Tensor Decompositions and Applications
Tamara G. Kolda and Brett W. Bader
Part I

Jiguang Shen

September 22, 2015
A $N$-th order **tensor** is an element of the tensor product of $N$ vector spaces, each of which has its own coordinate system.

\[
\begin{align*}
\mathbf{a} &= a_i \mathbf{e}_i & \mathbf{b} &= b_j \tilde{\mathbf{e}}_j & \mathbf{c} &= c_k \hat{\mathbf{e}}_k & \quad & \text{(Vector spaces)} \\
\mathbf{A} &= \mathbf{a} \circ \mathbf{b} = a_i b_j \mathbf{e}_i \circ \tilde{\mathbf{e}}_j & \quad & \text{(Matrix, second order tensor)} \\
\mathbf{X} &= \mathbf{a} \circ \mathbf{b} \circ \mathbf{c} = a_i b_j c_k \mathbf{e}_i \circ \tilde{\mathbf{e}}_j \circ \hat{\mathbf{e}}_k & \quad & \text{(Third order tensor)} \\
\end{align*}
\]

$N$-th order (ways or modes) tensor has $N$ dimensions.
### Table: Matrix to high order tensor

<table>
<thead>
<tr>
<th></th>
<th>Matrix</th>
<th>High order tensor</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Columns/Rows) $A_{i,:}, A_{,:j}$</td>
<td>(Fibers) $X_{i:j,:}, X_{i:k,:};$ (Slices) $X_{i::}, X_{j::}, X_{::k}$</td>
<td>mode-3, 2, 1; Horizontal, Lateral, Frontal slices</td>
</tr>
<tr>
<td>$A : B = \sum_{i=1}^{\text{Rows}} \sum_{j=1}^{\text{Columns}} A_{ij} B_{ij}$</td>
<td>$X : Y = \sum_{i=1}^{\text{Rows}} \sum_{j=1}^{\text{Columns}} \sum_{k=1}^{\text{Slices}} X_{ijk} Y_{ijk}$</td>
<td></td>
</tr>
<tr>
<td>$|A|<em>F = \sqrt{\sum</em>{i=1}^{\text{Rows}} \sum_{j=1}^{\text{Columns}} A_{ij}^2}$</td>
<td>$|X| = \sqrt{\sum_{i=1}^{\text{Rows}} \sum_{j=1}^{\text{Columns}} \sum_{k=1}^{\text{Slices}} X_{ijk}^2}$</td>
<td></td>
</tr>
<tr>
<td>(Rank one matrix) $A = ab^T$</td>
<td>(Rank one tensor) $X = a \circ b \circ c$</td>
<td>$X_{ijk} = a_i b_j c_k$</td>
</tr>
<tr>
<td>$A_{ij} = a_i b_j$</td>
<td>(Symmetric) $A = A^T$</td>
<td>(Supersymmetric: cubical + symmetry) $X \in \mathbb{R}^{I \times I \times I}$</td>
</tr>
<tr>
<td>$A = I, A_{ij} = \delta_{ij}$</td>
<td>$X = I, X_{ijk} = \delta_{ijk}$</td>
<td>$X_{ijk}$ is constant when permuting $i, j, k$</td>
</tr>
</tbody>
</table>

Jiguang Shen  
Tensor Decompositions and Applications
Vectorization of Matrix: arrange all columns of the matrix to be a column vector.

\[
\begin{pmatrix}
1 & 3 \\
2 & 4
\end{pmatrix} \Rightarrow 
\begin{pmatrix}
1 \\
2 \\
3 \\
4
\end{pmatrix}
\]

Mode-\(n\) matricization of a tensor: arrange all mode-\(n\) fibers to be the columns of the resulting matrix. (Let's skip the formal definition.)

\[
X_{ij1} = 
\begin{pmatrix}
12 & 4 & 7 \\
2 & 10 & 6 \\
1 & 16 & 9
\end{pmatrix} \quad X_{ij2} = 
\begin{pmatrix}
13 & 3 & 73 \\
21 & 0 & 26 \\
11 & 27 & 19
\end{pmatrix}
\]

Mode-1 matricization: \(X_{(1)}\)

\[
\begin{pmatrix}
12 & 4 & 7 & 13 & 3 & 73 \\
2 & 10 & 6 & 21 & 0 & 26 \\
1 & 16 & 9 & 11 & 27 & 19
\end{pmatrix}
\]
Mode-2 matricization: $X_{(2)}$

\[
\begin{pmatrix}
12 & 2 & 1 & 13 & 21 & 11 \\
4 & 10 & 16 & 3 & 0 & 27 \\
7 & 6 & 9 & 73 & 26 & 19
\end{pmatrix}
\]

Mode-3 matricization: $X_{(3)}$

\[
\begin{pmatrix}
12 & 2 & 1 & 4 & 10 & 16 & \cdots & 6 & 9 \\
13 & 21 & 11 & 3 & 0 & 27 & \cdots & 26 & 19
\end{pmatrix}
\]

How do we change the results if we have an additional slice?

\[
X_{ij3} = \begin{pmatrix}
-1 & -1 & -1 \\
-1 & -1 & -1 \\
-1 & -1 & -1
\end{pmatrix}
\]
Tensor multiplication: \( n \)-mode product

The \( n \)-mode product of a tensor \( \mathbf{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N} \) with a matrix \( \mathbf{U} \in \mathbb{R}^{J \times I_n} \) is denoted by \( \mathbf{X} \times_n \mathbf{U} \).

\[ (\mathbf{X} \times_n \mathbf{U})_{i_1 \cdots i_{n-1} j_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 \cdots i_{n-1} i_n+1 \cdots i_N} u_{j_n} \]

- \( \mathbf{Y} = \mathbf{X} \times_n \mathbf{U} \iff \mathbf{Y}_{(n)} = \mathbf{U} \mathbf{X}_{(n)} \)
- related to change of basis, when a tensor defines a multilinear operator.

The \( n \)-mode product of a tensor \( \mathbf{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N} \) with a vector \( \mathbf{v} \in \mathbb{R}^{I_n} \) is denoted by \( \mathbf{X} \bar{\times}_n \mathbf{v} \).

\[ (\mathbf{X} \bar{\times}_n \mathbf{v})_{i_1 \cdots i_{n-1} i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 \cdots i_N} v_{i_n} \]

- precedence matters, contraction of tensor.
Matrix Kronecker, Khatri–Rao and Hadamard Products

A, B

- Kronecker product: \( A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1J}B \\ \vdots & \vdots & \vdots \\ a_{I1}B & \cdots & a_{IJ}B \end{pmatrix} \)

- Khatri–Rao product: \( A \odot B = [a_1 \otimes b_1 \ a_2 \otimes b_2 \ \cdots \ a_K \otimes b_K] \) (column matching)

- Hadamard Product: \( (A \ast B)_{ij} = A_{ij}B_{ij} \)

Examples:

\[
A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}
\]

\( A \otimes B?A \odot B?A \ast B?A \ast A? \)

MATLAB:

```matlab
kron(A,B); % Kronecker
for i = 1:3 % Khatri-Rao
    C(:,i) = kron(A(:,i),B(:,i));
end
A.*B % Hadamard
```
CP Decomposition (CANDECOMP/PARAFAC Decomposition), factorizes a tensor into a sum of component rank-one tensors. CP decomposition can be treated as a generalization of SVD to higher order tensors.

$$A = \sum_{r=1}^{R} \sigma_r u_r \circ v_r, \text{ with } \sigma_1 \geq \sigma_2 \cdots \geq \sigma_R > 0 \ (SVD)$$

$$\mathcal{X} \in \mathbb{R}^{I \times J \times K} \approx \sum_{r=1}^{R} \lambda_r a_r \circ b_r \circ c_r = \{\lambda; A, B, C\}$$

$A$, $B$, and $C$ are combination of the vectors from the rank-one components and are normalized to length one.

In matricized form:

$$X_{(1)} \approx A\Lambda (C \odot B)^T, \ X_{(2)} \approx B\Lambda (C \odot A)^T, \ X_{(3)} \approx C\Lambda (B \odot A)^T$$

where $\Lambda = \text{diag}(\lambda)$. It can be extended to $n$-th order tensor.
The rank of a tensor $\mathcal{X}$, denoted $\text{rank}(\mathcal{X})$, is the smallest number of rank-one tensors that generate $\mathcal{X}$ as their sum. (the smallest number of components in an exact “=” CP decomposition. In other words, what is $R$?) An exact CP decomposition with $R = \text{rank}(\mathcal{X})$ components is called the rank decomposition.

- Definition is an exact analogue from matrix rank (the dimension of the vector space spanned by its columns).
- Properties are quite different:
  1. Field dependent: may not the same over $\mathbb{R}$ and $\mathbb{C}$.
  2. No straightforward algorithm to compute the rank of specific tensor (NP-hard problem).
- Matrix decompositions are not unique (why?), however high order tensor decompositions are often unique.
Matrix: a best rank-$k$ approximation is given by the leading $k$ factors of the SVD. A rank-$k$ approximation that minimizes $\|A - B\|$ is given by:

$$A = \sum_{r=1}^{k} \sigma_r u_r \odot v_r,$$

with $\sigma_1 \geq \sigma_2 \cdots \geq \sigma_R > 0$

Not true for high order tensors! a best rank-$k$ approximation may not exist! (degeneracy)
Computing CP Decomposition

Suppose $R$ is fixed, for a third order tensor $\mathcal{X}$, we are looking for CP Decomposition of $\hat{\mathcal{X}}$:

$$\min_{\hat{\mathcal{X}}} \| \mathcal{X} - \hat{\mathcal{X}} \| \quad with \quad \hat{\mathcal{X}} = [\lambda; A, B, C]$$

The alternating least squares (ALS) method: (fixed all but one matrix)

- Fix $C$ and $B$ (specify in any way) and rewrite (how?), $\hat{A} := A\Lambda$

$$\min_{\hat{A}} \| X^{(1)} - \hat{A}(C \odot B)^T \|_F$$

- Solution: $\hat{A} = X^{(1)}[(C \odot B)^T]^\dagger$, or a better version:
  $\hat{A} = X^{(1)}(C \odot B)[(C^T C \ast B^T B)]^\dagger$. $^\dagger$ refers to Moore–Penrose pseudoinverse (MATLAB: $B = \text{pinv}(A)$).

- Normalization to get $A$, storing norms as $\lambda$.

- Fix $A$ and $B$ to compute $C$. ..., until reach convergence (i.e. error stops decreasing).

Implementability? 😊 Convergence? 😞 Efficiency? 😞
Sandia National Laboratories provide a MATLAB Tensor Toolbox which includes an implementation of CP Decomposition.

- `cp_als` function implements ALS algorithm for CP Decomposition with fixed rank. e.g. \( P = cp_als(X, 2) \);
- other approaches are available `cp_apr`, `cp_nmu`, `cp_opt`, etc.
- special treatment for sparse tensor: a “greedy” CP.

Applications:

- signal/image processing
- neuroscience
- data mining (user × keyword × time: chatroom tensors)
- stochastic PDEs.
- …
THANK YOU!