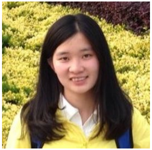
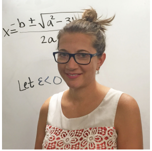
Getting more with less: matrix and tensor algorithms from subsampling modes

Keaton Hamm

University of Texas at Arlington

Warwick Algorithms Seminar

HanQin Cai
UCF
Math

Longxiu Huang
Michigan State
Math

Jiaqi Li
Sun-Yat Sen
Data and Comp. Sci

Deanna Needell
UCLA
Math

Tao Wang
Sun-Yat Sen
Data and Comp. Sci

Sponsor

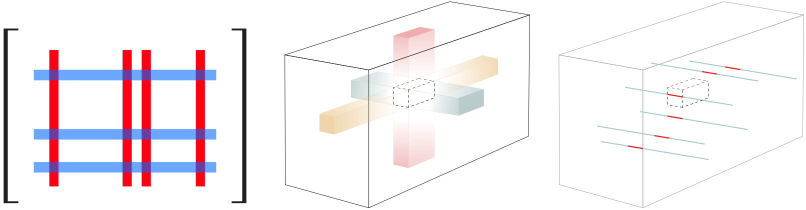
Premise

$$D = \sum_{1}^{k}$$  $$+$$ 

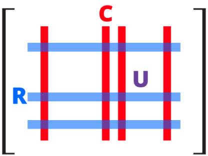
`data = low-rank + noise`

Paradigm

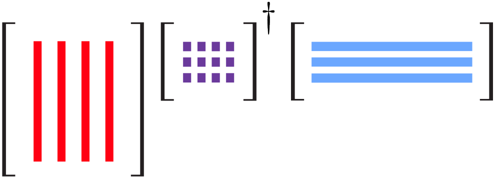
Algorithms that only view small subsets of the full data matrix or tensor



What is a CUR decomposition?



Theorem (folklore): If $\mathrm{rank}(U) = \mathrm{rank}(L)$, then $L = CU^{\dagger}R$



Proof

Case when U is invertible:

Proof

Case when U is invertible:
Columns of C form a basis for $\mathrm{Col}(A)$

Proof

Case when U is invertible:
Columns of C form a basis for $\mathrm{Col}(A)$

$$A = CX$$

Proof

Case when U is invertible:
Columns of C form a basis for Col(A)

$$A = CX$$

P_r - row selection matrix: $P_r A = R$

Proof

Case when U is invertible:
Columns of C form a basis for $\mathrm{Col}(A)$

$$A = CX$$

P_r - row selection matrix: $P_r A = R$

$$\begin{aligned} A = CX &\Leftrightarrow P_r A = P_r CX \\ &\Leftrightarrow R = UX \end{aligned}$$

Proof

Case when U is invertible:
Columns of C form a basis for Col(A)

$$A = CX$$

P_r - row selection matrix: $P_r A = R$

$$\begin{aligned} A = CX &\Leftrightarrow P_r A = P_r C X \\ &\Leftrightarrow R = UX \end{aligned}$$

$X = U^{-1} R$ is a solution to $R = UX$

Proof

Case when U is invertible:
Columns of C form a basis for Col(A)

$$A = CX$$

P_r - row selection matrix: $P_r A = R$

$$\begin{aligned} A = CX &\Leftrightarrow P_r A = P_r CX \\ &\Leftrightarrow R = UX \end{aligned}$$

$X = U^{-1}R$ is a solution to $R = UX$
Therefore $A = CU^{-1}R$.

Key Idea

- Use submatrices for reconstruction/approximation
- Choose “good" columns/rows that represent the matrix well

Key Idea

- Use submatrices for reconstruction/approximation
- Choose “good" columns/rows that represent the matrix well

Why?

- Interpretable representations
- Kernel matrix approximation
- Fast approximation to the SVD!
- Robust low-rank matrix approximation
- Preserves some structures (e.g., sparsity)

Key Idea

- Use submatrices for reconstruction/approximation
- Choose “good" columns/rows that represent the matrix well

Why?

- Interpretable representations
- Kernel matrix approximation
- Fast approximation to the SVD!
- Robust low-rank matrix approximation
- Preserves some structures (e.g., sparsity)

Applications

- Subspace Clustering
- Computer Vision Applications (Motion Segmentation, Facial Recognition)
- Sketching of massive data
- Image processing

Key Themes

- (Mildly) Oversampling (pk columns) is your friend: gives good approximations to truncated SVD (of order k)
- Good for (approximately) low-rank matrices – bad for full-rank matrices
- Randomized or hybrid random + deterministic column sampling is your friend
- Interpretability

Related Work

$A = CX$ – interpolative decompositions [Voronin–Martinsson, ACOM '17]

$A = CUR$, $C = A(:, J)$, $R = A(I, :)$, $U =$??? – CUR decompositions [Drineas–Mahoney–Muthukrishnan, SIMAX '08]

Synonyms/intimately related names

- Cross Approximation [Tyrtyshnikov, Computing '00]
- (Pseudo)skeleton decomposition [Goreinov–Tyrtyshnikov–Zamarashkin, LAA '97]
- Nyström method (when A is SPSD) [Williams–Seeger, NeurIPS '00]

Generalizations

- Generalized CUR decompositions [Gidisu–Hochstenbach, '22]
- Meta factorization [Karpowicz, '22]

Choosing U in CUR

Natural choice I: $U = A(I,J)^\dagger \quad (A \approx CU^\dagger R)$

Natural choice II: $U = C^\dagger A R^\dagger \quad (A \approx CC^\dagger AR^\dagger R)$

$$\underset{Z}{\operatorname{argmin}} \|A - CZR\|_F = C^\dagger A R^\dagger$$

Characterization

Theorem (H–Huang, ACHA '20)

Let $A \in \mathbb{R}^{m \times n}$ and $I \subseteq [m]$, $J \subseteq [n]$. Let $C = A(:, J)$, $U = A(I, J)$, and $R = A(I, :)$. Then the following are equivalent:

1. $\text{rank}(U) = \text{rank}(A)$,
2. $A = CU^\dagger R$,
3. $A = CC^\dagger AR^\dagger R$,
4. $A^\dagger = R^\dagger UC^\dagger$,
5. $\text{rank}(C) = \text{rank}(R) = \text{rank}(A)$,

Moreover, if any of the equivalent conditions above hold, then $U^\dagger = C^\dagger AR^\dagger$.

How do we choose columns/rows?

Random Sampling Methods I: Sample w/ or w/out replacement from some distribution over the column indices

How do we choose columns/rows?

Random Sampling Methods I: Sample w/ or w/out replacement from some distribution over the column indices

- Uniform: $p_i := \frac{1}{n}$

How do we choose columns/rows?

Random Sampling Methods I: Sample w/ or w/out replacement from some distribution over the column indices

- Uniform: $p_i := \frac{1}{n}$
- Column Length: $p_i = \frac{\|A(:,i)\|_2^2}{\|A\|_F^2}$

How do we choose columns/rows?

Random Sampling Methods I: Sample w/ or w/out replacement from some distribution over the column indices

- Uniform: $p_i := \frac{1}{n}$
- Column Length: $p_i = \frac{\|A(:,i)\|_2^2}{\|A\|_F^2}$
- Leverage Scores: $p_i^{(k)} := \frac{1}{k}\|V_k(i,:)\|_2^2$

How do we choose columns/rows?

Random Sampling Methods I: Sample w/ or w/out replacement from some distribution over the column indices

- Uniform: $p_i := \frac{1}{n}$
- Column Length: $p_i = \frac{\|A(:,i)\|_2^2}{\|A\|_F^2}$
- Leverage Scores: $p_i^{(k)} := \frac{1}{k}\|V_k(i,:)\|_2^2$

Random Sampling Methods II: Bernoulli trials on each column

- Typically Column Length – requires rescaling columns in the reconstruction phase

How do we choose columns/rows?

Random Sampling Methods I: Sample w/ or w/out replacement from some distribution over the column indices

- Uniform: $p_i := \dfrac{1}{n}$
- Column Length: $p_i = \dfrac{\|A(:,i)\|_2^2}{\|A\|_F^2}$
- Leverage Scores: $p_i^{(k)} := \dfrac{1}{k}\|V_k(i,:)\|_2^2$

Random Sampling Methods II: Bernoulli trials on each column

- Typically Column Length – requires rescaling columns in the reconstruction phase

Deterministic Sampling Methods:

- Discrete Empirical Interpolation Method (DEIM) [Gu–Eisenstat, SICOMP '96, Sorensen–Embree, SICOMP '16]
- Greedy Column Selection [Avron–Boutsidis, SIMAX '13]

Tradeoffs:

- Computational Complexity: Leverage $\gg$ Col Length $\gg$ Uniform

Tradeoffs:

- Computational Complexity: Leverage $\gg$ Col Length $\gg$ Uniform
- Guarantees: Leverage Scores $>$ Col Length $>$ Uniform

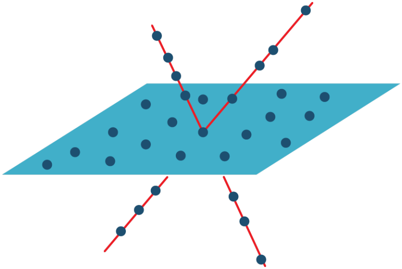
Tradeoffs:

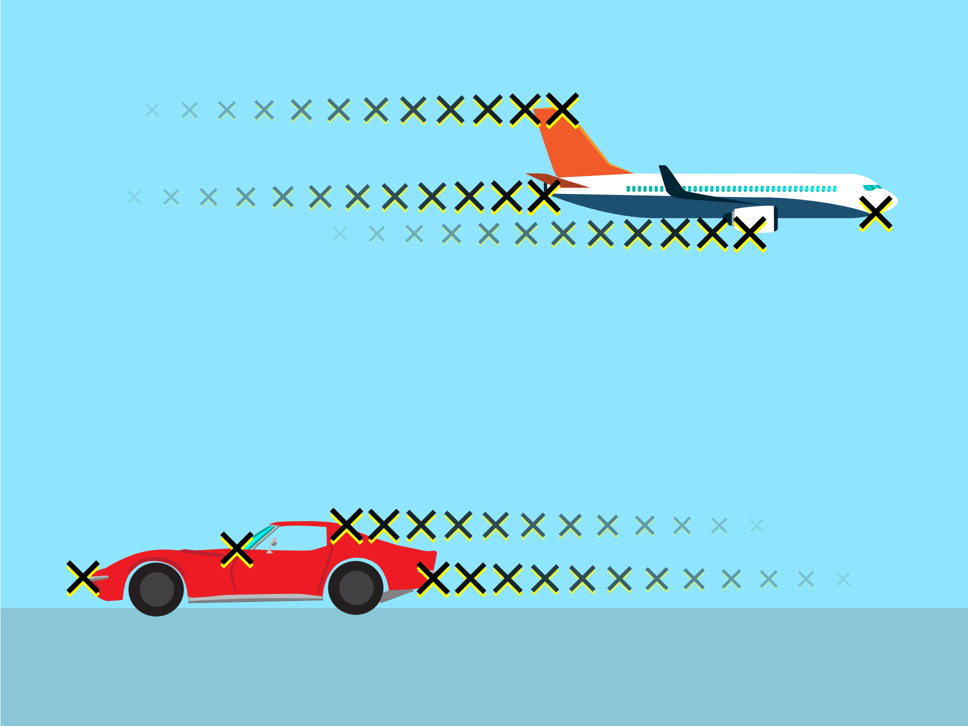
- Computational Complexity: Leverage $\gg$ Col Length $\gg$ Uniform
- Guarantees: Leverage Scores $>$ Col Length $>$ Uniform
- Oversampling Factor (p): Leverage $<$ Col Length $\approx$ Uniform

The Subspace Clustering Problem

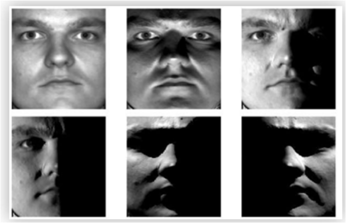
Goals:

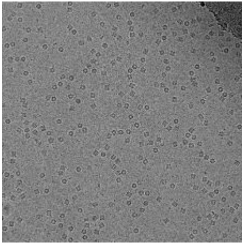
- # of Subspaces?
- $\dim(S_i)$?
- Basis for S_i?
- Cluster data $\{w_i\}_{i=1}^n$.

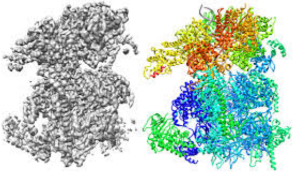




Other Applications







Tron–Vidal, CVPR ’07
Basri–Jacobs, TPAMI ’03
Hadani–Singer, Annals ’11

Meta-Theorem

Suppose A has columns drawn from a union of subspaces $\bigcup_{i=1}^{L} S_i \subset \mathbb{R}^n$. Under idealized assumptions on the subspaces, columns of A can be clustered via the representation $A = CX$. That is, one can find an assignment function Π such that $\Pi(a_i) = k$ iff $a_i \in S_k$.

- Elhamifar–Vidal, CVPR '09, TPAMI '13
- Liu–Lin–Yu, ICML '10
- Aldroubi–Sekmen–Koku–Çakmak, ACHA '18
- Aldroubi–H–Koku–Sekmen, Frontiers '19

Key takeaway: These algorithms are fast and robust to noise

Problem (Robust PCA)

$$D = L + S = \texttt{low-rank} + \texttt{sparse}$$

$$D = \sum_{1}^{k}$$

$$+$$

Problem (Robust PCA)

$$D = L + S = \texttt{low-rank} + \texttt{sparse}$$

$$D = \sum_{1}^{k}$$  $+$ 

Problem (Robust PCA)

$D = \sum_{1}^{k}$  $+$ 

Nonconvex Formulation

$$\min \|D - L - S\|_F \quad \text{s.t.} \quad \text{rank}(L) \leq r, \quad \|S\|_0 \leq \alpha n^2$$

Convex Relaxation[1]

$$\min \|L\|_* + \lambda \|S\|_1 \quad \text{s.t.} \quad L + S = D$$

---

[1] Candès et al. Journal of ACM '11

Properties

Incoherence of L

$$\mu_1(L) := \max_i \sqrt{\frac{n}{r}}\, \|U_r(i,:)\|_2 \qquad \mu_2(L) := \max_i \sqrt{\frac{n}{r}}\, \|V_r(i,:)\|_2$$

Sparsity of S

$$\max_i \|S(i,:)\|_0 \leq \alpha n \quad \text{and} \quad \max_j \|S(:,j)\|_0 \leq \alpha n$$

Robust CUR (RCUR)[2]

Parameter: RPCA – your favorite Robust PCA algorithm

Initialize: Sample $O(\mu r \log n)$ row and column indices (I, J, respectively) uniformly at random, $\widetilde{C} = D(:, J)$, $\widetilde{R} = D(I, :)$

$\widehat{L}(:, J), \widehat{S}(:, J) = \mathrm{RPCA}(\widetilde{C}, r)$
$\widehat{L}(I, :), \widehat{S}(I, :) = \mathrm{RPCA}(\widetilde{R}, r)$

Return : $\widehat{L}(:, J)(\widehat{L}(I, J))^{\dagger}\widehat{L}(I, :)$

Complexity: $O(r^3 n \log^2 n)$ (if using AltProj or AccAltProj as RPCA)

[2] Cai–H–Huang–Needell, SIIMS '21

Robust CUR (RCUR)[3]

Need to understand:

- How incoherence and sparsity transfer to submatrices
- The quantity $\beta := \sqrt{\frac{|J|}{n}}\|V_r(J,:)^\dagger\|_2$

Tools:

- Basic Linear Algebra
- Tropp's estimates on norms of pseudoinverses of submatrices of orthogonal matrices[4]

[3] Cai–H–Huang–Needell, SIIMS '21
[4] Tropp, Advances in Adaptive Data Analysis, '11

Robust CUR (RCUR)[3]

Need to understand:

- How incoherence and sparsity transfer to submatrices
- The quantity $\beta := \sqrt{\frac{|J|}{n}} \|V_r(J,:)^\dagger\|_2$

Tools:

- Basic Linear Algebra
- Tropp's estimates on norms of pseudoinverses of submatrices of orthogonal matrices[4]

Theorem [Tropp]: If L has incoherence $\mu_2(L)$ and $|J| \geq c\mu_2 r$ is sampled uniformly without replacement, then

$$\mathbb{P}\left(\beta \leq \frac{1}{\sqrt{1-\delta}}\right) \geq 1 - r\left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^c, \quad \text{for all} \quad \delta \in [0,1).$$

[3] Cai–H–Huang–Needell, SIIMS '21

[4] Tropp, Advances in Adaptive Data Analysis, '11

Robust CUR (RCUR)[5]

Theorem [Cai et al.]: If L has incoherence $\mu_1(L), \mu_2(L)$ and $|J| \geq c\mu_2 r \log(rn)$ is sampled uniformly without replacement, $C = L(:, J)$, then with probability $\geq 1 - \frac{1}{n}$,

$$\mu_1(C) \leq \mu_1(L), \qquad \mu_2(C) \leq 100\kappa(L)^2\mu_2(L).$$

Theorem [Cai et al.]: Under some relations on $\alpha, \kappa(L), \mu_1(L), \mu_2(L)$, if $|I|, |J| \gtrsim \mu_i(L) r \log n$ are sampled uniformly without replacement and AltProj is used as RPCA. Then RCUR outputs $\widehat{L}$ such that w.h.p.,

$$\frac{\|L - \widehat{L}\|_2}{\|L\|_2} \leq \varepsilon\kappa(L)^{-1}.$$

[5] Cai–H–Huang–Needell, SIIMS '21

Iterated Robust CUR (IRCUR)[6]

Initialize: Sample $O(\mu r \log n)$ row and column indices (I, J, respectively) uniformly at random, $C_0 = R_0 = U_0 = 0$

for $k = 1 : N$

$$
\begin{aligned}
C_k &= (D - S_{k-1})(:, J) \\
R_k &= (D - S_{k-1})(I, :) \\
U_k &= \texttt{TruncatedSVD}(D - S_{k-1})(I, J) \\
L_k &= C_k U_k^{\dagger} R_k
\end{aligned}
$$

$$
\begin{aligned}
S_k(I, :) &= \texttt{HardThreshold}(D - L_k)(I, :) \\
S_k(:, J) &= \texttt{HardThreshold}(D - L_k)(:, J)
\end{aligned}
$$

Return: C_N, U_N, R_N, S_N

Complexity: $O(r^2 n \log^2 n)$

[6] Cai–H–Huang–Li–Wang, IEEE SPL, '21

Iterated Robust CUR (IRCUR)[7]

Implementation Notes

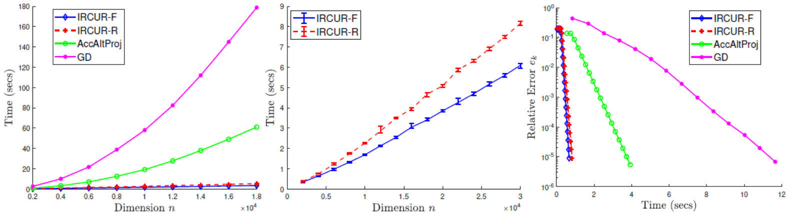
- L_k is never formed, only C_k, U_k, and R_k are formed and stored
- Optional element to resample columns/rows at each iteration (work on different parts of L and S)

[7] Cai–H–Huang–Li–Wang, IEEE SPL, '21

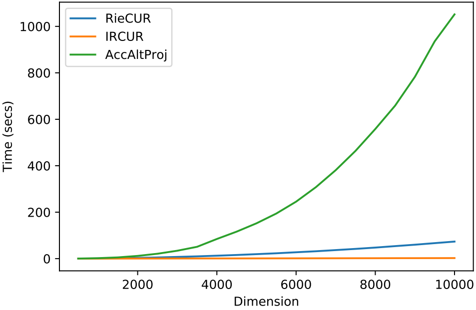
Numerics

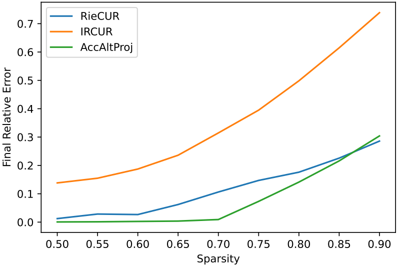
	AltProj	AccAltProj	RCUR	IRCUR	RieCUR
Complexity	$r^2 n^2$	rn^2	$r^3 n \log^2 n$	$r^2 n \log^2 n$	$r^2 n \log^2 n$

Experiments



Experiments

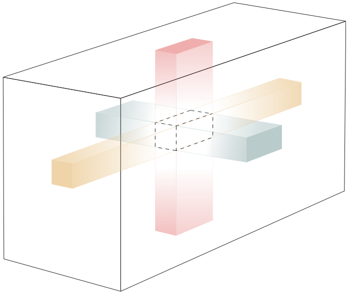




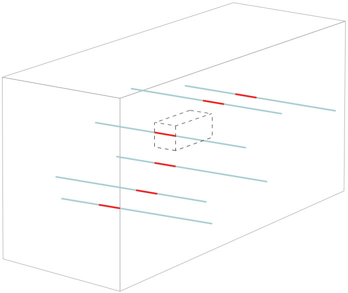
Experiments

	frame size	frame number	runtime (sec)			
			IRCUR-F	IRCUR-R	AccAltProj	GD
S	256×320	1000	2.03	2.16	23.04	93.18
R	120×160	3055	0.82	0.88	15.96	58.37

Tensors



Chidori CUR Decomposition[8]



Fiber CUR Decomposition

$$\mathcal{L} = \mathcal{R} \times_1 \mathcal{C}_{(1)}^{(1)} \mathcal{R}_{(1)} \times_2 \cdots \times_n \mathcal{C}_{(n)}^{(n)} \mathcal{R}_{(n)}$$

$$\mathcal{L} = \mathcal{R} \times_1 \mathcal{C}_{(1)}^{(1)} U^{(1)} \times_2 \cdots \times_n \mathcal{C}_{(n)}^{(n)} U^{(n)}$$

[8] Thanks to Dustin Mixon for this name!

Image/Hyperspectral Image Compression

Original Fiber CUR Chidori CUR HOSVD

		Ribeira	Braga	Ruivaes
Size		$1017 \times 1340 \times 33$	$1021 \times 1338 \times 33$	$1017 \times 1338 \times 33$
Rank		$(60, 60, 7)$	$(60, 60, 5)$	$(65, 65, 4)$
Runtime (seconds)	Fiber CUR	**0.29**	**0.26**	**0.31**
	Chidori CUR	0.66	0.59	0.55
	HOSVD	1.49	1.41	1.42
	st_HOSVD	0.83	0.77	0.76
	HOOI	2.29	2.67	3.30
SNR (dB)	Fiber CUR	24.14	17.93	15.53
	Chidori CUR	**24.39**	**18.56**	**15.84**
	HOSVD	22.99	17.70	15.48
	st_HOSVD	22.18	17.90	15.49
	HOOI	24.33	18.00	15.61

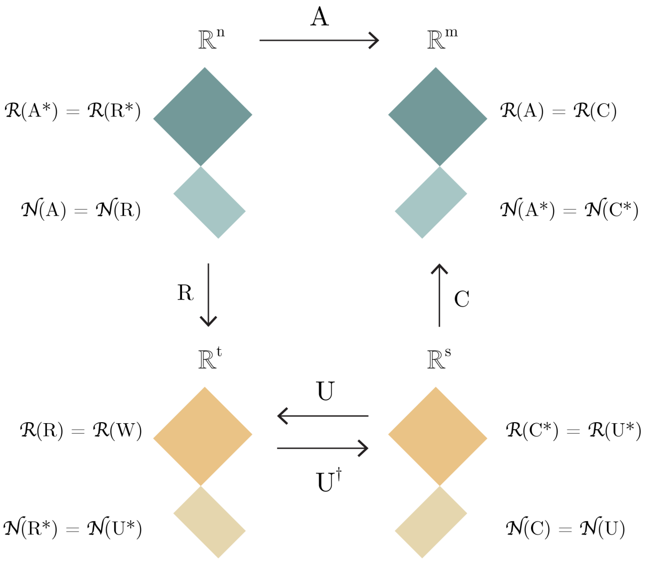
Future and Related Work

- Proof of convergence for IRCUR and RieCUR
  - Theorem for AccAltProj: Under certain relations on parameters $\alpha, \mu, r, n, \sigma_1(L), \sigma_1(D)$, initialization via AltProj is sufficiently good to guarantee linear convergence of L_k and S_k to L and S
- Further extension to tensors of IRCUR and RieCUR[9]
- Extensions to matrix/tensor completion (Cai et al., Henneberger et al.)

[9] Initial experimental work: Cai et al. ICCV '21

Thanks!

- H, Generalized pseudoskeleton decompositions, LAA '23
- H–Meskini–Cai, Riemannian CUR Decompositions for Robust Principle Component Analysis, ICML Workshop on Topology, Algebra, and Geometry in Machine Learning, '22
- Cai–H–Huang–Needell, Robust CUR Decompositions: Theory and Imaging Applications, SIIMS '21
- Cai–H–Huang–Needell, Mode-wise Tensor Decompositions: Multi-dimensional Generalizations of CUR Decompositions, JMLR '21
- Cai–H–Huang–Li–Wang, Rapid Robust Principal Component Analysis: CUR Accelerated Inexact Low Rank Estimation, IEEE SPL '20
- H–Huang, Stability of sampling for CUR Decompositions, Foundations of Data Science '20
- H–Huang, Perturbations of CUR Decompositions, SIMAX '20
- H–Huang, Perspectives on CUR Decompositions, ACHA '20


$\mathbb{R}^n$
A
$\mathbb{R}^m$
$\mathcal{R}(A^*) = \mathcal{R}(R^*)$
$\mathcal{R}(A) = \mathcal{R}(C)$
$\mathcal{N}(A) = \mathcal{N}(R)$
$\mathcal{N}(A^*) = \mathcal{N}(C^*)$
R
C
$\mathbb{R}^t$
$\mathbb{R}^s$
U
$\mathcal{R}(R) = \mathcal{R}(W)$
$\mathcal{R}(C^*) = \mathcal{R}(U^*)$
$U^\dagger$
$\mathcal{N}(R^*) = \mathcal{N}(U^*)$
$\mathcal{N}(C) = \mathcal{N}(U)$