 and DMS-1407655, and the University of Hong Kong
534 Stanley Ho Alumni Challenge Fund. Dong Wang’s and Young Truong’s work
535 was supported by US NSF grant DMS-1106962.

536 Appendix A. Proofs

537 Before we proceed to the proofs, we introduce some notations. Let u_{ij}
538 and v_{ij} denote the j th column of U_i and V_i respectively for $j = 1, \dots, r_i$.
539 We define P and Q as the aggregated matrices $[\tilde{U}_1 \tilde{D}_1^u, \tilde{U}_2 \tilde{D}_2^u, \dots, \tilde{U}_I \tilde{D}_I^u]$ and
540 $[\tilde{V}_1 \tilde{D}_1^v, \tilde{V}_2 \tilde{D}_2^v, \dots, \tilde{V}_I \tilde{D}_I^v]$. Write $M = PP^T$ and $N = QQ^T$. Let the eigen-

541 decomposition of M be

$$M = U_M \Lambda_M U_M^T,$$

542 where U_M is an m by r_M matrix with orthonormal columns, Λ_M is an r_M by
 543 r_M diagonal matrix with diagonal entries $\lambda_{M1} \geq \lambda_{M2} \geq \dots \geq \lambda_{Mr_M} > 0$, and
 544 r_M denotes the rank of M . Similarly, we can define the eigen-decomposition
 545 of N by

$$N = V_N \Lambda_N V_N^T,$$

546 with an orthonormal matrix V_N of size $n \times r_N$ and a diagonal matrix Λ_N
 547 with diagonal entries $\lambda_{N1} \geq \lambda_{N2} \geq \dots \geq \lambda_{Nr_N} > 0$.

548 From the relation between eigen-decomposition and SVD, we know that
 549 the eigenvectors of M and the left singular vectors of P are the same, and
 550 $\lambda_{Mj} = d_{Pj}^2$. Similarly, the eigenvectors of N and the left singular vectors of Q
 551 are the same, and $\lambda_{Nj} = d_{Qj}^2$. Hence, the estimates of the APVD procedure
 552 can be obtained from the eigenvectors of M and N .

553 To be more specific, let $\tilde{U}_M$ and $\tilde{V}_N$ consist of the first r_L and r_R columns
 554 of U_M and V_N respectively. Then $\tilde{U}_M$ and $\tilde{V}_N$ are the final estimates of L
 555 and R by the APVD approach. For the rest of this article, we will use the
 556 eigenvectors and eigenvalues of M and N instead of the left singular vectors
 557 of P and Q . Moreover, we define the following quantities

$$\theta^M = \frac{\sum_{t=1}^{r_L} \lambda_{Mt}}{\sum_{t=1}^{r_M} \lambda_{Mt}}, \quad \text{and} \quad \theta^N = \frac{\sum_{t=1}^{r_R} \lambda_{Nt}}{\sum_{t=1}^{r_N} \lambda_{Nt}}, \quad (\text{A.1})$$

558 where θ^M and θ^N represent the normalized information retained in the second
 559 SVD step. Then we have $\theta^M = \theta^P$ and $\theta^N = \theta^Q$.

Proof of Proposition 1. Let F denote the aggregated data matrix $[X_1, X_2, \dots, X_I]$
 and G denote the concatenated matrix $[U_1 D_1^u, U_2 D_2^u, \dots, U_I D_I^u]$. Firstly we

note that the left singular vectors of F and G are the same as the eigenvectors of FF^T and GG^T . The singular values of F and G are the square roots of the corresponding eigenvalues of FF^T and GG^T . Hence we only need to show that $FF^T = GG^T$. Observe that

$$FF^T = \sum_{i=1}^I X_i X_i^T = \sum_{i=1}^I \sum_{j=1}^{r_i} d_{ij}^2 u_{ij} u_{ij}^T,$$

and

$$GG^T = \sum_{i=1}^I U_i D_i^u D_i^u U_i^T = \sum_{i=1}^I \sum_{j=1}^{r_i} d_{ij}^2 u_{ij} u_{ij}^T.$$

Then we have

$$FF^T = GG^T.$$

Proof of Theorem 2. We can express M and N in terms of the singular vectors and the singular values of the individual matrices as follows:

$$M = PP^T = \sum_{i=1}^I \sum_{j=1}^{k_i^u} d_{ij}^2 u_{ij} u_{ij}^T, \quad (\text{A.2})$$

$$N = QQ^T = \sum_{i=1}^I \sum_{j=1}^{k_i^v} d_{ij}^2 v_{ij} v_{ij}^T. \quad (\text{A.3})$$

The estimate $\hat{L}$ is given by $\tilde{U}_M$. We first give a lower bound on the sum of squares of the reconstructions $\alpha = \sum_{i=1}^I \|\tilde{U}_M \tilde{U}_M^T X_i\|_F^2$. It is clear that

$$\alpha = \sum_{i=1}^I \text{tr}(\tilde{U}_M \tilde{U}_M^T X_i X_i^T \tilde{U}_M \tilde{U}_M^T) = \sum_{i=1}^I \text{tr}(\tilde{U}_M^T X_i X_i^T \tilde{U}_M),$$

where $\text{tr}(\cdot)$ denote the trace of a square matrix. The second equality holds because the columns of $\tilde{U}_M$ are orthonormal. If we exchange the order of the trace and sum operators and write $X_i X_i^T$ in terms of the singular vectors

and the singular values of X_i , we have

$$\alpha = \text{tr} \left(\tilde{U}_M^T \left(\sum_{i=1}^I X_i X_i^T \right) \tilde{U}_M \right) = \text{tr} \left(\tilde{U}_M^T \left(\sum_{i=1}^I \sum_{j=1}^{r_i} d_{ij}^2 u_{ij} u_{ij}^T \right) \tilde{U}_M \right).$$

Partitioning the first k_i^u singular vectors and singular values into one group and the rest into the other group for each matrix X_i yields

$$\alpha = \text{tr} \left(\tilde{U}_M^T \left(\sum_{i=1}^I \sum_{j=1}^{k_i^u} d_{ij}^2 u_{ij} u_{ij}^T \right) \tilde{U}_M \right) + \text{tr} \left(\tilde{U}_M^T \left(\sum_{i=1}^I \sum_{j=k_i^u+1}^{r_i} d_{ij}^2 u_{ij} u_{ij}^T \right) \tilde{U}_M \right).$$

The matrix $\sum_{i=1}^I \sum_{j=k_i^u+1}^{r_i} d_{ij}^2 u_{ij} u_{ij}^T$ is positive semidefinite. Then, it follows that

$$\text{tr} \left(\tilde{U}_M^T \left(\sum_{i=1}^I \sum_{j=k_i^u+1}^{r_i} d_{ij}^2 u_{ij} u_{ij}^T \right) \tilde{U}_M \right) \geq 0,$$

and hence

$$\alpha \geq \text{tr} \left(\tilde{U}_M^T \left(\sum_{i=1}^I \sum_{j=1}^{k_i^u} d_{ij}^2 u_{ij} u_{ij}^T \right) \tilde{U}_M \right).$$

Note that $\sum_{i=1}^I \sum_{j=1}^{k_i^u} d_{ij}^2 u_{ij} u_{ij}^T$ is M by equation (A.2). Together with equation (A.1), we have

$$\alpha \geq \text{tr}(\tilde{U}_M^T M \tilde{U}_M) = \sum_{t=1}^{r_L} \lambda_{Mt} = \theta^M \sum_{t=1}^{r_M} \lambda_{Mt}.$$

We note that $\sum_{t=1}^{r_M} \lambda_{Mt} = \text{tr}(M)$ and by (A.2), we have

$$\text{tr}(M) = \text{tr} \left(\sum_{i=1}^I \sum_{j=1}^{k_i^u} d_{ij}^2 u_{ij} u_{ij}^T \right) = \sum_{i=1}^I \sum_{j=1}^{k_i^u} d_{ij}^2.$$

Together with the definition of θ^u in equation (3), it follows that

$$\alpha \geq \theta^M \theta^u \sum_{i=1}^I \sum_{j=1}^{r_i} d_{ij}^2.$$

The Frobenius norm of each data matrix is connected with its singular values, which leads to $\|X_i\|_F^2 = \sum_{j=1}^{r_i} d_{ij}^2$. Thus,

$$\alpha \geq \theta^M \theta^u \sum_{i=1}^I \|X_i\|_F^2.$$

Now we can establish the upper bound for the normalized reconstruction error r through α as follows.

$$\frac{\sum_{i=1}^I \|X_i - \tilde{U}_M \tilde{U}_M^T X_i\|_F^2}{\sum_{i=1}^I \|X_i\|_F^2} = 1 - \frac{\sum_{i=1}^I \|\tilde{U}_M \tilde{U}_M^T X_i\|_F^2}{\sum_{i=1}^I \|X_i\|_F^2} \leq 1 - \theta^u \theta^M = 1 - \theta^u \theta^P.$$

Similarly, we can prove that

$$\frac{\sum_{i=1}^I \|X_i - X_i \tilde{V}_N \tilde{V}_N^T\|_F^2}{\sum_{i=1}^I \|X_i\|_F^2} = 1 - \frac{\sum_{i=1}^I \|X_i \tilde{V}_N \tilde{V}_N^T\|_F^2}{\sum_{i=1}^I \|X_i\|_F^2} \leq 1 - \theta^v \theta^N = 1 - \theta^v \theta^Q.$$

560 **Proof of Theorem 1.** Let

$$\alpha = \sum_{i=1}^I \|\tilde{U}_M \tilde{U}_M^T X_i\|_F^2, \quad \text{and} \quad \beta = \sum_{i=1}^I \|X_i \tilde{V}_N \tilde{V}_N^T\|_F^2.$$

561 Then from the proof of Theorem 2, we know that

$$\alpha = \sum_{i=1}^I \|\tilde{U}_M \tilde{U}_M^T X_i\|_F^2 = \sum_{i=1}^I \|\tilde{U}_M^T X_i\|_F^2 \geq \theta^M \theta^u \sum_{i=1}^I \|X_i\|_F^2, \quad (\text{A.4})$$

562 and

$$\beta = \sum_{i=1}^I \|X_i \tilde{V}_N \tilde{V}_N^T\|_F^2 = \sum_{i=1}^I \|X_i \tilde{V}_N\|_F^2 \geq \theta^N \theta^v \sum_{i=1}^I \|X_i\|_F^2. \quad (\text{A.5})$$

Let $\tilde{U}_M^c \in \mathbb{R}^{m \times (m-r_L)}$ and $\tilde{V}_N^c \in \mathbb{R}^{n \times (n-r_R)}$ be two orthonormal matrices such that

$$\tilde{U}_M \tilde{U}_M^T + \tilde{U}_M^c \tilde{U}_M^{cT} = I_{m \times m}, \quad \text{and} \quad \tilde{V}_N \tilde{V}_N^T + \tilde{V}_N^c \tilde{V}_N^{cT} = I_{n \times n}.$$

Then the reconstruction error is given by

$$\sum_{i=1}^I \|X_i - \tilde{U}_M \tilde{U}_M^T X_i \tilde{V}_N \tilde{V}_N^T\|_F^2 = \sum_{i=1}^I \text{tr}(\tilde{U}_M \tilde{U}_M^T X_i \tilde{V}_N^c \tilde{V}_N^{cT} X_i^T + \tilde{U}_M^c \tilde{U}_M^{cT} X_i X_i^T).$$

563 Since the matrix $\tilde{U}_M^{cT} X_i \tilde{V}_N^c \tilde{V}_N^{cT} X_i^T \tilde{U}_M^c$ is positive semidefinite, we have

$$\text{tr}(\tilde{U}_M^{cT} X_i \tilde{V}_N^c \tilde{V}_N^{cT} X_i^T \tilde{U}_M^c) \geq 0.$$

564 Thus,

$$\begin{aligned} & \sum_{i=1}^I \|X_i - \tilde{U}_M \tilde{U}_M^T X_i \tilde{V}_N \tilde{V}_N^T\|_F^2 \\ & \leq \sum_{i=1}^I \text{tr} \left(\tilde{U}_M \tilde{U}_M^T X_i \tilde{V}_N^c \tilde{V}_N^{cT} X_i^T + \tilde{U}_M^c \tilde{U}_M^{cT} X_i X_i^T + \tilde{U}_M^c \tilde{U}_M^{cT} X_i \tilde{V}_N^c \tilde{V}_N^{cT} X_i^T \right) \\ & = \sum_{i=1}^I \left(\|X_i \tilde{V}_N^c\|_F^2 + \|\tilde{U}_M^{cT} X_i\|_F^2 \right). \end{aligned}$$

By the following equalities

$$\|\tilde{U}_M^{cT} X_i\|_F^2 = \|X_i\|_F^2 - \|\tilde{U}_M^T X_i\|_F^2, \quad \text{and} \quad \|X_i \tilde{V}_N^c\|_F^2 = \|X_i\|_F^2 - \|X_i \tilde{V}_N\|_F^2,$$

565 and together with (A.4) and (A.5), we can establish the upper bound for the

566 normalized reconstruction error as follows.

$$\begin{aligned} & \frac{\sum_{i=1}^I \|X_i - \tilde{U}_M \tilde{U}_M^T X_i \tilde{V}_N \tilde{V}_N^T\|_F^2}{\sum_{i=1}^I \|X_i\|_F^2} \\ & \leq \frac{\sum_{i=1}^I \left(\|X_i\|_F^2 - \|X_i \tilde{V}_N\|_F^2 \right) + \sum_{i=1}^I \left(\|X_i\|_F^2 - \|\tilde{U}_M^T X_i\|_F^2 \right)}{\sum_{i=1}^I \|X_i\|_F^2} \\ & \leq (1 - \theta^N \theta^v) + (1 - \theta^M \theta^u) \\ & = (1 - \theta^v \theta^Q) + (1 - \theta^u \theta^P). \end{aligned}$$

567 **References**

- 568 [1] I. Jolliffe, *Principal Component Analysis*, Springer, second edn., 2002.
- 569 [2] J. Yang, D. Zhang, A. F. Frangi, J. Yang, Two-Dimensional PCA: A New
570 Approach to Appearance-Based Face Representation and Recognition,
571 IEEE Trans. Pattern Anal. Mach. Intell. 26 (1) (2004) 131–137.
- 572 [3] J. Ye, Generalized Low Rank Approximations of Matrices, Mach. Learn.
573 61 (1-3) (2005) 167–191.
- 574 [4] C. Ding, J. Ye, 2-Dimensional Singular Value Decomposition for 2D
575 Maps and Images, in: Proc. SIAM Int. Conf. Data Mining (SDM’05),
576 32–43, 2005.
- 577 [5] D. Zhang, Z. Zhou, (2D)²PCA: Two-directional two-dimensional PCA
578 for efficient face representation and recognition, Neurocomputing 69 (1-
579 3) (2005) 224–231.
- 580 [6] C. M. Crainiceanu, B. S. Caffo, S. Luo, V. M. Zipunnikov, N. M. Pun-
581 jabi, Population Value Decomposition, a Framework for the Analysis of
582 Image Populations, J. Am. Stat. Assoc. 106 (495) (2011) 775–790.
- 583 [7] A. Eloyan, C. M. Crainiceanu, B. S. Caffo, Likelihood-based population
584 independent component analysis, Biostatistics 14 (3) (2013) 514–527.
- 585 [8] E. F. Lock, A. B. Nobel, J. S. Marron, Comment: Population Value
586 Decomposition, a Framework for the Analysis of Image Populations, J.
587 Am. Stat. Assoc. 106 (495) (2011) 798–802.

- 588 [9] T. G. Kolda, B. W. Bader, Tensor Decompositions and Applications,
589 SIAM Rev. 51 (3) (2009) 455–500.
- 590 [10] J. D. Ospina, F. Commandeur, R. Ríos, G. Dréan, J. C. Correa, A. Si-
591 mon, P. Haigron, R. de Crevoisier, O. Acosta, A Tensor-Based Pop-
592 ulation Value Decomposition to Explain Rectal Toxicity after Prostate
593 Cancer Radiotherapy, in: Med. Image Comput. Comput. Assist. Interv.,
594 387–394, 2013.
- 595 [11] G. H. Golub, C. F. Van Loan, Matrix computations, Johns Hopkins
596 Studies in the Mathematical Sciences, Johns Hopkins University Press,
597 Baltimore, MD, third edn., 1996.
- 598 [12] F. S. Samaria, A. C. Harter, Parameterisation of a stochastic model for
599 human face identification, in: Proceedings of the Second IEEE Work-
600 shop on Applications of Computer Vision, 138–142, 1994.