Tensor Decomposition Notes

Longqian Huang

February 2021

1 Introduction

This notes is a summary about various *tensor decompositions*, where matrix is regarded as a special case of tensor.

2 Singular Value Decomposition

The Singular Value Decomposition (SVD) is a very useful tool in matrix analysis.

Theorem 1 (SVD)**.** For $\mathbf{A} \in \mathbb{C}^{m\times n}$, there exist unitary matrices $\mathbf{U} \in \mathbb{C}^{m\times m}$, $\mathbf{V} \in \mathbb{C}^{n\times n}$, and *uniquely* defined nonnegative numbers $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_{\min m,n} \geq 0$, called *singular values* of $\mathbf{A}$, such that

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\dagger, \quad \Sigma = \mathrm{diag}[\sigma_1, \ldots, \sigma_{\min m,n}] \in \mathbb{R}^{m\times n}.$$

Proof can be found in Appendix 2 of [1].

Remark. Define *operator norm* $\|\mathbf{A}\|_{p\to q} = \sup_{\|\mathbf{x}\|_p=1} \|\mathbf{A}\mathbf{x}\|_q$, we have

$$\sigma_{\max}(\mathbf{A}) = \sigma_1 = \|A\|_{2\to 2} = \max_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2$$
$$\sigma_{\min(\mathbf{A})} = \min_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2$$

Theorem 2 (reduced SVD)**.** For $\mathbf{A}$ of rank r with full SVD $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\dagger$, consider $\tilde{\Sigma} = \mathrm{diag}[\sigma_1, \ldots, \sigma_r] \in \mathbb{R}^{r\times r}$ and the submatrices $\tilde{\mathbf{U}} = [\mathbf{u_1}|\cdots|\mathbf{u_r}] \in \mathbb{C}^{m\times r}$ of $\mathbf{U}$,$\tilde{\mathbf{V}} = [\mathbf{v_1}|\cdots|\mathbf{v_r}] \in \mathbb{C}^{n\times r}$ of $\mathbf{V}$.We have

$$\mathbf{A} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^\dagger = \sum_{j=1}^{r} \sigma_j \mathbf{u_j}\mathbf{v_j}^\dagger.$$

Remark (truncated SVD)**.** Assume $\mathrm{rank}(\mathbf{A}) = r, 0 < k < r$, we have truncated SVD

$$\mathbf{A} \approx \mathbf{U_k}\mathbf{\Sigma_k}\mathbf{V_k}^\dagger,$$

where $\mathbf{U_k} \in \mathbb{C}^{m\times k}$ is from $\mathbf{U}$, $\mathbf{V_k} \in \mathbb{C}^{n\times k}$ from $\mathbf{V}$, $\mathbf{\Sigma_k} \in \mathbb{C}^{k\times k}$ from $\mathbf{\Sigma}$.

Remark (truncated SVD in PCA)**.** In Principle Component Analysis (PCA), SVD can be applied to obtain principle components.[2]

Let $X \in \mathbb{R}^{m\times n}$ be a n-sample matrix with column mean value $E(X_j) = 0$. Do truncated SVD with X , we have

$$X = U\Sigma V^\dagger = U\Sigma V^T,$$

where $U \in \mathbb{R}^{m\times k}, \Sigma \in \mathbb{R}^{k\times k}, V \in \mathbb{R}^{n\times k}$.

Since principle component matrix is a linear transform of X whose column is the eigenvector of sample covarience matrix $S_X = X^T X$, (ignore the coefficient $\frac{1}{n-1}$) we have

$$X^T X = V\Sigma^T \Sigma V^T \implies X^T X V^T = \Sigma^2 V^T,$$

hence V^T is the exact transform we want and the sample principle matrix is

$$Y = V^T X \in \mathbb{R}^{k\times n}.$$

Theorem 3 (best rank-k approximation)**.** A best rank-k approximation is given by the leading k-factors of the SVD. Let R be the rank of a matrix $\mathbf{A}$ and assume its SVD is given by

$$\mathbf{A} = \sum_{r=1}^{R} \sigma_r \mathbf{u_r} \times \mathbf{v_r} \quad \text{with } \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_R > 0.$$

Then a rank-k approximation that minimizes $\|\mathbf{A} - \mathbf{B}\|$ is given by

$$\mathbf{B} = \sum_{r=1}^{k} \sigma_r \mathbf{u_r} \times \mathbf{v_r}.$$

Proof can be done using Lagrange Multiplier Method, see [3] for details, or [2] for a Chinese version in Chapter 15.3.

3 Tensor Basics and Notations

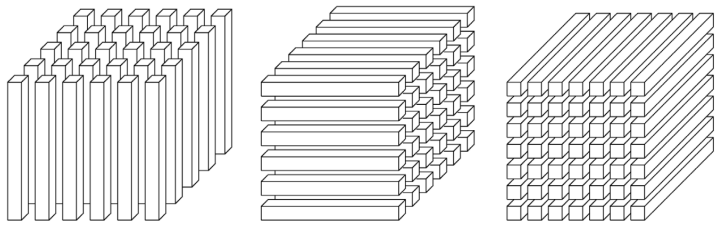
A *tensor* is a multidimensional array.An N-way or Nth-order tensor is an element of the tensor product of N vector spaces.[4]

3.1 Mode and Unfolding

Definition 1 (mode)**.** The *order* of a tensor is the number of dimensions, also known as *ways* or *modes*.

See Figure 1 for a visualization of tensor mode.

Definition 2 (unfolding)**.** Unfolding is tensor matricization, which means reshaping the tensor *along the specific mode* to a 2-way tensor.



(a) Mode-1 (column) fibers: $\mathbf{x}_{:jk}$ (b) Mode-2 (row) fibers: $\mathbf{x}_{i:k}$ (c) Mode-3 (tube) fibers: $\mathbf{x}_{ij:}$

Figure 1: mode visualization of a 3rd-order tensor

As an famous example in [4], consider a tensor $\mathcal{X} \in \mathbb{R}^{3\times4\times2}$ (note that dim 2 is put at the end by convention):

$$\mathbf{X}_1 = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}, \quad \mathbf{X}_2 = \begin{bmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{bmatrix},$$

the three mode-n unfoldings are

$$\mathbf{X}_{(1)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{bmatrix},$$

$$\mathbf{X}_{(2)} = \begin{bmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 24 \end{bmatrix},$$

$$\mathbf{X}_{(3)} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & \cdots & 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 & 17 & \cdots & 21 & 22 & 23 & 24 \end{bmatrix}.$$

As mentioned in kolda's paper, the specific permutation of columns is not important so long as it is consistent across related calculations. See [5] for a simple Python code of tensor unfoldings, decompositions, etc.

With the definition of tensor mode and unfolding, we come to tensor multiplication, which is the n-mode product.

Definition 3 (Tensor Multiplication)**.** The $n-mode$ product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ with a matrix $\mathbf{U} \in \mathbb{R}^{J\times I_n}$ is denoted by $\mathcal{X} \times_n \mathbf{U}$ and is of size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$.

Elementwise, we have

$$(\mathcal{X} \times_n \mathbf{U})_{i_1\cdots i_{n-1} j i_{n+1}\cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2\cdots i_N} u_{j i_n}.$$

Tensor n-mode multiplication is simply a reshape(unfolding) and a matrix multiplication. It can be expressed as matrix production

$$\mathcal{Y} = \mathcal{X} \times_n \mathbf{U} \Leftrightarrow \mathbf{Y}_{(n)} = \mathbf{U}\mathbf{X}_{(n)}.$$

Remark (order-irrelevant of n-mode multiplication)**.**

$$\begin{aligned} &\mathcal{X} \times_m \mathbf{A} \times_n \mathbf{B} = \mathcal{X} \times_n \mathbf{B} \times_m \mathbf{A} \quad (m \neq n), \\ &\mathcal{X} \times_n \mathbf{A} \times_n \mathbf{B} = \mathcal{X} \times_n (\mathbf{B}\mathbf{A}) \end{aligned}$$

3.2 Rank

Consider an N-way tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$.

Definition 4 (Rank-One Tensors)**.** Tensor $\mathcal{X}$ is called *rank-one* if it can be written as the outer product of N vectors, i.e.,

$$\mathcal{X} = \mathbf{a}^{(1)} \times \mathbf{a}^{(2)} \times \cdots \times \mathbf{a}^{(N)}.$$

For an elementwise expression, we have

$$x_{i_1 i_2 \cdots i_N} = a_{i_1}^{(1)} a_{i_2}^{(2)} \cdots a_{i_N}^{(N)} \text{ for all } 1 \leq i_n \leq I_n.$$

Definition 5 (Tensor Rank)**.** The *rank* of a tensor $\mathcal{X}$, denoted rank($\mathcal{X}$), is defined as the smallest number of rank-one tensors, which is the smallest number of components in CP decomposition in the next section.

Remark (k-rank)**.** Note that the definition of tensor rank is different from *k-rank* of a matrix. It is defined as the maximum value k such that any k columns are linearly independent of the matrix.

Remark (border rank)**.** The *border rank* is defined as the minimum number of rank-one tensors that are sufficient to approximate the given tensor with arbitrarily small nonzero error:

$$\begin{aligned} \widetilde{\text{rank}}(\mathcal{X}) = \min\{r \mid &\text{ for any } \epsilon > 0, \text{ there exists a tensor } \mathcal{E} \\ &\text{such that } \|\mathcal{X}\| < \epsilon \text{ and } \text{rank}(\mathcal{X} + \mathcal{E}) = r.\} \end{aligned}$$

Definition 6 (Tensor n-Rank)**.** The *n-rank* of a N-way tensor $\mathcal{X}$ is the matrix column rank of $\mathbf{X}_{(\mathbf{n})}$.

4 CP Decomposition

CP is an abbreviation for CANDECOMP/PARAFAC.The CP decomposition factorizes a tensor into a sum of component rank-one tensors. Here we take a third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ as an example.

Theorem 4 (CP decomposition)**.** In CP, we decomposes $\mathcal{X}$ as

$$\mathcal{X} \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \equiv \sum_{r=1}^{R} \mathbf{a_r} \times \mathbf{b_r} \times \mathbf{c_r} \tag{1}$$

and

$$x_{ijk} \approx \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} \text{ for } i \in [I], j \in [J], k \in [K]$$

elementwisely.

Figure 2 is a visualization of CP.

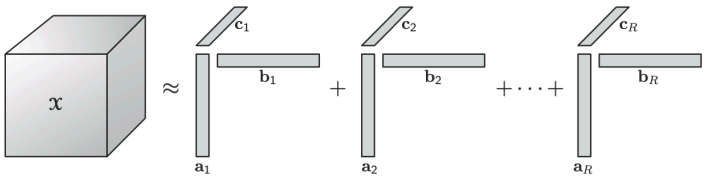


Figure 2: visualization of CP decomposition

Remark (higher order CP)**.** For a general Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the CP decomposition is

$$\mathcal{X} \approx [\![\lambda; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!] \equiv \sum_{r=1}^{R} \lambda_r \mathbf{a}_r^{(1)} \times \mathbf{a}_r^{(2)} \times \cdots \times \mathbf{a}_r^{(N)}$$

Definition 7 (Khatri-Rao product)**.** The *Kharti-Rao product* is the matching columnwise Kronecker product. Given matrices $\mathbf{A} \in \mathbb{R}^{I \times K}$ and $\mathbf{A} \in \mathbb{R}^{J \times K}$, their Khatri-Rao product is denoted by $\mathbf{A} \odot \mathbf{B}$. The result matrix is of size $(IJ) \times K$ defined by

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a_1} \times \mathbf{b_1} | \, \mathbf{a_2} \times \mathbf{b_2} | \, \cdots | \, \mathbf{a_K} \times \mathbf{b_K}].$$

Definition 8 (factor matrices)**.** Combine the vectors from rank-one components, i.e., $\mathbf{A} = [\mathbf{a_1} \; \mathbf{a_2} \; \cdots \; \mathbf{a_R}]$ and likewise for $\mathbf{B}$ and $\mathbf{C}$. Write 1 into matricized form

$$\begin{aligned} \mathbf{X}_{(1)} &\approx \mathbf{A}(\mathbf{C} \odot \mathbf{B})^\top \\ \mathbf{X}_{(2)} &\approx \mathbf{B}(\mathbf{C} \odot \mathbf{A})^\top \\ \mathbf{X}_{(3)} &\approx \mathbf{C}(\mathbf{B} \odot \mathbf{A})^\top \end{aligned}, \tag{2}$$

index below $\mathcal{X}$ denotes mode, $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are factor matrices.

Remark. In a more familiar expression as SVD, we can write 2 as

$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}^{(k)}\mathbf{B}^\top, \quad \text{where} \quad \mathbf{D}^{(k)} \equiv \text{diag}\,(\mathbf{c}_{k:}) \quad \text{for} \quad k =\in [K].$$

Remark. It is often assumed that the columns of factor matrices are normalized to length one with the weights absorbed into the vector $\boldsymbol{\lambda} \in \mathbb{R}^R$ so that

$$\mathcal{X} \approx [\![\boldsymbol{\lambda}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \equiv \sum_{r=1}^{R} \lambda_r \mathbf{a_r} \times \mathbf{b_r} \times \mathbf{c_r}$$

Remark (uniqueness)**.** Higher-order tensors' rank decompositions are often unique, whereas matrix decompositions are not. The CP decomposition is unique under much weaker conditions than SVD.

Computing CP Assume the number of rank-one tensors R is fixed. Now we want to do CP for a third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$. We translate the goal into an optimization model:

$$\min_{\hat{\mathcal{X}}} \|\mathcal{X} - \hat{\mathcal{X}}\| \text{ with } \hat{\mathcal{X}} = [\![\boldsymbol{\lambda}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!].$$

A famous algorithm is *alternating least squares*(ALS), it fix two factor matrices and compute another, and do the same by fixing the other two, and iterate. Note that when fix two factor matrices, the problem becomes

$$\min_{\hat{\mathbf{A}}} \left\| \mathbf{X}_{(1)} - \hat{\mathbf{A}} (\mathbf{C} \odot \mathbf{B})^\top \right\|_F,$$

where $\hat{\mathbf{A}} = \mathbf{A} \cdot \text{diag}(\boldsymbol{\lambda})$.

Remark (NNCP)**.** Sometimes we need to do CP decomposition in the condition that factors are non-negative, hence need to add a non-negative constraint and the decomposition is called NNCP. Simply use Tensorly [5] for Python implementation.

5 Tucker Decomposition

While we can extract little physical meaning from CP decomposition, Tucker decomposition provides a richer meaning and seems to be more elegant.

In fact, Tucker Decomposition a higher-order SVD, which decomposes a tensor into a core tensor multiplied by a matrix along each mode (i.e. do SVD for each mode).

Theorem 5 (Tucker Decomposition)**.** For a N-way tensor $\mathcal{X}$, Tucker decomposition is

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \cdots \times_N \mathbf{A}^{(N)}.$$

For a three-way tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, we have

$$\mathcal{X} \approx \mathbf{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} \mathbf{a}_p \times \mathbf{b}_q \times \mathbf{c}_r = [\![\mathbf{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!], \quad (3)$$

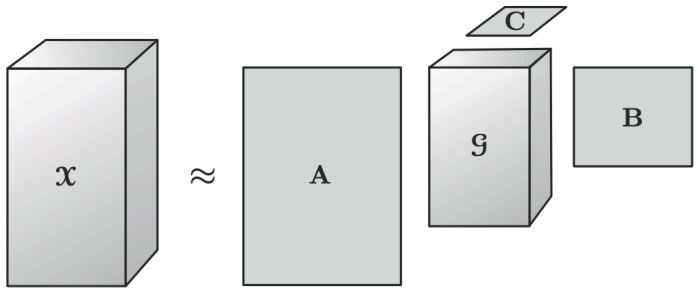


Figure 3: visualization of Tucker decomposition

and

$$x_{ijk} \approx \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_{ip} b_{jq} c_{kr} \quad \text{for} \quad i =\in [I], \quad j \in [J], k =\in [K]$$

elementwisely, where P, Q, R are the number of components in factor matrices chosen for to do decomposition.

Remark (matricized form)**.** The matricized form per mode are

$$\begin{aligned} \mathbf{X}_{(1)} &\approx \mathbf{A}\mathbf{G}_{(1)}(\mathbf{C} \otimes \mathbf{B})^\top \\ \mathbf{X}_{(2)} &\approx \mathbf{B}\mathbf{G}_{(2)}(\mathbf{C} \otimes \mathbf{A})^\top \\ \mathbf{X}_{(3)} &\approx \mathbf{C}\mathbf{G}_{(3)}(\mathbf{B} \otimes \mathbf{A})^\top \end{aligned},$$

where $\otimes$ is the *Kronecker product* for matrices.

Figure 3 is a visualization of Tucker decomposition.

Remark (uniqueness)**.** Same as SVD, Tucker decomposition is not unique. However, the freedom opens the door to choosing transformations that simplify the core structure, i.e. to make it sparser.

Computing Tucker To compute Tucker decomposition, we can do SVD for each mode to get factor matrices, and use

$$\mathcal{G} \leftarrow \mathcal{X} \times_1 \mathbf{A}^{(1)\top} \times_2 \mathbf{A}^{(2)\top} \ldots \times_N \mathbf{A}^{(N)\top}$$

to get core tensor. See [4] for more details.

Remark (NNT)**.** Similar to NNCP, when consider data with non-negative properties, we need to add non-negative constraint and the decomposition is called NNT.

References

[1] Simon Foucart and Holger Rauhut. A mathematical introduction to compressive sensing. *Bull. Am. Math*, 54(2017):151–165, 2017.

[2] Li Hang. *Statistical learning methods, Second Edition*. Tsinghua University Press, 2012.

[3] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[4] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[5] Jean Kossaifi, Yannis Panagakis, Anima Anandkumar, and Maja Pantic. Tensorly: Tensor learning in python. *Journal of Machine Learning Research (JMLR)*, 20(26), 2019.