A practical mode-parallel implementation of the (H-)Tucker decomposition via randomization

Martina Iannacito

joint work with Sascha Portaro, Claudio Arlandini, Domitilla Brandoni, and Davide Palitta

équipe ROMA, Centre Inria de Lyon

February 5, 2026

Finanziato dall'Unione europea NextGenerationEU

Ministero dell'Università e della Ricerca

Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

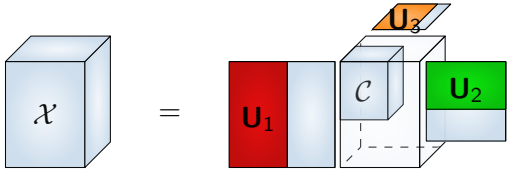
Tucker's decomposition [Tucker 1964]

Let $\mathcal{X}$ be an $(n, \dots, n)$ tensor of multilinear rank $\mathbf{r}$, we want to factorize it as

$$\mathcal{X} = (\mathbf{U}_1, \dots, \mathbf{U}_d)\mathcal{C}$$

where

- $\mathcal{C}$ is the core tensor of size $(r, \dots, r)$;
- $\mathbf{U}_k$ is the k-th factor matrix of size $(n \times r)$
- $\mathbf{U}_k$ a orthogonal matrix



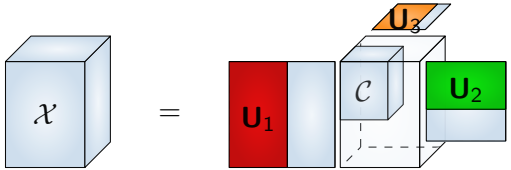
Tucker's decomposition [Tucker 1964]

Let $\mathcal{X}$ be an $(n, \ldots, n)$ tensor of multilinear rank $\mathbf{r}$, we want to factorize it as

$$\mathcal{X} = (\mathbf{U}_1, \ldots, \mathbf{U}_d)\mathcal{C}$$

where

- $\mathcal{C}$ is the core tensor of size $(r, \ldots, r)$;
- $\mathbf{U}_k$ is the k-th factor matrix of size $(n \times r)$
- $\mathbf{U}_k$ a orthogonal matrix

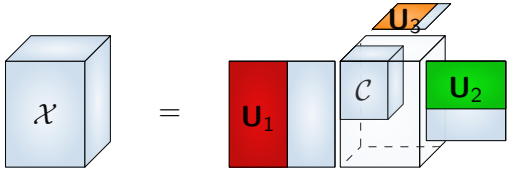


The storage costs are reduced from $\mathcal{O}(n^d)$ to $\mathcal{O}(dnr + r^d)$

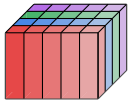
Tucker's decomposition: applications

Examples of applications of the Tucker decomposition (or approximation) are

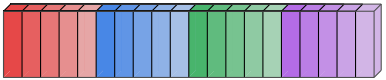
- compression and storage reduction in domains such as computer vision
- denoising in domains such as signal processing and data analysis
- computation in compressed format in domains such as simulations of 3D phenomena
- subspace identification in domains such as signal processing or high-dimensional simulations



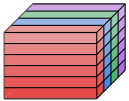
How to compute? Via matricization



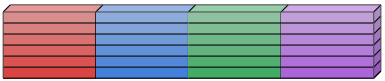
(a1) column fibers, $\mathcal{X}(\cdot, i_2, i_3)$



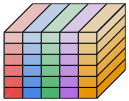
(a2) mode-1 matricization, $\mathbf{X}^{(1)}$



(b1) row fibers, $\mathcal{X}(i_1, \cdot, i_3)$



(b2) mode-2 matricization, $\mathbf{X}^{(2)}$



(c1) tube fibers, $\mathcal{X}(i_1, i_2, \cdot)$

(c2) mode-3 matricization, $\mathbf{X}^{(3)}$

HOSVD

Algorithm 1: $\mathcal{C},\ \{\mathbf{U}_k\}_k = \texttt{HOSVD}(\mathcal{X}, \mathbf{r})$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$

1 **for** $k = 1, \ldots, d$ **do**

2 reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$

3 compute $\mathbf{U}_k$ as the matrix formed by the first r left singular vectors of $\mathbf{X}^{(k)}$

4 set $\mathcal{C} = (\mathbf{U}_1^\top, \ldots, \mathbf{U}_d^\top)\mathcal{X}$

HOSVD

Algorithm 2: $\mathcal{C},\ \{\mathbf{U}_k\}_k = \texttt{HOSVD}(\mathcal{X}, \mathbf{r})$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$

for $k = 1, \ldots, d$ **do**
  reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$
  compute $\mathbf{U}_k$ as the matrix formed by the first r left singular vectors of $\mathbf{X}^{(k)}$
set $\mathcal{C} = (\mathbf{U}_1^\top, \ldots, \mathbf{U}_d^\top)\mathcal{X}$

$+$ parallelizable

HOSVD

Algorithm 3: $\mathcal{C}, \{\mathbf{U}_k\}_k = \texttt{HOSVD}(\mathcal{X}, \mathbf{r})$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$

1 **for** $k = 1, \ldots, d$ **do**
2 reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$
3 compute $\mathbf{U}_k$ as the matrix formed by the first r left singular vectors of $\mathbf{X}^{(k)}$
4 set $\mathcal{C} = (\mathbf{U}_1^\top, \ldots, \mathbf{U}_d^\top)\mathcal{X}$

+ parallelizable
− form d times an $(n \times n^{d-1})$ matrix

HOSVD

Algorithm 4: $\mathcal{C}, \{\mathbf{U}_k\}_k = \texttt{HOSVD}(\mathcal{X}, \mathbf{r})$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$

for $k = 1, \ldots, d$ **do**
  reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$
  compute $\mathbf{U}_k$ as the matrix formed by the first r left singular vectors of $\mathbf{X}^{(k)}$
set $\mathcal{C} = (\mathbf{U}_1^\top, \ldots, \mathbf{U}_d^\top)\mathcal{X}$

- \+ parallelizable
- — form d times an $(n \times n^{d-1})$ matrix
- — d SVDs of an $(n \times n^{d-1})$ matrix

HOSVD

Algorithm 5: $\mathcal{C}, \{\mathbf{U}_k\}_k = \texttt{HOSVD}(\mathcal{X}, \mathbf{r})$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$

1 **for** $k = 1, \dots, d$ **do**
2 reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$
3 compute $\mathbf{U}_k$ as the matrix formed by the first r left singular vectors of $\mathbf{X}^{(k)}$
4 set $\mathcal{C} = (\mathbf{U}_1^\top, \dots, \mathbf{U}_d^\top)\mathcal{X}$

- \+ parallelizable
- — form d times an $(n \times n^{d-1})$ matrix
- — d SVDs of an $(n \times n^{d-1})$ matrix

Can we do better maintaining the algorithm parallelizable?

Algorithm 6: $\mathcal{C},\ \{\mathbf{U}_k\}_k = \texttt{r-HOSVD}(\mathcal{X}, \mathbf{r}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, ℓ oversampling parameter

1 **for** $k = 1, \ldots, d$ **do**

2 reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$

3 compute $\mathbf{U}_k$ as the matrix formed by the first r left singular vectors of $\mathbf{X}^{(k)}$ by the randomized SVD

4 set $\mathcal{C} = (\mathbf{U}_1^\top, \ldots, \mathbf{U}_d^\top)\mathcal{X}$

Randomized HOSVD

Algorithm 7: $\mathcal{C},\ \{\mathbf{U}_k\}_k = \texttt{r-HOSVD}(\mathcal{X}, \mathbf{r}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, ℓ oversampling parameter

for $k = 1, \dots, d$ **do**
- reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$
- define a random sketching matrix $\mathbf{\Omega}_k$ of size $(n^{d-1} \times r + \ell)$
- compute $\mathbf{Q}_k$ from $\texttt{QR}(\mathbf{X}^{(k)}\mathbf{\Omega}_k)$
- compute $\tilde{\mathbf{U}}_k$ from $\texttt{SVD}(\mathbf{Q}_k^\top \mathbf{X}^{(k)})$ truncating at rank r
- define $\mathbf{U}_k = \mathbf{Q}_k \tilde{\mathbf{U}}_k$

set $\mathcal{C} = (\mathbf{U}_1^\top, \dots, \mathbf{U}_d^\top)\mathcal{X}$

Randomized HOSVD

Algorithm 8: $\mathcal{C},\ \{\mathbf{U}_k\}_k = \texttt{r-HOSVD}(\mathcal{X}, \mathbf{r}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, ℓ oversampling parameter

for $k = 1, \dots, d$ **do**
- reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$
- define a random sketching matrix $\mathbf{\Omega}_k$ of size $(n^{d-1} \times r + \ell)$
- compute $\mathbf{Q}_k$ from $\texttt{QR}(\mathbf{X}^{(k)}\mathbf{\Omega}_k)$
- compute $\tilde{\mathbf{U}}_k$ from $\texttt{SVD}(\mathbf{Q}_k^\top \mathbf{X}^{(k)})$ truncating at rank r
- define $\mathbf{U}_k = \mathbf{Q}_k \tilde{\mathbf{U}}_k$

set $\mathcal{C} = (\mathbf{U}_1^\top, \dots, \mathbf{U}_d^\top)\mathcal{X}$

$+$ parallelizable

Randomized HOSVD

Algorithm 9: $\mathcal{C},\ \{\mathbf{U}_k\}_k = \texttt{r-HOSVD}(\mathcal{X}, \mathbf{r}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, ℓ oversampling parameter

for $k = 1, \dots, d$ **do**
- reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$
- define a random sketching matrix $\mathbf{\Omega}_k$ of size $(n^{d-1} \times r + \ell)$
- compute $\mathbf{Q}_k$ from $\texttt{QR}(\mathbf{X}^{(k)}\mathbf{\Omega}_k)$
- compute $\tilde{\mathbf{U}}_k$ from $\texttt{SVD}(\mathbf{Q}_k^\top \mathbf{X}^{(k)})$ truncating at rank r
- define $\mathbf{U}_k = \mathbf{Q}_k \tilde{\mathbf{U}}_k$

set $\mathcal{C} = (\mathbf{U}_1^\top, \dots, \mathbf{U}_d^\top)\mathcal{X}$

$+$ parallelizable

$-$ form d times an $(n \times n^{d-1})$ and an $(n^{d-1} \times (r + \ell))$ matrix

Randomized HOSVD

Algorithm 10: $\mathcal{C},\ \{\mathbf{U}_k\}_k = \texttt{r-HOSVD}(\mathcal{X}, \mathbf{r}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, ℓ oversampling parameter

for $k = 1, \dots, d$ **do**
- reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$
- define a random sketching matrix $\mathbf{\Omega}_k$ of size $(n^{d-1} \times r + \ell)$
- compute $\mathbf{Q}_k$ from $\texttt{QR}(\mathbf{X}^{(k)}\mathbf{\Omega}_k)$
- compute $\tilde{\mathbf{U}}_k$ from $\texttt{SVD}(\mathbf{Q}_k^\top \mathbf{X}^{(k)})$ truncating at rank r
- define $\mathbf{U}_k = \mathbf{Q}_k \tilde{\mathbf{U}}_k$

set $\mathcal{C} = (\mathbf{U}_1^\top, \dots, \mathbf{U}_d^\top)\mathcal{X}$

$+$ parallelizable

$-$ form d times an $(n \times n^{d-1})$ and an $(n^{d-1} \times (r + \ell))$ matrix

$-$ d SVDs of an $((r + \ell) \times n^{d-1})$ matrix

Randomized HOSVD

Algorithm 11: $\mathcal{C},\ \{\mathbf{U}_k\}_k = \texttt{r-HOSVD}(\mathcal{X}, \mathbf{r}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, ℓ oversampling parameter

1 **for** $k = 1, \dots, d$ **do**
2 reshape $\mathcal{X}$ as a matrix $\mathbf{X}^{(k)}$ of size $(n \times n^{d-1})$
3 define a random sketching matrix $\mathbf{\Omega}_k$ of size $(n^{d-1} \times r + \ell)$
4 compute $\mathbf{Q}_k$ from $\texttt{QR}(\mathbf{X}^{(k)}\mathbf{\Omega}_k)$
5 compute $\tilde{\mathbf{U}}_k$ from $\texttt{SVD}(\mathbf{Q}_k^\top \mathbf{X}^{(k)})$ truncating at rank r
6 define $\mathbf{U}_k = \mathbf{Q}_k \tilde{\mathbf{U}}_k$
7 set $\mathcal{C} = (\mathbf{U}_1^\top, \dots, \mathbf{U}_d^\top)\mathcal{X}$

$+$ parallelizable

$-$ form d times an $(n \times n^{d-1})$ and an $(n^{d-1} \times (r + \ell))$ matrix

$-$ d SVDs of an $((r + \ell) \times n^{d-1})$ matrix

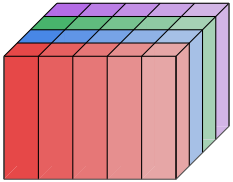
Can we reduce the number of columns of $\mathcal{X}$ reshapes?

A new HOSVD: subsampling and range finder

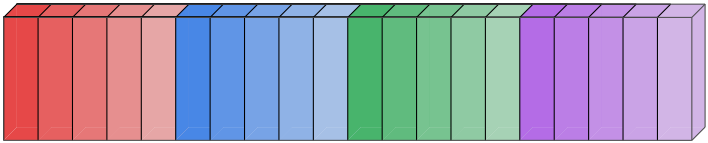
Idea: combining range finder with column sampling, avoiding tensor matricizations

A new HOSVD: subsampling and range finder

Idea: combining range finder with column sampling, avoiding tensor matricizations



(a1) column fibers, $\mathcal{X}(\cdot, i_2, i_3)$



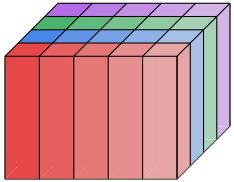
(a2) mode-1 matricization, $\mathbf{X}^{(1)}$

For each mode $k = 1, \ldots, d$

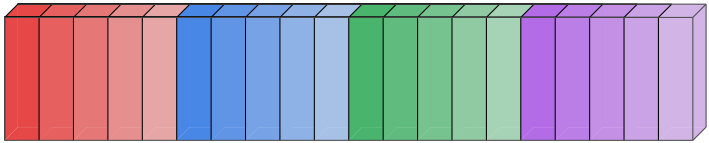
step 1 use **subsampling**: randomly extracting s columns of $\mathbf{X}^{(k)}$ from $\mathcal{X}$

A new HOSVD: subsampling and range finder

Idea: combining range finder with column sampling, avoiding tensor matricizations



(a1) column fibers, $\mathcal{X}(\cdot, i_2, i_3)$



(a2) mode-1 matricization, $\mathbf{X}^{(1)}$

For each mode $k = 1, \ldots, d$

step 1 use **subsampling**: randomly extracting s columns of $\mathbf{X}^{(k)}$ from $\mathcal{X}$

step 2 use **range finder**: create a sketching matrix to generate a basis for $\mathrm{range}(\mathbf{X}^{(k)})$ using only the s chosen columns

A new HOSVD: subsampling and range finder

Algorithm 12: $\mathcal{C}, \{\mathbf{U}_k\}_k = \texttt{subr-HOSVD}(\mathcal{X}, \mathbf{r}, \mathbf{s}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, $\mathbf{s}$ number of fibers to sample, ℓ oversampling parameter

1 **for** $k = 1, \ldots, d$ **do**
2 randomly pick s mode-k fibers and store them in the matrix $\mathbf{Y}_k$ of size $(n \times s)$;
3 define a random sketching matrix $\mathbf{\Omega}_k$ of size $(s \times (r + \ell))$;
4 compute $\mathbf{Q}_k, \mathbf{R}_k$ from $\texttt{QR}(\mathbf{Y}_k \mathbf{\Omega}_k)$;
5 compute $\tilde{\mathbf{U}}_k$ from $\texttt{SVD}(\mathbf{R}_k)$ truncating at rank r;
6 define $\mathbf{U}_k = \mathbf{Q}_k \tilde{\mathbf{U}}_k$
7 Compute $\mathcal{C} = (\mathbf{U}_1^T, \ldots, \mathbf{U}_d^T)\mathcal{X}$;

A new HOSVD: subsampling and range finder

Algorithm 13: $\mathcal{C}, \{\mathbf{U}_k\}_k = \texttt{subr-HOSVD}(\mathcal{X}, \mathbf{r}, \mathbf{s}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, $\mathbf{s}$ number of fibers to sample, ℓ oversampling parameter

1 **for** $k = 1, \ldots, d$ **do**
2 randomly pick s mode-k fibers and store them in the matrix $\mathbf{Y}_k$ of size $(n \times s)$;
3 define a random sketching matrix $\mathbf{\Omega}_k$ of size $(s \times (r + \ell))$;
4 compute $\mathbf{Q}_k, \mathbf{R}_k$ from $\texttt{QR}(\mathbf{Y}_k \mathbf{\Omega}_k)$;
5 compute $\tilde{\mathbf{U}}_k$ from $\texttt{SVD}(\mathbf{R}_k)$ truncating at rank r;
6 define $\mathbf{U}_k = \mathbf{Q}_k \tilde{\mathbf{U}}_k$
7 Compute $\mathcal{C} = (\mathbf{U}_1^T, \ldots, \mathbf{U}_d^T)\mathcal{X}$;

\+ **really** parallelizable

A new HOSVD: subsampling and range finder

Algorithm 14: $\mathcal{C}, \{\mathbf{U}_k\}_k = \texttt{subr-HOSVD}(\mathcal{X}, \mathbf{r}, \mathbf{s}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, $\mathbf{s}$ number of fibers to sample, ℓ oversampling parameter

1 **for** $k = 1, \ldots, d$ **do**
2 randomly pick s mode-k fibers and store them in the matrix $\mathbf{Y}_k$ of size $(n \times s)$;
3 define a random sketching matrix $\mathbf{\Omega}_k$ of size $(s \times (r + \ell))$;
4 compute $\mathbf{Q}_k, \mathbf{R}_k$ from $\texttt{QR}(\mathbf{Y}_k \mathbf{\Omega}_k)$;
5 compute $\tilde{\mathbf{U}}_k$ from $\texttt{SVD}(\mathbf{R}_k)$ truncating at rank r;
6 define $\mathbf{U}_k = \mathbf{Q}_k \tilde{\mathbf{U}}_k$
7 Compute $\mathcal{C} = (\mathbf{U}_1^T, \ldots, \mathbf{U}_d^T)\mathcal{X}$;

\+ **really** parallelizable

\+ never form the large matrices, at most $(n \times s)$ and $(s \times (r + \ell))$ matrices

A new HOSVD: subsampling and range finder

Algorithm 15: $\mathcal{C}, \{\mathbf{U}_k\}_k = \texttt{subr-HOSVD}(\mathcal{X}, \mathbf{r}, \mathbf{s}, \ell)$

Input: $\mathcal{X}$ order d tensor, its multilinear rank $\mathbf{r}$, $\mathbf{s}$ number of fibers to sample, ℓ oversampling parameter

1 **for** $k = 1, \dots, d$ **do**
2 randomly pick s mode-k fibers and store them in the matrix $\mathbf{Y}_k$ of size $(n \times s)$;
3 define a random sketching matrix $\mathbf{\Omega}_k$ of size $(s \times (r + \ell))$;
4 compute $\mathbf{Q}_k, \mathbf{R}_k$ from $\texttt{QR}(\mathbf{Y}_k \mathbf{\Omega}_k)$;
5 compute $\tilde{\mathbf{U}}_k$ from $\texttt{SVD}(\mathbf{R}_k)$ truncating at rank r;
6 define $\mathbf{U}_k = \mathbf{Q}_k \tilde{\mathbf{U}}_k$
7 Compute $\mathcal{C} = (\mathbf{U}_1^T, \dots, \mathbf{U}_d^T)\mathcal{X}$;

\+ **really** parallelizable

\+ never form the large matrices, at most $(n \times s)$ and $(s \times (r + \ell))$ matrices

\+ d SVDs of an $((r + \ell) \times (r + \ell))$ matrix

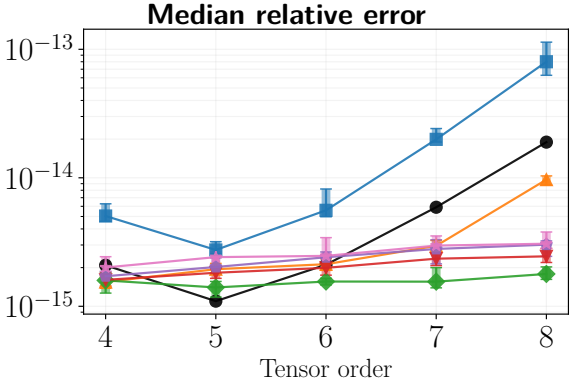
Numerical experiments

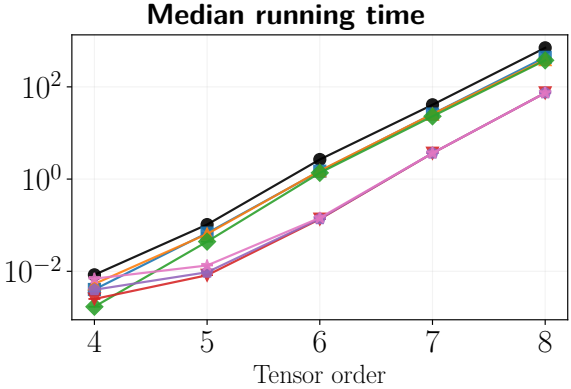
Tested competitor algorithms:

- High Order Singular Value Decomposition (HOSVD) [De Lathauwer et al. 2000], deterministic
- randomized HOSVD (rHOSVD) [Halko et al. 2011], that is a randomized algorithm in which the deterministic SVD is replaced by the SVD
- Subspace Iteration HOSVD (SI-HOSVD) [Halko et al. 2011], that is is a randomized algorithm in which a power of the sketching of matricization is used to compute the SVD for each mode
- randomized HOSVD with Kronecker product Reusing factor (ReKron-HOSVD) [Minster et al. 2024], that is is a randomized algorithm in which all modes except one are sketched iteratively to approximate the range of tensor matricization along the non-sketched mode.

Synthetic sequential results - I

Generate a low-rank tensor $\mathcal{X} = (\mathbf{U}_1, \ldots, \mathbf{U}_d)\mathcal{C}$ where $\mathcal{C}$ is an $(r, \ldots, r)$ tensor and $\mathbf{U}_k$ are orthogonal matrices of size $(n \times r)$ generated from the Gaussian distribution, with $n = 15$, $r = 5$, and $d = 4, \ldots, 8$.

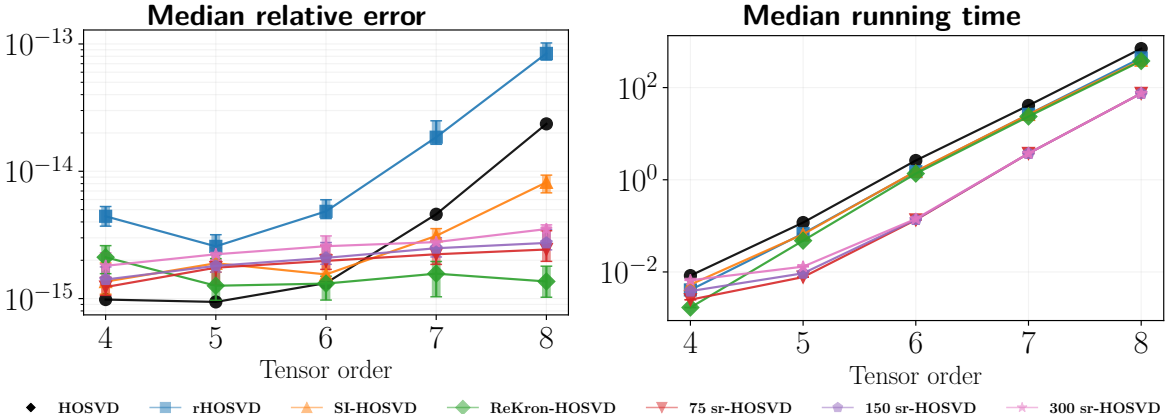




HOSVD rHOSVD SI-HOSVD ReKron-HOSVD 75 sr-HOSVD 150 sr-HOSVD 300 sr-HOSVD

Synthetic sequential results - II

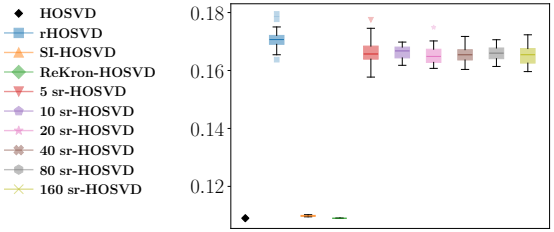
Generate a low-rank tensor $\mathcal{X} = (\mathbf{U}_1, \ldots, \mathbf{U}_d)\mathcal{C}$ where $\mathcal{C}(i_1, \ldots, i_d) = \left(\sqrt[5]{(i_1^5 + \cdots + i_d^5)}\right)^{-1}$ for $i_k = 1, \ldots, r$, and $\mathbf{U}_k$ are orthogonal randomly generated matrices of size $(n \times r)$, with $n = 15$, $r = 5$, and $d = 4, \ldots, 8$.



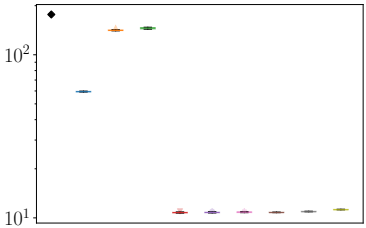
The coil data: results I

The coil-100 dataset [Nene et al. 1996] contains colored 128×128 images of 100 objects photographed with 72 different angles. We assemble them in a tensor of size $(100, 128, 128, 3, 72)$ and we approximate it at multilinear rank $(100, 64, 64, 3, 24)$.

Median relative error



Median running time



The coil data: results II

First image of the 14th object.

original

rHOSVD

SI-HOSVD

ReKron-HOSVD

5 sr-HOSVD

160 sr-HOSVD

Parallelized version

The parallelized `subr-HOSVD` is such that: given $\mathcal{X}$ an order-d tensor and c processes

- parallelization of the computation of the Tucker factors, $\mathbf{U}_k$
  1. processes collects s fibers along mode-k of $\mathcal{X}$ for a certain mode $k = 1, \ldots, d$;
  2. each process computes its $\mathbf{U}_k$ using the range finder;
- parallelization of the computation of the Tucker core tensor
  1. process-1 receives all the $\mathbf{U}_k$;
  2. process-1 distributes subtensors of $\mathcal{X}$ along the first mode and the last-$(d-1)$ $\mathbf{U}_k$;
  3. each process computes the ten-mat product of its subtensor and the last-$(d-1)$ $\mathbf{U}_k$;
  4. process-1 receives all the ten-mat product outputs and reassemble the full tensor;
  5. process-1 perform the mode-1 ten-mat product.

Synthetic parallel results I - scaling

Generate a low-rank tensor $\mathcal{X} = (\mathbf{U}_1, \ldots, \mathbf{U}_d)\mathcal{C}$ where $\mathcal{C}$ is an $(r, \ldots, r)$ tensor and $\mathbf{U}_k$ are orthogonal matrices of size $(n \times r)$ generated from the uniform distribution over $[0, 1]$, with $d = 8$, $r = 7$ and $n = 16, \ldots, 20$.

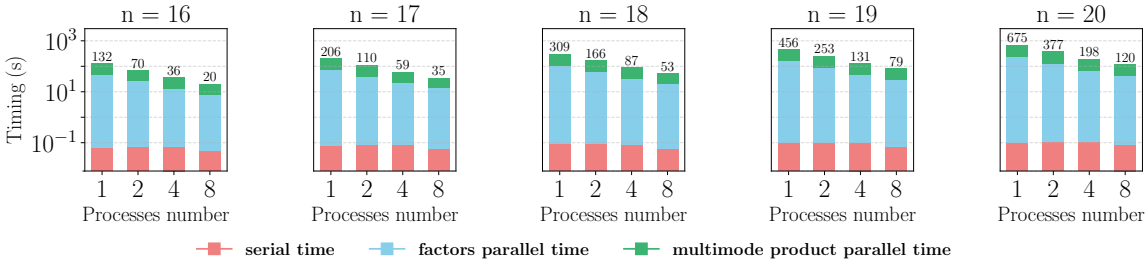


Figure: Scaling of execution times for the two parallel phases as the number of processes increases.

Synthetic parallel results II - speedup

Generate a low-rank tensor $\mathcal{X} = (\mathbf{U}_1, \ldots, \mathbf{U}_d)\mathcal{C}$ where $\mathcal{C}$ is an $(r, \ldots, r)$ tensor and $\mathbf{U}_k$ are orthogonal matrices of size $(n \times r)$ generated from the uniform distribution over $[0, 1]$, with $d = 8$, $r = 7$ and $n = 16, \ldots, 20$.

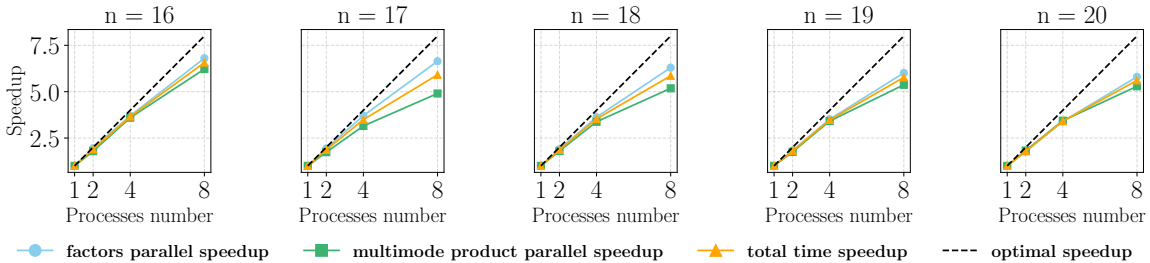
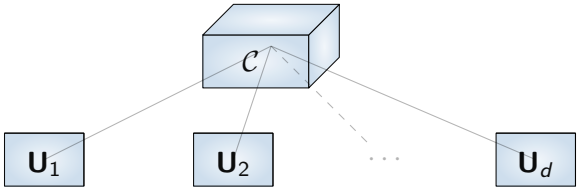


Figure: Speedup achieved in each of the two parallel phases and the total speedup, compared with the ideal linear speedup. The total speedup includes the contribution from the serial portion of the computation.

From Tucker to $\mathcal{H}$-Tucker

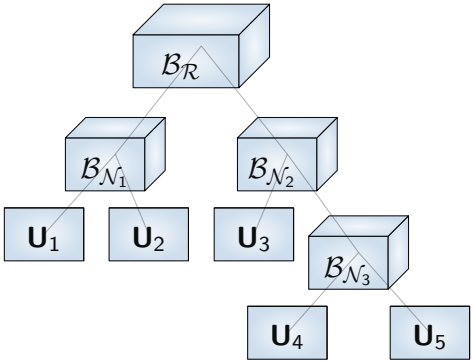
The Tucker decomposition of $\mathcal{X}$ of order-d can be visually described as

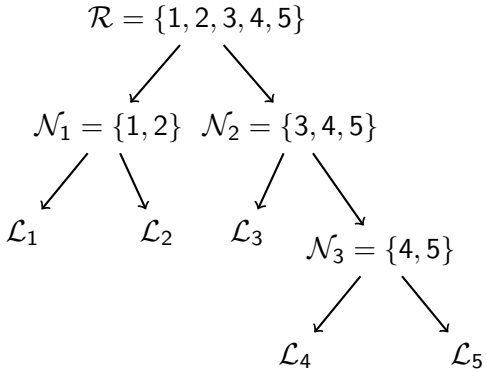


The storage costs are $\mathcal{O}(dnr + r^d)$ with $\mathcal{O}(dnr)$ for the leaves and $\mathcal{O}(r^d)$ for the root

Visual representation of H-Tucker

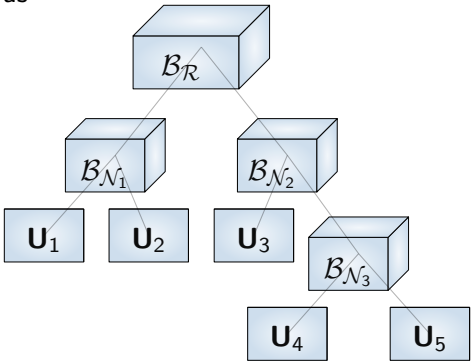
The Hierarchical-Tucker decomposition of $\mathcal{X}$ of order-5 can be visually described as





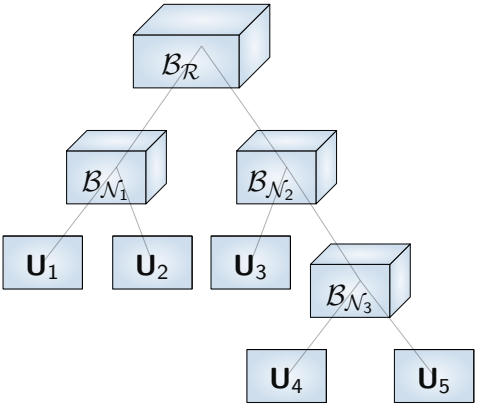
The storage costs are $\mathcal{O}(dnr + dr^3)$ with $\mathcal{O}(dnr)$ for the leaves and $\mathcal{O}(dr^3)$ for the inner nodes

The H-Tucker decomposition (or approximation) is used mainly in high-dimensional problems such as



- quantum chemistry simulations
- stochastic PDEs
- multi-dimensional integrals
- multivariate regression and machine learning

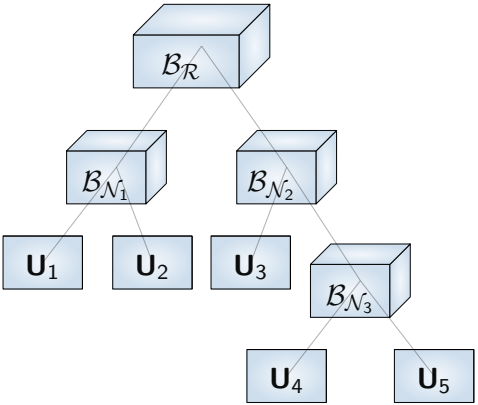
H-Tucker: leaves to root



- Tucker factors are in the leaves

$$\mathbf{U}_k \qquad \text{from} \qquad \texttt{SVD}(\mathbf{X}^{(k)})$$

H-Tucker: leaves to root



- Tucker factors are in the leaves

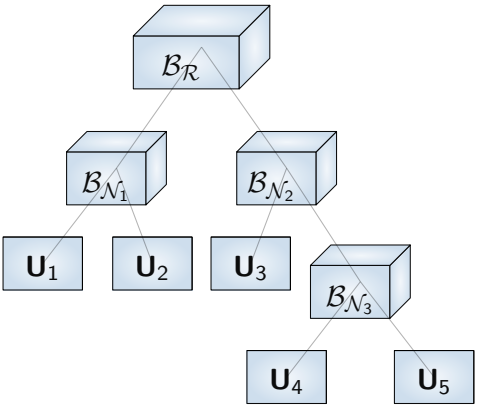
$$\mathbf{U}_k \qquad \text{from} \qquad \texttt{SVD}(\mathbf{X}^{(k)})$$

- Transfer tensors are in the inner nodes

$$\mathcal{B}_{\mathcal{N}_j} \qquad \text{from} \qquad \texttt{reshape}(\mathbf{U}_{\mathcal{N}_j})$$

where $\mathbf{U}_{\mathcal{N}_j}$ comes from $\texttt{SVD}(\tilde{\mathbf{X}}^{(\mathcal{N}_j)})$ with $\tilde{\mathcal{X}}$ coming from a transformation of $\mathcal{X}$ that takes into account the information in the children of $\mathcal{N}_j$

H-Tucker: leaves to root



- Tucker factors are in the leaves

$$\mathbf{U}_k \qquad \text{from} \qquad \texttt{SVD}(\mathbf{X}^{(k)})$$

- Transfer tensors are in the inner nodes

$$\mathcal{B}_{\mathcal{N}_j} \qquad \text{from} \qquad \texttt{reshape}(\mathbf{U}_{\mathcal{N}_j})$$

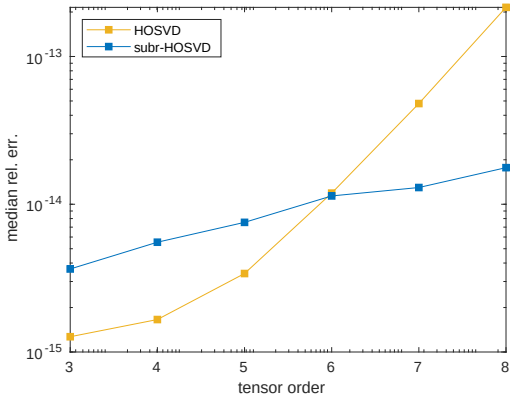
where $\mathbf{U}_{\mathcal{N}_j}$ comes from $\texttt{SVD}(\tilde{\mathbf{X}}^{(\mathcal{N}_j)})$ with $\tilde{\mathcal{X}}$ coming from a transformation of $\mathcal{X}$ that takes into account the information in the children of $\mathcal{N}_j$

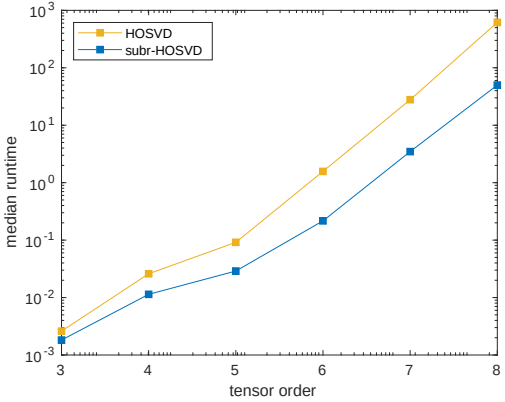
Approximately $2(d-1)$ SVDs are needed, of matrices of size $(n^h \times n^{d-h})$

Synthetic H-Tucker results

Idea: replacing SVD with randomized SVD, combining sampling and sketching

Generate a low-rank H-Tucker tensor $\mathcal{X}$ using the Gaussian random distribution and a balanced binary tree, with $n = 15$ and H-Tucker rank $r = 5$ for all nodes.





Summary

We presented randomized (H-)Tucker algorithms, discussing

- combination of subsampling and range finder
- parallelization version for `subr-HOSVD`
- sequential version for `subr-H-Tucker`
- numerical examples

The pre-print will be available soon!

References I

- De Lathauwer, L., B. De Moor, and J. Vandewalle (2000). "A Multilinear Singular Value Decomposition". In: *SIAM Journal on Matrix Analysis and Applications* 21.4, pp. 1253–1278.
- Halko, N., P. G. Martinsson, and J. A. Tropp (2011). "Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions". In: *SIAM Review* 53.2, pp. 217–288.
- Minster, R., Z. Li, and G. Ballard (2024). "Parallel Randomized Tucker Decomposition Algorithms". In: *SIAM Journal on Scientific Computing* 46.2, A1186–A1213.
- Nene, S. A., S. K. Nayar, and H. Murase (1996). *Columbia object image library (coil-100)*.
- Tucker, L. R. (1964). "The extension of factor analysis to three-dimensional matrices". In: *Contributions to mathematical psychology*. Ed. by H. Gulliksen and N. Frederiksen. New York: Holt, Rinehart and Winston, pp. 110–127.