Near-optimal hierarchical matrix approximation from matrix-vector products

Tyler Chen

June 10, 2024

`chen.pw/slides`

About this project

This is part of a broad research program on understanding what can be learned about a matrix from a small number of matrix-vector products.[1]

Collaboration between folks from NLA and TCS:

- Noah Amsel (NYU)
- Feyza Duman Keles (NYU)
- Diana Halikias (Cornell)
- Cameron Musco (UMass)
- Christopher Musco (NYU)
- David Persson (EPFL→NYU)

I'm particularly interested in feedback from this community about what kinds of theoretical analyses of algorithms for hierarchical matrices would be interesting.

[1] Halko, Martinsson, and Tropp 2011; Meyer, Musco, Musco, and Woodruff 2021; Halikias and Townsend 2023; Amsel et al. 2024, etc.

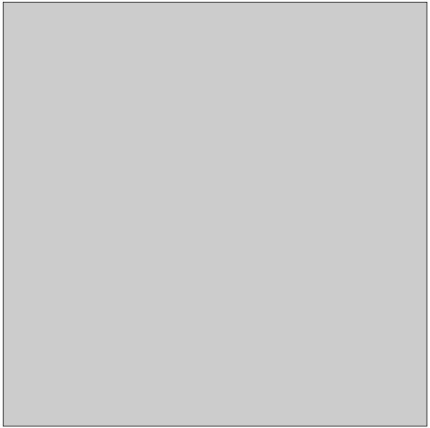
HODLR Matrices

Definition. Fix a rank parameter k. We say a $n \times n$ matrix $\mathbf{A}$ is HODLR(k) if $n \leq k$ or $\mathbf{A}$ can be partitioned into $(n/2) \times (n/2)$ blocks

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{bmatrix}$$

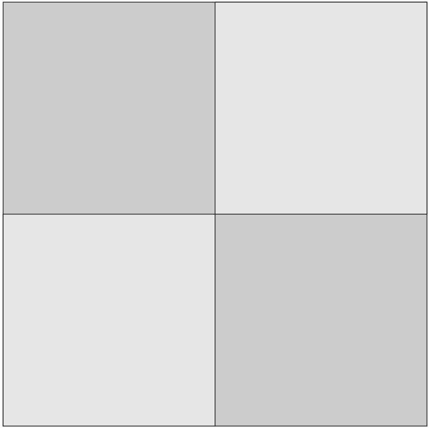
such that $\mathbf{A}_{1,2}$ and $\mathbf{A}_{2,1}$ are of rank at most k and $\mathbf{A}_{1,1}$ and $\mathbf{A}_{2,2}$ are each HODLR(k).

HODLR matrices



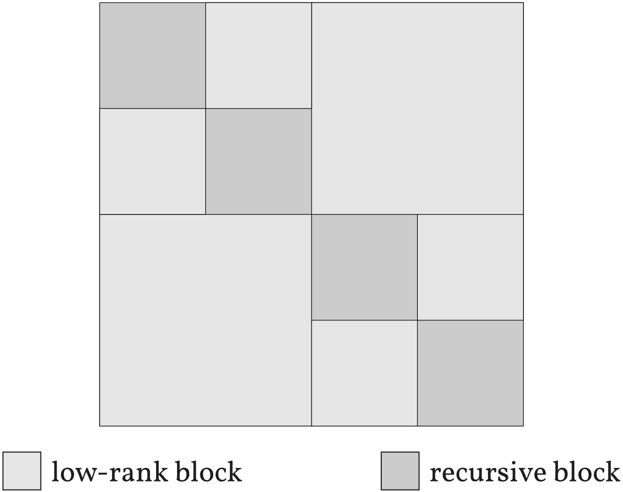
░ low-rank block ▒ recursive block

HODLR matrices

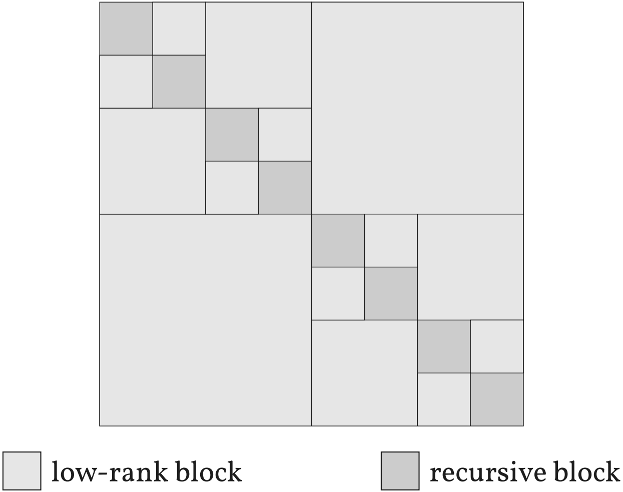


□ low-rank block ▨ recursive block

HODLR matrices



HODLR matrices



The HODLR approximation problem

Problem. Given an $n \times n$ matrix $\mathbf{A}$, accessible only by matrix-vector products, a rank parameter k, and an accuracy parameter ε, find a HODLR(k) matrix $\widetilde{\mathbf{A}}$ such that

$$\|\mathbf{A} - \widetilde{\mathbf{A}}\|_{\mathsf{F}} \le (1+\varepsilon) \min_{\mathbf{H} \in \mathrm{HODLR}(k)} \|\mathbf{A} - \mathbf{H}\|_{\mathsf{F}}.$$

The best HODLR approximation to $\mathbf{A}$ is obtained by applying a rank-k SVD to each low-rank block of $\mathbf{A}$.

- This is too expensive in the matrix-vector product model (n products)

In the special case that $\mathbf{A} \in \mathrm{HODLR}(k)$, then we require $\widetilde{\mathbf{A}} = \mathbf{A}$ (regardless of ε).

- There are several matvec algorithms for this setting[2]

[2] Lin, Lu, and Ying 2011; Martinsson 2016; Levitt and Martinsson 2022; Halikias and Townsend 2023.

The HODLR approximation problem

Problem. Given an $n \times n$ matrix $\mathbf{A}$, accessible only by matrix-vector products, a rank parameter k, and an accuracy parameter ε, find a HODLR(k) matrix $\widetilde{\mathbf{A}}$ such that

$$\|\mathbf{A} - \widetilde{\mathbf{A}}\|_{\mathsf{F}} \le (1+\varepsilon) \min_{\mathbf{H} \in \mathrm{HODLR}(k)} \|\mathbf{A} - \mathbf{H}\|_{\mathsf{F}}.$$

The best HODLR approximation to $\mathbf{A}$ is obtained by applying a rank-k SVD to each low-rank block of $\mathbf{A}$.

- This is too expensive in the matrix-vector product model (n products)

In the special case that $\mathbf{A} \in \mathrm{HODLR}(k)$, then we require $\widetilde{\mathbf{A}} = \mathbf{A}$ (regardless of ε).

- There are several matvec algorithms for this setting[2]

[2] Lin, Lu, and Ying 2011; Martinsson 2016; Levitt and Martinsson 2022; Halikias and Townsend 2023.

The HODLR approximation problem

Problem. Given an $n \times n$ matrix $\mathbf{A}$, accessible only by matrix-vector products, a rank parameter k, and an accuracy parameter ε, find a HODLR(k) matrix $\widetilde{\mathbf{A}}$ such that

$$\|\mathbf{A} - \widetilde{\mathbf{A}}\|_\mathsf{F} \leq (1+\varepsilon) \min_{\mathbf{H} \in \mathrm{HODLR}(k)} \|\mathbf{A} - \mathbf{H}\|_\mathsf{F}.$$

The best HODLR approximation to $\mathbf{A}$ is obtained by applying a rank-k SVD to each low-rank block of $\mathbf{A}$.

- This is too expensive in the matrix-vector product model (n products)

In the special case that $\mathbf{A} \in \mathrm{HODLR}(k)$, then we require $\widetilde{\mathbf{A}} = \mathbf{A}$ (regardless of ε).

- There are several matvec algorithms for this setting[2]

[2] Lin, Lu, and Ying 2011; Martinsson 2016; Levitt and Martinsson 2022; Halikias and Townsend 2023.

Learning low-rank matrices from matrix-vector products

The Randomized SVD (RSVD) is a well-known algorithm for obtaining a low-rank approximation to a matrix $\mathbf{B}$:

1. Sample Gaussian matrix $\mathbf{\Omega}$
2. Form $\mathbf{Q} = \operatorname{orth}(\mathbf{B\Omega})$
3. Compute $\mathbf{X} = \mathbf{Q}^\top \mathbf{B}$
4. Output $\mathbf{Q}[\![\mathbf{X}]\!]_k$

Theorem. If $\mathbf{\Omega}$ has $\sim k/\varepsilon$ columns, then

$$\|\mathbf{B} - \mathbf{Q}[\![\mathbf{X}]\!]_k\|_{\mathsf{F}} \le (1+\varepsilon) \min_{\operatorname{rank}(\mathbf{X}) \le k} \|\mathbf{B} - \mathbf{X}\|_{\mathsf{F}}.$$

Corollary. If $\mathbf{B}$ is rank-k, then $\mathbf{Q}[\![\mathbf{X}]\!]_k = \mathbf{B}$ (with probability one).

Peeling: an algorithm for the recovery problem[3]

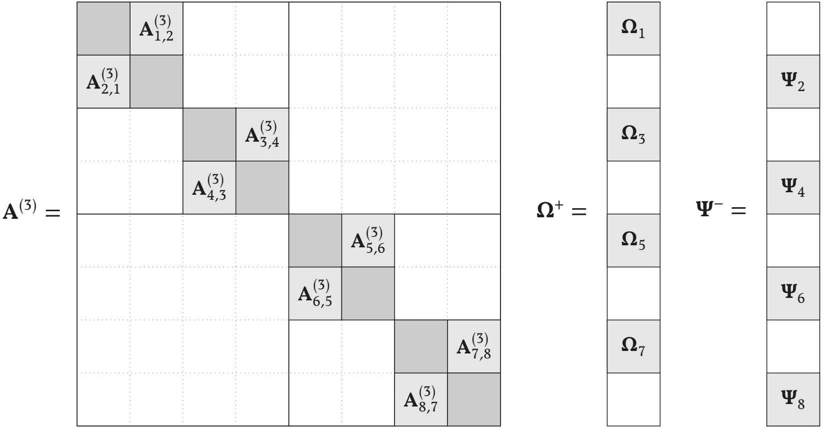
The algorithm works from the top layer down.

At each level, we simultaneosly apply the RSVD to the low-rank off-diagonal blocks.

We then “peel” off these blocks before proceeding to the next level

[3] Lin, Lu, and Ying 2011; Martinsson 2016.

Peeling: an algorithm for the recovery problem



Peeling: an algorithm for the recovery problem

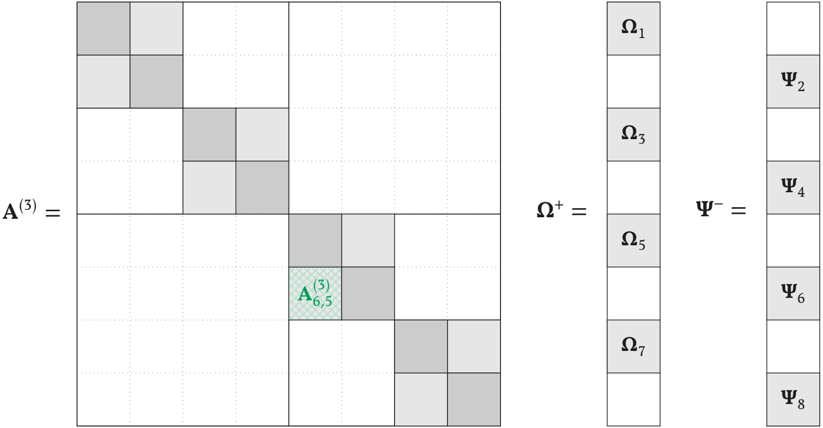
At each level we use $\sim k$ matrix-vector products with $\mathbf{A}$ and $\mathbf{A}^\top$.

There are $\log_2(n/k) \leq \log_2(n)$ levels until the blocks are of size k

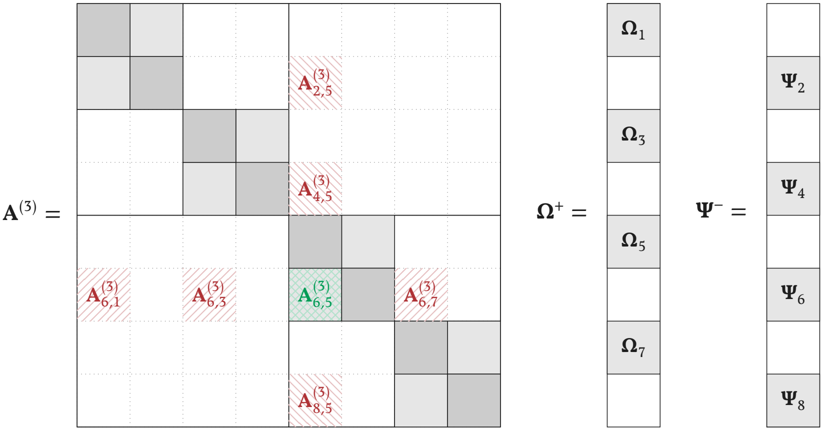
- then we can directly recover them at once with k products

Theorem. We can recover a HODLR matrix using $O\big(k \log_2(n)\big)$ matvecs.

Does peeling work on non-HODLR matrices?

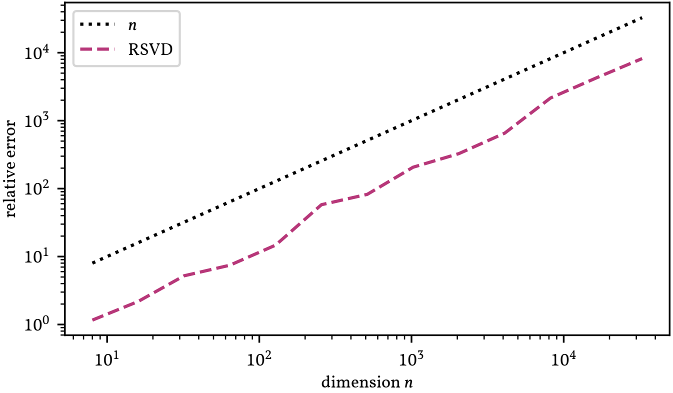


Does peeling work on non-HODLR matrices?



Does peeling work on non-HODLR matrices?

If all the error at a level can propagate to the next level, then the total error doubles at each level. Exponential blow-up in the number of levels (polynomial in n)!



What's going on? An illustration.

Suppose $\mathbf{X}$ and $\mathbf{Y}$ are rank k and $\mathbf{Y}$ is way bigger than $\mathbf{X}$. Consider

$$\mathbf{A} = \left[\begin{array}{cc:cc} \mathbf{0} & \mathbf{X} & \mathbf{Y} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \\ \hdashline \mathbf{Y} & \mathbf{X} & \mathbf{0} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \end{array}\right].$$

When we recover the low-rank blocks at the first level we will essentially get

$$\begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} \end{bmatrix}.$$

What’s going on? An illustration.

Next we subtract off these approximations:

$$\left[\begin{array}{cc:cc} \mathbf{0} & \mathbf{X} & \mathbf{Y} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \\ \hdashline \mathbf{Y} & \mathbf{X} & \mathbf{0} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \end{array}\right] - \left[\begin{array}{cc:cc} \mathbf{0} & \mathbf{0} & \mathbf{Y} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hdashline \mathbf{Y} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] = \left[\begin{array}{cc:cc} \mathbf{0} & \mathbf{X} & \mathbf{0} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{X} & \mathbf{0} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \end{array}\right].$$

What's going on? An illustration.

Now we sketch to learn the subspaces at the next level:

$$\left[\begin{array}{cc:cc} \mathbf{0} & \mathbf{X} & \mathbf{0} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{X} & \mathbf{0} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \end{array}\right]\left[\begin{array}{c} \boldsymbol{\Omega}_1^\dagger \\ \mathbf{0} \\ \hdashline \boldsymbol{\Omega}_3^\dagger \\ \mathbf{0} \end{array}\right] = \left[\begin{array}{c} \mathbf{0} \\ \mathbf{X}(\boldsymbol{\Omega}_1^\dagger + \boldsymbol{\Omega}_3^\dagger) \\ \hdashline \mathbf{0} \\ \mathbf{X}(\boldsymbol{\Omega}_1^\dagger + \boldsymbol{\Omega}_3^\dagger) \end{array}\right].$$

We then compute $\mathbf{Q} = \operatorname{orth}(\mathbf{X}(\boldsymbol{\Omega}_1^\dagger + \boldsymbol{\Omega}_3^\dagger))$ and get the correct range for $\mathbf{X}$

What's going on? An illustration.

However, we run into problems at the projection stage:

$$\left[\begin{array}{cc:cc} \mathbf{0} & \mathbf{Q}^\mathsf{T} & \mathbf{0} & \mathbf{Q}^\mathsf{T} \end{array}\right] \left[\begin{array}{cc:cc} \mathbf{0} & \mathbf{X} & \mathbf{0} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{X} & \mathbf{0} & \mathbf{X} \\ \mathbf{X} & \mathbf{0} & \mathbf{X} & \mathbf{0} \end{array}\right] = \left[\begin{array}{cc:cc} 2\mathbf{Q}^\mathsf{T}\mathbf{X} & \mathbf{0} & 2\mathbf{Q}^\mathsf{T}\mathbf{X} & \mathbf{0} \end{array}\right].$$

So our approximation to the off-diagonal blocks at this level is completely wrong… We get $2\mathbf{Q}\mathbf{Q}^\mathsf{T}\mathbf{X} = 2\mathbf{X}$ instead of $\mathbf{X}$.

All of the error from the first level propagated to the second level!

A perturbation bound for the RSVD

We prove a perturbation bound for the RSVD. This is likely of independent interest.

Theorem. Let $\mathbf{Q} = \operatorname{orth}(\mathbf{B}\boldsymbol{\Omega} + \mathbf{E}_1)$ and $\mathbf{X} = \mathbf{Q}^\top\mathbf{B} + \mathbf{E}_2$. Then

$$\|\mathbf{B} - \mathbf{Q}[\![\mathbf{Q}^\top\mathbf{B} + \mathbf{E}_2]\!]_k\|_{\mathsf{F}} \le \underbrace{\|\mathbf{E}_1\boldsymbol{\Omega}_{\mathrm{top}}^\dagger\|_{\mathsf{F}} + 2\|\mathbf{E}_2\|_{\mathsf{F}}}_{\text{perturbations}} + \underbrace{\left(\|\boldsymbol{\Sigma}_{\mathrm{bot}}\|_{\mathsf{F}}^2 + \|\boldsymbol{\Sigma}_{\mathrm{bot}}\boldsymbol{\Omega}_{\mathrm{bot}}\boldsymbol{\Omega}_{\mathrm{top}}^\dagger\|_{\mathsf{F}}^2\right)^{1/2}}_{\text{classical RSVD bound}}.$$

Takeaway: The pseudoinverse will help damp the perturbation $\mathbf{E}_1$, but (unsurprisingly) all of the perturbation $\mathbf{E}_2$ can propagate.

Generalized Nyström[4]

The RSVD tries to compute $\mathbf{Q}^\top \mathbf{B}$ directly; this is the solution to:

$$\min_{\mathbf{X}} \|\mathbf{A} - \mathbf{Q}\mathbf{X}\|_{\mathsf{F}}.$$

Instead, we can solve a sketched problem:

$$\min_{\mathbf{X}} \|\mathbf{\Psi}^\top \mathbf{A} - \mathbf{\Psi}^\top \mathbf{Q}\mathbf{X}\|_{\mathsf{F}}.$$

This means $\mathbf{X} = (\mathbf{\Psi}^\top \mathbf{Q})^\dagger \mathbf{\Psi}^\top \mathbf{A}$.

Observation. By adding columns to $\mathbf{\Psi}$, we can damp errors in the product $\mathbf{\Psi}^\top \mathbf{A}$.

The algorithm is also non-adaptive (we can do products with $\mathbf{\Psi}$ in advance)

[4] Clarkson and Woodruff 2009; Tropp, Yurtsever, Udell, and Cevher 2017; Nakatsukasa 2020.

Generalized Nyström[4]

The RSVD tries to compute $\mathbf{Q}^\top \mathbf{B}$ directly; this is the solution to:

$$\min_{\mathbf{X}} \|\mathbf{A} - \mathbf{Q}\mathbf{X}\|_{\mathsf{F}}.$$

Instead, we can solve a sketched problem:

$$\min_{\mathbf{X}} \|\mathbf{\Psi}^\top \mathbf{A} - \mathbf{\Psi}^\top \mathbf{Q}\mathbf{X}\|_{\mathsf{F}}.$$

This means $\mathbf{X} = (\mathbf{\Psi}^\top \mathbf{Q})^\dagger \mathbf{\Psi}^\top \mathbf{A}$.

Observation. By adding columns to $\mathbf{\Psi}$, we can damp errors in the product $\mathbf{\Psi}^\top \mathbf{A}$.

The algorithm is also non-adaptive (we can do products with $\mathbf{\Psi}$ in advance)

[4] Clarkson and Woodruff 2009; Tropp, Yurtsever, Udell, and Cevher 2017; Nakatsukasa 2020.

Generalized Nyström[4]

The RSVD tries to compute $\mathbf{Q}^\top\mathbf{B}$ directly; this is the solution to:

$$\min_{\mathbf{X}} \|\mathbf{A} - \mathbf{Q}\mathbf{X}\|_{\mathsf{F}}.$$

Instead, we can solve a sketched problem:

$$\min_{\mathbf{X}} \|\boldsymbol{\Psi}^\top\mathbf{A} - \boldsymbol{\Psi}^\top\mathbf{Q}\mathbf{X}\|_{\mathsf{F}}.$$

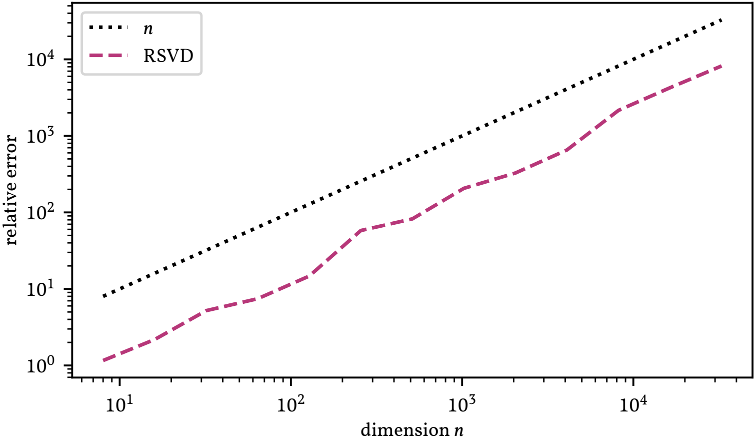
This means $\mathbf{X} = (\boldsymbol{\Psi}^\top\mathbf{Q})^\dagger\boldsymbol{\Psi}^\top\mathbf{A}$.

Observation. By adding columns to $\boldsymbol{\Psi}$, we can damp errors in the product $\boldsymbol{\Psi}^\top\mathbf{A}$.

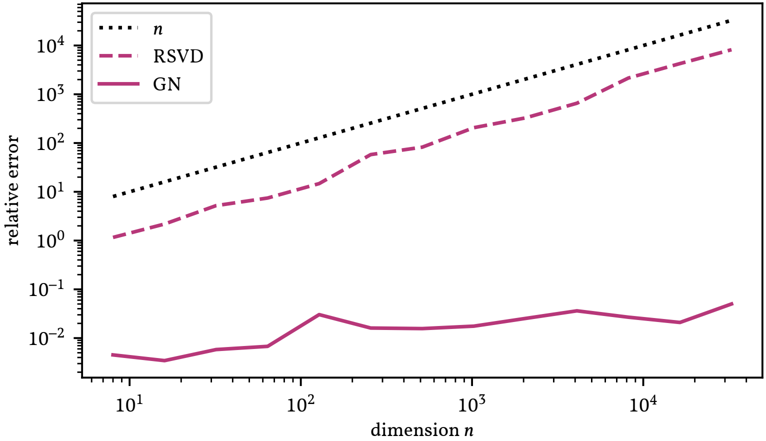
The algorithm is also non-adaptive (we can do products with $\boldsymbol{\Psi}$ in advance)

[4] Clarkson and Woodruff 2009; Tropp, Yurtsever, Udell, and Cevher 2017; Nakatsukasa 2020.

Back to the hard instance



Back to the hard instance

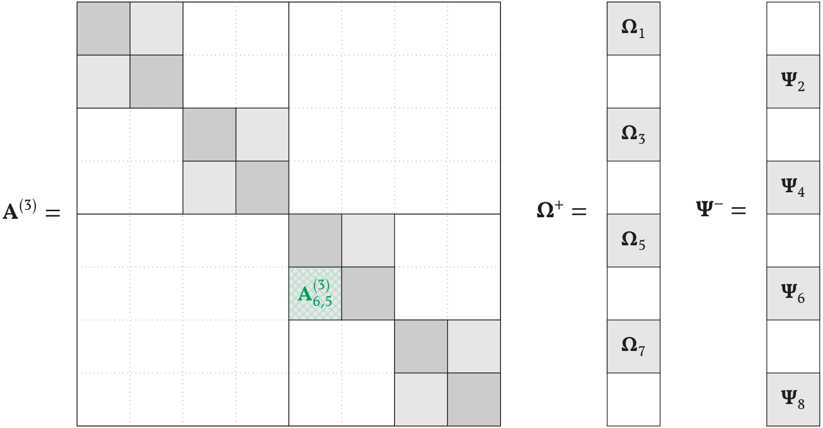


Another approach: perforated sketches

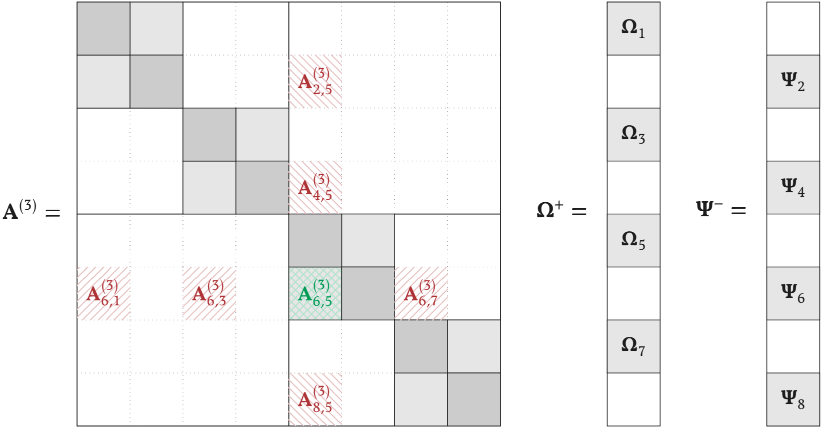
Because of the structure of peeling, the error happens when blocks of our sketch hit the error from our approximation of low-rank blocks at previous levels.

What if we just reduce how often this happens?

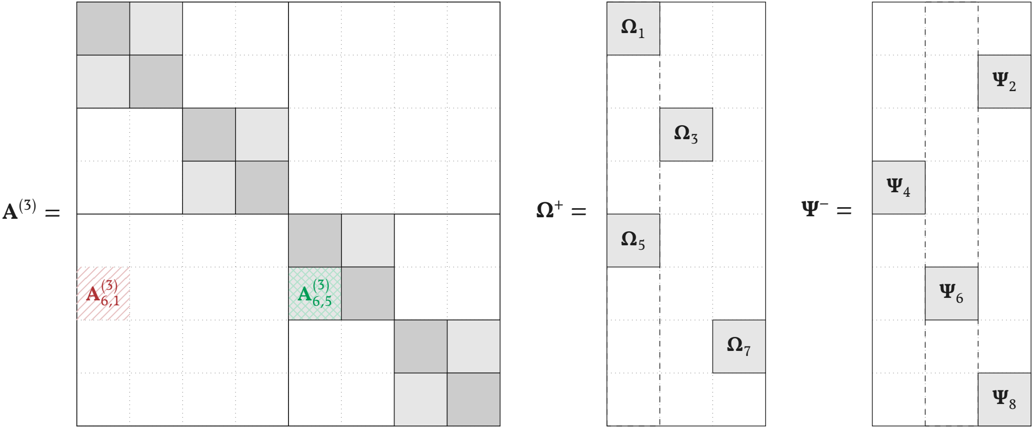
Perforated Block CountSketch



Perforated Block CountSketch



Perforated Block CountSketch



Main result

Theorem. There exist matvec algorithms which use $O\big(k \log(n) \cdot \mathrm{poly}(1/\beta)\big)$ products with $\mathbf{A}$ to obtain a HODLR(k) matrix $\widetilde{\mathbf{A}}$ satisfying[5]

$$\|\mathbf{A} - \widetilde{\mathbf{A}}\|_\mathsf{F} \le (1+\beta)^{\log_2(n)} \min_{\mathbf{H} \in \mathrm{HODLR}(k)} \|\mathbf{A} - \mathbf{H}\|_\mathsf{F}.$$

Corollary. $(1+\varepsilon)$-optimal approximation with $O\big(k \log(n) \cdot \mathrm{poly}(\log(n)/\varepsilon)\big)$ matvecs

Corollary. $n^{0.01}$-optimal approximation with $O\big(k \log(n)\big)$ matvecs

[5] The poly($\cdot$) factors are essentially matching the best known bounds for Generalized Nyström (although these bounds are thought to be loose).

Main result

Theorem. There exist matvec algorithms which use $O\big(k\log(n)\cdot\mathrm{poly}(1/\beta)\big)$ products with $\mathbf{A}$ to obtain a HODLR(k) matrix $\widetilde{\mathbf{A}}$ satisfying[5]

$$\|\mathbf{A}-\widetilde{\mathbf{A}}\|_{\mathsf{F}} \le (1+\beta)^{\log_2(n)} \min_{\mathbf{H}\in\mathrm{HODLR}(k)} \|\mathbf{A}-\mathbf{H}\|_{\mathsf{F}}.$$

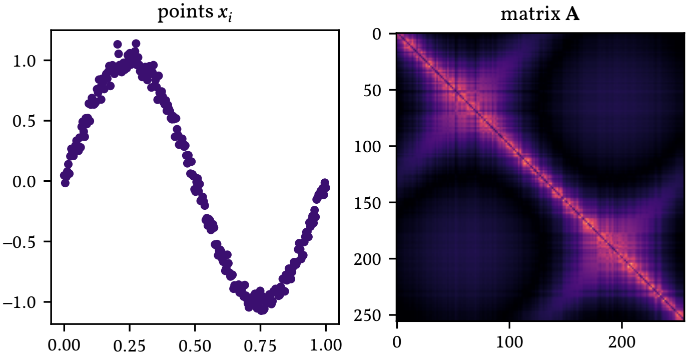
Corollary. $(1+\varepsilon)$-optimal approximation with $O\big(k\log(n)\cdot\mathrm{poly}(\log(n)/\varepsilon)\big)$ matvecs

Corollary. $n^{0.01}$-optimal approximation with $O\big(k\log(n)\big)$ matvecs

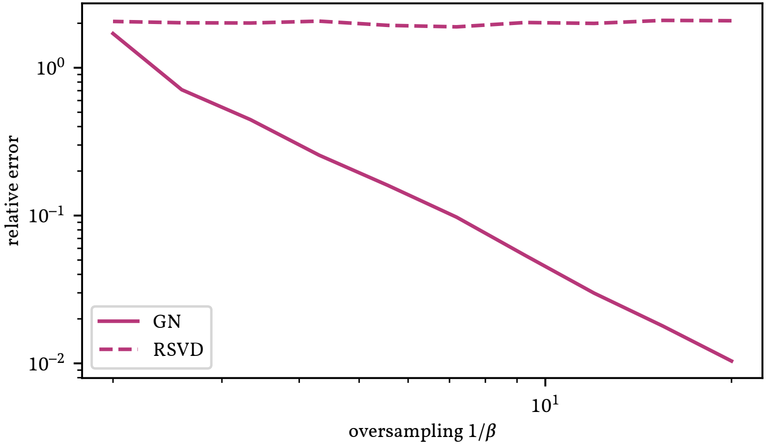
[5] The poly($\cdot$) factors are essentially matching the best known bounds for Generalized Nyström (although these bounds are thought to be loose).

Another experiment

Given points $x_i \in \mathbb{R}^2$, define $[\mathbf{A}]_{i,j} = -\log(\|x_i - x_j\|)$



Another experiment



Lower bounds?

The matrix-vector query model often lets us prove lower-bounds against any matvec algorithm for a given task; i.e. study the complexity of a task.

This provides a very different approach for understanding how good algorithms are (compared to classical numerical analysis).

Theorem. There is a constant $C > 0$ such that for any k, n, ε, there exists a matrix $\mathbf{A}$ such that getting a $(1+\varepsilon)$-optimal HODLR approximation requires at least $C\big(k \log_2(n/k) + k/\varepsilon\big)$ matvecs.

What's next?

- Correct $\log(n)$ and ε rates for the algorithms we study?
  - Limited by the best known bounds for Generalized Nyström: $O(k/\varepsilon^3)$
- True stability analysis (e.g. for floating point arithmetic)
- Adaptive algorithms
- Other hierarchical classes?
  - for $\mathcal{H}^1$ the generalization is probably straightforward
  - for nested families (e.g. HSS), it's not even clear how to get the best approximation, even if you know the matrix
- Better understanding of (non-adaptive) low-rank approximation

Questions for you

- What are important theoretical questions in this area?
- Does it matter if algorithms are provably correct if they work well in practice?

Generalized Nyström (perturbation) analysis

Extend $\mathbf{Q}$ to an orthogonal matrix $[\mathbf{Q}\ \widehat{\mathbf{Q}}]$, and write $\boldsymbol{\Psi}_1 = \boldsymbol{\Psi}^\top\mathbf{Q}$ and $\boldsymbol{\Psi}_2 = \boldsymbol{\Psi}^\top\widehat{\mathbf{Q}}$.

By orthogonal invariance, $\boldsymbol{\Psi}_1$ and $\boldsymbol{\Psi}_2$ are independent Gaussian matrices!

First observe:

$$\boldsymbol{\Psi}^\top\mathbf{B} = \boldsymbol{\Psi}^\top(\mathbf{Q}\mathbf{Q}^\top + \widehat{\mathbf{Q}}\widehat{\mathbf{Q}}^\top)\mathbf{B} = \boldsymbol{\Psi}_1\mathbf{Q}^\top\mathbf{B} + \boldsymbol{\Psi}_2\widehat{\mathbf{Q}}^\top\mathbf{B}.$$

Therefore:

$$\mathbf{X} = (\boldsymbol{\Psi}^\top\mathbf{Q})^\dagger(\boldsymbol{\Psi}^\top\mathbf{B}) = \boldsymbol{\Psi}_1^\dagger\boldsymbol{\Psi}_1\mathbf{Q}^\top\mathbf{B} + \boldsymbol{\Psi}_1^\dagger\boldsymbol{\Psi}_2\widehat{\mathbf{Q}}^\top\mathbf{B} = \mathbf{Q}^\top\mathbf{B} + \boldsymbol{\Psi}_1^\dagger\boldsymbol{\Psi}_2\widehat{\mathbf{Q}}^\top\mathbf{B}.$$

Adding more columns to $\boldsymbol{\Psi}$ (and hence $\boldsymbol{\Psi}_1$) reduces the error in the second term.

Generalized Nyström (perturbation) analysis

Extend $\mathbf{Q}$ to an orthogonal matrix $[\mathbf{Q}\;\widehat{\mathbf{Q}}]$, and write $\boldsymbol{\Psi}_1 = \boldsymbol{\Psi}^\top \mathbf{Q}$ and $\boldsymbol{\Psi}_2 = \boldsymbol{\Psi}^\top \widehat{\mathbf{Q}}$.

By orthogonal invariance, $\boldsymbol{\Psi}_1$ and $\boldsymbol{\Psi}_2$ are independent Gaussian matrices!

First observe:

$$\boldsymbol{\Psi}^\top \mathbf{B} + \mathbf{E} = \boldsymbol{\Psi}^\top (\mathbf{Q}\mathbf{Q}^\top + \widehat{\mathbf{Q}}\widehat{\mathbf{Q}}^\top)\mathbf{B} + \mathbf{E} = \boldsymbol{\Psi}_1 \mathbf{Q}^\top \mathbf{B} + \boldsymbol{\Psi}_2 \widehat{\mathbf{Q}}^\top \mathbf{B} + \mathbf{E}.$$

Therefore:

$$\mathbf{X} = (\boldsymbol{\Psi}^\top \mathbf{Q})^\dagger (\boldsymbol{\Psi}^\top \mathbf{B} + \mathbf{E}) = \boldsymbol{\Psi}_1^\dagger \boldsymbol{\Psi}_1 \mathbf{Q}^\top \mathbf{B} + \boldsymbol{\Psi}_1^\dagger \boldsymbol{\Psi}_2 \widehat{\mathbf{Q}}^\top \mathbf{B} + \boldsymbol{\Psi}_1^\dagger \mathbf{E} = \mathbf{Q}^\top \mathbf{B} + \boldsymbol{\Psi}_1^\dagger \boldsymbol{\Psi}_2 \widehat{\mathbf{Q}}^\top \mathbf{B} + \boldsymbol{\Psi}_1^\dagger \mathbf{E}.$$

Adding more columns to $\boldsymbol{\Psi}$ (and hence $\boldsymbol{\Psi}_1$) reduces the error in the second term.

References I

Amsel, Noah et al. (2024). *Fixed-sparsity matrix approximation from matrix-vector products*.

Clarkson, Kenneth L. and David P. Woodruff (May 2009). “Numerical linear algebra in the streaming model”. In: *Proceedings of the forty-first annual ACM symposium on Theory of computing*. STOC ’09. ACM.

Halikias, Diana and Alex Townsend (Sept. 2023). “Structured matrix recovery from matrix-vector products”. In: *Numerical Linear Algebra with Applications* 31.1.

Halko, N. , P. G. Martinsson, and J. A. Tropp (2011). “Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions”. In: *SIAM Rev.* 53.2, pp. 217–288.

Levitt, James and Per-Gunnar Martinsson (2022). *Randomized Compression of Rank-Structured Matrices Accelerated with Graph Coloring*.

Lin, Lin, Jianfeng Lu, and Lexing Ying (May 2011). “Fast construction of hierarchical matrix representation from matrix–vector multiplication”. In: *Journal of Computational Physics* 230.10, pp. 4071–4087.

Martinsson, Per-Gunnar (Jan. 2016). “Compressing Rank-Structured Matrices via Randomized Sampling”. In: *SIAM Journal on Scientific Computing* 38.4, A1959–A1986.

Meyer, Raphael A. et al. (Jan. 2021). “Hutch++: Optimal Stochastic Trace Estimation”. In: *Symposium on Simplicity in Algorithms (SOSA)*. Society for Industrial and Applied Mathematics, pp. 142–155.

Nakatsukasa, Yuji (2020). “Fast and stable randomized low-rank matrix approximation”. In: *ArXiv* abs/2009.11392.

Tropp, Joel A. et al. (Jan. 2017). “Practical Sketching Algorithms for Low-Rank Matrix Approximation”. In: *SIAM Journal on Matrix Analysis and Applications* 38.4, pp. 1454–1485.