

# Mathematical methods of forecasting

Intelligent systems, Phystech

**2022**

Рассмотрим квадратичный алгоритм решения этой задачи. Найдём последовательно векторы  $\mathbf{u}_k, \mathbf{v}_k$  и сингулярные числа  $\lambda_k$  для  $k = 1, \dots, r$ . В качестве этих векторов берутся нормированные значения векторов  $\mathbf{a}_k$  и  $\mathbf{b}_k$ , соответственно

$$\mathbf{u}_k = \frac{\mathbf{a}_k}{\|\mathbf{a}_k\|} \quad \text{и} \quad \mathbf{v}_k = \frac{\mathbf{b}_k}{\|\mathbf{b}_k\|}.$$

Векторы  $\mathbf{a}_k$  и  $\mathbf{b}_k$  находятся как пределы последовательностей векторов  $\{\mathbf{a}_{k_s}\}$  и  $\{\mathbf{b}_{k_s}\}$ , соответственно

$$\mathbf{a}_k = \lim_{s \rightarrow \infty} (\mathbf{a}_{k_s}) \quad \text{и} \quad \mathbf{b}_k = \lim_{s \rightarrow \infty} (\mathbf{b}_{k_s}).$$

Сингулярное число  $\lambda_k$  находится как произведение норм векторов

$$\lambda_k = \|\mathbf{a}_k\| \cdot \|\mathbf{b}_k\|.$$

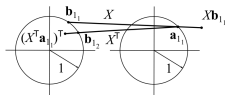


Рис. 13. Итеративная процедура оценивания сингулярных векторов.

Процедура нахождения последовательностей векторов  $\mathbf{a}_{k_s}, \mathbf{b}_{k_s}$ ,  $\mathbf{u}_k, \mathbf{v}_k$  начинается с выбора наибольшей по норме строки  $\mathbf{b}_{1_1}$  матрицы  $\mathbf{X}$ . Для  $k = 1$  формулы нахождения векторов  $\mathbf{a}_{1_s}, \mathbf{b}_{1_s}$  имеют вид:

$$\mathbf{a}_{1_s} = \frac{\mathbf{X} \mathbf{b}_{1_s}}{\|\mathbf{b}_{1_s}\|}, \quad \mathbf{b}_{1_{s+1}} = \frac{\mathbf{a}_{1_s}^T \mathbf{X}}{\|\mathbf{a}_{1_s}^T \mathbf{X}\|}, \quad s = 1, 2, \dots$$

Для вычисления векторов  $\mathbf{u}_k, \mathbf{v}_k$  при  $k = 2, \dots, r$  используется вышеприведенная формула, с той разницей, что матрица  $\mathbf{X}$  заменяется на скорректированную на  $k$ -м шаге матрицу  $\mathbf{X}_{k+1} = \mathbf{X}_k - \mathbf{u}_k \lambda_k \mathbf{v}_k$ . На рисунке ?? показаны две итерации,  $s = 1, 2$ , первого шага  $k = 1$  упрощенной процедуры нахождения сингулярного разложения.

# Tensor decomposition syllabus

- ① Least squares
- ② Singular value decomposition<sup>1</sup> and principal component analysis<sup>2</sup>
- ③ Tensor rank decomposition or canonical polyadic decomposition<sup>3</sup>
- ④ Alternating least squares<sup>4</sup>
- ⑤ Tucker decomposition<sup>5</sup>
- ⑥ Higher-order singular value decomposition<sup>6</sup>

---

<sup>1</sup>George E. Forsythe and Cleve B. Moler (1967) Computer solution of linear algebraic systems

<sup>2</sup>I. T. Jolliffe (1986) Principal Component Analysis

<sup>3</sup>Tamara G. Kolda (2009) Tensor Decompositions and Applications // SIAM Review

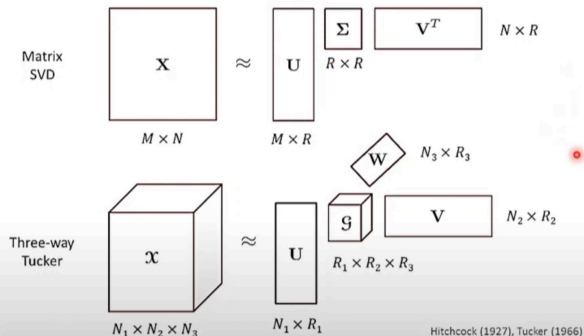
<sup>4</sup>Trevor Hastie et al. (2015) Matrix Completion and Low-Rank SVD via Fast Alternating Least Squares// JMLR

<sup>5</sup>L. R. Tucker (1966) Some mathematical notes on three-mode factor analysis // Psychometrika

<sup>6</sup>A. Cichocki et al. (2014) Tensor Decompositions for Signal Processing Applications From Two-way to Multiway Component Analysis // ArXiv



# Tucker Decomposition




Decompose tensor  $\rightarrow$

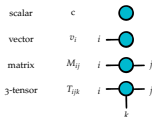
- Core tensor ( $\mathcal{G}$ )
- Factor matrices ( $U, V, W$ )

Tucker Compression: Extends the Matrix SVD to Multiway Arrays. Retrieved from Sandia National Laboratories

## Penrose tensor diagram notation

- ① Roger Penrose (1971) Applications of negative dimensional tensors // Combinatorial Mathematics and its Application
- ② Tai-Danae Bradley (2020) At the Interface of Algebra and Statistics // ArXiv

simply write  as representing the linear map itself. More generally, we have the following diagrams.

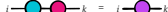


In this graphical notation, familiar notions have elegant pictures. Here is a brief showcase.

- Composition is tensor contraction.** Tensors can be composed along dimensions of matching indices, and tensor contraction corresponds to summing along this common index. Graphically, this corresponds to joining the corresponding edges between diagrams. For example, the product of two matrices



is illustrated by “gluing” the two edges labeled  $j$  and then fusing the two nodes into a single node.

$$\sum_j M_{ij} N_{jk} = (MN)_{ik}$$


The resulting diagram has two free indices,  $i$  and  $k$ , which indeed specify a new matrix. As another example, the product of a matrix  $M$  with a vector  $|v\rangle$  results in another vector  $M|v\rangle$ , which is a node with one free edge.

$$M |v\rangle = M|v\rangle$$


To keep the picture clean, we’ve now dropped the indices. More generally, the composition of two or more tensors is represented by a cluster of nodes and edges where the contractions occur along edges with matching indices.

Though these pictures might be new to some, I suspect the idea is familiar to all. When teaching students about functions, for instance, one often says, “A function is like a machine. You feed it an input, the machine does its job, and then it spits out the output.” The accompanying picture is something like this:





a function is a machine

This is an example of a tensor diagram, though we’re now interested in linear functions. Think of the node as the linear mapping and think of the edges as the input (domain) and output (codomain) vector spaces. I prefer to draw the nodes as circles, rather than squares, though it doesn’t matter.

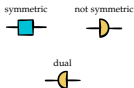


a matrix is a node



The diagram here illustrates another important point. There is great flexibility in how one chooses to orient the diagrams spatially. A vector, for instance, is characterized by the fact that it is *one node with one edge*. We will not imbue additional meaning to whether the edge is horizontal or vertical or otherwise. For example we take both  and  to represent the same vector.

2. **The shape of a node may convey additional meaning.** There is flexibility in the shapes used for nodes, as convention varies across the literature. This allows for creativity in how information can be conveyed through a diagram. When working with 2-tensors, for instance, we may wish to use a symmetric shape for symmetric matrices only. Then the dual mapping can be represented by reflecting its diagram,



so that the symmetry is preserved in the notation.



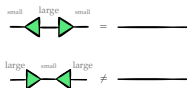
Another useful choice is to represent isometric embeddings as triangles:



An **isometric embedding**  $U$  is a linear map from a space  $V$  to a space  $W$  of larger dimension that preserves the lengths of vectors. Such a map satisfies  $U^t U = \text{id}_V$ , but  $U U^t \neq \text{id}_W$ . In words, projection of the large space  $W$  onto the embedded image  $U V \subseteq W$  won't distort the vectors in  $V$ . This operation is the identity on



$V$ . On the other hand, compressing  $W$  onto  $V$  necessarily loses information, so to speak. The asymmetry of the triangle serves as a visual reminder of this: the base ( $W$ ) is larger than its tip ( $V$ ).



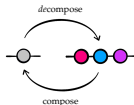
When  $W = V$  and when  $U$  satisfies both equalities  $UU^\dagger = U^\dagger U = \text{id}_V$ , then it is called a **unitary operator**. This illustrates another useful convention: the identity mapping is often represented as an edge with no node. Indeed, contraction with an identity leaves a tensor and its corresponding diagram unchanged.



3. **Tensor decomposition is node decomposition.** The flexibility in choosing different node shapes provides useful pictures for tensor decomposition. For example, the singular value decomposition of a matrix  $M = VDU^\dagger$  (see Section 2.5) can be illustrated as:

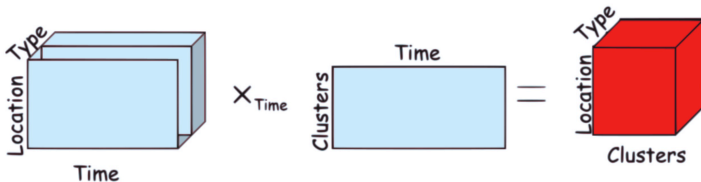


Here,  $U$  and  $V$  are unitary operators, hence isometries and hence triangles, while  $D$  is a diagonal operator drawn as a circle. More generally, tensor decomposition is the decomposition of one node into multiple nodes, while tensor composition is the fusion of multiple nodes into a single node.



The mode- $n$  product is the multiplication of a tensor by a matrix along the  $n^{th}$  mode of a tensor. This essentially means that each mode- $n$  fiber should be multiplied by this matrix. Mathematically, this is expressed as:

$$\underline{\mathbf{X}} \times_n \mathbf{A} = \underline{\mathbf{Y}} \Leftrightarrow \mathbf{Y}_{(n)} = \mathbf{A} \mathbf{X}_{(n)}$$



Important properties of the mode- $n$  product:

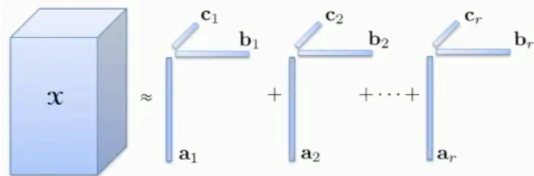
1. For distinct modes in a series of multiplications, the order of the multiplication is irrelevant:

$$\underline{\mathbf{X}} \times_n \mathbf{A} \times_m \mathbf{B} = \underline{\mathbf{X}} \times_m \mathbf{B} \times_n \mathbf{A} \quad (m \neq n)$$

- However, it does not hold if the modes are the same :

$$\underline{\mathbf{X}} \times_n \mathbf{A} \times_n \mathbf{B} = \underline{\mathbf{X}} \times_n (\mathbf{BA})$$

# Alternating least squares to fit CP



$$\min_{\mathbf{A}} \sum_{ijk} \left( x_{ijk} - \sum_{\ell} a_{i\ell} b_{j\ell} c_{k\ell} \right)^2$$

$$\min_{\mathbf{A}, \mathbf{B}, \mathbf{C}} \|\mathbf{X} - \mathbf{M}\|^2 \text{ s.t. } \mathbf{M} = [\mathbf{A}, \mathbf{B}, \mathbf{C}]$$



$$\min_{\mathbf{A}, \mathbf{B}, \mathbf{C}} \sum_{ijk} \left( x_{ijk} - \sum_{\ell} a_{i\ell} b_{j\ell} c_{k\ell} \right)^2$$

# Outer and Kronecker product

## (Cartesian, Tensor, Hadamard)

The outer product and Kronecker product are closely related; in fact the same symbol is commonly used to denote both operations.

If  $u = [1 \ 2 \ 3]^T$  and  $v = [4 \ 5]^T$ , we have:

$$u \otimes_{\text{Kron}} v = \begin{bmatrix} 4 \\ 5 \\ 8 \\ 10 \\ 12 \\ 15 \end{bmatrix}, \quad u \otimes_{\text{outer}} v = \begin{bmatrix} 4 & 5 \\ 8 & 10 \\ 12 & 15 \end{bmatrix}.$$

In the case of column vectors, the Kronecker product can be viewed as a form of vectorization of the outer product. In particular, for two column vector  $u$  and  $v$ , we can write:

$$u \otimes_{\text{Kron}} v = \text{vec}(v \otimes_{\text{outer}} u)$$

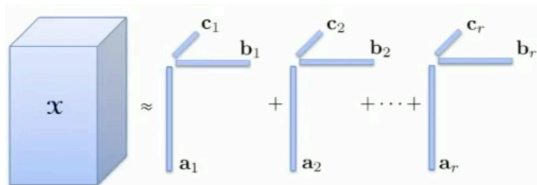
Note that the order of the vectors is reversed in the right side of the equation.

Another similar identity that further highlights the similarity between the operations is

$$u \otimes_{\text{Kron}} v^T = uv^T = u \otimes_{\text{outer}} v$$

where the order of vectors needs not be flipped. The middle expression uses matrix multiplication, where the vectors are considered as column-row matrices.

# Alternating least squares, optimization problem



$$\min_{\mathbf{A}, \mathbf{B}, \mathbf{C}} \|\mathbf{X} - \mathbf{M}\|^2 \text{ s.t. } \mathbf{M} = [\mathbf{A}, \mathbf{B}, \mathbf{C}]$$



$$\min_{\mathbf{A}, \mathbf{B}, \mathbf{C}} \sum_{ijk} \left( x_{ijk} - \sum_{\ell} a_{i\ell} b_{j\ell} c_{k\ell} \right)^2$$

Repeat until convergence:

$$\text{Step 1: } \min_{\mathbf{A}} \sum_{ijk} \left( x_{ijk} - \sum_{\ell} a_{i\ell} b_{j\ell} c_{k\ell} \right)^2$$

$$\text{Step 2: } \min_{\mathbf{B}} \sum_{ijk} \left( x_{ijk} - \sum_{\ell} a_{i\ell} b_{j\ell} c_{k\ell} \right)^2$$

$$\text{Step 3: } \min_{\mathbf{C}} \sum_{ijk} \left( x_{ijk} - \sum_{\ell} a_{i\ell} b_{j\ell} c_{k\ell} \right)^2$$

Nonconvex problem with convex subproblems.

# Least squares problem, solution

$$\min_{\mathbf{A}} \sum_{ijk} \left( x_{ijk} - \sum_{\ell} a_{i\ell} b_{j\ell} c_{k\ell} \right)^2 \quad \longrightarrow \quad \min_{\mathbf{A}} \|\mathbf{X}_{(1)} - \mathbf{A}(\mathbf{C} \odot \mathbf{B})'\|_F^2$$

"right hand sides"

$\mathbf{X}_{(1)}$

Matrix Unfolding

3-way case  $n \times n^2$   
 d-way case  $n \times n^{d-1}$



$\mathbf{A}$

$n \times r$   
 $n \times r$

"matrix"

$(c_1 \otimes b_1)'$   
 $\vdots$   
 $(c_r \otimes b_r)'$

Khatri-Rao Product

$(\mathbf{C} \odot \mathbf{B})'$

$r \times n^2$   
 $r \times n^{d-1}$

Short & Very Wide Matrix

In mathematics, the **Khatri–Rao product** is defined as<sup>[1][2]</sup>

$$\mathbf{A} * \mathbf{B} = (\mathbf{A}_{ij} \otimes \mathbf{B}_{ij})_{ij}$$

in which the  $ij$ -th block is the  $m_i p_i \times n_j q_j$  sized **Kronecker product** of the corresponding blocks of  $\mathbf{A}$  and  $\mathbf{B}$ , assuming the number of row and column partitions of both **matrices** is equal. The size of the product is then  $(\sum_i m_i p_i) \times (\sum_j n_j q_j)$ .

For example, if  $\mathbf{A}$  and  $\mathbf{B}$  both are  $2 \times 2$  partitioned matrices e.g.:

$$\mathbf{A} = \left[ \begin{array}{c|c} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \hline \mathbf{A}_{21} & \mathbf{A}_{22} \end{array} \right] = \left[ \begin{array}{cc|c} 1 & 2 & 3 \\ 4 & 5 & 6 \\ \hline 7 & 8 & 9 \end{array} \right], \quad \mathbf{B} = \left[ \begin{array}{c|c} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \hline \mathbf{B}_{21} & \mathbf{B}_{22} \end{array} \right] = \left[ \begin{array}{c|cc} 1 & 4 & 7 \\ \hline 2 & 5 & 8 \\ 3 & 6 & 9 \end{array} \right],$$

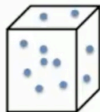
we obtain:

$$\mathbf{A} * \mathbf{B} = \left[ \begin{array}{cc|cc} \mathbf{A}_{11} \otimes \mathbf{B}_{11} & \mathbf{A}_{12} \otimes \mathbf{B}_{12} \\ \hline \mathbf{A}_{21} \otimes \mathbf{B}_{21} & \mathbf{A}_{22} \otimes \mathbf{B}_{22} \end{array} \right] = \left[ \begin{array}{cc|cc} 1 & 2 & 12 & 21 \\ 4 & 5 & 24 & 42 \\ \hline 14 & 16 & 45 & 72 \\ 21 & 24 & 54 & 81 \end{array} \right].$$

This is a submatrix of the **Tracy–Singh product** of the two matrices (each partition in this example is a partition in a corner of the **Tracy–Singh product**) and also may be called the block Kronecker product.

# Least squares problem, randomized convergence estimation

$$F(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \sum_{ijk} \left( x_{ijk} - \sum_{\ell} a_{i\ell} b_{j\ell} c_{k\ell} \right)^2$$

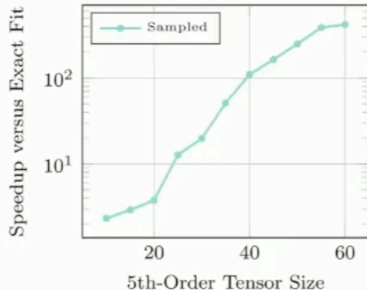


Estimate convergence of function values using small random subset of elements in function evaluation (use Chernoff-Hoeffding to bound accuracy)

$$\hat{F}(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \omega \sum_{ijk \in \Omega} \left( x_{ijk} - \sum_{\ell} a_{i\ell} b_{j\ell} c_{k\ell} \right)^2$$

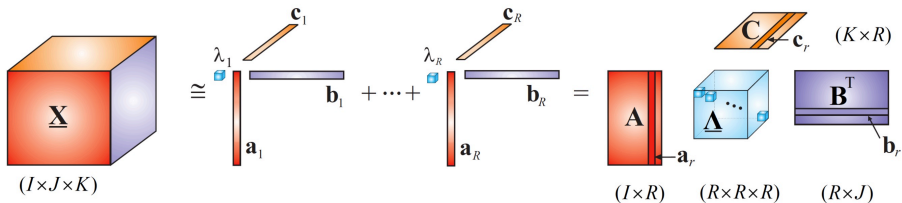
16000 samples < 1% of full data

$$\frac{|F - \hat{F}|}{|F|} < 10^{-3}$$





# Canonical Polyadic Decomposition (CPD)



## Theoretical background

The **Canonical Polyadic Decomposition (CPD)** (also referred to as PARAFAC or CANDECOMP) is an algorithm that factorizes an 3-rd order tensor  $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$  into a linear combination of terms  $\underline{\mathbf{X}}_r = \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$ , which are rank-1 tensors. In other words the tensor  $\underline{\mathbf{X}}$  is decomposed as

$$\begin{aligned} \underline{\mathbf{X}} &\approx \sum_{r=1}^R \lambda_r \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \\ &= \underline{\Delta} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} \quad (1) \\ &= [\underline{\Delta}; \mathbf{A}, \mathbf{B}, \mathbf{C}] \end{aligned}$$

where

- $\underline{\Delta}$  is an 3-rd order core tensor having  $\lambda_r$  as entries in positions  $\underline{\Delta}[i, j, k]$ , where  $i = j = k$ , and zeroes elsewhere
- $\mathbf{A}, \mathbf{B}, \mathbf{C}$  are factor matrix obtained as the concatenation of the corresponding factor vectors, i.e  $\mathbf{A} = [\mathbf{a}_1 \mathbf{a}_2 \dots \mathbf{a}_R]$

Assuming the kruskal rank is fixed, there are many algorithms to compute a CPD. The most popular approach is via the alternating least squares (ALS) method. The goal is to find such CP representation  $[\underline{\Delta}; \mathbf{A}, \mathbf{B}, \mathbf{C}]$  which provides the best approximation of the original tensor  $\underline{\mathbf{X}}$ :

$$\min \|\underline{\mathbf{X}} - [\underline{\Delta}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\|_F^2$$

The alternating least squares approach fixes  $\mathbf{B}$  and  $\mathbf{C}$  to solve for  $\mathbf{A}$ , then fixes  $\mathbf{A}$  and  $\mathbf{C}$  to solve for  $\mathbf{B}$ , then fixes  $\mathbf{A}$  and  $\mathbf{B}$  to solve for  $\mathbf{C}$ , and continues to repeat the entire procedure until some convergence criterion is satisfied.

## Higher Order Singular Value Decomposition (HOSVD)

Consider an 3-rd order tensor  $\underline{\mathbf{X}} \in \mathbb{R}^I \times J \times K$ , decomposed in the Tucker format as

$$\underline{\mathbf{X}} = \underline{\mathbf{G}} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$$

The HOSVD is a special case of the Tucker decomposition, in which all the factor matrices are constrained to be orthogonal. They are computed as truncated version of the left singular matrices of all possible mode- $n$  unfoldings of tensor  $\underline{\mathbf{X}}$ :

$$\mathbf{X}_{(1)} = \mathbf{U}_1 \mathbf{\Sigma}_1 \mathbf{V}_1^T \rightarrow \mathbf{A} = \mathbf{U}_1[1 : R_1]$$

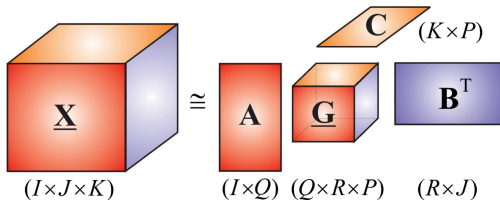
$$\mathbf{X}_{(2)} = \mathbf{U}_2 \mathbf{\Sigma}_2 \mathbf{V}_2^T \rightarrow \mathbf{B} = \mathbf{U}_2[1 : R_2]$$

$$\mathbf{X}_{(3)} = \mathbf{U}_3 \mathbf{\Sigma}_3 \mathbf{V}_3^T \rightarrow \mathbf{C} = \mathbf{U}_3[1 : R_3]$$

For a general order- $N$  tensor, the  $N$ -tuple  $(R_1, \dots, R_N)$  is called the **multi-linear rank** and provides flexibility in compression and approximation of the original tensor. For our order-3 tensor in the multilinear rank is therefore  $(R_1, R_2, R_3)$ . After factor matrices are obtained, the core tensor  $\underline{\mathbf{G}}$  is computed as

$$\underline{\mathbf{G}} = \underline{\mathbf{X}} \times_1 \mathbf{A}^T \times_2 \mathbf{B}^T \times_3 \mathbf{C}^T$$

## Tucker Decomposition

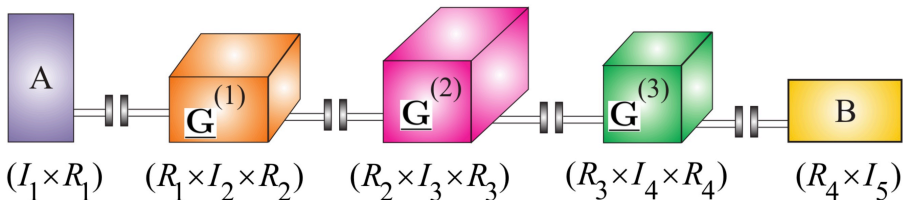


**Tucker Decomposition** represents a given tensor  $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$  in the form of a dense core tensor  $\underline{\mathbf{G}}$  with multi-linear rank  $(Q, R, P)$  and a set of factor matrices  $\mathbf{A} \in \mathbb{R}^{I \times Q}$ ,  $\mathbf{B} \in \mathbb{R}^{J \times R}$  and  $\mathbf{C} \in \mathbb{R}^{K \times P}$  as illustrated above. In other words, the tensor  $\underline{\mathbf{X}}$  can be represented in Tucker form as

$$\begin{aligned}
 \underline{\mathbf{X}} &\simeq \sum_{q=1}^Q \sum_{r=1}^R \sum_{p=1}^P g_{qrp} \mathbf{a}_q \circ \mathbf{b}_r \circ \mathbf{c}_p \\
 &= \underline{\mathbf{G}} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} \\
 &= [\underline{\mathbf{G}}; \mathbf{A}, \mathbf{B}, \mathbf{C}]
 \end{aligned} \tag{1}$$

In practice, there exist several algorithms to represent a given tensor in the Tucker format. The two most used ones are Higher Order Singular Value Decomposition (HOSVD), and Higher Order Orthogonal Iteration (HOOI), which are implemented through the `HOSVD` and `HOOI` classes respectively.

## Tensor Train Decomposition via SVD



## Theoretical background

**Tensor train decomposition** represents a given tensor as a set of sparsely interconnected lower-order tensors and factor matrices. Mathematically speaking, the obtained TT representation of an  $N$ -th order tensor  $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$  can be expressed as a TT as

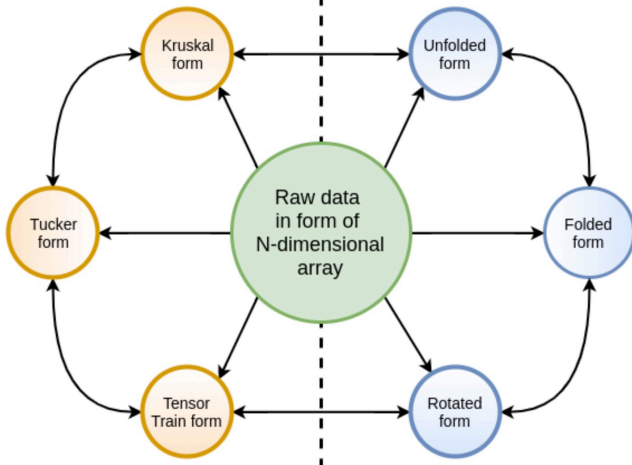
$$\begin{aligned} \underline{\mathbf{X}} &= [\mathbf{A}, \underline{\mathbf{G}}^{(1)}, \underline{\mathbf{G}}^{(2)}, \dots, \underline{\mathbf{G}}^{(N-1)}, \mathbf{B}] \\ &= \mathbf{A} \times_2^1 \underline{\mathbf{G}}^{(1)} \times_3^1 \underline{\mathbf{G}}^{(2)} \times_3^1 \dots \times_3^1 \underline{\mathbf{G}}^{(N-1)} \times_3^1 \mathbf{B} \end{aligned}$$

Each element of a TT is generally referred to as a **tt-core** with sizes of its dimensions:  $\mathbf{A} \in \mathbb{R}^{I_1 \times R_1}$ ,  $\mathbf{B} \in \mathbb{R}^{R_{N-1} \times I_N}$ ,  $\underline{\mathbf{G}}^{(n)} \in \mathbb{R}^{R_n \times I_{n+1} \times R_{n+1}}$

The TTSVD algorithm involves iteratively performing a series of foldings and unfoldings on an original tensor  $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$  in conjunction with SVD. At every iteration a core  $\underline{\mathbf{G}}^{(n)} \in \mathbb{R}^{R_n \times I_{n+1} \times R_{n+1}}$  is computed, where the TT-rank  $(R_1, R_2, \dots, R_N)$  has been specified a priori.

Numerical  
methods

Data  
rearrangement



# Tucker decomposition example 1/2

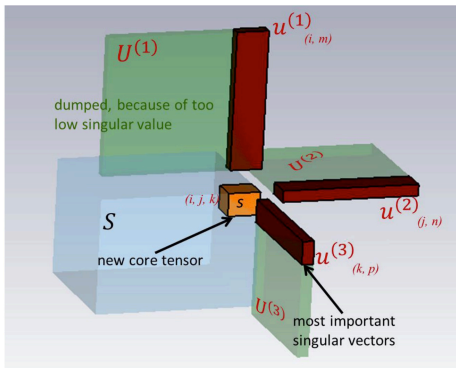


FIG. 3. Illustration of trimming a 3rd-order tensor through SVD. Light green blocks represent the sets of bases  $U^{(n)}$  and the blue block represents the initial core tensor  $S$ , i.e., the projections of the original data including noise onto these bases. Brown blocks represent the reduced bases  $u^{(1)} \in \mathbb{R}^{i \times m}$ ,  $u^{(2)} \in \mathbb{R}^{j \times n}$ ,  $u^{(3)} \in \mathbb{R}^{k \times p}$ . The core tensor  $S \in \mathbb{R}^{m \times n \times p}$  is reduced to  $s \in \mathbb{R}^{i \times j \times k}$  ( $i < m, j < n, p < k$ ) (orange block).

# Tucker decomposition example 2/2

## A. Comparisons to analytical solution

The analytic expressions for the electric field components inside a cubic cavity are

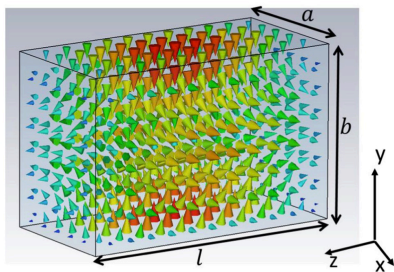


FIG. 7. Simulated electric field map of the  $TM_{111}$ -mode inside a cubic cavity. Colors correspond to absolute field values.

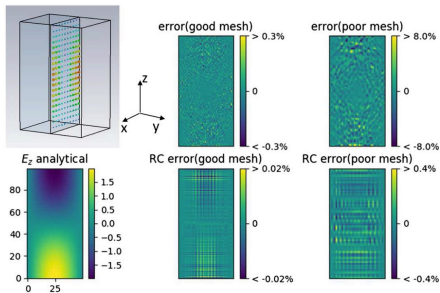
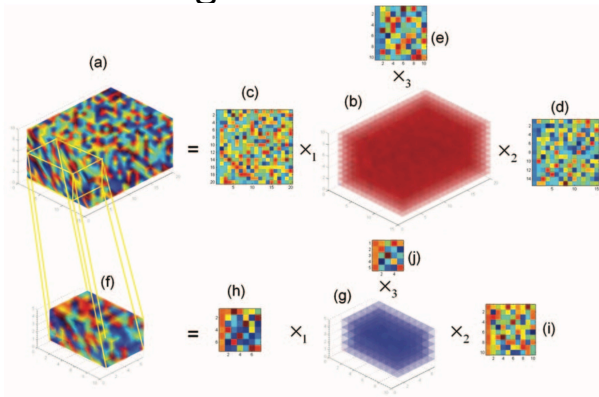


FIG. 8. Upper left: definition of the cutting plane within the cubic cavity. Lower left: analytic  $E_z$  at the cutting plane. Upper center: relative difference of  $E_z$  from analytical solution and from simulations with fine mesh. Upper right: relative difference of  $E_z$  from analytical solution and from simulations with rough mesh. Lower center: relative difference of  $E_z$  from analytical solution and from HOSVD starting from fine mesh simulation. Lower right: relative difference of  $E_z$  from analytical solution and from HOSVD starting from rough mesh simulations.

# HOSVD downdating



**Fig. 2.** Tensor HOSVD Downdating: (a) Tensor  $A$  of size  $20 \times 15 \times 10$  (b) Core tensor  $S$  of  $A$ . (c) Basis Matrix  $U_1$  (d) Basis Matrix  $U_2$  (e) Basis Matrix  $U_3$  (f) Tensor  $A^*$  extracted from  $A$ , of size  $7 \times 10 \times 5$ . (g) Core tensor  $S^*$  of  $A^*$  (h) Basis Matrix  $U_1^*$  (i) Basis Matrix  $U_2^*$  (j) Basis Matrix  $U_3^*$

Dan Schonfeld (2009). Dynamic updating and downdating matrix SVD and tensor HOSVD for adaptive indexing and retrieval of motion trajectories



# References on tensor decomposition

## Papers

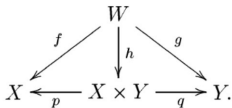
- ① A. Cichocki et al. (2014) Tensor Decompositions for Signal Processing Applications From Two-way to Multiway Component Analysis // ArXiv
- ② L. R. Tucker (1966) Some mathematical notes on three-mode factor analysis, Psychometrika
- ③ L. D. Lathauwer et al. (2000) A multilinear singular value decomposition, SIAM J. Matrix Analysis and Applications
- ④ N. Vannieuwenhoven et al. (2012) A New Truncation Strategy for the Higher-Order Singular Value Decomposition // SIAM J. Scientific Computing
- ⑤ Tamara G. Kolda and Brett W. Bader (2009) Tensor decompositions and applications // SIAM review
- ⑥ Ivan V. Oseledets (2011) Tensor-train decomposition // SIAM Journal on Scientific Computing

## Code

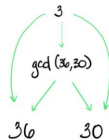
- ① [hottbox.github.io](https://github.com/hottbox) Higher Order Tensors ToolBOX
- ② [tensorly.org](https://tensorly.org) Fast and Simple Tensor Learning In Python
- ③ [tensortoolbox.org](https://tensortoolbox.org) Tensor Toolbox for MATLAB

# Cartesian product as an object in category theory

Рассмотрим декартово произведение  $X \times Y$  двух множеств, состоящее, как обычно, из всех упорядоченных пар  $\langle x, y \rangle$  элементов  $x \in X$  и  $y \in Y$ . Проекции произведения  $\langle x, y \rangle \mapsto x$ ,  $\langle x, y \rangle \mapsto y$  на его оси  $X$  и  $Y$  представляют собой функции  $p: X \times Y \rightarrow X$ ,  $q: X \times Y \rightarrow Y$ . Любая функция  $h: W \rightarrow X \times Y$  из третьего множества  $W$  однозначно определяется композициями  $p \circ h$  и  $q \circ h$ . Обратно, если дано множество  $W$  и функции  $f$  и  $g$ , такие, как на последующей диаграмме, то существует единственная функция  $h$ , которая делает диаграмму коммутативной; а именно,  $hw = \langle fw, gw \rangle$  для каждого  $w \in W$ :



Таким образом, для данных  $X$  и  $Y$  функция  $\langle p, q \rangle$  универсальна среди всех пар функций, отображающих некоторое множество в  $X$  и в  $Y$ , поскольку любая другая такая пара  $\langle f, g \rangle$  однозначно пропускается (посредством  $h$ ) через пару  $\langle p, q \rangle$ . Это свойство определяет декартово произведение единственным образом (с точностью до биекции):



An arrow  $n \rightarrow m$  means " $n$  evenly divides  $m$ ."  
In category theory,  $\text{gcd}(n, m)$  is the product of  $n$  and  $m$ .

An element of the form  $v \otimes w$  is called the **tensor product** of  $v$  and  $w$ . An element of  $V \otimes W$  is a **tensor**, and the tensor product of two vectors is sometimes called an *elementary tensor* or a *decomposable tensor*. The elementary tensors span  $V \otimes W$  in the sense that every element of  $V \otimes W$  is a sum of elementary tensors. If **bases** are given for  $V$  and  $W$ , a basis of  $V \otimes W$  is formed by all tensor products of a basis element of  $V$  and a basis element of  $W$ .

The tensor product of two vector spaces captures the properties of all bilinear maps in the sense that a bilinear map from  $V \times W$  into another vector space  $Z$  factors uniquely through a **linear map**  $V \otimes W \rightarrow Z$  (see **Universal property**).

$$\begin{array}{ccc}
 V \times W & \xrightarrow{\varphi} & V \otimes W \\
 & \searrow h & \downarrow \tilde{h} \\
 & & Z
 \end{array}$$

Universal property of tensor product: if  $h$  is bilinear, there is a unique linear map  $\tilde{h}$  that makes the diagram **commutative** (that is,  $h = \tilde{h} \circ \varphi$ ). □

