

CSE 392: Matrix and Tensor Algorithms for Data

Instructor: Shashanka Ubaru

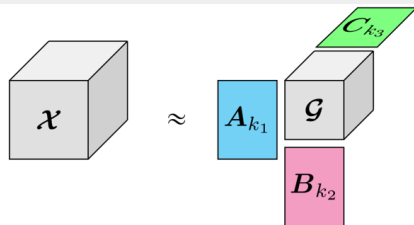
University of Texas, Austin
Spring 2024

Lecture 20: Randomized Tucker, TensorSketch

Outline

- 1 Randomized Tucker
- 2 TensorSketch
- 3 Tensor Train Decomposition

Recall: Tucker Decomposition



- The *Tucker decomposition* of a three-mode tensor $\mathcal{X} \in \mathbb{R}^{m \times n \times p}$ is given by:

$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} =: \llbracket \mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C} \rrbracket,$$

where $\mathcal{G} \in \mathbb{R}^{k_1 \times k_2 \times k_3}$ is called the core tensor and $\mathbf{A} \in \mathbb{R}^{m \times k_1}$, $\mathbf{B} \in \mathbb{R}^{n \times k_2}$ and $\mathbf{C} \in \mathbb{R}^{p \times k_3}$ are factor matrices.

- Elementwise:

$$x_{ij\ell} \approx \sum_{q=1}^{k_1} \sum_{r=1}^{k_2} \sum_{s=1}^{k_3} g_{qrs} a_{iq} b_{jr} c_{\ell s} \text{ for } i \in [m], j \in [n], \ell \in [p]$$

HOSVD Algorithm

Inputs: Tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_d}$, ranks $\{r_1, \dots, r_d\} \in \mathbb{N}$.

- ❶ **for** $\ell = 1, \dots, d$ **do**
- ❷ $\mathbf{U}^{(\ell)} \leftarrow r_\ell$ leading left singular vectors of $\mathbf{A}_{(\ell)}$
- ❸ **end for**
- ❹ $\mathcal{G} = \mathcal{A} \times_1 \mathbf{U}^{(1)\top} \times_2 \mathbf{U}^{(2)\top} \dots \times_d \mathbf{U}^{(d)\top}$
- ❺ **return** $\mathcal{G}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \dots, \mathbf{U}^{(d)}$

HOOI Algorithm

Inputs: Tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_d}$, ranks $\{r_1, \dots, r_d\} \in \mathbb{N}$.

- ➊ Initialize $\mathbf{U}^{(\ell)} \in \mathbb{R}^{n_\ell \times r_\ell}$ for all $\ell \in [d]$
- ➋ repeat
- ➌ for $\ell = 1, \dots, d$ do
- ➍ $\mathcal{Y} = \mathcal{A} \times_1 \mathbf{U}^{(1)\top} \dots \times_{\ell-1} \mathbf{U}^{(\ell-1)\top} \times_{\ell+1} \mathbf{U}^{(\ell+1)\top} \dots \times_d \mathbf{U}^{(d)\top}$
- ➎ $\mathbf{U}^{(\ell)} \leftarrow r_\ell$ leading left singular vectors of $\mathbf{Y}_{(\ell)}$
- ➏ end for
- ➐ until fit ceases to improve or maximum iterations exhausted
- ➑ $\mathcal{G} = \mathcal{A} \times_1 \mathbf{U}^{(1)\top} \times_2 \mathbf{U}^{(2)\top} \dots \times_d \mathbf{U}^{(d)\top}$
- ➒ return $\mathcal{G}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \dots, \mathbf{U}^{(d)}$

Sequential Truncated HOSVD (ST-HOSVD)

- Choose an ordering in which to visit the modes
- Once left singular vectors for a mode are computed, immediately project. Then only operate on the projected core result
- Example (ordering 1,2,3 and truncation (k_1, k_2, k_3)):
 - ▶ Compute $\mathbf{U}^{(1)}$ from SVD of $\mathcal{A}_{(1)}$
 - ▶ Compute $\mathbf{U}^{(2)}$ from SVD of $\hat{\mathcal{C}} := \mathcal{A} \times_1 (\mathbf{U}^{(1)})^\top$
 - ▶ Compute $\mathbf{U}^{(3)}$ from SVD of $\tilde{\mathcal{C}} := \hat{\mathcal{C}} \times_2 (\mathbf{U}^{(2)})^\top$
 - ▶ $\mathcal{C} = \tilde{\mathcal{C}} \times_3 (\mathbf{U}^{(3)})^\top$
- Now let $\mathcal{A} \approx [\mathcal{C}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}]$. Worst case error bound is **same** as for tr-HOSVD.
- Computing on successively smaller objects, more efficient; often near-comparable, or better, behavior than tr-HOSVD!

Towards Randomized Tucker

Both HOSVD and STHOSVD rely on matrix SVDs. The obvious first choice for randomization - use calls to matrix routine randSVD.

Matrix RandSVD: Given $\mathbf{X} \in \mathbb{R}^{m \times n}$, target rank r , oversampling parameter $p \geq 0$ such that $r + p \leq \min\{m, n\}$,

- Draw Gaussian random matrix $\Omega \in \mathbb{R}^{n \times (r+p)}$
- Multiply $\mathbf{Y} \leftarrow \mathbf{X}\Omega$
- Thin QR factorization $\mathbf{Y} = \mathbf{Q}\mathbf{R}$
- Form $\mathbf{B} \leftarrow \mathbf{Q}^\top \mathbf{X}$
- Calculate thin SVD $\mathbf{B} = \hat{\mathbf{U}}_{\mathbf{B}} \hat{\mathbf{S}} \hat{\mathbf{V}}^\top$
- Form $\hat{\mathbf{U}} \leftarrow \mathbf{Q}(\hat{\mathbf{U}}_{\mathbf{B}})_{:,1:r}$
- Compress $\hat{\mathbf{S}} \leftarrow \hat{\mathbf{S}}_{1:r,1:r}$, and $\hat{\mathbf{V}} \leftarrow \hat{\mathbf{V}}_{:,1:r}$, so $\mathbf{X} = \hat{\mathbf{U}}\hat{\mathbf{S}}\hat{\mathbf{V}}$

Halko, Martinsson, Tropp, SIREV, 2011

Towards Randomized Tucker

Both HOSVD and STHOSVD rely on matrix SVDs. The obvious first choice for randomization - use calls to matrix routine randSVD.

Given: $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_d}$; target rank vector $\mathbf{r} \in \mathbb{N}^d$; oversampling parameter $p \geq 0$

$j = 1 : d$

- Draw random Gaussian matrix $\mathbf{\Omega}_j \in \mathbb{R}^{\prod_{i \neq j} n_i \times (r_j + p)}$
- $[\hat{\mathbf{U}}, \hat{\mathbf{\Sigma}}, \hat{\mathbf{V}}] = \text{RandSVD}(\mathcal{A}_{(j)}, r_j, p, \mathbf{\Omega}_j)$
- Set $\mathbf{U}^{(j)} \leftarrow \hat{\mathbf{U}}$

Form $\mathcal{C} = \mathcal{A} \times_{j=1}^d (\mathbf{U}^{(j)})^\top$

G. Zhou, A. Cichocki, and S. Xie, Decomposition of Big Tensors with Low Multilinear Rank, arXiv 1412.1885, 2014

Result¹

The output (with minor assumptions) satisfies

Theorem (Randomized HOSVD)

$$\mathbb{E}_{\{\mathbf{\Omega}_k\}_{k=1}^d} \|\mathcal{A} - \hat{\mathcal{A}}\|_F \leq \left(d + \frac{\sum_{j=1}^d r_j}{p-1} \right)^{1/2} \|\mathcal{A} - \hat{\mathcal{A}}_{opt}\|_F.$$

special cases: Let $r = \max_{1 \leq j \leq d} r_j$. Then, if $p = r + 1$,

$$\mathbb{E}_{\{\mathbf{\Omega}_k\}_{k=1}^d} \|\mathcal{A} - \hat{\mathcal{A}}\|_F \leq \sqrt{2} \|\mathcal{A} - \hat{\mathcal{A}}_{\text{HOSVD}}\|_F \leq \sqrt{2d} \|\mathcal{A} - \hat{\mathcal{A}}_{opt}\|_F.$$

If $p = \lceil \frac{r}{\epsilon} \rceil + 1$ for some $\epsilon > 0$,

$$\mathbb{E}_{\{\mathbf{\Omega}_k\}_{k=1}^d} \|\mathcal{A} - \hat{\mathcal{A}}\|_F \leq \sqrt{1 + \epsilon} \|\mathcal{A} - \hat{\mathcal{A}}_{\text{HOSVD}}\|_F \leq \sqrt{d(1 + \epsilon)} \|\mathcal{A} - \hat{\mathcal{A}}_{opt}\|_F.$$

¹Minster, Saibaba, Kilmer, SIMODS, 2020

Randomized Sequentially Truncated-HOSVD (r-STHOSVD)

- Similar structure, except feed current intermediate (mode-wise unfolded) core tensor.
- Complication for the theoretical result: at each intermediate step, the partially truncated core tensor is a random tensor.
- The theoretical result gives the same upper bound (fix the processing order; but ultimately independent of this)!
- Again, we see that in the worst case, the randomized versions of either algorithm can have the same performance.
- The latter is cheaper, and in practice can perform better. Use processing order that makes it cheapest.

Dynamic Randomized HOSVD and STHOSVD ³

If target rank is unknown, how to find $\hat{\mathcal{A}}$ such that

$$\|\mathcal{A} - \hat{\mathcal{A}}\|_F \leq \epsilon \|\mathcal{A}\|_F?$$

Utilize matrix adaptive randomized range finders² for interim calculations. Given a matrix \mathbf{A} and a tolerance $\epsilon > 0$, the goal is to find a matrix \mathbf{Q} with orthonormal columns that satisfies

$$\|\mathbf{A} - \mathbf{Q}\mathbf{Q}^\top \mathbf{A}\| \leq \epsilon \|\mathbf{A}\|.$$

No. columns in \mathbf{Q} is assumed rank of the low rank approximation.

Example: equally apportioned per mode: choose the factor matrices \mathbf{U}_j to satisfy

$$\|\mathcal{A}_{(j)} - \mathbf{U}^{(j)}(\mathbf{U}^{(j)})^\top \mathcal{A}_{(j)}\|_F = \|\mathcal{A} \times_j (\mathbf{I} - \mathbf{U}^{(j)}(\mathbf{U}^{(j)})^\top)\|_F \leq \frac{\epsilon}{\sqrt{d}} \|\mathcal{A}\|_F.$$

²See W. Yu, Y. Gu, and Y. Li, SIMAX, 2018 and references therein.

³Minster, Saibaba, Kilmer, SIMODS, 2020

Preserving Structure?

We know that the (intermediate) core can be dense in any of the above mentioned processes.

What if \mathcal{A} is sparse, or has other structure (e.g. non-negativity)?

Can we compute a factorization such that the core tensor inherits properties of \mathcal{A} ?

Yes!

Process for a matrix \mathbf{X}

Instead of using $\mathbf{X} \approx \mathbf{Q}\mathbf{Q}^\top \mathbf{X}$ where $\mathbf{Q}\mathbf{R} = (\mathbf{X}\Omega_{r+p})$ do:

Compute strong RRQR of \mathbf{Q}^\top

$$\mathbf{Q}^\top \mathbf{S} = \mathbf{Z}\mathbf{N},$$

then $\mathbf{P} = \mathbf{S}_{:,1:s}$, so $\mathbf{P}^\top \mathbf{Q}$ well-conditioned rows of \mathbf{Q} .

Define the oblique projector:

$$\mathbf{Q}(\mathbf{P}^\top \mathbf{Q})^{-1} \mathbf{P}^\top$$

and apply this to \mathbf{X} .

$$\mathbf{X} \approx (\mathbf{Q}(\mathbf{P}^\top \mathbf{Q})^{-1}) \underbrace{\mathbf{P}^\top \mathbf{X}}_{\widehat{\mathbf{X}}}.$$

Note that $\widehat{\mathbf{X}}$ is **subselected rows** of \mathbf{X} .

Structure Preserving STHOSVD

Idea:

- pick an order to visit
- apply the previous idea to the current unfolded core
- update the current core (it will have subselected rows), and the resulting factor matrix is the left matrix product.

The final core will have multirank $(r_1 + p, r_2 + p, \dots, r_d + p)$ and contain portions of the original tensor. The factor matrices are not orthogonal.

Randomized Variants that Handle Sparsity

Formidable Repository of Sparse Tensors and Tools database.

| Tensor | Dimensions | Nonzeros |
|--------|--|------------|
| NELL-2 | $12092 \times 9184 \times 28818$ | 76,879,419 |
| Enron | $6066 \times 5699 \times 244268 \times 1176$ | 54,202,099 |

NELL-2: entity \times relation \times entity (NELL is a machine learning system that relates different categories)

Enron: sender \times receiver \times word \times date (word counts in emails released during an investigation by the FERC)

Approximate truncated (r, r, r) HOSVD and ST-HOSVD

Results

| r | Relative Error | | Runtime in seconds | |
|-----|----------------|-----------|--------------------|-----------|
| | SP-STHOSVD | R-STHOSVD | SP-STHOSVD | R-STHOSVD |
| 20 | 0.6015 | 0.2081 | 0.4086 | 31.5615 |
| 45 | 0.3854 | 0.1259 | 0.7965 | 34.5802 |
| 145 | 0.0976 | 0.0332 | 3.5659 | 42.0969 |
| 195 | 0.0578 | 0.0180 | 6.8285 | 50.2907 |

Table: Results, Subsampled Enron dataset.

Taking advantage of the sparsity structure allows for faster compression⁴.

⁴R. Minster, A.K. Saibaba, and M. E. Kilmer, “Randomized Algorithms for low-rank Decompositions in the Tucker Format,” SIMODS, 2020.

Recall: TUCKER-ALS

- Minimize the objective function:

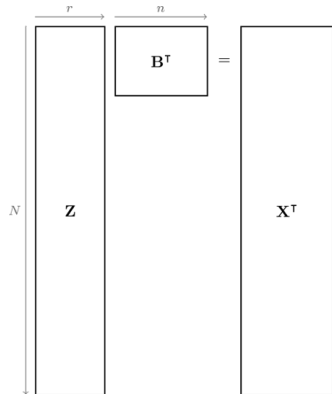
$$F(\mathcal{G}, \mathbf{A}, \mathbf{B}, \mathbf{C}) = \|\mathcal{X} - \llbracket \mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C} \rrbracket\|_F^2$$

- The canonical TUCKER-ALS - repeatedly solve until convergence:

- ▶ $\mathbf{A}_{t+1} = \arg \min_{\mathbf{A}} F(\mathcal{G}_t, \mathbf{A}, \mathbf{B}_t, \mathbf{C}_t) = \arg \min_{\mathbf{A}} \left\| (\mathbf{C}_t \otimes \mathbf{B}_t) \mathbf{G}_{(1),t}^\top \mathbf{A}^\top - \mathbf{X}_{(1)}^\top \right\|_F^2$
- ▶ $\mathbf{B}_{t+1} = \arg \min_{\mathbf{B}} F(\mathcal{G}_t, \mathbf{A}_{t+1}, \mathbf{B}, \mathbf{C}_t) = \arg \min_{\mathbf{B}} \left\| (\mathbf{C}_t \otimes \mathbf{A}_{t+1}) \mathbf{G}_{(2),t}^\top \mathbf{B}^\top - \mathbf{X}_{(2)}^\top \right\|_F^2$
- ▶ $\mathbf{C}_{t+1} = \arg \min_{\mathbf{C}} F(\mathcal{G}_t, \mathbf{A}_{t+1}, \mathbf{B}_{t+1}, \mathbf{C}) = \arg \min_{\mathbf{C}} \left\| (\mathbf{B}_{t+1} \otimes \mathbf{A}_{t+1}) \mathbf{G}_{(3),t}^\top \mathbf{C}^\top - \mathbf{X}_{(3)}^\top \right\|_F^2$
- ▶ $\mathcal{G}_{t+1} = \arg \min_{\mathcal{G}} \left\| (\mathbf{C}_{t+1} \otimes \mathbf{B}_{t+1} \otimes \mathbf{A}_{t+1}) \mathbf{g}_{(\cdot)} - \mathbf{x}_{(\cdot)} \right\|_2^2$

Recall: Kronecker FJLTs

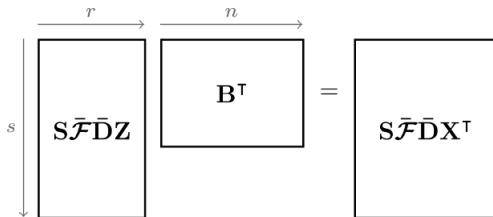
$$\min_B \|ZB^\top - X^\top\|_F^2$$



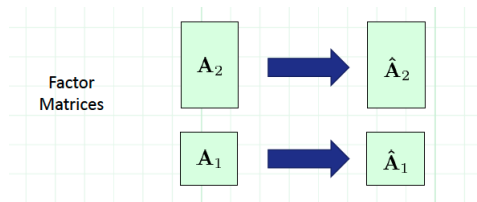
$$Z = A_d \odot \cdots \odot A_{k+1} \odot A_{k-1} \odot \cdots \odot A_1$$

$$\min_B \|S\bar{\mathcal{F}}\bar{D}ZB^\top - S\bar{\mathcal{F}}\bar{D}X^\top\|_F^2$$

- S is $s \times N$ sampling matrix
- $\bar{\mathcal{F}} = \mathcal{F}_d \otimes \cdots \otimes \mathcal{F}_{k+1} \otimes \mathcal{F}_{k-1} \otimes \cdots \otimes \mathcal{F}_1$.
- $\bar{D} = D_d \otimes \cdots \otimes D_{k+1} \otimes D_{k-1} \otimes \cdots \otimes D_1$.



Mixing Kronecker product Efficiently Using KFJLT



$$\begin{aligned} S\bar{F}\bar{D}Z &= S(\mathcal{F}_2 \otimes \mathcal{F}_1)(D_2 \otimes D_1)(A_2 \odot A_1) \\ &= S((\mathcal{F}_2 D_2) \otimes (\mathcal{F}_1 D_1))(A_2 \odot A_1) \\ &= S((\mathcal{F}_2 D_2 A_2) \odot (\mathcal{F}_1 D_1 A_1)) \\ &= S(\hat{A}_2 \odot \hat{A}_1) \end{aligned}$$

Same approach holds for Kronecker products too:

$$\begin{aligned} S\bar{F}\bar{D}Z &= S(\mathcal{F}_2 \otimes \mathcal{F}_1)(D_2 \otimes D_1)(A_2 \otimes A_1) \\ &= S((\mathcal{F}_2 D_2) \otimes (\mathcal{F}_1 D_1))(A_2 \otimes A_1) \\ &= S((\mathcal{F}_2 D_2 A_2) \otimes (\mathcal{F}_1 D_1 A_1)) \\ &= S(\hat{A}_2 \otimes \hat{A}_1) \end{aligned}$$

Structured Gaussian sketch

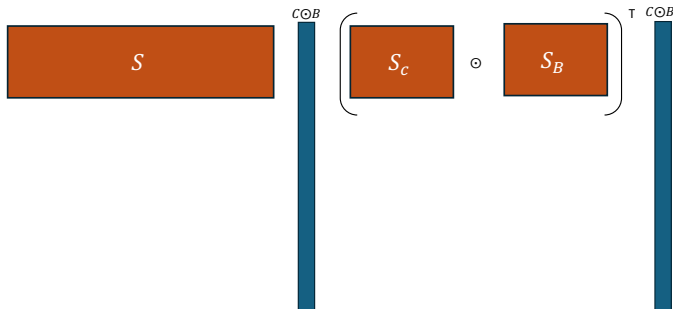
Standard Gaussian sketch: $\mathbf{S} \in \mathbb{R}^{m \times n_2 n_3}$
All entries are i.i.d Gaussian, $\mathbf{S}_{ij} \sim \mathcal{N}(0, 1)$.

Structured Gaussian sketch [BBB15]:

$\mathbf{S} = (\mathbf{S}_C \odot \mathbf{S}_B)^\top$, where $\mathbf{S}_C \in \mathbb{R}^{n_3 \times m}$ and $\mathbf{S}_B \in \mathbb{R}^{n_2 \times m}$. Then,

$$(\mathbf{S}_C \odot \mathbf{S}_B)^\top (\mathbf{C} \odot \mathbf{B}) = (\mathbf{S}_C^\top \mathbf{C}) * (\mathbf{S}_B^\top \mathbf{B})$$

Efficient! Can prove JL-type results.



- **TensorSketch:** Structured CountSketch.
- *CountSketch:* \mathbf{S} is of the form:

$$\begin{bmatrix} 0 & -1 & 0 & 0 & \cdots & 0 \\ +1 & 0 & 0 & +1 & \cdots & 0 \\ 0 & 0 & -1 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & \cdots & -1 \end{bmatrix}$$

One random ± 1 per column. Row \mathbf{A}_{i*} of \mathbf{A} contributes $\pm \mathbf{A}_{i*}$ to one of the rows of \mathbf{SA} .

- TensorSketch was first introduced in [Pham & Pagh, 2013] for compressed matrix multiplication and approximating SVM polynomial kernels efficiently.
- Avron et al. show that TensorSketch provides an oblivious subspace embedding.

TensorSketch: Structured CountSketch

Countsketch: $\mathbf{S} = \mathbf{P}\mathbf{D} \in \mathbb{R}^{m \times N}$

where \mathbf{P} has 1 nonzero per column $\mathbb{R}^{n_2 \times n_2}$. Then,

$\mathbf{P}_{:,j} \sim \text{Unif}(\mathbf{e}_1, \dots, \mathbf{e}_N)$.

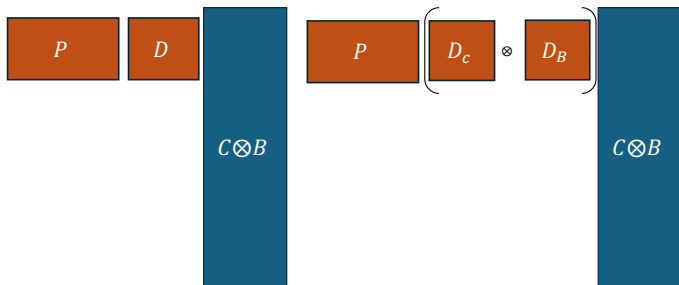
and \mathbf{D} is diagonal with i.i.d ± 1 entries.

TensorSketch:

$\mathbf{S} = \mathbf{P}(\mathbf{D}_C \otimes \mathbf{D}_B)$, where $\mathbf{D}_C \in \mathbb{R}^{n_3 \times n_3}$ and $\mathbf{D}_B \in \mathbb{R}^{n_2 \times n_2}$.

$$\mathbf{S}(\mathbf{C} \odot \mathbf{B}) = \text{FFT}^{-1} \left(\text{FFT}(\mathbf{S}_C \mathbf{C}) * \text{FFT}(\mathbf{S}_B^\top \mathbf{B}) \right)$$

$$\mathbf{S}(\mathbf{C} \otimes \mathbf{B}) = \text{FFT}^{-1} \left(\left(\text{FFT}(\mathbf{S}_C \mathbf{C})^\top \odot \text{FFT}(\mathbf{S}_B \mathbf{B})^\top \right)^\top \right)$$



Algorithm 2: TUCKER-TS (proposal)

input : \mathcal{Y} , target rank (R_1, R_2, \dots, R_N) , sketch dimensions (J_1, J_2)
output : Rank- (R_1, R_2, \dots, R_N) Tucker decomposition $\llbracket \mathcal{G}; \mathbf{A}^{(1)}, \dots, \mathbf{A}^{(N)} \rrbracket$ of \mathcal{Y}

- 1 Initialize $\mathcal{G}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}, \dots, \mathbf{A}^{(N)}$
- 2 Define TENSORSKETCH operators $\mathbf{T}^{(n)} \in \mathbb{R}^{J_1 \times \prod_{i \neq n} I_i}$, for $n \in [N]$, and $\mathbf{T}^{(N+1)} \in \mathbb{R}^{J_2 \times \prod_i I_i}$
- 3 **repeat**
- 4 **for** $n = 1, \dots, N$ **do**
- 5 $\mathbf{A}^{(n)} = \arg \min_{\mathbf{A}} \left\| \left(\mathbf{T}^{(n)} \otimes_{i=N, i \neq n}^1 \mathbf{A}^{(i)} \right) \mathbf{G}_{(n)}^\top \mathbf{A}^\top - \mathbf{T}^{(n)} \mathbf{Y}_{(n)}^\top \right\|_F^2$
- 6 **end**
- 7 $\mathcal{G} = \arg \min_{\mathcal{G}} \left\| \left(\mathbf{T}^{(N+1)} \otimes_{i=N}^1 \mathbf{A}^{(i)} \right) \mathbf{z}_{(:)} - \mathbf{T}^{(N+1)} \mathbf{y}_{(:)} \right\|_2^2$
- 8 Orthogonalize each $\mathbf{A}^{(i)}$ and update \mathcal{G}
- 9 **until** *termination criteria met*
- 10 **return** $\mathcal{G}, \mathbf{A}^{(1)}, \dots, \mathbf{A}^{(N)}$

$$\mathbf{T}\mathbf{A} = \mathbf{T} \bigotimes_{i=N}^1 \mathbf{A}^{(i)} = \text{FFT}^{-1} \left(\left(\bigodot_{i=N}^1 \left(\text{FFT} \left(\mathbf{S}^{(i)} \mathbf{A}^{(i)} \right) \right)^\top \right)^\top \right).$$

TensorSketch: Results

[Avron et al, 2014]

(AMM) Given $\mathbf{A} \in \mathbb{R}^{n^q \times d}$; $\mathbf{B} \in \mathbb{R}^{d' \times n^q}$; $\epsilon, \delta > 0$.

Let $\mathbf{S} \in \mathbb{R}^{m \times n^q}$ be a TensorSketch matrix, and if $m \geq \frac{(2+3^q)}{\epsilon^2 \delta}$, then with probability at least $1 - \delta$:

$$\|\mathbf{B}\mathbf{S}^\top \mathbf{S}\mathbf{A} - \mathbf{B}\mathbf{A}\|_F \leq \epsilon \|\mathbf{A}\|_F \|\mathbf{B}\|_F.$$

(Subspace embedding) For any fixed r -dimensional subspace \mathbf{U} , if $m \geq \frac{r^2(2+3^q)}{\epsilon^2 \delta}$, then with probability at least $1 - \delta$:

$$\|\mathbf{U}^\top \mathbf{S}^\top \mathbf{S} \mathbf{U} - \mathbf{I}\|_2 \leq \epsilon$$

TensorSketch: Results

[Malik & Becker, 2018]

Assume $\mathbf{T}^{(q+1)}$ be a TensorSketch matrix as in TuckerTS algorithm, and let $\mathbf{M} := \left(\mathbf{T}^{(q+1)} \otimes_{i=q}^1 \mathbf{A}^{(i)} \right)^\top \left(\mathbf{T}^{(q+1)} \otimes_{i=q}^1 \mathbf{A}^{(i)} \right)$, where all $\mathbf{A}^{(i)}$ have orthonormal columns. If $m \geq \frac{(\prod_i r_i)^2 (2+3^q)}{\epsilon^2 \delta}$, then with probability at least $1 - \delta$:

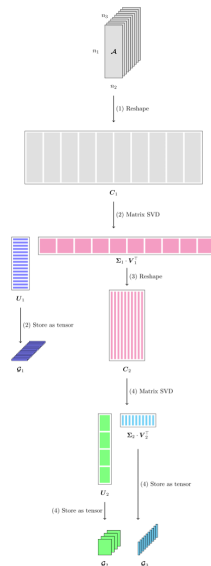
$$\|\mathbf{M} - \mathbf{I}\|_2 \leq \epsilon$$

Tensor Train Decomposition

- Based on the notion of variable splitting, consider an unfolding \mathbf{A} of a tensor \mathcal{A}

$$\mathbf{A}(i_1 i_2; i_3 i_4 i_5 i_6) = \sum_{\alpha_2} \mathbf{U}(i_1 i_2; \alpha_2) \mathbf{V}(i_3 i_4 i_5 i_6; \alpha_2)$$

- Provided a-priori knowledge as for separability of variables, the dimension has reduced (e.g. 6-dimensional tensor is decomposed into a product of 3- and 5-dimensional tensors)
- The process can obviously be repeated recursively leading to the Tensor Train decomposition



Tensor Train Decomposition

TT format of a tensor \mathcal{A}

$$\mathcal{A}(i_1, \dots, i_d) = \sum_{\alpha_0, \dots, \alpha_d} \mathcal{G}_1(\alpha_0, i_1, \alpha_1) \mathcal{G}_2(\alpha_1, i_2, \alpha_2), \dots, \mathcal{G}_d(\alpha_{d-1}, i_d, \alpha_d)$$

Can be represented compactly as a matrix product:

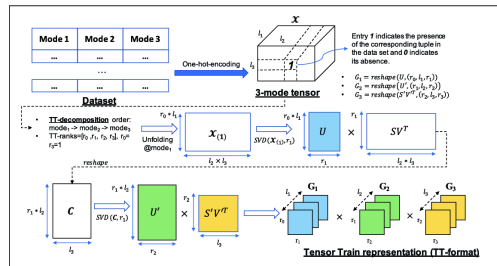
$$\mathcal{A}(i_1, \dots, i_d) = \underbrace{\mathcal{G}_1[i_1]}_{1 \times r_1} \underbrace{\mathcal{G}_2[i_2]}_{r_1 \times r_2} \dots \underbrace{\mathcal{G}_d[i_d]}_{r_{d-1} \times 1}$$

- \mathcal{G}_i : TT-cores (collections of matrices)
- r_i : TT-ranks
- $r = \max r_i$: the maximal TT-rank

TT uses $\mathcal{O}(dnr^2)$ memory to store $\mathcal{O}(nd)$ elements

Efficient only if the ranks are small

Oseledets, Tensor-train decomposition, 2011



References:

- Biagioni, David J., Daniel Beylkin, and Gregory Beylkin. “Randomized interpolative decomposition of separated representations.” *Journal of Computational Physics* 281 (2015): 116-134.
- Pham, Ninh, and Rasmus Pagh. “Fast and scalable polynomial kernels via explicit feature maps” *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2013.
- Avron, Haim, Huy Nguyen, and David Woodruff. “Subspace embeddings for the polynomial kernel.” *Advances in neural information processing systems* 27 (2014).
- Malik, Osman Asif, and Stephen Becker. “Low-rank tucker decomposition of large tensors using tensorsketch.” *Advances in neural information processing systems* 31 (2018).

Questions?