Higher Order Orthogonal Iteration of Tensors (HOOI) and its Relation to PCA and GLRAM *

Bernard N. Sheehan † Yousef Saad ‡

March 19, 2007

Abstract

This paper presents a unified view of a number of dimension reduction techniques under the common framework of tensors. Specifically, it is established that PCA, and the recently introduced 2-D PCA and Generalized Low Rank Approximation of Matrices (GLRAM), are special instances of the higher order orthogonal iteration of tensors (HOOI). The connection of these algorithms to HOOI has not been pointed out before in the literature. The pros and cons of these specializations versus HOOI are discussed.

Keywords: Tensor, HOOI , HOSVD, Dimension Reduction, Principal Component Analysis, GLRAM

1 Introduction

Recently there has been a surge of interest in the use of low rank approximations to tensors as a general technique for dimension reduction of large amounts of data in applications such as data mining and information retrieval [12, 8], face recognition [14], texture modeling [15, 17], speech discrimination [11], and computer graphics [16]. The High-Order Singular Value Decomposition (HOSVD) algorithm [3], and its low-rank counterpart, the Higher Order Orthogonal Iteration of Tensors (HOOI), see [4], can be viewed as natural extensions to the Singular Value Decomposition (SVD) and Principal Component Analysis (PCA), when one is confronted with multifactorial or N-way data rather than a common matrix.

A model problem along these lines is the following. We are given a set of matrices $M_k, k = 1, 2, \ldots K$, all of the same dimension $I \times J$. For concreteness, we can think of these matrices as being bitmap pictures in a face database, or successive images in a motion picture, or images of a texture viewed from various angles and under various lighting conditions. Whatever the source and nature of these images, the problem we have before us is how to approximately represent these images by a lower rank approximation. Figure 1 shows how such a set of images might be viewed as a tensor. The tensor viewpoint will be emphasized in this paper.

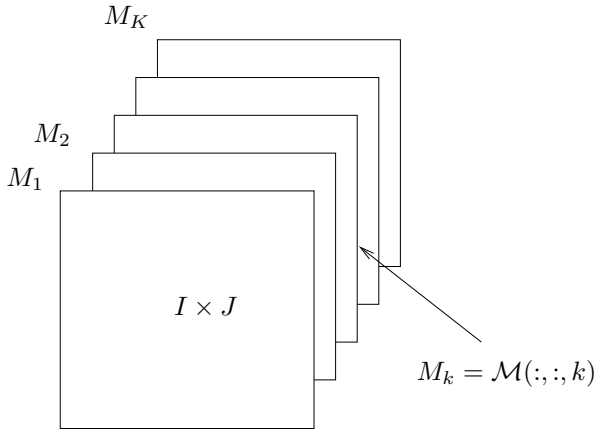

Figure 1: Set of K matrices $M_k \in \mathbb{R}^{I \times J}$ represented as an $I \times J \times K$ tensor $\mathcal{M}$

There are a number of approaches to compressing such a set of images, each of which gives a different interpretation to the phrase 'lower rank approximation'. Three typical approaches to dimension reduction of image sets are:

1. Standard SVD and PCA
2. GLRAM and 2DPCA
3. HOSVD and HOOI

Principal Component Analysis (PCA), which is founded on the Singular Value Decomposition (SVD), has been widely used in statistics and elsewhere [6]. An algorithm known as 'Generalized Low Rank Approximation of Matrices' (GLRAM) [20] has been recently proposed as an alternative to PCA that retains the 2-D structure of images and avoids first

*This work was supported by NSF grants DMS 0510131 and DMS 0528492 and by the Minnesota Supercomputing Institute

†Computer Science & Engineering, University of Minnesota, Twin Cities. sheehan@cs.umn.edu

‡Computer Science & Engineering, University of Minnesota, Twin Cities. saad@cs.umn.edu

vectorizing them as is done in PCA. A number of researcher have proposed essentially the same algorithm as GLRAM but under different names, G2DPCA and Coupled Subspace Analysis [9, 18]. Higher-Order Orthogonal Iteration or HOOI [4] takes a further conceptual leap by regarding a set of matrices as a single entity, a 'tensor', or multi-dimensional data array, and attempts to extend the truncated SVD algorithm to such data objects.

While on the surface PCA, GLRAM, and HOOI appear to be distinct computational choices, we will demonstrate in this paper that HOOI encompases both PCA and GLRAM as special cases. The fact that well-known and successful algorithms like PCA and GLRAM can be regarded merely as special cases of HOOI provides, in itself, a compelling argument for the power and generality of the tensor point of view. The subordinate relation of PCA and GLRAM to HOOI has not, to our knowledge, been documented elsewhere.

In [20] it is argued that a combination of GLRAM and PCA provides better reconstruction accuracy that GLRAM by itself; we shall also demonstrate that such a composite dimension-reduction strategy is simply a disguised HOOI subjected to a restricted iteration scheme.

1.1 SVD and PCA To apply Principal Component Analysis (PCA) to dimension reduction of a set of images, one must first *vectorize* the images. Each image $M_i \in R^{I\times J}$ is rearranged into a vector $x_i \in R^N$ where $N = IJ$ is the number of pixels in the image. First, define the mean image

$$\mu = \frac{1}{K}\sum_{i=1}^{K} x_i. \tag{1.1}$$

and the covariance matrix

$$C = AA^T \in R^{N\times N} \tag{1.2}$$

where

$$A \equiv [y_1, \ldots, y_K] \tag{1.3}$$

$$y_i = x_i - \mu,\ i = 1, \ldots, K\ . \tag{1.4}$$

Low rank approximations to the data set A are obtained by computing a small number $r \ll K$ of the largest eigenvalues of C,

$$Cu_i = \lambda u_i, \quad i = 1, \ldots, r \tag{1.5}$$

and writing

$$C \approx U_r \Lambda_r U_r^T \tag{1.6}$$

where $U_r = [u_1, \ldots, u_r]$ and $\Lambda_r = diag\{\lambda_1, \ldots, \lambda_r\}$. Low rank apprximations to the images are obtained by projecting each vectorized image into the range space of U_r:

$$\tilde{x}_i = U_r^T (x_i - \mu), \tag{1.7}$$

$$x_i \approx \mu + U_r \tilde{x}_i. \tag{1.8}$$

An alternative way to obtain the orthogonal matrix U_r is by computing a truncated SVD of $A \in R^{N\times K}$, i.e.,

$$A \approx U_r \Sigma_r V_r^T, \tag{1.9}$$

where $\Sigma_r = diag\{\sigma_1, \ldots, \sigma_r\} \in R^{r\times r}$ is the diagonal matrix containing the r largest singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ of A, and $U_r \in R^{N\times r}$ and $V_r \in R^{K\times r}$ are matrices whose columns are the leading r left and right singular vectors of A, respectively. Regardless of how U_r is obtained, the dimension-reduced images are given by (1.8).

1.2 GLRAM While PCA requires the images M_k to be vectorized, GLRAM maintains the 2-dimensional nature of each image. It seeks a set of core matrices $\tilde{M}_i \in R^{r_1\times r_2}$, $i = 1, \ldots, K$, together with two projection matrices $L \in R^{I\times r_1}$ and $R \in R^{J\times r_2}$ having orthonormal columns, such that the following *optimal projection* condition is reached:

$$\min_{\substack{L\in R^{I\times r_1},\ L^TL=I \\ R\in R^{J\times r_2},\ R^TR=I \\ \tilde{M}_i\in R^{r_1\times r_2},\ i=1,\ldots,K}} \sum_{i=1}^{K} \|M_i - L\tilde{M}_i R^T\|_F^2. \tag{1.10}$$

Here $r_1 \ll I$ and $r_2 \ll J$ are prescribed numbers controlling the degree of compression of the data.

The solution to (1.10) is computed iteratively. First, it is observed [20] that minimizing (1.10) is equivalent to maximizing $\Phi = \sum_i \|L^T M_i R\|_F^2$. This removes the core matrices $\tilde{M}_i$ as unknowns. Then, an alternating procedure is invoked in which the measure Φ is minimized for L while R is fixed and then for R while L is fixed. Specifically, starting with an initial guess for R, the matrix

$$C_L \equiv \sum_{i=1}^{K} M_i R R^T M_i^T. \tag{1.11}$$

is formed and its eigenvectors associated with the r_1 largest eigenvalues are computed, i. e.,

$$C_L u_i^L = \lambda_i^L u_i^L, \quad i = 1, \ldots, r_1. \tag{1.12}$$

Then L is set as $L = [u_1^L, \ldots, u_{r_1}^L]$.

Interchanging the roles of R and L, one proceeds similarly to compute a new R by forming

$$C_R \equiv \sum_{i=1}^{K} M_i^T L L^T M_i. \tag{1.13}$$

The eigenvectors $u_i^R, i = 1, \ldots, r_2$, associated with the largest r_2 eigenvalues of C_R are computed and then R is set to $R = [u_1^R, \ldots, u_{r_2}^R]$. The process is iterated until L and R converge to a stable pair of matrices. The iteration does not necessarily converge to the optimal solution of (1.10).

Finally, the core matrices $\tilde{M}_i$ are obtained by

$$\tilde{M}_i = L^T M_i R, \quad i = 1, \ldots, K \tag{1.14}$$

and the low rank approximation to M_i is

$$M_i \approx L \tilde{M}_i R^T, \quad i = 1, \ldots, K. \tag{1.15}$$

The reader should compare (1.14) and (1.15) and note the analogy to (1.7) and (1.8) of the PCA method.

The parallel to PCA can be tightened by defining a mean image $\mu = \frac{1}{K} \sum_{i=1}^{K} M_i$ and then replacing (1.14) and (1.15) by

$$\tilde{M}_i = L^T (M_i - \mu) R \tag{1.16}$$

$$M_i \approx \mu + L \tilde{M}_i R^T \tag{1.17}$$

for $i = 1, \ldots, K$.

The 2D-PCA method proposed in [19] can be considered a special case of GLRAM in which only the right-side R projector is considered (in other words, L is set to the $I \times I$ identity and drops out of consideration).

2 Tensors

Next, we explain how the set of images M_k can be compresssed in a way analogous to PCA and GLRAM but by treating the data as a three-dimensional tensor. The approach is known as Higher Order Orthogonal Iteration (HOOI). It will be helpful to first review some basic properties of tensors.

A tensor is a generalization of a vector and a matrix. A vector is one-dimensional; a matrix is two dimensional; a tensor can have any number of dimensions. A three dimensional tensor is a box of numbers—a vector of matrices.

Tensors were originally introduced in physics to describe linear relations between two vectors or matices (e. g., moment of inertial tensor, stress and strain tensors). Tensor analysis became immensely popular after Einstein used tensors as the natural language to describe laws of physics in a way that does not depend on the inertial frame of reference. In the 1960's, Tucker [13] introduced tensors into psychometrics as a way to analyze multi-way data sets (e. g., scores versus students versus test conditions). *Circa* 1970, more or less contemporaneously, Harshmann proposed PARAFAC [5] and Carrol and Chang proposed CANDECOMP [2] (the two algorithms are essentially the same) as ways to perform factorial analysis on n-way data. In the early 1980's Kroonenberg [10] gave a range of practical examples of 'three-mode principal component analysis'. Recently, De Lathauwer *et al.* popularized HOSVD, a high-order generalization of the SVD algorithm. HOSVD has recently been applied to face recognition, image compression, and other applications as well. MATLAB toolboxes for tensors are available [7, 1].

2.1 Review of Tensors A T'th order tensor is denoted by $\mathcal{A} \in R^{I_1 \times I_2 \times \cdots \times I_T}$; a typical element is $\mathcal{A}_{i_1,i_2,\ldots,i_T}$ or $\mathcal{A}(i_1, i_2, \ldots, i_T)$, $1 \leq i_m \leq I_m$. An *inner product* can be defined on tensors:

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_T=1}^{I_T} \mathcal{A}_{i_1,i_2,\ldots,i_T} \mathcal{B}_{i_1,i_2,\ldots,i_T} \tag{2.18}$$

which induces the Frobenius norm $\|\mathcal{A}\| = \langle \mathcal{A}, \mathcal{A} \rangle^{1/2}$. Note that this inner product represents the common dot product of the two tensors viewed as two long vectors of length $I_1 I_2 ... I_T$ each.

The *mode-n product* of a $I_1 \times I_2 \times \cdots \times I_T$ tensor and a $J_n \times I_n$ matrix U is the tensor $\mathcal{B}_n = \mathcal{A} \times_n U$ whose elements are

$$\mathcal{B}_n (i_1, \ldots, i_{n-1}, i, i_{n+1}, \ldots, i_T) = \sum_{k=1}^{I_n} \mathcal{A} (i_1, i_2, i_{n-1}, k, i_{n+1}, \ldots, i_T) \; U (i, k) \tag{2.19}$$

Note that the n-th dimension of the tensor $\mathcal{B}$ is J_n, the row-dimension of U, while the other dimensions are the same as those of $\mathcal{A}$. For a 2-tensor $\mathcal{A}$ represented by matrix A, the mode-1 product $\mathcal{A} \times_1 U$ is simply the product UA, while the mode-2 product $\mathcal{A} \times_2 U$ corresponds to the product AU^T. For a 3-tensor $\mathcal{A}$ the mode-1 product $\mathcal{B}_1 = \mathcal{A} \times_1 U$ represents the tensor with entries

$$\mathcal{B}_1(i, i_2, i_3) = \sum_{k=1}^{I_1} U(i, k) \mathcal{A}(k, i_2, i_3)$$

So, using MATLAB notation, for each fixed i_2 the matrix $\mathcal{B}_1(:, i_2, :)$ is the result of the common matrix product $U\mathcal{A}(:, i_2, :)$ and, similarly, for each fixed i_3, the matrix $\mathcal{B}_1(:, :, i_3)$ is the result of the common

matrix product $U\mathcal{A}(:,:,i_3)$. Similar interpretations hold for products in the other 2 modes.

One can multiply by more than one mode, i. e.

$$\mathcal{A} \times_{n_1} U_{n_1} \times_{n_2} U_{n_2} \cdots \times_{n_p} U_{n_p}. \tag{2.20}$$

The order of the products is immaterial provided the modes are all distinct ($n_i \neq n_j, \forall\ i,j$). Using the same examples as before, if A is an array representing a second order tensor $\mathcal{A}$ then

$$\mathcal{A} \times_1 U \times_2 V = (\mathcal{A} \times_1 U) \times_2 V = UAV^T\ .$$

When the same mode is involved, one has the relation $\mathcal{A} \times_p U \times_p V = \mathcal{A} \times_p (VU)$. In particular, if $U^TU = I$, then $\mathcal{A} \times_p U \times_p U^T = \mathcal{A}$. Other properties of n-mode products are:

1. $\langle \mathcal{A} \times_p U, \mathcal{B} \rangle = \langle \mathcal{A}, \mathcal{B} \times_p U^T \rangle$.
2. If $U^TU = I$, $\|\mathcal{A} \times_p U\|_F^2 = \|\mathcal{A}\|_F^2$.
3. $\mathcal{A} \times_n I = \mathcal{A}$, where $I \in R^{I_n \times I_n}$ is the appropriately sized identity matrix.

The unfolding operation provides a bridge between the concept of tensors and the mathematical machinery of matrices. The *mode-n filaments* of $\mathcal{A}$ are the set of vectors $x \in R^{I_n}$ of the form

$$x = \mathcal{A}(i_1, \ldots, i_{n-1}, :, i_{n+1}, \ldots, i_T) \tag{2.21}$$

where, $1 \leq i_k \leq I_k$, $k \neq n$. (the colon ':' in dimension n is to be interpreted in the MATLAB sense).

The *mode-n unfolding* of $\mathcal{A}$, denoted by $\mathcal{A}_{(n)}$, is the $I_n \times S_n$ dimensional matrix ,where $S_n = \prod_{i \neq n} I_i$, whose columns are $\mathcal{A}$'s n-mode filaments ordered in some way. In other words, if

$$\pi : I_1 \times \cdots \times I_{n-1} \times I_{n+1} \times \cdots \times I_T \rightarrow S_n \tag{2.22}$$

is a 1-1 mapping from the set of indices of $\mathcal{A}$, except for the n'th index, to the integers $1, 2, \ldots, S_n$, then the pq'th element of $\mathcal{A}_{(n)}$ is

$$\left[\mathcal{A}_{(n)}\right]_{pq} = A(i_1, \ldots, i_{n-1}, p, i_{n+1}, \ldots, i_T) \tag{2.23}$$

where $\pi(i_1, \ldots, i_{n-1}, i_{n+1}, \ldots, i_T) = q$. Equation (2.23) can be interpreted as saying that index q selects the filament (through the mapping π), and index p selects the component of that filament. As indicated by the relation

$$U\mathcal{A}_{(n)} = \left[\mathcal{A} \times_n U\right]_{(n)}, \tag{2.24}$$

we see that $\mathcal{A} \times_n U$ can be thought of as replacing each mode-n filament x by Ux.

In what follows we will deal with matrices of the form $C = \mathcal{A}_{(n)}{\mathcal{A}_{(n)}}^T$, i. e., outer products of unfolded tensors. Such matrices can be formulated directly in terms of the original tensor $\mathcal{A}$ without the intermediary of an unfolding step. For example, if $\mathcal{A}$ is a third order tensor with dimensions $I_1 \times I_2 \times I_3$ and $C = \mathcal{A}_{(1)}{\mathcal{A}_{(1)}}^T$, then

$$C_{ij} = \sum_{p=1}^{I_2} \sum_{q=1}^{I_3} \mathcal{A}_{ipq}\mathcal{A}_{jpq}, \quad i,j \in \{1, 2, \ldots, I_1\} \tag{2.25}$$

$$= \langle \mathcal{A}(i,:,:), \mathcal{A}(j,:,:) \rangle\ . \tag{2.26}$$

In words, element C_{ij} is the inner product of the i^{th} and j^{th} slices of $\mathcal{A}$, those slices being taken perpendicularly to mode 1. Outer-products of unfoldings taken along other dimensions can be reformulated in a similar way.

2.2 Higher Order Orthogonal Iteration

Higher Order Orthogonal Iteration (HOOI) [3] is an iterative algorithm for computing low-rank approximations to tensors. Let $\mathcal{A}$ be an $I_1 \times I_2 \times \cdots \times I_T$ tensor and let $r_1, r_2, \ldots, r_T$ be a set of integers satisfying $1 \leq r_n \leq I_n$, for $n = 1, \cdots, T$. The *Rank-*$\{r_1, r_2, \ldots, r_T\}$ approximation problem is to find a set of $I_n \times r_n$ matrices $U^{(n)}$ with orthogonal columns, $n = 1, 2, \ldots, T$, and a $r_1 \times \cdots \times r_T$ core tensor $\mathcal{B}$ such than the optimization problem

$$\min_{\substack{U^{(1)}, U^{(2)} \ldots, U^{(T)} \\ \mathcal{B}}} \|\mathcal{A} - \mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \cdots \times_T U^{(T)}\|_F^2 \tag{2.27}$$

is satisfied. It can be shown [3] that the optimal $\mathcal{B}$ is given by

$$\mathcal{B} = \mathcal{A} \times_1 {U^{(1)}}^T \times_2 {U^{(2)}}^T \cdots \times_T {U^{(T)}}^T, \tag{2.28}$$

and that it is sufficient to find $U^{(n)}$'s satisfying $U^TU = I_{R_n}$ that maximize $\|\mathcal{B}\|_F^2$.

HOOI is an alternating least squares (ALS) approach to solving the *Rank-*$\{r_1, r_2, \ldots, r_T\}$ problem. It successively solves the restricted optimization problems

$$\min_{U^{(p)}} \|\mathcal{A} - \mathcal{B} \times_1 U_1 \times_2 U_2 \cdots \times_T U_T\|_F^2 \tag{2.29}$$

in which optimization is done over the p^{th} matrix $U^{(p)}$ while the latest available values of the other $U^{(i)}$'s, $i \neq p$, are used in (2.28) and (2.29). It is a Gauss-Seidel-like procedure that cycles dimension p through $1, 2, \ldots, T$ etc. and for each p treats only $U^{(p)}$ as the unknown.

For simplicity, the HOOI algorithm will be stated for 3^{rd} order tensors only. The extension to higher

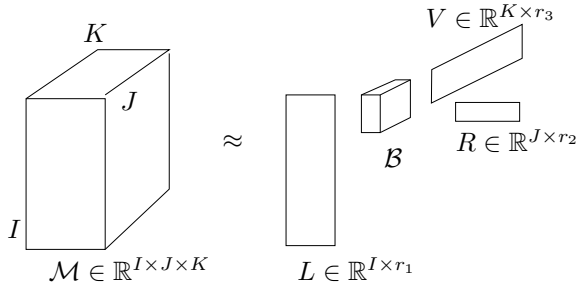

Figure 2: $\mathcal{M}$ approximated by $\mathcal{B} \times_1 L \times_2 R \times_3 V$. Here $\mathcal{B} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$ is called the *core* tensor and $L \in \mathbb{R}^{I \times r_1}$, $R \in \mathbb{R}^{J \times r_2}$, and $V \in \mathbb{R}^{K \times r_3}$ are *projection* matrices whose orthonormal columns approximately span the range spaces of the various dimensions of $\mathcal{M}$.

order tensors is straightforward. To conform to the notation used in subsequent sections, we will write R for $U^{(1)}$, L for $U^{(2)}$, and V for $U^{(3)}$.

ALGORITHM 2.1. Higher-Order Orthogonal Iteration (HOOI)

Input: $I \times J \times K$ tensor $\mathcal{A}$ and numbers r_1, r_2, r_3.
Output: $L \in \mathbb{R}^{I \times r_1}, R \in \mathbb{R}^{J \times r_2}$, $V \in \mathbb{R}^{K \times r_3}$, $\mathcal{B}$.
Choose initial R, V, with orthonormal columns.
Until Convergence Do:
$\mathcal{C} = \mathcal{A} \times_2 R^T \times_3 V^T$
$L = SVD(r_1, \mathcal{C}_{(1)})$
$\mathcal{D} = \mathcal{A} \times_1 L^T \times_3 V^T$
$R = SVD(r_2, \mathcal{D}_{(2)})$
$\mathcal{E} = \mathcal{A} \times_1 L^T \times_2 R^T$
$V = SVD(r_3, \mathcal{E}_{(3)})$
EndDo
$\mathcal{B} = \mathcal{E} \times_3 V^T$

Here $U = SVD(k, C)$ means compute the k'th-order truncated SVD of C and then set $U = [u_1, u_2, \ldots, u_k]$ to the matrix whose columns are the k largest left singular vectors u_i of C. Just as PCA can be based on either SVD or eigenvalue computations, the same is true of HOOI. In place of $U = SVD(k, C)$, one can obtain U as the matrix whose column vectors are the unit eigenvectors associated with the largest k eigenvalues of the matrix CC^T. We denote this alternative way of computing U by $U = EIG(k, CC^T)$.

If any of the reduced dimensions are maximum, e. g., if $r_1 = I$, say, or $r_2 = J$, then it is unnecessary to iterate on the first or second dimension, respectively. L would be the $I \times I$ identity matrix, or R the $J \times J$ identity. If $r_3 = K$, V would be the $K \times K$ identity. As will be seen, such end cases are important when trying to place other methods into the tensor framework.

3 Relation of PCA and HOOI

We next show that PCA can be regarded as a special case of HOOI.

THEOREM 3.1. *Let $\mathcal{M}$ be a 3^{rd} order tensor with dimensions $I \times J \times K$, and define individual matrices by $M_k = \mathcal{M}(:, :, k)$. Let $I_I \in R^{I \times I}$ and $I_J \in R^{J \times J}$ be identity matrices. Assume that the matrices are centered, i. e., that $\mu = \frac{1}{K} \sum_{p=1}^{K} M_p = 0$. Then the following are equivalent:*

1. *Using HOOI to compute a $Rank(I, J, r)$ approximation $\mathcal{B} \times_1 I_I \times_2 I_J \times_3 V_r = \mathcal{B} \times_3 V_r$ to $\mathcal{M}$, where $V_r \in R^{K \times r}$ is the projection matrix determined by HOOI, and*
2. *Using PCA to compute projection matrix $U_r \in R^{IJ \times r}$.*

In other words, PCA is a special case of HOOI in which maximum ranks ($r_1 = I, r_2 = J$) are used for the 1^{st} and 2^{nd} dimensions.

Proof. The PCA method rearranges each matrix M_k into a vector $x_k \in R^{IJ}$, forms the matrix $A = [x_1, \ldots, x_K]$, and sets $U_r = SVD(r, A)$. HOOI, on the other hand, forms the matrix $C = (\mathcal{M} \times_1 I_I \times_2 I_J)_{(3)} = \mathcal{M}_{(3)}$ and sets $V_r = SVD(r, \mathcal{M}_{(3)})$. Note that $A \in R^{IJ \times K}$ and $\mathcal{M}_{(3)} \in R^{K \times IJ}$. Because unfolding along the 3^{rd} dimension in the HOOI algorithm plays the same role as vectorization in PCA, we can, in fact, write

$$\mathcal{M}_{(3)} = A^T P, \tag{3.30}$$

where permutation matrix $P \in \mathbb{R}^{IJ \times IJ}$ accounts for the possibility that the mapping used for the unfolding operation may be different than the mapping that vectorizes M_k into x_k.

Now suppose the singular value decomposition of A is

$$A = \begin{bmatrix} U_r & U_c \end{bmatrix} \begin{bmatrix} \Sigma_{rr} & 0 \\ 0 & \Sigma_{cc} \end{bmatrix} \begin{bmatrix} V_r^T \\ V_c^T \end{bmatrix}. \tag{3.31}$$

PCA projects each vectorized matrix x_k using $\tilde{x}_k = U_r^T x_k$, i. e.,

$$\tilde{A} = [\tilde{x}_1, \ldots, \tilde{x}_k] = [U_r^T x_1, \ldots, U_r^T x_k] = U_r^T A = \Sigma_{rr} V_r^T. \tag{3.32}$$

It then approximately reconstructs each vector x_k from $x_k \approx U_r \tilde{x}_k$, i. e.,

$$A \approx A_{re} \equiv U_r \tilde{A} = U_r \Sigma_{rr} V_r^T. \tag{3.33}$$

A_{re} is the matrix whose columns are the (vectorized) reconstructed matrices.

HOOI, on the other hand, projects $\mathcal{M}$ into the core tensor $\mathcal{B} = \mathcal{M} \times_3 V_r^T$ and then approximately reconstructs $\mathcal{M}$ with $\mathcal{B} \times_3 V_r$. Now

$$\mathcal{B} = \mathcal{M} \times_3 V_r^T \Rightarrow \mathcal{B}_{(3)} = V_r^T \mathcal{M}_{(3)}, \tag{3.34}$$

that is,

$$\mathcal{B}_{(3)} = V_r^T A^T P = \Sigma_{rr} U_r^T P. \tag{3.35}$$

Accordingly, for the reconstructed tensor $\mathcal{M}^{re}$,

$$\mathcal{M} \approx \mathcal{M}^{re} \equiv \mathcal{B} \times_3 V_r \Rightarrow \mathcal{M}^{re}_{(3)} = V_r \mathcal{B}_{(3)}. \tag{3.36}$$

Using (3.35),

$$\mathcal{M}^{re}_{(3)} = V_r \Sigma_{rr} U_r^T P = A^T_{re} P. \tag{3.37}$$

In conclusion, just as $\mathcal{M}_{(3)} = A^T P$, we also have $\mathcal{M}^{re}_{(3)} = A^T_{re} P$. Although one method computes a matrix of left singular vectors U_r and the other computes a matrix of right singular vectors V_r, both methods lead to the same reconstructed matrices. ■

4 Relation of GLRAM and HOOI

In this section we elucidate the relation of GLRAM to HOOI applied to 3rd order tensors.

4.1 Model Problem Solved with Tensors It is easy to see how the model problem of K size $I \times J$ images M_k, $k = 1, \ldots, K$, can be compressed using tensors. According to the tensor point of view, we conceive of the images as arranged in a deck, like a stack of photo-prints. With the images assembled into a single, 3-rd order tensor $\mathcal{M}_{i,j,k}$, $1 \le i \le I$, $1 \le j \le J$, $1 \le k \le K$, the k^{th} sagittal plane $\mathcal{M}(:,:,k)$ becomes the image M_k.

To apply HOOI to $\mathcal{M}$, we take as given numbers r_1, r_2, and r_3, where $1 \le r_1 \le I$, $1 \le r_2 \le J$, and $1 \le r_3 \le K$. The HOOI algorithm yields three projection matrices $R \in R^{I \times r_1}$, $L \in R^{J \times r_2}$, and $V \in R^{K \times r_3}$, all of which have orthonormal columns, and a core tensor $\mathcal{B} \in R^{r_1 \times r_2 \times r_3}$ such that

$$\mathcal{M} \approx \mathcal{B} \times_1 L \times_2 R \times_3 V. \tag{4.38}$$

We can rewrite (4.38) in a form that is almost identical to (1.14) and (1.15) for GLRAM. To do this, define core matrices,

$$\tilde{M}_k \equiv \mathcal{B} \times_3 V(k,:),\ k = 1, 2, \ldots, K, \tag{4.39}$$

i. e. form K core $r_1 \times r_2$ matrices by taking the inner produce of each mode-3 filament of $\mathcal{B}$ with the k^{th} row of V. Then (4.38) can be written equivalently as

$$M_k = L \tilde{M}_k R^T,\ k = 1, 2, \ldots, K. \tag{4.40}$$

In other words, HOOI is a form of GLRAM in which the core matrices $\tilde{M}_k$ themselves are recontructed from $\mathcal{B}$ and the $V(k,:)$'s.

To see this in detail, note first that (4.39) means

$$\left(\tilde{M}_k\right)_{pq} = \sum_{r=1}^{r_3} \mathcal{B}_{pqr} V_{kr}. \tag{4.41}$$

Accordingly, (4.38) can successively be reworked as

$$\mathcal{M}_{ijk} = \sum_{p=1}^{r_1} \sum_{q=1}^{r_2} \sum_{r=1}^{r_3} \mathcal{B}_{pqr} L_{ip} R_{jq} V_{kr} \tag{4.42}$$

$$= \sum_{p=1}^{r_1} \sum_{q=1}^{r_2} L_{ip} \left(\sum_{r=1}^{r_3} \mathcal{B}_{pqr} V_{kr} \right) R_{jq} \tag{4.43}$$

$$= \sum_{p=1}^{r_1} \sum_{q=1}^{r_2} L_{ip} \left(\tilde{M}_k\right)_{pq} R_{jq} \tag{4.44}$$

$$= \left(L \tilde{M}_k R^T \right)_{ij}, \tag{4.45}$$

which is the component-wise statement of (4.40).

4.2 GLRAM as a Special Case of HOOI The relation between GLRAM and HOOI is further clarified by the following theorem.

THEOREM 4.1. *Let $\mathcal{M}$ be a 3^{rd} order tensor with dimensions $I \times J \times K$, and let I_K be the $K \times K$ identity matrix. Then the following are equivalent:*

1. *Using HOOI to compute a $Rank(r_1, r_2, K)$ approximation $\mathcal{B} \times_1 L \times_2 R \times_3 I_K$ to $\mathcal{M}$, and*
2. *Using GLRAM to compute projection matrices $L \in R^{I \times r_1}$, $R \in R^{J \times r_2}$ and core matrices $\tilde{M}_k = \mathcal{B}(:,:,k)$.*

In other words, GLRAM is a special case of HOOI in which the maximum dimension is used for r_3 (i. e., $r_3 = K$).

Proof. The proof consists in working out the formulas for HOOI when maximum rank, i. e. $r_3 = K$, is specified for the third dimension, and showing that the resulting formulas coincide with GLRAM. An immediate consequence of maximum rank on the 3^{rd} dimension is that we can choose any orthognal $K \times K$ matrix as the projector for the third dimension; for simplicity, we choose the identity I_K.

With $U^{(1)} = L$, $U^{(2)} = R$, and $U^{(3)} = I_K$, consider first the calculation of L. Using the eigenvalue formulation of HOOI, we have for the L update:

$$C = \left(\mathcal{M} \times_2 R^T \times_3 I_K^T\right)_{(1)} \tag{4.46}$$

$$L = EIG(r_1, CC^T). \tag{4.47}$$

Let $\mathcal{C} = \mathcal{M} \times_2 R^T \times_3 I_K^T$. Then, from the definition of mode-n products,

$$\mathcal{C}_{ipq} = \sum_{s=1}^{r_2} \sum_{t=1}^{K} \mathcal{M}_{ist} (R^T)_{ps} \delta_{tq} \tag{4.48}$$

$$= \sum_{s=1}^{r_2} \mathcal{M}_{isq} R_{sp} = (M_q R)_{ip}\,, \tag{4.49}$$

where δ_{tq} is the Kronkecker delta function and where we have written $M_q = \mathcal{M}(:,:,q)$ for the matrices used in GLRAM.

Since C is the mode-1 unfolding of $\mathcal{C}$,

$$(CC^T)_{ij} = \sum_{p=1}^{J} \sum_{q=1}^{K} \mathcal{C}_{ipq} \mathcal{C}_{jpq} \tag{4.50}$$

$$= \sum_{p=1}^{J} \sum_{q=1}^{K} (M_q R)_{ip} (M_q R)_{jp} \tag{4.51}$$

$$= \sum_{q=1}^{K} (M_q R R^T M_q^T)_{ij}. \tag{4.52}$$

In HOOI, the columns of L are set to the leading eigenvectors of CC^T; in GLRAM, they are set to the leading eigenvectors of $\sum_{q=1}^{K} M_q R R^T M_q^T$. But we have just shown that these matrices are the same.

In a similar way, HOOI updates R from

$$D = \left(\mathcal{M} \times_1 L^T \times_3 I_K^T\right)_{(2)} \tag{4.53}$$

$$R = EIG(r_2, DD^T). \tag{4.54}$$

By interchanging the roles of the first and second modes, a similar argument shows that

$$DD^T = \sum_{q=1}^{K} M_q^T L L^T M_q; \tag{4.55}$$

the R computed by HOOI will therefore be the same as the R computed by GLRAM. ■

Our earlier remark that 2DPCA amounts to GLRAM with the left projection matrix L set to the identity I_I establishes the following immediate collorary of the above theorem.

COROLLARY 4.1. *The 2DPCA algorithm can be considered to be a $Rank(I, r_2, K)$ HOOI.*

4.3 GLRAM Plus SVD The identification of the GLRAM algorithm as a special case of HOOI leads to some practical consequences. For example, just as a HOOI implementation can be based on either eigenvalue or SVD computations, so can GLRAM. In [20] only the eigenvalue approach is presented. The following algorithm indicates how GLRAM can also be carried out using SVD decompositions.

ALGORITHM 4.1. SVD-based GLRAM
Input: M_k, $k = 1, 2, \ldots, K$; intergers r_1, r_2.
Output: $L \in R^{I \times r_1}$, $R \in R^{J \times r_2}$.
Choose initial R with orthonormal columns.
Until Converged Do
  Form tensor $\mathcal{C}$ defined by $\mathcal{C}(:,:,q) = M_q R$
  $L = SVD(r_1, \mathcal{C}_{(1)})$
  Form tensor $\mathcal{D}$ defined by $\mathcal{D}(:,:,q) = L^T M_q$
  $R = SVD(r_2, \mathcal{D}_{(2)})$
EndDo

In terms of tensor notation, $\mathcal{C} = \mathcal{M} \times_2 R^T$ and $\mathcal{D} = \mathcal{M} \times_1 L^T$.

In [20], Ye *et al.* consider in some detail a composite algorithm GLRAM+SVD consisting of the following steps:

1. Apply GLRAM to generate feature matrices $\tilde{M}_k = L^T M_k R$.
2. Vectorize the feature matrices: $\tilde{x}_k = vec(\tilde{M}_k),\ k = 1, 2, \ldots, K$.
3. Compute an r^{th} order truncated SVD of $\tilde{A} = [\tilde{x}_1, \ldots, \tilde{x}_K]$, i. e. $\tilde{A} \approx U_r \Sigma_r V_r$.
4. Approximate $\tilde{x}_k$ by $U_r \hat{x}_k$, where $\hat{x}_k = U_r^T \tilde{x}_k$, $k = 1, 2, \ldots, K$.

Not surprisingly, GLRAM+SVD is intimately related to HOOI, as is made clear by the following theorem.

THEOREM 4.2. *Let $\mathcal{M}$ be a 3^{rd} order tensor and define matrices $M_k = \mathcal{M}(:,:,k)$. Assume that the matrices M_k are centered, i. e., $\mu = \frac{1}{K} \sum_{k=1}^{K} M_k = 0$. Then the following are equivalent:*

1. *Use HOOI to compute a $Rank(r_1, r_2, r_3)$ approximation $\mathcal{B} \times_1 L \times_2 R \times_3 V$ to $\mathcal{M}$, provided one uses a restricted iteration scheme that only iterates between L and R until convergence, and then calculates V without further iteration.*
2. *First use GLRAM to compute projection matrices $L \in R^{I \times r_1}$, $R \in R^{J \times r_2}$ and feature matrices $\tilde{M}_k = L^T M_i R$; second, follow GLAM with an r_3^{th} order PCA applied to the vectorized $\tilde{M}_k$.*

In other words, GLRAM+SVD is HOOI with a restricted iteration that does not go back to update L and R after computing V.

Proof. This theorem follows directly from theorems (4.1) and (3.1). By theorem (4.1), the HOOI iteration on the first and second dimensions leads to the same projection matices L and R as GLRAM. Further,

by the definition of mode-n products, one has the correspondences

$$\tilde{M}_i = L^T M_i R \;\leftrightarrow\; \mathcal{B} = \mathcal{M} \times_1 L^T \times_2 R^T \tag{4.56}$$

$$M_i^{re} = L\tilde{M}_i R^T \;\leftrightarrow\; \mathcal{M}^{re} = \mathcal{B} \times_1 L \times_2 R \tag{4.57}$$

where $\tilde{M}_i = \mathcal{B}(:,:,i)$ and $M_i^{re} = \mathcal{M}^{re}(:,:,i)$.

Note that the projected matrices $\tilde{M}_i$ will also be centered, since

$$\tilde{\mu} = \frac{1}{K}\sum_{k=1}^{K} \tilde{M}_k = L^T \left(\frac{1}{K}\sum_{k=1}^{K} M_k \right) R = L^T \mu R \tag{4.58}$$

and $\mu = 0$ by assumption.

The final step in the HOOI algorithm calculates matrix $V \in \mathbb{R}^{K\times r}$ and forms a new reconstruction tensor

$$\left(\mathcal{M}^{re} \times_3 V^T\right) \times_3 V. \tag{4.59}$$

By theorem (3.1), this process is equivalent to PCA applied to $A^{re} = [x_1^{re}, \ldots, x_K^{re}]$, each x_i^{re} being the vectorized M_i^{re} from the GLRAM step. ■

Because the standard HOOI goes back and adjusts L and R after V has been calculated, one expects that the resulting reconstruction error will be smaller than with GLRAM-SVD. The amount of improvement in reconstruction error may not be much in practice, however, as we shall see.

5 Application to Dimension Reduction

In light of the relationships uncovered in this paper, we might consider PCA, 2DPCA, GLRAM, and HOOI as belonging to the same family or class of techniques, which, for want of a better name, we might call *Alternating Least Squares Projection (ALSP) Methods*. The relation between these various methods is summarized in Figure 3.

The papers in the bibliography evidence the effectiveness of ALSP techniques applied to dimension reduction and recognition problems. We focus here on dimension reduction. HOOI is useful for dimension reduction because the memory required to store projection matrices L, R, and V and the core matrix $\mathcal{B}$, i. e.,

$$Ir_1 + Jr_2 + Kr_3 + r_1r_2r_3$$

is often significantly less than the storage IJK required for the original $I \times J \times K$ tensor. For a given choice for r_1, r_2, r_3, PCA, 2DPCA, and GLRAM achieve less compression of the core tensor (compared to HOOI), but benefit from not having to store three projection matrices. The storage for PCA, for example, is [1]

$$Kr_3 + IJr_3$$

and that for GLRAM is

$$Ir_1 + Jr_2 + r_1r_2K.$$

The storage for GLRAM+SVD will be the same as HOOI. If centering is desired, an addition storage of IJ is required for any of the ALSP methods.

Figure 4: Three representative face images from the Yale Face Database. The first column shows the original images as they appear in the Yale Database. The remaining columns, from left to right, are reconstructed images using HOOI, GLRAM+SVD, GLRAM, and GLRAM + CENTERING, respectively. For the reconstructed images, $r_1 = 20$ and $r_2 = 20$. HOOI and GLRAM+SVD used $r_3 = 20$ as well. GLRAM and its centered variant do not project along the 3^{rd} dimension, and for these, in effect, $r_3 = 165$.

5.1 Example Let us briefly consider an example of dimension reduction and reconstruction of images using HOOI and GLRAM.

Figure 4 shows a selection of three reconstructed face images obtained by applying HOOI, GLRAM+SVD, GLRAM, and GLRAM+CENTERING to the entire set of images in the Yale Face Database. This publically available database consists of 165 GIF images of 15 subjects, each in 11 poses [2]. In this example, the original

[1] It is interesting to note that for the standard version of PCA as outlined in section 1.1, projection matrix U_{r_3} uses storage IJr_3 and the 'feature' vectors $\tilde{x}_j, j = 1, 2, \ldots, K$ take up storage Kr_3. For the HOOI version of PCA, the situation is reversed: the projection matrix V requires Kr_3 for storage and the core tensor requires IJr_3.

[2] The 11 poses are: centerlight, glassees, happy, leftlight, noglasses, normal, rightlight, sad, sleepy, surprised, and wink.

(5.60)

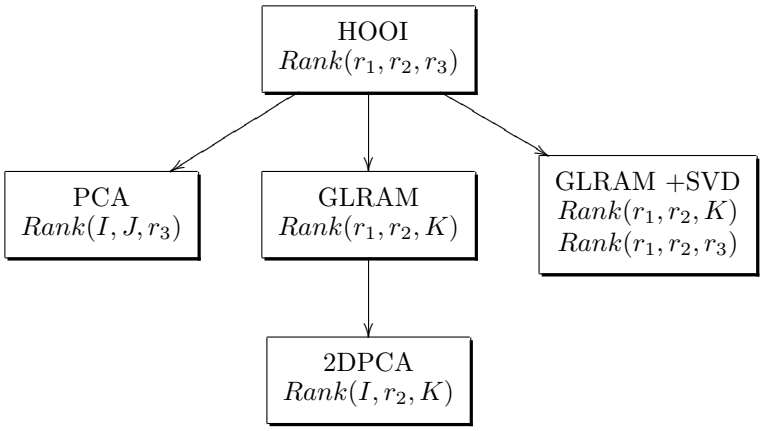

Figure 3: Relation of ALSP methods for reducing a $I \times J \times K$ tensor. HOOI reduces all dimensions. PCA reduces only the 3^{rd} dimension. GLRAM reduces the 1^{st} and 2^{nd} dimensions; as a special case, 2DPCA reduces only the 2^{nd}. GLRAM+SVD reduces first on the 1^{st} and 2^{nd} dimensions, then reduces on the 3^{rd}.

image tensor $\mathcal{M}$ had dimensions $I = 112, J = 92$, and $K = 165$. All the reconstructed images used $r_1 = 20$ and $r_2 = 20$ for the projectors L and R. For the projector along the third dimension (V or U), HOOI and GLRAM+SVD used $r_3 = 20$. GLRAM and GLRAM+CENTERING, of course, used the trivial full-dimension projector $r_3 = 165$. By GLRAM+CENTERING we mean subtracting away and later adding back the average image μ as in equation (1.16). Table 1 gives some statistics for this example. In the table, *Root-Mean-Squared-Error* or RMSE is defined by

$$RMSE = \sqrt{\sum_{k=1}^{K} \|M_k - \tilde{M}_k\|_F^2}. \tag{5.61}$$

As expected from our theory, HOOI and GLRAM+SVD yield images of essentially the same quality. The quality of the GLRAM images (and GLRAM+CENTER) is much better of course, but this is due to the much lower compression ratios of these latter methods (24 and 21, respectively, compared to 111). Referring to the RMSE row in table 1, we see that the reconstruction error is *slightly* lower for HOOI compared to GLRAM+SVD. This is because HOOI goes back and computes new values for L and R based on the V computed from the previous iteration; GLRAM+SVD does not. The reduction in error from this is negligibly small, however, and it is done at a cost: HOOI has significantly greater run time than GLRAM+SVD. (The question of run-time efficiency bears closer scrutiny, but is beyond the scope of this paper.) Looking at the last two columns of Table 1, we see that centering also improves the RMSE, but again the amount of improvement is small, and the need to store the average image μ brings down the compression ratio from 24 for GLRAM to 21 for GLRAM+μ. Judging from the example with the YALE Faces Database, it seems that centering is not a particularly useful operation for image reconstruction.

Given the range of choices available for the selection of r_1, r_2, and r_3, one may ask whether it is possible for HOOI to outperform GLRAM in terms of cost for the same resulting accuracy. Though the issue of cost is beyond the scope of this paper, we consider the more limited question of whether, for a given accuracy, HOOI may lead to better compression than GLRAM. To investigate this question, we apply HOOI to the Yale Faces Database using a wide variety of choices for r_1, r_2, and r_3, taking care that the compression ratio remains fixed at 24, which is the compression attained by GLRAM in Table 1; for this level of compression, GLRAM has a RMSE of 0.09689. What sort of accuracies can HOOI achieve at this compression ratio?

The results are given in Table 2. Surprisingly, HOOI yields only a slight improvement in accuracy when r_1, r_2, r_3 are favorably chosen (the decrease in RMSE is only about 1.9% when $r_1, r_2, r_3 = 24, 24, 89$, the best choice of the cases we tried). Part of the reason for this lackluster improvement is that HOOI, *vis-a-vis* GLRAM, has the additional overhead of storing a projection matrix for the third dimension, and the storage cost of V not negligible since K and r_3 are both relatively large.

Specializations of HOOI which only project along a subset of a tensor's dimensions have the advantage

	HOOI	GLRAM+SVD	GLRAM	GLRAM+μ
$r1, r2, r3$	20x20x20	20x20x20	20x20x165	20x20x165
run time(s)	2.93	1.66	1.17	1.18
compression ratio	111	111	24	21
RMSE	0.17324	0.17348	0.09689	0.09636

Table 1: Statistics for Yale Faces Database. $I \times J \times K = 112 \times 92 \times 165$. HOOI used 2 iterations, and the projection matrices R and V were initialized to matrices with random element taken from a uniform distribution.

of needing to store fewer projectors. In our example, the advantage of economization of projectors largely offsets the cost of GLRAM's larger core matrices. In other examples the tradeoff may tip differently to one or the other method.

6 Conclusion

This paper provides a taxonomy of Alternating Least-Squares Projection methods in which a variety of recently published techniques—2D-PCA, GLRAM, CLRAM+SVD, G2DPCA, Coupled Subspace Analysis (CSA)—are all shown to be special cases of HOOI. The contribution of this paper, then, is not the discovery of a new method; rather, it consists in the systemization of an outcropping of new methods, showing that what was thought to be a multitude is really an organized unity, this unity being achieved through the concept of tensors.

Certain connections between the algorithms discussed in this paper have been pointed out by others. The authors of GLRAM, G2DPCA, and CSA (essentially the same algorithm) were certainly aware of 2D-PCA, and it is likely that they were lead to the discovery of their own algorithms by attempting to generalize 2D-PCA; hence, the connection of GLRAM (G2DPCA, CSA) to 2D-PCA is not new. Likewise, the authors in [18] explicitly discuss the relation of CSA to PCA. Yet none of these authors draws a link between their algorithms and the tensor-reduction algorithm HOOI.

The elucidation of these links bears some practical fruit. For example, the relation of GLRAM to HOOI has suggested a way to do GLRAM using SVD rather than eigenvalue computations. While we have shown GLRAM+SVD to be equivalent to HOOI with a restricted iteration scheme, the drop in RMSE by going back and updating L and R after V is computed, in fact, was found to be negligible, at least for our example. Moreover, GLRAM+SVD seemed to be significantly more efficient than HOOI, and it is phrased entirely in the familiar terms of matrices, a pedagogical advantage. Tensor proponents should not, therefore, scoff at GLRAM+SVD: it can be thought of as a judicous specialization of HOOI that is more efficient and more easily understood. Nevertheless, without climbing the hill to the tensor vantage point, we suspect other 'new' algorithms will be discovered that turn out to be specializations of HOOI in disguise.

References

[1] Claus A. Andersson and Rasmus Bro. The n-way toolbox for MATLAB. *Chemometrics and Intelligent Laboratory Systems*, 54:1–4, 2000.

[2] J. D. Carroll and J. J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of 'Eckart-Young' decomposition. *Psychometrika*, 1970.

[3] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 2000.

[4] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. On the best rank-1 and rank-$(r_1, r_2, \ldots, r_n)$ approximation of higher-order tensors. *SIAM J. Matrix Anal. Appl.*, 2000.

[5] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis. In *UCLA Working Papers In Phonics*, volume 16, pages 1–84. UCLA, 1970.

[6] I. T. Jolliffe. *Principal Component Analysis*. Springer Verlag, New York, 1986.

[7] Tamara G. Kolda and Brett W. Bader. Algorithm xxx: MATLAB tensor classes for fast algorithm prototyping. *ACM Transactions on Mathematical Software*. To appear. See also http://csmr.ca.sandia.gov/ tgkolda/TensorToolbox.

[8] Tamara G. Kolda, Brett W. Bader, and Joseph P. Kenny. Higher-order web link analysis using multilinear algebra. In *Data Mining, Fifth IEEE International Conference*. IEEE, Nov 2005.

[9] Hui Kong, Lei Wang, Eam Teoh, Li Xuchun, Jian-Gang Wang, and Ronda Venkateswarlu. Generalized 2d principal component analysis for face image representation and recognition. *Neural Networks*, 2005.

[10] Pieter M. Kroonenberg. *Three-Mode Principal Component Analysis*. DSWO Press, 1983. See http://three-mode.leidenuniv.nl.

method	r_1, r_2, r_3	compression ratio	RMSE	iters
HOOI	20x20x118	24	0.099702	2
HOOI	21x21x110	24	0.097204	2
HOOI	22x22x102	24	0.095800	2
HOOI	23x23x 95	24	0.095105	2
HOOI	24x24x 89	24	0.095097	2
HOOI	25x25x 83	24	0.095872	2
HOOI	27x27x 73	24	0.098825	2
HOOI	30x30x 61	24	0.105281	2
HOOI	35x35x 46	24	0.119002	2
HOOI	40x40x 35	24	0.134444	2
HOOI	23x23x 95	24	0.095094	4
GLRAM	20x20x165	24	0.096895	N/A

Table 2: RMSE of HOOI applied to Yale Faces Database, when compression ratio is fixed at 24. Next to last row shows effect of performing additional iterations. Last row give comparison data for GLRAM.

[11] Nima Mesgarani and S. A. Slaney, M.and Shamma. Speech discrimination based on multiscale spectro-temporal features. In *IEEE International Conference on Acoustic, Speech and Signal Processing*, 2004.

[12] Jian-Tao Sun, Hua-Jun Zeng, Huan Liu, Yuchang Lu, and Zheng Chen. CubeSVD: a novel approach to personalized web search. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 382–390, New York, NY, USA, 2005. ACM Press.

[13] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 1966.

[14] M. Alex O. Vasilescu and Demetri Terzopoulos. Multilinear analysis of images ensembles: Tensorfaces. In *Proc. 7th European Conference on Computer Vision (ECCV'02)*, 2002.

[15] M. Alex O. Vasilescu and Demetri Terzopoulos. Tensortextures: Multilinear image-based rendering. *ACM Transactions on Graphics*, 2004.

[16] Hongcheng Wang and Narendra Ahuja. Facial expression decomposition. In *Proc. of the Ninth IEEE Int. Conf. on Computer Vision (ICCV 2003)*, 2003.

[17] Hongcheng Wang, Qing Wu, Lin Shi, Yizhou Yu, and Narendra Ahuja. Out-of-core tensor approximation of multi-dimensional matrices of visual data. *ACM Trans. Graph.*, 24(3):527–535, 2005.

[18] Dong Xu, Shuicheng Yan, Lei Zhang, ZhengKai Liu, and Hongjiang Zhang. Coupled subspaces analysis. Technical report, Microsoft Research Asia, 2004.

[19] Jian Yang, David Zhang, Alehandro Frangi, and Jing-yu Yuang. Two-dimensional pca: A new approach to appearance-based face representation and recognition. *IEEE Trans. on Pattern Analysis and Machine Learning*, 26(1):131–137, Jan 2004.

[20] Jieping Ye. Generalized low rank approximations of matrices. *Machine Learning*, 2005.