IFT 6760A - Lecture 10
Tensor Decompositions - Part 2

Scribe(s): David Venuto, Junhao Wang and Nicolas Gagné **Instructor:** Guillaume Rabusseau

1 Summary

In the previous lecture we saw some of the basic tensor operations, such as the Kronecker product, the Khatri-Rao product and the outer product. We saw the notion of CP decomposition and noted that the minimum CP rank approximation problem is ill-defined as the set of tensors with a rank less than k is not closed. We also covered the alternating minimization method (ALS).

In this lecture we covered the CP decomposition and two algorithms for computing it: Jennrich's algorithm and the Alternating Least Squares algorithm. We then introduced the Tucker decomposition and argued that it can be seen as a higher order SVD. After comparing the pros and cons of the CP and Tucker decompositions, we concluded with a quick introduction to the tensor Train decomposition.

2 Canonical Polyadic (CP) Decomposition

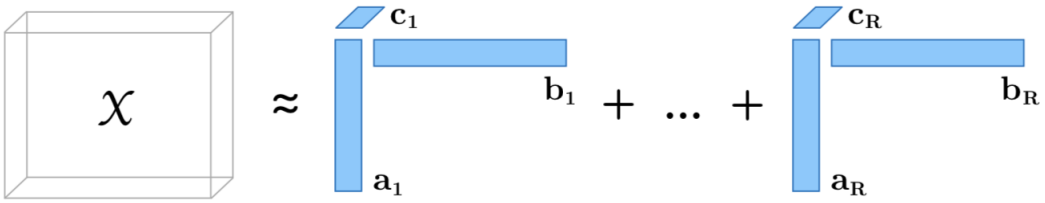


Figure 1: CP Decomposition, Image From [4]

Rank decomposition expresses tensors as the function of a finite number of rank-one tensors. We have Canonical Decomposition [1] and Parallel Factors Decomposition [3] methods, that overall solve the same problem and are termed canonical polyadic decomposition methods (CPD) [4]. Recall CPD aims to represent order-d tensor $\mathcal{T}$ as a linear combination of suitably large R number of rank-1 tensors. The graphical view of CP Decomposition is also given in Figure 1, image taken from [4]. The formulation of CPD for a third-order tensor is given by:

Definition 1. *The objective in CPD is to find* $\min_{\hat{\mathcal{T}}} \|\mathcal{T} - \hat{\mathcal{T}}\|_F$ *where* $\hat{\mathcal{T}} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ *where* $\mathbf{A}, \mathbf{B}, \mathbf{C}$ *are called factor matrices and hold the combinations of vectors from the rank-1 components as columns. For* $\mathcal{T}^{d_1 \times d_2 \times d_3}$, $\mathbf{A}, \mathbf{B}, \mathbf{C}$ *are of size* $d_1 \times R, d_2 \times R, d_3 \times R$ *and* $A = [a_1\ a_2\ \dots\ a_R]$.

If $\min_{\hat{\mathcal{T}}} \|\mathcal{T} - \hat{\mathcal{T}}\|_F = 0$ then we have the exact low rank approximation of $\mathcal{T}$. If $\mathcal{T} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, then the CPD of $\mathcal{T}$ can also be stated as simple operations on its factor matrices using matricization of $\hat{\mathcal{T}}$: if $\hat{\mathcal{T}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ we have

$$\begin{aligned}\hat{\mathcal{T}}_{(1)} &= (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T \\ \hat{\mathcal{T}}_{(2)} &= (\mathbf{C} \odot \mathbf{A})\mathbf{B}^T \\ \hat{\mathcal{T}}_{(3)} &= (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T\end{aligned} \tag{1}$$

Let $\mathcal{T}^{d_1 \times d_2 \times d_3}$ be a tensor of CP rank R. We recall that, by definition, this means we can write:

$$\mathcal{T} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \tag{2}$$

where we have matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$. Our goal in the remaining of this section is to recover a (approximate) CP decomposition of $\mathcal{T}$ given its rank R.
For the general case we have:

$$\begin{aligned}\hat{\mathcal{T}} &= \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \ldots \mathbf{a}_r^{(n)} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(n)}]\!] \\ \hat{\mathcal{T}}_{(i)} &= (\mathbf{A}^{(n)} \odot \ldots \odot \mathbf{A}^{(i+1)} \odot \mathbf{A}^{(i-1)} \odot \ldots \odot \mathbf{A}^{(1)})\mathbf{A}^{(i)^T}\end{aligned} \tag{3}$$

2.1 Alternating Least Squares (ALS) Algorithm

One way to compute a CP decomposition of a tensor is Alternating Least Squares Algorithm. Our objective is a minimized least squares loss for matrices in $\mathcal{T} = [\![\mathbf{A_1}, \mathbf{A_2}, \ldots, \mathbf{A}_n]\!]$:

$$\min_{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n} \mathcal{L}(\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n) \tag{4}$$

where $\mathcal{L}(\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n) = ||\mathcal{T} - [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n]\!]||_F$
In this algorithm, all factor matrices are fixed except for one. This allow the optimization of of the non-fixed matrix. This step of not fixing matrix i is repeated for every matrix until we meet a stopping criteria. The algorithm for ALS is formalized as:

Algorithm 1 ALS Algorithm

Initialize $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n$ Randomly
repeat
  for $i = 1, 2, \ldots, n$ **do**
    $\mathbf{A}_i = \arg\min_{\mathbf{X}} \mathcal{L}(\mathbf{A}_1, \ldots, \mathbf{A}_{i-1}, \mathbf{X}, \mathbf{A}_{i+1}, \ldots, \mathbf{A}_n)$
  end for
until convergence
Return $\mathbf{A}_1, \ldots, \mathbf{A}_n$

In the 3-way case for $\mathcal{T} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, we have the following 3 steps performed until convergence criteria is met. Recalling that $\mathbf{T}_{(i)}$ is the mode-i matricization of $\mathcal{T}$ [4], we have:

$$\begin{aligned}\mathbf{A} &\leftarrow \arg\min_{\mathbf{A}} ||\mathbf{T}_{(1)} - (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T||_F \\ \mathbf{B} &\leftarrow \arg\min_{\mathbf{B}} ||\mathbf{T}_{(2)} - (\mathbf{C} \odot \mathbf{A})\mathbf{B}^T||_F \\ \mathbf{C} &\leftarrow \arg\min_{\mathbf{C}} ||\mathbf{T}_{(3)} - (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T||_F\end{aligned} \tag{5}$$

The optimal solution to thus minimization problem is given below where $*$ refers to Hadamard product:

$$\begin{aligned}\hat{\mathbf{A}} &= \mathcal{T}_{(1)}[(\mathbf{C} \odot \mathbf{B})^T]^\dagger = \mathcal{T}_{(1)}(\mathbf{C} \odot \mathbf{B})(\mathbf{C}^T\mathbf{C} * \mathbf{B}^T\mathbf{B})^\dagger \\ \hat{\mathbf{B}} &= \mathcal{T}_{(2)}[(\mathbf{C} \odot \mathbf{A})^T]^\dagger = \mathcal{T}_{(2)}(\mathbf{C} \odot \mathbf{A})(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A})^\dagger \\ \hat{\mathbf{C}} &= \mathcal{T}_{(3)}[(\mathbf{B} \odot \mathbf{A})^T]^\dagger = \mathcal{T}_{(3)}(\mathbf{B} \odot \mathbf{A})(\mathbf{B}^T\mathbf{B} * \mathbf{A}^T\mathbf{A})^\dagger\end{aligned} \tag{6}$$

Note that ALS is not guaranteed to converge to the optimal solution, as the problem is NP-hard. Furthermore, a normalization step for matrix columns can be performed for $\mathbf{A}_i$ at each iteration to improve numerical stability.

2.2 Jennrich's algorithm

Another way of retrieving the CP decomposition $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ of a rank R tensor $\boldsymbol{\mathcal{T}}$ is the Jennrich's algorithm. In this subsection, we will assume that $\mathbf{A}$ and $\mathbf{B}$ are full rank with rank $R \leq \min(d_1, d_2)$. The Jennrich's algorithm works as follows.

First, we let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{d_3}$ be two random vectors. Hitting $\boldsymbol{\mathcal{T}}$ with random vector $\mathbf{x} \in \mathbb{R}^{d_3}$, we define $\mathbf{M}_x = \boldsymbol{\mathcal{T}} \bullet_3 \mathbf{x} = \sum_{r=1}^{R} \langle \mathbf{c}_r, \mathbf{x} \rangle \, \mathbf{a}_r \circ \mathbf{b}_r = \mathbf{A}\boldsymbol{\Lambda}_x \mathbf{B}^\top$, where $\boldsymbol{\Lambda}_x = \mathbf{Diag}(\langle \mathbf{c_r}, \mathbf{x} \rangle) \in \mathbb{R}^{R \times R}$ and and $\mathbf{a}_r$, $\mathbf{b}_r$ and $\mathbf{c}_r$ are the respective column vectors of $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. Then, we take another slice throught the tensor by hitting it with the random vector $\mathbf{y} \in \mathbb{R}^{d_3}$: $\mathbf{M}_y = \boldsymbol{\mathcal{T}} \bullet_3 \mathbf{y} = \mathbf{A}\boldsymbol{\Lambda}_y \mathbf{B}^\top$, where $\boldsymbol{\Lambda}_y = \mathbf{Diag}(\langle \mathbf{c_r}, \mathbf{y} \rangle) \in \mathbb{R}^{R \times R}$. To recover $\mathbf{A}$, we look at

$$\mathbf{M}_x \mathbf{M}_y^\dagger = \mathbf{A}\boldsymbol{\Lambda}_x \mathbf{B}^\top \left(\mathbf{B}^\top\right)^\dagger \boldsymbol{\Lambda}_y^{-1} \mathbf{A}^\dagger = \mathbf{A}\boldsymbol{\Lambda}_x \boldsymbol{\Lambda}_y^{-1} \mathbf{A}^\dagger. \tag{7}$$

This means that the columns of $\mathbf{A}$ are the eigenvectors of $\mathbf{M}_x \mathbf{M}_y^\dagger$. Further assuming that no columns of $\mathbf{C}$ are multiples of another, we have that, since the vectors $\mathbf{x}$ and $\mathbf{y}$ were random, all the elements of $\boldsymbol{\Lambda}_x \boldsymbol{\Lambda}_y^{-1}$ are distinct with probability 1. [1] We can therefore recover $\mathbf{A}$ with probability 1.
To recover $\mathbf{B}$ with probability 1, we proceed analogously by looking at

$$\left(\mathbf{M}_x^\dagger \mathbf{M}_y\right)^T = \mathbf{B}\boldsymbol{\Lambda}_y \boldsymbol{\Lambda}_x^{-1} \mathbf{B}^\dagger. \tag{8}$$

Lastly, to recover $\mathbf{C}$, we pair the columns of $\mathbf{A}$ and $\mathbf{B}$ and solve the resulting linear system:

$$\boldsymbol{\mathcal{T}}_{(3)} = \mathbf{C} \left(\mathbf{B} \odot \mathbf{A}\right)^\top. \tag{9}$$

While working well for some problems, Jennrich's algorithm only takes random slices of a tensor and hence does not use the full tensor structure. Moreover, it requires good eigen-gap on the eigendecompositions of the factor matrices, the lack of which could lead to numerical instability.
We have shown two algorithms—the ALS algorithm and the Jennrich's algorithm— for retrieving a CP decomposition of $\boldsymbol{\mathcal{T}}$, but is the CP decomposition unique?

2.3 Uniqueness

For a tensor $\boldsymbol{\mathcal{T}}$ of rank R, we can write $\boldsymbol{\mathcal{T}} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$. Since the product is invariant under permutation and rescaling, this decomposition is not unique. Indeed, we have

$$\boldsymbol{\mathcal{T}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = [\![\mathbf{A}\boldsymbol{\Pi}, \mathbf{B}\boldsymbol{\Pi}, \mathbf{C}\boldsymbol{\Pi}]\!] = [\![\mathbf{A}\mathbf{D}_1, \mathbf{B}\mathbf{D}_2, \mathbf{C}\mathbf{D}_3]\!],$$

for a permutation matrix $\boldsymbol{\Pi}$ and for diagonal matrices $\mathbf{D}_1, \mathbf{D}_2$ and $\mathbf{D}_3$ satisfying $\mathbf{D}_1\mathbf{D}_2\mathbf{D}_3 = \mathbf{I}$. Modulo those two operations, is a decomposition unique? If in addition to assuming that $\mathbf{A}$ and $\mathbf{B}$ have full rank, we further assume that no columns of $\mathbf{C}$ is a multiple of another column, the answer is yes! Indeed, we get the following uniqueness theorem [3].

Theorem 2 (Harshman). *If $\boldsymbol{\mathcal{T}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ is a CP decomposition of rank R, with $R \leq \min(d_1, d_2, d_3)$, $\mathbf{A}, \mathbf{B}$ are full rank and no columns of $\mathbf{C}$ is a multiple of another column, then the decomposition $T = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ is unique.*

The proof follows from Jennrich's algorithm presetned earlier. We next introduce another form of decomposition.

3 Tucker decomposition

The Tucker decomposition decomposes a tensor into a so-called "core tensor" and multiple matrices which correspond to different core scalings along each mode. The model gives a summary of the information in the data, in the same way as principal components analysis does for two-way data. Therefore, the Tucker decomposition can be seen as a higher-order PCA. More formally, we have the following definition.

[1] Note that if a column of $\mathbf{C}$ is a multiple of another column, say $\mathbf{c}_j = K\mathbf{c}_i$ ($K \neq 0$), then we have not distinct entries as we have: $\langle \mathbf{c}_i, \mathbf{x} \rangle \langle \mathbf{c}_i, \mathbf{y} \rangle^{-1} = \langle K\mathbf{c}_j, \mathbf{x} \rangle \langle K\mathbf{c}_j, \mathbf{y} \rangle^{-1} = K \langle \mathbf{c}_j, \mathbf{x} \rangle K^{-1} \langle \mathbf{c}_j, \mathbf{y} \rangle^{-1} = \langle \mathbf{c}_j, \mathbf{x} \rangle \langle \mathbf{c}_j, \mathbf{y} \rangle^{-1}$.

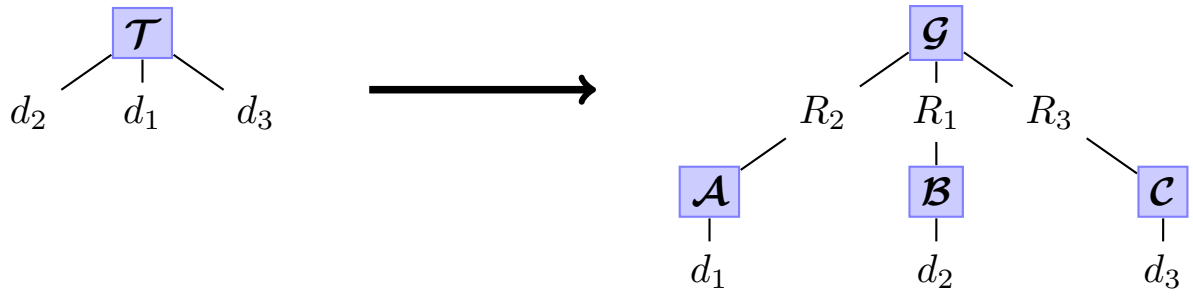


Figure 2: A diagram of Tucker decomposition

Definition 3. *Let* $\mathcal{T} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$*, then a decomposition of the form* $\mathcal{T} = \sum_{i=1}^{R_1}\sum_{j=1}^{R_2}\sum_{k=1}^{R_3} \mathcal{G}_{ijk}\ \mathbf{a}_i \circ \mathbf{b}_j \circ \mathbf{c}_k = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$ *with* $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$, $\mathbf{A} \in \mathbb{R}^{d_1 \times R_1}$, $\mathbf{B} \in \mathbb{R}^{d_2 \times R_2}$, $\mathbf{C} \in \mathbb{R}^{d_3 \times R_3}$ *with corresponding column vectors* $\mathbf{a}_i, \mathbf{b}_j, \mathbf{c}_k$*, is called a* Tucker decomposition. [a]

[a] Note that we can assume that $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ are orthogonal.

Terminology 1. *The matrices* $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ *are called* factor matrices *and the tensor* $\mathcal{G}$ *is called the* core tensor.

We note that the core tensor $\mathcal{G}$ is a 3rd-order tensor that contains the 1-mode, 2-mode and 3-mode singular values of $\mathcal{T}$.
There is a low rank approximation problem that is naturally assoicated with the Tucker decomposition. For instance, for an order 3 tensor, the Tucker decomposition is the solution to the following optimization problem:

$$\arg\min_{\hat{\mathcal{T}}} \|\hat{\mathcal{T}} - \mathcal{T}\| \text{ subject to } \hat{\mathcal{T}} = \sum_{i=1}^{R_2}\sum_{j=1}^{R_2}\sum_{k=1}^{R_3} \mathcal{G}_{ijk}\ \mathbf{a}_i \odot \mathbf{b}_j \odot \mathbf{c}_k.$$

Given a tensor decomposition, it is natural to ask what is the "smallest" such decomposition. The following definition captures this idea.

Definition 4. *The* multilinear rank *of* $\mathcal{T}$ *is the smallest* (R_1, R_2, R_3) *such that* $\mathcal{T} = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$ *with* $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$.

Contrary to the CP rank, the multilinear rank is easy to compute. The following proposition is our first clue.

Proposition 5. *The multilinear rank of* $\mathcal{T}$ *is given by* $rank(\mathcal{T}_{(i)})$ *for* $i = 1, 2, 3$.

Proof. Let $\mathcal{G}_{(i)}$ be mode-i matricization of $\mathcal{G}$. If $\mathcal{T} = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$, then $\mathcal{T}_{(1)} = \underbrace{\mathbf{A}}_{d_1 \times R_1} \underbrace{\mathcal{G}_{(1)} (\mathbf{C} \otimes \mathbf{B})^\top}_{R_1 \times d_2 d_3}$ and hence we have $\text{rank}(\mathcal{T}_{(1)}) \leq R_1$. Conversely, if $\text{rank}(\mathcal{T}_{(i)}) = R_i$ for $i = 1, 2, 3$, then let $\mathcal{T}_{(i)} = \mathbf{U}_i \mathbf{D}_i \mathbf{V}_i^\top$ be a truncated SVD of $\mathcal{T}_{(i)}$ with $\mathbf{U}_i \in \mathbb{R}^{d_i \times R_i}$. We have

$$\begin{aligned}
\mathcal{T} &= \mathcal{T} \times_1 \mathbf{U}_1\mathbf{U}_1^\top \times_2 \mathbf{U}_2\mathbf{U}_2^\top \times_3 \mathbf{U}_3\mathbf{U}_3^\top && (10)\\
&= \left(\mathcal{T} \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \mathbf{U}_3^\top\right) \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3 && (11)\\
&= \mathcal{G} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3 && (12)
\end{aligned}$$

For (10), we have $(\mathcal{T} \times_1 \mathbf{U}_1\mathbf{U}_1^\top)_{(1)} = \mathbf{U}_1\mathbf{U}_1^\top\mathcal{T}_{(1)} = \mathbf{U}_1\mathbf{U}_1^\top\mathbf{V}_1\mathbf{D}_1\mathbf{V}_1^\top = \mathbf{U}_1\mathbf{D}_1\mathbf{V}_1^\top = \mathcal{T}_{(1)}$, and similarly for the other modes. □

Note that the "converse" part of the above proof is constructive. This suggests an algorithm:

3.1 HOSVD (Higher order SVD)

The higher order singular value decomposition (HOSVD) algorithm takes as input a tensor $\mathcal{T} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ and a target rank (R_1, R_2, R_3) and returns a core tensor $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ and factor matrices $\mathbf{A} \in \mathbb{R}^{d_1 \times R_1}, \mathbf{B} \in \mathbb{R}^{d_2 \times R_2}, \mathbf{C} \in \mathbb{R}^{d_3 \times R_3}$ such that $\mathcal{T} \approx \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$.

The algorithm is simple and works as follows. We let $\mathcal{T}_{(i)}$ be the mode-i matricization of $\mathcal{T}$ and let $\mathbf{U}_i \mathbf{D} \mathbf{V}_i^\top$ be the rank R_i truncated SVD of $\mathcal{T}_{(i)}$ and then simply return $\mathcal{T} \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \mathbf{U}_3^\top$ as the core tensor $\mathcal{G}$ and $\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3$ as the factor matrices $\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3$.

Algorithm 2 HOSVD Algorithm

Input $\mathcal{T}, R_1, \ldots, R_n$
for all $i = 1, 2, \ldots, n$ **do**
  $\mathbf{U}_i = R_i$ leading left singular vectors of $\mathcal{T}_{(i)}$
end for
$\mathcal{G} \leftarrow \mathcal{T} \times_1 \mathbf{U}_1^T \times_2 \ldots \times_n \mathbf{U}_n^T$
Return $\mathcal{G}, \mathbf{U}_1 \ldots \mathbf{U}_n$

Of course, the error depends on the values of the singular values that we truncated. Even if HOSVD might not return the very best solution, we next see that it is fortunately never too far from the optimal solution.

3.2 Quasi-optimality of HOSVD

Given tensor $\mathcal{T}$, consider the problem:

$$\min_{\hat{\mathcal{T}}} \|\hat{\mathcal{T}} - \mathcal{T}\|_F \quad \text{subject to } \operatorname{rank}(\hat{\mathcal{T}}) \leq (R_1, R_2, R_3) \tag{13}$$

This problem is NP-hard. Fortunately, if we let $\tilde{\mathcal{T}}$ be the ouput of HOSVD and let $\mathcal{T}^*$ be the solution of (13), then $\|\tilde{\mathcal{T}} - \mathcal{T}\|_F \leq \sqrt{3}\|\mathcal{T}^* - \mathcal{T}\|_F$. We therefore say that HOSVD is a *quasi-optimal* algorithm for the low-rank approximation problem.
Before moving to the tensor train decomposition, we note that we can recycle algorithm 1— the alternating minimization procedure— to solve problem (13). This alternating minimization approach is termed the Higher Order Orthogonal Iteration (HOOI) algorithm. It reformulates problem (13) as

$$\min_{\mathcal{G}, \mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3} \|\mathcal{T} - \mathcal{G} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3\|_F^2$$
$$\text{subject to:} \quad \mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3} \text{ and orthogonal } \mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3.$$

It can be shown [2] that the optimal $\mathcal{G}$ is given by $\mathcal{G} = \mathcal{T} \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \mathbf{U}_3^\top$ and that it is sufficient to find $\mathbf{U}_i$ satisfying $\mathbf{U}_i^\top \mathbf{U} = \mathbf{I}$ that maximizes $\|\mathcal{G}\|_F^2$. One then solves the above problem by following the ALS algorithm, i.e., by sequentially optimizing each component while keeping the other components fixed. Let $\mathcal{Y}_{(i)}$ be the mode-i matricization of $\mathcal{Y}$.

Algorithm 3 HOOI Algorithm

Input $\mathcal{T} \in \mathbb{R}^{d_1 \times \cdots \times d_n}, R_1, \ldots, R_n$
Initialize $\mathbf{U}_1 \in \mathbb{R}^{d_1 \times R_1} \ldots \mathbf{U}_n \in \mathbb{R}^{d_n \times R_n}$ using HOSVD
repeat
  for $i = 1, 2, \ldots, n$ **do**
    $\mathcal{Y} = \mathcal{T} \times_1 \mathbf{U}_1^T \times_2 \ldots \times_{i-1} \mathbf{U}_{i-1}^T \times_{i+1} \mathbf{U}_{i+1}^T \ldots \times_n \mathbf{U}_n^T$
    $\mathbf{U}_i = R_i$ leading left singular vectors of $\mathcal{Y}_{(i)}$
  end for
until convergence
$\mathcal{G} \leftarrow \mathcal{T} \times_1 \mathbf{U}_1^T \times_2 \ldots \times_n \mathbf{U}_n^T$
Return $\mathcal{G}, \mathbf{U}_1 \ldots \mathbf{U}_n$

In contrast to the CPD, the Tucker decomposition is generally not unique. This intuitively follows from the fact that the core tensor $\mathcal{G}$ can be arbitrarily structured and might allow interactions between any component. Imposing additional constraints on the structure of $\mathcal{G}$ can therefore lead to more relaxed uniqueness properties. The HOSVD generates an all-orthogonal core tensor and hence relies on one type of special core structure.
Having covered the Tucker decomposition, the multilinear rank and algorithms (HOSVD and HOOI) for computing low rank approximations to the Tucker decomposition, we are now ready to introduce our next tensor decomposition.

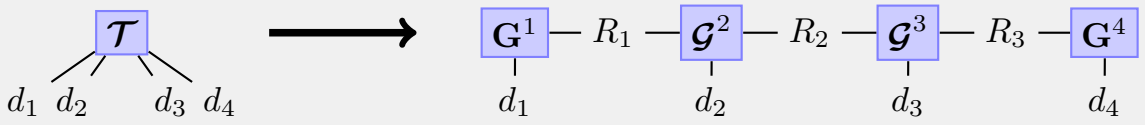
4 Tensor train decomposition (TT)

To understand why we want to introduce yet another tensor decomposition, we first look at the pros and cons between the CP decomposition and the tucker decomposition:

	CP	Tucker
# Parameters	$R\sum_{i=1}^{k} d_i$	$\sum_{i=1}^{k} R_i + \sum_{i=1}^{k} R_i d_i$
Computing the rank	NP-hard	Polynomial
Space of low-rank tensors	Not closed	Closed
Low-rank approximations	?	Quasi-optimal algorithm (HOSVD)

One can notice that, on one hand, the number of parameters scales better in the case of the CP decomposition than the Tucker decomposition. On the other hand, the Tucker decomposition has many other advantages. Perhaps we can find a new decomposition that has the best of both worlds? Next lecture we will see that the tensor train decomposition has the combined benefits of the CP and Tucker decomposition.

Definition 6. *The* tensor train decomposition *of* $\mathcal{T} \in \mathbb{R}^{d_1 \times d_2 \times d_3 \times d_4}$ *is*



where $\mathbf{G}^1 \in \mathbb{R}^{d_1 \times R_1}$, $\mathcal{G}^2 \in \mathbb{R}^{R_1 \times d_2 \times R_2}$, $\mathcal{G}^3 \in \mathbb{R}^{R_2 \times d_3 \times R_3}$ *and* $\mathbf{G}^4 \in \mathbb{R}^{R_3 \times d_4}$.

A note on notation: we write $\mathcal{T} = \langle\!\langle \mathcal{G}^1, \mathcal{G}^2, \mathcal{G}^3, \mathcal{G}^4 \rangle\!\rangle$ as a shortcut for the train

$$\mathcal{T}_{i_1,i_2,i_3,i_4} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \mathcal{G}^1_{i_1,r_1} \mathcal{G}^2_{r_1,i_2,r_3} \mathcal{G}^3_{r_2,i_3,r_3} \mathcal{G}^4_{r_3,i_4}.$$

We conclude by introducing the rank associated to the tensor train decomposition.

Definition 7 (TT-rank decomposition)**.** *The* TT-rank *decomposition of* $\mathcal{T}$ *is the smallest* (R_1, R_2, R_3) *such that* $\mathcal{T} = \langle\!\langle \mathcal{G}^1, \mathcal{G}^2, \mathcal{G}^3, \mathcal{G}^4 \rangle\!\rangle$ *is a TT decomposition of size* (R_1, R_2, R_3).

References

[1] J. D. Carroll and J.-J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition. *Psychometrika*, 35(3):283–319, 1970.

[2] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[3] R. A. Harshman et al. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

[4] S. Rabanser and Gunnemann. Introduction to tensor decompositions and their applications in machine learning. 2017.