

Low Rank Tucker Approximation of a Tensor from Streaming Data

Madeleine Udell

Operations Research and Information Engineering
Cornell University

Based on joint work with
Yiming Sun (Cornell), Yang Guo (UW Madison),
Charlene Luo (Columbia), and Joel Tropp (Caltech)

April 1, 2019

Outline

Applications

Tucker factorization

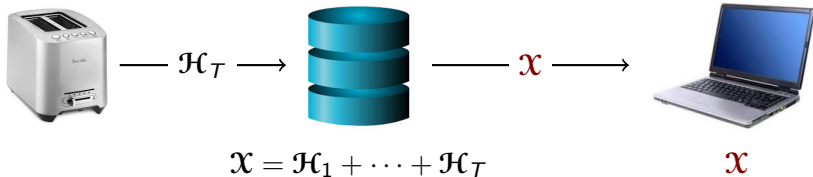
Sketching

Reconstruction

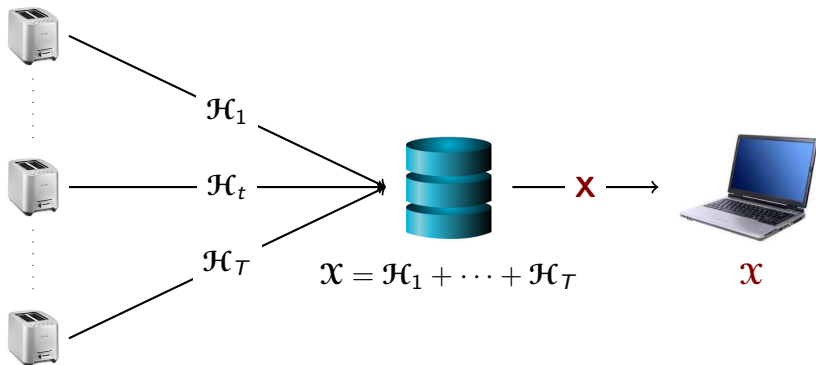
Numerics

*

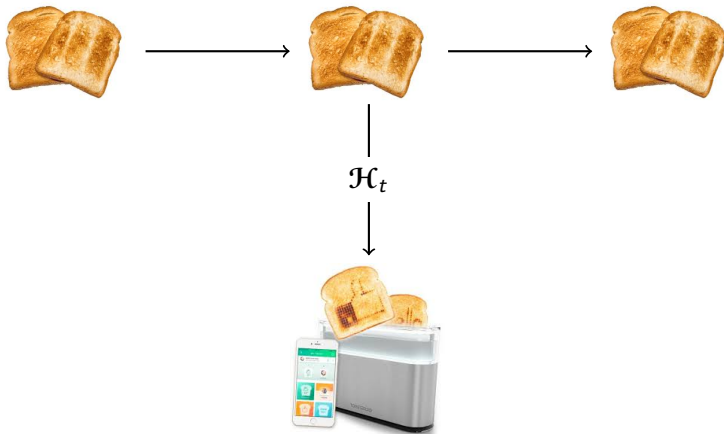
Big data, small laptop



Distributed data



Streaming data



$$\mathbf{x}^{(t)} = \mathcal{H}_1 + \dots + \mathcal{H}_t$$

Streaming multilinear algebra

turnstile model:

$$\mathcal{X} = \mathcal{H}_1 + \cdots + \mathcal{H}_T$$

- ▶ tensor \mathcal{X} presented as sum of smaller, simpler tensors \mathcal{H}_t
- ▶ must discard \mathcal{H}_t after it is processed
- ▶ **Goal:** without storing \mathcal{X} , approximate \mathcal{X} after seeing all updates (with guaranteed accuracy)

applications:

- ▶ scientific simulation
- ▶ sensor measurements
- ▶ memory- or communication-limited computing
- ▶ low memory optimization

Linear sketch

$$\begin{aligned}\mathcal{X} &= \mathcal{H}_1 + \cdots + \mathcal{H}_T \\ \mathcal{L}(\mathcal{X}) &= \mathcal{L}(\mathcal{H}_1) + \cdots + \mathcal{L}(\mathcal{H}_T)\end{aligned}$$

- ▶ select a linear map \mathcal{L} independent of \mathcal{X}
- ▶ sketch $\mathcal{L}(\mathcal{X})$ is much smaller than input tensor \mathcal{X}
- ▶ use randomness so sketch works for an arbitrary input
- ▶ essentially the only way to handle the turnstile model [Li, Nguyen & Woodruff 2014]

examples:

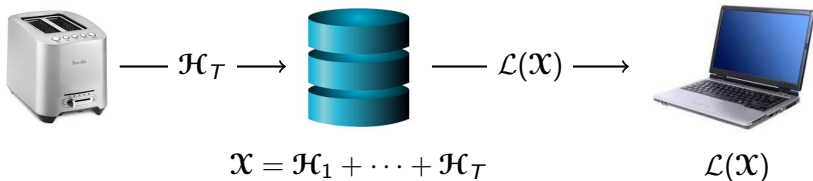
- ▶ $\mathcal{L}(\mathcal{X}) = \mathcal{X} \times_n \Omega$ for some matrix Ω
- ▶ $\mathcal{L}(\mathcal{X}) = \{\mathcal{X} \times_n \Omega_n\}_{n \in [M]}$ for some matrices $\{\Omega_n\}_{n \in [M]}$

Main idea

sketch suffices for (Tucker) approximation:

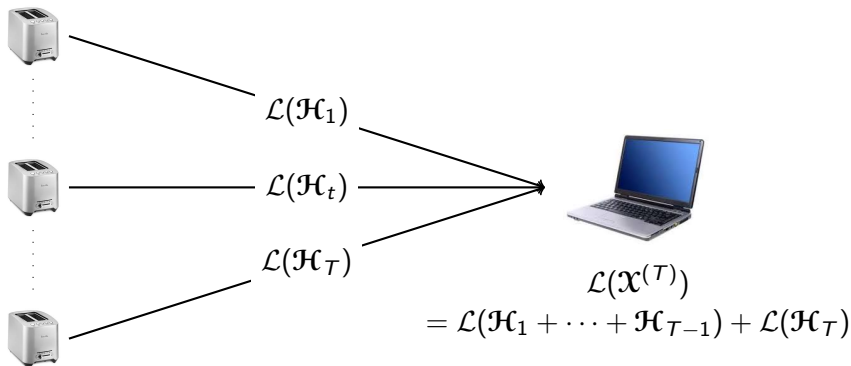
- ▶ compute (randomized) linear sketch of tensor
- ▶ recover low rank (Tucker) approximation from sketch
- ▶ (optional) improve approximation by revisiting data

Big data, small laptop: sketch



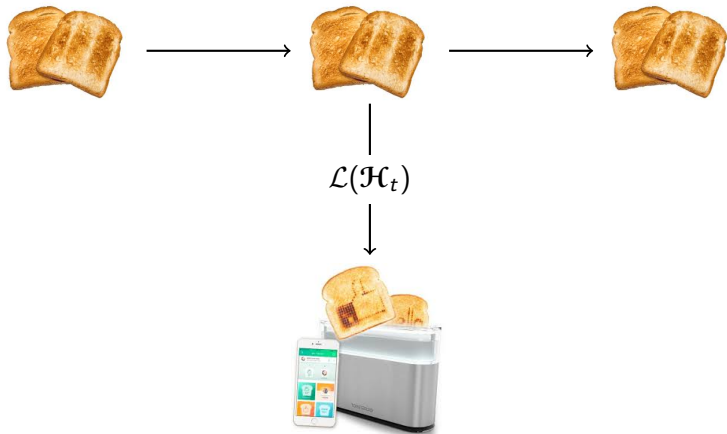
- ▶ (+) reduced communication
- ▶ (+) sketch of data fits on laptop

Distributed data: sketch



- ▶ (+) reduced communication
- ▶ (+) no PITI (personally identifiable toast information)
- ▶ (+) sketch of data fits on laptop

Streaming data: sketch



$$\mathcal{L}(\mathcal{X}^{(t)}) = \mathcal{L}(\mathcal{H}_1 + \cdots + \mathcal{H}_{t-1}) + \mathcal{L}(\mathcal{H}_t)$$

- ▶ (+) even a toaster can form sketch

Outline

Applications

Tucker factorization

Sketching

Reconstruction

Numerics

*

Notation

tensor to compress:

- ▶ tensor $\mathcal{X} \in \mathbf{R}^{I_1 \times \dots \times I_N}$ with N modes
- ▶ sometimes assume $I_1 = \dots = I_N = I$ for simplicity

indexing:

- ▶ $[N] = 1, \dots, N$
- ▶ $I_{(-n)} = I_1 \times \dots \times I_{n-1} \times I_{n+1} \times \dots \times I_N$

tensor operations:

- ▶ mode n product: for $\mathcal{A} \in \mathbf{R}^{k \times I_n}$,
 $\mathcal{X} \times_n \mathbf{A} \in \mathbf{R}^{I_1 \times \dots \times I_{n-1} \times k \times I_{n+1} \times \dots \times I_N}$
- ▶ unfolding $\mathbf{X}^{(n)} \in \mathbf{R}^{I_n \times I_{(-n)}}$ stacks mode- n fibers of \mathcal{X} as columns of matrix

Tucker factorization

rank $\mathbf{r} = (r_1, \dots, r_N)$ **Tucker factorization** of $\mathcal{X} \in \mathbf{R}^{I_1 \times \dots \times I_N}$:

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{U}_1 \cdots \times_N \mathbf{U}_N =: \llbracket \mathcal{G}; \mathbf{U}_1, \dots, \mathbf{U}_N \rrbracket$$

where

- ▶ $\mathcal{G} \in \mathbf{R}^{r_1 \times \dots \times r_N}$ is the **core matrix**
- ▶ $\mathbf{U}_n \in \mathbf{R}^{I_n \times r_n}$ is the **factor matrix** for each mode $n \in [N]$

(sometimes assume $r_1 = \dots = r_N = r$ for simplicity)

Tucker is useful for compression: when N is small,

- ▶ Tucker stores $O(rNI)$ numbers for rank r^3 approximation
- ▶ CP stores $O(rNI)$ numbers for rank r approximation

future work: one pass ST-HOSVD / tensor train?

Computing Tucker: HOSVD

Algorithm Higher order singular value decomposition (HOSVD)
[De Lathauwer, De Moor & Vandewalle 2000, Tucker 1966]

Given: tensor \mathcal{X} , rank $\mathbf{r} = (r_1, \dots, r_N)$

1. *Factors.* Compute top r_n left singular vectors \mathbf{U}_n of the unfolding $\mathbf{X}^{(n)}$ for each $n \in [N]$.
2. *Core.* Contract these with \mathcal{X} to form the core

$$\mathcal{G} = \mathcal{X} \times_1 \mathbf{U}_1^T \cdots \times_N \mathbf{U}_N^T.$$

Return: Tucker approximation $\mathcal{X}_{\text{HOSVD}} = \llbracket \mathcal{G}; \mathbf{U}_1, \dots, \mathbf{U}_N \rrbracket$

Two pass HOSVD

HOSVD can be computed in two passes over the tensor:

- ▶ **Factors.** use randomized linear algebra
 - ▶ need to find span of fibers of \mathcal{X} along n th mode:

$$\text{range}(\mathbf{U}_n) \approx \text{range}(\mathbf{X}^{(n)})$$

- ▶ if $\text{rank}(\Omega) \geq \text{rank}(\mathbf{X}^{(n)})$, then whp for random Ω ,

$$\text{range}(\mathbf{X}^{(n)}) = \text{range}(\mathbf{X}^{(n)}\Omega)$$

algorithm:

1. compute sketch $\mathcal{L}(\mathcal{X}) = \{\mathbf{X}^{(n)}\Omega_n\}_{n \in [N]}$
 2. use QR on sketch to approximate $\text{range}(\mathbf{X}^{(n)})$
- ▶ **Core.** Computation is linear in \mathcal{X} :

$$\mathcal{G} = \mathcal{X} \times_1 \mathbf{U}_1^T \cdots \times_N \mathbf{U}_N^T.$$

Source: [Halko, Martinsson & Tropp 2011, Zhou, Cichocki & Xie 2014, Battaglino, Ballard & Kolda 2019]

Computing Tucker: HOOI

Algorithm Higher order orthogonal iteration (HOOI)

[De Lathauwer et al. 2000]

Given: tensor \mathcal{X} , rank $\mathbf{r} = (r_1, \dots, r_N)$

Initialize: compute $\mathcal{X} \approx \llbracket \mathcal{G}; \mathbf{U}_1, \dots, \mathbf{U}_N \rrbracket$ using HOSVD

Repeat:

1. *Factors.* For $n \in [N]$,

$$\mathbf{U}_n \leftarrow \underset{\mathbf{U}_n}{\operatorname{argmin}} \|\llbracket \mathcal{G}; \mathbf{U}_1, \dots, \mathbf{U}_N \rrbracket - \mathcal{X}\|_F^2,$$

2. *Core.*

$$\mathcal{G} \leftarrow \underset{\mathcal{G}}{\operatorname{argmin}} \|\llbracket \mathcal{G}; \mathbf{U}_1, \dots, \mathbf{U}_N \rrbracket - \mathcal{X}\|_F^2.$$

Return: Tucker approximation $\mathcal{X}_{\text{HOOI}} = \llbracket \mathcal{G}; \mathbf{U}_1, \dots, \mathbf{U}_N \rrbracket$

► core update has closed form $\mathcal{G} \leftarrow \mathcal{X} \times_1 \mathbf{U}_1^\top \cdots \times_N \mathbf{U}_N^\top$

Previous work: one pass algorithm via HOOI

[Malik & Becker 2018]:

- ▶ (+) sketch design matrix to reduce size of HOOI subproblems
- ▶ (+) exploit Tucker structure of design matrix
- ▶ (-) expensive slow reconstruction (via iterative optimization)
- ▶ (-) no error guarantees for one pass algorithm

Outline

Applications

Tucker factorization

Sketching

Reconstruction

Numerics

*

Background: randomized sketches

idea: random matrix Ω is not orthogonal to range of interest (whp)

$$\text{range}(\mathbf{X}^{(n)}) = \text{range}(\mathbf{X}^{(n)}\Omega)$$

a **dimension reduction map** (DRM) (approximately) preserves range of its argument

examples of DRMS: multiplication by random matrix Ω that is

- ▶ gaussian
- ▶ sparse [Achlioptas 2003, Li, Hastie & Church 2006]
- ▶ SSFRT [Woolfe, Liberty, Rokhlin & Tygert 2008]
- ▶ tensor random projection (TRP) [Sun, Guo, Tropp & Udell 2018]
- ▶ ...

The sketch

approximate factor matrices and core:

- ▶ **Factor sketch** (\mathbf{k}). For each $n \in [N]$, fix random DRM $\mathbf{\Omega}_n \in \mathbb{R}^{l_{(-n)} \times k_n}$ and compute the sketch

$$\mathbf{V}_n = \mathbf{X}^{(n)} \mathbf{\Omega}_n \in \mathbb{R}^{l_n \times k_n}.$$

- ▶ **Core sketch** (\mathbf{s}). For each $n \in [N]$, fix random DRM $\mathbf{\Phi}_n \in \mathbb{R}^{l_n \times s_n}$. Compute the sketch

$$\mathcal{H} = \mathcal{X} \times_1 \mathbf{\Phi}_1^\top \cdots \times_N \mathbf{\Phi}_N^\top \in \mathbb{R}^{s_1 \times \cdots \times s_N}.$$

- ▶ *Rule of thumb.* Pick \mathbf{k} as big as you can afford, pick $\mathbf{s} = 2\mathbf{k}$.
- ▶ define $(\mathcal{H}, \mathbf{V}_1, \dots, \mathbf{V}_N) = \text{SKETCH}(\mathcal{X}; \{\mathbf{\Phi}_n, \mathbf{\Omega}_n\}_{n \in [N]})$

Low memory DRMs

factor sketch DRMs are big!

- ▶ $I_{(-n)} \times k_n$ for each $n \in [N]$

how to store?

- ▶ don't store DRMs; instead, use pseudorandom number generator to generate (parts of) DRMs as needed.
- ▶ use structured DRM:
 - ▶ TRP generates DRM as Khatri-Rao product of simpler, smaller DRMs
 - ▶ behaves approximately like a Gaussian sketch

Source: [Sun et al. 2018, Rudelson 2012]

Outline

Applications

Tucker factorization

Sketching

Reconstruction

Numerics

*

Recovery: factor matrices

- ▶ compute QR factorization of each factor sketch \mathbf{V}_n :

$$\mathbf{V}_n = \mathbf{Q}_n \mathbf{R}_n$$

where \mathbf{Q}_n is orthonormal and \mathbf{R}_n is triangular

Two pass algorithm

Algorithm Two Pass Sketch and Low Rank Recovery

Given: tensor \mathcal{X} , rank $\mathbf{r} = (r_1, \dots, r_N)$, DRMs $\{\Phi_n, \Omega_n\}_{n \in [N]}$

- ▶ *Sketch.* $(\mathcal{H}, \mathbf{V}_1, \dots, \mathbf{V}_N) = \text{SKETCH}(\mathcal{X}; \{\Phi_n, \Omega_n\}_{n \in [N]})$
- ▶ *Recover factor matrices.* For $n \in [N]$,

$$(\mathbf{Q}_n, \sim) \leftarrow \text{QR}(\mathbf{V}_n)$$

- ▶ *Recover core.*

$$\mathcal{W} \leftarrow \mathcal{X} \times_1 \mathbf{Q}_1 \cdots \times_N \mathbf{Q}_N$$

Return: Tucker approximation $\tilde{\mathcal{X}} = \llbracket \mathcal{W}; \mathbf{Q}_1, \dots, \mathbf{Q}_N \rrbracket$

accesses \mathcal{X} twice: 1) to sketch 2) to recover core

Intuition: one pass core recovery

- ▶ we want to know \mathcal{W} :
compression of \mathcal{X} using factor range approximations \mathbf{Q}_n
- ▶ we observe \mathcal{H} :
compression of \mathcal{X} using random projections Φ_n

how to approximate \mathcal{W} ?

$$\begin{aligned}\mathcal{X} &\approx \mathcal{X} \times_1 \mathbf{Q}_1 \mathbf{Q}_1^\top \times \cdots \times_N \mathbf{Q}_N \mathbf{Q}_N^\top \\ &= \left(\mathcal{X} \times_1 \mathbf{Q}_1^\top \times_N \cdots \times_N \mathbf{Q}_N^\top \right) \times_1 \mathbf{Q}_1 \cdots \times_N \mathbf{Q}_N \\ &= \mathcal{W} \times_1 \mathbf{Q}_1 \cdots \times_N \mathbf{Q}_N \\ \underbrace{\mathcal{X} \times_1 \Phi_1^\top \cdots \times_N \Phi_N^\top}_{\mathcal{H}} &\approx \mathcal{W} \times_1 \Phi_1^\top \mathbf{Q}_1 \times \cdots \times_N \Phi_N^\top \mathbf{Q}_N\end{aligned}$$

we can solve for \mathcal{W} : $s > k$, so each $\Phi_n^\top \mathbf{Q}_n$ has a left inverse (whp):

$$\mathcal{W} \approx \mathcal{H} \times_1 (\Phi_1^\top \mathbf{Q}_1)^\dagger \times \cdots \times_N (\Phi_N^\top \mathbf{Q}_N)^\dagger$$

One pass algorithm

Algorithm One Pass Sketch and Low Rank Recovery

Given: tensor \mathcal{X} , rank $\mathbf{r} = (r_1, \dots, r_N)$, DRMs $\{\Phi_n, \Omega_n\}_{n \in [N]}$

- ▶ *Sketch.* $(\mathcal{H}, \mathbf{V}_1, \dots, \mathbf{V}_N) = \text{SKETCH}(\mathcal{X}; \{\Phi_n, \Omega_n\}_{n \in [N]})$
- ▶ *Recover factor matrices.* For $n \in [N]$,

$$(\mathbf{Q}_n, \sim) \leftarrow \text{QR}(\mathbf{V}_n)$$

- ▶ *Recover core.*

$$\mathcal{W} \leftarrow \mathcal{H} \times_1 (\Phi_1^\top \mathbf{Q}_1)^\dagger \times \dots \times_N (\Phi_N^\top \mathbf{Q}_N)^\dagger$$

Return: Tucker approximation $\hat{\mathcal{X}} = \llbracket \mathcal{W}; \mathbf{Q}_1, \dots, \mathbf{Q}_N \rrbracket$

accesses \mathcal{X} only once, to sketch

Source: [Sun, Guo, Luo, Tropp & Udell 2019]

Fixed rank approximation

to truncate reconstruction to rank \mathbf{r} , truncate core:

Lemma

For a tensor $\mathcal{W} \in \mathbb{R}^{k_1 \times \dots \times k_N}$, orthogonal matrices $\mathbf{Q}_n \in \mathbb{R}^{k_n \times r_n}$,

$$\llbracket \mathcal{W} \times_1 \mathbf{Q}_1 \cdots \times_N \mathbf{Q}_N \rrbracket_{\mathbf{r}} = \llbracket \mathcal{W} \rrbracket_{\mathbf{r}} \times_1 \mathbf{Q}_1 \cdots \times_N \mathbf{Q}_N,$$

where $\llbracket \cdot \rrbracket$ denotes the best rank \mathbf{r} Tucker approximation.

\implies compute fixed rank approximation using, e.g., HOOI on (small) core approximation \mathcal{W}

Tail energy

For each unfolding $\mathbf{X}^{(n)}$, define its ρ th tail energy as

$$(\tau_{\rho}^{(n)})^2 := \sum_{k > \rho}^{\min(l_n, l_{(-n)})} \sigma_k^2(\mathbf{X}^{(n)}),$$

where $\sigma_k(\mathbf{X}^{(n)})$ is the k th largest singular value of $\mathbf{X}^{(n)}$.

Guarantees (I)

Theorem (Recommended parameters [Sun et al. 2019])

Sketch \mathcal{X} with Gaussian DRMs of parameters \mathbf{k} , $\mathbf{s} = 2\mathbf{k} + 1$.
Form a rank \mathbf{r} Tucker approximation $\hat{\mathcal{X}}$ using the one pass algorithm. Then

$$\mathbb{E}\|\mathcal{X} - \hat{\mathcal{X}}\|_F^2 \leq 4 \sum_{n=1}^N (\tau_{r_n}^{(n)})^2.$$

If \mathcal{X} is truly rank \mathbf{r} , we obtain the true Tucker factorization!

Guarantees (II)

Theorem (Detailed guarantee [Sun et al. 2019])

Sketch \mathcal{X} with Gaussian DRMs of parameters \mathbf{k} , \mathbf{s} . Form a rank \mathbf{r} Tucker approximation $\hat{\mathcal{X}}$ using the one pass algorithm. Then

$$\mathbb{E}\|\mathcal{X} - \hat{\mathcal{X}}\|_F^2 \leq (1 + \Delta) \min_{1 \leq \rho_n < k_n - 1} \sum_{n=1}^N \left(1 + \frac{\rho_n}{k_n - \rho_n - 1} \right) (\tau_{\rho_n}^{(n)})^2$$

where $\Delta = \max_{n=1}^N k_n / (s_n - k_n - 1)$

Outline

Applications

Tucker factorization

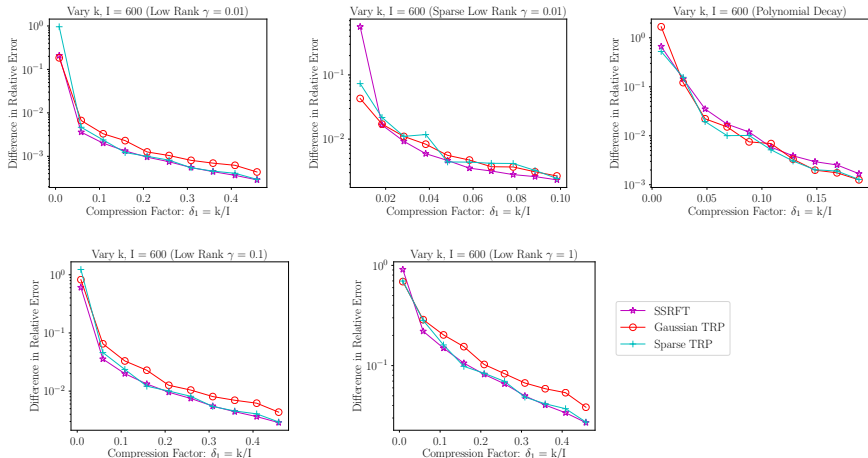
Sketching

Reconstruction

Numerics

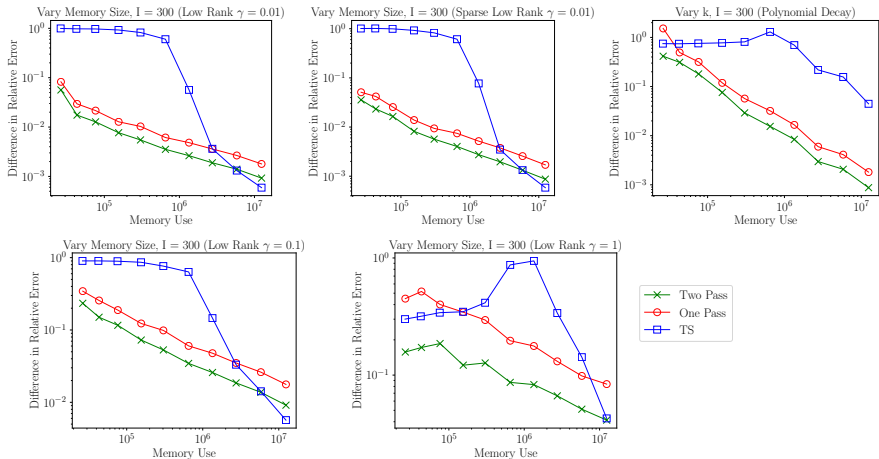
*

Different DRMs perform similarly



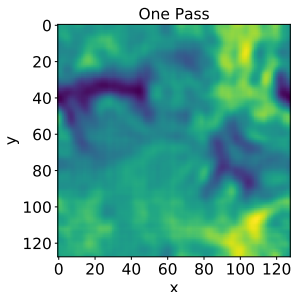
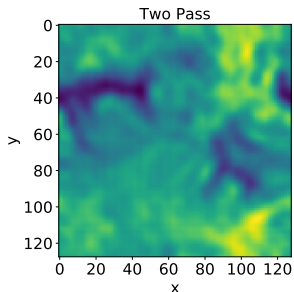
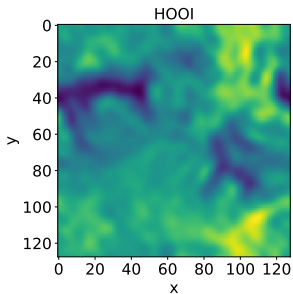
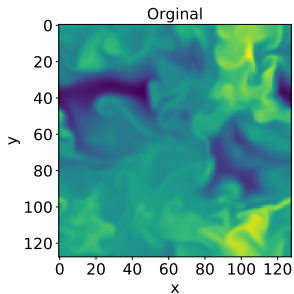
Comments: Synthetic data, $I = 600$ and $\mathbf{r} = (5, 5, 5)$. $k/I = .4 \implies 20\times$ compression.

Sensible reconstruction at practical compression level



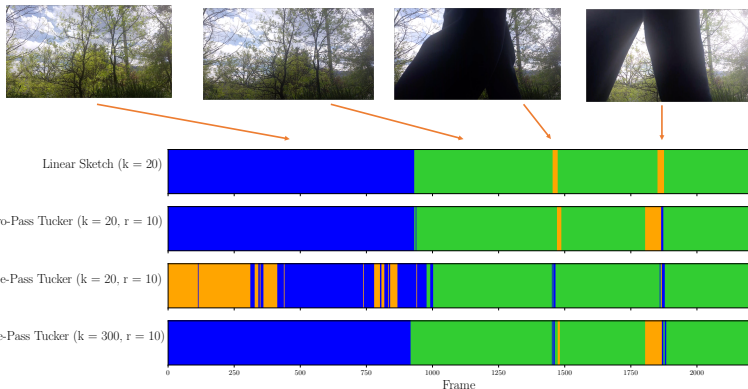
Comments: Error of fixed-rank approximation relative to HOOI for $r = 10$, $l = 300$ using TRP. Total memory use is $((2k + 1)^N + kIN)$ and $(Kr^{2N} + K * r^{2N-2})$. Low-rank data uses $\gamma = 0.01, 0.1, 1$.

Combustion simulation



Comments:
1408 × 128 × 128
simulated
combustion data
from [Lapointe,
Savard & Blanquart
2015].

Video scene classification



Comments: Video data $2200 \times 1080 \times 1980$. Classify scenes using k -means on: 1) linear sketch along the time dimension $k = 20$ (Row 1); 2) The Tucker factor along the time dimension, computed via our two pass (Row 2) and one pass (Row 3) sketching algorithm $(r, k, s) = (10, 20, 41)$. 3) The Tucker factor along the time dimension, computed via our one pass (Row 4) sketching algorithm $(r, k, s) = (10, 300, 601)$.

Summary

Streaming Tucker approximation **compresses tensor** without **storing it**.

useful for:

- ▶ streaming data
- ▶ distributed data
- ▶ low memory compute

key ideas:

- ▶ form linear sketch of tensor and recover from sketch
- ▶ random projection of tensor preserves dominant information

Future work + references

let's talk!

- ▶ bigger tensors to compress?
- ▶ streaming compression for ⟨your research⟩?

references:

- ▶ Sun, Y., Guo, Y., Tropp, J. A., and Udell, M. (2018). Tensor random projection for low memory dimension reduction. In NeurIPS Workshop on Relational Representation Learning.
- ▶ Sun, Y., Guo, Y., Luo, C., Tropp, J. A., and Udell, M. (2019). Low rank tucker approximation of a tensor from streaming data. In preparation.
- ▶ Tropp, J. A., Yurtsever, A., Udell, M., and Cevher, V. (2019). Streaming low-rank matrix approximation with an application to scientific simulation. Submitted to SISC.

Outline

Applications

Tucker factorization

Sketching

Reconstruction

Numerics

*

References

- Achlioptas, D. (2003). Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of computer and System Sciences*, 66(4), 671–687.
- Battaglino, C., Ballard, G., & Kolda, T. G. (2019). Faster parallel tucker tensor decomposition using randomization.
- De Lathauwer, L., De Moor, B., & Vandewalle, J. (2000). A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4), 1253–1278.
- Halko, N., Martinsson, P.-G., & Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2), 217–288.
- Lapointe, S., Savard, B., & Blanquart, G. (2015). Differential diffusion effects, distributed burning, and local extinctions in high karlovitz premixed flames. *Combustion and flame*, 162(9), 3341–3355.
- Li, P., Hastie, T. J., & Church, K. W. (2006). Very sparse random projections. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, (pp. 287–296). ACM.
- Li, Y., Nguyen, H. L., & Woodruff, D. P. (2014). Turnstile streaming algorithms might as well be linear sketches. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, (pp. 174–183). ACM.
- Malik, O. A. & Becker, S. (2018). Low-rank tucker decomposition of large tensors using tensorsketch. In *Advances in Neural Information Processing Systems*, (pp. 10116–10126).

- Rudelson, M. (2012). Row products of random matrices. *Advances in Mathematics*, 231(6), 3199–3231.
- Sun, Y., Guo, Y., Luo, C., Tropp, J. A., & Udell, M. (2019). Low rank tucker approximation of a tensor from streaming data. *In preparation*.
- Sun, Y., Guo, Y., Tropp, J. A., & Udell, M. (2018). Tensor random projection for low memory dimension reduction. In *NeurIPS Workshop on Relational Representation Learning*.
- Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3), 279–311.
- Woolfe, F., Liberty, E., Rokhlin, V., & Tygert, M. (2008). A fast randomized algorithm for the approximation of matrices. *Applied and Computational Harmonic Analysis*, 25(3), 335–366.
- Zhou, G., Cichocki, A., & Xie, S. (2014). Decomposition of big tensors with low multilinear rank. *arXiv preprint arXiv:1412.1885*.