250085 VU
TENSOR METHODS FOR DATA SCIENCE AND SCIENTIFIC COMPUTING
WINTER SEMESTER 2021

HOMEWORK ASSIGNMENT 2

VLADIMIR KAZEEV

Due by **13:00 on Monday, 8th November 2021**. To be submitted via Moodle.

The theoretical part may be (i) typeset, (ii) handwritten digitally or (iii) handwritten on a physical medium and then digitized by scanning or photographing. For the programming part, any programming language and environment may be used.

For each student submitting a solution, the submission should consist of (i) a single PDF file presenting the solution of the therietical part and the results of the programming part in a self-contained fashion (so that running the student's code not be required for understanding the results) and (ii) the code (if any), ready to run and reproduce the results presented in the PDF file, organized in any reasonable number of files.

Throughout this assignment, we will work with tensors representing the multiplication of real square matrices of order $n \in \mathbb{N}$ (small in all the specific cases to be considered). For $A, B, C \in \mathbb{R}^{n \times n}$, we enumerate the n^2 entries of each of the matrices involved "linearly" according to the *column-major* convention:

$$A = \left[a_{i+n(j-1)}\right]_{i,j=1,\ldots,n}, \quad B = \left[b_{i+n(j-1)}\right]_{i,j=1,\ldots,n}, \quad C = \left[c_{i+n(j-1)}\right]_{i,j=1,\ldots,n}.$$

This *column-major* enumeration, in which the column index is the major one (the most significant, "the slowest") and the row index is the minor one (the least significant, "the fastest"), is used for matrices in Fortran, Matlab and Julia, whereas the *row-major* convention, where the row and column indices swap roles, is used in C and NumPy (Python). This may need to be taken into account in the course of implementation.

Whenever $AB = C$ holds, since matrix multiplication is a bilinear operation, there exists a tensor $T = \left[t_{ijk}\right] \in \mathbb{R}^{n^2 \times n^2 \times n^2}$ such that

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

This tensor T is referred to as the tensor of matrix multiplication of order n (the latter is the order of the square matrices whose multiplication the tensor represents; the dimensionality of T, which is often called the order of the tensor in the literature, is $d = 3$ for any $n \in \mathbb{N}$).

1. Forming the multiplication tensor "on paper".

For $n = 2$, write out the multiplication tensor entrywise (namely, its slices along the third dimension) using the slice notation: for $k \in \{1, 2\}$, the kth one is

$$T_{\cdot,\cdot,k} = \begin{bmatrix} t_{11k} & t_{12k} & t_{13k} & t_{14k} \\ t_{21k} & t_{22k} & t_{23k} & t_{24k} \\ t_{31k} & t_{32k} & t_{33k} & t_{34k} \\ t_{41k} & t_{42k} & t_{43k} & t_{44k} \end{bmatrix} .$$

2. Forming a CPD of the multiplication tensor "on paper".

For $n = 2$, write out a rank-eight CPD of T by specifying $U, V, W \in \mathbb{R}^{4\times 8}$ such that the columns of these matrices satisfy

$$T = \sum_{\alpha=1}^{8} u_\alpha \otimes v_\alpha \otimes w_\alpha \, .$$

3. Implementing mode contraction.

For $d \in \mathbb{N}$, $n_1, \ldots, n_d \in \mathbb{N}$ and $k \in \{1, \ldots, d\}$, implement the mode-k contraction of a d-dimensional tensor $S \in \mathbb{R}^{n_1\times\cdots\times n_d}$ with a matrix $Z \in \mathbb{R}^{r\times n_k}$:

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

where $I_\ell = \{1, \ldots, n_\ell\}$ for each $\ell \in \{1, \ldots, d\}$.

4. Evaluating a CPD.

Implement a function that converts a rank-r CPD

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

of a d-dimensional tensor S of size $n_1 \times \cdots \times n_d$, specified in terms of matrices $U^{(k)} \in \mathbb{R}^{n_k\times r}$ with $k \in \{1, \ldots, d\}$, into its entrywise representation (an array of size $n_1 \times \cdots \times n_d$).

5. Implementing the multiplication tensor.

For each $n \in \{2, 3\}$, form the respective multiplication tensor as a three-dimensional array in the programming environment of your choice and use random matrices A and B with $n = 2, 3$ to verify that (1) yields the correct product C.

For each $n \in \{2, 3\}$, form a rank-n^3 CPD of the respective multiplication tensor and verify that evaluating this CPD yields the correct array.

6. Implementing multiplication via the CPD.

Implement a function that, for $n \geq 2$ (small), computes the product C of two matrices A and B (input arguments) using (1) and a rank-r CPD of T (represented by input arguments $U, V, W \in \mathbb{R}^{n^2 \times r}$) *without evaluating the entries of* T.

Use random matrices A and B to verify that this function yields the correct product C.

7. $\star$ **Interpreting the Strassen algorithm in terms of CPD.**

For $n = 2$, write out the Strassen algorithm[1] for the multiplication of square matrices of order n using the column-major enumeration specified above.

Use this algorithm to write out a rank-seven CPD of the repsective multiplication tensor T by specifying $\widehat{U}, \widehat{V}, \widehat{W} \in \mathbb{R}^{4 \times 7}$ such that the columns of these matrices satisfy

$$T = \sum_{\alpha=1}^{7} \widehat{u}_\alpha \otimes \widehat{v}_\alpha \otimes \widehat{w}_\alpha \, .$$

Form this CPD on a computer and verify that evaluating this CPD yields the correct array.

Use random matrices A and B and the implementation from your soution to the previous problem to verify that using (1) with T given by this CPD *(without evaluating the entries of* T*)* yields the correct product C.

8. **The CP rank of a Laplace-like tensor**

For $\widehat{a} = [\,1 \quad 0\,]^\mathsf{T} \in \mathbb{R}^2$ and $\widehat{b} = [\,0 \quad 1\,]^\mathsf{T} \in \mathbb{R}^2$, consider the Laplace-like tensor

$$\widehat{T} = \widehat{b} \otimes \widehat{a} \otimes \widehat{a} + \widehat{a} \otimes \widehat{b} \otimes \widehat{a} + \widehat{a} \otimes \widehat{a} \otimes \widehat{b} \in \mathbb{R}^{2 \times 2 \times 2} \, .$$

This equality represents $\widehat{T}$ by a CPD of rank three. Prove that the CP rank of $\widehat{T}$ is *equal* to three, i.e., that $\widehat{T}$ does not have a CPD of rank lower than three.

9. **The Tucker structure of a Laplace-like tensor**

Consider $d \in \mathbb{N}$ greater than two and $n_1, \ldots, n_d \in \mathbb{N}$ greater than one. For $k \in \{1, 2, 3\}$, consider $a_k, b_k \in \mathbb{R}^{n_k}$ linearly independent and, for $k \in \{4, \ldots, d\}$, consider $c_k \in \mathbb{R}^{n_k}$ nonzero. For the Laplace-like tensor

$$T = b_1 \otimes a_2 \otimes a_3 \otimes c_4 \otimes \cdots \otimes c_d + a_1 \otimes b_2 \otimes a_3 \otimes c_4 \otimes \cdots \otimes c_d + a_1 \otimes a_2 \otimes b_3 \otimes c_4 \otimes \cdots \otimes c_d \in \mathbb{R}^{n_1 \times \cdots \times n_d} \, ,$$

write every Tucker unfolding matrix (consider using the Kronecker-product notation for convenience) and derive a Tucker decomposition with the minimum possible ranks.

Show that the following statements are equivalent: (i) the CP rank of T is equal to three (ii) the CP rank of $\widehat{T}$ is equal to three. Here, $\widehat{T}$ is the tensor from the previous problem.

[1] see, for example, en.wikipedia.org/wiki/Strassen_algorithm