An introduction to fractional calculus

Fundamental ideas and numerics

Fabio Durastante
Università di Pisa
✉ fabio.durastante@unipi.it
fdurastante.github.io

October, 2022

Sylvester with quasiseparable matrices

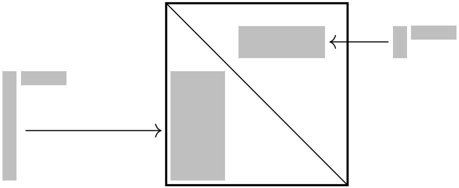
Let's start again from the problem we wanted to solve

$$AX + XB^T = C, \qquad A \in \mathbb{R}^{n\times n},\ \ B \in \mathbb{R}^{m\times m},\ \ X, C \in \mathbb{R}^{n\times m},$$

with A, B, and C **quasiseparable**

Quasiseparable matrix

A matrix A is *quasiseparable* of order k if the maximum of the ranks of all its submatrices contained in the strictly upper or lower part is less or equal than k.



💡 We have seen that A, B, and C quasiseparable $\Rightarrow$ X with decay of the singular values of off-diagonal blocks of C.

Sylvester with quasiseparable matrices

Theorem (Massei, Palitta, and Robol 2018, Theorem 2.12)

Let A, B be matrices of quasiseparable rank k_A and k_B respectively and such that $W(A) \subseteq E$ and $W(-B) \subseteq F$. Consider the Sylvester equation $AX + XB = C$, with C of quasiseparable rank k_C. Then a generic off-diagonal block Y of the solution X satisfies

$$\frac{\sigma_{1+k\ell}(Y)}{\sigma_1(Y)} \leq \mathcal{C}^2 \cdot Z_\ell(E, F), \qquad k := k_A + k_B + k_C.$$

Where $Z_\ell(E, F)$ is the solution of the **Zolotarev problem**

$$Z_\ell(E, F) \triangleq \inf_{r(x) \in \mathcal{R}_{\ell,\ell}} \frac{\max_{x \in E} |r(x)|}{\min_{y \in F} |r(y)|}, \qquad \ell \geq 1,$$

for $\mathcal{R}_{\ell,\ell}$ is the set of rational functions of degree at most (ℓ, ℓ), and $\mathcal{C}$ is the Crouzeix universal constant.

Sylvester with quasiseparable matrices

❓ Do the *decaying singular values* in the blocks implies the existence of a **quasiseparable approximant**?

Sylvester with quasiseparable matrices

❓ Do the *decaying singular values* in the blocks implies the existence of a **quasiseparable approximant**?

ϵ-quasiseparable matrices of rank k (ε-qsrank k)

We say that A has *ϵ-quasiseparable rank k* if, for every off-diagonal block Y, $\sigma_{k+1}(Y) \leq \epsilon$. If the property holds for the lower (respectively upper) offdiagonal blocks, we say that A has lower (respectively upper) ϵ-quasiseparable rank k.

Sylvester with quasiseparable matrices

❓ Do the *decaying singular values* in the blocks implies the existence of a **quasiseparable approximant**?

ϵ-quasiseparable matrices of rank k (ε-qsrank k)

We say that A has *ϵ-quasiseparable rank k* if, for every off-diagonal block Y, $\sigma_{k+1}(Y) \leq \epsilon$. If the property holds for the lower (respectively upper) offdiagonal blocks, we say that A has lower (respectively upper) ϵ-quasiseparable rank k.

Submatrices and off-diagonal blocks

If a matrix A has ϵ-quasiseparable rank k, then any of its principal submatrix A' has ϵ-quasiseparable rank k.

Any off-diagonal block Y of A' is also an off-diagonal block of $A \Rightarrow \sigma_{k+1}(Y) \leq \epsilon$.

Sylvester with quasiseparable matrices

❓ Do the *decaying singular values* in the blocks implies the existence of a **quasiseparable approximant**?

ϵ-quasiseparable matrices of rank k (ε-qsrank k)

We say that A has *ϵ-quasiseparable rank k* if, for every off-diagonal block Y, $\sigma_{k+1}(Y) \leq \epsilon$. If the property holds for the lower (respectively upper) offdiagonal blocks, we say that A has lower (respectively upper) ϵ-quasiseparable rank k.

Submatrices and off-diagonal blocks

If a matrix A has ϵ-quasiseparable rank k, then any of its principal submatrix A' has ϵ-quasiseparable rank k.

Any off-diagonal block Y of A' is also an off-diagonal block of $A \Rightarrow \sigma_{k+1}(Y) \leq \epsilon$.

For $\oplus$ the direct sum

Technical lemma

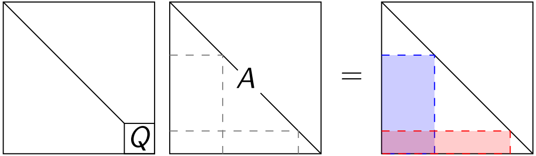
Let A be a matrix with ϵ-quasiseparable rank k, Q any $(k+1) \times (k+1)$ unitary matrix. Then, $(I_{n-k-1} \oplus Q)A$ also has ϵ-quasiseparable rank k.

Sylvester with quasiseparable matrices

❓ Do the *decaying singular values* in the blocks implies the existence of a **quasiseparable approximant**?

ϵ-quasiseparable matrices of rank k (ε-qsrank k)

We say that A has *ϵ-quasiseparable rank k* if, for every off-diagonal block Y, $\sigma_{k+1}(Y) \leq \epsilon$. If the property holds for the lower (respectively upper) offdiagonal blocks, we say that A has lower (respectively upper) ϵ-quasiseparable rank k.



Q acts on the tall block of A without changing its singular values, while the small one has small rank thanks to the small number of rows.

Sylvester with quasiseparable matrices

Theorem (Massei, Palitta, and Robol 2018, Theorem 2.16)

Let A be of ϵ-quasiseparable rank k, for $\epsilon > 0$. Then, there exists a matrix δA of norm bounded by $\|\delta A\|_2 \leq 2\sqrt{n} \cdot \epsilon$ so that $A + \delta A$ is k-quasiseparable.

Sylvester with quasiseparable matrices

Theorem (Massei, Palitta, and Robol 2018, Theorem 2.16)

Let A be of ϵ-quasiseparable rank k, for $\epsilon > 0$. Then, there exists a matrix δA of norm bounded by $\|\delta A\|_2 \leq 2\sqrt{n} \cdot \epsilon$ so that $A + \delta A$ is k-quasiseparable.

👁 *A matrix with ϵ-quasiseparable rank of k can be well-approximated by a matrix with exact quasiseparable rank k!*

Sylvester with quasiseparable matrices

Theorem (Massei, Palitta, and Robol 2018, Theorem 2.16)

Let A be of ϵ-quasiseparable rank k, for $\epsilon > 0$. Then, there exists a matrix δA of norm bounded by $\|\delta A\|_2 \leq 2\sqrt{n} \cdot \epsilon$ so that $A + \delta A$ is k-quasiseparable.

👁 *A matrix with ϵ-quasiseparable rank of k can be well-approximated by a matrix with exact quasiseparable rank k!*

👁 *If the* ***spectra*** *of A and $-B$ are* ***well-separated*** *in the Zolotarev sense, we can* ***preserve structure****!*

Sylvester with quasiseparable matrices

Theorem (Massei, Palitta, and Robol 2018, Theorem 2.16)

Let A be of ϵ-quasiseparable rank k, for $\epsilon > 0$. Then, there exists a matrix δA of norm bounded by $\|\delta A\|_2 \leq 2\sqrt{n} \cdot \epsilon$ so that $A + \delta A$ is k-quasiseparable.

👁 *A matrix with ϵ-quasiseparable rank of k can be well-approximated by a matrix with exact quasiseparable rank k!*

👁 *If the* ***spectra*** *of A and $-B$ are* ***well-separated*** *in the Zolotarev sense, we can* ***preserve structure****!*

❓ How can we operate efficiently with these matrix structures?

Sylvester with quasiseparable matrices

Theorem (Massei, Palitta, and Robol 2018, Theorem 2.16)

Let A be of ϵ-quasiseparable rank k, for $\epsilon > 0$. Then, there exists a matrix δA of norm bounded by $\|\delta A\|_2 \leq 2\sqrt{n} \cdot \epsilon$ so that $A + \delta A$ is k-quasiseparable.

👁 *A matrix with ϵ-quasiseparable rank of k can be well-approximated by a matrix with exact quasiseparable rank k!*

👁 *If the* **spectra** *of A and $-B$ are* **well-separated** *in the Zolotarev sense, we can* **preserve structure***!*

❓ How can we operate efficiently with these matrix structures?

💵 **Reduced cost** for BLAS-like operations,

Sylvester with quasiseparable matrices

Theorem (Massei, Palitta, and Robol 2018, Theorem 2.16)

Let A be of ϵ-quasiseparable rank k, for $\epsilon > 0$. Then, there exists a matrix δA of norm bounded by $\|\delta A\|_2 \leq 2\sqrt{n} \cdot \epsilon$ so that $A + \delta A$ is k-quasiseparable.

- *A matrix with ϵ-quasiseparable rank of k can be well-approximated by a matrix with exact quasiseparable rank k!*
- *If the* **spectra** *of A and $-B$ are* **well-separated** *in the Zolotarev sense, we can* **preserve structure***!*

- How can we operate efficiently with these matrix structures?
- **Reduced cost** for BLAS-like operations,
- *Contained storage cost.*

Sylvester with quasiseparable matrices

Theorem (Massei, Palitta, and Robol 2018, Theorem 2.16)

Let A be of ϵ-quasiseparable rank k, for $\epsilon > 0$. Then, there exists a matrix δA of norm bounded by $\|\delta A\|_2 \leq 2\sqrt{n} \cdot \epsilon$ so that $A + \delta A$ is k-quasiseparable.

- *A matrix with ϵ-quasiseparable rank of k can be well-approximated by a matrix with exact quasiseparable rank k!*
- *If the* **spectra** *of A and $-B$ are* **well-separated** *in the Zolotarev sense, we can* **preserve structure**!

How can we operate efficiently with these matrix structures?

- **Reduced cost** for BLAS-like operations,
- *Contained storage cost.*
- Hierarchical matrix formats!

Hierarchical matrix formats

There exist **many hierarchical matrix formats**:

- $\mathcal{H}$-Matrices,
- $\mathcal{H}^2$-Matrices,
- **H**ierarchical **O**ff-**D**iagonal **L**ow-**R**ank (HODLR),
- **H**ierarchically **S**emi**S**eparable (HSS),
- **B**lock **L**ow-**R**ank (BLR).

Hierarchical matrix formats

There exist **many hierarchical matrix formats**:

- $\mathcal{H}$-Matrices,
- $\mathcal{H}^2$-Matrices,
- **H**ierarchical **O**ff-**D**iagonal **L**ow-**R**ank (HODLR),
- **H**ierarchically **S**emi**S**eparable (HSS),
- **B**lock **L**ow-**R**ank (BLR).

🕓 The topic would deserve a Ph.D. course on its own...

Hierarchical matrix formats

There exist **many hierarchical matrix formats**:

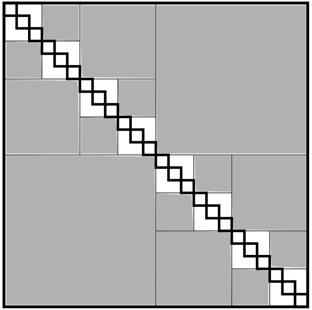
- $\mathcal{H}$-Matrices,
- $\mathcal{H}^2$-Matrices,
- **H**ierarchical **O**ff-**D**iagonal **L**ow-**R**ank (HODLR),
- **H**ierarchically **S**emi**S**eparable (HSS),
- **B**lock **L**ow-**R**ank (BLR).

The topic would deserve a Ph.D. course on its own… We are gonna focus only on the case of HODLR matrices (Hackbusch 2015, Chapter 3).

Hierarchical matrix formats

There exist **many hierarchical matrix formats**:

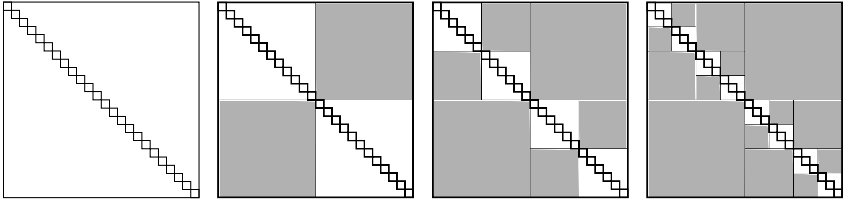
- $\mathcal{H}$-Matrices,
- $\mathcal{H}^2$-Matrices,
- **H**ierarchical **O**ff-**D**iagonal **L**ow-**R**ank (HODLR),
- **H**ierarchically **S**emi**S**eparable (HSS),
- **B**lock **L**ow-**R**ank (BLR).



The topic would deserve a Ph.D. course on its own... We are gonna focus only on the case of HODLR matrices (Hackbusch 2015, Chapter 3).

HODLR-matrices

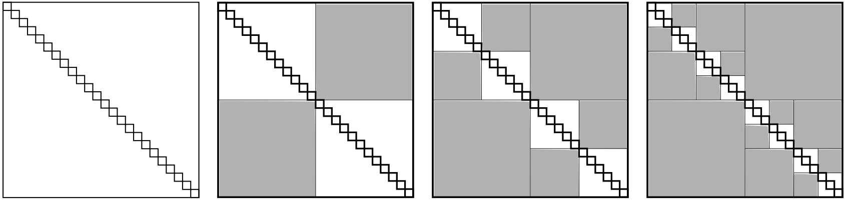
The **general idea**:



- The **grey blocks** are **low rank matrices** represented **in** a **compressed form**,
- the *diagonal blocks* in the last step are *stored as dense matrices*.

HODLR-matrices

The **general idea**:



- The **grey blocks** are **low rank matrices** represented **in** a **compressed form**,
- the *diagonal blocks* in the last step are *stored as dense matrices*.
- We need now a **formal definition** and a way to **define operations**.

HODLR-matrices: trees

Cluster tree

Given $n \in \mathbb{N}$, let $\mathcal{T}_p$ be a completely balanced binary tree of depth p whose nodes are subsets of $\{1, \dots, n\}$. We say that $\mathcal{T}_p$ is a *cluster tree* if it satisfies:

HODLR-matrices: trees

Cluster tree

Given $n \in \mathbb{N}$, let $\mathcal{T}_p$ be a completely balanced binary tree of depth p whose nodes are subsets of $\{1, \ldots, n\}$. We say that $\mathcal{T}_p$ is a *cluster tree* if it satisfies:

- The root is $I_1^0 := I = \{1, \ldots, n\}$.

HODLR-matrices: trees

Cluster tree

Given $n \in \mathbb{N}$, let $\mathcal{T}_p$ be a completely balanced binary tree of depth p whose nodes are subsets of $\{1, \ldots, n\}$. We say that $\mathcal{T}_p$ is a *cluster tree* if it satisfies:

- The root is $I_1^0 := I = \{1, \ldots, n\}$.
- The nodes at level ℓ, denoted by $I_1^\ell, \ldots, I_{2^\ell}^\ell$, form a partitioning of $\{1, \ldots, n\}$ into consecutive indices:
$$I_i^\ell = \{n_{i-1}^{(\ell)} + 1 \ldots, n_i^{(\ell)} - 1, n_i^{(\ell)}\}$$
for some integers $0 = n_0^{(\ell)} \leq n_1^{(\ell)} \leq \cdots \leq n_{2^\ell}^{(\ell)} = n$, $\ell = 0, \ldots p$. In particular, if $n_{i-1}^{(\ell)} = n_i^{(\ell)}$ then $I_i^\ell = \emptyset$.

HODLR-matrices: trees

Cluster tree

Given $n \in \mathbb{N}$, let $\mathcal{T}_p$ be a completely balanced binary tree of depth p whose nodes are subsets of $\{1, \ldots, n\}$. We say that $\mathcal{T}_p$ is a *cluster tree* if it satisfies:

- The root is $I_1^0 := I = \{1, \ldots, n\}$.
- The nodes at level ℓ, denoted by $I_1^\ell, \ldots, I_{2^\ell}^\ell$, form a partitioning of $\{1, \ldots, n\}$ into consecutive indices:
$$I_i^\ell = \{n_{i-1}^{(\ell)} + 1 \ldots, n_i^{(\ell)} - 1, n_i^{(\ell)}\}$$
for some integers $0 = n_0^{(\ell)} \leq n_1^{(\ell)} \leq \cdots \leq n_{2^\ell}^{(\ell)} = n$, $\ell = 0, \ldots p$. In particular, if $n_{i-1}^{(\ell)} = n_i^{(\ell)}$ then $I_i^\ell = \emptyset$.
- The node I_i^ℓ has children $I_{2i-1}^{\ell+1}$ and $I_{2i}^{\ell+1}$, for any $1 \leq \ell \leq p - 1$. The children form a partitioning of their parent.

HODLR-matrices: trees

Cluster tree

Given $n \in \mathbb{N}$, let $\mathcal{T}_p$ be a completely balanced binary tree of depth p whose nodes are subsets of $\{1, \ldots, n\}$. We say that $\mathcal{T}_p$ is a *cluster tree* if it satisfies:

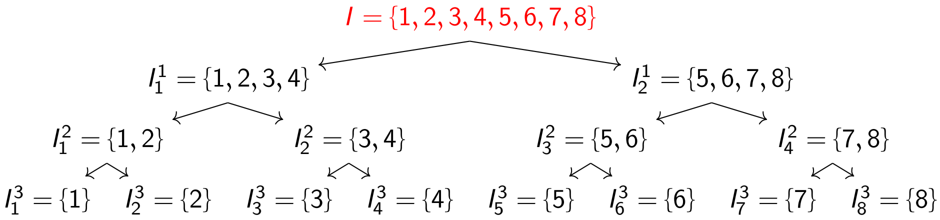
- The root is $I_1^0 := I = \{1, \ldots, n\}$.
- The nodes at level ℓ, denoted by $I_1^\ell, \ldots, I_{2^\ell}^\ell$, form a partitioning of $\{1, \ldots, n\}$ into consecutive indices:

  $$I_i^\ell = \{n_{i-1}^{(\ell)} + 1 \ldots, n_i^{(\ell)} - 1, n_i^{(\ell)}\}$$

  for some integers $0 = n_0^{(\ell)} \leq n_1^{(\ell)} \leq \cdots \leq n_{2^\ell}^{(\ell)} = n$, $\ell = 0, \ldots p$. In particular, if $n_{i-1}^{(\ell)} = n_i^{(\ell)}$ then $I_i^\ell = \emptyset$.
- The node I_i^ℓ has children $I_{2i-1}^{\ell+1}$ and $I_{2i}^{\ell+1}$, for any $1 \leq \ell \leq p - 1$. The children form a partitioning of their parent.

Nodes at a level ℓ partition A into a $2^\ell \times 2^\ell$ block matrix with blocks $\{A(I_i^\ell, I_j^\ell)\}_{i,j=1}^{2^\ell}$.

HODLR-matrices: trees



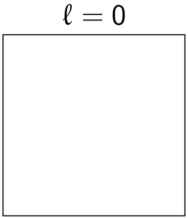
$I = \{1,2,3,4,5,6,7,8\}$

$I_1^1 = \{1,2,3,4\}$ $I_2^1 = \{5,6,7,8\}$

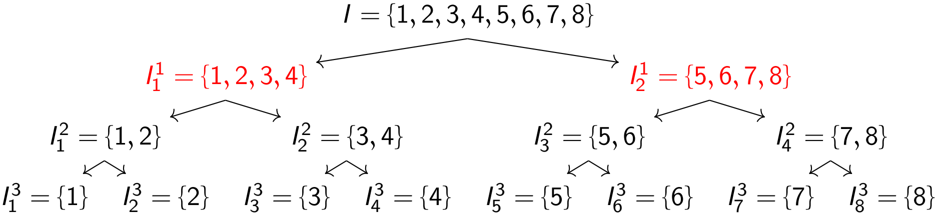
$I_1^2 = \{1,2\}$ $I_2^2 = \{3,4\}$ $I_3^2 = \{5,6\}$ $I_4^2 = \{7,8\}$

$I_1^3 = \{1\}$ $I_2^3 = \{2\}$ $I_3^3 = \{3\}$ $I_4^3 = \{4\}$ $I_5^3 = \{5\}$ $I_6^3 = \{6\}$ $I_7^3 = \{7\}$ $I_8^3 = \{8\}$

- The root $I = \{1, \ldots, 8\}$,



HODLR-matrices: trees



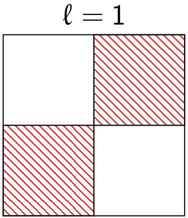
$I = \{1,2,3,4,5,6,7,8\}$

$I_1^1 = \{1,2,3,4\}$ $I_2^1 = \{5,6,7,8\}$

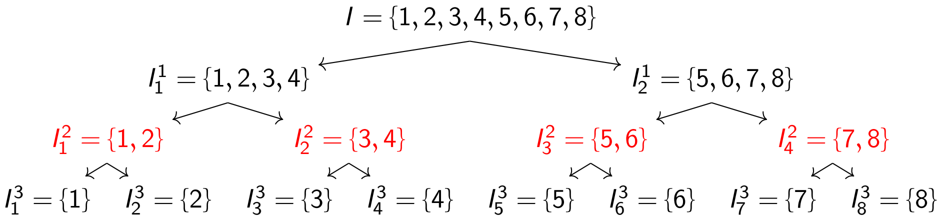
$I_1^2 = \{1,2\}$ $I_2^2 = \{3,4\}$ $I_3^2 = \{5,6\}$ $I_4^2 = \{7,8\}$

$I_1^3 = \{1\}$ $I_2^3 = \{2\}$ $I_3^3 = \{3\}$ $I_4^3 = \{4\}$ $I_5^3 = \{5\}$ $I_6^3 = \{6\}$ $I_7^3 = \{7\}$ $I_8^3 = \{8\}$

- The root $I = \{1, \dots, 8\}$,
- Nodes at level 1: I_1^1 and I_1^2,



HODLR-matrices: trees



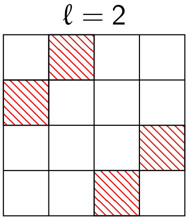
$I = \{1,2,3,4,5,6,7,8\}$

$I_1^1 = \{1,2,3,4\}$ $I_2^1 = \{5,6,7,8\}$

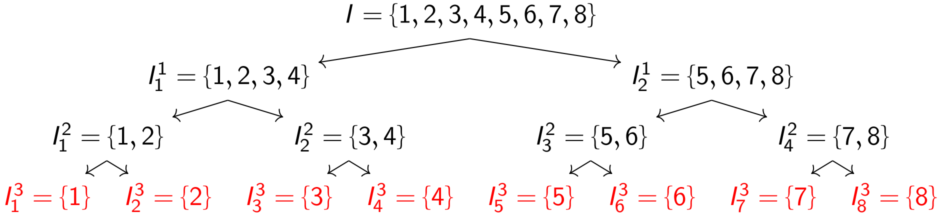
$I_1^2 = \{1,2\}$ $I_2^2 = \{3,4\}$ $I_3^2 = \{5,6\}$ $I_4^2 = \{7,8\}$

$I_1^3 = \{1\}$ $I_2^3 = \{2\}$ $I_3^3 = \{3\}$ $I_4^3 = \{4\}$ $I_5^3 = \{5\}$ $I_6^3 = \{6\}$ $I_7^3 = \{7\}$ $I_8^3 = \{8\}$

- The root $I = \{1, \ldots, 8\}$,
- Nodes at level 1: I_1^1 and I_1^2,
- Nodes at level 2: 🍃$(I_1^1)$$= \{I_1^2, I_2^2\}$, 🍃$(I_2^1)$$= \{I_3^2, I_4^2\}$,



HODLR-matrices: trees



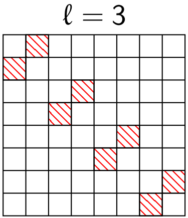
$I = \{1, 2, 3, 4, 5, 6, 7, 8\}$

$I_1^1 = \{1, 2, 3, 4\}$ $I_2^1 = \{5, 6, 7, 8\}$

$I_1^2 = \{1, 2\}$ $I_2^2 = \{3, 4\}$ $I_3^2 = \{5, 6\}$ $I_4^2 = \{7, 8\}$

$I_1^3 = \{1\}$ $I_2^3 = \{2\}$ $I_3^3 = \{3\}$ $I_4^3 = \{4\}$ $I_5^3 = \{5\}$ $I_6^3 = \{6\}$ $I_7^3 = \{7\}$ $I_8^3 = \{8\}$

- The root $I = \{1, \ldots, 8\}$,
- Nodes at level 1: I_1^1 and I_1^2,
- Nodes at level 2: 🍃$(I_1^1) = \{I_1^2, I_2^2\}$, 🍃$(I_2^1) = \{I_3^2, I_4^2\}$,
- Nodes at level 3: 🍃$(I_1^2) = \{I_1^3, I_2^3\}$, ..., 🍃$(I_4^2) = \{I_7^3, I_8^3\}$.



HODLR-matrices: definition

HODLR matrix

Let $A \in \mathbb{R}^{n \times n}$ and consider a cluster tree $\mathcal{T}_p$.

1. Given $k \in \mathbb{N}$, A is said to be a $(\mathcal{T}_p, k)$-HODLR matrix if every off-diagonal block

$$A(I_i^\ell, I_j^\ell) \quad \text{such that } I_i^\ell \text{ and } I_j^\ell \text{ are siblings in } \mathcal{T}_p, \quad \ell = 1, \dots, p,$$

   has rank at most k.
2. The HODLR rank of A (with respect to $\mathcal{T}_p$) is the smallest integer k such that A is a $(\mathcal{T}_p, k)$-HODLR matrix.

HODLR-matrices: definition

HODLR matrix

Let $A \in \mathbb{R}^{n \times n}$ and consider a cluster tree $\mathcal{T}_p$.

1. Given $k \in \mathbb{N}$, A is said to be a $(\mathcal{T}_p, k)$-HODLR matrix if every off-diagonal block
$$A(I_i^\ell, I_j^\ell) \quad \text{such that } I_i^\ell \text{ and } I_j^\ell \text{ are siblings in } \mathcal{T}_p, \quad \ell = 1, \ldots, p,$$
has rank at most k.
2. The HODLR rank of A (with respect to $\mathcal{T}_p$) is the smallest integer k such that A is a $(\mathcal{T}_p, k)$-HODLR matrix.

- $\mathcal{T}_p$ is often chosen to be as balanced as possible, i.e., cardinalities of I_i^ℓ are nearly equal for a given ℓ, with a dept determined by a minimal diagonal block size $n_{\min}$.

HODLR-matrices: definition

HODLR matrix

Let $A \in \mathbb{R}^{n \times n}$ and consider a cluster tree $\mathcal{T}_p$.

1. Given $k \in \mathbb{N}$, A is said to be a $(\mathcal{T}_p, k)$-HODLR matrix if every off-diagonal block

$$A(I_i^\ell, I_j^\ell) \quad \text{such that } I_i^\ell \text{ and } I_j^\ell \text{ are siblings in } \mathcal{T}_p, \quad \ell = 1, \ldots, p,$$

   has rank at most k.
2. The HODLR rank of A (with respect to $\mathcal{T}_p$) is the smallest integer k such that A is a $(\mathcal{T}_p, k)$-HODLR matrix.

- $\mathcal{T}_p$ is often chosen to be as balanced as possible, i.e., cardinalities of I_i^ℓ are nearly equal for a given ℓ, with a dept determined by a minimal diagonal block size $n_{\min}$.
- The classical choice is to have a **binary tree**, i.e., $n = 2^p n_{\min}$.

HODLR-matrices: occupied space

If we assume **identical ranks** k and a **balanced partitioning** then

- Storage for off-diagonal blocks $A(I_i^\ell, I_j^\ell) = U_i^{(\ell)} (V_j^{(\ell)})^T,\ U_i^{(\ell)}, V_j^{(\ell)} \in \mathbb{R}^{m_\ell \times k}$:
  On level $\ell > 0$ there are 2^ℓ off-diagonal blocks

$$2k \sum_{\ell=1}^{p} 2^\ell m_\ell = 2kn_0 \sum_{\ell=1}^{p} 2^\ell 2^{p-\ell} 2kn_0 p 2^p = 2knp = 2kn \log_2(n/n_0),$$

HODLR-matrices: occupied space

If we assume **identical ranks** k and a **balanced partitioning** then

- Storage for off-diagonal blocks $A(I_i^\ell, I_j^\ell) = U_i^{(\ell)}(V_j^{(\ell)})^T$, $U_i^{(\ell)}, V_j^{(\ell)} \in \mathbb{R}^{m_\ell \times k}$:
  On level $\ell > 0$ there are 2^ℓ off-diagonal blocks

$$2k \sum_{\ell=1}^{p} 2^\ell m_\ell = 2kn_0 \sum_{\ell=1}^{p} 2^\ell 2^{p-\ell} 2kn_0 p 2^p = 2knp = 2kn \log_2(n/n_0),$$

- Storage requirements for diagonal blocks

$$2^p n_0^2 = nn_0,$$

HODLR-matrices: occupied space

If we assume **identical ranks** k and a **balanced partitioning** then

- Storage for off-diagonal blocks $A(I_i^\ell, I_j^\ell) = U_i^{(\ell)}(V_j^{(\ell)})^T$, $U_i^{(\ell)}, V_j^{(\ell)} \in \mathbb{R}^{m_\ell \times k}$:
  On level $\ell > 0$ there are 2^ℓ off-diagonal blocks

$$2k \sum_{\ell=1}^{p} 2^\ell m_\ell = 2kn_0 \sum_{\ell=1}^{p} 2^\ell 2^{p-\ell} 2kn_0 p 2^p = 2knp = 2kn \log_2({}^{n}/{}_{n_0}),$$

- Storage requirements for diagonal blocks

$$2^p n_0^2 = nn_0,$$

- Total, assuming $n_0 = O(1)$, is then

$$O(kn \log n).$$

HODLR-matrices: occupied space

If we assume **identical ranks** k and a **balanced partitioning** then

- Storage for off-diagonal blocks $A(I_i^\ell, I_j^\ell) = U_i^{(\ell)}(V_j^{(\ell)})^T$, $U_i^{(\ell)}, V_j^{(\ell)} \in \mathbb{R}^{m_\ell \times k}$: On level $\ell > 0$ there are 2^ℓ off-diagonal blocks

$$2k \sum_{\ell=1}^{p} 2^\ell m_\ell = 2kn_0 \sum_{\ell=1}^{p} 2^\ell 2^{p-\ell} 2kn_0 p 2^p = 2knp = 2kn \log_2(n/n_0),$$

- Storage requirements for diagonal blocks

$$2^p n_0^2 = nn_0,$$

- Total, assuming $n_0 = O(1)$, is then

$$O(kn \log n).$$

- Both requirements on ranks and partitioning can be relaxed to obtain similar results.

HODLR-matrices: building the representation

⚠ Is **non trivial** to construct structured representations efficiently, especially if you want to avoid computing the whole n^2 coefficients!

HODLR-matrices: building the representation

⚠ Is **non trivial** to construct structured representations efficiently, especially if you want to avoid computing the whole n^2 coefficients!

- Build a **cluster tree** $\mathcal{T}_p$ for the given index set,

HODLR-matrices: building the representation

⚠ Is **non trivial** to construct structured representations efficiently, especially if you want to avoid computing the whole n^2 coefficients!

- Build a **cluster tree** $\mathcal{T}_p$ for the given index set,

If A is **dense**:

- Use Householder **QR decomposition** with column pivoting or **SVD** on off-diagonal blocks,

HODLR-matrices: building the representation

⚠ Is **non trivial** to construct structured representations efficiently, especially if you want to avoid computing the whole n^2 coefficients!

- Build a **cluster tree** $\mathcal{T}_p$ for the given index set,

If A is **dense**:

- Use Householder **QR decomposition** with column pivoting or **SVD** on off-diagonal blocks,
- The **rank** of each off-diagonal block $A(I_i^p, I_j^p)$ is chosen such that the spectral **norm of the approximation error** is bounded by ϵ times $\|A(I_i^p, I_j^p)\|_2$.

HODLR-matrices: building the representation

⚠ Is **non trivial** to construct structured representations efficiently, especially if you want to avoid computing the whole n^2 coefficients!

- Build a **cluster tree** $\mathcal{T}_p$ for the given index set,

If A is **dense**:

- Use Householder **QR decomposition** with column pivoting or **SVD** on off-diagonal blocks,
- The **rank** of each off-diagonal block $A(I_i^p, I_j^p)$ is chosen such that the spectral **norm of the approximation error** is bounded by ϵ times $\|A(I_i^p, I_j^p)\|_2$.

If A is **sparse**:

- Use a **two sided Lanczos method** only requiring matrix-vector multiplications with an off-diagonal block and its transpose, combined with recompression to each off-diagonal block.

HODLR-matrices: building the representation

⚠ Is **non trivial** to construct structured representations efficiently, especially if you want to avoid computing the whole n^2 coefficients!

- Build a **cluster tree** $\mathcal{T}_p$ for the given index set,

If A is **dense**:

- Use Householder **QR decomposition** with column pivoting or **SVD** on off-diagonal blocks,
- The **rank** of each off-diagonal block $A(I_i^p, I_j^p)$ is chosen such that the spectral **norm of the approximation error** is bounded by ϵ times $\|A(I_i^p, I_j^p)\|_2$.

If A is **sparse**:

- Use a **two sided Lanczos method** only requiring matrix-vector multiplications with an off-diagonal block and its transpose, combined with recompression to each off-diagonal block.

If A is **structured** use an *ad-hoc* constructor!

HODLR of Grünwald–Letnikov

Theorem (Fiedler 2010, Theorem A)

Let $\mathbf{x}, \mathbf{y}$ two real vectors of length N, with ascending and descending ordered entries, respectively. Moreover, we denote with $C(\mathbf{x}, \mathbf{y})$ the Cauchy matrix defined by

$$C_{ij} = \frac{1}{x_i - y_j}, \qquad i, j = 1, \dots, N.$$

If $C(\mathbf{x}, \mathbf{y}) = C(\mathbf{x}, \mathbf{y})^T$, $x_i \in [a, b]$, $y_j \in [c, d]$ with $a > d$, then $C(\mathbf{x}, \mathbf{y})$ is positive definite.

HODLR of Grünwald–Letnikov

Theorem (Fiedler 2010, Theorem A)

Let $\mathbf{x}, \mathbf{y}$ two real vectors of length N, with ascending and descending ordered entries, respectively. Moreover, we denote with $C(\mathbf{x}, \mathbf{y})$ the Cauchy matrix defined by

$$C_{ij} = \frac{1}{x_i - y_j}, \qquad i, j = 1, \ldots, N.$$

If $C(\mathbf{x}, \mathbf{y}) = C(\mathbf{x}, \mathbf{y})^T$, $x_i \in [a, b]$, $y_j \in [c, d]$ with $a > d$, then $C(\mathbf{x}, \mathbf{y})$ is positive definite.

Theorem (Beckermann and Townsend 2019, Theorem 5.5)

Let H be a positive semidefinite Hankel matrix of size N. Then, the ϵ-rank of H is bounded by

$$\mathrm{rank}_\epsilon(H) \leq 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi}N\right) \log\left(\frac{16}{\epsilon}\right) \right\rceil \triangleq \mathfrak{B}(N, \epsilon).$$

HODLR of Grünwald–Letnikov

We need to work with $G_N \in \mathbb{R}^{N\times N}$

$$G_N = -\begin{bmatrix} g_1^{(\alpha)} & g_0^{(\alpha)} & 0 & \cdots & 0 & 0 \\ g_2^{(\alpha)} & g_1^{(\alpha)} & g_0^{(\alpha)} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ g_{N-1}^{(\alpha)} & \ddots & \ddots & \ddots & g_1^{(\alpha)} & g_0^{(\alpha)} \\ g_N^{(\alpha)} & g_{N-1}^{(\alpha)} & \cdots & \cdots & g_2^{(\alpha)} & g_1^{(\alpha)} \end{bmatrix}$$

Lemma (Massei, Mazza, and Robol 2019)

Consider the Hankel matrix H defined as

$$H = (h_{ij}), \qquad h_{ij} = g_{i+j}^{(\alpha)},$$

for $1 \leq \alpha \leq 2$. Then, H is positive semidefinite.

- Show that H is obtained as the sum of a positive definite Cauchy matrix and a positive semidefinite matrix.
- Use the result by Beckermann and Townsend 2019.

HODLR of Grünwald–Letnikov

Proof. For $k \geq 2$ we rewrite $g_k^{(\alpha)}$ as

$$\begin{aligned} g_k^{(\alpha)} &= \frac{(-1)^k}{k!}\alpha(\alpha-1)\dots(\alpha-k+1) \\ &= \frac{\alpha(\alpha-1)}{k!}(k-\alpha-1)(k-\alpha-2)\dots(2-\alpha) \\ &= \alpha(\alpha-1)\frac{\Gamma(k-\alpha)}{\Gamma(k+1)\Gamma(2-\alpha)}. \end{aligned}$$

HODLR of Grünwald–Letnikov

Proof. For $k \geq 2$ we rewrite $g_k^{(\alpha)}$ as

$$g_k^{(\alpha)} = \alpha(\alpha-1)\frac{\Gamma(k-\alpha)}{\Gamma(k+1)\Gamma(2-\alpha)}.$$

Use the Gauss representation of the Euler Γ

$$\Gamma(z) = \lim_{m\to\infty} \frac{m!m^z}{z(z+1)(z+2)\dots(z+m)}, \quad z \neq \{0,-1,-2,\dots\},$$

we rewrite

$$g_k^{(\alpha)} = \alpha(\alpha-1)\lim_{m\to\infty}\frac{1}{m!m^3}\prod_{p=0}^{m}\frac{k+1+p}{k-\alpha+p}(2-\alpha+p).$$

HODLR of Grünwald–Letnikov

Proof. For $k \geq 2$ we rewrite $g_k^{(\alpha)}$ as

$$g_k^{(\alpha)} = \alpha(\alpha-1)\frac{\Gamma(k-\alpha)}{\Gamma(k+1)\Gamma(2-\alpha)}.$$

We rewrite

$$H = \lim_{m\to+\infty} H_0 \circ \ldots \circ H_m, \qquad (H_p)_{ij} = \frac{i+j+1+p}{i+j-\alpha+p}$$

for $\circ$ the Hadamard product, $\{H_j\}_{j=0}^m$ Hankel matrices.

HODLR of Grünwald–Letnikov

Proof. For $k \geq 2$ we rewrite $g_k^{(\alpha)}$ as

$$g_k^{(\alpha)} = \alpha(\alpha - 1)\frac{\Gamma(k-\alpha)}{\Gamma(k+1)\Gamma(2-\alpha)}.$$

We rewrite

$$H = \lim_{m \to +\infty} H_0 \circ \ldots \circ H_m, \qquad (H_p)_{ij} = \frac{i+j+1+p}{i+j-\alpha+p}$$

for $\circ$ the Hadamard product, $\{H_j\}_{j=0}^{m}$ Hankel matrices. Schur Product Theorem tells us that "*the Hadamard product of two positive definite matrices is also a positive definite matrix*" $\Rightarrow$ If $H_0 \circ \ldots \circ H_m$ is positive semidefinite for every m then H is also positive semidefinite.

HODLR of Grünwald–Letnikov

Proof. For $k \geq 2$ we rewrite $g_k^{(\alpha)}$ as

$$g_k^{(\alpha)} = \alpha(\alpha - 1)\frac{\Gamma(k-\alpha)}{\Gamma(k+1)\Gamma(2-\alpha)}.$$

We rewrite

$$H = \lim_{m\to+\infty} H_0 \circ \ldots \circ H_m, \qquad (H_p)_{ij} = \frac{i+j+1+p}{i+j-\alpha+p}$$

for $\circ$ the Hadamard product, $\{H_j\}_{j=0}^m$ Hankel matrices. Rewrite

$$(H_p)_{ij} = \frac{i+j+1+p}{i+j-\alpha+p} = 1 + \frac{\alpha+1}{i+j-\alpha+p}$$

HODLR of Grünwald–Letnikov

Proof. For $k \geq 2$ we rewrite $g_k^{(\alpha)}$ as

$$g_k^{(\alpha)} = \alpha(\alpha - 1)\frac{\Gamma(k-\alpha)}{\Gamma(k+1)\Gamma(2-\alpha)}.$$

We rewrite

$$H = \lim_{m\to+\infty} H_0 \circ \ldots \circ H_m, \qquad (H_p)_{ij} = \frac{i+j+1+p}{i+j-\alpha+p}$$

for $\circ$ the Hadamard product, $\{H_j\}_{j=0}^m$ Hankel matrices. Rewrite

$$(H_p)_{ij} = 1 + \frac{\alpha+1}{i+j-\alpha+p}, \quad H_p = \mathbf{1}\mathbf{1}^T + (\alpha+1)\cdot C(\mathbf{x}, -\mathbf{x}), \quad \mathbf{x} = \begin{bmatrix} 1 \\ \vdots \\ N \end{bmatrix} + \frac{p-\alpha}{2}\mathbf{1},$$

$\mathbf{x} \geq 0$ for $\alpha < 2$, thus $C(\mathbf{x}, -\mathbf{x})$ is PD. Then H_p is positive semidefinite as the sum of a PD and positive semidefinite matrix. □

HODLR of Grünwald–Letnikov

Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\mathrm{qsrank}_\epsilon(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2}\log\left(\frac{4}{\pi}N\right)\log\left(\frac{32}{\epsilon}\right)\right\rceil.$$

Proof.

HODLR of Grünwald–Letnikov

Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\mathrm{qsrank}_{\epsilon}(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi} N\right) \log\left(\frac{32}{\epsilon}\right) \right\rceil.$$

Proof. We just need to work on the lower triangle, for the upper the rank is at most 1 (Hessenberg).

HODLR of Grünwald–Letnikov

Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\mathrm{qsrank}_\epsilon(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi} N\right) \log\left(\frac{32}{\epsilon}\right) \right\rceil.$$

Proof. Let $Y \in \mathbb{R}^{s\times t}$ be any lower off-diagonal block of G_N. *Without loss of generality* we assume that Y is maximal, i.e. $s + t = N$.

HODLR of Grünwald–Letnikov

Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\operatorname{qsrank}_\epsilon(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi} N\right) \log\left(\frac{32}{\epsilon}\right) \right\rceil .$$

Proof. Let $Y \in \mathbb{R}^{s \times t}$ be any lower off-diagonal block of G_N. *Without loss of generality* we assume that Y is maximal, i.e. $s + t = N$. (If $\operatorname{rank}(Y + \delta Y) = k$ and $\|\delta Y\|_2 \leq \epsilon \| G_N \|_2$ then the submatrices of δY verify the analogous claim for the corresponding ones of Y.)

HODLR of Grünwald–Letnikov

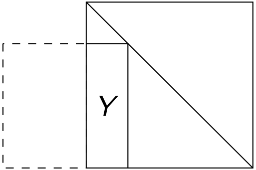
Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\mathrm{qsrank}_\epsilon(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi}N\right) \log\left(\frac{32}{\epsilon}\right)\right\rceil.$$

Proof. Let $Y \in \mathbb{R}^{s \times t}$ be any lower off-diagonal block of G_N. *Without loss of generality* we assume that Y is maximal, i.e. $s + t = N$.
Entries Y are given by $Y_{ij} = -g^{(\alpha)}_{1+i-j+t}$. Call $h = \max\{s, t\}$, and A the $h \times h$ matrix defined by $A_{ij} = -g^{(\alpha)}_{1+i-j+h}$.

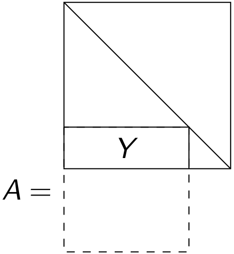
HODLR of Grünwald–Letnikov

$A =$ 

For every $1 \leq i \leq s$ and $1 \leq j \leq t$ one have
$Y_{ij} = -g^{(\alpha)}_{1+i-j+t} = -g^{(\alpha)}_{1+i-(j-t+h)+h} = A_{i,j-t+h}$.

Proof. Let $Y \in \mathbb{R}^{s\times t}$ be any lower off-diagonal block of G_N. *Without loss of generality* we assume that Y is maximal, i.e. $s + t = N$.
Entries Y are given by $Y_{ij} = -g^{(\alpha)}_{1+i-j+t}$. Call $h = \max\{s, t\}$, and A the $h \times h$ matrix defined by $A_{ij} = -g^{(\alpha)}_{1+i-j+h}$. Y coincides with either the last t columns or the first s rows of A.

HODLR of Grünwald–Letnikov



For every $1 \leq i \leq s$ and $1 \leq j \leq t$ one have
$Y_{ij} = -g^{(\alpha)}_{1+i-j+t} = -g^{(\alpha)}_{1+i-(j-t+h)+h} = A_{i,j-t+h}$.

Proof. Let $Y \in \mathbb{R}^{s\times t}$ be any lower off-diagonal block of G_N. *Without loss of generality* we assume that Y is maximal, i.e. $s + t = N$.
Entries Y are given by $Y_{ij} = -g^{(\alpha)}_{1+i-j+t}$. Call $h = \max\{s, t\}$, and A the $h \times h$ matrix defined by $A_{ij} = -g^{(\alpha)}_{1+i-j+h}$.$Y$ coincides with either the last t columns or the first s rows of A.

HODLR of Grünwald–Letnikov

Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\operatorname{qsrank}_{\epsilon}(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi}N\right) \log\left(\frac{32}{\epsilon}\right)\right\rceil.$$

Proof. Let $Y \in \mathbb{R}^{s\times t}$ be any lower off-diagonal block of G_N. *Without loss of generality* we assume that Y is maximal, i.e. $s + t = N$.
Entries Y are given by $Y_{ij} = -g^{(\alpha)}_{1+i-j+t}$. Call $h = \max\{s, t\}$, and A the $h \times h$ matrix defined by $A_{ij} = -g^{(\alpha)}_{1+i-j+h}$.$Y$ coincides with either the last t columns or the first s rows of A. In particular, Y is a submatrix of A and therefore $\|Y\|_2 \leq \|A\|_2$.

HODLR of Grünwald–Letnikov

Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\operatorname{qsrank}_{\epsilon}(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi}N\right) \log\left(\frac{32}{\epsilon}\right) \right\rceil .$$

Proof. Let $Y \in \mathbb{R}^{s \times t}$ be any lower off-diagonal block of G_N. *Without loss of generality* we assume that Y is maximal, i.e. $s + t = N$.

Entries Y are given by $Y_{ij} = -g^{(\alpha)}_{1+i-j+t}$. Call $h = \max\{s, t\}$, and A the $h \times h$ matrix defined by $A_{ij} = -g^{(\alpha)}_{1+i-j+h}$.$Y$ coincides with either the last t columns or the first s rows of A. In particular, Y is a submatrix of A and therefore $\|Y\|_2 \leq \|A\|_2$. ⚙ We need now to estimate $\|A\|_2$ in terms of $\|G_N\|_2$, thus we partition

$$A = \begin{bmatrix} A^{(11)} & A^{(12)} \\ A^{(21)} & A^{(22)} \end{bmatrix}, \qquad A^{(ij)} \in \mathbb{C}^{m_{ij} \times n_{ij}}, \qquad \begin{cases} m_{1j} = n_{i1} = \lceil \frac{h}{2} \rceil \\ m_{2j} = n_{i2} = \lfloor \frac{h}{2} \rfloor \end{cases}, \quad \begin{cases} h \leq N - 1, \\ m_{i,j} + n_{i,j} \leq N, \end{cases}$$

HODLR of Grünwald–Letnikov

Proof. and consider the subdiagonal block $T^{(ij)}$ of G_N defined by

$$T^{(ij)} = G_N(N-m_{ij}+1 : N, N-m_{ij}-n_{ij}+1 : N-m_{ij}), \qquad i,j = 1,2, \qquad \begin{array}{l} T^{(ij)} \in \mathbb{R}^{m_{ij} \times n_{ij}}, \\ m_{ij} + n_{ij} \leq N. \end{array}$$

HODLR of Grünwald–Letnikov

Proof. and consider the subdiagonal block $T^{(ij)}$ of G_N defined by

$$T^{(ij)} = G_N(N-m_{ij}+1 : N, N-m_{ij}-n_{ij}+1 : N-m_{ij}), \qquad i,j = 1,2, \qquad \begin{array}{l} T^{(ij)} \in \mathbb{R}^{m_{ij}\times n_{ij}}, \\ m_{ij}+n_{ij} \le N. \end{array}$$

- 👁 Since $g_j^{(\alpha)} > g_{j+1}^{(\alpha)} > 0$, $|T^{(ij)}| \ge |A^{(ij)}|$ for every $i,j = 1,2$,

HODLR of Grünwald–Letnikov

Proof. and consider the subdiagonal block $T^{(ij)}$ of G_N defined by

$$T^{(ij)} = G_N(N-m_{ij}+1 : N, N-m_{ij}-n_{ij}+1 : N-m_{ij}), \qquad i,j = 1,2, \qquad \begin{array}{l} T^{(ij)} \in \mathbb{R}^{m_{ij}\times n_{ij}}, \\ m_{ij}+n_{ij} \leq N. \end{array}$$

- 👁 Since $g_j^{(\alpha)} > g_{j+1}^{(\alpha)} > 0$, $|T^{(ij)}| \geq |A^{(ij)}|$ for every $i,j = 1,2$,
- 🔨 Being $T^{(ij)}$ and $A^{(ij)}$ nonpositive and the 2 norm monotonous, $\|A^{(ij)}\|_2 \leq \|T^{(ij)}\|_2$.

HODLR of Grünwald–Letnikov

Proof. and consider the subdiagonal block $T^{(ij)}$ of G_N defined by

$$T^{(ij)} = G_N(N - m_{ij} + 1 : N, N - m_{ij} - n_{ij} + 1 : N - m_{ij}), \qquad i, j = 1, 2, \qquad \begin{array}{l} T^{(ij)} \in \mathbb{R}^{m_{ij} \times n_{ij}}, \\ m_{ij} + n_{ij} \leq N. \end{array}$$

- Since $g_j^{(\alpha)} > g_{j+1}^{(\alpha)} > 0$, $|T^{(ij)}| \geq |A^{(ij)}|$ for every $i, j = 1, 2$,
- Being $T^{(ij)}$ and $A^{(ij)}$ nonpositive and the 2 norm monotonous, $\|A^{(ij)}\|_2 \leq \|T^{(ij)}\|_2$.
- By exploiting

$$\begin{aligned} \|A\|_2 &\leq \left\| \begin{bmatrix} A^{(11)} & \\ & A^{(22)} \end{bmatrix} \right\|_2 + \left\| \begin{bmatrix} & A^{(12)} \\ A^{(21)} & \end{bmatrix} \right\|_2 \\ &= \max\{\|A^{(11)}\|_2, \|A^{(22)}\|_2\} + \max\{\|A^{(12)}\|_2, \|A^{(21)}\|_2\} \end{aligned} \qquad \Rightarrow \quad \|A\|_2 \leq 2\|G_N\|_2.$$

HODLR of Grünwald–Letnikov

Proof. and consider the subdiagonal block $T^{(ij)}$ of G_N defined by

$$T^{(ij)} = G_N(N-m_{ij}+1:N, N-m_{ij}-n_{ij}+1:N-m_{ij}), \qquad i,j=1,2, \qquad \begin{array}{l} T^{(ij)} \in \mathbb{R}^{m_{ij}\times n_{ij}}, \\ m_{ij}+n_{ij} \le N. \end{array}$$

- Since $g_j^{(\alpha)} > g_{j+1}^{(\alpha)} > 0$, $|T^{(ij)}| \ge |A^{(ij)}|$ for every $i,j=1,2$,
- Being $T^{(ij)}$ and $A^{(ij)}$ nonpositive and the 2 norm monotonous, $\|A^{(ij)}\|_2 \le \|T^{(ij)}\|_2$.
- By exploiting

$$\begin{aligned} \|A\|_2 &\le \left\| \begin{bmatrix} A^{(11)} & \\ & A^{(22)} \end{bmatrix} \right\|_2 + \left\| \begin{bmatrix} & A^{(12)} \\ A^{(21)} & \end{bmatrix} \right\|_2 \qquad \Rightarrow \|A\|_2 \le 2\|G_N\|_2. \\ &= \max\{\|A^{(11)}\|_2, \|A^{(22)}\|_2\} + \max\{\|A^{(12)}\|_2, \|A^{(21)}\|_2\} \end{aligned}$$

- Conclude by the result on Hankel matrices!

HODLR of Grünwald–Letnikov

Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\mathrm{qsrank}_\epsilon(G_N) \le \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi} N\right) \log\left(\frac{32}{\epsilon}\right)\right\rceil.$$

Proof. We call J the $h \times h$ flip matrix, so that $-AJ$ is Hankel and positive semidefinite:

$$\mathrm{rank}_{\frac{\epsilon}{2}}(A) = \mathrm{rank}_{\frac{\epsilon}{2}}(AJ) \le \mathfrak{B}\left(N, \frac{\epsilon}{2}\right).$$

HODLR of Grünwald–Letnikov

Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\mathrm{qsrank}_{\epsilon}(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi} N\right) \log\left(\frac{32}{\epsilon}\right)\right\rceil.$$

Proof. We call J the $h \times h$ flip matrix, so that $-AJ$ is Hankel and positive semidefinite:

$$\mathrm{rank}_{\frac{\epsilon}{2}}(A) = \mathrm{rank}_{\frac{\epsilon}{2}}(AJ) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right).$$

Y is a submatrix of A, thus there exists δY such that

$$\|\delta Y\|_2 \leq \varepsilon \|G_N\|_2 \text{ and } \mathrm{rank}(Y + \delta Y) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right).$$

HODLR of Grünwald–Letnikov

Proposition (Massei, Mazza, and Robol 2019, Lemma 3.15)

For every $\epsilon > 0$, the ϵ-qsrank of G_N is bounded by

$$\mathrm{qsrank}_{\epsilon}(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right) = 2 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi} N\right) \log\left(\frac{32}{\epsilon}\right)\right\rceil .$$

Proof. We call J the $h \times h$ flip matrix, so that $-AJ$ is Hankel and positive semidefinite:

$$\mathrm{rank}_{\frac{\epsilon}{2}}(A) = \mathrm{rank}_{\frac{\epsilon}{2}}(AJ) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right).$$

Y is a submatrix of A, thus there exists δY such that

$$\|\delta Y\|_2 \leq \varepsilon \|G_N\|_2 \text{ and } \mathrm{rank}(Y + \delta Y) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right).$$

$\Rightarrow \mathrm{qsrank}_{\epsilon}(G_N) \leq \mathfrak{B}\left(N, \frac{\epsilon}{2}\right).$ $\square$

HODLR of Grünwald–Letnikov

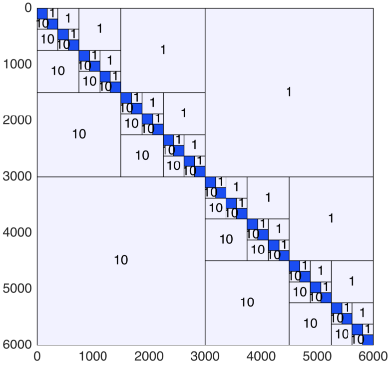
Let's do some experiments with the `hm-toolbox` (Massei, Robol, and Kressner 2020).

```
function G = glhodlrmatrix(N,alpha,tol)
%%GLMATRIX produces the GL discretization of
% the Riemann-Liouville derivative in HODLR
% format
g = gl(N,alpha);
c = zeros(N,1);
r = zeros(1,N);
r(1:2) = g(2:-1:1);
c(1:N) = g(2:end);
hodlroption( 'threshold', tol);
G = hodlr('toeplitz',c,r);
end
```

HODLR of Grünwald–Letnikov

Let's do some experiments with the `hm-toolbox` (Massei, Robol, and Kressner 2020).

```
function G = glhodlrmatrix(N,alpha,tol)
%%GLMATRIX produces the GL discretization of
% the Riemann-Liouville derivative in HODLR
% format
g = gl(N,alpha);
c = zeros(N,1);
r = zeros(1,N);
r(1:2) = g(2:-1:1);
c(1:N) = g(2:end);
hodlroption( 'threshold', tol);
G = hodlr('toeplitz',c,r);
end
```

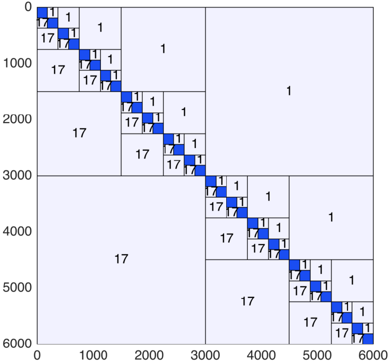


```
G = glhodlrmatrix(6000,1.5,1e-6);
```

HODLR of Grünwald–Letnikov

Let's do some experiments with the `hm-toolbox` (Massei, Robol, and Kressner 2020).

```
function G = glhodlrmatrix(N,alpha,tol)
%%GLMATRIX produces the GL discretization of
% the Riemann-Liouville derivative in HODLR
% format
g = gl(N,alpha);
c = zeros(N,1);
r = zeros(1,N);
r(1:2) = g(2:-1:1);
c(1:N) = g(2:end);
hodlroption( 'threshold', tol);
G = hodlr('toeplitz',c,r);
end
```

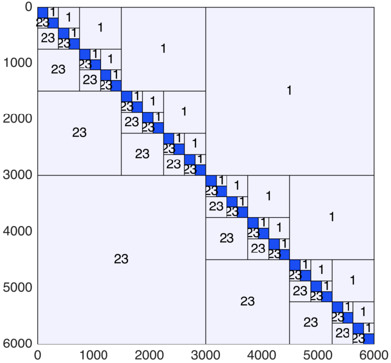


```
G = glhodlrmatrix(6000,1.5,1e-9);
```

HODLR of Grünwald–Letnikov

Let's do some experiments with the `hm-toolbox` (Massei, Robol, and Kressner 2020).

```
function G = glhodlrmatrix(N,alpha,tol)
%%GLMATRIX produces the GL discretization of
% the Riemann-Liouville derivative in HODLR
% format
g = gl(N,alpha);
c = zeros(N,1);
r = zeros(1,N);
r(1:2) = g(2:-1:1);
c(1:N) = g(2:end);
hodlroption( 'threshold', tol);
G = hodlr('toeplitz',c,r);
end
```



```
G = glhodlrmatrix(6000,1.5,1e-12);
```

HODLR Matrix: the whole discretization

Matrix G_N was only a piece of the whole discretization matrix

$$A_N = I_N + \frac{\Delta t}{h^\alpha}\left(D^{+}_{(m)} G_N + D^{-}_{(m)} G_N^T\right),$$

does it share the same structure?

HODLR Matrix: the whole discretization

Matrix G_N was only a piece of the whole discretization matrix

$$A_N = I_N + \frac{\Delta t}{h^{\alpha}} \left(D^{+}_{(m)} G_N + D^{-}_{(m)} G_N^T \right),$$

does it share the same structure?

Corollary (Massei, Mazza, and Robol 2019, Corollary 3.16)

$$\operatorname{qsrank}_{\epsilon}(A_N) \leq 3 + 2 \left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi} N\right) \log\left(\frac{32}{\hat{\epsilon}}\right) \right\rceil, \quad \hat{\epsilon} \triangleq \frac{\|A_N\|}{\|G_N\| \cdot \max\{\|D^{+}_{(m)}\|, \|D^{-}_{(m)}\|\}} \epsilon.$$

HODLR Matrix: the whole discretization

Matrix G_N was only a piece of the whole discretization matrix

$$A_N = I_N + \frac{\Delta t}{h^{\alpha}} \left(D^{+}_{(m)} G_N + D^{-}_{(m)} G_N^T \right),$$

does it share the same structure?

Corollary (Massei, Mazza, and Robol 2019, Corollary 3.16)

$$\operatorname{qsrank}_{\epsilon}(A_N) \leq 3 + 2 \left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi} N\right) \log\left(\frac{32}{\hat{\epsilon}}\right) \right\rceil, \quad \hat{\epsilon} \triangleq \frac{\|A_N\|}{\|G_N\| \cdot \max\{\|D^{+}_{(m)}\|, \|D^{-}_{(m)}\|\}} \epsilon.$$

Proof. Result is invariant under scaling, so assume wlog that $\frac{\Delta t}{h^{\alpha}} = 1$.

HODLR Matrix: the whole discretization

Matrix G_N was only a piece of the whole discretization matrix

$$A_N = I_N + \frac{\Delta t}{h^\alpha}\left(D^+_{(m)} G_N + D^-_{(m)} G_N^T\right),$$

does it share the same structure?

Corollary (Massei, Mazza, and Robol 2019, Corollary 3.16)

$$\operatorname{qsrank}_\epsilon(A_N) \leq 3 + 2\left\lceil \frac{2}{\pi^2} \log\left(\frac{4}{\pi}N\right) \log\left(\frac{32}{\hat{\epsilon}}\right)\right\rceil, \quad \hat{\epsilon} \triangleq \frac{\|A_N\|}{\|G_N\| \cdot \max\{\|D^+_{(m)}\|, \|D^-_{(m)}\|\}}\epsilon.$$

Proof. Result is invariant under scaling, so assume wlog that $\frac{\Delta t}{h^\alpha} = 1$. A generic off-diagonal block Y, wlog in the lower triangular part, If Y does not intersect the first subdiagonal, is a subblock of $D^+_{(m)} G_N$, so there exists a perturbation δY with norm bounded by $\|\delta Y\| \leq \|D^+_{(m)}\| \|G_N\| \cdot \hat{\epsilon}$ such that $Y + \delta Y$ has rank at most $\mathfrak{B}(N, \hat{\epsilon}/2)$. In particular, δY satisfies $\|\delta Y\| \leq \|A_N\| \cdot \epsilon$.

HODLR Matrix: the whole discretization

Matrix G_N was only a piece of the whole discretization matrix

$$A_N = I_N + \frac{\Delta t}{h^\alpha}\left(D_{(m)}^{+} G_N + D_{(m)}^{-} G_N^T\right),$$

does it share the same structure?

Corollary (Massei, Mazza, and Robol 2019, Corollary 3.16)

$$\operatorname{qsrank}_\epsilon(A_N) \leq 3 + 2\left\lceil \frac{2}{\pi^2}\log\left(\frac{4}{\pi}N\right)\log\left(\frac{32}{\hat{\epsilon}}\right)\right\rceil, \quad \hat{\epsilon} \triangleq \frac{\|A_N\|}{\|G_N\| \cdot \max\{\|D_{(m)}^{+}\|, \|D_{(m)}^{-}\|\}}\epsilon.$$

Proof. Result is invariant under scaling, so assume wlog that $\frac{\Delta t}{h^\alpha} = 1$. Since we have excluded one subdiagonal, a generic off-diagonal block Y we can find a perturbation with norm bounded by $\|A_N\| \cdot \epsilon$ such that $Y + \delta Y$ has rank $1 + \mathfrak{B}(N, \hat{\epsilon}/2)$. $\square$

A HODLR right-hand side

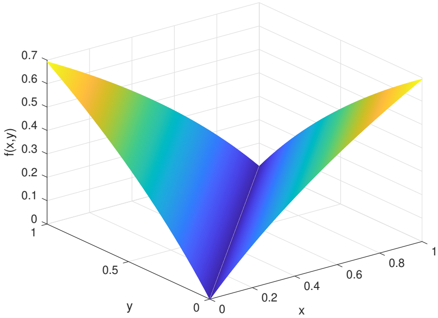
❓ What are right-hand sides functions $f(x, y, t)$ so that the matrix C has a HODLR structure?

A HODLR right-hand side

❓ What are right-hand sides functions $f(x, y, t)$ so that the matrix C has a HODLR structure?

Consider the function

$$f(x, y) = \log(\tau + |x - y|), \quad \tau > 0.$$



A HODLR right-hand side

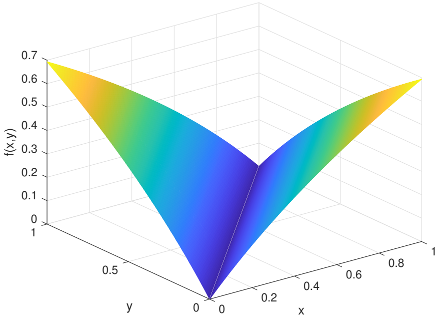
❓ What are right-hand sides functions $f(x, y, t)$ so that the matrix C has a HODLR structure?

Consider the function

$$f(x, y) = \log(\tau + |x - y|), \quad \tau > 0.$$

⚙ If we discretize it by *finite differences* on a rectangular domain we find

$$C_{i,j} = \log(\tau + |x_i - y_j|)$$



A HODLR right-hand side

❓ What are right-hand sides functions $f(x, y, t)$ so that the matrix C has a HODLR structure?

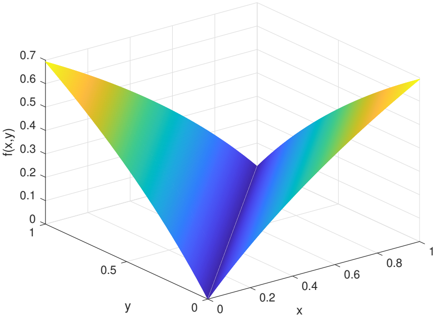
Consider the function

$$f(x, y) = \log(\tau + |x - y|), \quad \tau > 0.$$

⚙ If we discretize it by *finite differences* on a rectangular domain we find

$$C_{i,j} = \log(\tau + |x_i - y_j|)$$

💡 The modulus function it is **not regular in the whole domain** but it is **analytic** when the **sign** of $x - y$ is **constant**.



A HODLR right-hand side

What are right-hand sides functions $f(x, y, t)$ so that the matrix C has a HODLR structure?

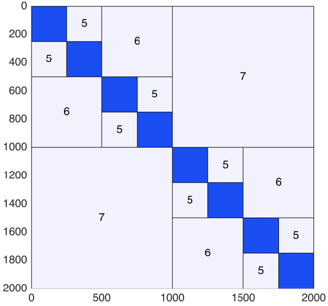
Consider the function

$$f(x, y) = \log(\tau + |x - y|), \quad \tau > 0.$$

- If we discretize it by *finite differences* on a rectangular domain we find

$$C_{i,j} = \log(\tau + |x_i - y_j|)$$

- The modulus function it is **not regular in the whole domain** but it is **analytic** when the **sign** of $x - y$ is **constant**.
- We can use again Chebyshev basis to approximate it in a separable fashion.



```
x = linspace(0,1,N); y = linspace(0,1,N);
[X,Y] = meshgrid(x,y); tau = 1;
C = log(tau + abs(X-Y)); hC = hodlr(C);
```

Separability (a bit more formally)

Separable expansion (Hackbusch 2015, Definition 4.4)

Take a function $\chi(x, y) : X \times Y \to \mathbb{R}$, we call

$$\chi(x,y) = \sum_{\nu=1}^{r} \phi_{\nu}^{(r)}(x)\psi_{\nu}^{(r)}(y) + R_r(x,y), \qquad \text{for } x \in X,\ y \in Y,$$

a *separable expansion* of χ with r terms in $X \times Y$ with remainder R_r.

- 🔧 To have an idea of the **goodness** of the *separable expansion*, we would like to have $\{\|R_r\|_{\infty}, \|R_r\|_{\mathbb{L}^p}\} \xrightarrow{r \to 0} 0$ as fast as possible, e.g., **exponentially**.

Separability (a bit more formally)

Separable expansion (Hackbusch 2015, Definition 4.4)

Take a function $\chi(x, y) : X \times Y \to \mathbb{R}$, we call

$$\chi(x, y) = \sum_{\nu=1}^{r} \phi_{\nu}^{(r)}(x)\psi_{\nu}^{(r)}(y) + R_r(x, y), \qquad \text{for } x \in X,\ y \in Y,$$

a *separable expansion* of χ with r terms in $X \times Y$ with remainder R_r.

- To have an idea of the **goodness** of the *separable expansion*, we would like to have $\{\|R_r\|_\infty, \|R_r\|_{\mathbb{L}^p}\} \xrightarrow{r \to 0} 0$ as fast as possible, e.g., **exponentially**.
- If $\|R_r\| \leq c_1 \exp(-c_2 r^\alpha) \Rightarrow \|R_r\| \leq \varepsilon$ if $r \geq \left\lceil \left(\frac{1}{c_2} \log^{1/\alpha} \frac{c_1}{\varepsilon} \right) \right\rceil = O(\log^{1/\alpha} 1/\varepsilon)$ $\varepsilon \to 0$.

Separability (a bit more formally)

Separable expansion (Hackbusch 2015, Definition 4.4)

Take a function $\chi(x, y) : X \times Y \to \mathbb{R}$, we call

$$\chi(x, y) = \sum_{\nu=1}^{r} \phi_{\nu}^{(r)}(x)\psi_{\nu}^{(r)}(y) + R_r(x, y), \qquad \text{for } x \in X,\ y \in Y,$$

a *separable expansion* of χ with r terms in $X \times Y$ with remainder R_r.

- To have an idea of the **goodness** of the *separable expansion*, we would like to have $\{\|R_r\|_{\infty}, \|R_r\|_{\mathbb{L}^p}\} \xrightarrow{r \to 0} 0$ as fast as possible, e.g., **exponentially**.
- If $\|R_r\| \leq c_1 \exp(-c_2 r^{\alpha}) \Rightarrow \|R_r\| \leq \varepsilon$ if $r \geq \left\lceil \left(\frac{1}{c_2} \log^{1/\alpha} \frac{c_1}{\varepsilon} \right) \right\rceil = O(\log^{1/\alpha} 1/\varepsilon)$ $\varepsilon \to 0$.
- We can use Taylor expansions, Chebyshev expansion, Hermite/Lagrange interpolation, cross approximation... In all the cases, the behavior of R_r is tied to the regularity of $\chi(x, y)$; see (Hackbusch 2015, Chapter 4).

BLAS with HODLR format

? We now have **everything represented in the right format**, but can we operate with it?

BLAS with HODLR format

❓ We now have **everything represented in the right format**, but can we operate with it?

$\mathbf{y} = A\mathbf{x}$: Matrix-vector products, *recursively*:

$$\begin{aligned}
\mathbf{y}(I_1^1) &= A(I_1^1, I_1^1)\mathbf{x}(I_1^1) + A(I_1^1, I_2^1)\mathbf{x}(I_2^1),\\
\mathbf{y}(I_2^1) &= A(I_2^1, I_1^1)\mathbf{x}(I_1^1) + A(I_2^1, I_2^1)\mathbf{x}(I_2^1).
\end{aligned}$$

BLAS with HODLR format

❓ We now have **everything represented in the right format**, but can we operate with it?

$\mathbf{y} = A\mathbf{x}$: Matrix-vector products, *recursively*:

$$\begin{aligned}\mathbf{y}(I_1^1) &= A(I_1^1, I_1^1)\mathbf{x}(I_1^1) + A(I_1^1, I_2^1)\mathbf{x}(I_2^1),\\ \mathbf{y}(I_2^1) &= A(I_2^1, I_1^1)\mathbf{x}(I_1^1) + A(I_2^1, I_2^1)\mathbf{x}(I_2^1).\end{aligned}$$

⚙ Off-diagonal blocks $A(I_1^1, I_2^1)$ and $A(I_2^1, I_1^1)$ are obtained by multiplying $n/2 \times n/2$ low-rank matrix with vector. This **cost** $c_{\mathrm{LR}\cdot\mathbf{x}}(n/2) = 2nk$.

BLAS with HODLR format

❓ We now have **everything represented in the right format**, but can we operate with it?

$\mathbf{y} = A\mathbf{x}$: Matrix-vector products, *recursively*:

$$\mathbf{y}(I_1^1) = A(I_1^1, I_1^1)\mathbf{x}(I_1^1) + A(I_1^1, I_2^1)\mathbf{x}(I_2^1),$$
$$\mathbf{y}(I_2^1) = A(I_2^1, I_1^1)\mathbf{x}(I_1^1) + A(I_2^1, I_2^1)\mathbf{x}(I_2^1).$$

- ⚙ Off-diagonal blocks $A(I_1^1, I_2^1)$ and $A(I_2^1, I_1^1)$ are obtained by multiplying $n/2 \times n/2$ low-rank matrix with vector. This **cost** $c_{\text{LR}\cdot\mathbf{x}}(n/2) = 2nk$.
- ⚙ Diagonal blocks are processed recursively at a **cost**

$$c_{A\cdot\mathbf{x}}(n) = 2c_{A\cdot\mathbf{x}}(n/2) + 4kn + n.$$

BLAS with HODLR format

❓ We now have **everything represented in the right format**, but can we operate with it?

$\mathbf{y} = A\mathbf{x}$: Matrix-vector products, *recursively*:

$$\begin{aligned}\mathbf{y}(I_1^1) &= A(I_1^1, I_1^1)\mathbf{x}(I_1^1) + A(I_1^1, I_2^1)\mathbf{x}(I_2^1),\\ \mathbf{y}(I_2^1) &= A(I_2^1, I_1^1)\mathbf{x}(I_1^1) + A(I_2^1, I_2^1)\mathbf{x}(I_2^1).\end{aligned}$$

- Off-diagonal blocks $A(I_1^1, I_2^1)$ and $A(I_2^1, I_1^1)$ are obtained by multiplying $n/2 \times n/2$ low-rank matrix with vector. This **cost** $c_{\mathrm{LR}\cdot\mathbf{x}}(n/2) = 2nk$.
- Diagonal blocks are processed recursively at a **cost**

$$c_{A\cdot\mathbf{x}}(n) = 2c_{A\cdot\mathbf{x}}(n/2) + 4kn + n.$$

Master theorem (*divide and conquer*): $c_{A\cdot\mathbf{x}}(n) = (4k+1)\log_2(n)n.$

BLAS with HODLR format

$C = A + B$: Adding two equally partitioned HODLR matrices **increases the ranks** of off-diagonal blocks by a factor 2.

BLAS with HODLR format

$C = A + B$: Adding two equally partitioned HODLR matrices **increases the ranks** of off-diagonal blocks by a factor 2.

- ⚙ We need truncation $\mathfrak{T}_k(A(I_1^\ell, I_j^\ell) + B(I_1^\ell, I_j^\ell))$, costs

$$c_{\mathrm{LR+LR}} = c_{\mathrm{SVD}} \times (nk^2 + k^3),$$

  where c_{SVD} is the cost of the given low-rank truncation algorithm (SVD, rand-SVD, QR, ...)

BLAS with HODLR format

$C = A + B$: Adding two equally partitioned HODLR matrices **increases the ranks** of off-diagonal blocks by a factor 2.

- We need truncation $\mathfrak{T}_k(A(I_1^\ell, I_j^\ell) + B(I_1^\ell, I_j^\ell))$, costs

$$c_{\mathrm{LR+LR}} = c_{\mathrm{SVD}} \times (nk^2 + k^3),$$

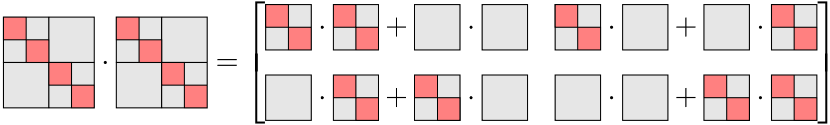
where c_{SVD} is the cost of the given low-rank truncation algorithm (SVD, rand-SVD, QR, ...)

Total cost is then:

$$\begin{aligned}\sum_{\ell=1}^{p} 2^\ell c_{\mathrm{LR+LR}}(m_\ell) =& c_{\mathrm{SVD}} \sum_{\ell=1}^{p} 2^\ell (k^3 + m_\ell k^2) \\ \leq& c_{\mathrm{SVD}} \left(2^{p+1} k^3 + \sum_{\ell=1}^{p} 2^\ell 2^{p-\ell} n_0 k^2 \right) \\ \leq& c_{\mathrm{SVD}} \left(2nk^3 + n \log_2(n) k^2 \right).\end{aligned}$$

BLAS with HODLR format

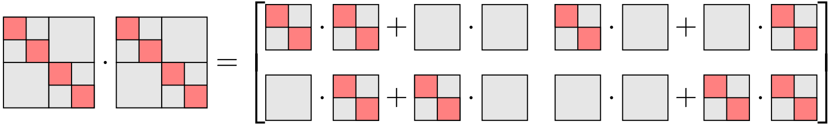
$C = AB$: Matrix-matrix multiplication can also be done recursively



where ▚ is a $n/2 \times n/2$ HODLR matrix and □ is a low-rank block.

BLAS with HODLR format

$C = AB$: Matrix-matrix multiplication can also be done recursively

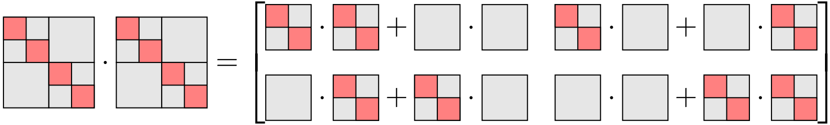


where ▦ is a $n/2 \times n/2$ HODLR matrix and □ is a low-rank block.

1. ▦ · ▦ · of 2 HODLR $n/2$ matrices,

BLAS with HODLR format

$C = AB$: Matrix-matrix multiplication can also be done recursively

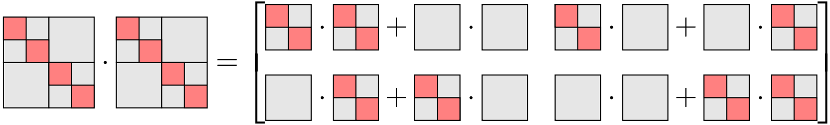


where ⊞ is a $n/2 \times n/2$ HODLR matrix and □ is a low-rank block.

1. ⊞ · ⊞ · of 2 HODLR $n/2$ matrices,
2. □ · □ · of 2 low-rank blocks,

BLAS with HODLR format

$C = AB$: Matrix-matrix multiplication can also be done recursively

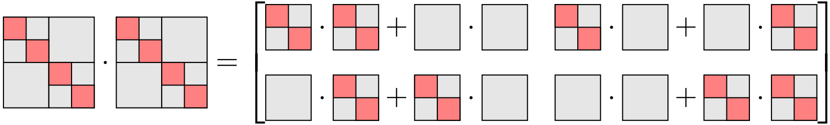


where ⊞ is a $n/2 \times n/2$ HODLR matrix and □ is a low-rank block.

1. ⊞ · ⊞ · of 2 HODLR $n/2$ matrices,
2. □ · □ · of 2 low-rank blocks,
3. ⊞ · □ · of HODLR times low-rank,

BLAS with HODLR format

$C = AB$: Matrix-matrix multiplication can also be done recursively

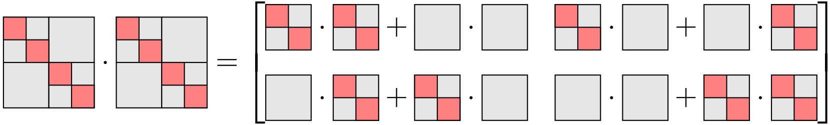


where ⊞ is a $n/2 \times n/2$ HODLR matrix and □ is a low-rank block.

1. ⊞ · ⊞ · of 2 HODLR $n/2$ matrices,
2. □ · □ · of 2 low-rank blocks,
3. ⊞ · □ · of HODLR times low-rank,
4. □ · ⊞ · of low-rank times HODLR.

BLAS with HODLR format

$C = AB$: Matrix-matrix multiplication can also be done recursively



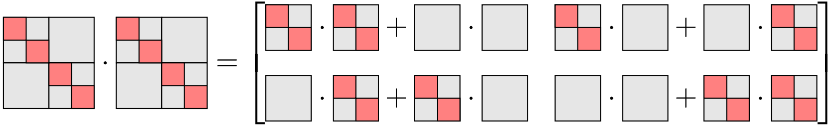
where ⊞ is a $^{n}/_{2} \times {}^{n}/_{2}$ HODLR matrix and □ is a low-rank block.

1. ⊞ · ⊞ · of 2 HODLR $^{n}/_{2}$ matrices,
2. □ · □ · of 2 low-rank blocks,
3. ⊞ · □ · of HODLR times low-rank,
4. □ · ⊞ · of low-rank times HODLR.

$$\begin{aligned} c_{H\cdot H}(n) = & 2\left(c_{H\cdot H}(^{n}/_{2}) + c_{\mathrm{LR}\cdot \mathrm{LR}}(^{n}/_{2}) + c_{H\cdot \mathrm{LR}}(^{n}/_{2}) + c_{\mathrm{LR}\cdot H}(^{n}/_{2})\right. \\ & \left. + c_{H+\mathrm{LR}}(^{n}/_{2}) + c_{\mathrm{LR}+\mathrm{LR}}(^{n}/_{2})\right) \end{aligned}$$

BLAS with HODLR format

$C = AB$: Matrix-matrix multiplication can also be done recursively



where ⊞ is a $n/2 \times n/2$ HODLR matrix and □ is a low-rank block.

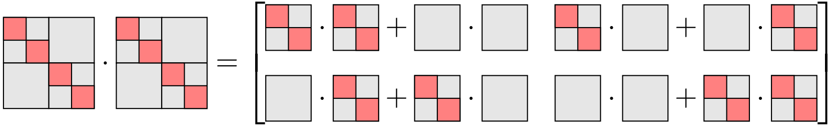
1. ⊞ · ⊞ · of 2 HODLR $n/2$ matrices,
2. □ · □ · of 2 low-rank blocks,
3. ⊞ · □ · of HODLR times low-rank,
4. □ · ⊞ · of low-rank times HODLR.

$$c_{H\cdot H}(n) = 2\left(c_{H\cdot H}(n/2) + c_{\mathrm{LR}\cdot\mathrm{LR}}(n/2) + c_{H\cdot\mathrm{LR}}(n/2) + c_{\mathrm{LR}\cdot H}(n/2) + c_{H+\mathrm{LR}}(n/2) + c_{\mathrm{LR}+\mathrm{LR}}(n/2)\right)$$

$c_{\mathrm{LR}\cdot\mathrm{LR}}(n) = 4nk^2$

BLAS with HODLR format

$C = AB$: Matrix-matrix multiplication can also be done recursively



where ⊞ is a $n/2 \times n/2$ HODLR matrix and □ is a low-rank block.

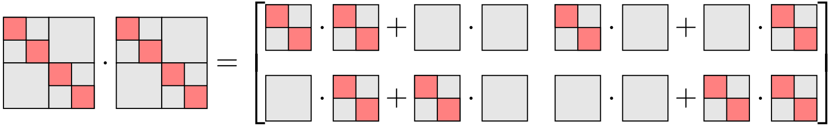
1. ⊞ · ⊞ · of 2 HODLR $n/2$ matrices,
2. □ · □ · of 2 low-rank blocks,
3. ⊞ · □ · of HODLR times low-rank,
4. □ · ⊞ · of low-rank times HODLR.

$$c_{H\cdot H}(n) = 2\left(c_{H\cdot H}(n/2) + c_{\mathrm{LR}\cdot\mathrm{LR}}(n/2) + {\color{red}c_{H\cdot\mathrm{LR}}(n/2)} + {\color{red}c_{\mathrm{LR}\cdot H}(n/2)}\right.$$
$$\left.+{\color{red}c_{H+\mathrm{LR}}(n/2)} + c_{\mathrm{LR}+\mathrm{LR}}(n/2)\right)$$

$$c_{H\cdot\mathrm{LR}}(n) = c_{\mathrm{LR}\cdot H} = k c_{H\mathbf{v}}(n) = k(4k+1)\log_2(n)n$$

BLAS with HODLR format

$C = AB$: Matrix-matrix multiplication can also be done recursively



where ⊞ is a $n/2 \times n/2$ HODLR matrix and □ is a low-rank block.

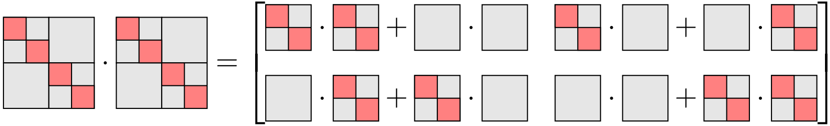
1. ⊞ · ⊞ · of 2 HODLR $n/2$ matrices,
2. □ · □ · of 2 low-rank blocks,
3. ⊞ · □ · of HODLR times low-rank,
4. □ · ⊞ · of low-rank times HODLR.

$$\begin{aligned} c_{H\cdot H}(n) =& 2\left(c_{H\cdot H}(n/2) + c_{\mathrm{LR}\cdot \mathrm{LR}}(n/2) + c_{H\cdot \mathrm{LR}}(n/2) + c_{\mathrm{LR}\cdot H}(n/2)\right. \\ & \left. + c_{H+\mathrm{LR}}(n/2) + c_{\mathrm{LR}+\mathrm{LR}}(n/2)\right) \end{aligned}$$

$$c_{H+\mathrm{LR}}(n) = c_{H+H}(n) = c_{\mathrm{SVD}}(nk^3 + n\log(n)k^2)$$

BLAS with HODLR format

$C = AB$: Matrix-matrix multiplication can also be done recursively



where ⊞ is a ${}^{n}\!/_{2} \times {}^{n}\!/_{2}$ HODLR matrix and □ is a low-rank block.

1. ⊞ $\cdot$ ⊞ $\cdot$ of 2 HODLR ${}^{n}\!/_{2}$ matrices,
2. □ $\cdot$ □ $\cdot$ of 2 low-rank blocks,
3. ⊞ $\cdot$ □ $\cdot$ of HODLR times low-rank,
4. □ $\cdot$ ⊞ $\cdot$ of low-rank times HODLR.

Total cost $c_{H \cdot H}(n) \in O(k^3 n \log n + k^2 n \log^2 n)$.

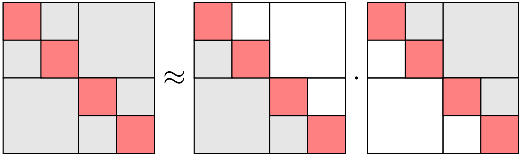
BLAS with HODLR format

Approximate solution of a linear system $A\mathbf{x} = \mathbf{b}$ with HODLR matrix A:

BLAS with HODLR format

Approximate solution of a linear system $A\mathbf{x} = \mathbf{b}$ with HODLR matrix A:

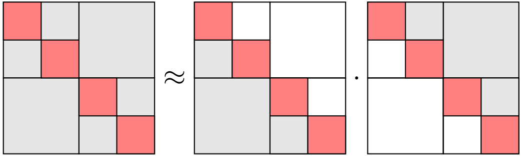
$A \approx LU$ Approximate LU-factorization $A \approx LU$ in HODLR format:



BLAS with HODLR format

Approximate solution of a linear system $A\mathbf{x} = \mathbf{b}$ with HODLR matrix A:

$A \approx LU$ Approximate LU-factorization $A \approx LU$ in HODLR format:



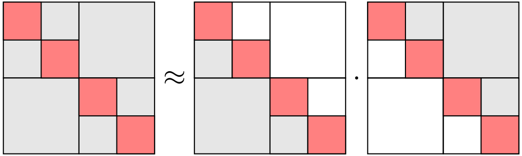
Forward substitution to solve $L\mathbf{y} = \mathbf{b}$,

Backward substitution to solve $U\mathbf{x} = \mathbf{y}$.

BLAS with HODLR format

Approximate solution of a linear system $A\mathbf{x} = \mathbf{b}$ with HODLR matrix A:

$A \approx LU$ Approximate LU-factorization $A \approx LU$ in HODLR format:



Forward substitution to solve $L\mathbf{y} = \mathbf{b}$,

Backward substitution to solve $U\mathbf{x} = \mathbf{y}$.

We need to analyze the two steps separately.

BLAS with HODLR format

Approximate LU factorization, on level $\ell = 1$:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad U = \begin{bmatrix} U_{11} & U_{12} \\ O & U_{22} \end{bmatrix}$$

It is done in four steps

BLAS with HODLR format

Approximate LU factorization, on level $\ell = 1$:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad U = \begin{bmatrix} U_{11} & U_{12} \\ O & U_{22} \end{bmatrix}$$

It is done in four steps

1. Compute LU factors L_{11}, U_{11} of A_{11},

BLAS with HODLR format

Approximate LU factorization, on level $\ell = 1$:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad U = \begin{bmatrix} U_{11} & U_{12} \\ O & U_{22} \end{bmatrix}$$

It is done in four steps

1. Compute LU factors L_{11}, U_{11} of A_{11},
2. Compute $U_{12} = L_{11}^{-1} A_{12}$ by **forward substitution**,

BLAS with HODLR format

Approximate LU factorization, on level $\ell = 1$:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad U = \begin{bmatrix} U_{11} & U_{12} \\ O & U_{22} \end{bmatrix}$$

It is done in four steps

1. Compute LU factors L_{11}, U_{11} of A_{11},
2. Compute $U_{12} = L_{11}^{-1} A_{12}$ by **forward substitution**,
3. Compute $L_{21} = A_{21} U_{11}^{-1}$ by **backward substitution**,

BLAS with HODLR format

Approximate LU factorization, on level $\ell = 1$:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad U = \begin{bmatrix} U_{11} & U_{12} \\ O & U_{22} \end{bmatrix}$$

It is done in four steps

1. Compute LU factors L_{11}, U_{11} of A_{11},
2. Compute $U_{12} = L_{11}^{-1} A_{12}$ by **forward substitution**,
3. Compute $L_{21} = A_{21} U_{11}^{-1}$ by **backward substitution**,
4. Compute LU factors L_{22}, U_{22} of $A_{22} - L_{21} U_{12}$.

BLAS with HODLR format

Approximate LU factorization, on level $\ell = 1$:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad U = \begin{bmatrix} U_{11} & U_{12} \\ O & U_{22} \end{bmatrix}$$

It is done in four steps

1. Compute LU factors L_{11}, U_{11} of A_{11},
2. Compute $U_{12} = L_{11}^{-1} A_{12}$ by **forward substitution**,
3. Compute $L_{21} = A_{21} U_{11}^{-1}$ by **backward substitution**,
4. Compute LU factors L_{22}, U_{22} of $A_{22} - L_{21} U_{12}$.

The analysis of the cost is *analogous to the matrix-matrix multiplication case*, **but** we need to know how to do and how-much does forward/backward substitution costs.

BLAS with HODLR format

Forward substitution with lower triangular L in HODLR format: $\mathbf{y} = L^{-1}\mathbf{b}$

$$L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

with L_{21} low-rank, and L_{11}, L_{22} HODLR.

BLAS with HODLR format

Forward substitution with lower triangular L in HODLR format: $\mathbf{y} = L^{-1}\mathbf{b}$

$$L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

with L_{21} low-rank, and L_{11}, L_{22} HODLR.

1. Solve $L_{11}\mathbf{y}_1 = \mathbf{b}_1$,

BLAS with HODLR format

Forward substitution with lower triangular L in HODLR format: $\mathbf{y} = L^{-1}\mathbf{b}$

$$L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

with L_{21} low-rank, and L_{11}, L_{22} HODLR.

1. Solve $L_{11}\mathbf{y}_1 = \mathbf{b}_1$,
2. Compute $\tilde{\mathbf{b}}_2 = \mathbf{b}_2 - L_{21}\mathbf{y}_1$,

BLAS with HODLR format

Forward substitution with lower triangular L in HODLR format: $\mathbf{y} = L^{-1}\mathbf{b}$

$$L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

with L_{21} low-rank, and L_{11}, L_{22} HODLR.

1. Solve $L_{11}\mathbf{y}_1 = \mathbf{b}_1$,
2. Compute $\tilde{\mathbf{b}}_2 = \mathbf{b}_2 - L_{21}\mathbf{y}_1$,
3. Solve $L_{22}\mathbf{y}_2 = \tilde{\mathbf{b}}_2$.

BLAS with HODLR format

Forward substitution with lower triangular L in HODLR format: $\mathbf{y} = L^{-1}\mathbf{b}$

$$L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

with L_{21} low-rank, and L_{11}, L_{22} HODLR.

1. Solve $L_{11}\mathbf{y}_1 = \mathbf{b}_1$,
2. Compute $\tilde{\mathbf{b}}_2 = \mathbf{b}_2 - L_{21}\mathbf{y}_1$,
3. Solve $L_{22}\mathbf{y}_2 = \tilde{\mathbf{b}}_2$.

Cost recursively:

$$c_{\text{forw}} = 2c_{\text{forw}}(n/2) + (2k+1)n.$$

BLAS with HODLR format

Forward substitution with lower triangular L in HODLR format: $\mathbf{y} = L^{-1}\mathbf{b}$

$$L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

with L_{21} low-rank, and L_{11}, L_{22} HODLR.

1. Solve $L_{11}\mathbf{y}_1 = \mathbf{b}_1$,
2. Compute $\tilde{\mathbf{b}}_2 = \mathbf{b}_2 - L_{21}\mathbf{y}_1$,
3. Solve $L_{22}\mathbf{y}_2 = \tilde{\mathbf{b}}_2$.

Cost recursively:

$$c_{\text{forw}} = 2c_{\text{forw}}(n/2) + (2k+1)n.$$

On level $\ell = p$, we have the direct solution of $2^p = n/n_0$ linear systems of size $n_0 \times n_0$.

BLAS with HODLR format

Forward substitution with lower triangular L in HODLR format: $\mathbf{y} = L^{-1}\mathbf{b}$

$$L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

with L_{21} low-rank, and L_{11}, L_{22} HODLR.

1. Solve $L_{11}\mathbf{y}_1 = \mathbf{b}_1$,
2. Compute $\tilde{\mathbf{b}}_2 = \mathbf{b}_2 - L_{21}\mathbf{y}_1$,
3. Solve $L_{22}\mathbf{y}_2 = \tilde{\mathbf{b}}_2$.

Cost recursively:

$$c_{\mathrm{forw}} = 2c_{\mathrm{forw}}({}^{n}\!/_{2}) + (2k+1)n.$$

On level $\ell = p$, we have the direct solution of $2^p = {}^{n}\!/_{n_0}$ linear systems of size $n_0 \times n_0$.

Total cost $c_{\mathrm{forw}} \in O(kn\log(n))$, and analogously for backward substitution.

BLAS with HODLR format

Forward substitution with lower triangular L in HODLR format: $\mathbf{y} = L^{-1}\mathbf{b}$

$$L = \begin{bmatrix} L_{11} & O \\ L_{21} & L_{22} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

with L_{21} low-rank, and L_{11}, L_{22} HODLR.

1. Solve $L_{11}\mathbf{y}_1 = \mathbf{b}_1$,
2. Compute $\tilde{\mathbf{b}}_2 = \mathbf{b}_2 - L_{21}\mathbf{y}_1$,
3. Solve $L_{22}\mathbf{y}_2 = \tilde{\mathbf{b}}_2$.

Cost recursively:

$$c_{\text{forw}} = 2c_{\text{forw}}(n/2) + (2k+1)n.$$

On level $\ell = p$, we have the direct solution of $2^p = n/n_0$ linear systems of size $n_0 \times n_0$.

- Total cost $c_{\text{forw}} \in O(kn\log(n))$, and analogously for backward substitution.
- Total cost $c_{\text{LU}}(n) \lesssim c_{H\cdot H}(n) \in O(k^3 n \log n + k^2 n \log^2 n)$.

BLAS with HODLR format

The `hm-toolbox` (Massei, Robol, and Kressner 2020) contains all the routines.

- They *overload* the standard MATLAB operation by the same name, i.e., if you have variables in the right class you operate directly in this format.
- One can use different **cluster tree** $\mathcal{T}_p$ to get smaller ranks. They are determined by the partitioning of the index set on the leaf level and represented as the vector $\mathbf{c} = [n_1^{(p)}, \dots, n_{2^p}^{(p)}]$, change it to change the HODLR matrix.

Operation	HODLR complexity
`A*v`	$\mathcal{O}(kn\log n)$
`A\v`	$\mathcal{O}(k^2 n\log^2 n)$
`A+B`	$\mathcal{O}(k^2 n\log n)$
`A*B`	$\mathcal{O}(k^2 n\log^2 n)$
`A\B`	$\mathcal{O}(k^2 n\log^2 n)$
`inv(A)`	$\mathcal{O}(k^2 n\log^2 n)$
`A.*B`[2]	$\mathcal{O}(k^4 n\log n)$
`lu(A)`, `chol(A)`	$\mathcal{O}(k^2 n\log^2 n)$
`qr(A)`	$\mathcal{O}(k^2 n\log^2 n)$
compression	$\mathcal{O}(k^2 n\log(n))$

---

[2] The complexity of the Hadamard product is dominated by the recompression stage due to the k^2 HODLR rank of $A \circ B$. Without recompression the cost is $\mathcal{O}(k^2 n\log n)$.

HODLR solver for the 1D case

We can modify our first example to get a solution for the 1D problem in the new format.

```
%% Discretization
N = 2^7; hN = 1/(N-1);  x = 0:hN:1; dt = hN;
alpha = 1.5; % Coefficients
dplus=@(x)gamma(3-alpha).*x.^alpha;
dminus=@(x)gamma(3-alpha).*(1-x).^alpha;
w = @(x) 5*x.*(1-x);
tol = 1e-9; % HODLR building
tic;
G = glhodlrmatrix(N,alpha,tol);
Dplus = hodlr('diagonal',dplus(x));
Dminus = hodlr('diagonal',dminus(x));
I = hodlr('eye', N);
nu = hN^alpha/dt;
A = nu*I -(Dplus*G + Dminus*G');
buildtime = toc;
```

```
%% Solving
[L,U] = lu(A);
flu = @() lu(A);
timelu = timeit(flu,2);
w = w(x).';
solvetime = 0;
for i=1:N
 tic;
 w = U\(L\(nu*w));
 solvetime = solvetime + toc;
end
solvetime = solvetime/N;
```

HODLR solver for the 1D case

We can modify our first example to get a solution for the 1D problem in the new format.

```
%% Discretization
N = 2^7; hN = 1/(N-1);  x = 0:hN:1; dt = hN;
alpha = 1.5; % Coefficients
dplus=@(x)gamma(3-alpha).*x.^alpha;
dminus=@(x)gamma(3-alpha).*(1-x).^alpha;
w = @(x) 5*x.*(1-x);
tol = 1e-9; % HODLR building
tic;
G = glhodlrmatrix(N,alpha,tol);
Dplus = hodlr('diagonal',dplus(x));
Dminus = hodlr('diagonal',dminus(x));
I = hodlr('eye', N);
nu = hN^alpha/dt;
A = nu*I -(Dplus*G + Dminus*G');
buildtime = toc;
```

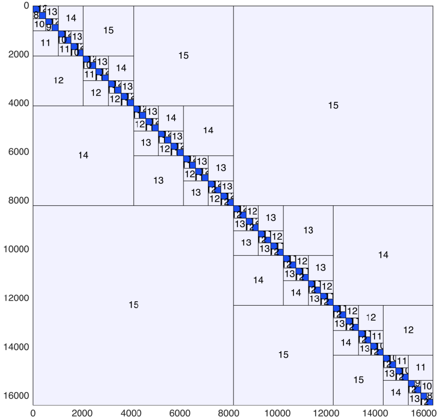
```
%% Solving
[L,U] = lu(A);
flu = @() lu(A);
timelu = timeit(flu,2);
w = w(x).';
solvetime = 0;
for i=1:N
 tic;
 w = U\(L\(nu*w));
 solvetime = solvetime + toc;
end
solvetime = solvetime/N;
```

Let us try looking at the timings.

HODLR solver for the 1D case

We take $\alpha = 1.5$, and $\varepsilon = 10^{-9}$

N	Build (s)	LU (s)	Avg. Solve (s)
2^7	8.96e-03	1.44e-04	2.93e-04
2^8	1.35e-02	4.63e-04	3.33e-04
2^9	3.14e-02	2.05e-03	5.41e-04
2^{10}	7.28e-02	6.21e-03	9.35e-04
2^{11}	1.59e-01	1.63e-02	1.75e-03
2^{12}	3.85e-01	4.33e-02	3.68e-03
2^{13}	8.81e-01	1.27e-01	7.99e-03
2^{14}	2.19e+00	3.73e-01	1.55e-02

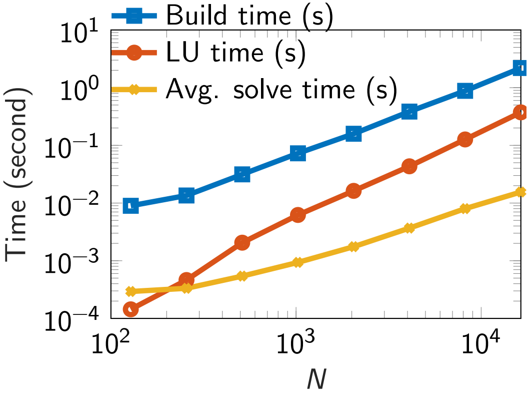


Largest matrix occupies 46.25 Mb, against the 2 Gb of the dense storage and the 0.87 Mb of storing three diagonals and $2 \times (2N - 1)$ for the Toeplitz storage.

HODLR solver for the 1D case

We take $\alpha = 1.5$, and $\varepsilon = 10^{-9}$

N	Build (s)	LU (s)	Avg. Solve (s)
2^7	8.96e-03	1.44e-04	2.93e-04
2^8	1.35e-02	4.63e-04	3.33e-04
2^9	3.14e-02	2.05e-03	5.41e-04
2^{10}	7.28e-02	6.21e-03	9.35e-04
2^{11}	1.59e-01	1.63e-02	1.75e-03
2^{12}	3.85e-01	4.33e-02	3.68e-03
2^{13}	8.81e-01	1.27e-01	7.99e-03
2^{14}	2.19e+00	3.73e-01	1.55e-02

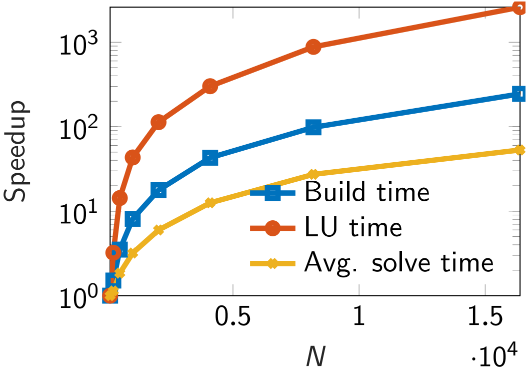


Largest matrix occupies 46.25 Mb, against the 2 Gb of the dense storage and the 0.87 Mb of storing three diagonals and $2 \times (2N - 1)$ for the Toeplitz storage.

HODLR solver for the 1D case

We take $\alpha = 1.5$, and $\varepsilon = 10^{-9}$

N	Build (s)	LU (s)	Avg. Solve (s)
2^7	8.96e-03	1.44e-04	2.93e-04
2^8	1.35e-02	4.63e-04	3.33e-04
2^9	3.14e-02	2.05e-03	5.41e-04
2^{10}	7.28e-02	6.21e-03	9.35e-04
2^{11}	1.59e-01	1.63e-02	1.75e-03
2^{12}	3.85e-01	4.33e-02	3.68e-03
2^{13}	8.81e-01	1.27e-01	7.99e-03
2^{14}	2.19e+00	3.73e-01	1.55e-02



Largest matrix occupies 46.25 Mb, against the 2 Gb of the dense storage and the 0.87 Mb of storing three diagonals and $2 \times (2N - 1)$ for the Toeplitz storage.

Back to Sylvester (Massei, Palitta, and Robol 2018)

To solve the Sylvester equation with HODLR coefficients

$$AX + XB^T = C, \qquad A \in \mathbb{R}^{n\times n},\ \ B \in \mathbb{R}^{m\times m},\ \ X, C \in \mathbb{R}^{n\times m},$$

we can use the integral formulation

$$X = \int_0^{+\infty} e^{-At} C e^{-B^T t}\, \mathrm{d}t.$$

Back to Sylvester (Massei, Palitta, and Robol 2018)

To solve the Sylvester equation with HODLR coefficients

$$AX + XB^T = C, \qquad A \in \mathbb{R}^{n\times n},\ \ B \in \mathbb{R}^{m\times m},\ \ X, C \in \mathbb{R}^{n\times m},$$

we can use the integral formulation

$$X = \int_0^{+\infty} e^{-At} C e^{-B^T t}\,\mathrm{d}t.$$

We perform the *change of variables*: $t = f(\theta) \triangleq L \cdot \cot\left(\frac{\theta}{2}\right)^2$, rewriting the integral as

$$X = 2L \int_0^{\pi} \frac{\sin(\theta)}{(1-\cos(\theta))^2}\ e^{-Af(\theta)} C e^{-B^T f(\theta)}\,\mathrm{d}\theta,$$

with L a parameter to be optimized for convergence.

Back to Sylvester (Massei, Palitta, and Robol 2018)

We now have an integral on a finite domain $\Rightarrow$ **Gauss-Legendre quadrature**

$$X \approx \sum_{j=1}^{m} \omega_j \cdot e^{-Af(\theta_j)} C e^{-B^T f(\theta_j)},$$

for $\{\theta_j, w_j\}_{j=1}^{m}$ are the Legendre points and weights, and $\omega_j = 2Lw_j \cdot \frac{\sin(\theta_j)}{(1-\cos(\theta_j))^2}$.

Back to Sylvester (Massei, Palitta, and Robol 2018)

We now have an integral on a finite domain ⇒ **Gauss-Legendre quadrature**

$$X \approx \sum_{j=1}^{m} \omega_j \cdot e^{-Af(\theta_j)} C e^{-B^T f(\theta_j)},$$

for $\{\theta_j, w_j\}_{j=1}^m$ are the Legendre points and weights, and $\omega_j = 2Lw_j \cdot \frac{\sin(\theta_j)}{(1-\cos(\theta_j))^2}$.

❓ The **dominant cost** is now computing $e^{-Af(\theta_j)}$ and $e^{-B^T f(\theta_j)}$, how do we do it?

Back to Sylvester (Massei, Palitta, and Robol 2018)

We now have an integral on a finite domain ⇒ **Gauss-Legendre quadrature**

$$X \approx \sum_{j=1}^{m} \omega_j \cdot e^{-Af(\theta_j)} C e^{-B^T f(\theta_j)},$$

for $\{\theta_j, w_j\}_{j=1}^m$ are the Legendre points and weights, and $\omega_j = 2Lw_j \cdot \frac{\sin(\theta_j)}{(1-\cos(\theta_j))^2}$.

❓ The **dominant cost** is now computing $e^{-Af(\theta_j)}$ and $e^{-B^T f(\theta_j)}$, how do we do it?

</> A **good idea** could be using *rational approximation* to $\exp(t)$

Back to Sylvester (Massei, Palitta, and Robol 2018)

We now have an integral on a finite domain ⇒ **Gauss-Legendre quadrature**

$$X \approx \sum_{j=1}^{m} \omega_j \cdot e^{-Af(\theta_j)} C e^{-B^T f(\theta_j)},$$

for $\{\theta_j, w_j\}_{j=1}^{m}$ are the Legendre points and weights, and $\omega_j = 2Lw_j \cdot \frac{\sin(\theta_j)}{(1-\cos(\theta_j))^2}$.

- The **dominant cost** is now computing $e^{-Af(\theta_j)}$ and $e^{-B^T f(\theta_j)}$, how do we do it?
- A **good idea** could be using *rational approximation* to $\exp(t)$
- (d, d)-Padé with *scaling and squaring* $e^A = (e^{2^{-k}A})^{2^k}$ and $k = \lceil \log_2 \|A\|_2 \rceil$.

Back to Sylvester (Massei, Palitta, and Robol 2018)

We now have an integral on a finite domain ⇒ **Gauss-Legendre quadrature**

$$X \approx \sum_{j=1}^{m} \omega_j \cdot e^{-Af(\theta_j)} C e^{-B^T f(\theta_j)},$$

for $\{\theta_j, w_j\}_{j=1}^m$ are the Legendre points and weights, and $\omega_j = 2Lw_j \cdot \frac{\sin(\theta_j)}{(1-\cos(\theta_j))^2}$.

❓ The **dominant cost** is now computing $e^{-Af(\theta_j)}$ and $e^{-B^T f(\theta_j)}$, how do we do it?

</> A **good idea** could be using *rational approximation* to $\exp(t)$

🚪 (d,d)-Padé with *scaling and squaring* $e^A = (e^{2^{-k}A})^{2^k}$ and $k = \lceil \log_2 \|A\|_2 \rceil$.

🚪 Rational Chebyshev function (Popolizio and Simoncini 2008):

$$e^x \approx \frac{r_1}{x - s_1} + \ldots + \frac{r_d}{x - s_d}.$$

requiring d inversions and additions that is uniformly accurate for every positive value of t, and thus is better in the case in which $\|A\|_2$ is large.

Back to Sylvester (Massei, Palitta, and Robol 2018)

```
Input: lyap_integral
$A, B, C, m$;
/* Solves $AX + XB^T = C$ with $m$
   integration points                */
$L \leftarrow 100$ ;       /* Should be tuned for
 accuracy!  */
$[w, \theta] \leftarrow \mathrm{GaussLegendrePts}\, m$ ;
 /* Integration points and weights
 on $[0, \pi]$ */
$X \leftarrow 0_{n \times n}$;
for $i = 1, \dots, m$ do
    $f \leftarrow L \cdot \cot(\frac{\theta_i}{2})^2$;
    $X \leftarrow X + w_i \frac{\sin(\theta_i)}{(1-\cos\theta_i)^2} \cdot \mathrm{expm}\,(-f \cdot A) \cdot$
      $C \cdot \mathrm{expm}\,(-f \cdot B^T)$;
end
$X \leftarrow 2L \cdot X$;
```

Back to Sylvester (Massei, Palitta, and Robol 2018)

```
Input: lyap_integral
$A, B, C, m$;
/* Solves $AX + XB^T = C$ with $m$
   integration points                    */
$L \leftarrow 100$ ;        /* Should be tuned for
 accuracy!  */
$[w, \theta] \leftarrow \mathrm{GaussLegendrePts}\, m$ ;
 /* Integration points and weights
 on $[0, \pi]$ */
$X \leftarrow 0_{n\times n}$;
for $i = 1, \dots, m$ do
    $f \leftarrow L \cdot \cot(\frac{\theta_i}{2})^2$;
    $X \leftarrow X + w_i \frac{\sin(\theta_i)}{(1-\cos\theta_i)^2} \cdot \mathrm{expm}(-f \cdot A) \cdot$
      $C \cdot \mathrm{expm}(-f \cdot B^T)$;
end
$X \leftarrow 2L \cdot X$;
```

Mixed structures

If the right-hand side C is low-rank, and the structure in the matrices A and B is HODLR, thus permitting to perform fast matrix vector multiplications and system solutions; then we can apply the *extended Krylov subspace method* we had already seen.

Back to Sylvester (Massei, Palitta, and Robol 2018)

```
Input: lyap_integral
$A, B, C, m$;
/* Solves $AX + XB^T = C$ with $m$ integration points */
$L \leftarrow 100$ ;          /* Should be tuned for accuracy! */
$[w, \theta] \leftarrow \mathrm{GaussLegendrePts}\, m$ ;
    /* Integration points and weights on $[0, \pi]$ */
$X \leftarrow 0_{n \times n}$;
for $i = 1, \ldots, m$ do
    $f \leftarrow L \cdot \cot(\frac{\theta_i}{2})^2$;
    $X \leftarrow X + w_i \frac{\sin(\theta_i)}{(1-\cos\theta_i)^2} \cdot \mathrm{expm}\,(-f \cdot A) \cdot C \cdot \mathrm{expm}\,(-f \cdot B^T)$;
end
$X \leftarrow 2L \cdot X$;
```

Mixed structures

If the right-hand side C is low-rank, and the structure in the matrices A and B is HODLR, thus permitting to perform fast matrix vector multiplications and system solutions; then we can apply the *extended Krylov subspace method* we had already seen.

Build

$$\mathbb{EK}_s(A, U) = \mathrm{span}\{U, A^{-1}U, AU, \ldots\}$$

$$\mathbb{EK}_s(B^T, V) = \mathrm{span}\{V, B^{-T}V, B^TV, \ldots\},$$

Back to Sylvester (Massei, Palitta, and Robol 2018)

```
Input: lyap_integral
$A, B, C, m$;
/* Solves $AX + XB^T = C$ with $m$ integration points */
$L \leftarrow 100$ ;   /* Should be tuned for accuracy! */
$[w, \theta] \leftarrow \mathrm{GaussLegendrePts}\, m$ ;
/* Integration points and weights on $[0, \pi]$ */
$X \leftarrow 0_{n\times n}$;
for $i = 1, \dots, m$ do
    $f \leftarrow L \cdot \cot(\frac{\theta_i}{2})^2$;
    $X \leftarrow X + w_i \frac{\sin(\theta_i)}{(1-\cos\theta_i)^2} \cdot \mathrm{expm}\,(-f \cdot A) \cdot C \cdot \mathrm{expm}\,(-f \cdot B^T)$;
end
$X \leftarrow 2L \cdot X$;
```

Mixed structures

If the right-hand side C is low-rank, and the structure in the matrices A and B is HODLR, thus permitting to perform fast matrix vector multiplications and system solutions; then we can apply the *extended Krylov subspace method* we had already seen.

Build $\mathbb{EK}_s(A, U)$, $\mathbb{EK}_s(B^T, V)$, project on $\tilde{A}_s = U_s^* A U_s$, $\tilde{B}_s = V_s^* B V_s$, $\tilde{U} = U_s^* U$, and $\tilde{V} = V_s^* V$.

Back to Sylvester (Massei, Palitta, and Robol 2018)

```
Input: lyap_integral
$A, B, C, m$;
/* Solves $AX + XB^T = C$ with $m$
   integration points                */
$L \leftarrow 100$ ;       /* Should be tuned for
 accuracy!  */
$[w, \theta] \leftarrow \mathrm{GaussLegendrePts}\, m$ ;
 /* Integration points and weights
 on $[0, \pi]$ */
$X \leftarrow 0_{n \times n}$;
for $i = 1, \ldots, m$ do
    $f \leftarrow L \cdot \cot(\frac{\theta_i}{2})^2$;
    $X \leftarrow X + w_i \frac{\sin(\theta_i)}{(1-\cos\theta_i)^2} \cdot \mathrm{expm}(-f \cdot A) \cdot$
      $C \cdot \mathrm{expm}(-f \cdot B^T)$;
end
$X \leftarrow 2L \cdot X$;
```

Mixed structures

If the right-hand side C is low-rank, and the structure in the matrices A and B is HODLR, thus permitting to perform fast matrix vector multiplications and system solutions; then we can apply the *extended Krylov subspace method* we had already seen.

Build $\mathbb{EK}_s(A, U)$, $\mathbb{EK}_s(B^T, V)$, project on $\tilde{A}_s = U_s^* A U_s$, $\tilde{B}_s = V_s^* B V_s$, $\tilde{U} = U_s^* U$, and $\tilde{V} = V_s^* V$. Solve $\tilde{A}_s X_s + X_s \tilde{B}_s = \tilde{U}\tilde{V}^T$ with **dense arithmetic**.

Back to Sylvester (Massei, Palitta, and Robol 2018)

```
Input: lyap_integral
$A, B, C, m$;
/* Solves $AX + XB^T = C$ with $m$
   integration points                          */
$L \leftarrow 100$ ;          /* Should be tuned for
 accuracy!   */
$[w, \theta] \leftarrow \mathrm{GaussLegendrePts}\, m$ ;
 /* Integration points and weights
 on $[0, \pi]$  */
$X \leftarrow 0_{n \times n}$;
for $i = 1, \ldots, m$ do
    $f \leftarrow L \cdot \cot(\frac{\theta_i}{2})^2$;
    $X \leftarrow X + w_i \frac{\sin(\theta_i)}{(1-\cos\theta_i)^2} \cdot \mathrm{expm}\,(-f \cdot A) \cdot$
      $C \cdot \mathrm{expm}\,(-f \cdot B^T)$;
end
$X \leftarrow 2L \cdot X$;
```

Mixed structures

If the right-hand side C is low-rank, and the structure in the matrices A and B is HODLR, thus permitting to perform fast matrix vector multiplications and system solutions; then we can apply the *extended Krylov subspace method* we had already seen.

Build $\mathbb{EK}_s(A, U)$, $\mathbb{EK}_s(B^T, V)$, project on $\tilde{A}_s = U_s^* A U_s$, $\tilde{B}_s = V_s^* B V_s$, $\tilde{U} = U_s^* U$, and $\tilde{V} = V_s^* V$. Solve $\tilde{A}_s X_s + X_s \tilde{B}_s = \tilde{U}\tilde{V}^T$ with **dense arithmetic**. An approximation is $U_s X_s V_s^*$.

Back to Sylvester (Massei, Palitta, and Robol 2018)

```
Input: lyap_integral
$A, B, C, m$;
/* Solves $AX + XB^T = C$ with $m$ integration points */
$L \leftarrow 100$ ;   /* Should be tuned for accuracy! */
$[w, \theta] \leftarrow \mathrm{GaussLegendrePts}\, m$ ;   /* Integration points and weights on $[0, \pi]$ */
$X \leftarrow 0_{n \times n}$;
for $i = 1, \dots, m$ do
    $f \leftarrow L \cdot \cot(\frac{\theta_i}{2})^2$;
    $X \leftarrow X + w_i \frac{\sin(\theta_i)}{(1-\cos\theta_i)^2} \cdot \mathrm{expm}(-f \cdot A) \cdot C \cdot \mathrm{expm}(-f \cdot B^T)$;
end
$X \leftarrow 2L \cdot X$;
```

Mixed structures

If the right-hand side C is low-rank, and the structure in the matrices A and B is HODLR, thus permitting to perform fast matrix vector multiplications and system solutions; then we can apply the *extended Krylov subspace method* we had already seen.

Build $\mathbb{EK}_s(A, U)$, $\mathbb{EK}_s(B^T, V)$, project on $\tilde{A}_s = U_s^* A U_s$, $\tilde{B}_s = V_s^* B V_s$, $\tilde{U} = U_s^* U$, and $\tilde{V} = V_s^* V$. Solve $\tilde{A}_s X_s + X_s \tilde{B}_s = \tilde{U} \tilde{V}^T$ with **dense arithmetic**. An approximation is $U_s X_s V_s^*$. Another viable approach in the literature is (Kressner, Massei, and Robol 2019).

A numerical test (Massei, Mazza, and Robol 2019)

We use the usual square $[0,1]^2$, and the source f

$$f(x,y,t) = 100 \cdot (\sin(10\pi x)\cos(\pi y) + \sin(10t)\sin(\pi x) \cdot y(1-y)) \,.$$

for both **constant coefficient** $d^+ = d^- = 1$, and variable coefficients

$$\begin{aligned} d_1^+(x) &= \Gamma(1.2)(1+x)^{\alpha_1}, & d_1^-(x) &= \Gamma(1.2)(2-x)^{\alpha_1}, \\ d_2^+(y) &= \Gamma(1.2)(1+y)^{\alpha_2}, & d_2^-(y) &= \Gamma(1.2)(2-y)^{\alpha_2}. \end{aligned}$$

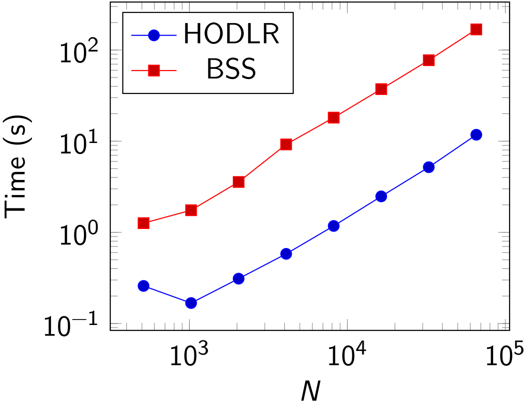
The fractional orders are $\alpha_1 = 1.3, \alpha_2 = 1.7$, and $\alpha_1 = 1.7, \alpha_2 = 1.9$. Methods are

- Sylvester by Extended-Krylov with stopping $\epsilon = 10^{-6}$ (HODLR),
  - HODLR arithmetic is set to work with a truncation of 10^{-8}.
- Sylvester by Extended-Krylov with stopping $\epsilon = 10^{-6}$ (Breiten, Simoncini, and Stoll 2016),
  - Inner solve with: GMRES with tolerance 10^{-7} and *structured preconditioners*,

A numerical test (Massei, Mazza, and Robol 2019)

Constant coefficient with $\alpha_1 = 1.3$ and $\alpha_2 = 1.7$.

N	t_{HODLR}	t_{BSS}	rank_ϵ	qsrank_ϵ
512	0.26	1.26	14	11
1,024	0.17	1.75	15	11
2,048	0.31	3.57	15	12
4,096	0.58	9.21	16	12
8,192	1.17	18.14	16	13
16,384	2.48	37.24	16	13
32,768	5.18	77.28	16	14
65,536	11.76	168.29	15	14

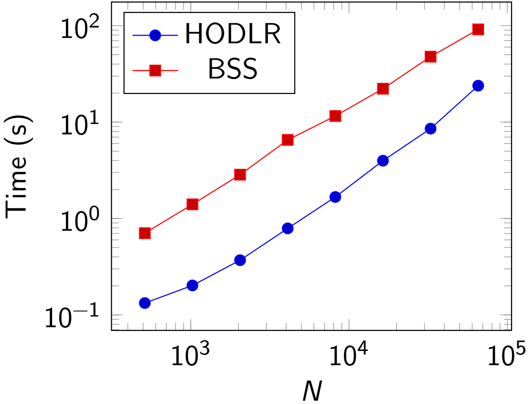


`FD_Example.m` from github.com/numpi/fme

A numerical test (Massei, Mazza, and Robol 2019)

Constant coefficient with $\alpha_1 = 1.7$ and $\alpha_2 = 1.9$.

N	t_{HODLR}	t_{BSS}	rank_ϵ	qsrank_ϵ
512	0.13	0.7	17	10
1,024	0.2	1.4	18	10
2,048	0.37	2.85	19	11
4,096	0.79	6.53	20	11
8,192	1.67	11.57	20	11
16,384	3.98	22.2	21	11
32,768	8.56	47.75	22	11
65,536	23.86	91.53	23	11

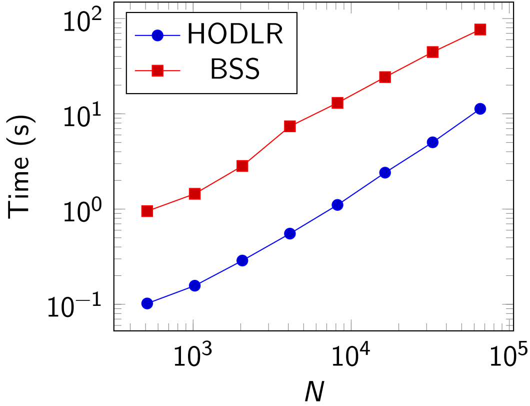


`FD_Example.m` from github.com/numpi/fme

A numerical test (Massei, Mazza, and Robol 2019)

Non-constant coefficient case with $\alpha_1 = 1.3$ and $\alpha_2 = 1.7$.

N	t_{HODLR}	t_{BSS}	rank_ϵ	qsrank_ϵ
512	0.1	0.95	14	10
1,024	0.16	1.45	14	11
2,048	0.29	2.83	15	12
4,096	0.55	7.39	16	12
8,192	1.11	13.02	16	13
16,384	2.41	24.27	16	13
32,768	5.02	44.5	16	14
65,536	11.28	76.78	16	14

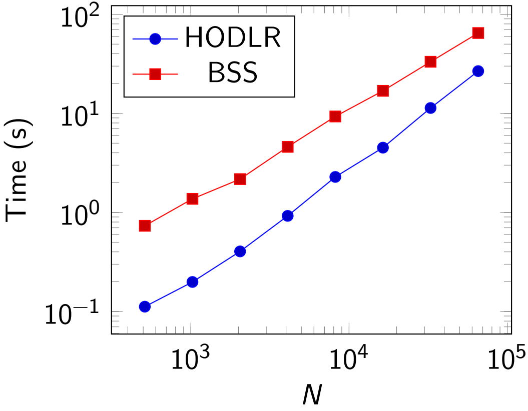


`FD_Example_vc.m` from github.com/numpi/fme

A numerical test (Massei, Mazza, and Robol 2019)

Non-constant coefficient case with $\alpha_1 = 1.7$ and $\alpha_2 = 1.9$.

N	t_{HODLR}	t_{BSS}	rank_ϵ	qsrank_ϵ
512	0.11	0.73	18	10
1,024	0.2	1.37	19	10
2,048	0.4	2.17	20	11
4,096	0.92	4.59	21	11
8,192	2.28	9.31	22	11
16,384	4.51	16.89	22	11
32,768	11.33	33.19	23	12
65,536	26.71	64.73	24	12



`FD_Example_vc.m` from github.com/numpi/fme

The tale of the matrix equation: the moral of the story.

- There is an advantage with respect to using Toeplitz-based BLAS like operations,

The tale of the matrix equation: the moral of the story.

- There is an advantage with respect to using Toeplitz-based BLAS like operations,
- In (Massei, Mazza, and Robol 2019) they are solving the case

$$\left(\frac{1}{2}I_{N_x} - \Delta t\tilde{G}_{N_x}\right)\tilde{W}^{(m+1)} + \tilde{W}^{(m+1)}\left(\frac{1}{2}I_{N_y} - \Delta t\tilde{G}_{N_y}\right)^T = \tilde{W}^{(m)} + \Delta t F^{(m+1)}, \;\; m = 0, \ldots, M-1.$$

here the spectrum is *fictitiously independent from the discretization*, i.e., all matrix-equation solvers perform a number of iteration independent from the system size: the cost is reduced to the extended Krylov subspace cost! **But** we still have time-stepping to do.

The tale of the matrix equation: the moral of the story.

- There is an advantage with respect to using Toeplitz-based BLAS like operations,
- In (Massei, Mazza, and Robol 2019) they are solving the case

$$\left(\frac{1}{2}I_{N_x} - \Delta t \tilde{G}_{N_x}\right)\tilde{W}^{(m+1)} + \tilde{W}^{(m+1)}\left(\frac{1}{2}I_{N_y} - \Delta t \tilde{G}_{N_y}\right)^T = \tilde{W}^{(m)} + \Delta t F^{(m+1)}, \quad m = 0, \ldots, M-1.$$

  here the spectrum is *fictitiously independent from the discretization*, i.e., all matrix-equation solvers perform a number of iteration independent from the system size: the cost is reduced to the extended Krylov subspace cost! **But** we still have time-stepping to do.

- The case in which the matrix equation solver has a number of iterations dependent on the problem size is not yet resolved:

  ☻ Low-rank *but* 😱 **no preconditioner** – VS – 😱 **Full memory** *but* ☻ preconditioners

The tale of the matrix equation: the moral of the story.

- 🕐 There is an advantage with respect to using Toeplitz-based BLAS like operations,
- ⏩ In (Massei, Mazza, and Robol 2019) they are solving the case

$$\left(\frac{1}{2}I_{N_x} - \Delta t\,\tilde{G}_{N_x}\right)\tilde{W}^{(m+1)} + \tilde{W}^{(m+1)}\left(\frac{1}{2}I_{N_y} - \Delta t\,\tilde{G}_{N_y}\right)^{T} = \tilde{W}^{(m)} + \Delta t F^{(m+1)}, \quad m = 0, \ldots, M-1.$$

  here the spectrum is *fictitiously independent from the discretization*, i.e., all matrix-equation solvers perform a number of iteration independent from the system size: the cost is reduced to the extended Krylov subspace cost! **But** we still have time-stepping to do.

- ❓ The case in which the matrix equation solver has a number of iterations dependent on the problem size is not yet resolved:

  ☻ Low-rank *but* 😱 **no preconditioner** – VS – 😱 **Full memory** *but* ☻ preconditioners

- ❗ Still looking for a way to solve **everything** *all-at-once* compactly.

Conclusion and summary

- ✓ We have seen how to work with matrices in HODLR format,
- ✓ We have discussed a couple of strategy to solve Sylvester equations with HODLR coefficients,
- ✓ We have applied all the machinery to solve a time step of a 2D equation FDE.

Next up

- Back to *all-at-once* solution with respect to both space and time,
- Linear multistep formulas in boundary value form,
- Structured preconditioner for LMFs,
- Tensor-Train reformulation of the problem.

Bibliography I

- Beckermann, B. and A. Townsend (2019). “Bounds on the singular values of matrices with displacement structure”. In: *SIAM Rev.* 61.2. Revised reprint of ”On the singular values of matrices with displacement structure” [MR3717820], pp. 319–344. ISSN: 0036-1445. DOI: `10.1137/19M1244433`. URL: `https://doi.org/10.1137/19M1244433`.
- Breiten, T., V. Simoncini, and M. Stoll (2016). “Low-rank solvers for fractional differential equations”. In: *Electron. Trans. Numer. Anal.* 45, pp. 107–132.
- Fiedler, M. (2010). “Notes on Hilbert and Cauchy matrices”. In: *Linear Algebra Appl.* 432.1, pp. 351–356. ISSN: 0024-3795. DOI: `10.1016/j.laa.2009.08.014`. URL: `https://doi.org/10.1016/j.laa.2009.08.014`.
- Hackbusch, W. (2015). *Hierarchical matrices: algorithms and analysis*. Vol. 49. Springer Series in Computational Mathematics. Springer, Heidelberg, pp. xxv+511. ISBN: 978-3-662-47323-8; 978-3-662-47324-5. DOI: `10.1007/978-3-662-47324-5`. URL: `https://doi.org/10.1007/978-3-662-47324-5`.

Bibliography II

Kressner, D., S. Massei, and L. Robol (2019). "Low-rank updates and a divide-and-conquer method for linear matrix equations". In: *SIAM J. Sci. Comput.* 41.2, A848–A876. ISSN: 1064-8275. DOI: 10.1137/17M1161038. URL: https://doi.org/10.1137/17M1161038.

Massei, S., M. Mazza, and L. Robol (2019). "Fast solvers for two-dimensional fractional diffusion equations using rank structured matrices". In: *SIAM J. Sci. Comput.* 41.4, A2627–A2656. ISSN: 1064-8275. DOI: 10.1137/18M1180803. URL: https://doi.org/10.1137/18M1180803.

Massei, S., D. Palitta, and L. Robol (2018). "Solving rank-structured Sylvester and Lyapunov equations". In: *SIAM J. Matrix Anal. Appl.* 39.4, pp. 1564–1590. ISSN: 0895-4798. DOI: 10.1137/17M1157155. URL: https://doi.org/10.1137/17M1157155.

Massei, S., L. Robol, and D. Kressner (2020). "hm-toolbox: MATLAB software for HODLR and HSS matrices". In: *SIAM J. Sci. Comput.* 42.2, pp. C43–C68. ISSN: 1064-8275. DOI: 10.1137/19M1288048. URL: https://doi.org/10.1137/19M1288048.

Bibliography III

Popolizio, M. and V. Simoncini (2008). “Acceleration techniques for approximating the matrix exponential operator”. In: *SIAM J. Matrix Anal. Appl.* 30.2, pp. 657–683. ISSN: 0895-4798. DOI: `10.1137/060672856`. URL: `https://doi.org/10.1137/060672856`.