Tensor Starter Kit - 1

Kim Batselier

Relevant literature are the following three articles: 2009_Tensordecomp_Kolda and 2015_Tensor_decomps_cich are review papers that contain a lot of information on tensors and the most important decompositions. The 2011oseledetsTT paper is specifically about a particular tensor decomposition called the Tensor Train (TT). I have constructed some Matlab exercises for which you can find the answers in these articles. My hope is that by solving these exercises you get to have more 'feeling' on basic tensor concepts like k-mode product, matricization, etc.... Also, think about how you can **verify/validate** what comes out of your algorithms.

Useful Matlab commands

- `a=uint8(A)` Convert the elements of a tensor A into unsigned 8-bit integers.
- `a=double(A)` Convert the elements of a tensor A into double precision numbers.
- `a=rand(`$N_1, N_2, \ldots, N_k$`)` creates a k-way array of dimensions $N_1 \times N_2 \times \cdots \times N_k$ with uniformly distributed random entries. Use the command "randn" for Gaussian distributed entries.
- `kron(A,B)` computes the Kronecker product of matrices A and B
- `B=reshape(A,[`$N_1, N_2, \ldots, N_k$`]` reshapes a given d-way array (=tensor, matrix, vector) A into a new k-way array B with dimensions $N_1 \times N_2 \times \cdots \times N_k$. Obviously the product of the dimensions of A and B must be identical.
- `B=permute(A,order)` rearranges the dimensions of A so that they are in the order specified by the vector order. B has the same values of A but the order of the subscripts needed to access any particular element is rearranged as specified by order. All the elements of order must be unique, real, positive, integer values.
- `[U,S,V]=svd(A)` computes the Singular Value Decomposition of a matrix A, with orthogonal matrices U,V and diagonal S matrix.
- `A = imread('filename.jpg')` loads a jpeg picture 'filename.jpg' as a tensor A of 8byte integer pixel values into the Matlab workspace.
- `imshow(A)` displays an image. If A is a colour image then it is a 3-way tensor (pixels x pixels x colour), while a grayscale image is a matrix (pixels x pixels). The values of A need to be of type uint8 (as opposed to double).

Exercises

Exercise 1

Write a Matlab function that efficiently computes the **outer product** of any given number of arbitrary vectors. Now use your function to compute the outer product A1=a o b o c (the circle o stands for the outer product) of three random vectors a,b,c with dimensions(=lengths) 4,2,3 respectively. Then compute the other product A2=b o a o c. Are these two tensors identical or different? Can you write Matlab code to transform the second tensor A2 into A1?

Exercise 2

Write an efficient (=no for-loops) function to compute the **inner product** of two arbitrary-order tensors A,B.

Exercise 3

Write a Matlab function that returns the **mode-n matricization** of a given arbitrary-order tensor A.

Exercise 4

Write a Matlab function that computes the **n-mode product** of an arbitrary d-way tensor with a matrix/vector along any specified mode k ($1 \leq k \leq d$).

Exercise 5

Write a Matlab function that computes a **Tensor Train approximation** of a given tensor for a given tolerance. This is the TT-SVD algorithm, described in the Tensor Train paper.

1. Load the lena.jpg image into Matlab.
2. Convert the image into a 9th order cubical tensor of dimensions 4 ($=4 \times 4 \times \cdots \times 4 = 4^9$).
3. Compute its TT-SVD (convert the uint8 image into type double) for different choices of the tolerance (from the machine precision to 1).

- What are the corresponding TT-ranks?
- What are the TT-ranks when you set the tolerance to 10^{-1} and reconstruct the image from the TT (convert the reconstructed back into type uint8 before using the 'imshow' command). Can you still recognize what is shown in the reconstructed picture?
- How much storage do you need to store the original image, original TT and the truncated TT? Simply count the number of elements of either the tensor or its decomposition.

Exercise 6

Write a Matlab function that implements **Tucker's Method I** (HOSVD).

- Can you reduce the size of the matrices during the SVD computation? (hint: you are looking to find the left singular vectors of a narrow rectangular matrix.)
- Compare the execution speed (using tic and toc command in Matlab) of this method with the speed of the TT-SVD.
- Compare the storage requirement of HOSVD with TT on the Lena image for similar ranks or similar relative approximation error.