Structured Matrix Approximation from Matrix-Vector Products

New York University, Christopher Musco

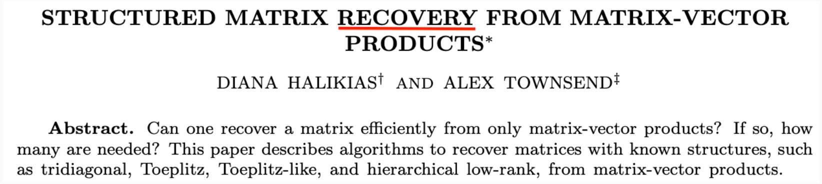
COLLABORATORS

Noah Amsel, Tyler Chen, Feyza Duman Keles, Diana Halikias, David Persson, Cameron Musco

Paper to appear at SODA 2025. Available at:
https://arxiv.org/abs/2407.04686.

PROBLEM WE ARE STUDYING

Starting point:



STRUCTURED MATRIX RECOVERY FROM MATRIX-VECTOR PRODUCTS[*]

DIANA HALIKIAS[†] AND ALEX TOWNSEND[‡]

Abstract. Can one recover a matrix efficiently from only matrix-vector products? If so, how many are needed? This paper describes algorithms to recover matrices with known structures, such as tridiagonal, Toeplitz, Toeplitz-like, and hierarchical low-rank, from matrix-vector products.

Task: Let $\mathcal{S} \subset \mathbb{R}^{n\times n}$ be a class of structured matrices. Recover $A \in \mathcal{S}$ given access to black-box matrix-vector products:

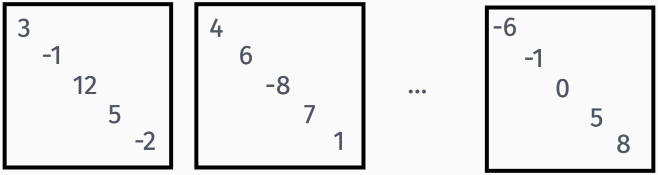
$$\mathrm{A}\mathrm{x} \qquad\qquad\qquad \mathrm{A}^T\mathrm{x}$$

Example classes: Low-rank, banded, Toeplitz, sparse, etc.

MATRIX RECOVERY

Task: Let $\mathcal{S} \subset \mathbb{R}^{n\times n}$ be a class of structured matrices. Recover $A \in \mathcal{S}$ given access to $\mathbf{A}\mathbf{x}_1, \mathbf{A}\mathbf{x}_2, \ldots, \mathbf{A}\mathbf{x}_q$ or $\mathbf{A}^T\mathbf{x}_1, \mathbf{A}^T\mathbf{x}_2, \ldots, \mathbf{A}^T\mathbf{x}_q$ for adaptively chosen "query vectors" $\mathbf{x}_1, \ldots, \mathbf{x}_q$.

Quiz: How many matrix-vector products are needed if $\mathcal{S}$ is the class of diagonal matrices?



MATRIX RECOVERY

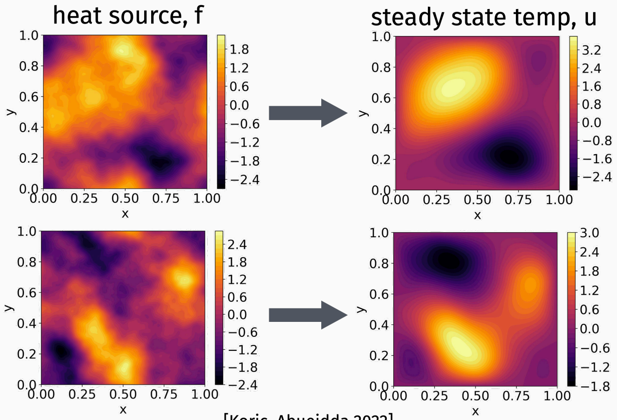
Halikias and Townsend, 2024:

Structure	# of matvecs
Diagonal	1
Toeplitz	2
Tridiagonal	3
k-banded	k
rank k	k
k-sparse rows	k
k-sparse columns	k
$\vdots$	$\vdots$

Intuition: Richer classes (e.g., with more parameters) require more matrix-vector products to recover.

MOTIVATION

Physical processes often map a function f to a function μ. I.e., implement some operator $\mathcal{L}(f) \rightarrow \mu$.

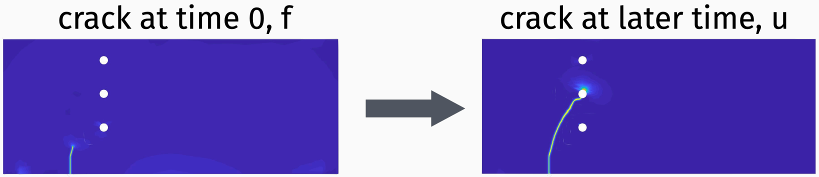


[Koric, Abueidda 2022]

Operator learning: Learn mapping from input-output pairs.

MOTIVATION

Physical processes often map a function f to a function μ. I.e., implement some operator $\mathcal{L}(f) \rightarrow \mu$.



[Goswami, Anitescu, Rabczuk 2019]

Operator learning: Learn mapping from input-output pairs. Concretely, given pairs $(f_1, u_1), \ldots, (f_q, u_q)$, the goal is to learn $\mathcal{L}$.

Central task is Scientific Machine Learning (SciML). Often $\mathcal{L}$ is parameterized by a neural network (e.g., as in DeepONet)

OPERATOR LEARNING

Two main reasons for interest in SciML:

1. **Constructing efficient surrogate models.** Given f, we can solve for u using simulation, a PDE solver, etc. but doing so is expensive. Goal is to learn a representation of $\mathcal{L}$ that is cheaper to apply to future values of f.
2. **Learning physics.** We do not understand the mapping from f to u, but can obtain pairs via physical experimentation.

In both of these settings, we typically have <u>freedom</u> in how each f_i is chosen. This differs from a typically ML setup, where inputs are drawn randomly from a distribution.

OPERATOR LEARNING

In the important special case when $\mathcal{L}$ is linear, operator learning corresponds to matrix learning (after discretization).

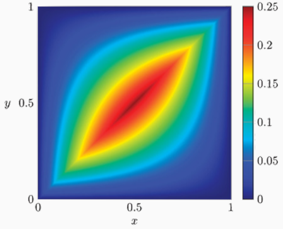
Input:

$$\mathbf{u}_1 = \mathbf{L}\mathbf{f}_1 \qquad \mathbf{u}_2 = \mathbf{L}\mathbf{f}_2 \qquad \ldots \qquad \mathbf{u}_q = \mathbf{L}\mathbf{f}_q,$$

where $\mathbf{f}_i, \mathbf{u}_i \in \mathbb{R}^n$, $\mathbf{L} \in \mathbb{R}^{n \times n}$.

Goal is to learn the matrix L.

Can only hope to do so efficiently (i.e., with $< n$ queries) if L has some structure:



MATRIX RECOVERY VS. MATRIX APPROXIMATION

Let $\mathcal{S} \subset \mathbb{R}^{n \times n}$ be a class of structured matrices.

Matrix recovery: Given access to black-box matrix-vector products with A, recover the matrix if $\mathsf{A} \in \mathcal{S}$.

MATRIX RECOVERY VS. MATRIX APPROXIMATION

Let $\mathcal{S} \subset \mathbb{R}^{n\times n}$ be a class of structured matrices.

~~**Matrix recovery:** Given access to black-box matrix-vector products, recover $\mathbf{A} \in \mathcal{S}$.~~

- Optimal or near optimal methods known for many problems.

Matrix approximation: For tolerance parameter $\epsilon > 0$, find near-optimal approximation $\tilde{\mathbf{B}} \in \mathcal{S}$ satisfying:

$$\|\mathbf{A} - \tilde{\mathbf{B}}\|_F \leq (1+\epsilon) \min_{\mathbf{B}\in\mathcal{S}} \|\mathbf{A} - \mathbf{B}\|_F.$$

- More relevant in actual applications.
- Harder problem. Fewer results.

MATRIX APPROXIMATION

Matrix approximation: For tolerance parameter $\epsilon > 0$, find near-optimal approximation $\tilde{B} \in \mathcal{S}$ satisfying:

$$\|A - \tilde{B}\|_F \leq (1+\epsilon) \min_{B \in \mathcal{S}} \|A - B\|_F.$$

$$\begin{bmatrix} \mathbf{3} & .1 & .1 & .1 & .1 \\ .1 & \mathbf{-2} & .1 & .1 & .1 \\ .1 & .1 & \mathbf{4} & .1 & .1 \\ .1 & .1 & .1 & \mathbf{6} & .1 \\ .1 & .1 & .1 & .1 & \mathbf{-1} \end{bmatrix}$$

target matrix A

$$\begin{bmatrix} \mathbf{3} & & & & \\ & \mathbf{-2} & & & \\ & & \mathbf{4} & & \\ & & & \mathbf{6} & \\ & & & & \mathbf{-1} \end{bmatrix}$$

optimal diagonal approximation B*

$$\begin{bmatrix} 0 & .1 & .1 & .1 & .1 \\ .1 & 0 & .1 & .1 & .1 \\ .1 & .1 & 0 & .1 & .1 \\ .1 & .1 & .1 & 0 & .1 \\ .1 & .1 & .1 & .1 & 0 \end{bmatrix}$$

error of optimal approximation

$$\begin{bmatrix} \mathbf{2.8} & & & & \\ & \mathbf{-2.1} & & & \\ & & \mathbf{4} & & \\ & & & \mathbf{6.1} & \\ & & & & \mathbf{-.9} \end{bmatrix}$$

near-optimal diagonal approximation $\tilde{B}$

$$\begin{bmatrix} .2 & .1 & .1 & .1 & .1 \\ .1 & .1 & .1 & .1 & .1 \\ .1 & .1 & 0 & .1 & .1 \\ .1 & .1 & .1 & .1 & .1 \\ .1 & .1 & .1 & .1 & .1 \end{bmatrix}$$

error of near-optimal approximation

OTHER MOTIVATIONS

Applications beyond operator learning:

- Diagonal approximation to Hessian used, e.g., for approximate second order optimization (AdaGrad, ADAM optimizer, etc.). Matvecs with AutoDiff.
- Low-rank approximation used to compress matrices throughout computational science, data science, machine learning, etc.
- Toeplitz approximation used to approximation nearly shift-invariance covariance matrices.

In many of these applications, we only have access to matrix-vector products with $\mathbf{A}$, or organizing our operations into matrix-vector products makes sense computationally.

DIFFERENCE BETWEEN RECOVERY AND APPROXIMATION

Approximation often requires very different algorithms!

$$\begin{bmatrix} 3 & & & & \\ & -2 & & & \\ & & 4 & & \\ & & & 6 & \\ & & & & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ -2 \\ 4 \\ 6 \\ -1 \end{bmatrix}$$

recovery

$$\begin{bmatrix} 3 & .1 & .1 & .1 & .1 \\ .1 & -2 & .1 & .1 & .1 \\ .1 & .1 & 4 & .1 & .1 \\ .1 & .1 & .1 & 6 & .1 \\ .1 & .1 & .1 & .1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3.4 \\ -1.6 \\ 4.4 \\ 6.4 \\ -.6 \end{bmatrix}$$

approximation

Goal is to ensure:

$$\|\mathbf{A} - \tilde{\mathbf{B}}\|_F \leq (1+\epsilon) \min_{\mathbf{B} \in \mathcal{S}} \|\mathbf{A} - \mathbf{B}\|_F \approx .1 \cdot n.$$

Error of naive algorithm:

$$\lesssim \sqrt{.1^2 \cdot n^2 + n \cdot (.1 \cdot n)^2} \approx .1 \cdot n^{1.5}.$$

BETTER APPROACH

Pick random sign vector $\mathbf{r} \in \{-1, 1\}^n$. Return $\mathbf{r} \circ (\mathbf{A}\mathbf{r})$.

$$\begin{bmatrix} 3 & .1 & .1 & .1 & .1 \\ .1 & -2 & .1 & .1 & .1 \\ .1 & .1 & 4 & .1 & .1 \\ .1 & .1 & .1 & 6 & .1 \\ .1 & .1 & .1 & .1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3.4 \\ -1.6 \\ 4.4 \\ 6.4 \\ -.6 \end{bmatrix}$$

deterministic

$$\begin{bmatrix} 3 & .1 & .1 & .1 & .1 \\ .1 & -2 & .1 & .1 & .1 \\ .1 & .1 & 4 & .1 & .1 \\ .1 & .1 & .1 & 6 & .1 \\ .1 & .1 & .1 & .1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 2.8 \\ 2 \\ 3.8 \\ -6 \\ 1 \end{bmatrix}$$

random

Error of randomized algorithm:

$$\approx \sqrt{.1^2 \cdot n^2 + n \cdot (.1 \cdot \sqrt{n})^2} \approx .2 \cdot n.$$

Can improve error by repeating and averaging.

OPTIMAL DIAGONAL APPROXIMATION

Theorem (Dharangutte, Musco, 2023)

Let $\mathbf{r}_1, \ldots, \mathbf{r}_q$ *be random sign vectors and let:*

$$\tilde{\mathbf{B}} = \text{diag}\left(\frac{1}{q}\sum_{i=1}^{q} \mathbf{r}_i \circ (\mathbf{A}\mathbf{r}_\mathbf{i})\right)$$

If $q = O\left(\frac{1}{\epsilon}\right)$, *then with high probability,*

$$\|\mathbf{A} - \tilde{\mathbf{B}}\|_F \leq (1+\epsilon) \min_{\textit{diagonal } \mathbf{B}} \|\mathbf{A} - \mathbf{B}\|_F.$$

Computing $\tilde{\mathbf{B}}$ *requires* $O\left(\frac{1}{\epsilon}\right)$ *matrix-vector products with* $\mathbf{A}$.

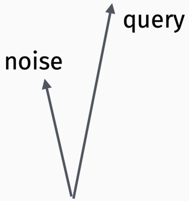
This method is called Hutchinson's estimator. It is possible to show that the result is tight: no algorithm can get away with $q \leq \frac{.5}{\epsilon}$ matrix-vector products [Amsel et al. 2024].

OPTIMAL DIAGONAL APPROXIMATION

Lesson: Need query vectors that both extract information from "signal" in A, but have small inner product with "noise".

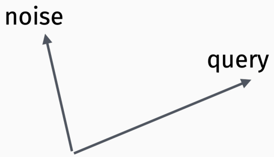
$$\begin{bmatrix} \mathbf{3} & .1 & .1 & .1 & .1 \\ .1 & \mathbf{-2} & .1 & .1 & .1 \\ .1 & .1 & \mathbf{4} & .1 & .1 \\ .1 & .1 & .1 & \mathbf{6} & .1 \\ .1 & .1 & .1 & .1 & \mathbf{-1} \end{bmatrix} \quad \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

bad



$$\begin{bmatrix} \mathbf{3} & .1 & .1 & .1 & .1 \\ .1 & \mathbf{-2} & .1 & .1 & .1 \\ .1 & .1 & \mathbf{4} & .1 & .1 \\ .1 & .1 & .1 & \mathbf{6} & .1 \\ .1 & .1 & .1 & .1 & \mathbf{-1} \end{bmatrix} \quad \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \\ -1 \end{bmatrix}$$

good



Randomness in queries is usually essential!

WHAT IS KNOWN?

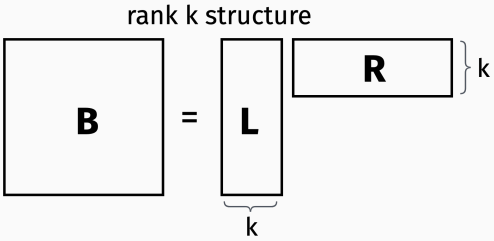
Best known matrix-vector query complexity for recovery vs. $(1+\epsilon)$ approximation.

Structure	# for recovery	# for approx.
Diagonal	1	$O(1/\epsilon)$
Toeplitz	2	$O(\log n/\epsilon)$
Tridiagonal	3	$O(1/\epsilon)$
k-banded	k	$O(k/\epsilon)$
rank k	k	$O(k/\epsilon^{1/3})$
k-sparse rows	k	$O(k/\epsilon)$
k-sparse columns	k	$O(k/\epsilon)$
k-sized linear family	$O(\sqrt{nk})$	??
$\vdots$	$\vdots$	$\vdots$

Tons of interesting open questions here.

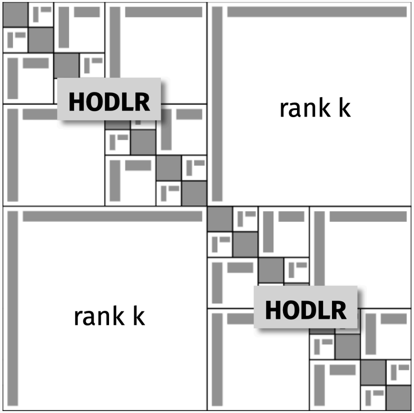
HIERARCHICAL LOW-RANK APPROXIMATION

One of the most important structures in SciML / scientific computing applications is hierarchical low-rank structure.



HIERARCHICAL LOW-RANK APPROXIMATION

One of the most important structures in SciML / scientific computing applications is hierarchical low-rank structure.



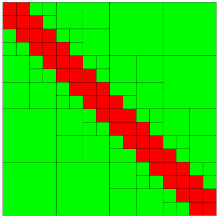
Classic example: Hierarchical off-diagonal low-rank (HODLR).

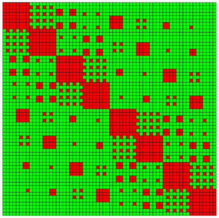
HODLR matrices

Properties of HODLR matrices:

- $O(nk\log(n/k))$ space to store.
- Matrix-vector multiplication in $O(nk\log(n/k))$ time.
- Linear system solving in $O(nk^3\log(n/k))$ time.

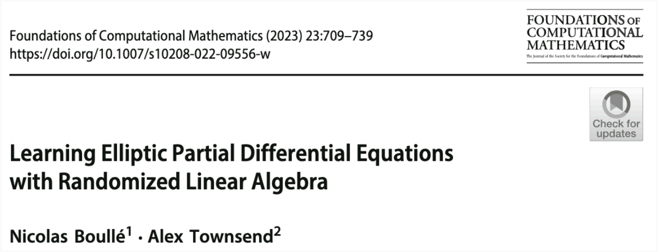
Many variants exist. Examples include e.g. Hierarchical Semi-separable (HSS) matrices, variants tailored to higher dimensional problems, different splits, etc.





Applications of HODLR matrices:

- Underly ubiquitous algorithms for structured matrices like the Fast Multipole Method. Near-optiomal HOLDR approximation yields even faster solvers.
- At the core of recent progress on operator learning: can be used to approximate the solution operator of elliptic PDEs (2024 SIAM Linear Algebra Best Paper Prize winner).



Foundations of Computational Mathematics (2023) 23:709–739
https://doi.org/10.1007/s10208-022-09556-w

FOUNDATIONS OF COMPUTATIONAL MATHEMATICS
The Journal of the Society for the Foundations of Computational Mathematics

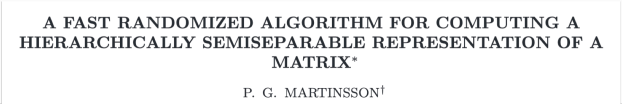
Check for updates

Learning Elliptic Partial Differential Equations with Randomized Linear Algebra

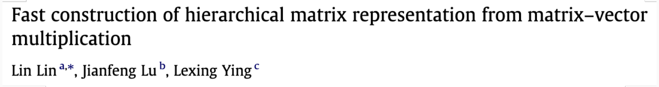
Nicolas Boullé[1] · Alex Townsend[2]

HODLR MATRICES

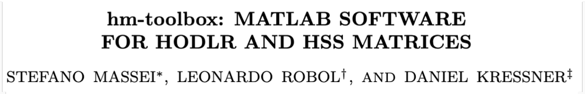
Lots of prior work on HODLR matrices:


A FAST RANDOMIZED ALGORITHM FOR COMPUTING A HIERARCHICALLY SEMISEPARABLE REPRESENTATION OF A MATRIX*

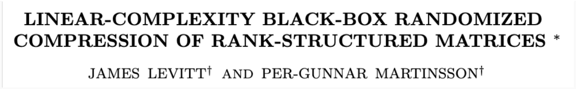
P. G. MARTINSSON†


Fast construction of hierarchical matrix representation from matrix–vector multiplication

Lin Lin [a,*], Jianfeng Lu [b], Lexing Ying [c]


hm-toolbox: MATLAB SOFTWARE FOR HODLR AND HSS MATRICES

STEFANO MASSEI*, LEONARDO ROBOL†, AND DANIEL KRESSNER‡


LINEAR-COMPLEXITY BLACK-BOX RANDOMIZED COMPRESSION OF RANK-STRUCTURED MATRICES *

JAMES LEVITT† AND PER-GUNNAR MARTINSSON†

Several algorithms that solve the recovery problem with $O(k \log n)$ matrix-vector products.

No prior methods for the approximation problem.

MAIN RESULT

Theorem (Chen, Duman Keles, Halikias, Musco, Musco, Persson, to appear at SODA 2025)

There is an algorithm that, based on $O(k \log^4(n/k)/\epsilon^3)$ *black-box matrix-vector products with* $\mathbf{A} \in \mathbb{R}^{n\times n}$*, computes a rank-*$k$ *HODLR matrix* $\tilde{\mathbf{B}}$ *satisfying:*

$$\|\mathbf{A} - \tilde{\mathbf{B}}\|_F \leq (1+\epsilon) \min_{\textit{HODLR } \mathbf{B}} \|\mathbf{A} - \mathbf{B}\|_F.$$

For an n^c approximation for any constant c, complexity can be improved to $O(k \log(n/k))$ matvecs to obtain an n^c approximation for any constant c.

Resolves main open question of [Boullé, Townsend, FoCM 2022].

APPROACH

Three major ingredients:

1. **Randomized low-rank approximation**. Specifically, the Generalized Nyström method.
2. The **peeling algorithm** of Lin, Lu, Ying.
3. **Perforated Gaussian sketches** based on CountSketch.

RANDOMIZED LOW-RANK APPROXIMATION

What if we want to solve the approximation problem for the simpler class of (non-hierarchical) low-rank matrices? I.e., find rank-k matrix $\tilde{\mathbf{B}}$ satisfying:

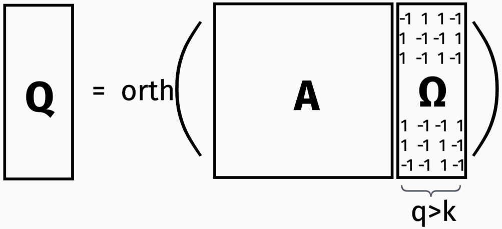
$$\|\mathbf{A} - \tilde{\mathbf{B}}\|_F \leq (1+\epsilon) \min_{\text{rank } k \ \mathbf{B}} \|\mathbf{A} - \mathbf{B}\|_F$$

This can be done using $O(k/\epsilon)$ matrix-vector products using the **Randomized SVD (RandSVD) method** [Sarlós, 2006, Halko, Martinsson, Tropp, 2011, many others].

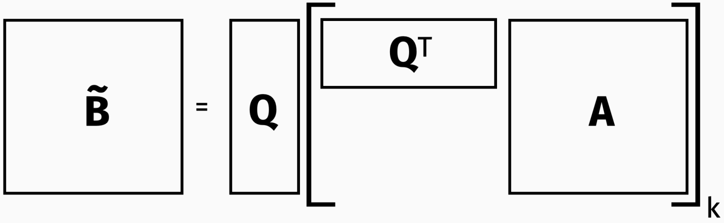
This is a sketching algorithm. We compute products with $\mathbf{A}$ and a set of random vectors.

RANDOMIZED SVD

1. Find approximation to top subspace.



2. Compute best rank k approximation in span of **Q**



RANDOMIZED LOW-RANK APPROXIMATION

Theorem (Halko, Martinsson, Tropp, 2011)

If the sketch $\boldsymbol{\Omega}$ is chosen to have i.i.d. Gaussian or Rademacher entries and $q = O(k/\epsilon)$ columns, then with high probability,

$$\|\mathbf{A} - \tilde{\mathbf{B}}\|_F \leq (1+\epsilon)\|\mathbf{A} - \mathbf{A}_k\|_F,$$

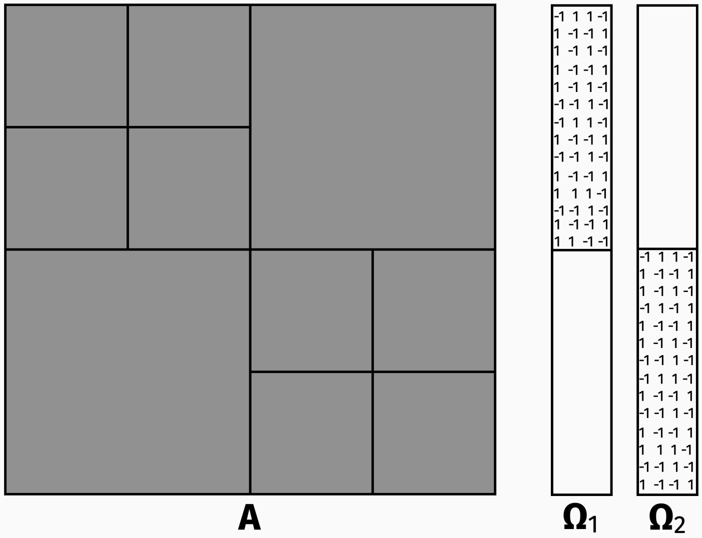
where $\mathbf{A}_k$ is the optimal rank k approximation to $\mathbf{A}$.

Overall, we requires $2q$ matrix vector products:

- q to compute $\mathbf{A}\boldsymbol{\Omega}$.
- q to compute $\mathbf{A}^T\mathbf{Q}$, where $\mathbf{Q} = \text{orth}(\mathbf{A}\boldsymbol{\Omega})$

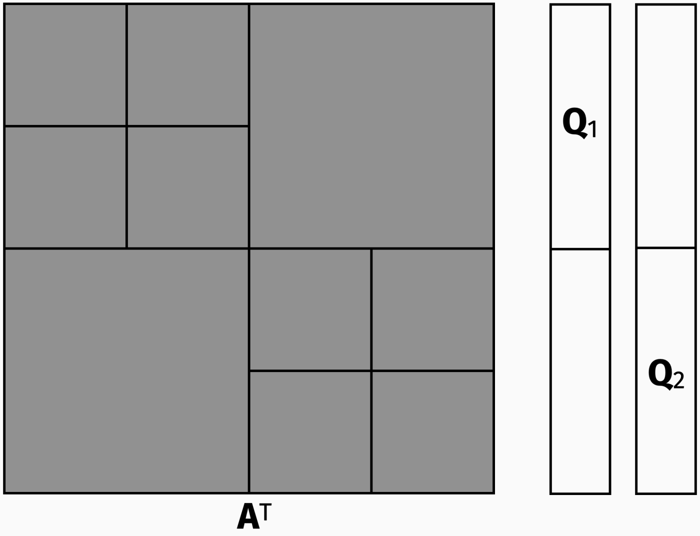
PEELING ALGORITHM

Second ingredient: Peeling Algorithm of Lin, Lu, Ying.

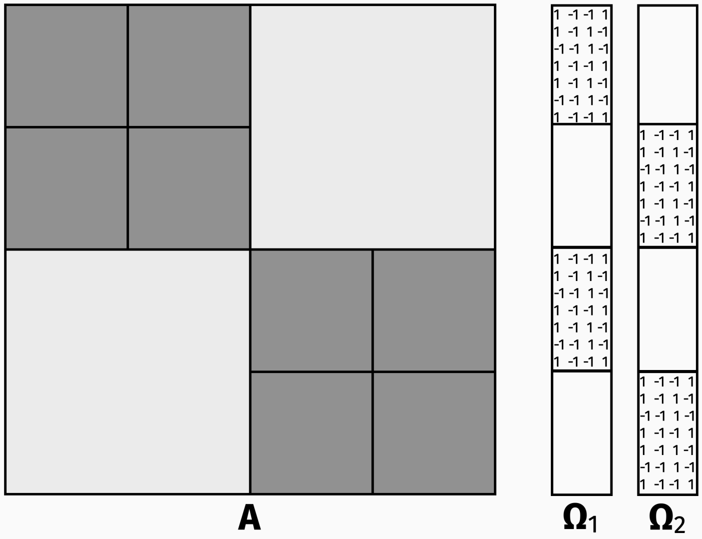


PEELING ALGORITHM

Second ingredient: Peeling Algorithm of Lin, Lu, Ying.

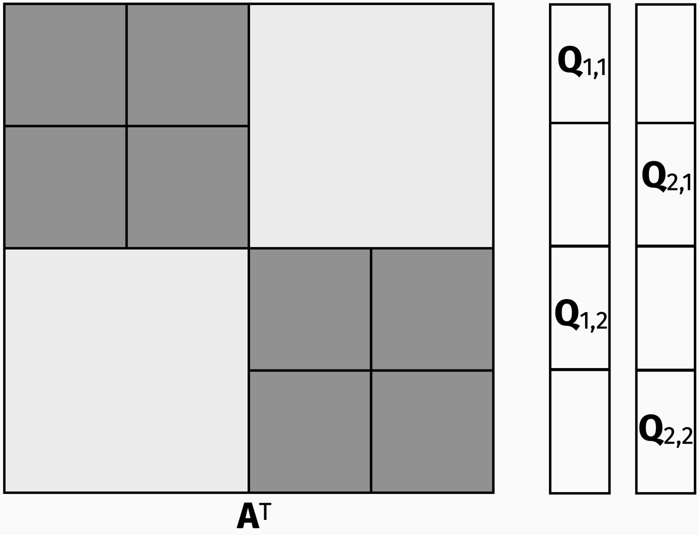


Second ingredient: Peeling Algorithm of Lin, Lu, Ying.

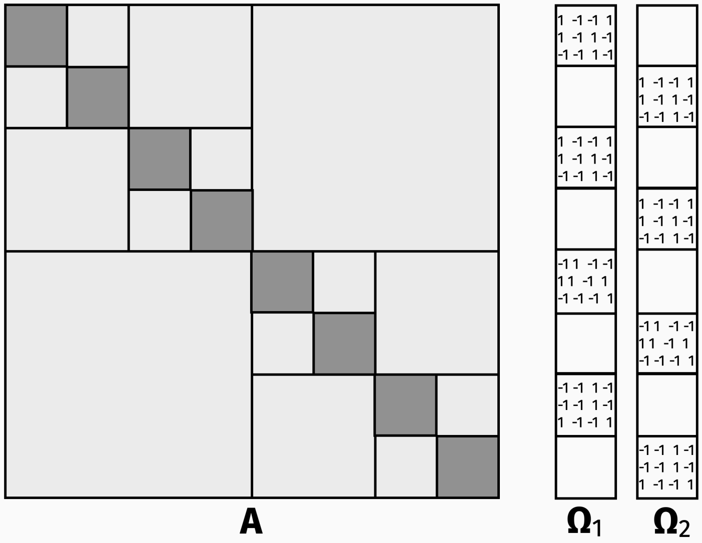


PEELING ALGORITHM

Second ingredient: Peeling Algorithm of Lin, Lu, Ying.



Second ingredient: Peeling Algorithm of Lin, Lu, Ying.



PEELING ALGORITHM

Second ingredient: Peeling Algorithm of Lin, Lu, Ying.

For a matrix with $L = \log(n/k)$ levels, require:

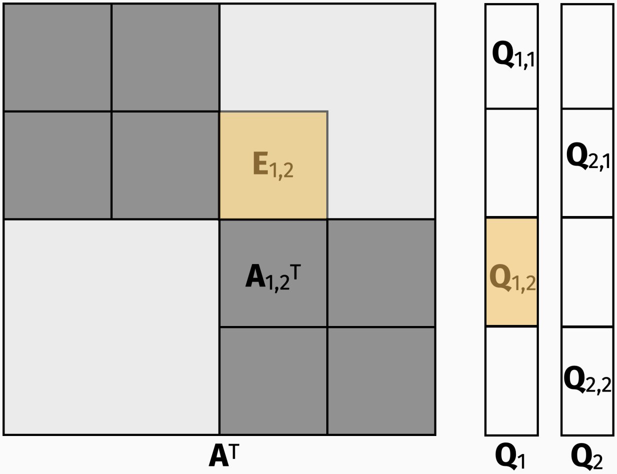
$$L \cdot 2 \cdot O(k) = O(k \log(n/k)) \text{ matrix-vector products.}$$

Works perfectly well for recovery. Fails in worse case for approximation.

Issue: Error can propagate across levels.

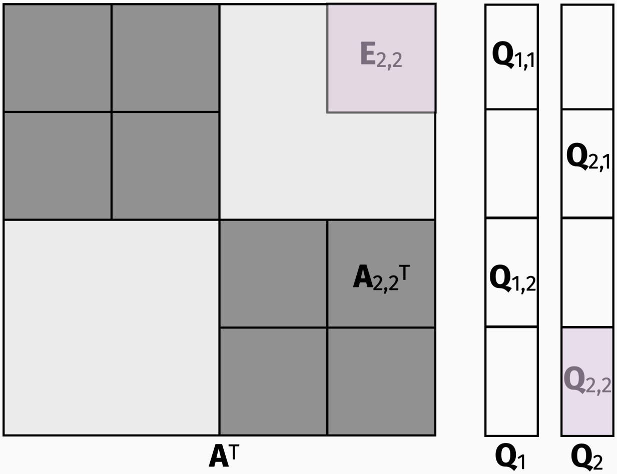
ERROR PROPAGATION

Issue arises in second step of Randomized SVD. Here, $Q_{1,2}$ aligns with top subspace of $A_{1,2}$. Could have noise $E_{1,2}$ with rows that point in the same direction.



ERROR PROPAGATION

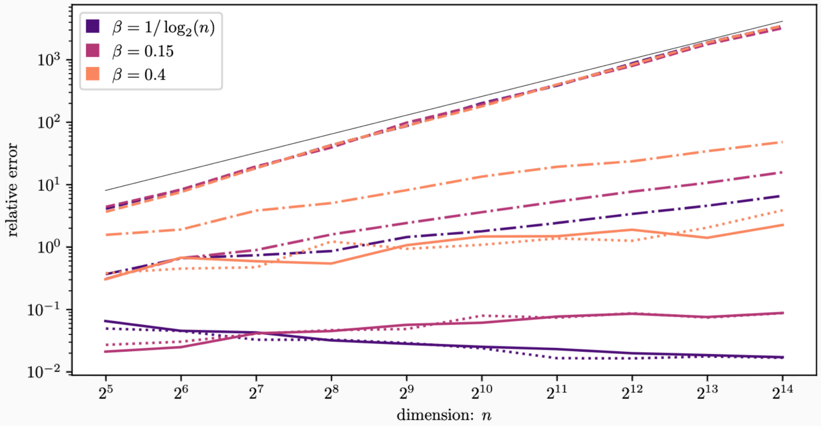
Here, $Q_{2,2}$ aligns with top subspace of $A_{22,2}$. Could have noise $E_{2,2}$ with rows that point in the same direction.



ERROR PROPAGATION

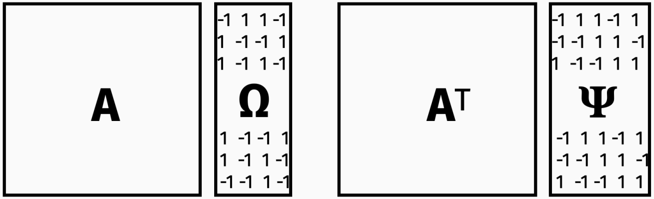
In the paper we give an example which shows that the standard Peeling + RandSVD method cannot give better than an $O(n)$ approximation.

GN1 (——), GN2 (······), RSVD1 (- - -), RSVD2 (–·–).



Generalized Nyström Method:

1. Sketch $\mathbf{A}$ on <u>both</u> left and right.



2. Let $\mathbf{Q} = \text{orth}(\mathbf{A}\boldsymbol{\Omega})$.
3. Return low-rank approximation $\tilde{\mathbf{B}} = \mathbf{Q}\left[(\boldsymbol{\Psi}^T\mathbf{Q})^+\boldsymbol{\Psi}\mathbf{A}\right]_k$.

In RandSVD, we project $\mathbf{A}$ onto span of $\mathbf{Q}$ and return best low-rank approximation.

Generalized Nyström can be viewed as performing an an <u>approximate projection</u> using sketching.

GENERALIZED NYSTRÖM ANALYSIS

Theorem (Clarkson, Woodruff, 2009)

If $\boldsymbol{\Omega}$ is chosen to have $O(k/\epsilon)$ columns and $\boldsymbol{\Psi}$ is chosen to have $O(k/\epsilon^3)$ columns, then with high probability,

$$\|\mathbf{A} - \tilde{\mathbf{B}}\|_F \leq (1+\epsilon)\|\mathbf{A} - \mathbf{A}_k\|_F,$$

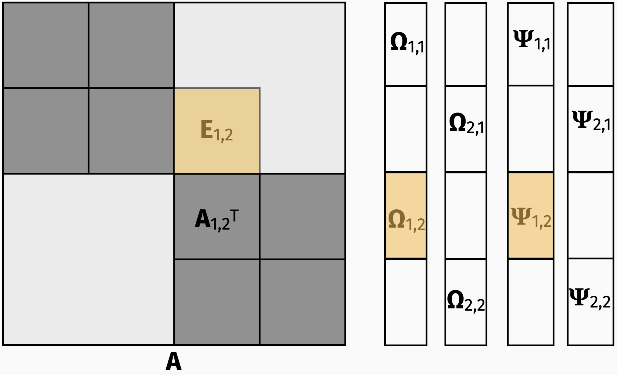
where $\mathbf{A}_k$ is the optimal rank k approximation to $\mathbf{A}$.

Cost: Higher matrix-vector product complexity than standard RandSVD.

Benefit: The method is far more robust to error in the matrix-vector products.

GENERALIZED NYSTRÖM ANALYSIS

Formalized in paper with a complete stability analysis of the Generalized Nyström method.



Takes advantage of fact that any noise in matrix vector products is of the form $\mathbf{ER}$, where $\mathbf{R}$ is a random matrix that is <u>independent</u> from $\mathbf{E}$.

MAIN RESULT

Theorem

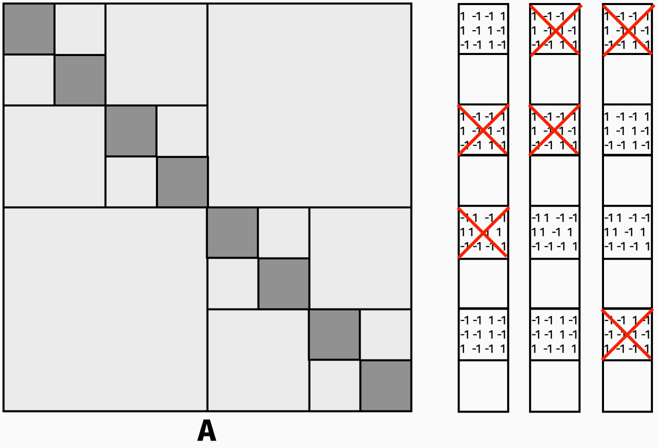
There is an algorithm that, based on $O(k \log^5(n/k)/\epsilon^4)$ *black-box matrix-vector products with* $\mathbf{A} \in \mathbb{R}^{n\times n}$*, computes a rank-*$k$ *HODLR matrix* $\tilde{\mathbf{B}}$ *satisfying:*

$$\|\mathbf{A} - \tilde{\mathbf{B}}\|_F \leq (1+\epsilon) \min_{HODLR\ \mathbf{B}} \|\mathbf{A} - \mathbf{B}\|_F.$$

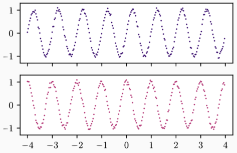
We are able to get an improvement to $O(k \log^4(n/k)/\epsilon^3)$ using a third ingredient: **randomized perforation**.

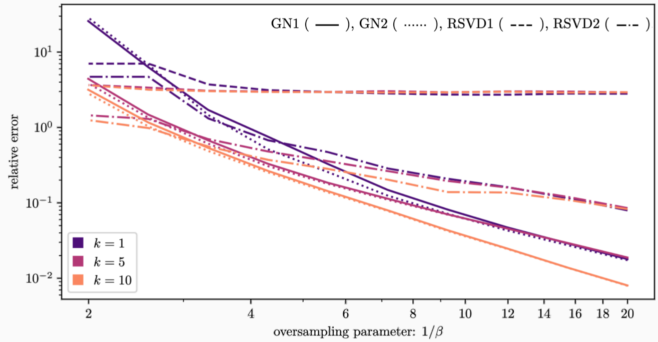
SKETCH PERFORATION

Basic idea: Randomly zero our portions of the peeling sketches. If we perforate to c fraction of original blocks, reduce expected noise by a factor of c.



Takeaway: Simply replacing RandSVD with Generalized Nyström immediately yields improved results.





CONCLUSION

Lots of open questions:

- Can we improve complexity to $O(k \log(n)/\epsilon)$? We prove a lower bound of $\Omega(k \log(n) + k/\epsilon)$ in the paper.
- Generalized Nyström often performs far better in practice than theoretical bounds suggest. Can we show that it requires $< O(k/\epsilon^3)$ matvecs for near-optimal k-rank approximation?
- Develop algorithms + analysis for related classes of hierarchical matrices like HSS matrices. Challenge: for some classes, obtained a near optimal approximation is hard <u>even if we know A explicitly</u>.
- Understand the gap between the approximation + recovery problems for general matrix family $\mathcal{S}$?

QUESTIONS?