Fast Exact Leverage Score Sampling from Khatri-Rao Products with Applications to Tensor Decomposition

Vivek Bharadwaj [1], Osman Asif Malik [2], Riley Murray [3,2,1], Laura Grigori [4], Aydın Buluç [2,1], James Demmel [1]

[1] Electrical Engineering and Computer Science Department, UC Berkeley

[2] Computational Research Division, Lawrence Berkeley National Lab

[3] International Computer Science Institute

[4] Institute of Mathematics, EPFL & Lab for Simulation and Modelling, Paul Scherrer Institute

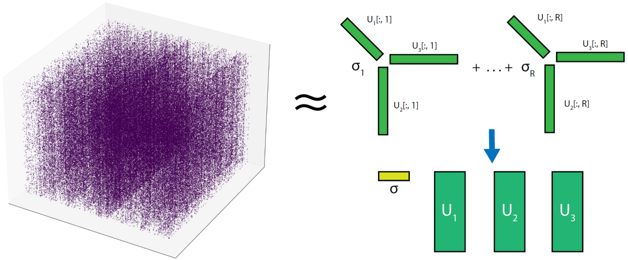
BERKELEY LAB

U.S. DEPARTMENT OF ENERGY
Office of Science

NEURAL INFORMATION PROCESSING SYSTEMS

Sparse Tensor Candecomp / PARAFAC Decomposition

Our Goal: Compute an approximate rank-R CP decomposition of an $(N+1)$-dimensional $I \times \ldots \times I$ sparse tensor $\mathcal{T}$:



Focus on large sparse tensors (mode sizes in the millions) and moderate decomposition rank $R \approx 10^2$. Assume $I \geq R$.

Alternating Least-Squares CP Decomposition

- ALS procedure: Randomly initialize factors $U_1, ..., U_{N+1}$, iteratively optimize one factor at a time while keeping others constant.

- Optimal value for U_{N+1}:

$$\mathrm{argmin}_X \|AX - B\|_F$$

where $B = \mathrm{mat}(\mathcal{T}, j)^\top$ and $A = U_N \odot ... \odot U_1$. Here, $\odot$ denotes a **Khatri-Rao product**, a column-wise Kronecker Product of two matrices:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \odot \begin{bmatrix} w & x \\ y & z \end{bmatrix} = \begin{bmatrix} aw & bx \\ cw & dx \\ ay & bz \\ cy & dz \end{bmatrix}$$

Randomized Linear Least-Squares

- Apply sketching operator S to both A and B, solve reduced problem

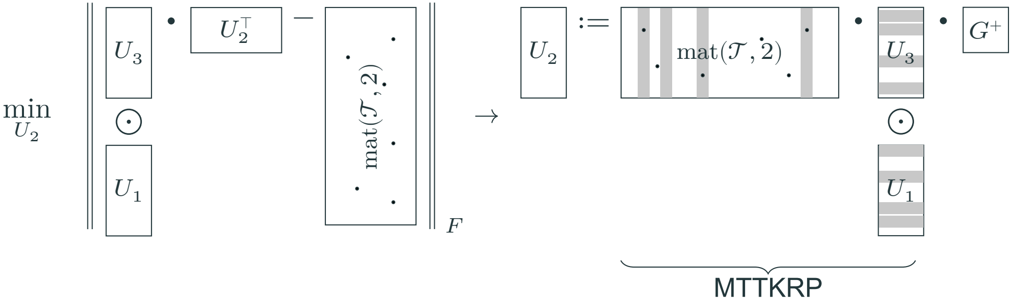
$$\min_{\tilde{X}} \left\| SA\tilde{X} - SB \right\|_F$$

- Want an (ε, δ) guarantee on solution quality: with high probability $(1 - \delta)$,

$$\left\| A\tilde{X} - B \right\|_F \leq (1 + \varepsilon) \min_X \|AX - B\|$$

Effect of Sketching Operator

$$\min_{U_j} \left\| \left[\bigodot_{k\neq j} U_k \right] \cdot U_j^\top - \mathrm{mat}(\mathcal{T}, j)^\top \right\|_F$$



Our Contributions

Method	**Round Complexity ($\tilde{O}$ notation)**
CP-ALS	$N(N+I)I^N R$
CP-ARLS-LEV (2022)	$N(R+I)R^{N+1}/(\varepsilon\delta)$
TNS-CP (2022)	$N^3 I R^3/(\varepsilon\delta)$
GTNE (2022)	$N^2(N^{1.5}R^{3.5}/\varepsilon^3 + IR^2)/\varepsilon^2$
STS-CP (ours, 2023)	$N(NR^3 \log I + IR^2)/(\varepsilon\delta)$

- We build a data structure with runtime **logarithmic** in the height of the KRP and quadratic in R to sample from leverage scores of A.

- Yields the **STS-CP** algorithm: lower asymptotic runtime for randomized CP decomposition than recent SOTA methods (practical too!)

Leverage Score Sampling

We will sample rows i.i.d. from A according to the *leverage score distribution* on its rows. Leverage score ℓ_i of row i is

$$\ell_i = A[i,:](A^\top A)^+ A[i,:]^\top$$

Theorem (Leverage Score Sampling Guarantees)

Suppose $S \in \mathbb{R}^{J \times I}$ is a leverage-score sampling matrix for $A \in \mathbb{R}^{I \times R}$, and define

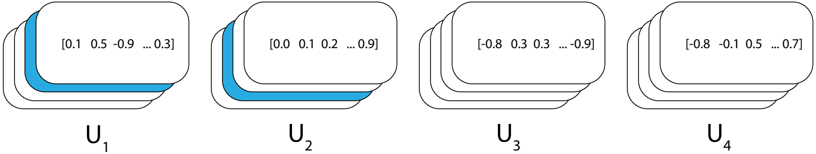
$$\tilde{X} := \arg\min_{\tilde{X}} \left\| SA\tilde{X} - SB \right\|_F$$

If $J \gtrsim R \max(\log(R/\delta), 1/(\varepsilon\delta))$, then with probability at least $1 - \delta$,

$$\left\| A\tilde{X} - B \right\|_F \leq (1+\varepsilon) \min_X \|AX - B\|_F$$

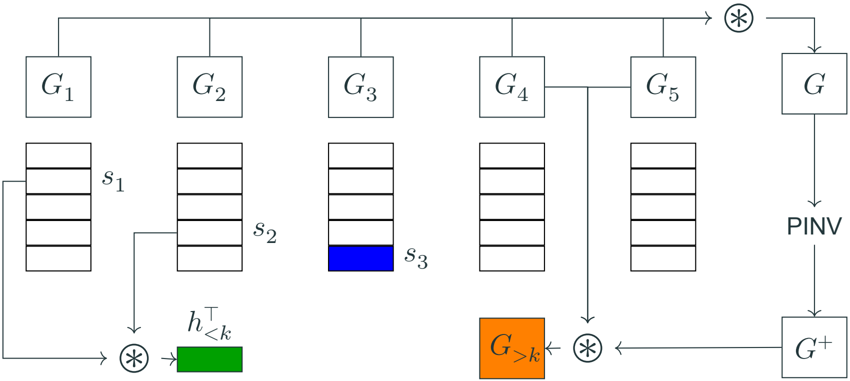
Leverage Score Sampling

- For $I = 10^7, N = 3$, matrix A has 10^{21} rows. Far too expensive to compute all leverage scores - can't even index rows with 64-bit integers.
- To sample from $\bigodot_{j=1}^{N} U_j$: draw a row from each of $U_1, ...U_N$ return their Hadamard product.



- Let $\hat{s}_j$ be a r.v. for the row index drawn from U_j. Assume $(\hat{s}_1, ..., \hat{s}_N)$ jointly follows the leverage score distribution on $U_N \odot ... \odot U_1$.

The Conditional Distribution of $\hat{s}_k$



Theorem

$$p(\hat{s}_k = s_k \mid \hat{s}_{<k} = s_{<k}) \propto \langle h_{<k} h_{<k}^\top, U_k[s_k, :]^\top U_k[s_k, :], G_{>k} \rangle$$

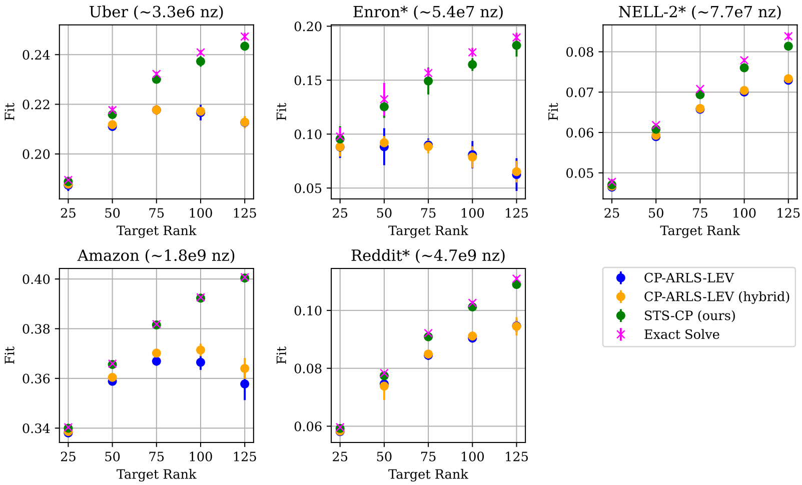
Key Sampling Primitive

- **Reduce to the Following Problem:** Given a matrix $U \in \mathbb{R}^{I \times R}$, design a data structure so that for any query vector $h \in \mathbb{R}^R$, you can efficiently draw a sample according proportional to the weight vector

$$q = (U \cdot h)^2$$

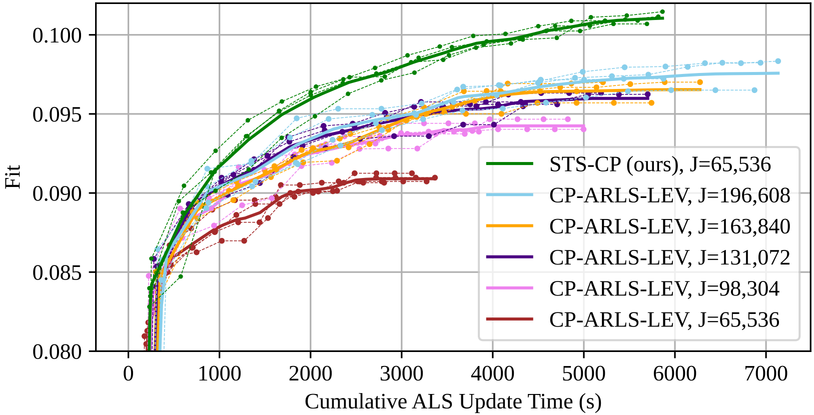
- We give a DS based on a binary tree caching scheme with:
  - $O(IR^2)$ construction time
  - $O(IR)$ storage space
  - $O(R^2 \log(I/R))$ time per query
- We used this data structure twice in succession to sample from the conditional distribution.

Accuracy Comparison for Fixed Sample Count



ALS Accuracy Comparison for $J = 2^{16}$ samples.

Fit vs. ALS Update Time



Fit vs. ALS Update Time, Reddit Tensor, $R = 100$.

Thank you! Read the full paper online.