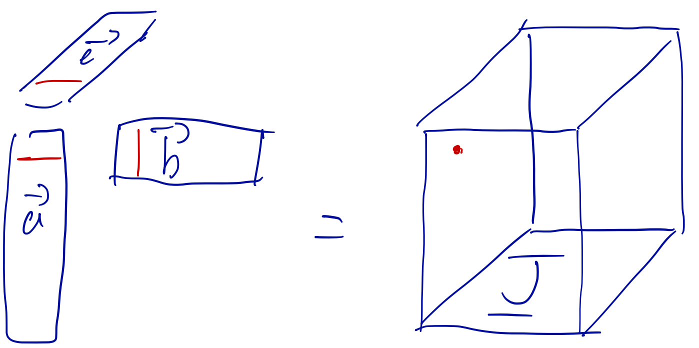
The CP Decomposition and the Rank of a Tensor

Vector outer product of N vectors $\vec{a}^{(1)}, \vec{a}^{(2)}, \dots, \vec{a}^{(N)}$ is an N-way tensor

$$\underline{T} = \vec{a}^{(1)} \circ \vec{a}^{(2)} \circ \dots \circ \vec{a}^{(N)}$$

with every element defined as the product of the corresponding elements of the vectors

$$t_{i_1 i_2 \dots i_N} = a^{(1)}_{i_1} a^{(2)}_{i_2} \cdots a^{(N)}_{i_N} .$$



$$t_{ijk} = a_i b_j c_k$$

The CP decomposition

The exact CP decomposition of an N-way tensor $\underline{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ has the form

$$\underline{T} = \sum_{r=1}^{R} \vec{a}_r^{(1)} \circ \vec{a}_r^{(2)} \circ \cdots \circ \vec{a}_r^{(N)},$$

where $R \in \mathbb{N}$ and $\vec{a}_r^{(i)} \in \mathbb{R}^{I_i}$ for all $i \in [N]$ and $r \in [R]$.

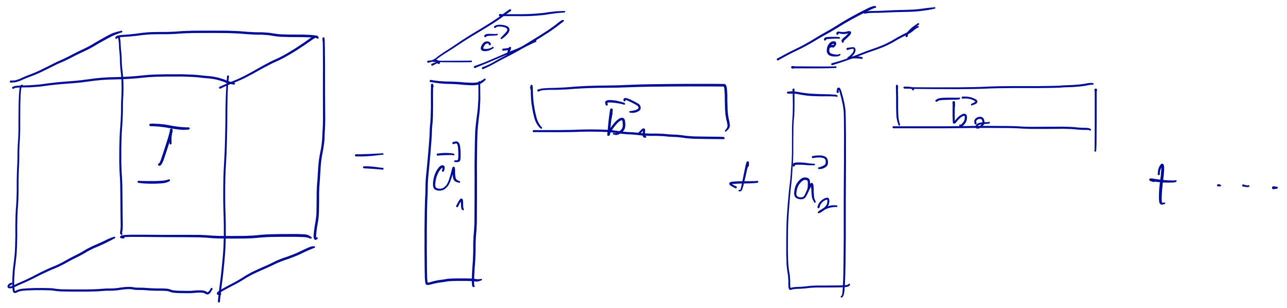
In the approximate (or fixed-rank) CP decomposition, the size R is given, and we're looking for the least-error decomposition

$$\left\| \underline{T} - \sum_{r=1}^{R} \vec{a}_r^{(1)} \circ \vec{a}_r^{(2)} \circ \cdots \circ a_r^{(N)} \right\|.$$

For now, we concentrate on 3-way tensors, and write

$$\underline{T} = \sum_{r=1}^{R} \vec{a}_r \circ \vec{b}_r \circ \vec{c}_r .$$

Visually, the 3-way CP is



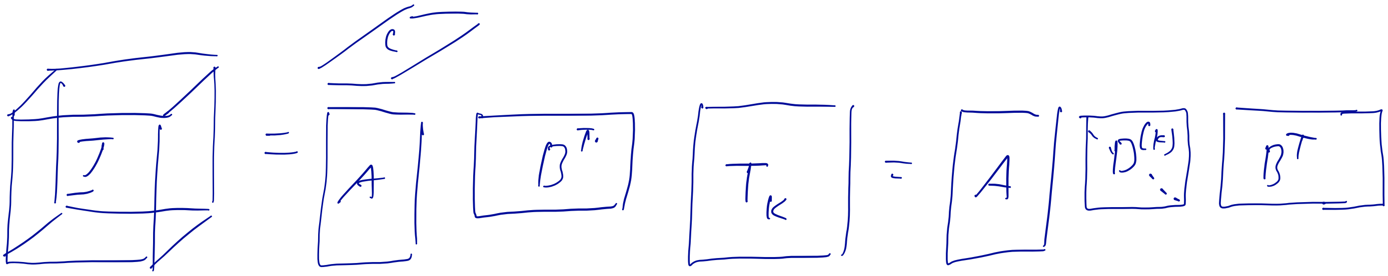
We can gather the vector for each mode in <u>factor matrices</u>. In the 3-way setting, for $\underline{T} \in \mathbb{R}^{I \times J \times K}$, we have

$$A = [\vec{a}_1 \ \vec{a}_2 \cdots \vec{a}_R], \quad B = [\vec{b}_1 \ \vec{b}_2 \cdots \vec{b}_R], \quad C = [\vec{c}_1 \ \vec{c}_2 \cdots \vec{c}_R]$$
$$I \times R \qquad\qquad J \times R \qquad\qquad K \times R$$

We can express the 3-way CP decomposition using the frontal slices of $\underline{T}$ and the factor matrices:

$$T_k = A D^{(k)} B^T,$$

where $D^{(k)} = \mathrm{diag}(C(k,:))$, i.e. a diagonal matrix with the k-th row of C on its diagonal.



The frontal slice formulation doesn't generalize easily for more than 3 modes. For more generalized representation, we need the <u>Khatri-Rao matrix product</u>: given matrices $A \in \mathbb{R}^{I \times K}$ and $B \in \mathbb{R}^{J \times K}$, their Khatri-Rao product is

$$A \odot B = \begin{pmatrix} a_{11}\vec{b}_1 & a_{12}\vec{b}_2 & \cdots & a_{K1}\vec{b}_K \\ a_{21}\vec{b}_1 & a_{22}\vec{b}_2 & \cdots & a_{K2}\vec{b}_K \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\vec{b}_1 & a_{I2}\vec{b}_2 & \cdots & a_{IK}\vec{b}_K \end{pmatrix} \in \mathbb{R}^{IJ \times K}.$$

That is, each column of B is copied I times, and the i-th copy of the k-th column B multiplied by a_{ik}. The Khatri-Rao product can be written more concisely using the

Kronecker matrix product $A \otimes B$.

If $A \in \mathbb{R}^{I \times J}$ and $B \in \mathbb{R}^{K \times L}$, their Kronecker product is

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1J}B \\ a_{21}B & a_{22}B & \cdots & a_{2J}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}B & a_{I2}B & \cdots & a_{IJ}B \end{pmatrix} \in \mathbb{R}^{(IK) \times (JL)}.$$

Notice that in Kronecker product, the matrices can be of arbitrary size, whereas in Khatri-Rao, they must have the same number of columns.

The Khatri-Rao product of $A \in \mathbb{R}^{I \times K}$ and $B \in \mathbb{R}^{J \times K}$ can now be written as

$$A \odot B = [\vec{a}_1 \otimes \vec{b}_1 \;\; \vec{a}_2 \otimes \vec{b}_2 \; \cdots \; \vec{a}_K \otimes \vec{b}_K],$$

that is, Khatri-Rao is "column-wise Kronecker" product.

If $A \in \mathbb{R}^{I \times R}$, $B \in \mathbb{R}^{J \times R}$, and $C \in \mathbb{R}^{K \times R}$ are the factor matrices of a CP decomposition of tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$, then

$$T_{(1)} = A(C \odot B)^T$$

$$T_{(2)} = B(C \odot A)^T$$

$$T_{(3)} = C(B \odot A)^T.$$

More generally, if $\mathcal{T}$ has N modes and factor matrices $A^{(1)}, A^{(2)}, \dots, A^{(N)}$,

$$T_{(n)} = A^{(n)}\left(A^{(N)} \odot \cdots \odot A^{(n+1)} \odot A^{(n-1)} \odot \cdots \odot A^{(1)}\right)^T.$$

To gain intuition on the Khatri-Rao formulation, consider the frontal slice formulation of CP:

$$T_k = A D^{(k)} B^T,$$

with $D^{(k)} = \mathrm{diag}(C(:,k))$. The same factor A appears with all frontal slices, so we can just stack them:

$$\begin{bmatrix} T_1 & T_2 & \cdots & T_K \end{bmatrix} = A \underbrace{\begin{bmatrix} D^{(1)}B^T & D^{(2)}B^T & \cdots & D^{(K)}B^T \end{bmatrix}}_{E}$$

$$\begin{bmatrix} T_1 & T_2 & \cdots & T_K \end{bmatrix} = A \times \underbrace{\begin{bmatrix} D^{(1)} \times B^T & D^{(2)} \times B^T & \cdots & D^{(K)} \times B^T \end{bmatrix}}_{E}$$

The first row of E has the first column of B multiplied by c_{11} followed by the first column of B multiplied by c_{21}, and so on. Hence

$$\vec{e}_{1:} = [c_{11}\vec{b}_1 \;\; c_{21}\vec{b}_1 \;\cdots\; c_{K1}\vec{b}_1] = (\vec{c}_1 \otimes \vec{b}_1)^T.$$

Extending this to all rows of E we see that

$$E^T = (C \odot B)^T$$

and hence

$$T_{(1)} = [T_1 \; T_2 \cdots T_K] = A\,(C \odot B)^T.$$

The connections in the other modes can be derived analogously.

One sometimes normalizes the columns of the factor matrices to unit length. The lengths are then stored in factors $\lambda_r = \|\vec{a}_r\| \cdot \|\vec{b}_r\| \cdot \|\vec{c}_r\|$, collected in a vector $\vec{\lambda} \in \mathbb{R}^R$, or in a matrix $\Lambda = \mathrm{diag}(\vec{\lambda}) \in \mathbb{R}^{R \times R}$. Then $T_{(1)} = A \Lambda (C \odot B)^T$ etc. and

$$\underline{T} = \sum_{r=1}^{R} \lambda_r \vec{a}_r \circ \vec{b}_r \circ \vec{c}_r .$$

A common notation for the CP decomposition is to write

$$\underline{T} = [\![A, B, C]\!] = \sum_{r=1}^{R} \vec{a}_r \circ \vec{b}_r \circ \vec{c}_r ,$$

or with the scaling

$$\underline{T} = [\![\vec{\lambda} ; A, B, C]\!] = \sum_{r=1}^{R} \lambda_r \vec{a}_r \circ \vec{b}_r \circ \vec{c}_r .$$

ALS algorithm for CP

The formulations

$$T_{(1)} = A(C \odot B)^T \quad \text{etc}$$

provide a way to solve the (approximate) CP decomposition. When C and B are fixed ($C \odot B$ is a fixed matrix, call it D, and the problem becomes: "Given matrices $T_{(1)}$ and D, find matrix A that minimizes $\|T_{(1)} - AD^T\|_F$". This can be solved using the SVD and pseudo-inverse as $A = T_{(1)}(D^T)^+$, where $(\ )^+$ is the Moore–Penrose pseudo-inverse. This leads to the following algorithm:

sample random B and C
<u>repeat</u>
    let $A \leftarrow T_{(1)}((C \odot B)^T)^+$
    let $B \leftarrow T_{(2)}((C \odot A)^T)^+$
    let $C \leftarrow T_{(3)}((B \odot A)^T)^+$
<u>until</u> convergence

The ALS algorithm requires us to compute the pseudo-inverses of $(C \odot B)^T$, $(C \odot A)^T$, and $(B \odot A)^T$, which are R-by-JK, R-by-IK, and R-by-IJ matrices, respectively. This is an expensive operation, but if these matrices have a full row rank — which is likely, as often $R \ll \min\{IJ, IK, JK\}$ — then we can use the following equality

$$(A \odot B)^+ = \left((A^T A) * (B^T B)\right)^+ (A \odot B)^T, \quad (\circledast)$$

where $X * Y$ is the <u>Hadamard matrix product</u> (or elementwise product) between $X \in \mathbb{R}^{I \times J}$ and $Y \in \mathbb{R}^{I \times J}$

$$X * Y = \begin{pmatrix} x_{11} y_{11} & x_{12} y_{12} & \cdots & x_{1J} y_{1J} \\ x_{21} y_{21} & x_{22} y_{22} & \cdots & x_{2J} y_{2J} \\ \vdots & \vdots & \ddots & \vdots \\ x_{I1} y_{I1} & x_{I2} y_{I2} & \cdots & x_{IJ} y_{IJ} \end{pmatrix} \in \mathbb{R}^{I \times J}$$

The proof of identity (⊛) is left as a homework, but it involves the following identity that is also occassionally useful on itself: For $X \in \mathbb{R}^{I \times K}$ and $Y \in \mathbb{R}^{J \times K}$, we have

$$(X \odot Y)^T (X \odot Y) = X^T X * Y^T Y.$$

<u>Proof</u>: Let $X' = X^T X$ and notice that $x'_{ik} = \langle \vec{x}_i, \vec{x}_k \rangle$. Similarly, for $Y' = Y^T Y$, we have $y'_{jk} = \langle \vec{y}_j, \vec{y}_k \rangle$. Now, let

$$Z = (X \odot Y)^T (X \odot Y)$$
$$= [\vec{x}_1 \otimes \vec{y}_1 \; \cdots \vec{x}_K \otimes \vec{y}_K]^T [\vec{x}_1 \otimes \vec{y}_1 \; \cdots \vec{x}_K \otimes \vec{y}_K],$$

and consider a single element z_{kl}:

$$z_{kl} = \langle \vec{x}_k \otimes \vec{y}_k, \vec{x}_l \otimes \vec{y}_l \rangle = \sum_{i=1}^{I} \langle x_{ik} \vec{y}_k, x_{il} \vec{y}_l \rangle$$
$$= \sum_{i=1}^{I} x_{ik} x_{il} \langle \vec{y}_k, \vec{y}_l \rangle$$
$$= \langle \vec{x}_k, \vec{x}_l \rangle \langle \vec{y}_k, \vec{y}_l \rangle = x'_{kl} y'_{kl},$$

and hence $Z = X' * Y' = X^T X * Y^T Y$. □

With equality (⊛) we can write

$$A = T_{(1)} \Big[\underbrace{\left((C \odot B)^T \right)}_{R \times JK} \Big]^+$$

as

$$A = T_{(1)} (C \odot B) \underbrace{\left[C^T C * B^T B \right]^+}_{R \times R},$$

and we only have to take the pseudo-inverse from a much smaller matrix. This formulation can, however, cause issues with the numerical stability.

ALS is not the only possibility. We can instead use, for instance, gradient-based methods: each row $\vec{a}_{i:}$ can be updated based on the gradient

$$\vec{a}_{i:} \leftarrow \vec{a}_{i:} - \delta \frac{\partial}{\partial \vec{a}_{i:}} \sum_{j=1}^{JK} \left(T_{(1)}(i,j) - \left(\vec{a}_{i:} \left(C \odot B \right)^T \right)_{j} \right)^2 .$$

ALS is the most commonly used approach, though.