Tensor Decompositions Workshop Discussion Notes*
American Institute of Mathematics (AIM)
Palo Alto, CA

Carla D. Moravitz Martin †

July 19-23, 2004

*This work was supported by AIM.

†Center for Applied Mathematics, Cornell University, 657 Rhodes Hall, Ithaca, NY 14853-7510, carlam@cam.cornell.edu

Contents

1 Introduction

A workshop on Tensor Decompositions was held July 19-23, 2004 at the American Institute of Mathematics Research Conference Center in Palo Alto, CA. The workshop was organized by Gene Golub, Tammy Kolda, James Nagy, and Charles Van Loan. The workshop brought together researchers specializing in tensor decompositions as well as specialists in scientific computing, linear algebra, and applications. The purpose of the workshop was to collaborate in order to develop new theoretical and computational tools necessary to tackle larger problems and new applications of tensor decompositions. The workshop consisted of both pre-arranged and impromptu talks as well as break-out sessions and group discussions. The workshop also included a banquet and a hike in nearby Sam McDonald County Park.

Throughout the writeup there are many references to the talks given by participants. In most cases, the talks can be found online at `http://csmr.sandia.ca.gov/~tgkolda/tdw2004/`.

2 Background

Matrix decompositions such as the singular value decomposition (SVD) are ubiquitous in numerical analysis.[1] One way to think of the SVD is that it decomposes a matrix into a sum of rank-1 matrices. In other words, an $I \times J$ matrix $\mathbf{A}$ is expressed as a minimal sum of rank-1 matrices:

$$\mathbf{A} = (\mathbf{u}_1 \circ \mathbf{v}_1) + (\mathbf{u}_2 \circ \mathbf{v}_2) + \cdots + (\mathbf{u}_r \circ \mathbf{v}_r),$$

where, $\mathbf{u}_i \in \mathbb{R}^I$ and $\mathbf{v}_i \in \mathbb{R}^J$ for all $i = 1, 2, \ldots, r$. The operator "$\circ$" denotes the outer product; thus the ij-th entry of the rank-1 matrix $\mathbf{a} \circ \mathbf{b}$ is the product of the i-th entry of $\mathbf{a}$ with the j-th entry of $\mathbf{b}$, that is, $(\mathbf{a} \circ \mathbf{b})_{ij} = a_i b_j$. Such decompositions underlie fundamental concepts such as matrix rank and approximation theory and impact a range of applications including searching the world wide web, signal processing, medical imaging, and principal component analysis. The decompositions are well-understood mathematically, numerically, and computationally.

If we have data in three or more dimensions, then we are dealing with a higher-order tensor. Higher-order tensor (also known as multidimensional, multiway, or n-way array) decompositions are used in many applications and also have theoretical interest. A matrix is a tensor of order two. Extending matrix decompositions such as the SVD to higher-order tensors has proven to be quite difficult. Familiar matrix concepts such as rank become ambiguous and more complicated. One goal of a tensor decomposition is the same as for a matrix decomposition: to rewrite the tensor as a sum of rank-1 tensors. Consider, for example, an $I \times J \times K$ tensor $\mathcal{A}$. We would like to express $\mathcal{A}$ as the sum of rank-1 third-order tensors; i.e.,

$$\mathcal{A} = (\mathbf{u}_1 \circ \mathbf{v}_1 \circ \mathbf{w}_1) + (\mathbf{u}_2 \circ \mathbf{v}_2 \circ \mathbf{w}_2) + \cdots + (\mathbf{u}_r \circ \mathbf{v}_r \circ \mathbf{w}_r),$$

where, $\mathbf{u}_i \in \mathbb{R}^I$, $\mathbf{v}_i \in \mathbb{R}^J$, and $\mathbf{w}_i \in \mathbb{R}^K$ for $i = 1, 2, \ldots, r$. Note that if $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are vectors, then $(\mathbf{a} \circ \mathbf{b} \circ \mathbf{c})_{ijk} = a_i b_j c_k$.

[1] Wording in this section was taken from the workshop proposal written by the workshop organizers.

3 Notation

Although there was much discussion regarding notation, a consensus emerged towards the end of the workshop. [2]

Indices are denoted by lowercase letters and span the range from 1 to the uppercase letter of the index, e.g., $n = 1, 2, \ldots, N$. Vectors are denoted by lowercase boldface letters, e.g., $\mathbf{u}$. Matrices by uppercase boldface letters, e.g., $\mathbf{U}$. Tensors by calligraphic letters, e.g., $\mathcal{A}$.

If $\mathcal{A}$ is an $I_1 \times I_2 \times \cdots \times I_N$ tensor, then the *order* of $\mathcal{A}$ is N. The n-th *mode*, *way*, or *dimension* of $\mathcal{A}$ is of size I_n.

3.1 Vectorizing

In some expressions that follow, we make use of representing matrices and tensors as vectors. We define this notation here.

Let $\mathbf{B} \in \mathbb{R}^{I \times J}$. Then $\texttt{vec}(\mathbf{B})$ is defined as

$$\texttt{vec}(\mathbf{B}) = \begin{bmatrix} \mathbf{B}(:,1) \\ \vdots \\ \mathbf{B}(:,J) \end{bmatrix} \in \mathbb{R}^{IJ}$$

Let $\mathbf{b} \in \mathbb{R}^{IJ}$. Then $\texttt{reshape}(\mathbf{b}, I, J)$ is defined as

$$\texttt{reshape}(\mathbf{b}, I, J) = \left[\begin{array}{c|c|c|c} \mathbf{b}(1:I) & \mathbf{b}(I+1:2I) & \ldots & \mathbf{b}((J-1)I+1:IJ) \end{array} \right] \in \mathbb{R}^{I \times J}.$$

In other words, $\texttt{reshape}(\mathbf{b}, I, J)$ creates an $I \times J$ matrix from $\mathbf{b}$. The operators `vec` and `reshape` are related. Note that $\mathbf{b} = \texttt{vec}(\texttt{reshape}(\mathbf{b}, I, J))$ and also that

$$\texttt{vec}(\mathbf{B}) = \texttt{reshape}(\mathbf{B}, IJ, 1).$$

Now we use the `vec` operators on tensors. Let $\mathcal{B} \in \mathbb{R}^{I \times J \times K}$. Then $\texttt{vec}(\mathcal{B})$ is defined as

$$\texttt{vec}(\mathcal{B}) = \begin{bmatrix} \texttt{vec}(\mathcal{B}(:,:,1)) \\ \vdots \\ \texttt{vec}(\mathcal{B}(:,:,K)) \end{bmatrix} \in \mathbb{R}^{IJK}.$$

3.2 Matricizing

It is convenient to be able to represent tensors as matrices. Typically, all the columns along a certain mode are rearranged to form a matrix. Turning a tensor into a matrix in this way is called *flattening* or *matricizing*. There are multiple ways to order the columns, but we use the notation presented in

[2] Wording and notation in this section is from the presentation and report by participants, T. Kolda and B. Bader.

L. De Lathauwer, B. De Moor, and J. Vandewalle, A multilinear singular value decomposition. *SIAM Journal of Matrix Analysis and Applications*, 21 (2000) 1253-1278.

For a third-order tensor, there are three possible flattening matrices. The $2 \times 2 \times 2$ case presented in figure 1 makes this clear.

3.3 n-mode Multiplication

To multiply a tensor times a matrix, we need to specify which mode of the tensor is multiplied by the columns of the matrix. If $\mathcal{A}$ is an $I_1 \times I_2 \times \cdots \times I_N$ tensor and $\mathbf{U}$ is $J_n \times I_n$, then the n-mode product is of size $I_1 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N$ and is denoted by

$$\mathcal{A} \times_n \mathbf{U}.$$

The formula for n-mode multiplication can be found in the paper by De Lathauwer, De Moor, and Vandewalle referenced above. The n-mode product exactly relates to multiplying $\mathbf{U}$ by the appropriate flattening of $\mathcal{A}$. See table 1 for an example of n-mode multiplication when $\mathcal{A}$ is order three.

4 Tensor Decompositions

There are two types of decompositions used most in applications. The first decomposition was independently proposed in the following two papers:

J.D. Carroll and J. Chang, Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35 (1970) 283-319.

R.A. Harshman, Foundations of the PARAFAC procedure: Model and conditions for an 'explanatory' multi-mode factor analysis. *UCLA Working Papers in phonetics*, 16 (1970) 1-84.

Carroll and Chang named the model the CANDECOMP (CANonical DECOMPosition) and Harshman named the model PARAFAC (PARAlell FACtors). Thus, the model is currently known as the CANDECOMP-PARAFAC (CP) model.

The second decomposition is called the TUCKER model and was proposed in

L.R. Tucker, Some mathematical notes of three-mode factor analysis. *Psychometrika*, 31 (1966) 279-311.

The CP and TUCKER models extend to arbitrary ordered tensors. For clarity, we present them for tensors of order three.

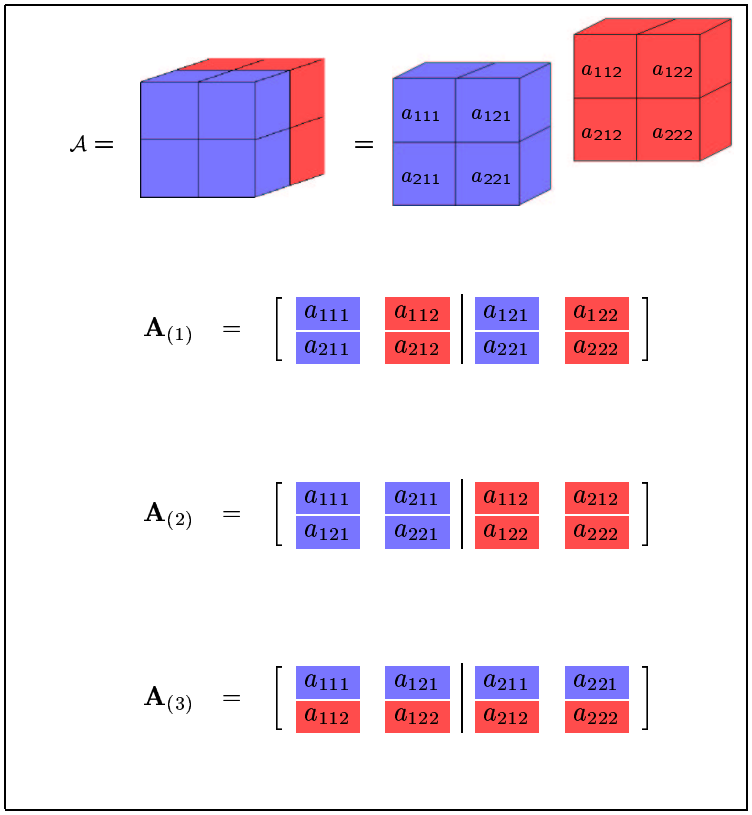


Figure 1: Illustration of matricizing a third-order tensor

	n-mode product	equivalence with flattenings	vectorized version
$\mathbf{B}_1 \in \mathbb{R}^{L \times I}$	$\mathcal{A} \times_1 \mathbf{B}_1$ $(L \times J \times K)$	$\mathbf{B}_1 \cdot \mathbf{A}_{(1)}$	$(\mathbf{I} \otimes \mathbf{B}_1) \cdot \mathtt{vec}(\mathbf{A}_{(1)})$
$\mathbf{B}_2 \in \mathbb{R}^{L \times J}$	$\mathcal{A} \times_2 \mathbf{B}_2$ $(I \times L \times K)$	$\mathbf{B}_2 \cdot \mathbf{A}_{(2)}$	$(\mathbf{I} \otimes \mathbf{B}_2) \cdot \mathtt{vec}(\mathbf{A}_{(2)})$
$\mathbf{B}_3 \in \mathbb{R}^{L \times K}$	$\mathcal{A} \times_3 \mathbf{B}_3$ $(I \times J \times L)$	$\mathbf{B}_3 \cdot \mathbf{A}_{(3)}$	$(\mathbf{I} \otimes \mathbf{B}_3) \cdot \mathtt{vec}(\mathbf{A}_{(3)})$

Table 1: n-mode Multiplication in Terms of Flattening Matrices on an $I \times J \times K$ tensor, $\mathcal{A}$

4.1 CANDECOMP-PARAFAC Decomposition

Given an $I \times J \times K$ tensor, $\mathcal{A}$, the CP model is a decomposition of the form

$$\mathcal{A} = \sum_{i=1}^{R} \mathbf{u}_i \circ \mathbf{v}_i \circ \mathbf{w}_i$$

where $\mathbf{u}_i \in \mathbb{R}^I$, $\mathbf{v}_i \in \mathbb{R}^J$, and $\mathbf{w}_i \in \mathbb{R}^K$ for $i = 1, \ldots, R$. For the time being, the vectors are assumed to be real. However, the model is also valid for for complex-valued vectors. Note that there are no constraints (such as orthogonality) on the vectors $\mathbf{u}_i, \mathbf{v}_i, \mathbf{w}_i$. However, one can impose constraints such as orthogonality, nonnegativity, or unimodality when needed.[3]

Note that a CP decomposition always exists (take R to be the product of the sizes of each mode and take outer products of scaled standard basis vectors). Ideally, R is chosen to be the minimal number of terms needed to sum to $\mathcal{A}$. When R is minimal, then R is known as the *tensor rank* and is discussed in §6.

4.2 TUCKER Decomposition

Given an $I \times J \times K$ tensor, $\mathcal{A}$, the TUCKER model is a decomposition of the form

$$\mathcal{A} = \sum_{i=1}^{R_1} \sum_{j=1}^{R_2} \sum_{k=1}^{R_3} \sigma_{ijk} (\mathbf{u}_i \circ \mathbf{v}_j \circ \mathbf{w}_k)$$

where $R_1 \leq I, R_2 \leq J, R_3 \leq K$, $\mathbf{u}_i \in \mathbb{R}^{R_1}$, $\mathbf{v}_j \in \mathbb{R}^{R_2}$, and $\mathbf{w}_k \in \mathbb{R}^{R_3}$ for all i, j, k. The tensor, $\mathcal{S} = (\sigma_{ijk})$, is called the ***core tensor***. Note that the core tensor does not always need to have the same dimensions as $\mathcal{A}$. The CP decomposition is a special case of the TUCKER decomposition.

There are no constraints on the vectors $\mathbf{u}_i, \mathbf{v}_j, \mathbf{w}_k$ in the TUCKER decomposition. However, one may impose constraints when needed. If the $\mathbf{u}_i, \mathbf{v}_j, \mathbf{w}_k$ are columns from orthogonal matrices $\mathbf{U}, \mathbf{V}, \mathbf{W}$, then the TUCKER model is referred to as the Higher-Order Singular Value Decomposition, or HOSVD. The HOSVD can also be written in terms of n-mode products:

$$\mathcal{A} = \mathcal{S} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}$$

where $\mathcal{A}, \mathcal{S} \in \mathbb{R}^{I \times J \times K}$, $U \in \mathbb{R}^{I \times I}$, $V \in \mathbb{R}^{J \times J}$, and $W \in \mathbb{R}^{K \times K}$. The HOSVD always exists (it can be computed by taking the SVD of each of the flattening matrices). However, the HOSVD does not reveal the tensor rank unless the core is diagonal (see next section).

4.3 Diagonalization

We say that a tensor is *diagonalizeable* if the HOSVD yields a diagonal core tensor (i.e., $\sigma_{ijk} = 0$ unless $i = j = k$). Note that if the core is diagonal, we can write the HOSVD as a PARAFAC

[3] See talk by R. Bro on the conference website.

with orthogonality contraints. In general a tensor cannot be diagonalized. It is also not clear under what conditions permit a diagonalizeable core.

5 MATLAB

We can store tensors in MATLAB as multiway arrays. For example, a random $4 \times 5 \times 6$ array can be initialized by

```
>> A=rand(4,5,6);
```

We can extract slices using the colon operator. Therefore the second slice of the tensor `A` can be referenced using `A(:,:,2)`. Flattening matrices can be computed by permuting the indices. If $\mathcal{A}$ is $2 \times 2 \times 2$, then the flattening matrices in figure 1 can be computed by

```
>> A1=reshape(permute(A,[1 3 2]),2,4);
>> A2=reshape(permute(A,[2 1 3]),2,4);
>> A3=reshape(permute(A,[3 2 1]),2,4);
```

We mention that there are other ways to flatten a tensor in MATLAB. Note than n-mode multiplication can then be performed using table 1.

For the workshop, Tammy Kolda and Brett Bader created three MATLAB classes for manipulating tensors.[4] The classes allow for n-mode multiplication, flattening, and representing tensors in terms of the CP and TUCKER models that were described in §4. The documentation is given in

> B.W. Bader and T.G. Kolda, A Preliminary Report on the development of MATLAB Tensor Classes for Fast Algorithm Prototyping, *Technical Report SAND2004-3487*, Sandia National Laboratories, Livermore, CA, July 2004.

It is important to note that the MATLAB classes do not contain algorithms, but rather should be used as a tool to quickly develop algorithms. Andersson and Bro have created an N-way MATLAB toolbox to compute the CP and TUCKER decompositions.[5]

6 Tensor Rank

The first group discussion dealt with important questions regarding tensor rank. Recall that when R is minimal and a tensor $\mathcal{A}$ can be written as

$$\mathcal{A} = \sum_{i=1}^{R} \mathbf{u}_i \circ \mathbf{v}_i \circ \mathbf{w}_i$$

then the ***tensor rank*** is R (we continue to use third-order examples for clarity).

The discussion centered around the importance of rank, uniqueness, and orthogonal decompositions.

[4] The associated `m`-files can be found at `http://csmr.sandia.gov/~tgkolda/`.

[5] The toolbox is available for download at `http://www.models.kvl.dk/source/nwaytoolbox/`.

6.1 Uniqueness

The advantage of the CP representation is its uniqueness under certain conditions. Specifically, for a matrix $\mathbf{U}$, let $K_{\mathbf{U}}$ be the maximum number of randomly chosen columns having full rank. In other words, every set of $K_{\mathbf{U}}$ columns is linearly independent. For example, if $\mathbf{U}$ is a matrix with two identical columns, then $K_{\mathbf{U}} = 2$. Given the CP decomposition,

$$\mathcal{A} = \sum_{i=1}^{R} \mathbf{u}_i \circ \mathbf{v}_i \circ \mathbf{w}_i,$$

then the decomposition is unique if

$$K_{\mathbf{U}} + K_{\mathbf{V}} + K_{\mathbf{W}} \geq 2R + 2.$$

This fact was discussed in the talk by R. Bro and proved in

> J.B. Kruskal. Rank, Decomposition, and Uniqueness for 3-Way and N-Way Arrays. In R. Coppi and S. Bolasco (editors), *Multiway Data Analysis*, 7-18. Amsterdam: Elsevier, 1989.

6.2 Why is rank and uniqueness of solutions important?

In some applications (e.g., chemometrics and psychometrics), the decomposition of a tensor into rank-1 tensors has empirical meaning. For example, in chemometrics uniqueness is required when analyzing fluorescence data to exactly determine the underlying factors. This is because the spectral decompositions reflect the true spectra of what is measured. The rank in this case is assumed to be known *a priori*.

6.3 Can we rely on orthogonal techniques for stability?

The answer to this question depends on the application. Some participants argued that uniqueness of solutions is not meaningful if the stability properties of the algorithm are unknown. On the other hand, in some applications (e.g., chemometrics), uniqueness is necessary and orthogonal decompositions do not have a good interpretation. The disadvantage of nonorthogonal decompositions is that sometimes they lead to degenerate solutions.[6]

6.4 Computation

There is no known method to compute tensor rank in general (see §15.14). A paper on the computation of tensor rank brought to the attention of participants by L.-H. Lim is

> J. Håstad, "Tensor rank is NP-complete," *J. Algorithms*, **11** (4), December 1990, pp. 644–654.

[6] See talk by R. Harshman on the conference website.

7 Small Examples

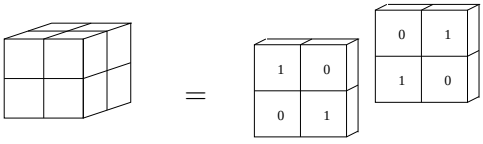
In numerous discussions, participants cited various examples of tensors that had interesting properties. These examples were usually discussed within the context of comparing matrix rank to tensor rank and the complexity of understanding tensor rank. It was suggested to compile a list of these small examples, some of which are given below.

7.1 Maximum Rank of a $2 \times 2 \times 2$ tensor

The maximum rank of a tensor depends on the field of scalars. The following illustrative examples are taken from

> J.B. Kruskal. Rank, Decomposition, and Uniqueness for 3-Way and N-Way Arrays. In R. Coppi and S. Bolasco (editors), *Multiway Data Analysis*, 7-18. Amsterdam: Elsevier, 1989.

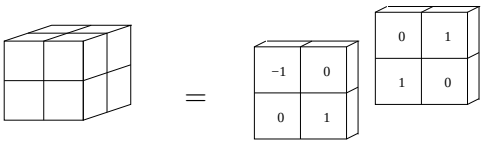
Example 7.1.1 Let the field of scalars be $\mathbb{R}$, and suppose that

$\mathcal{A} =$ 

Then $\text{rank}(\mathcal{A}) = 2$:

$$\mathcal{A} = \frac{1}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 1 \\ -1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ -1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

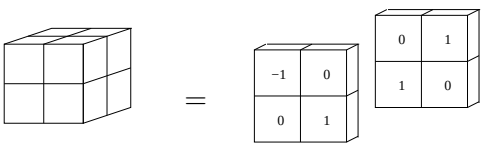
Example 7.1.2 Let the field of scalars be $\mathbb{R}$, and suppose that

$\mathcal{A} =$ 

Then $\text{rank}(\mathcal{A}) = 3$:

$$\mathcal{A} = \frac{1}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 1 \\ -1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ -1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ -1 \end{bmatrix} - 2\begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Example 7.1.3 Let the field of scalars be $\mathbb{C}$, and suppose that

$\mathcal{A} =$ 

Then $\text{rank}(\mathcal{A}) = 2$:

$$\mathcal{A} = \frac{1}{2}\begin{bmatrix} -i \\ 1 \end{bmatrix} \circ \begin{bmatrix} -i \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ i \end{bmatrix} + \frac{1}{2}\begin{bmatrix} i \\ 1 \end{bmatrix} \circ \begin{bmatrix} i \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ -i \end{bmatrix}$$

Examples 7.1.2 and 7.1.3 illustrate that the rank depends on the field of scalars. In general, the maximum rank of a $2 \times 2 \times 2$ tensor over $\mathbb{R}$ is 3 but the maximum rank over $\mathbb{C}$ is 2.

Kruskal performed a Monte-Carlo simulation which showed that the set of rank-2, $2 \times 2 \times 2$ real tensors fill about 79% of the 8-dimensional space, whereas rank-3 tensors fill about 21% of the space. The surprising result is that rank-deficient tensors have positive volume, as opposed to the matrix case.

During the discussion of this surprising rank result, C. Van Loan presented a follow up for the group to verify. In particular, assume that $\mathcal{A}$ is $2 \times 2 \times 2$ and $\text{rank}(\mathcal{A}) = 2$. Then

$$\mathcal{A} = \mathbf{u}_1 \circ \mathbf{v}_1 \circ \mathbf{w}_1 + \mathbf{u}_2 \circ \mathbf{v}_2 \circ \mathbf{w}_2$$

This implies that

$$\begin{aligned} \mathbf{A}(:,:,1) &= w_{11}\mathbf{u}_1\mathbf{v}_1^T + w_{12}\mathbf{u}_2\mathbf{v}_2^T \\ &= \mathbf{U}\begin{bmatrix} w_{11} & 0 \\ 0 & w_{12} \end{bmatrix}\mathbf{V}^T \\ \mathbf{A}(:,:,2) &= w_{21}\mathbf{u}_1\mathbf{v}_1^T + w_{22}\mathbf{u}_2\mathbf{v}_2^T \\ &= \mathbf{U}\begin{bmatrix} w_{21} & 0 \\ 0 & w_{22} \end{bmatrix}\mathbf{V}^T, \end{aligned}$$

which means that two matrices can be simultaneously diagonalized 79% of the time. The group offered a partial explanation by working backwards. Let $\mathbf{A}_1 = \mathbf{A}(:,:,1)$ and $\mathbf{A}_2 = \mathbf{A}(:,:,2)$ and form the matrices $\mathbf{A}_1\mathbf{A}_2^{-1}$ and $\mathbf{A}_2^{-1}\mathbf{A}_1$. We know that a tensor will be rank-2 if we can diagonalizing these matrices to get

$$\begin{aligned} \mathbf{A}_1\mathbf{A}_2^{-1} &= \mathbf{X}\begin{bmatrix} \frac{w_{11}}{w_{21}} & 0 \\ 0 & \frac{w_{12}}{w_{22}} \end{bmatrix}\mathbf{X}^{-1} \\ \mathbf{A}_2^{-1}\mathbf{A}_1 &= \mathbf{Y}\begin{bmatrix} \frac{w_{11}}{w_{21}} & 0 \\ 0 & \frac{w_{12}}{w_{22}} \end{bmatrix}\mathbf{Y}^{-1} \end{aligned} \tag{1}$$

R. Bro ran a MATLAB simulation that showed that 21% of the time (1) resulted in getting complex eigenvalues. Note that if the eigenvalues are complex, then the steps cannot be reversed.

7.2 Known Maximal Ranks

If the field of scalars is $\mathbb{R}$, then

- The maximum rank of a $2 \times 2 \times 2$ tensor is 3.
- The maximum rank of a $3 \times 3 \times 3$ tensor is 5.
- The maximum rank of an $8 \times 8 \times 8$ tensor is 11.

The first two items above are shown in

J.B. Kruskal. Rank, Decomposition, and Uniqueness for 3-Way and N-Way Arrays. In R. Coppi and S. Bolasco (editors), *Multiway Data Analysis*, 7-18. Amsterdam: Elsevier, 1989.

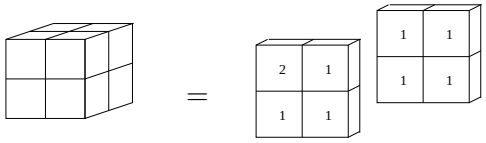
The third item was brought up in group discussion by R. Harshman.

7.3 Other Tensor Rank Properties

- The minimal tensor decomposition is not always orthogonal. That is, if R is minimal in

$$\mathcal{A} = \sum_{i=1}^{R} \mathbf{u}_i \circ \mathbf{v}_i \circ \mathbf{w}_i$$

then the $\{\mathbf{u}_i\}, \{\mathbf{v}_i\}, \{\mathbf{w}_i\}$ do not necessarily form sets of orthonormal vectors. Let

$\mathcal{A} =$ 

Then $\text{rank}(\mathcal{A}) = 2$:

$$\mathcal{A} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

However, it is impossible to write $\mathcal{A}$ as the sum of two orthogonal tensors (i.e., $u_1 \perp u_2$, $v_1 \perp v_2$, and $w_1 \perp w_2$). The proof of this can be found in

> J.B. Denis and T. Dhorne. Orthogonal tensor decomposition of 3-way tables. In R .Coppi and S. Bolasco (editors), *Multiway Data Analysis*, 31-37. Amsterdam: Elsevier, 1989.

- Uniqueness of the CP solution depends on factors other than just the order and dimension (see §6.1). If a $6 \times 6 \times 6$ tensor has a CP decomposition with 8 components, then the solution is unique.[7]

- A $7 \times 7 \times 7 \times 7$ supersymmetric tensor (see §9) will yield a unique solution if it is decomposed into 30 terms, but other fourth-order tensors of lower dimensions will not. This is explained in

> P. Comon. Tensor Decompositions. In J.G. McWhirter and I.K. Proudler (editors), *Mathematics in Signal Processing V*, 1-24. Clarendon Press, Oxford, 2002.

[7] See talk by R. Bro on the conference website.

8 Extending the Matrix SVD

Another discussion group centered around the "ideal" extension of the matrix SVD. Participants listed important properties of the SVD with the idea of trying to extend those to higher-order tensors. The following properties of the matrix SVD were identified as starting points in this regard.

1. The SVD has a *geometric* interpretation, i.e., the singular values of a matrix are the lengths of the semi-axes of the hyperellipse defined by its image.
2. The SVD gives us information about optimal low-rank approximations since the singular values tell us how close the matrix is to one of lower rank.
3. The SVD always exists for any $m \times n$ matrix and the singular values are uniquely determined.
4. Successive best rank-1 approximations give the best rank-k approximation. If the rank of a matrix is R, then R successive rank-1 approximations yield the SVD.
5. The computation is stable with respect to perturbations.
6. The SVD is readily computable and algorithms have been analyzed for efficiency and stability.
7. Methods to compute the SVD do not involve alternating least squares (ALS) methods.
8. All compact operators have a singular value expansion (SVE). There is a nice connection between the SVE of the continuous operator and the SVD of the discretized operator.
9. There is a nice relationship between the SVD of a matrix, A, and the eigendecompositions of A^TA and AA^T (namely the singular values of A are the square roots of the eigenvalues of A^TA and AA^T).
10. The SVD has a proven usefulness and strong pedigree.

9 Supersymmetric Tensors: Importance and Properties

A *supersymmetric* tensor is a tensor whose entries are invariant under any permutation of the indices. For example, a third-order supersymmetric tensor has

$$a_{ijk} = a_{ikj} = a_{jik} = a_{jki} = a_{kij} = a_{kji}.$$

9.1 Applications

Supersymmetric tensors arise naturally in higher-order statistics and blind source separation. In addition, Pierre Comon has shown a nice relationship between supersymmetric tensors and polynomials. A tensor element, a_{ijk}, can be associated with a monomial, $a_{ijk}x_ix_jx_k$. The space of supersymmetric tensors can then be associated with the space of homogeneous polynomials. See

P. Comon. Tensor Decompositions. In J.G. McWhirter and I.K. Proudler (editors), *Mathematics in Signal Processing V*, 1-24. Clarendon Press, Oxford, 2002.

9.2 Open Questions related to Supersymmetric Tensors

9.2.1 Can we find the nearest supersymmetric approximation to a tensor?

9.2.2 Can we decompose a tensor into a sum of a supersymmetric tensor and a nonsupersymmetric tensor?

We can write a matrix $\mathbf{A}$ as a sum of a symmetric matrix and a skew-symmetric matrix:

$$\mathbf{A} = \frac{\mathbf{A} + \mathbf{A}^T}{2} + \frac{\mathbf{A} - \mathbf{A}^T}{2}.$$

Is there a tensor equivalent?

9.2.3 Can we embed a non-supersymmetric tensor into a supersymmetric tensor? Does this simplify computation?

The idea of embedding a tensor into a supersymmetric tensor is an extension of the matrix case. Indeed, if $\mathbf{A}$ is a matrix, then

$$\mathbf{B} = \begin{bmatrix} \mathbf{0} & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{bmatrix}$$

is symmetric. Since there is a relationship between the singular values of $\mathbf{A}$ and the eigenvalues of $\mathbf{B}$, we can use eigenvalue algorithms for symmetric matrices to determine the singular values of $\mathbf{A}$.

Let $\mathcal{A}$ be $2 \times 2 \times 2$. It turns out there are multiple ways to embed $\mathcal{A}$ into a supersymmetric tensor. One way results in a $2 \times 2 \times 2$ tensor embedded into a supersymmetric $6 \times 6 \times 6$ tensor.

Most importantly, we should determine what properties of the supersymmetric tensor relate back to the original embedded tensor. In addition, we need to determine when it is "worth the effort", computationally, to write tensors in terms of symmetric tensors. Is the gain in symmetry more important than the computational aspects that go along with the increase in size?

10 Operations with the Khatri-Rao-Bro product and its relationship to the CP Model

Let matrices $\mathbf{A}$ and $\mathbf{B}$ be represented in terms of their columns:

$$\mathbf{A} = \left[\begin{array}{c|c|c|c} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_n \end{array} \right] \qquad \text{and} \qquad \mathbf{B} = \left[\begin{array}{c|c|c|c} \mathbf{b}_1 & \mathbf{b}_2 & \dots & \mathbf{b}_n \end{array} \right]$$

Then the *Khatri-Rao-Bro product*, $\odot$, of $\mathbf{A}$ and $\mathbf{B}$ is defined as

$$\mathbf{A} \odot \mathbf{B} = \left[\begin{array}{c|c|c|c} \mathbf{a}_1 \otimes \mathbf{b}_1 & \mathbf{a}_2 \otimes \mathbf{b}_2 & \ldots & \mathbf{a}_n \otimes \mathbf{b}_n \end{array} \right]$$

where "$\otimes$" denotes the *Kronecker product*. Recall that the Kronecker product of two vectors $\mathbf{c} \in \mathbb{R}^\ell$, $\mathbf{d} \in \mathbb{R}^m$ is

$$\mathbf{c} \otimes \mathbf{d} = \begin{bmatrix} c_1\mathbf{d} \\ c_2\mathbf{d} \\ \vdots \\ c_\ell\mathbf{d} \end{bmatrix} \in \mathbb{R}^{\ell m}$$

Note that

$$\mathbf{A} \otimes \mathbf{B} = \left[\begin{array}{c|c|c|c|c|c|c|c} \mathbf{a}_1 \otimes \mathbf{b}_1 & \mathbf{a}_1 \otimes \mathbf{b}_2 & \ldots & \mathbf{a}_1 \otimes \mathbf{b}_n & \ldots & \mathbf{a}_n \otimes \mathbf{b}_1 & \mathbf{a}_n \otimes \mathbf{b}_2 & \ldots & \mathbf{a}_n \otimes \mathbf{b}_n \end{array} \right]$$

We can relate the Khatri-Rao-Bro product and Kronecker product using MATLAB notation. If $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$, then

$$\mathbf{A} \odot \mathbf{B} = \mathbf{C}(\ :\ , 1 : n+1 : n^2).$$

The Kronecker product relates directly to the TUCKER model since it involves all combinations of the columns of $\mathbf{A}$ and $\mathbf{B}$. However, the Khatri-Rao-Bro product relates directly to the CP model and is used to speed up the algorithm.[8] Recall the three-way CP model:

$$\mathcal{A} = \sum_{i=1}^{R} \mathbf{u}_i \circ \mathbf{v}_i \circ \mathbf{w}_i.$$

Rewriting this in terms of flattening matrices, gives

$$\mathbf{A}_{(1)} = \mathbf{U}(\mathbf{V} \odot \mathbf{W})^T,$$

whereas the Kronecker product version is

$$\mathbf{A}_{(1)} = \mathbf{U}\tilde{\mathbf{I}}(\mathbf{V} \otimes \mathbf{W})^T$$

where $\tilde{\mathbf{I}}$ is an appropriate flattening of the tensor identity (ones on the super-diagonal and zeros elsewhere). For example, if $\mathcal{A}$ is $3 \times 3 \times 3$ and $\mathbf{U}, \mathbf{V}, \mathbf{W}$ are 3×3 matrices, then $\mathbf{A}_{(1)}$ is 3×9 and

$$\tilde{\mathbf{I}} = \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right].$$

[8] See talk by R. Bro on the conference website.

11 Zero-ing out elements in the core array

Using the HOSVD, we can introduce zeros into the core array of the TUCKER decomposition. In particular, the orthogonal matrices can be strategically chosen to zero out certain elements. For an order-p $N \times N \times \cdots \times N$ array, we can introduce $\frac{pN(N-1)}{2}$ zeros since each of the p rotations have $\frac{N(N-1)}{2}$ degrees of freedom. There are multiple places to introduce zeros. One idea was shown in the talk by Vince Fernando. Zeros were introduced along each edge of each subcube. Another idea is to introduce zeros everywhere but the super-diagonal (when possible) or to create a diagonally positioned matrix in the middle. See figure 2.

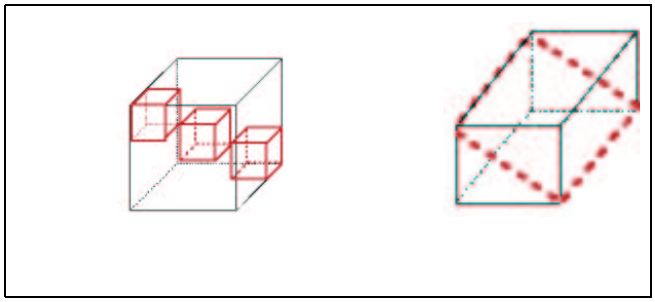


Figure 2: Diagonal third-order tensor (left); and diagonally positioned matrix in a third-order tensor (right)

In the higher-order Hessenburg, $\frac{p(N-2)(N-1)}{2}$ zeros are introduced (L. De Lathauwer).

12 Efficiency in Computation of the TUCKER decomposition

In certain applications, we may not want to compute the entire decomposition. For example, in the matrix case we can compute only a few singular values or vectors without computing the entire decomposition. Here are a few open questions that involve something less than a full decomposition.

12.1 Open Questions related to computation of TUCKER

12.1.1 How can we compute only a few specified core elements without computing others?

12.1.2 Can we compute the TUCKER decomposition with some sparsity or other structure, such as "upper triangular"?

12.1.3 Do iterative methods exist for tensors? That is can we devise a subspace method that is an extension of Lanzcos iteration?

12.1.4 Do we ever have deflation in the TUCKER computation?

12.1.5 We may have different decompositions based on different topologies. Thus far, we have been thinking of N-way arrays in terms of hypercubes. There may be other decompositions based on other types of representations. What are they?

12.1.6 Do structured tensors arise in applications such as N-way Toeplitz arrays? Are there displacement based fast algorithms?

13 Alternate decompositions based on matrices rather than just vectors

13.1 Open Questions related to alternate decompositions

This section contains questions regarding decompositions in other forms rather than the traditional outer product decomposition.

13.1.1 Does representing a tensor $\mathcal{A}$ in terms of blocks lead to another type of decomposition?

If $\mathbf{A}$ is a block matrix we can compute its *Kronecker product SVD* (KPSVD) to represent $\mathbf{A}$ as a Kronecker product of *matrices* rather than an outer product of vectors. For example, let $\mathbf{A}$ be a 2×2 block matrix with $m \times n$ blocks:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}.$$

Also suppose

$$\tilde{\mathbf{A}} = \begin{bmatrix} \mathtt{vec}(\mathbf{A}_{11})^T \\ \mathtt{vec}(\mathbf{A}_{21})^T \\ \mathtt{vec}(\mathbf{A}_{12})^T \\ \mathtt{vec}(\mathbf{A}_{22})^T \end{bmatrix},$$

where $\mathtt{vec}(\mathbf{A}_{ij}) = \mathtt{reshape}(\mathbf{A}_{ij}, mn, 1)$.

If the SVD of $\tilde{\mathbf{A}}$ is given by $\tilde{\mathbf{A}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ then

$$\mathbf{A} = \sum_{i=1}^{\tilde{R}} \sigma_i(\mathbf{U}_i \otimes \mathbf{V}_i),$$

where $\texttt{vec}(\mathbf{U}_i), \texttt{vec}(\mathbf{V}_i)$ are the i-th columns of $\mathbf{U}$ and $\mathbf{V}$, respectively and $\tilde{R} = \text{rank}(\tilde{\mathbf{A}})$.

Beginning to extending this idea to tensors, we can *matricize* a tensor $\mathcal{A}$ in terms of recursive blocking. Indeed, one way to represent a $n_1 \times n_2 \times n_3 \times n_4$ tensor is to write it as a $n_1 \times n_2$ block matrix with block size $n_3 \times n_4$. For example, if $\mathcal{A}$ is $3 \times 4 \times 5 \times 6$, then we can write $\mathcal{A}$ as the 3×4 block matrix,

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{A}_{13} & \mathbf{A}_{14} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{A}_{23} & \mathbf{A}_{24} \\ \mathbf{A}_{31} & \mathbf{A}_{32} & \mathbf{A}_{33} & \mathbf{A}_{34} \end{bmatrix},$$

where each $\mathbf{A}_{ij}$ is 5×6.

As the order increases we get block matrices whose entries are also block matrices. This can be thought of as a *recursive* block representation. The recursive blocking idea is not restricted to p being even. If $\mathcal{A}$ is $3 \times 4 \times 5 \times 6 \times 7$ then we can write $\mathcal{A}$ as a 3×4 block matrix where each block, $\mathcal{A}_{ij}$ is a $5 \times 6 \times 7$ tensor, which could be matricized if desireable.

In addition, a sixth-order tensor can be represented as a third-order tensor. If $\mathcal{A}$ is $n_1 \times n_2 \times n_3 \times n_4 \times n_5 \times n_6$ then we can represent $\mathcal{A}$ as a $n_1 \times n_2 \times n_3$ block tensor where each block is size $n_4 \times n_5 \times n_6$.

13.1.2 Can we decompose a tensor into second-order or higher objects?

Related to the KPSVD idea for matrices, it may be possible to decompose a tensor into a second-order to higher objects rather than just outer products of vectors. If so, what would be the algorithm to compute such a decomposition?

14 Extending Linear Algebra Concepts and Algorithms

This section contains open problems that stemmed from participant discussions and individual questions. Some were discussed in small groups and others were proposed to the entire group. Attempts at answering or references cited are stated where applicable.

14.1 Open Questions related to extending linear algebra theory

The open questions in this section relate to linear algebra theory.

14.1.1 The SVD has a nice geometric interpretation in terms of the singular values. What would be the geometric interpretation for tensor decompositions?

14.1.2 What exactly are the objects that are operated on by tensors? What are the singular values or eigenvalues? Could they be vectors rather than scalars?

14.1.3 A typical linear algebra problem is to solve $Ax = b$. Are there analogous problems for tensors? Where do they come from? How are they solved?

- There is a fast Poisson Solver that is written as a tensor product of matrices (G. Golub). See the following paper:

  > R.E. Lynch, J.R. Rice, D.H. Thomas, Tensor product analysis of partial differential equations, *Bull. Amer. Math. Soc.*, 70 (1964), pp. 378-384.

  In addition one may want to see how FFTs can be used to solve this problem in (G. Golub):

  > R. Bellman, *Introduction to Matrix Analysis (2nd ed.)*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1997.

- In chemistry, multiway regression and rank-reduced regression is used (R. Bro)

- The below paper was cited as another example:

  > V. Pereyra and G. Scherer, Efficient Computer Manipulation of Tensor Products with Applications to Multidimensional Approximation, *Mathematics of Computation*, 27(123):595-605, July 1973

- The first reference on tensor GMRES is

  > R.B. Schnabel and P.D. Frank, Tensor methods for nonlinear equations. *SIAM J. Num. Anal.*, 21 (1984), pp. 815843.

  L.-H. Lim also cited the following paper on tensor GMRES

  > D. Feng and T.H. Pulliam, "Tensor-GMRES method for large systems of nonlinear equations," *SIAM J. Optim.*, **7** (3), August 1997, pp. 757–779.

14.1.4 Is there an analogue to a symmetric positive definite (SPD) matrix for tensors? Does this say anything about the HOSVD?

Yes. Every tensor can be thought of as a multilinear map (L.-H. Lim). The group defined the SPD equivalent for (say) fourth-order tensors as

$$\mathcal{A} \times_1 \mathbf{y} \times_2 \mathbf{y} \times_3 \mathbf{y} \times_4 \mathbf{y} \geq 0$$

for all $\mathbf{y}$ of appropriate dimension. L.-H. Lim points out that the above definition only works for tensors of even order (replace $\mathbf{y}$ by $-\mathbf{y}$ and that forces $\mathcal{A}$ to be the zero tensor in odd-ordered tensors).

With the recursive flattening approach, we could get blocks of SPD matrices. Is there an SPD diagonal dominance connection?

14.1.5 Are there analogies to other decompositions, such as LU and QR? What about other rank revealing decompositions such as CUR (see §15.11)

- There is an algorithm for structured tensors which is an extension of the Schur algorithm (P.Comon). The reference is:

  > V. S. Grigorascu and P. A. Regalia, "Tensor displacement structures and polyspectral matching", *Fast Reliable Algorithms for Structured Matrices*, T. Kailath and A. H. Sayed, eds., SIAM Publications, Philadelphia, PA, 1999

- In general, LU and QR are not possible if they are direct extensions of the matrix case. This can be seen by counting degrees of freedom: $n^3 - n^2$ for LU and $n^3 - \frac{n(n-1)}{2}$ for QR (L.-H. Lim). However, there may be other ways of thinking about LU.
- If the tensor is *supersymmetric* then LU corresponds to Groebner bases and could be related to different types of rank. The LU algorithm is an elimination-type procedure (P. Comon)
- A third-order "Toeplitz" tensor would have equal components for values of $(i-j)$,$(j-k)$ or some other permutation of indices
- Generalization of Vandermonde and Hankel would work similarly. Generalizations of these appear in the reference by Grigorascu and Regalia cited above.

14.1.6 Is there an analog to Jordan Canonical Form for tensors?

No (C. Moler). This is transforming from same space to same space which is not happening here.

14.2 Open Questions related to extending linear algebra algorithms

The open questions in this section are algorithmic-based.

14.2.1 Given an arbitrary tensor $\mathcal{A}$, is there an operation that reduces the tensor rank by one?

14.2.2 How does principal component analysis (PCA) generalize to N-way?

14.2.3 It seems that by matricizing tensors we lose information since we are lessening the dimension. What information are we really losing?

14.2.4 For matrices, there exists a notion of Krylov subspace. What's the analogy for tensors? Do such subspaces arise in tensor SVD methods?

There exists a GMRES-type method for tensors (see question 14.1.3). There is also PLS related to Lanczos, but the connection to the multilinear case is unclear (R. Bro).

14.2.5 How do you perform tensor computations without flattening? For example, can you directly map a three-way array into a three dimensional parallel computing topology? If so, what does tensor multiplication look like now?

It is probably possible to perform operations without flattening, but we would need to define the multiplication. We need to be careful to work mode-wise in order not to destroy the tensor structure (R. Bro).

14.2.6 We have the HOSVD. What improvements do we hope to achieve by developing something better than the HOSVD? That is, why do we need something else?

PARAFAC gives us information about rank and in some applications is more informative than the HOSVD. In addition, a rank-revealing decomposition would tell us about correlations in our data. Unfortunately, to the best of our knowledge there is no connection between the tensor rank and the number of significant terms in the HOSVD. However, one could compute the HOSVD and use orthogonal transformations to make the core as *diagonal as possible* to gain information about rank (L. De Lathauwer).

14.2.7 When is n-mode multiplication of tensors invertible? If you perform a set of n-mode multiplications (fixed n) with a standard set of tensors, would it be possible to invert the result?

It should be possible if each face (n-mode wise) is invertible.

15 Workshop Talks

This section briefly summarizes each workshop talk. In most cases, slides to the talks can be found at `http://csmr.ca.sandia.gov/~tgkolda/tdw2004/` or by contacting the presenter.

15.1 *Basics of Tensors*, Carla D. Martin

There are many ways to represent tensors. This talk gave an introduction of some of the options including vectorizing and matricizing in order to facilitate use of linear algebra tools. Most examples used the third-order case for illustration. The tensor rank problem was introduced as well as other notions of rank such as n-rank, strong orthogonal rank, orthogonal rank, and combinatorial rank. When possible, differences and similaries with the matrix case were emphasized. An overview of the algorithms to compute the CP and TUCKER decompositions was presented as were algorithms that compute the best rank-1 approximation to a tensor. The best rank-1 approximation problem is to solve

$$\min_{\mathbf{u},\mathbf{v},\mathbf{w}} ||\mathcal{A} - \mathbf{u} \circ \mathbf{v} \circ \mathbf{w}||.$$

15.2 *Applications of three-mode techniques and outline of major difficulties*, Pieter Kroonenburg

Three-mode analysis is used in the social and behavioral sciences, chemometrics, signal processing, agriculture, tensor faces, and more. This talk began with an introduction to three-way data which included visualization, notation, and terminology. *Modes*, *slices*, and *fibers* were defined as elements of a three-way data array. The *mode* defines the direction of the array (or index); *slices* refer to holding one index constant and varying the others (in other words slices are matrices); and *fibers* are the column vectors of the slices. Several examples of three-way models were presented and analysis of the data was explained in terms of correlation among factors. An brief explanation of the various models used in applications was given (PCA, TUCKER2, TUCKER3, and CANDECOMP-PARAFAC). The talk concluded with a list of technical areas of interest and interpretational issues. Most interesting is how to get others to use these methods and how to explain the results to a group unfamiliar with these methods.

15.3 *Tensor Notation and a* Matlab *Tensor Class for Fast Algorithm Prototyping*, Tammy Kolda and Brett Bader

An introduction to the notation of order-N tensors was presented as well as the possible ways to matricize a tensor. Examples of n-mode multiplication of a tensor and a matrix or vector were also described. Three MATLAB classes for manipulating tensors were developed for the workshop and distributed during the talk. The classes were described using examples and preexisting algorithms. The `tensor` class allows for tensor multiplication and matricizing; whereas the other classes are for representing tensors in terms of the CP model and TUCKER model. The MATLAB classes were developed to aid in algorithm creation. The associated `m`-files can be found at `http://csmr.sandia.gov/~tgkolda/Tensor.zip`.

15.4 *Canonical Tensor Decompositions*, Pierre Comon (impromptu)

This talk explored the intricities of tensor rank and the computation of the CANDECOMP-PARAFAC model. Although one cannot find the tensor rank for an arbitrary tensor, certain properties are known about tensor rank. If R is the tensor rank, then R could be smaller or larger than each mode size. In addition, the product of the two highest mode sizes is an upper bound for the maximum possible rank for a tensor (e.g., if $\mathcal{A}$ is $2 \times 3 \times 4$, then the maximum rank is 12). The tensor rank also depends on the field of scalars (see Section 7.1) and structure of the decomposition (e.g,. symmetry). *Generic* rank of a tensor was defined. There was also a discussion of uniquess of the CP model.

15.5 *Practical Problems and Solutions in Applied Multiway Analysis*, Rasmus Bro

Monitoring pollution from water samples, fluorescence spectroscopy, and monitoring food quality were just some of the applications mentioned that use the CANDECOMP-PARAFAC (CP) model. The advantages of using the CP model were shown by using the data from the water pollution and fluorescence examples. In particular, the uniqueness conditions on the CP model enable

researchers to exactly determine the underlying factors (true spectra) of what is being measured. The current state-of-the-art alternating least squares algorithm to compute the CP decomposition was explained. The current algorithm is more efficient than the one originally proposed because it makes use of the Khatri-Rao-Bro product, rather than the Kronecker product in computations. Problems and possible solutions with convergence and speed of the algorithm were also addressed.

15.6 *3D SVD*, Vince Fernando

An SVD for order-three tensors was proposed which has many similarities to the matrix SVD. In order to explain the 3D SVD, a new norm for third-order tensors based on the matrix spectral norm was defined. Since the matrix spectral norm is the largest singular value of a matrix, the 3D spectral norm can be used to define *singular values* of third-order tensors. Using this definition of singular value, an algorithm involving SVDs of the flattening matrices to compute the largest singular value of a third-order tensor was presented. Repeated application leads to computation of the second, third, etc.- largest singular value of the tensor. The resulting core tensor has these singular values along the super-diagonal and the corresponding edges of each subcube are zeros.

15.7 *An Introduction to Multilinear Algebra based Independent Component Analysis*, Lieven De Lathauwer

Starting with an introduction to higher-order statistics, it was shown that when the sources are independent, tensor decompositions can be used for ICA. In particular, the CANDECOMP-PARAFAC model is used for ICA since the fourth-order cumulant, represented as a tensor, is super-diagonal. In addition an explanation of prewhitening-based algorithms and higher-order schemes was givnen.

15.8 *TensorFaces*, Alex Vasilescu

This work uses the HOSVD for computer facial recognition. Starting with a database of facial images of different people photographed with different expressions, head poses, lighting conditions, and viewpoint, the HOSVD is used for facial recognition of an unknown facial image. This representation is called Tensorfaces. It was shown that Tensorfaces has a higher recognition rate than facial recognition using PCA.

15.9 *Tensor Approximations and Computation of Inverse Matrices*, Eugene Tyrtyshnikov

Matrices arising in solving integral equations are often quite large. Due to memory constraints, these matrices must be represented implicitly. It was shown how matrices are represented using tensor approximations. Specifically, the approximation involves decomposing a tensor as a sum of Kronecker products of *matrices* as in the KPSVD (see §13.1.1). Conditions under which the KPSVD representation may be advantageous were described. Using the KPSVD, the ϵ-rank was defined as the minimal representation in this format. Again, due to memory constraints, a matrix inverse also needs to be computed efficiently. A modified Newton's method algorithm to compute the nearest inverse to a matrix was also described.

15.10 *PCA and Human Gait Patterns*, Cleve Moler (impromptu)

Examples of MATLAB algorithms using PCA were presented. In particular, a MATLAB demonstration of gender-specific differences in walking style was shown. This was work of Nikolaus Troje at Ruhr-University, Bochum, Germany (`http://www.biomotionlab.ca/`).

15.11 *Extracting Structure from Matrices and Tensors by Random Sampling*, Michael Mahoney (impromptu)

Developing fast Monte-Carlo algorithms for matrix computations is useful in order to deal with very large matrices in practice (too large to store in RAM). Such computations include matrix multiplication and the SVD. The idea involves making a *sketch* of the matrix and storing the sketch in RAM for computations. The sketch, A', to a matrix A is made using a CUR decomposition; i.e., $A' = CUR$ where C and R are random samples of the columns and rows of A. Current work presented involves extending the CUR decomposition to tensors.

15.12 *A Real Linear SVD Framework*, Marko Huhtanen (impromptu)

Small rank approximations to a matrix are used in data compression and in solving ill-conditioned problems. This talk was about extending SVD ideas to real linear operators. First, real linear operators were introduced that generalize rank-1 matrices. The SVD was then extended in the language of these real linear operators, including a generalization of the singular values.

15.13 *Questions on Fast Solvers for Kronecker Decompositions*, Ed D'Azevdeo (impromptu)

A fast solver of a sum of two Kronecker products, the QZ decomposition, nearest Kronecker product (NKP) problem, and a complex product were described. Specific questions regarding each of the above ideas were posed to the group.

15.14 *What's Possible and What's Not Possible in Tensor Decompositions*, Lek-Heng Lim (impromptu)

Importants results were presented regarding the extension of the Eckart-Young theorem for matrices. Specifically, it is well known that successive rank-1 approximations to a matrix yield the best rank-k approximation to that matrix. It was shown in the talk that such an extension for tensors is impossible. That is, for some tensors there is no best rank-K approximation ($K < R$) regardless of the norm that is chosen. An example of a rank-3 tensor with no best rank-2 approximation was shown as well as other similar examples. These results hold whether or not orthogonality constraints are imposed on the vectors in the decomposition. In addition, imposing such constraints is often counter-productive if one is interested only in a low-rank approximation, as the minimal representation may be nonorthogonal (see section 7.3). It was also pointed out that orthogonality constraints can only be imposed if there is a natural inner product on the tensor product space (e.g., it does not make sense to impose constraints when dealing with 1-norm or ∞-norm). Important to the theme of the workshop, it was shown that it is not possible to develop globally

convergent algorithms to determine tensor rank, regardless of whether orthogonality constraints are imposed.

15.15 *Handwritten Digit Recognition by HOSVD*, Berkant Savas (impromptu)

This talk was based on work from Savas' masters thesis. Specifically, the HOSVD was used to classify unknown handwritten digits. The algorithm works by constructing an orthogonal basis matrix for each digit using the HOSVD. The second step is to solve a least squares problem to actually compute the unknown digit. Two datasets were used for testing the algorithsm; U.S. Postal Service handwritten digits and the Modified National Institute of Standards and Technology (MNIST) database of handwritten digits. Results of the algorithm and error rates were also presented.

15.16 *Genomic Signal Processing*, Orly Alter

This talk explained how the SVD and GSVD are used to compare and contrast the yeast and human cell-cycles. In particular, expression datasets containing molecular data of each of the cell cycles were used. The GSVD helped to find biological similarity and dissimiliarity between each cell cycle.

15.17 *Nature of Degenerate Solutions*, Richard Harshman

This talk explained the problem of degeneracy that sometimes occurs in the CP model. *Degenerate* solutions are when two or more factors are highly negatively correlated. This can occur when the CP model is applied to data that is better explained using the TUCKER model. An example of such solutions was shown using a dataset of TV show ratings. In some cases one can prevent degeneracy by imposing orthogonality constraints on the factors in one mode. A theorem was presented that explains why some of the degeneracies occur.

16 Acknowledgements

I would like to thank Pierre Comon, Lieven De Lathauwer, and Lek-Heng Lim for their help in clarifying sections of these notes. I would also like to thank Misha Kilmer for her thorough review and insightful comments about many points raised during the workshop. Finally, I would like to thank Tammy Kolda and Charles Van Loan for their comments and suggested revisions on the entire document.