PRACTICAL SKETCHING-BASED RANDOMIZED TENSOR RING DECOMPOSITION[1]

YAJIE YU

CHONGQING UNIVERSITY, CHONGQING, P.R. CHINA

SEPTEMBER 25, 2022

---

[1] A joint work with Hanyu Li
E-mail: zqyu@cqu.equ.cn

PRESENTATION OUTLINE

TENSOR

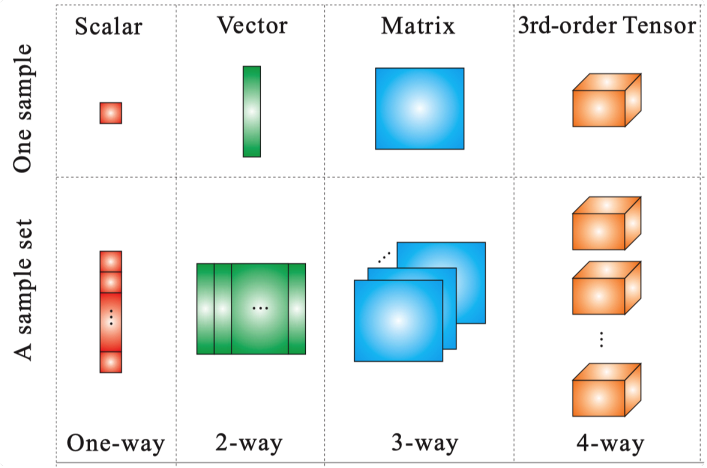


Figure: Graphical representation of multiway array (tensor) data.

TENSOR

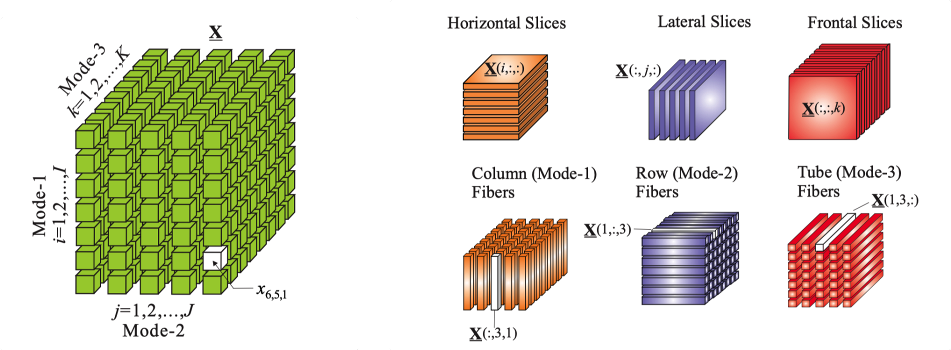


Figure: A 3rd-order tensor with entries, slices and fibers.

CP & Tucker Decompositions

- CANDECOMP/PARAFAC (CP) decomposition.
  - The CP tensor decomposition aims to approximate an Nth-order tensor as a sum of R rank-one tensors;
  - $\boldsymbol{\mathcal{X}} \approx \tilde{\boldsymbol{\mathcal{X}}} = \sum_{r=1}^{R} \boldsymbol{a}_r^{(1)} \circ \boldsymbol{a}_r^{(2)} \circ \cdots \circ \boldsymbol{a}_r^{(N)} = [[\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \cdots, \mathbf{A}^{(N)}]]$;
  - $\mathcal{O}(NIR)$ parameters: is linear to the tensor order N.
- Tucker decomposition
  - The Tucker decomposition decomposes a tensor into a core tensor multiplied (or transformed) by a matrix along each mode;
  - $\boldsymbol{\mathcal{X}} \approx \tilde{\boldsymbol{\mathcal{X}}} = \boldsymbol{\mathcal{G}} \times_1 \mathbf{A}^{(1)} \cdots \times_N \mathbf{A}^{(N)} = [[\boldsymbol{\mathcal{G}}; \mathbf{A}^{(1)}, \cdots, \mathbf{A}^{(N)}]]$;
  - $\mathcal{O}\left(NIR + R^N\right)$ parameters: is exponential to the tensor order N.
- Some limitations

CP	Its optimization problem is difficult; it is difficult to find the optimal solution and CP-rank (NP-hard);
Tucker	Its number of parameters is exponential to tensor order. (Curse of Dimensionality)

TENSOR TRAIN (TT) DECOMPOSITION: ILLUSTRATION

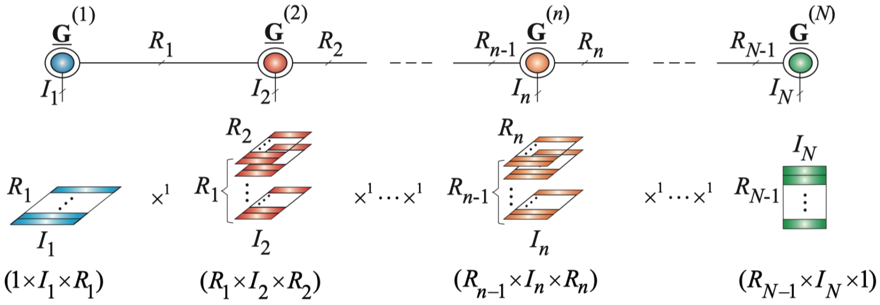


Figure: TT/MPS decomposition of an Nth-order tensor $\boldsymbol{\mathcal{X}}$.

- Slice representation:

$$\boldsymbol{\mathcal{X}}(i_1,\cdots,i_N) = \mathbf{G}_1(i_1)\mathbf{G}_1(i_2)\cdots\mathbf{G}_N(i_N)$$

TENSOR TRAIN (TT) DECOMPOSITION: LIMITATIONS

- Limitations of TT decomposition:
    - The constraint on TT-ranks, i.e., $R_1 = R_{N+1} = 1$, leads to the limited representation ability and flexibility;
    - TT-ranks always have a fixed pattern, i.e., smaller for the border cores and larger for the middle cores, which might not be the optimum for specific data tensor;
    - The multilinear products of cores in TT decomposition must follow a strict order such that the optimized TT cores highly depend on the permutation of tensor dimensions. Hence, finding the optimal permutation remains a challenging problem.

TENSOR RING (TR) DECOMPOSITION: ILLUSTRATION

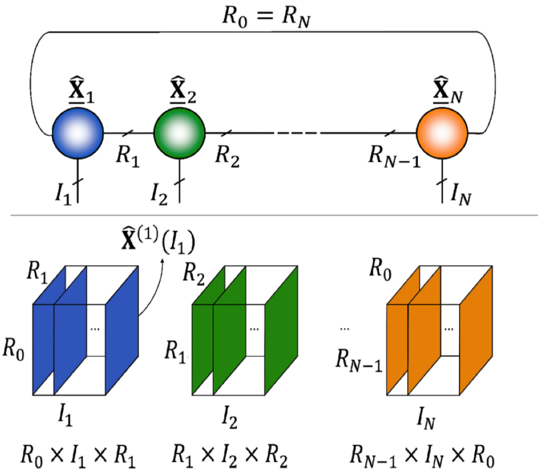


Figure: TR decomposition of an Nth-order tensor $\mathcal{X}$.

TENSOR RING (TR) DECOMPOSITION: DIFFERENT REPRESENTATIONS

- Scalar representation:

$$\boldsymbol{\mathcal{X}}(i_1, \cdots, i_N) = \sum_{r_1,\cdots,r_N=1}^{R_1,\cdots,R_N} \prod_{n=1}^{N} \boldsymbol{\mathcal{G}}_n(r_n, i_n, r_{n+1}); \;\; R_1 = R_{N+1}$$

- Slice representation:

$$\boldsymbol{\mathcal{X}}(i_1, \cdots, i_N) = \text{Trace}\{\mathbf{G}_1(i_1)\mathbf{G}_1(i_2)\cdots\mathbf{G}_N(i_N)\};$$

- Tensor representation:

$$\boldsymbol{\mathcal{X}} = \text{Trace}\left(\mathbf{G}_1 \times^1 \mathbf{G}_2 \times^1 \cdots \times^1 \mathbf{G}_N\right);$$

- $\mathcal{O}\left(NIR^2\right)$ parameters: is linear to the tensor order N.

TENSOR RING (TR) DECOMPOSITION: ADVANTAGES

- Advantages of TR decomposition:
  - TR model has a more generalized and powerful representation ability than TT model, due to relaxation of the strict condition $R_1 = R_{N+1} = 1$ in TT decomposition. In fact, TT decomposition can be viewed as a special case of TR model; (Overcome the first limitation of TT decomposition.)
  - TR model is more flexible than TT model because TR-ranks can be equally distributed in the cores; (Overcome the second limitation of TT decomposition.)
  - The multilinear products of cores in TR decomposition don't need a strict order, i.e., the circular dimensional permutation invariance. (Overcome the third limitation of TT decomposition.)
  - TR-ranks are usually smaller than TT-ranks because TR model can be represented as a linear combination of TT decompositions whose cores are partially shared.

CLASSICAL ALGORITHMS FOR TR DECOMPOSITION

- SVD-based algorithm (TR-SVD).
- ALS-based algorithm (TR-ALS).

Algorithm 1 TR-ALS[2][3]

```
function $\{\mathcal{G}_n\}_{n=1}^{N}$ = TR-ALS($\mathcal{X}, R_1, \cdots, R_N$)    ▷ $\mathcal{X}$ is the input tensor
                                                                              ▷ $R_1, \cdots, R_N$ are the TR-ranks
    Initialize cores $\mathcal{G}_1, \cdots, \mathcal{G}_N$
    repeat
        for $n = 1, \cdots, N$ do
            Compute $\mathbf{G}_{[2]}^{\neq n}$ from cores
            Update $\mathcal{G}_n = \arg\min_{\mathcal{Z}} \|\mathbf{G}_{[2]}^{\neq n}\mathbf{Z}_{(2)}^{\mathsf{T}} - \mathbf{X}_{[n]}^{\mathsf{T}}\|_F$
        end for
    until termination criteria met
end function
```

---

[2] Qibin Zhao et al. "Tensor Ring Decomposition". In: *arXiv preprint arXiv:1606.05535* (2016).

[3] More details: (1) ALS with adaptive ranks and (2) block-wise ALS

RANDOMIZED ALGORITHMS FOR TR DECOMPOSITION

Algorithm 2 rTR-ALS[4][5]

```
function $\{\mathcal{G}_n\}_{n=1}^{N}$= TR-RALS($\boldsymbol{\mathcal{X}}, R_1, \cdots, R_N, K_1, \cdots, K_N$)
    for $n = 1, \cdots, N$ do
        Create matrix $\mathbf{M} \in \mathbb{R}_{i \neq n} I_i \times K_n$ following the Gaussian distribution.
        Compute $\mathbf{Y} = \mathbf{X}_{(n)}\mathbf{M}$                    ▷ random projection
        $[\mathbf{Q}_n, \ ] = \text{QR}(\mathbf{Y})$                      ▷ economy QR decomposition
        $\boldsymbol{\mathcal{P}} \leftarrow \boldsymbol{\mathcal{X}} \times_n \mathbf{Q}_n^{\mathsf{T}}$
    end for
    Obtain TR factors $[\boldsymbol{\mathcal{Z}}_n]$ of $\boldsymbol{\mathcal{P}}$ by TR-ALS or TR-SVD
    for $n = 1, \cdots, N$ do
        $\boldsymbol{\mathcal{G}}_n = \boldsymbol{\mathcal{Z}}_n \times_2 \mathbf{Q}_n$
    end for
end function
```

---

[4] Longhao Yuan et al. "Randomized Tensor Ring Decomposition and Its Application to Large-Scale Data Reconstruction". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Brighton Conference Centre Brighton, U.K.: IEEE, 2019, pp. 2127–2131.

[5] Salman Ahmadi-Asl et al. "Randomized Algorithms for Fast Computation of Low Rank Tensor Ring Model". In: *Mach. Learn.: Sci. Technol.* 2.1 (2020), p. 011001. DOI: 10.1088/2632-2153/abad87.

Randomized Algorithms for TR Decomposition

Algorithm 3 TR-ALS-Sampled[6]

```
function $\{\boldsymbol{\mathcal{G}}_n\}_{n=1}^{N}$ = TR-ALS-SAMPLED($\boldsymbol{\mathcal{X}}, R_1, \cdots, R_N, m$)    ▷ $m$ is the sampling size
    Initialize cores $\boldsymbol{\mathcal{G}}_2, \cdots, \boldsymbol{\mathcal{G}}_N$
    Using the leverage scores to compute distributions $\boldsymbol{p}^{(2)}, \cdots, \boldsymbol{p}^{(N)}$ without explicitly forming the subchain unfold matrix.
    repeat
        for $n = 1, \cdots, N$ do
            Set sample size $J$
            Draw sampling matrix $\mathbf{S} \sim \mathcal{D}(J, \mathbf{q}^{\neq n})$
            Compute $\hat{\boldsymbol{\mathcal{G}}}^{\neq n} = \mathrm{SST}(\texttt{idxs}, \boldsymbol{\mathcal{G}}_{n+1}, \boldsymbol{\mathcal{G}}_N, \boldsymbol{\mathcal{G}}_1, \boldsymbol{\mathcal{G}}_{n-1})$ and $\hat{\mathbf{G}}_{[2]}^{\neq n}$
            Compute $\hat{\mathbf{X}}_{[n]}^{\intercal} = \mathbf{S}\mathbf{X}_{[n]}^{\intercal}$
            Update $\boldsymbol{\mathcal{G}}_n = \arg\min_{\boldsymbol{\mathcal{Z}}} \|\hat{\mathbf{G}}_{[2]}^{\neq n}\mathbf{Z}_{(2)}^{\intercal} - \hat{\mathbf{X}}_{[n]}^{\intercal}\|_F$
            Update $n$-th distribution $\boldsymbol{p}^{(n)}$
        end for
    until termination criteria met
end function
```

---

[6] Osman Asif Malik and Stephen Becker. “A Sampling-Based Method for Tensor Ring Decomposition”. In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Virtual Event: PMLR, 2021, pp. 7400–7411.

EFFICIENT SAMPLING STRATEGY SUMMARIZED FROM [MB21A]

Algorithm 4 Sampled Subchain Tensor (SST)[7]

```
function $\mathcal{G}_S^{\neq n}$ = SST($\texttt{idxs}, \mathcal{G}_{n+1}, \cdots, \mathcal{G}_N, \mathcal{G}_1, \cdots, \mathcal{G}_{n-1}$)    ▷ $\mathcal{G}_n \in \mathbb{R}^{R_n \times I_n \times R_{n+1}}, n \in [N]$
        ▷ idxs $\in \mathbb{R}^{m \times (N-1)}$ is from the set of tuples $\{i_{n+1}^{(j)}, \cdots, i_N^{(j)}, i_1^{(j)}, \cdots, i_{n-1}^{(j)}\}$ for $j \in [m]$
        ▷ idxs can be retrieved from the sampling matrix $\mathbf{S} \in \mathbb{R}^{m \times \prod_{k \neq n} I_k}$ or the specific sampling with given probabilities
    Let $\mathcal{G}_S^{\neq n}$ be a tensor of size $R_{n+1} \times m \times R_n$, where every lateral slice is an $R_{n+1} \times R_n$ identity matrix
    for $k = n+1, \cdots, N, 1, \cdots, n-1$ do
        $\mathcal{G}_{(k)S} \leftarrow \mathcal{G}_k(:, \texttt{idxs}(:, k), :)$
        $\mathcal{G}_S^{\neq n} \leftarrow \mathcal{G}_S^{\neq n} \boxtimes_2 \mathcal{G}_{(k)S}$    ▷ see Definition 3.2 for $\boxtimes_2$
    end for
    return $\mathcal{G}_S^{\neq n}$
end function
```

---

[7] Osman Asif Malik and Stephen Becker. “A Sampling-Based Method for Tensor Ring Decomposition”. In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Virtual Event: PMLR, 2021, pp. 7400–7411.

SOME SKETCHING TECHNIQUES

- Randomized Algorithms
  - Sampling
    - Uniform
    - Importance
      - Based on norm
      - Based on leverage scores
  - Projection
    - Gaussian
      - Kronecker Gaussian
      - Khatri-Rao Gaussian
    - SRFT/SRHT
      - Kronecker FJLT
      - Khatri-Rao FJLT
    - CountSketch
      - TensorSketch
      - Higher-order CountSketch

SUB-SAMPLED RANDOMIZED FOURIER TRANSFORM: SRFT

Definition 1.1 (SRFT)

The **SRFT** is constructed as a matrix of the form

$$\mathbf{\Phi} = \mathbf{S}\mathcal{F}\mathbf{D},$$

where

- $\mathbf{S} \in \mathbb{R}^{m \times N}$: m random rows of the $N \times N$ identity matrix;
- $\mathcal{F} \in \mathbb{C}^{N \times N}$: (unitary) discrete Fourier transform of dimension N;
- $\mathbf{D} \in \mathbb{R}^{N \times N}$: diagonal matrix with diagonal entries drawn uniformly from $\{+1, -1\}$.

Kronecker SRFT: KSRFT

Definition 1.2 (KSRFT [BBK18; JKW21][8][9])

The KSRFT is defined as

$$\mathbf{\Phi} = \sqrt{\frac{\prod_{j=1}^{N} I_j}{m}} \mathbf{S} \left(\bigotimes_{j=1}^{N} (\mathbf{F}_j \mathbf{D}_j) \right),$$

where

- $\mathbf{S} \in \mathbb{R}^{m \times \prod_{j=1}^{N} I_j}$: m rows of the $\prod_{i=j}^{N} I_j \times \prod_{j=1}^{N} I_j$ identity matrix drawn uniformly at random with replacement from the identity matrix;
- $\mathbf{F}_j \in \mathbb{C}^{I_j \times I_j}$: (unitary) discrete Fourier transform of dimension I_j (also called DFT/FFT matrix);
- $\mathbf{D}_j \in \mathbb{R}^{I_j \times I_j}$: a diagonal matrix with independent random diagonal entries drawn uniformly from $\{+1, -1\}$ (also called random sign-flip operator).

---

[8] Casey Battaglino, Grey Ballard, and Tamara G. Kolda. "A Practical Randomized CP Tensor Decomposition". In: *SIAM J. Matrix Anal. Appl.* 39.2 (2018), pp. 876–901. DOI: `10.1137/17M1112303`.

[9] Ruhui Jin, Tamara G. Kolda, and Rachel Ward. "Faster Johnson-Lindenstrauss Transforms via Kronecker Products". In: *Inf. Inference* 10.4 (2021), pp. 1533–1562. DOI: `10.1093/imaiai/iaaa028`.

CountSketch

Definition 1.3 (CountSketch[10])

The **CountSketch** is constructed as a matrix of the form

$$\mathbf{\Phi} = \mathbf{\Omega}\mathbf{D},$$

where

- $\mathbf{\Omega} \in \mathbb{R}^{m \times N}$: a matrix with $\mathbf{\Omega}(j,i) = 1$ if $j = h(i), \forall i \in [N]$ and $\mathbf{\Omega}(j,i) = 0$ otherwise, where $h : [N] \rightarrow [m]$ is a hash map such that $\forall i \in [N]$ and $\forall j \in [m]$, $\Pr[h(i) = j] = 1/m$;
- $\mathbf{D} \in \mathbb{R}^{N \times N}$: diagonal matrix with diagonal entries drawn uniformly from $\{+1, -1\}$.

---

[10] Kenneth L. Clarkson and David P. Woodruff. “Low-Rank Approximation and Regression in Input Sparsity Time”. In: *J. ACM* 63.6 (2017), 54:1–45. DOI: 10.1145/3019134.

TensorSketch

Definition 1.4 (TensorSketch)

The TensorSketch is defined as $\mathbf{T} = \mathbf{\Omega}\mathbf{D}$, where

- $\mathbf{\Omega} \in \mathbb{R}^{m \times \prod_{j=1}^{N} I_j}$: a matrix with $\mathbf{\Omega}(j,i) = 1$ if $j = H(i)$ for all $i \in \left[\prod_{j=1}^{N} I_j\right]$ and $\mathbf{\Omega}(j,i) = 0$ otherwise;
- $\mathbf{D} \in \mathbb{R}^{\prod_{j=1}^{N} I_j \times \prod_{i=j}^{N} I_j}$: a diagonal matrix with $\mathbf{D}(i,i) = S(i)$.

In the definitions of $\mathbf{\Omega}$ and $\mathbf{D}$,

$$\begin{aligned} H &: [I_1] \times [I_2] \times \cdots \times [I_N] \to [m] : (i_1, \ldots, i_N) \mapsto \left(\sum_{n=1}^{N} (H_n(i_n) - 1) \mod m \right) + 1, \\ S &: [I_1] \times [I_2] \times \cdots \times [I_N] \to \{-1, 1\} : (i_1, \ldots, i_N) \mapsto \prod_{n=1}^{N} S_n(i_n), \end{aligned}$$

where each H_n for $n \in [N]$ is a 3-wise independent hash map that maps $[I_n] \to [m]$, and each S_n is a 4-wise independent hash map that maps $[I_n] \to \{-1, 1\}$. Recall that a hash map is k-wise independent if all the designated k keys are independent random variables.

Above we use the notation $H(i) = H(\overline{i_1 i_2 \cdots i_N})$ and $S(i) = S(\overline{i_1 i_2 \cdots i_N})$, where $\overline{i_1 i_2 \cdots i_N}$ denotes the **big-endian convention**.

PRESENTATION OUTLINE

MOTIVATION: CP-ALS

- Classical CP: CP-ALS

$$\arg\min_{\mathbf{A}_n} \|\mathbf{Z}^{(n)}\mathbf{A}_n^{\intercal} - \mathbf{X}_{(n)}^{\intercal}\|_F,$$

where $\mathbf{Z}^{(n)} = \mathbf{A}_N \odot \cdots \odot \mathbf{A}_{n+1} \odot \mathbf{A}_{n+1} \odot \cdots \odot \mathbf{A}_1$.

- Randomized CP[11]: CPRAND

$$\arg\min_{\mathbf{A}_n} \|\mathbf{S}\left(\bigotimes_{j=N,j\neq n}^{1} \mathcal{F}_j\mathbf{D}_j\right)\mathbf{Z}^{(n)}\mathbf{A}_n^{\intercal} - \mathbf{S}\left(\bigotimes_{j=N,j\neq n}^{1} \mathcal{F}_j\mathbf{D}_j\right)\mathbf{X}_{(n)}^{\intercal}\|_F,$$

where $\hat{\mathbf{Z}}^{(n)} = \left(\bigotimes_{j=N,j\neq n}^{1} \mathcal{F}_j\mathbf{D}_j\right)\mathbf{Z}^{(n)} = \bigodot_{j=N,j\neq n}^{1}(\mathcal{F}_j\mathbf{D}_j\mathbf{A}_j)$.

---

[11] Casey Battaglino, Grey Ballard, and Tamara G. Kolda. "A Practical Randomized CP Tensor Decomposition". In: *SIAM J. Matrix Anal. Appl.* 39.2 (2018), pp. 876–901. DOI: 10.1137/17M1112303.

IDEAS

- Original problem: TR-ALS

$$\arg\min_{\mathbf{G}_{n(2)}} \|\mathbf{G}_{[2]}^{\neq n}\mathbf{G}_{n(2)}^{\intercal} - \mathbf{X}_{[n]}^{\intercal}\|_F. \tag{2.1}$$

- Reduced problem: Sketched TR-ALS

$$\arg\min_{\mathbf{G}_{n(2)}} \left\|\mathcal{S}\mathbf{G}_{[2]}^{\neq n}\mathbf{G}_{n(2)}^{\intercal} - \mathcal{S}\mathbf{X}_{[n]}^{\intercal}\right\|_F.$$

- Ideas
    - Avoid forming $\mathcal{S}$ explicitly.
    - Avoid forming $\mathbf{G}_{[2]}^{\neq n}$ explicitly.
    - Avoid the classical matrix multiplication of $\mathcal{S}$ and $\mathbf{G}_{[2]}^{\neq n}$ directly.

New Findings

- Mixing the rows of $\mathbf{G}^{\neq n}_{[2]}$ is equivalent to mixing the lateral slides of $\mathcal{G}^{\neq n}$, i.e., $\mathcal{S}\mathbf{G}^{\neq n}_{[2]} = (\mathcal{G}^{\neq n} \times_2 \mathcal{S})_{[2]}$.
- $\mathcal{G}^{\neq n}$ may be written as a Kronecker-like or KR-like product of TR-cores.

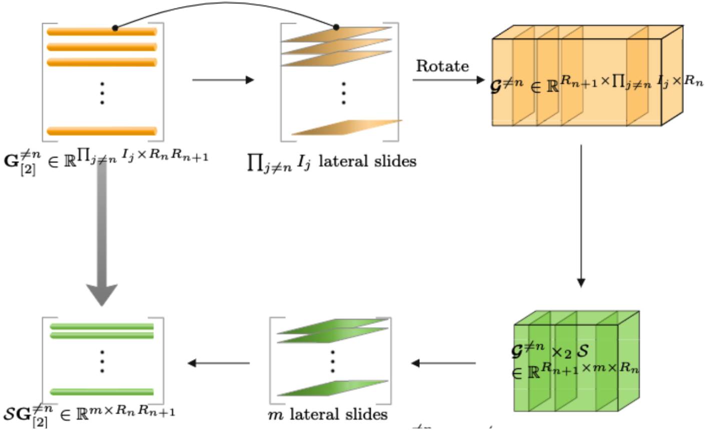


Figure: Illustration of the process for obtaining $\mathcal{S}\mathbf{G}^{\neq n}_{[2]}$ via $\mathcal{G}^{\neq n} \times_2 \mathcal{S}$

NEW DEFINITION: SUBCHAIN PRODUCT

Definition 2.1

Let $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times J_1 \times K}$ and $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{K \times J_2 \times I_2}$ be two 3rd-order tensors, and $\mathbf{A}(j_1)$ and $\mathbf{B}(j_2)$ be the j_1-th and j_2-th lateral slices of $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$, respectively. The mode-2 **subchain product** of $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$ is a tensor of size $I_1 \times J_1 J_2 \times I_2$ denoted by $\boldsymbol{\mathcal{A}} \boxtimes_2 \boldsymbol{\mathcal{B}}$ and defined as

$$(\boldsymbol{\mathcal{A}} \boxtimes_2 \boldsymbol{\mathcal{B}})(\overline{j_1 j_2}) = \boldsymbol{\mathcal{A}}(j_1)\boldsymbol{\mathcal{B}}(j_2).$$

That is, with respect to the correspondence on indices, the lateral slices of $\boldsymbol{\mathcal{A}} \boxtimes_2 \boldsymbol{\mathcal{B}}$ are the classical matrix products of the lateral slices of $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$. The mode-1 and mode-3 subchain products can be defined similarly.

Therefore, $\boldsymbol{\mathcal{G}}^{\neq n}$ can be rewritten as

$$\boldsymbol{\mathcal{G}}^{\neq n} = \boldsymbol{\mathcal{G}}_{n+1} \boxtimes_2 \cdots \boxtimes_2 \boldsymbol{\mathcal{G}}_N \boxtimes_2 \boldsymbol{\mathcal{G}}_1 \boxtimes_2 \cdots \boxtimes_2 \boldsymbol{\mathcal{G}}_{n-1}. \tag{2.2}$$

New Proposition

$$\mathcal{S}\mathbf{G}_{[2]}^{\neq n} = (\boldsymbol{\mathcal{G}}^{\neq n} \times_2 \mathcal{S})_{[2]}$$
$$= ((\boldsymbol{\mathcal{G}}_{n+1} \boxtimes_2 \cdots \boxtimes_2 \boldsymbol{\mathcal{G}}_N \boxtimes_2 \boldsymbol{\mathcal{G}}_1 \boxtimes_2 \cdots \boxtimes_2 \boldsymbol{\mathcal{G}}_{n-1}) \times_2 \mathcal{S})_{[2]}$$

Proposition 2.2

Let $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times J_1 \times K}$ *and* $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{K \times J_2 \times I_2}$ *be two 3rd-order tensors, and* $\mathbf{A} \in \mathbb{R}^{R_1 \times J_1}$ *and* $\mathbf{B} \in \mathbb{R}^{R_2 \times J_2}$ *be two matrices. Then*

$$(\boldsymbol{\mathcal{A}} \times_2 \mathbf{A}) \boxtimes_2 (\boldsymbol{\mathcal{B}} \times_2 \mathbf{B}) = (\boldsymbol{\mathcal{A}} \boxtimes_2 \boldsymbol{\mathcal{B}}) \times_2 (\mathbf{B} \otimes \mathbf{A}).$$

IDEA ON ALGORITHM: SELECT A SUITABLE "$\mathcal{S}$"

- Let $\mathcal{S} = \mathbf{S}\mathcal{F}\mathbf{D}$, where

$$\mathcal{F} = \left(\bigotimes_{j=n-1,\cdots,1,N,\cdots,n+1} \mathcal{F}_j \right), \mathbf{D} = \left(\bigotimes_{j=n-1,\cdots,1,N,\cdots,n+1} \mathbf{D}_j \right).$$

That is,

$$\mathcal{S} = \mathbf{S} \left(\bigotimes_{j=n-1,\cdots,1,N,\cdots,n+1} \mathcal{F}_j \mathbf{D}_j \right).$$

- Thus,

$$\arg\min_{\mathbf{G}_{n(2)}} \left\| \mathbf{S}\mathcal{F}\mathbf{D}\mathbf{G}_{[2]}^{\neq n} \mathbf{G}_{n(2)}^{\intercal} - \mathbf{S}\mathcal{F}\mathbf{D}\mathbf{X}_{[n]}^{\intercal} \right\|_F, \tag{2.3}$$

THE FIRST TERM IN (2.3): $\mathbf{S}\mathcal{F}\mathbf{D}\mathbf{G}_{[2]}^{\neq n}$

- Mixing step. Using Proposition 2.2 and (2.2)

$$\begin{aligned}\hat{\boldsymbol{\mathcal{G}}}^{\neq n} &= \boldsymbol{\mathcal{G}}^{\neq n} \times_2 \mathcal{F}\mathbf{D} \\ &= (\boldsymbol{\mathcal{G}}_{n+1} \times_2 (\mathcal{F}_{n+1}\mathbf{D}_{n+1}))\boxtimes_2 \\ &\cdots \boxtimes_2 (\boldsymbol{\mathcal{G}}_N \times_2 (\mathcal{F}_N\mathbf{D}_N)) \boxtimes_2 (\boldsymbol{\mathcal{G}}_1 \times_2 (\mathcal{F}_1\mathbf{D}_1))\boxtimes_2 \\ &\cdots \boxtimes_2 (\boldsymbol{\mathcal{G}}_{n-1} \times_2 (\mathcal{F}_{n-1}\mathbf{D}_{n-1})).\end{aligned}$$

  i.e. $\mathcal{F}\mathbf{D}\mathbf{G}_{[2]}^{\neq n} = \hat{\mathbf{G}}_{[2]}^{\neq n}$.

- Sampling step. According to the sampling method in Algorithm 4 (SST)[12], we have

$$\begin{aligned}\hat{\boldsymbol{\mathcal{G}}}^{\neq n} \times_2 \mathbf{S} = &(\boldsymbol{\mathcal{G}}_{n+1} \times_2 (\mathbf{S}_{n+1}\mathcal{F}_{n+1}\mathbf{D}_{n+1}))\boxast_2 \\ &\cdots \boxast_2 (\boldsymbol{\mathcal{G}}_N \times_2 (\mathbf{S}_N\mathcal{F}_N\mathbf{D}_N)) \boxast_2 (\boldsymbol{\mathcal{G}}_1 \times_2 (\mathbf{S}_1\mathcal{F}_1\mathbf{D}_1))\boxast_2 \\ &\cdots \boxast_2 (\boldsymbol{\mathcal{G}}_{n-1} \times_2 (\mathbf{S}_{n-1}\mathcal{F}_{n-1}\mathbf{D}_{n-1})),\end{aligned}$$

  using Proposition 3.3, we have $\mathbf{S} = \left(\bigodot_{\substack{j=n-1,\cdots,1,\\ N,\cdots,n+1}} \mathbf{S}_j^{\intercal}\right)^{\intercal}$

---

[12] Osman Asif Malik and Stephen Becker. "A Sampling-Based Method for Tensor Ring Decomposition". In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Virtual Event: PMLR, 2021, pp. 7400–7411.

THE SECOND TERM IN (2.3): $\mathbf{S}\mathcal{F}\mathbf{D}\mathbf{X}_{[n]}^{\intercal}$

- Let $\hat{\boldsymbol{\mathcal{X}}} = \boldsymbol{\mathcal{X}} \times_1 \mathcal{F}_1\mathbf{D}_1 \times_2 \mathcal{F}_2\mathbf{D}_2 \cdots \times_N \mathcal{F}_N\mathbf{D}_N$.
- The second term is equivalent to

$$\mathbf{S}\hat{\mathbf{X}}_{[n]}^{\intercal}(\mathbf{D}_n\mathcal{F}_n^*)^{\intercal}.$$

Rewrite (2.3) as

$$\arg\min_{\mathbf{G}_{n(2)}} \| \left(\mathbf{S}\hat{\mathbf{G}}_{[2]}^{\neq n}\right)\mathbf{G}_{n(2)}^{\intercal} - \left(\mathbf{S}\hat{\mathbf{X}}_{[n]}^{\intercal}\right)(\mathbf{D}_n\mathcal{F}_n^*)^{\intercal}\|_F.$$

PROPOSED ALGORITHM: TR-KSRFT-ALS

Algorithm 5 TR-KSRFT-ALS (Proposal)

```
function $\{\mathcal{G}_n\}_{n=1}^N$ = TR-KSRFT-ALS($\mathcal{X}, R_1, \cdots, R_N, m$)
        ▷ $\mathcal{G}_n \in \mathbb{R}^{R_n \times I_n \times R_{n+1}}, n \in [N]$; $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$
        ▷ $R_1, \cdots, R_N$ are the TR-ranks
        ▷ $m$ is the uniform sampling size
    Initialize cores $\mathcal{G}_2, \cdots, \mathcal{G}_N$
    Define random sign-flip operators $\mathbf{D}_j$ and FFT matrices $\mathbf{F}_j$, for $j \in [N]$
    Mix cores: $\hat{\mathcal{G}}_n \leftarrow \mathcal{G}_n \times_2 (\mathbf{F}_n\mathbf{D}_n)$, for $n = 2, \cdots, N$
    Mix tensor: $\hat{\mathcal{X}} \leftarrow \mathcal{X} \times_1 (\mathbf{F}_1\mathbf{D}_1) \times_2 (\mathbf{F}_2\mathbf{D}_2) \cdots \times_N (\mathbf{F}_N\mathbf{D}_N)$
    repeat
        for $n = 1, \cdots, N$ do
            Define sampling operator $\mathbf{S} \in \mathbb{R}^{m \times \prod_{j \neq n} I_j}$
            Retrieve idxs from $\mathbf{S}$
            $\hat{\mathcal{G}}_S^{\neq n}$ = SST(idxs, $\hat{\mathcal{G}}_{n+1}, \cdots, \hat{\mathcal{G}}_N, \hat{\mathcal{G}}_1, \cdots, \hat{\mathcal{G}}_{n-1}$)
            $\hat{\mathbf{X}}_{S[n]}^\intercal \leftarrow \mathbf{S}\hat{\mathbf{X}}_{[n]}^\intercal (\mathbf{D}_n\mathbf{F}_n^*)^\intercal$
            Update $\mathcal{G}_n = \arg\min_{\mathcal{Z}} \|\hat{\mathbf{G}}_{S[2]}^{\neq n}\mathbf{Z}_{(2)}^\intercal - \hat{\mathbf{X}}_{S[n]}^\intercal\|_F$ subject to $\mathcal{G}_n$ being real-valued
            $\hat{\mathcal{G}}_n \leftarrow \mathcal{G}_n \times_2 (\mathbf{F}_n\mathbf{D}_n)$
        end for
    until termination criteria met
    return $\mathcal{G}_1, \cdots, \mathcal{G}_N$
end function
```

FURTHER IMPROVEMENT: PREMIX

- Recall that

$$\arg\min_{\mathbf{G}_{n(2)}} \| \left(\mathbf{S}\hat{\mathbf{G}}_{[2]}^{\neq n}\right) \mathbf{G}_{n(2)}^{\intercal} - \left(\mathbf{S}\hat{\mathbf{X}}_{[n]}^{\intercal}\right) (\mathbf{D}_n \mathcal{F}_n^*)^{\intercal} \|_F.$$

- Rewrite it as

$$\arg\min_{\mathbf{G}_{n(2)}} \| \left(\mathbf{S}\hat{\mathbf{G}}_{[2]}^{\neq n}\right) \mathbf{G}_{n(2)}^{\intercal} (\mathcal{F}_n \mathbf{D}_n)^{\intercal} - \mathbf{S}\hat{\mathbf{X}}_{[n]}^{\intercal} \|_F,$$

- Let $\hat{\mathbf{G}}_{n(2)} = \mathcal{F}_n \mathbf{D}_n \mathbf{G}_{n(2)}$

$$\arg\min_{\hat{\mathbf{G}}_{n(2)}} \| \left(\mathbf{S}\hat{\mathbf{G}}_{[2]}^{\neq n}\right) \hat{\mathbf{G}}_{n(2)}^{\intercal} - \left(\mathbf{S}\hat{\mathbf{X}}_{[n]}^{\intercal}\right) \|_F.$$

- Solve the problem above to get $\hat{\mathcal{G}}_n$ first and then recover the original cores $\mathcal{G}_n$.

Algorithm: TR-SRFT-ALS-Premix

Algorithm 6 TR-KSRFT-ALS-Premix (Proposal)

```
function $\{\mathcal{G}_n\}_{n=1}^{N}$ = TR-KSRFT-ALS-Premix($\mathcal{X}, R_1, \cdots, R_N, m$)
                                   ▷ $\mathcal{G}_n \in \mathbb{C}^{R_n \times I_n \times R_{n+1}}, n \in [N]$; $\mathcal{X} \in \mathbb{C}^{I_1 \times \cdots \times I_N}$
                                   ▷ $R_1, \cdots, R_N$ are the TR-ranks
                                   ▷ $m$ is the uniform sampling size
    Define random sign-flip operators $\mathbf{D}_j$ and FFT matrices $\mathbf{F}_j$, for $j \in [N]$
    Mix tensor: $\hat{\mathcal{X}} \leftarrow \mathcal{X} \times_1 (\mathbf{F}_1\mathbf{D}_1) \times_2 (\mathbf{F}_2\mathbf{D}_2) \cdots \times_N (\mathbf{F}_N\mathbf{D}_N)$
    Initialize cores $\hat{\mathcal{G}}_2, \cdots, \hat{\mathcal{G}}_N$
    repeat
        for $n = 1, \cdots, N$ do
            Define sampling operator $\mathbf{S} \in \mathbb{R}^{m \times \prod_{j \neq n} I_j}$
            Retrieve idxs from $\mathbf{S}$
            $\hat{\mathcal{G}}_S^{\neq n}$ = SST(idxs, $\hat{\mathcal{G}}_{n+1}, \cdots, \hat{\mathcal{G}}_N, \hat{\mathcal{G}}_1, \cdots, \hat{\mathcal{G}}_{n-1}$)
            $\hat{\mathbf{X}}_{S[n]}^{\intercal} \leftarrow \mathbf{S}\hat{\mathbf{X}}_{[n]}^{\intercal}$
            Update $\hat{\mathcal{G}}_n = \arg\min_{\mathcal{Z}} \|\hat{\mathbf{G}}_{S[2]}^{\neq n}\mathbf{Z}_{(2)}^{\intercal} - \hat{\mathbf{X}}_{S[n]}^{\intercal}\|_F$
        end for
    until termination criteria met
    for $n = 1, \cdots, N$ do
        Unmix cores: $\mathcal{G}_n \leftarrow \hat{\mathcal{G}}_n \times_2 (\mathbf{D}_n\mathbf{F}_n^*)$
    end for
    return $\mathcal{G}_1, \cdots, \mathcal{G}_N$
end function
```

SOME REMARKS

- Like the algorithms for CP decomposition, i.e., CPRAND[13], but with a new tensor product and property;
- Compared with TR-ALS-Sampled[14], our method may work better for some special data, such as for the data with core tensors may include outliers;
- $\mathbf{F}_j\mathbf{D}_j$ can be any suitable randomized matrices: CountSketch, rTR-ALS[15], unified form.

---

[13] Casey Battaglino, Grey Ballard, and Tamara G. Kolda. "A Practical Randomized CP Tensor Decomposition". In: *SIAM J. Matrix Anal. Appl.* 39.2 (2018), pp. 876–901. DOI: `10.1137/17M1112303`.

[14] Osman Asif Malik and Stephen Becker. "A Sampling-Based Method for Tensor Ring Decomposition". In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Virtual Event: PMLR, 2021, pp. 7400–7411.

[15] Longhao Yuan et al. "Randomized Tensor Ring Decomposition and Its Application to Large-Scale Data Reconstruction". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Brighton Conference Centre Brighton, U.K.: IEEE, 2019, pp. 2127–2131.

ILLUSTRATION

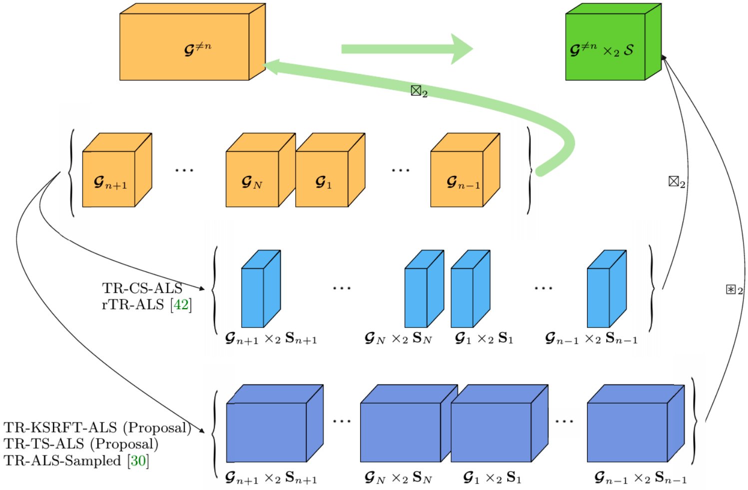


Figure: Illustration of how to efficiently construct $\mathcal{G}^{\neq n} \times_2 \mathcal{S}$ by sketching the core tensors.

THEORETICAL ANALYSIS: TR-KSRFT-ALS & TR-KSRFT-ALS-PREMIX

Theorem 2.3

For the matrices $\mathbf{G}_{[2]}^{\neq n}$ *and* $\mathbf{X}_{[n]}^{\intercal}$ *in* (2.1), *denote* $rank(\mathbf{G}_{[2]}^{\neq n}) = r$ *and fix* $\varepsilon, \eta \in (0,1)$ *such that* $\prod_{j\neq n} I_j \lesssim 1/\varepsilon^r$ *with* $r \geq 2$. *Then a sketching matrix* $\mathcal{S} \in \mathbb{C}^{m\times \prod_{j\neq n} I_j}$ *used in Algorithm 5 or 6 with*

$$m = \mathcal{O}\left(\varepsilon^{-1} r^{2(N-1)} \log^{2N-3}(\frac{r}{\varepsilon}) \log^4(\frac{r}{\varepsilon}\log(\frac{r}{\varepsilon})) \log \prod_{j\neq n} I_j\right)$$

is sufficient to output

$$\tilde{\mathbf{G}}_{n(2)}^{\intercal} = \underset{\mathbf{G}_{n(2)}^{\intercal}\in\mathbb{R}^{R_n R_{n+1}\times I_n}}{\arg\min} \|\mathcal{S}\mathbf{G}_{[2]}^{\neq n}\mathbf{G}_{n(2)}^{\intercal} - \mathcal{S}\mathbf{X}_{[n]}^{\intercal}\|_F,$$

such that

$$\mathbf{Pr}\left(\|\mathbf{G}_{[2]}^{\neq n}\tilde{\mathbf{G}}_{n(2)}^{\intercal} - \mathbf{X}_{[n]}^{\intercal}\|_F = (1\pm\mathcal{O}(\varepsilon)) \min \|\mathbf{G}_{[2]}^{\neq n}\mathbf{G}_{n(2)}^{\intercal} - \mathbf{X}_{[n]}^{\intercal}\|_F\right)$$
$$\geq 1 - \eta - 2^{-\Omega(\log \prod_{j\neq n} I_j)}.$$

PRESENTATION OUTLINE

TensorSketch for Subchain Product

Definition 3.1

The TensorSketch is defined as $\mathbf{T} = \mathbf{\Omega}\mathbf{D}$, where

- $\mathbf{\Omega} \in \mathbb{R}^{m \times \prod_{j=1}^{N} I_j}$: a matrix with $\mathbf{\Omega}(j,i) = 1$ if $j = H(i)$ for all $i \in \left[\prod_{j=1}^{N} I_j\right]$ and $\mathbf{\Omega}(j,i) = 0$ otherwise;
- $\mathbf{D} \in \mathbb{R}^{\prod_{j=1}^{N} I_j \times \prod_{i=j}^{N} I_j}$: a diagonal matrix with $\mathbf{D}(i,i) = S(i)$.

In the definitions of $\mathbf{\Omega}$ and $\mathbf{D}$,

$$H \quad : \quad [I_1] \times [I_2] \times \cdots \times [I_N] \to [m] : (i_1, \ldots, i_N) \mapsto \left(\sum_{n=1}^{N} (H_n(i_n) - 1) \mod m \right) + 1,$$

$$S \quad : \quad [I_1] \times [I_2] \times \cdots \times [I_N] \to \{-1, 1\} : (i_1, \ldots, i_N) \mapsto \prod_{n=1}^{N} S_n(i_n),$$

where each H_n for $n \in [N]$ is a 3-wise independent hash map that maps $[I_n] \to [m]$, and each S_n is a 4-wise independent hash map that maps $[I_n] \to \{-1, 1\}$. Recall that a hash map is k-wise independent if all the designated k keys are independent random variables.

Above we use the notation $H(i) = H(\overline{i_1 i_2 \cdots i_N})$ and $S(i) = S(\overline{i_1 i_2 \cdots i_N})$, where $\overline{i_1 i_2 \cdots i_N}$ denotes the **little-endian convention**.

Related Works

- Osman Asif Malik and Stephen Becker. “Fast Randomized Matrix and Tensor Interpolative Decomposition Using CountSketch”. In: *Adv. Comput. Math.* 46 (2020), p. 76. DOI: `10.1007/s10444-020-09816-9`
  - $\mathbf{P} = \mathbf{A}^{(1)} \odot \mathbf{A}^{(2)} \odot \cdots \odot \mathbf{A}^{(N)}$ for $n \in [N]$.
  - $\mathbf{TP} = \mathrm{FFT}^{-1}\left(\circledast_{n=1}^{N} \mathrm{FFT}\left(\mathbf{S}^{(n)}\mathbf{A}^{(n)}\right)\right)$.
- Osman Asif Malik and Stephen Becker. “Low-Rank Tucker Decomposition of Large Tensors Using TensorSketch”. In: *Advances in Neural Information Processing Systems.* Vol. 31. Montréal, Canada: Curran Associates, Inc., 2018, pp. 10117–10127
  - $\mathbf{P} = \mathbf{A}^{(1)} \otimes \mathbf{A}^{(2)} \otimes \cdots \otimes \mathbf{A}^{(N)}$ for $n \in [N]$.
  - $\mathbf{TP} = \mathrm{FFT}^{-1}\left(\left(\bigodot_{n=1}^{N}\left(\mathrm{FFT}\left(\mathbf{S}^{(n)}\mathbf{A}^{(n)}\right)\right)^{\mathsf{T}}\right)^{\mathsf{T}}\right)$.
- Rasmus Pagh. “Compressed Matrix Multiplication”. In: *ACM Trans. Comput. Theory* 5.3 (2013), pp. 1–17. DOI: `10.1145/2493252.2493254`
- Huaian Diao et al. “Sketching for Kronecker Product Regression and P-splines”. In: *International Conference on Artificial Intelligence and Statistics.* Vol. 84. Playa Blanca, Lanzarote, Canary Islands: PMLR, 2018, pp. 1299–1308
- What about $\mathbf{TG}_{[2]}^{\neq n}$? Recall that

$$\boldsymbol{\mathcal{G}}^{\neq n} = \boldsymbol{\mathcal{G}}_{n+1} \boxtimes_2 \cdots \boxtimes_2 \boldsymbol{\mathcal{G}}_N \boxtimes_2 \boldsymbol{\mathcal{G}}_1 \boxtimes_2 \cdots \boxtimes_2 \boldsymbol{\mathcal{G}}_{n-1}.$$

NEW DEFINITION: SLICES-HADAMARD PRODUCT

Definition 3.2

Let $\mathcal{A} \in \mathbb{R}^{I_1 \times J \times K}$ and $\mathcal{B} \in \mathbb{R}^{K \times J \times I_2}$ be two 3rd-order tensors, and $\mathbf{A}(j)$ and $\mathbf{B}(j)$ are the j-th lateral slices of $\mathcal{A}$ and $\mathcal{B}$, respectively. The mode-2 **slices-Hadamard product** of $\mathcal{A}$ and $\mathcal{B}$ is a tensor of size $I_1 \times J \times I_2$ denoted by $\mathcal{A} \circledast_2 \mathcal{B}$ and defined as

$$(\mathcal{A} \boxast_2 \mathcal{B})(j) = \mathcal{A}(j)\mathcal{B}(j).$$

That is, the j-th lateral slice of $\mathcal{A} \boxast_2 \mathcal{B}$ is the classical matrix product of the j-th lateral slices of $\mathcal{A}$ and $\mathcal{B}$. The mode-1 and mode-3 slices-Hadamard product can be defined similarly.

New Propositions

Proposition 3.3

Let $\mathcal{A} \in \mathbb{R}^{I_1 \times J_1 \times K}$ *and* $\mathcal{B} \in \mathbb{R}^{K \times J_2 \times I_2}$ *be two 3rd-order tensors, and* $\mathbf{A} \in \mathbb{R}^{M \times J_1}$ *and* $\mathbf{B} \in \mathbb{R}^{M \times J_2}$ *be two matrices. Then*

$$(\mathcal{A} \times_2 \mathbf{A}) \boxast_2 (\mathcal{B} \times_2 \mathbf{B}) = (\mathcal{A} \boxtimes_2 \mathcal{B}) \times_2 (\mathbf{B}^\intercal \odot \mathbf{A}^\intercal)^\intercal .$$

Proposition 3.4

Let $\mathbf{S}_n = \mathbf{\Omega}_n \mathbf{D}_n \in \mathbb{R}^{m \times I_n}$*, where* $\mathbf{\Omega}_n \in \mathbb{R}^{m \times I_n}$ *and* $\mathbf{D}_n \in \mathbb{R}^{I_n \times I_n}$ *are defined based on* H_n *and* S_n *in Definition 3.1, respectively. Let* $\mathbf{T} \in \mathbb{R}^{m \times \prod_{j=1}^{N} I_N}$ *be defined in Definition 3.1 and* $\mathcal{P} = \mathcal{A}^{(1)} \boxtimes_2 \mathcal{A}^{(2)} \boxtimes_2 \cdots \boxtimes_2 \mathcal{A}^{(N)}$ *with* $\mathcal{A}^{(n)} \in \mathbb{R}^{R_n \times I_n \times R_{n+1}}$ *for* $n \in [N]$*. Then*

$$\mathcal{P} \times_2 \mathbf{T} = \mathrm{FFT}^{-1}\left(\boxast_2{}_{n=1}^{N} \mathrm{FFT}\left(\mathcal{A}^{(n)} \times_2 \mathbf{S}_n, [\,], 2 \right), [\,], 2 \right).$$

ALGORITHM: TR-TS-ALS

Algorithm 7 TR-TS-ALS (Proposal)

```
function $\{\boldsymbol{\mathcal{G}}_n\}_{n=1}^N$ = TR-TS-ALS($\boldsymbol{\mathcal{X}}, R_1, \cdots, R_N, m$)
        ▷ $\boldsymbol{\mathcal{G}}_n \in \mathbb{R}^{R_n \times I_n \times R_{n+1}}, n \in [N]$; $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$
        ▷ $R_1, \cdots, R_N$ are the TR-ranks
        ▷ $m$ is the embedding size
    Define $\mathbf{S}_j$, i.e., the CountSketch, based on $H_n$ and $S_n$ in Definition 3.1, for $j \in [N]$
    for $n = 1, \cdots, N$ do
        Compute the sketch of $\mathbf{X}_{[n]}^\mathsf{T}$: $\hat{\mathbf{X}}_{[n]}^\mathsf{T} \leftarrow \mathbf{T}_{\neq n}\mathbf{X}_{[n]}^\mathsf{T}$
    end for
    Initialize cores $\boldsymbol{\mathcal{G}}_2, \cdots, \boldsymbol{\mathcal{G}}_N$
    repeat
        for $n = 1, \cdots, N$ do
            Compute $\hat{\boldsymbol{\mathcal{G}}}^{\neq n} = \mathrm{FFT}^{-1}\left(\boxast_{2\ j=n+1,\cdots,N}^{1,\cdots,n-1} \mathrm{FFT}\left(\boldsymbol{\mathcal{G}}_j \times_2 \mathbf{S}_j, [\,], 2\right), [\,], 2\right)$
            Update $\boldsymbol{\mathcal{G}}_n = \arg\min_{\boldsymbol{\mathcal{Z}}} \|\hat{\mathbf{G}}_{[2]}^{\neq n}\mathbf{Z}_{(2)}^\mathsf{T} - \hat{\mathbf{X}}_{[n]}^\mathsf{T}\|_F$
        end for
    until termination criteria met
    return $\boldsymbol{\mathcal{G}}_1, \cdots, \boldsymbol{\mathcal{G}}_N$
end function
```

THEORETICAL ANALYSIS: TR-TS-ALS

Theorem 3.5

For the matrices $\mathbf{G}_{[2]}^{\neq n}$ *and* $\mathbf{X}_{[n]}^{\intercal}$ *in* (2.1), *fix* $\varepsilon, \eta \in (0,1)$. *Then a TensorSketch* $\mathbf{T}_{\neq n}$ *used in Algorithm 7 with*

$$m = \mathcal{O}\left(((R_n R_{n+1} \cdot 3^{N-1})((R_n R_{n+1} + 1/\varepsilon^2)/\eta\right),$$

is sufficient to output

$$\tilde{\mathbf{G}}_{n(2)}^{\intercal} = \underset{\mathbf{G}_{n(2)}^{\intercal} \in \mathbb{R}^{R_n R_{n+1} \times I_n}}{\arg\min} \|\mathbf{T}_{\neq n} \mathbf{G}_{[2]}^{\neq n} \mathbf{G}_{n(2)}^{\intercal} - \mathbf{T}_{\neq n} \mathbf{X}_{[n]}^{\intercal}\|_F,$$

such that

$$\mathbf{Pr}\left(\|\mathbf{G}_{[2]}^{\neq n} \tilde{\mathbf{G}}_{n(2)}^{\intercal} - \mathbf{X}_{[n]}^{\intercal}\|_F = (1 \pm \mathcal{O}(\varepsilon)) \min \|\mathbf{G}_{[2]}^{\neq n} \mathbf{G}_{n(2)}^{\intercal} - \mathbf{X}_{[n]}^{\intercal}\|_F\right) \geq 1 - \eta.$$

PRESENTATION OUTLINE

EXPERIMENTAL OUTLINE

- Baselines
    - TR-ALS
    - TR-ALS-Sampled
- Synthetic data
    - The 1st experiment: low rank tensor
    - The 2nd experiment: sparse tensor
    - The 3rd experiment: sparse tensor with high coherence
    - The 4th experiment: complex tensor
- Real data
    - Indian Pines
    - SalinasA.
    - C1-vertebrae
    - Uber

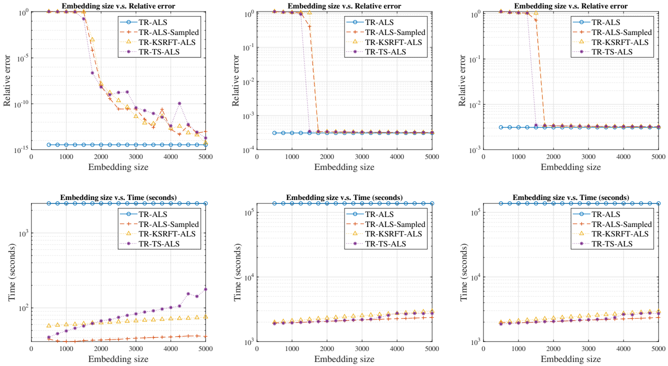
THE FIRST EXPERIMENT: SETUPS

- generate_low_rank_tensor($sz, ranks, noise, large\_elem$)[16]
    - Create 3 cores of size $R_{true} \times I \times R_{true}$ with entries drawn independently from a standard normal distribution.
    - Set $large\_elem$ to increase the coherence;
    - $R_{true} = 10$;
    - sz = [I,I,I]= [500,500,500];
    - $ranks = R$;
    - $large\_elem$ = 20;
    - $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{X}}_{ture} + noise \left(\frac{\|\boldsymbol{\mathcal{X}}_{ture}\|}{\|\boldsymbol{\mathcal{N}}\|} \right) \boldsymbol{\mathcal{N}}$.

---

[16] Osman Asif Malik and Stephen Becker. "A Sampling-Based Method for Tensor Ring Decomposition". In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Virtual Event: PMLR, 2021, pp. 7400–7411.

THE FIRST EXPERIMENT: RESULTS



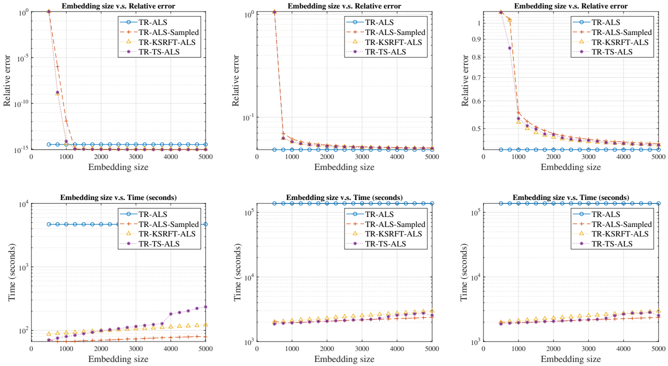
(a) $noise = 0$ **(b)** $noise = 0.01$ **(c)** $noise = 0.1$

Figure: Embedding sizes v.s. relative errors and running time (seconds) of the first synthetic experiment with true and target ranks $R_{true} = R = 10$ and different noises.

THE SECOND EXPERIMENT: SETUPS

- generate_sparse_low_rank_tensor($sz, ranks, density, noise$)
  - Create 3 cores of size $R_{true} \times I \times R_{true}$ with non-zero entries drawn from a standard normal distribution;
  - $R_{true} = 10$;
  - sz = [I,I,I]= [500,500,500];
  - $ranks = R$;
  - $density$ = 0.05;

THE SECOND EXPERIMENT: RESULTS



(a) $noise = 0$ **(b)** $noise = 0.01$ **(c)** $noise = 0.1$

Figure: Embedding sizes v.s. relative errors and running time (seconds) of the second synthetic experiment with true and target ranks $R_{true} = R = 10$ and different noises.

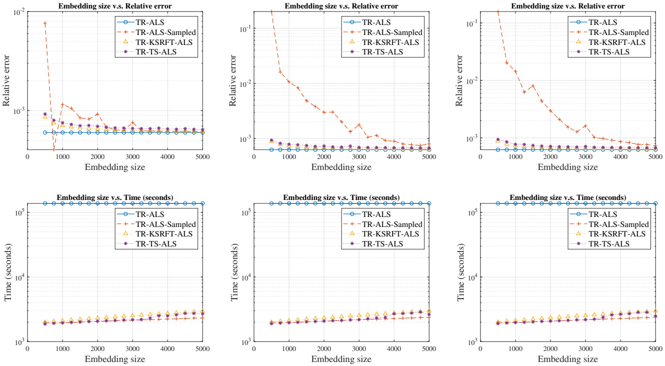
THE THIRD EXPERIMENT: SETUPS

- generate_sptr_tensor($sz, ranks, noise, spread, magnitude$)[17]
    - Create 3 cores of size $R_{true} \times I \times R_{true}$ with entries drawn independently from a standard normal distribution;
    - $spread$: How many non-zeros elements are added to each of these first three columns;
    - $magnitude$: Those non-zero elements are chosen;
    - $R_{true} = 10$;
    - sz = [I,I,I]= [500,500,500];
    - $ranks = R$;

---

[17] Brett W. Larsen and Tamara G. Kolda. "Practical Leverage-Based Sampling for Low-Rank Tensor Decomposition". In: *arXiv preprint arXiv:2006.16438* (2020).

THE THIRD EXPERIMENT: RESULTS



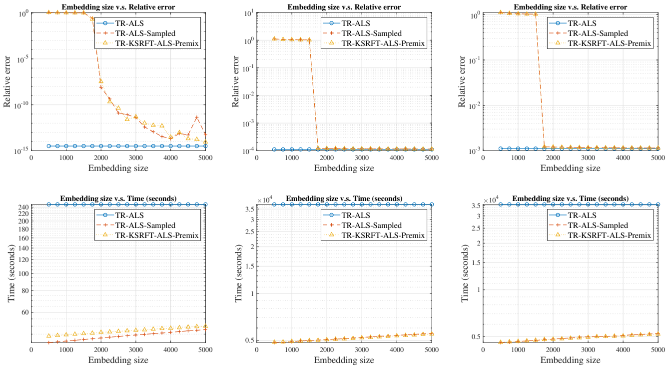
(a) $noise = 0$ **(b)** $noise = 0.01$ **(c)** $noise = 0.1$

Figure: Embedding sizes v.s. relative errors and running time (seconds) of the third synthetic experiment with true and target ranks $R_{true} = R = 10$ and different noises.

THE FORTH EXPERIMENT: SETUPS

- generate_complex_low_rank_tensor($sz, ranks, noise, large\_elem$)
    - Create 3 cores of size $R_{true} \times I \times R_{true}$ with entries drawn independently from a standard normal distribution and add imaginary part;
    - Set $large\_elem$ to increase the coherence;
    - $R_{true} = 10$;
    - sz = [I,I,I]= [500,500,500];
    - $ranks = R$;
    - $large\_elem$ = 20;

The Forth Experiment: Results



(a) $noise = 0$ **(b)** $noise = 0.01$ **(c)** $noise = 0.1$

Figure: Embedding sizes v.s. relative errors and running time (seconds) of the fourth synthetic experiment with true and target ranks $R_{true} = R = 10$ and different noises.

REAL DATA: BRIEF INFORMATION

Table: Size and type of real datasets.

Dataset	Size	Type
Indian Pines	$145 \times 145 \times 220$	Hyperspectral
SalinasA.	$83 \times 86 \times 224$	Hyperspectral
C1-vertebrae	$512 \times 512 \times 47$	CT Images
Uber.Hour	$183 \times 1140 \times 1717$	Sparse
Uber.Date	$24 \times 1140 \times 1717$	Sparse

REAL DATA: RESULTS

Method	Indian Pines ($R = 20$)			SalinasA. ($R = 15$)			C1-vertebrae ($R = 25$)		
	Error	Time	num	Error	Time	num	Error	Time	num
TR-ALS	0.0263	32.9536		0.0066	4.0225		0.0804	409.7951	
TR-ALS-Sampled	0.0289	13.7424	120	0.0069	2.4166	54	0.0882	128.3391	228
TR-SRFT-ALS	0.0289	12.3571	53	0.0073	1.8510	23	0.0883	101.7646	88
TR-SRFT-ALS (No pre-time)		11.9446			1.7093			101.4037	
TR-TS-ALS	0.0289	12.0229	73	0.0073	2.2868	30	0.0883	156.5089	217

Method	Uber.Hour ($R = 15$)			Uber.Date ($R = 18$)		
	Error	Time	num	Error	Time	num
TR-ALS	0.7530	869.1631		0.3864	1452.1900	
TR-ALS-Sampled	0.8246	64.7240	230	0.4226	159.1936	320
TR-SRFT-ALS	0.8272	39.0307	40	0.4246	51.3584	46
TR-SRFT-ALS (No pre-time)		21.9817			48.9433	
TR-TS-ALS	0.8274	45.3829	47	0.4239	113.8542	147

PRESENTATION OUTLINE

CONCLUSIONS

- We propose two randomized algorithms for TR decomposition, TR-SRFT-ALS and TR-TS-ALS.
- We propose two new tensor products and find their interesting properties.
- Numerical experiments are provided to test the proposed methods.

Thanks!

REFERENCES I

[AHM+20] Salman Ahmadi-Asl et al. “Randomized Algorithms for Fast Computation of Low Rank Tensor Ring Model”. In: *Mach. Learn.: Sci. Technol.* 2.1 (2020), p. 011001. DOI: `10.1088/2632-2153/abad87`.

[BBK18] Casey Battaglino, Grey Ballard, and Tamara G. Kolda. “A Practical Randomized CP Tensor Decomposition”. In: *SIAM J. Matrix Anal. Appl.* 39.2 (2018), pp. 876–901. DOI: `10.1137/17M1112303`.

[CW17] Kenneth L. Clarkson and David P. Woodruff. “Low-Rank Approximation and Regression in Input Sparsity Time”. In: *J. ACM* 63.6 (2017), 54:1–45. DOI: `10.1145/3019134`.

[DIA+18] Huaian Diao et al. “Sketching for Kronecker Product Regression and P-splines”. In: *International Conference on Artificial Intelligence and Statistics.* Vol. 84. Playa Blanca, Lanzarote, Canary Islands: PMLR, 2018, pp. 1299–1308.

REFERENCES II

[JKW21] Ruhui Jin, Tamara G. Kolda, and Rachel Ward. "Faster Johnson-Lindenstrauss Transforms via Kronecker Products". In: *Inf. Inference* 10.4 (2021), pp. 1533–1562. DOI: `10.1093/imaiai/iaaa028`.

[LK20] Brett W. Larsen and Tamara G. Kolda. "Practical Leverage-Based Sampling for Low-Rank Tensor Decomposition". In: *arXiv preprint arXiv:2006.16438* (2020).

[MB18] Osman Asif Malik and Stephen Becker. "Low-Rank Tucker Decomposition of Large Tensors Using TensorSketch". In: *Advances in Neural Information Processing Systems*. Vol. 31. Montréal, Canada: Curran Associates, Inc., 2018, pp. 10117–10127.

[MB20] Osman Asif Malik and Stephen Becker. "Fast Randomized Matrix and Tensor Interpolative Decomposition Using CountSketch". In: *Adv. Comput. Math.* 46 (2020), p. 76. DOI: `10.1007/s10444-020-09816-9`.

REFERENCES III

[MB21a] Osman Asif Malik and Stephen Becker. “A Sampling-Based Method for Tensor Ring Decomposition”. In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Virtual Event: PMLR, 2021, pp. 7400–7411.

[MB21b] Osman Asif Malik and Stephen Becker. “A Sampling-Based Method for Tensor Ring Decomposition”. In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Virtual Event: PMLR, 2021, pp. 7400–7411.

[Pag13] Rasmus Pagh. “Compressed Matrix Multiplication”. In: *ACM Trans. Comput. Theory* 5.3 (2013), pp. 1–17. doi: 10.1145/2493252.2493254.

[Yua+19] Longhao Yuan et al. “Randomized Tensor Ring Decomposition and Its Application to Large-Scale Data Reconstruction”. In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Brighton Conference Centre Brighton, U.K.: IEEE, 2019, pp. 2127–2131.

REFERENCES IV

[ZHA+16] Qibin Zhao et al. “Tensor Ring Decomposition”. In: *arXiv preprint arXiv:1606.05535* (2016).