Fast Algorithms for Rank-Structured Matrices

Tristan Goodwill and Evan Toler

September 23, 2020

Introduction

We focus on matrix operations exploiting *internal structure*.

- low (numerical) rank blocks can be *compressed*
- hierarchically off-diagonal low-rank (HODLR) matrices
- more generally, $\mathcal{H}$ and $\mathcal{H}^2$ classes of matrices

5.1 Inversion of a 2×2 Block Matrix

We first look at a building block for HODLR matrices:

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}. \tag{1}$$

Assume that off-diagonal blocks $\mathbf{A}_{12}$ and $\mathbf{A}_{21}$ have low *exact* rank for simplicity.

Lemma (5.1)

Let $\mathbf{A}$ *be as above in* (1)*. If* $\mathbf{A}$ *and* $\mathbf{A}_{22}$ *are both invertible, then the matrix* $\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}$ *is also invertible and*

$$\mathbf{A}^{-1} = \begin{pmatrix} \mathbf{X}_1 & -\mathbf{X}_1\mathbf{A}_{12}\mathbf{A}_{22}^{-1} \\ -\mathbf{A}_{22}^{-1}\mathbf{A}_{21}\mathbf{X}_1 & \mathbf{A}_{22}^{-1} + \mathbf{A}_{22}^{-1}\mathbf{A}_{21}\mathbf{X}_1\mathbf{A}_{12}\mathbf{A}_{22}^{-1} \end{pmatrix},$$

where

$$\mathbf{X}_1 = (\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21})^{-1}.$$

5.1 Inversion of a 2×2 Block Matrix

Lemma 5.1 gives an algorithm to compute $\mathbf{A}^{-1}$:

- Compute $\mathbf{X}_2 = \mathbf{A}_{22}^{-1}$.
- Compute $\mathbf{X}_1 = (\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{X}_2\mathbf{A}_{21})^{-1}$.
- Compute $\mathbf{A}^{-1} = \begin{pmatrix} \mathbf{X}_1 & -\mathbf{X}_1\mathbf{A}_{12}\mathbf{X}_2 \\ -\mathbf{X}_2\mathbf{A}_{21}\mathbf{X}_1 & \mathbf{X}_2 + \mathbf{X}_2\mathbf{A}_{21}\mathbf{X}_1\mathbf{A}_{12}\mathbf{X}_2 \end{pmatrix}$.

Proof of Lemma 5.1 is in the text: all linear algebra.

Using the low rank structure of $\mathbf{A}$ offers advantages. If all blocks are $N \times N$ matrices and $\mathbf{A}_{12}$ and $\mathbf{A}_{21}$ have rank k, then:

- matrix multiplication with $\mathbf{A}_{12}$ and $\mathbf{A}_{21}$ is inexpensive ($O(kN^2)$)
- cost is dominated by $\mathbf{X}_1$ and $\mathbf{X}_2$ inversions
- only need to directly invert two $N \times N$ matrices ($2 \times O(N^3)$) instead of one full $2N \times 2N$ matrix ($O((2N)^3) = 8 \times O(N^3)$)

5.2 HODLR Matrices

A HODLR matrix has the same block structure as Section 5.1 applied recursively.

Definition (5.2)

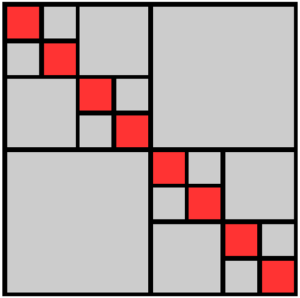
Let $\mathbf{A}$ be a matrix of size $N \times N$, and let k be an integer such that $k < N$. We then say that $\mathbf{A}$ is a hierarchically off-diagonal low-rank (HODLR) matrix with rank k if either of the following two conditions hold:

- $\mathbf{A}$ is itself of size at most $2k \times 2k$
- If $\mathbf{A}$ is partitioned into four equal-sized blocks,

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix},$$

  then $\mathbf{A}_{12}$ and $\mathbf{A}_{21}$ have rank at most k, and $\mathbf{A}_{11}$ and $\mathbf{A}_{22}$ are HODLR matrices of rank k.

5.2 HODLR Matrices



function f $= \texttt{HODLR\_matvec}(\mathbf{A}, \mathbf{q})$
if $\dim(\mathbf{A}) < 2k$ **then**
  Evaluate by brute force: $\mathbf{f} = \mathbf{A}\mathbf{q}$.
else
  Split $\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}$ and $\mathbf{q} = \begin{pmatrix} \mathbf{q}_1 \\ \mathbf{q}_2 \end{pmatrix}$.
  $\mathbf{f}_1 = \texttt{HODLR\_matvec}(\mathbf{A}_{11}, \mathbf{q}_1) + \mathbf{A}_{12}\mathbf{q}_2$.
  $\mathbf{f}_2 = \texttt{HODLR\_matvec}(\mathbf{A}_{22}, \mathbf{q}_2) + \mathbf{A}_{21}\mathbf{q}_1$.
  $\mathbf{f} = \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{pmatrix}$.
end if

If we have already factored off-diagonal blocks, this algorithm has complexity $O(kN \log N)$.

5.3 Inversion of Compressible Matrices

- Idea: Invert a HODLR matrix by recursively applying Lemma 5.1
- Issue: Is the lower-right block $\mathbf{X}_2 + \mathbf{X}_2\mathbf{A}_{21}\mathbf{X}_1\mathbf{A}_{12}\mathbf{X}_2$ HODLR with the *same* rank k of $\mathbf{A}_{12}$ and $\mathbf{A}_{21}$?
  - No, not in general. Adding $\underbrace{\mathbf{X}_2\mathbf{A}_{21}\mathbf{X}_1\mathbf{A}_{12}\mathbf{X}_2}_{\text{rank}=k}$ can increase the ranks of $\mathbf{X}_2$'s blocks by k.
  - But often it should still be compressible, if we want to preserve the physics of a PDE.
  - Combat the potential increase in rank by recompressing the off-diagonal blocks.

There is no guarantee that the inverse of a rank-k HODLR matrix is necessarily a HODLR matrix of rank k.

5.3 Inversion of Compressible Matrices

function $\mathbf{C} = \texttt{HODLR\_invert}(\mathbf{A})$
if $\dim(\mathbf{A}) < 2k$ **then**
  Invert by brute force: $\mathbf{C} = \mathbf{A}^{-1}$.
else
  Split $\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}$.
  $\mathbf{X}_{22} = \texttt{HODLR\_invert}(\mathbf{A}_{22})$.
  $\mathbf{X}_{11} = \texttt{HODLR\_invert}(\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{X}_{22}\mathbf{A}_{21})$.
  $\mathbf{C} = \begin{pmatrix} \mathbf{X}_{11} & -\mathbf{X}_{11}\mathbf{A}_{12}\mathbf{X}_{22} \\ -\mathbf{X}_{22}\mathbf{A}_{21}\mathbf{X}_{11} & \mathbf{X}_{22} + \mathbf{X}_{22}\mathbf{A}_{21}\mathbf{X}_{11}\mathbf{A}_{12}\mathbf{X}_{22} \end{pmatrix}$.
  Recompress the lower right block of $\mathbf{C}$.
end if

How can we *LU* factor a HODLR matrix $\mathbf{A}$?

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} \\ \mathbf{0} & \mathbf{U}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{11}\mathbf{U}_{11} & \mathbf{L}_{11}\mathbf{U}_{12} \\ \mathbf{L}_{21}\mathbf{U}_{11} & \mathbf{L}_{21}\mathbf{U}_{12} + \mathbf{L}_{22}\mathbf{U}_{22} \end{pmatrix}$$

We see from block matrix multiplication that we should first factorize

$$\mathbf{A}_{11} = \mathbf{L}_{11}\mathbf{U}_{11}.$$

Next, comparing block elements yields the expressions:

- $\mathbf{L}_{21} = \mathbf{A}_{21}\mathbf{U}_{11}^{-1}$
- $\mathbf{U}_{12} = \mathbf{L}_{11}^{-1}\mathbf{A}_{12}$.

What remains is to factor the Schur complement

$$\mathbf{L}_{22}\mathbf{U}_{22} = \mathbf{A}_{22} - \mathbf{L}_{21}\mathbf{U}_{12} = \mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}.$$

5.4 LU factorization and matrix-matrix multiplication

A recursive algorithm for HODLR LU:

- $[\mathbf{L}_{11}, \mathbf{U}_{11}] = \texttt{HODLR\_LU}(\mathbf{A}_{11})$
- $\mathbf{L}_{21} = \mathbf{A}_{21}\mathbf{U}_{11}^{-1}$
- $\mathbf{U}_{12} = \mathbf{L}_{11}^{-1}\mathbf{A}_{12}$
- $[\mathbf{L}_{22}, \mathbf{U}_{22}] = \texttt{HODLR\_LU}(\mathbf{A}_{22} - \mathbf{L}_{21}\mathbf{U}_{12})$

Some remarks:

- Exploit that $\mathbf{A}_{12}$ and $\mathbf{A}_{21}$ have rank k.
- Exploit that $\mathbf{U}_{11}$ and $\mathbf{L}_{11}$ are triangular.
- Recompress the Schur complement $\mathbf{A}_{22} - \mathbf{L}_{21}\mathbf{U}_{12}$ before recursion.
- Sequential structure: first factor $\mathbf{A}_{11}$, then the Schur complement.

5.4 LU factorization and matrix-matrix multiplication

Matrix-matrix multiplication for HODLR matrices A and B:

$$\begin{pmatrix} \mathbf{C}_{11} & \mathbf{C}_{12} \\ \mathbf{C}_{21} & \mathbf{C}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{pmatrix}$$
$$= \begin{pmatrix} \mathbf{A}_{11}\mathbf{B}_{11} + \mathbf{A}_{12}\mathbf{B}_{21} & \mathbf{A}_{11}\mathbf{B}_{12} + \mathbf{A}_{12}\mathbf{B}_{22} \\ \mathbf{A}_{21}\mathbf{B}_{11} + \mathbf{A}_{22}\mathbf{B}_{21} & \mathbf{A}_{21}\mathbf{B}_{12} + \mathbf{A}_{22}\mathbf{B}_{22} \end{pmatrix}.$$

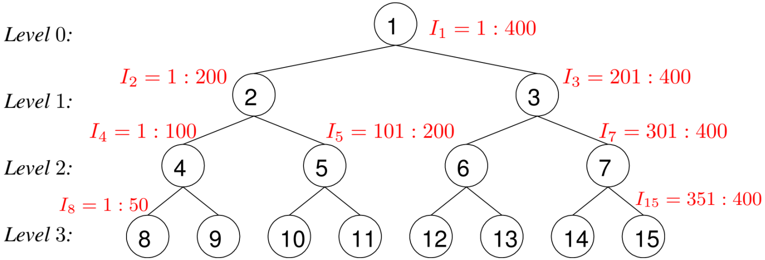
Observations:

- Off-diagonal blocks remain low rank (inherited from **A** and **B**).
- Diagonal blocks allow recursion through $\mathbf{A}_{11}\mathbf{B}_{11}$ and $\mathbf{A}_{22}\mathbf{B}_{22}$.
- We should recompress the diagonal blocks of **C** at each step.
- One $N \times N$ operation becomes two $N/2 \times N/2$ operations.

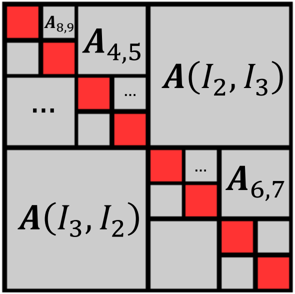
5.5 Hierarchical partitions of the index vector

We need an indexing system to allow us to refer to the hierarchical sub-blocks.

- Let $I_1 = [1, 2, \ldots, N]$. This is level 0 of our tree.
- Split I_1 into two siblings I_2 and I_3 such that $|I_2| \approx |I_3|$, $I_2 \cup I_3 = I_1$ and $I_2 \cap I_3 = \emptyset$. These form level 1.
- Keep splitting into siblings as above until we reach a level L where every vector is smaller than a threshold size bk.



5.5 Hierarchical partitions of the index vector



Definition (HODLR Matrices) (non-recursive)

A matrix $\mathbf{A}$ is said to be if HODLR if $\exists$ a k and an indexing system as above such that for every sibling pair $\{\alpha, \beta\}$ the off-diagonal block $\mathbf{A}(I_\alpha, I_\beta) = \mathbf{A}_{\alpha,\beta}$ is rank at most k. *i.e.* we can write

$$\begin{array}{cccc} \mathbf{A}_{\alpha,\beta} = & \mathbf{U}_\alpha & \tilde{\mathbf{A}}_{\alpha,\beta} & \mathbf{V}_\beta^*. \\ N_\alpha \times N_\beta & N_\alpha \times k & k \times k & k \times N_\beta \end{array}$$

We also note that the memory complexity to store an $N \times N$ HODLR matrix is

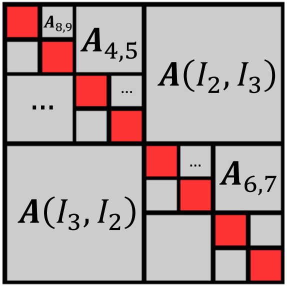
$$M = M_{diag} + M_{offdiag} \sim Nk + Nk \log(N/k) \sim Nk \log(N/k).$$

5.6 Nonrecursive formulas for HODLR matrix operations (MatVec)

An example of the use of our indexing:

function f $= \texttt{HODLR\_matvec}(\mathbf{A},\mathbf{q})$
$\mathbf{f} = \mathbf{0}$
for τ is a node in the tree **do**
  if τ is a leaf node **then**
    $\mathbf{f}(I_\tau) = \mathbf{f}(I_\tau) + \mathbf{A}(I_\tau, I_\tau)\mathbf{q}(I_\tau)$
  else
    Let $\{\alpha, \beta\}$ denote the children of τ.
    $\mathbf{f}(I_\alpha) = \mathbf{f}(I_\alpha) + \mathbf{U}_\alpha(\tilde{\mathbf{A}}_{\alpha,\beta}(\mathbf{V}_\beta^*\mathbf{q}(I_\beta)))$.
    $\mathbf{f}(I_\beta) = \mathbf{f}(I_\beta) + \mathbf{U}_\beta(\tilde{\mathbf{A}}_{\beta,\alpha}(\mathbf{V}_\alpha^*\mathbf{q}(I_\alpha)))$.
  end if
end for

Note that the tree can be traversed in any order.



Recall that
$\mathbf{A}_{\alpha,\beta} = \mathbf{U}_\alpha \tilde{\mathbf{A}}_{\alpha,\beta} \mathbf{V}_\beta^*$

5.6 Nonrecursive formulas for HODLR matrix operations (Inversion)

This time, we write

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{A}_{11}^{-1}\mathbf{A}_{12} \\ \mathbf{A}_{22}^{-1}\mathbf{A}_{21} & \mathbf{I} \end{pmatrix},$$

so that

$$\mathbf{A}^{-1} = \begin{pmatrix} \mathbf{I} & \mathbf{A}_{11}^{-1}\mathbf{A}_{12} \\ \mathbf{A}_{22}^{-1}\mathbf{A}_{21} & \mathbf{I} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{A}_{11}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{22}^{-1} \end{pmatrix}.$$

If we already know $\mathbf{C}_1 = \mathbf{A}_{11}^{-1}$ and $\mathbf{C}_2 = \mathbf{A}_{22}^{-1}$, then the left matrix is known. As the off diagonal blocks have rank at most k, we can factor it as

$$\begin{pmatrix} \mathbf{I} & \mathbf{C}_1\mathbf{A}_{12} \\ \mathbf{C}_2\mathbf{A}_{21} & \mathbf{I} \end{pmatrix} = \mathbf{I} + \mathbf{U}\mathbf{D}\mathbf{V}^*,$$

where D is a $2k \times 2k$ matrix. The Woodbury identity then tells us that

$$(\mathbf{I} + \mathbf{U}\mathbf{D}\mathbf{V}^*)^{-1} = \mathbf{I} - \mathbf{U}(\mathbf{D}^{-1} + \mathbf{V}^*\mathbf{U})^{-1}\mathbf{V}^*,$$

so we need only construct and invert the $2k \times 2k$ matrix $\mathbf{D}^{-1} + \mathbf{V}^*\mathbf{U}$.

5.6 Nonrecursive formulas for HODLR matrix operations (Inversion)

Applying the above formulas, we can make a new algorithm working up the tree.

function $\mathbf{C} = \texttt{HODLR\_invert}(\mathbf{A})$

for $\tau = N_{boxes} : (-1) : 1$ **do**

  if τ is a leaf node **then**

    Invert by brute force: $\mathbf{C}_\tau = (\mathbf{A}(I_\tau, I_\tau))^{-1}$

  else

    Let $\{\alpha, \beta\}$ denote the children of τ.

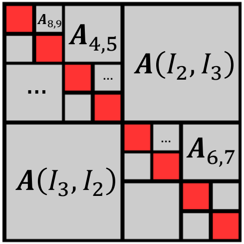
$$\mathbf{C}_\tau = \begin{pmatrix} \mathbf{I} & \mathbf{C}_\alpha \mathbf{A}_{\alpha,\beta} \\ \mathbf{C}_\beta \mathbf{A}_{\beta,\alpha} & \mathbf{I} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{C}_\alpha & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_\beta \end{pmatrix}$$

    Recompress $\mathbf{C}_\tau$ to combat potential increase in ranks of off-diagonal blocks.

  end if

end for

$\mathbf{C} = \mathbf{C}_1$



5.7 Extensions

Some considerations untouched so far:

- How to choose the tree for the index vector?
  - important for keeping off-diagonal blocks low-rank
  - could come from physical considerations if I indexes points in space
- What about integral equation solvers?
  - challenging to find the compressed representation
  - addressed later in the book (Ch. 17)
- What about nonuniform trees?
  - could appear in adaptive/local mesh refinement
  - tricky to maintain high efficiency
- Do we need to store $\mathbf{U}_\tau$ and $\mathbf{V}_\tau$ explicitly?
  - Often, no. We can "recycle" basis matrices and use recursion.
  - improves complexity from $O(N \log N)$ to $O(N)$
  - addressed later in the book (Ch. 13–16)

Summary

Definitions/Ideas

- HODLR matrix
- indexing tree

Algorithms for HODLR matrices

- Matrix-vector multiplication (recursive and non-recursive)
- Matrix inversion (recursive and non-recursive)
- LU factorization and matrix-matrix multiplication (recursive)