

A cross-based alternating least squares method for CP tensor decomposition

Bonan Sun (MPI Magdeburg)

SIAM MDS26, Salt Lake City, Nov 16–20, 2026

Based on a joint work with
Peter Benner (MPI Magdeburg)

Organization

1. CP tensor decomposition and ALS
2. Deterministic cross-based sampling
3. Numerical experiments
4. Conclusion

CP tensor decomposition

- Tensors: $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ describe multiway data in chemometrics, neuroscience, parametric PDEs, recommender systems, ...
- CP tensor decomposition with factor matrices:

$$\mathcal{X} \approx \tilde{\mathcal{X}} = \underbrace{\sum_{j=1}^{r} \boldsymbol{a}_j^{(1)} \otimes \cdots \otimes \boldsymbol{a}_j^{(d)}}_{=:[\![A_1,\ldots,A_d]\!]} \in \mathbb{R}^{n_1 \times \cdots \times n_d}, \quad A_k = [\boldsymbol{a}_1^{(k)}, \ldots, \boldsymbol{a}_r^{(k)}] \in \mathbb{R}^{n_k \times r}.$$

CP tensor decomposition

- CP tensor decomposition with factor matrices:

$$\mathcal{X} \approx \tilde{\mathcal{X}} = \sum_{j=1}^{r} \boldsymbol{a}_j^{(1)} \otimes \cdots \otimes \boldsymbol{a}_j^{(d)} \in \mathbb{R}^{n_1 \times \cdots \times n_d}, \quad A_k = [\boldsymbol{a}_1^{(k)}, \ldots, \boldsymbol{a}_r^{(k)}] \in \mathbb{R}^{n_k \times r}.$$

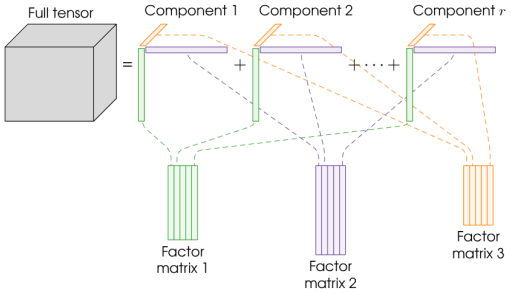


Figure: Taken from Ballard and Kolda, Tensor Decompositions for Data Science, Cambridge University Press, 2025.

CP tensor decomposition

- Tensors: $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ describe multiway data in chemometrics, neuroscience, parametric PDEs, recommender systems, ...
- CP tensor decomposition with factor matrices:

$$\mathcal{X} \approx \tilde{\mathcal{X}} = \underbrace{\sum_{j=1}^{r} \boldsymbol{a}_j^{(1)} \otimes \cdots \otimes \boldsymbol{a}_j^{(d)}}_{=:[\![A_1,\ldots,A_d]\!]} \in \mathbb{R}^{n_1 \times \cdots \times n_d}, \quad A_k = [\boldsymbol{a}_1^{(k)}, \ldots, \boldsymbol{a}_r^{(k)}] \in \mathbb{R}^{n_k \times r}.$$

- Objective: find factors with moderate rank $r \ll n_k$:

$$A_1, \ldots, A_d = \operatorname*{arg\,min}_{A_1,\ldots,A_d} \|\mathcal{X} - [\![A_1, \ldots, A_d]\!]\|_{\mathrm{F}}^2.$$

- Once fitted: data compression $O(dnr)$, interpretation of factors, cheap downstream computations, ...

Alternating least squares (ALS) method

- Objective: find factors with moderate rank $r \ll n_k$:

$$A_1, \ldots, A_d = \underset{A_1,\ldots,A_d}{\arg\min} \|\mathcal{X} - [\![A_1, \ldots, A_d]\!]\|_{\mathrm{F}}^2.$$

- ALS: update one factor at a time while fixing others.

1: Initialize factor matrices A_k (e.g., randomly).
2: **for** $k = 1, 2, \ldots, d$ **do**
3: $\quad A_k = \arg\min_{Y \in \mathbb{R}^{n_k \times r}} \|\mathcal{X} - [\![A_1, \ldots, A_{k-1}, Y, A_{k+1}, \ldots, A_d]\!]\|_{\mathrm{F}}^2$
4: **end for**

- In matrix form, ALS update solves

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \|\mathcal{X}_{(k)} - Y Z_k^\top\|_{\mathrm{F}}^2.$$

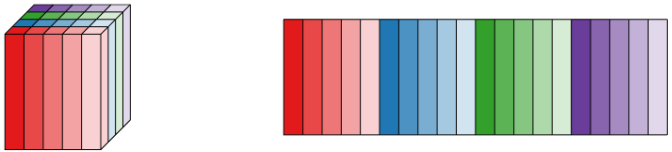
- $\mathcal{X}_{(k)} \in \mathbb{R}^{n_k \times N_k}$: mode-$k$ unfolding of $\mathcal{X}$.
- $Z_k = A_d \odot \cdots \odot A_{k+1} \odot A_{k-1} \odot \cdots \odot A_1 \in \mathbb{R}^{N_k \times r}$, $N_k = \prod_{j \neq k} n_j$.

Alternating least squares (ALS) method

- In matrix form, ALS update solves

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \| \mathcal{X}_{(k)} - Y Z_k^\top \|_{\mathrm{F}}^2.$$

- $\mathcal{X}_{(k)} \in \mathbb{R}^{n_k \times N_k}$: mode-$k$ unfolding of $\mathcal{X}$.



(a) Mode-1 fibers. (b) Mode-1 unfolding.

Figure: Taken from Ballard and Kolda, Tensor Decompositions for Data Science, Cambridge University Press, 2025.

Alternating least squares (ALS) method

- In matrix form, ALS update solves

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \| \mathcal{X}_{(k)} - Y Z_k^\top \|_{\mathrm{F}}^2.$$

- $\mathcal{X}_{(k)} \in \mathbb{R}^{n_k \times N_k}$: mode-$k$ unfolding of $\mathcal{X}$.
- $Z_k = A_d \odot \cdots \odot A_{k+1} \odot A_{k-1} \odot \cdots \odot A_1 \in \mathbb{R}^{N_k \times r}$, $N_k = \prod_{j \neq k} n_j$.
- $\odot$ denotes the Khatri–Rao (columnwise Kronecker) product:

$$A \odot B = [\boldsymbol{a}_1 \otimes \boldsymbol{b}_1,\ \boldsymbol{a}_2 \otimes \boldsymbol{b}_2,\ \ldots,\ \boldsymbol{a}_r \otimes \boldsymbol{b}_r] \in \mathbb{R}^{(mp) \times r} \quad \text{for } A \in \mathbb{R}^{m \times r}, B \in \mathbb{R}^{p \times r}.$$

- Rewriting the LS problem:

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \| Z_k Y^\top - \mathcal{X}_{(k)}^\top \|_{\mathrm{F}}^2.$$

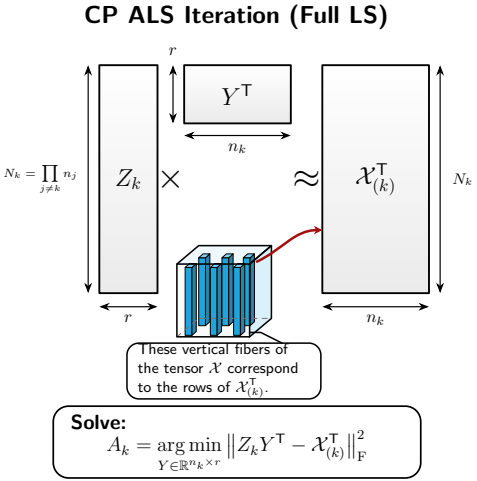
- Very tall and skinny $Z_k \in \mathbb{R}^{N_k \times r}$ with $N_k \gg r$.

Classical CP-ALS: curse of dimensionality

- In matrix form, CP-ALS update solves:

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \|Z_k Y^\top - \mathcal{X}_{(k)}^\top\|_{\mathrm{F}}^2.$$

- Very tall and skinny Z_k with $N_k \gg n_k > r$. Each solve costs $O(n^d)$.
- Result: classical ALS becomes infeasible for moderate d unless we can shrink each LS.



Randomized sampling

- Idea: sample s rows ($s \ll N_k$) to form

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \|Z_k Y^\top - \mathcal{X}_{(k)}^\top\|_{\mathrm{F}}^2.$$

Randomized sampling

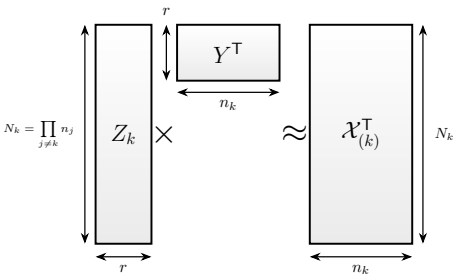
- Idea: sample s rows ($s \ll N_k$) to form

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \|Z_k(\mathcal{I}, :)Y^\top - \mathcal{X}_{(k)}^\top(\mathcal{I}, :)\|_{\mathrm{F}}^2.$$

- Here $Z_k(\mathcal{I}, :)$ denotes rows indexed by $\mathcal{I} \subseteq \{1, \ldots, N_k\}$, $\#\mathcal{I} = s$.

Randomized sampling

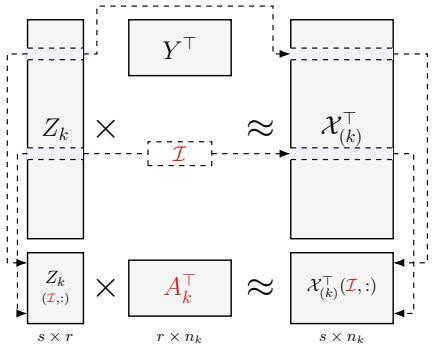
CP ALS Iteration (Full LS)



Solve:

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \left\| Z_k Y^\top - \mathcal{X}_{(k)}^\top \right\|_{\mathrm{F}}^2$$

Sampled CP ALS Iteration (Reduced LS)



Solve:

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \left\| Z_k(\mathcal{I},:) Y^\top - \mathcal{X}_{(k)}^\top(\mathcal{I},:) \right\|_{\mathrm{F}}^2$$

Randomized sampling

- Idea: sample s rows ($s \ll N_k$) to form

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \| Z_k(\mathcal{I}, :) Y^\top - \mathcal{X}_{(k)}^\top(\mathcal{I}, :) \|_{\mathrm{F}}^2.$$

- Here $Z_k(\mathcal{I}, :)$ denotes rows indexed by $\mathcal{I} \subseteq \{1, \ldots, N_k\}$, $\#\mathcal{I} = s$.
- Randomized choices (uniform[1], leverage-score[2]) reduce cost but:
  - Need $s \gg r$.
  - Variance: unlucky draws miss informative fibers.
- Can we deterministically select exactly $s = r$ informative rows instead?

[1] Battaglino, Ballard, Kolda, A practical randomized CP tensor decomposition, SIMAX, 2018.
[2] Larsen, Kolda, Practical leverage-based sampling for low-rank tensor decomposition, SIMAX, 2022.



Organization

1. CP tensor decomposition and ALS
2. Deterministic cross-based sampling
3. Numerical experiments
4. Conclusion

Selecting exactly r informative rows

- Recall ALS update:

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \|Z_k Y^\top - \mathcal{X}_{(k)}^\top\|_{\mathrm{F}}^2.$$

- Goal: choose $\mathcal{I} \subseteq \{1, \ldots, N_k\}$ "wisely" with $\#\mathcal{I} = r$ to solve instead

$$A_k = \underset{Y \in \mathbb{R}^{n_k \times r}}{\arg\min} \|Z_k(\mathcal{I}, :) Y^\top - \mathcal{X}_{(k)}^\top(\mathcal{I}, :)\|_{\mathrm{F}}^2.$$

Lemma (Following Lemma 4.1 of [Sorensen/Embree'16, A DEIM induced CUR factorization])

The sampled LS solution A_k is not much worse than the full LS solution A_k:

$$\|Z_k A_k{}^\top - \mathcal{X}_{(k)}^\top\|_{\mathrm{F}} \leq \|Z_k Z_k(\mathcal{I}, :)^{-1}\|_2 \, \|Z_k A_k^\top - \mathcal{X}_{(k)}^\top\|_{\mathrm{F}}.$$

- The quality of A_k relies on $\|Z_k Z_k(\mathcal{I}, :)^{-1}\|_2$. How to select $\mathcal{I}$ to control this norm?

`maxvol` provides the needed control

Lemma (Following Lemma 4.1 of [Sorensen/Embree'16, A DEIM induced CUR factorization])

The sampled LS solution A_k is not much worse than the full LS solution A_k:

$$\|Z_k A_k{}^\top - \mathcal{X}_{(k)}^\top\|_{\mathrm{F}} \le \|Z_k Z_k(\mathcal{I},:)^{-1}\|_2 \, \|Z_k A_k^\top - \mathcal{X}_{(k)}^\top\|_{\mathrm{F}}.$$

- Goal: select $\mathcal{I}$ to control the spectral norm $\|Z_k Z_k(\mathcal{I},:)^{-1}\|_2$.
- `maxvol`[3] greedily finds a submatrix with (approx.) maximal volume.

Lemma (Following Lemma 2.1 of [Goreinov/Tyrtyshnikov/Zamarashkin'97])

If $\mathcal{I} = \texttt{maxvol}_\mu(A)$ for $A \in \mathbb{R}^{m\times r}$, then $\|AA(\mathcal{I},:)^{-1}\|_2 \le \sqrt{1+\mu^2(m-r)r}$ (tight bound).

- μ: slightly larger than 1 as a tolerance (e.g. $\mu = 1.01$). Cost per call: $O(mr^2)$.
- Perfect for our goal—if only we could run it on $Z_k \in \mathbb{R}^{N_k\times r}$, $N_k = \prod_{j\neq k} n_j$!

[3] Goreinov, Oseledets, Savostyanov, Tyrtyshnikov, Zamarashkin, How to find a good submatrix, 2010.

Why we cannot apply `maxvol` directly

- Recall
$$Z_k = A_d \odot \cdots \odot A_{k+1} \odot A_{k-1} \odot \cdots \odot A_1 \in \mathbb{R}^{N_k \times r}, \quad N_k = \prod_{j \neq k} n_j.$$
- Z_k has $N_k = \prod_{j \neq k} n_j$ rows: even storing one column is prohibitive.
- A single $\texttt{maxvol}(Z_k)$ call would cost $O(N_k r^2)$ and require explicit Khatri–Rao evaluation.
- Need a structured workaround that never forms Z_k but inherits (close to) `maxvol` guarantees.
- Solution: leverage the Khatri–Rao structure of Z_k to build a hierarchical sampling scheme.

Hierarchical selection via Khatri–Rao tree

- Recall

$$Z_k = \underbrace{A_d \odot \cdots \odot A_{k+1}}_{=Z_{\geq k+1}} \odot \underbrace{A_{k-1} \odot \cdots \odot A_1}_{=Z_{\leq k-1}} \in \mathbb{R}^{N_k \times r}, \quad N_k = \prod_{j \neq k} n_j.$$

- With base cases $Z_{\geq d} = A_d$, $Z_{\leq 1} = A_1$, recursively build:

$$Z_{\geq k+1} = Z_{\geq k+2} \odot A_{k+1}, \quad Z_{\leq k-1} = A_{k-1} \odot Z_{\leq k-2},$$

- Let $\hat{Z}_{\geq d} = \texttt{maxvol}(A_d) \in \mathbb{R}^{r \times r}$, $\hat{Z}_{\leq 1} = \texttt{maxvol}(A_1) \in \mathbb{R}^{r \times r}$. Recursively compute:

$$\hat{Z}_{\geq k+1} = \texttt{maxvol}(\hat{Z}_{\geq k+2} \odot A_{k+1}) \in \mathbb{R}^{r \times r}, \quad \hat{Z}_{\leq k-1} = \texttt{maxvol}(A_{k-1} \odot \hat{Z}_{\leq k-2}) \in \mathbb{R}^{r \times r}.$$

- Final merging: $\hat{Z}_k = \texttt{maxvol}(\hat{Z}_{\geq k+1} \odot \hat{Z}_{\leq k-1})$.
- Note: $\hat{Z}_{\geq k+1}, \hat{Z}_{\leq k-1}$ can be stored and reused across ALS iterations.

The CP-ACLS algorithm

Algorithm 1 CP-ACLS with hierarchical `maxvol` sampling

for $k = d, \ldots, 2$ **do** ▷ Initialize upper chain $O(dnr^3)$
  $\hat{Z}_{\geq k} = \texttt{maxvol}(\hat{Z}_{\geq k+1} \odot A_k)$
end for
repeat
  for $k = 1, \ldots, d$ **do** ▷ forward sweep
    $\hat{Z}_k = \texttt{maxvol}(\hat{Z}_{\geq k+1} \odot \hat{Z}_{\leq k-1})$ (for $k \neq 1, d$) ▷ $O(r^4)$
    Compute physical indices $\mathcal{I}$ s.t. $\hat{Z}_k = Z_k(\mathcal{I}, :)$.
    Solve sampled LS: $A_k = \arg\min_{Y \in \mathbb{R}^{n_k \times r}} \|\hat{Z}_k Y^\top - \mathcal{X}^\top_{(k)}(\mathcal{I}, :)\|_{\mathrm{F}}^2$. ▷ $O(nr^2 + r^3)$
    $\hat{Z}_{\leq k} = \texttt{maxvol}(A_k \odot \hat{Z}_{\leq k-1})$ (for $k \neq d$) ▷ update lower chain $O(nr^3)$
  end for
  Do backward sweep $k = d-1, \ldots, 1$ similarly ▷ reuse lower chain, update upper chain
until maximum epochs or tolerance satisfied

Error guarantees of hierarchical `maxvol`

Theorem

Assume every `maxvol` *call is performed with tolerance* $\mu \geq 1$. *Then the selected rows* $\hat{Z}_k$ *of* Z_k *satisfy*

$$\|Z_k \hat{Z}_k^{-1}\|_2 \leq \sqrt{1 + \mu^{2d} r^{2(d-1)}(n^{d-1} - r)r},$$

therefore

$$\|Z_k A_k^\top - \mathcal{X}_{(k)}^\top\|_{\mathrm{F}} \leq \sqrt{1 + \mu^{2d} r^{2(d-1)}(n^{d-1} - r)r}\, \|Z_k A_k^\top - \mathcal{X}_{(k)}^\top\|_{\mathrm{F}}.$$

- Bound is very pessimistic, but in practice growth is much milder.
- Even directly applying DEIM (or LUPP, QRCP) on full Z_k yields similar bounds:

$$\|Z_k \hat{Z}_k^{-1}\|_2 \leq \sqrt{n^{d-1} r}\, 2^{r-1}.$$

- Cost: $O(dnr^3)$ per sweep.

Cross vs. random sampling for CP-ALS

	CP-ACLS	CP-ARLS (lev)[4]	CP-ARLS (uni)[5]
LS accuracy $\|\hat{R}\|/\|R\| \leq$	$1+\mu^d r^d n^{d/2}$	$1+\varepsilon$ (w.p. $1-\delta$)	None
# samples	r	$r^d \max\{1700 \cdot \log(r/\delta), 1/(\delta\varepsilon)\}$	None
Complexity per sweep	$O(dnr^3)$	$O(dsnr)$	$O(dsnr)$

- CP-ACLS needs only r samples, but with a **pessimistic** accuracy guarantee.
- CP-ARLS (lev) guarantees accuracy with high $\mathbb{P}$, but needs **pessimistic** # samples.
- CP-ACLS may be more efficient when $s > r^2$.
- Not clear a priori which method is better from theory alone.
- Cross-based methods are very successful in the world of tensor trains, so promising to try here too!

[4] Larsen, Kolda, Practical leverage-based sampling for low-rank tensor decomposition, SIMAX, 2022.
[5] Battaglino, Ballard, Kolda, A practical randomized CP tensor decomposition, SIMAX, 2018.



Organization

1. CP tensor decomposition and ALS
2. Deterministic cross-based sampling
3. Numerical experiments
4. Conclusion

Numerical experiments: setting

- Use MATLAB TensorToolbox[6] implementation of CP-ALS, ARLS (lev), ARLS (uni).
- Implemented CP-ACLS as a new solver in the toolbox.
- Default sample size for CP-ARLS (lev) is $s = 2^{17}$, CP-ARLS (uni) is $s = 10r\log(r)$.
- Compare randomized methods at $s = r$ and $s = r^2$ (vs. $s = r$ for CP-ACLS).
- Measure of fit (close to 1 is better):

$$\text{Fit} = 1 - \frac{\|\mathcal{X} - \tilde{\mathcal{X}}\|_{\text{F}}}{\|\mathcal{X}\|_{\text{F}}} \in [0, 1].$$

[6] Bader, Kolda, MATLAB Tensor Toolbox Version 3.7, 2025.

Experiment 1: synthetic tensor

- Standard collinearity benchmark from CP-ARLS paper.
- Generate rank-5 CP tensor of size $400 \times 400 \times 400$ with collinearity 0.9 and 10% relative Gaussian noise.
- Rank-5 CP fits from shared random initializations (medians over 5 runs).

Method	Samples s	Time (s)	Fit
CP-ACLS	$5\ (r)$	0.444	0.8855
CP-ARLS (lev)	$5\ (r)$	0.367	0.6155
CP-ARLS (uni)	$5\ (r)$	0.230	0.5131
CP-ARLS (lev)	$25\ (r^2)$	0.436	0.8658
CP-ARLS (uni)	$25\ (r^2)$	0.312	0.8616
CP-ALS	full (1.6×10^5)	41.57	0.9002

Experiment 2: turbulent flow tensor

- An example tensor from TensorToolbox documentation.
- Dataset: Miranda turbulent flow tensor ($2048 \times 256 \times 256$) from SDRBench[7].
- Rank-10 CP fits from shared random initializations (medians over 5 runs).

Method	Samples s	Time (s)	Fit
CP-ACLS	$10\ (r)$	0.932	0.8545
CP-ARLS (lev)	$10\ (r)$	0.604	0.5474
CP-ARLS (uni)	$10\ (r)$	0.401	0.6225
CP-ARLS (lev)	$100\ (r^2)$	1.966	0.9151
CP-ARLS (uni)	$100\ (r^2)$	1.654	0.9132
CP-ALS	full (6.6×10^4)	193.1	0.9270

[7] K. Zhao et al., SDRBench: Scientific Data Reduction Benchmark for Lossy Compressors, IEEE Big Data 2020.

Experiment 3: discretization of a function

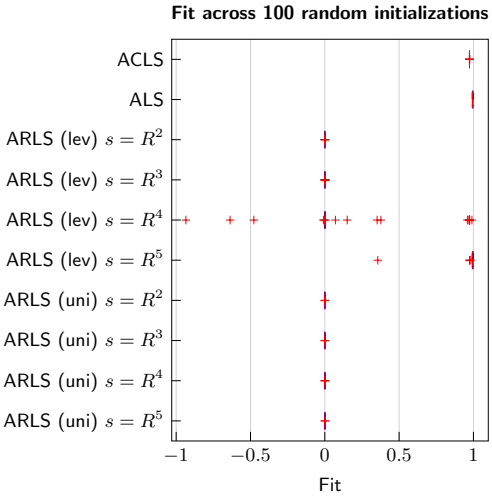
- An example from Chebfun3F paper[8].
- Consider

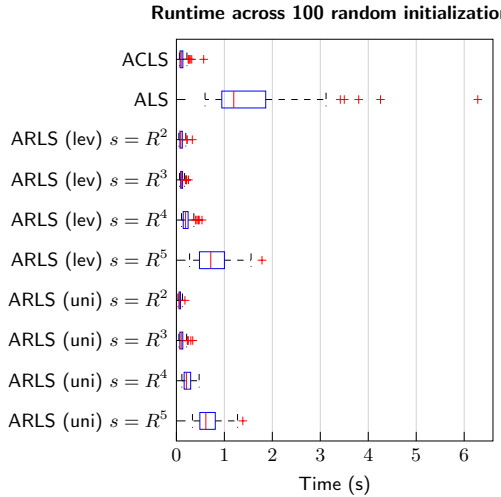
$$f(x_1, \ldots, x_d) = \frac{1}{10^{-5} + \sum_{k=1}^{d} x_k^2}, \quad x_k \in [-1, 1].$$

- Discretize f with $d = 5$ and $n = 30$ points in each dimension on Chebyshev grid.
- Rank-5 CP fits for resulting tensor of size $30 \times 30 \times 30 \times 30 \times 30$.
- Vary sample sizes for CP-ARLS (lev) and CP-ARLS (uni) $s = r^2, r^3, r^4, r^5$.

[8] Dolgov, Kressner, Strössner, Functional Tucker approximation using Chebyshev interpolation, SISC, 2021.

Experiment 3: discretization of a function







Organization

1. CP tensor decomposition and ALS
2. Deterministic cross-based sampling
3. Numerical experiments
4. Conclusion

Conclusions and outlook

- Proposed CP-ACLS: a cross-based ALS variant selecting exactly r rows per LS via hierarchical `maxvol` calls on Khatri–Rao factors.
- Hierarchical `maxvol` selection costs $O(dnr^3)$ per sweep and provides a deterministic residual bound.
- Numerical experiments demonstrate competitive runtimes and accuracy at the smallest sampling budget.
- Natural extensions include oversampled cross selection and other pivoting methods.

Thank you for your attention!