University of Vienna
Faculty of Mathematics

TENSOR METHODS FOR DATA SCIENCE AND SCIENTIFIC COMPUTING

Milutin Popovic

November 9, 2021

Contents

1 Assignment 2

Let us introduce the *column-major* convention for writing matrices in the indices notation. For $A \in \mathbb{R}^{n\times n}$, which has n^2 entries can be written down in the *column-major* convention as

$$A = [a_{i+n(j-1)}]_{i,j=1,\dots,n}. \tag{1}$$

1.1 Matrix multiplication tensor

The matrix multiplication $AB = C$, for $B, C \in \mathbb{R}^{n\times n}$ is a bilinear operation, thereby there exists a tensor $T = [t_{ijk}] \in \mathbb{R}^{n^2\times n^2\times n^2}$ such that the matrix multiplication can be represented as

$$c_k = \sum_{i=1}^{n^2}\sum_{j=1}^{n^2} t_{ijk} a_i b_k. \tag{2}$$

The tensor T is referred to as the matrix multiplication tensor of order n.

For $n = 2$ we can easily see the non-zero entries by writing out the matrix multiplication in the column major convention

$$c_1 = a_1 b_1 + a_3 b_2, \tag{3}$$
$$c_2 = a_2 b_1 + a_4 b_2, \tag{4}$$
$$c_3 = a_1 b_3 + a_3 b_4, \tag{5}$$
$$c_4 = a_2 b_3 + a_4 b_4. \tag{6}$$

So the non-zero entries in e.g. $k = 1$ are $(1,1)$ and $(3,2)$, and so on. Thus the matrix multiplication Tensor of $n = 2$ is

$$t_{ij1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \tag{7}$$

$$t_{ij2} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \tag{8}$$

$$t_{ij3} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \tag{9}$$

$$t_{ij4} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{10}$$

The *CPD* (Canonical Polyadic Decomposition) of rank-n^3 of the multiplication tensor of order $n = 2$, specified by matrices $U, V, W \in \mathbb{R}^{4\times 8}$ can be represented in such a way that the columns of these matrices satisfy

$$T = \sum_{\alpha=1} u_\alpha \otimes v_\alpha \times w_\alpha. \tag{11}$$

Each nonzero entry in u_α represents the column of the nonzero entry in T, each nonzero entry in v_α represents the row of the nonzero entry in T and each nonzero entry in w_α represents the location of the k-th slice of the nonzero entry, so we have

$$U = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{12}$$

$$V = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}, \tag{13}$$

$$W = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \tag{14}$$

1.2 Mode Contraction

Let us introduce a notion of multiplying a matrix with a tensor which is called the mode-k contraction. For $d \in \mathbb{N}, n_1, \ldots, n_d \in \mathbb{N}$ and $k \in \{1, \ldots, d\}$ the mode-k contraction of a d-dimensional tensor $S \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ with a matrix $Z \in \mathbb{R}^{r \times n_k}$ is an operation

$$\times_k : [s_{i_1,\ldots,i_d}]_{i_1 \in I_1,\ldots,i_d \in I_d} \mapsto \left[\sum_{i_k=1}^{n_k} z_{\alpha i_k} s_{i_1,\ldots,i_d}\right]_{i_1 \in I_1,\ldots,i_{k-1} \in I_{k-1},\alpha \in \{1,\ldots,r\},i_{k+1} \in I_{k+1},\ldots,i_d \in I_d}, \tag{15}$$

where $I_l = \{1, \ldots, n_l\}$ for $l \in \{1, \ldots, d\}$. Now that we know the definition we may write

$$T = Z \times_k S, \tag{16}$$

and T is in $\mathbb{R}^{n_1 \times \cdots n_{k-1} \times r \times n_{k-1} \times \cdots \times n_d}$. The algorithm constructed for the mode-k contraction is structured as follows

1. initialize an empty array T
2. iterate α from 1 to r
3. initialize a zero array s of size $n_1 \times \cdots n_{k-1} \times n_{n+1} \times \cdots n_d$
4. for each $j \in \{1, \dots, n_k\}$ add $Z_{\alpha,j} S_{i_1,\dots,i_{n-1},j,i_{n+1},\dots,i_d}$ to s
5. append s to T
6. end iteration
7. reshape T to $n_1 \times \cdots n_{k-1} \times r \times n_{k-1} \times \cdots \times n_d$ and return it

1.3 Evaluating a CPD

Let us now implement a function that will convert a rank-r CPD of a d-dimensional tensor S of size $n_1 \times \cdots \times n_d$, given $U^{(k)} \in \mathbb{R}^{n_k \times r}$ for $k \in \{1, \dots, k\}$.

$$S = \sum_{\alpha=1}^{r} u_\alpha^{(1)} \otimes \cdots \otimes u_\alpha^{(d)}. \tag{17}$$

The implementation of this is straight forward. While iterating over α all we have to do is call a function that will do a Kronecker product of the α-th column slice of a matrix $U^{(k)}$ with the same order column slice of the matrix $U^{(k+1)}$, for each $k \in \{1, \dots, d\}$. In "julia" the only thing we have to be aware of is that the *kron* function reverses the order for the Kronecker product that is it does the following

$$u \otimes v = kron(v, u). \tag{18}$$

Meaning that if we pass a list of CPD matrices $[U^{(1)}, \dots, U^{(k)}]$ as arguments, we need to reverse its order[1].

1.4 Implementing the multiplication tensor and its CPD

Once again to recapitulate the multiplication tensor is of the size $T \in \mathbb{R}^{n^2 \times n^2 \times n^2}$. With some tinkering we see that the nonzero entries in the k-th end-dimension corresponds to the matrix multiplication in the column major convention, and with some more tinkering we found a way to construct an multiplication tensor for an arbitrary dimension (finite) using three loops.

1. loop over the column indices in the first row in the column major convention $m = 1 : n : n^2$
2. loop over the row indices in the first column in the column major convention $l = 1 : n$
3. $I = l : n : n^2$, $J = m : (m+n)$, $K = (l-1) + m$
4. $T_{i,j,k} = 1$ for every $i \in I, j \in J, k \in K$

Additionally we can even construct a CPD of this tensor in the same loop since we know the indices i, j, k of the nonzero entries, U would be filled with the i-th row entry of each n^3 columns, V would be filled in the j-th row entry of the each n^3 columns and finally W would be filled in the corresponding k-th row entry of the each n^3 columns [1].

Furthermore we can evaluate the matrix multiplication only with the CPD of the matrix multiplication tensor T, with U, V, W without computing T. This is done by rewriting the matrix multiplication in the row-major convention with U, V, W into

$$c_k = \sum_{\alpha=1}^{r} \left(\sum_{i=1}^{n^2} a_i u_{i\alpha} \right) \cdot \left(\sum_{j=1}^{n^2} b_j v_{j\alpha} \right) \cdot w_{k\alpha}. \tag{19}$$

The implementation is straight forward it uses one loop and only a summation function [1].

1.5 Interpreting the Strassen algorithm in terms of CPD

For $n = 2$ we will write the Strassen algorithm and its CPD in the column major convention. This algorithm allows matrix multiplication of square 2×2 matrices (even of order $2^n \times 2^n$) in seven essential multiplication steps instead of the stubborn eight. The Strassen algorithm defines seven new coefficients M_l, where $l \in 1, \cdots, 7$

$$\begin{aligned}
M_1 &:= (a_1 + a_4)(b_1 + b_4), && (20)\\
M_2 &:= (a_2 + a_4)b_1, && (21)\\
M_3 &:= a_1(b_3 - b_4), && (22)\\
M_4 &:= a_4(b_2 - b_1), && (23)\\
M_5 &:= (a_1 + a_3)b_4, && (24)\\
M_6 &:= (a_2 - a_1)(b_1 + b_3), && (25)\\
M_7 &:= (a_3 - a_4)(b_2 + b_4). && (26)
\end{aligned}$$

Then the coefficients c_k can be calculated as

$$\begin{aligned}
c_1 &= M_1 + M_4 - M_5 + M_7, && (27)\\
c_2 &= M_2 + M_4, && (28)\\
c_3 &= M_3 + M_5, && (29)\\
c_4 &= M_1 - M_2 + M_3 + M_6. && (30)\\
& && (31)
\end{aligned}$$

This ultimately means that there exists an rank-7 CPD decomposition of the matrix multiplication tensor T, with matrices $U, V, W \in \mathbb{R}^{4\times 7}$. By that U represents the placement of a_i in the definition of M_l filled in the columns of U, V represents the existence of b_j in the definition of M_l filled in the columns V and W represents the M_l placement in the coefficients c_k. Do note that these matrices have entries $-1, 0, 1$ corresponding to the addition/subtraction as defined in the M_l's. The matrices U, V, W are the following

$$U = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & -1 \end{pmatrix}, \tag{32}$$

$$V = \begin{pmatrix} 1 & 1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 & 1 & 0 & 1 \end{pmatrix}, \tag{33}$$

$$W = \begin{pmatrix} 1 & 0 & 0 & 1 & -1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}. \tag{34}$$

1.6 Tucker Structure and CP rank !!(NO real idea what should be done here)!!

Let $\hat{a} = \begin{pmatrix} 1 & 0 \end{pmatrix}^T$ and $\hat{b} = \begin{pmatrix} 0 & 1 \end{pmatrix}$, we can define a Laplace-like tensor T using the Kronecker product

$$\hat{T} = \hat{b} \otimes \hat{a} \otimes \hat{a} + \hat{a} \otimes \hat{b} \otimes \hat{a} + \hat{a} \otimes \hat{a} \otimes \hat{b} \quad \in \mathbb{R}^{2\times 2\times 2}, \tag{35}$$

which represents T as a CPD of rank 3. The CPD is a special case of the Tucker decomposition with a super diagonal tucker core, in this case the tucker core is

$$S_{i_1 i_2 i_3} = \begin{cases} 1 & \text{for } i_1 = i_2 = i_3 \\ 0 & \text{else} \end{cases} \tag{36}$$

for $i_1,i_2,i_3 = 1,2,3$. We may write the tucker decomposition of T in terms of matrices $U_1,U_2,U_3 \in \mathbb{R}^{2\times 3}$ constructed by the $\hat{a}$ and $\hat{b}$ as in 35 in each of the summation steps. Then U_1 would have $\hat{b},\hat{a},\hat{a}$ in the columns and so on. Now we may write the Tucker decomposition for T_{ijk} for $i_1,i_2,i_3 = 1,2$ as

$$\hat{T}_{i_1 i_2 i_3} = \sum_{\alpha_1}^{3}\sum_{\alpha_2}^{3}\sum_{\alpha_2}^{3}(U_1)_{i_1\alpha_1}(U_2)_{i_2\alpha_2}(U_3)_{i_3\alpha_3}S_{\alpha_1\alpha_2\alpha_3}. \tag{37}$$

With this we can derive the first, second and third Tucker unfolding matrix by rewriting the Tucker decomposition

$$\hat{T}_{i_1 i_2 i_3} = \sum_{\alpha_k=1}^{3}(U_k)_{i_k\alpha_k}\sum_{\alpha_{k-1}=1}^{3}\sum_{\alpha_{k+1}=1}^{3}(U_{k-1})_{i_{k-1}\alpha_{k-1}}(U_{k+1})_{i_{k+1}\alpha_3}S_{\alpha_1\alpha_2\alpha_3} \tag{38}$$

$$= \sum_{\alpha_k=1}^{3}(U_k)_{i_k\alpha_k}(Z_k)_{i_{k-1}i_{k+1}\alpha_k}, \tag{39}$$

for $k = 1,2,3$ cyclic. We call

$$\mathscr{U}_k^T := U_k Z_k^T \tag{40}$$

the $k-th$ Tucker unfolding. It turns out in our case all of them are unique and thereby the CPD decomposition is unique, thereby the CPD rank of $\hat{T}$ is three.

Let us consider a richer structure, for $2 < d \in \mathbb{N}$ and $n_1,\cdot,n_d \in \mathbb{N}$ all greater then one. For $k \in \{1,2,3\}$ consider $a_k,b_k \in \mathbb{R}^{n_k}$ linearly independent and for $\mathbb{N} \ni k \geq 4$ consider $c_k \in \mathbb{R}^{n_k}$ nonzero. Then we define a Laplace-like tensor

$$T = b_1\otimes a_2\otimes a_3\otimes c_4\otimes\cdots\otimes c_d + a_1\otimes b_2\otimes a_3\otimes c_4\otimes\cdots\otimes c_d + a_1\otimes a_2\otimes b_3\otimes c_4\otimes\cdots\otimes c_d. \tag{41}$$

The matrices $U^{(k)} \in \mathbb{R}^{n_k\times 3}$ for $k = 1,2,3$ are the following

$$U^{(1)} = (b_1\ a_2\ a_3), \tag{42}$$

$$U^{(2)} = (a_1\ b_2\ a_3), \tag{43}$$

$$U^{(3)} = (a_1\ a_2\ b_3). \tag{44}$$

$$\tag{45}$$

The matrices $U^{(k)} \in \mathbb{R}^{n_k\times 3}$ for $k = 4,\ldots,d$ are

$$U^{(4)} = (c_4\ c_4\ c_4). \tag{46}$$

$$\vdots$$

$$U^{(d)} = (c_d\ c_d\ c_d). \tag{47}$$

The Tucker core $S \in \mathbb{R}^{3\times 3\times 3}$ is a superdiagonal tensor of order three. We can write the Tucker decomposition of T as

$$\hat{T}_{i_1\ldots i_d} = \sum_{\alpha_k=1}^{3}(U^{(k)})_{i_k\alpha_k}\cdot$$

$$\cdot\sum_{\alpha_1,\ldots,\alpha_{k-1},\alpha_{k+1},\ldots,\alpha_d}^{3}(U^{(1)})_{i_1\alpha_1}\cdots(U^{(k-1)})_{i_{k-1}\alpha_{k-1}}(U^{(k+1)})_{i_{k+1}\alpha_3}\cdots(U^{(d)})_{i_d\alpha_d}S_{\alpha_1\ldots\alpha_d} \tag{48}$$

$$= \sum_{\alpha_k=1}^{3}(U^{(k)})_{i_k\alpha_k}(Z_k)_{i_1,\ldots i_{k-1}i_{k+1}\ldots i_d\alpha_k}. \tag{49}$$

All the Tucker unfoldings $\mathscr{U}_k$ for $k \geq 4$ are non-unique, because the matrices $U^{(k)}$ for $k \geq 4$ are constructed with the same column vectors. On the other hand we may write the Tucker decomposition in with the Kronecker product like this

$$T_{(k)} = U^{(k)}\cdot S_k\cdot(U^{(1)}\otimes\cdots\otimes U^{(k-1)}\otimes U^{(k+1)}\otimes\cdots\otimes U^{(d)}) \tag{50}$$

References

[1] Popovic Milutin. *Git Instance, Tensor Methods for Data Science and Scientific Computing*. URL: `git://popovic.xyz/tensor_methods.git`.