Matrices with Hierarchical Low-Rank Structures, Part II

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Daniel Kressner
Chair for Numerical Algorithms and HPC
MATHICSE, EPFL
daniel.kressner@epfl.ch

Contents

Status

- Know how to do $A \approx UV^T$.
- Global approximation can be difficult for “nonsmooth” matrices.
- For matrices arising from discretization of functions: Know in which blocks we can expect good low-rank approximability.

HODLR matrices

- Definition
- Addition of HODLR matrices
- Multiplication of HODLR matrices
- Factorization of HOLDR matrices

HODLR matrices: Definition

HODLR (Hierarchical Off-Diagonal Low-Rank) matrices are the fruit flies of hierarchically partitioned low-rank matrices.

Partitioning of row/column indices of $A \in \mathbb{R}^{n\times n}$ with $n = 2^p n_0$ (for simplicity):

- Start: $I_1^0 := \{1,\dots,n\}$
- Level $\ell = 0$: Partition

$$I_1^0 = \underbrace{\{1,\dots,n/2\}}_{=I_1^1} \cup \underbrace{\{n/2,\dots,n\}}_{=I_2^1}$$

- Level $\ell = 1$: Partition

$$I_1^1 = \underbrace{\{1,\dots,n/4\}}_{=I_1^2} \cup \underbrace{\{n/4+1,\dots,n/2\}}_{=I_2^2}$$

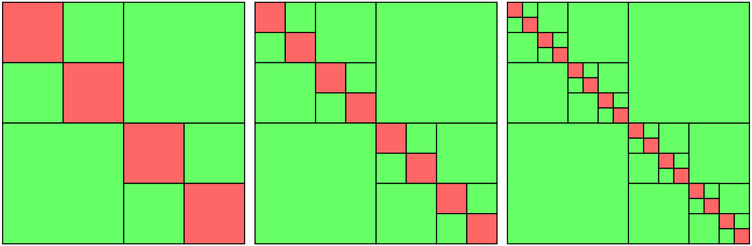
$$I_2^1 = \underbrace{\{n/2+1,\dots,3n/4\}}_{=I_1^2} \cup \underbrace{\{3n/4+1,\dots,n\}}_{=I_2^2}$$

$$\vdots$$

HODLR matrices: Definition

General:

$$I_i^\ell = I_{2i-1}^{\ell+1} \cup I_{2i}^{\ell+1}, \quad i = 1, \dots, 2^\ell, \quad \ell = 1, \dots, p-1.$$



For each off-diagonal block $A(I_i^\ell, I_j^\ell)$, $i \neq j$, assume rank at most k:

$$A(I_i^\ell, I_j^\ell) = U_i^{(\ell)} (V_j^{(\ell)})^T, \qquad U_i^{(\ell)}, V_j^{(\ell)} \in \mathbb{R}^{m_\ell \times k},$$

where $m_\ell := \# I_i^\ell = \# I_j^\ell = 2^{p-\ell} n_0$.
Diagonal blocks are dense $n_0 \times n_0$ matrices.

HODLR matrices: Storage complexity

For simplicity, assume identical ranks k and balanced partitioning. Both can be relaxed (especially ranks).

- Storage requirements for off-diagonal blocks.
  There are 2^ℓ off-diagonal blocks on level $\ell > 0$:

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

- Storage requirements for diagonal blocks.

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

- Total. Assuming $n_0 = O(1) \rightsquigarrow O(kn \log n)$.

HODLR matrices: MatVec

Matrix-vector multiplication $y = Ax$ performed recursively: On level $\ell = 1$, compute

$$y(I_1^1) = A(I_1^1, I_1^1)x(I_1^1) + A(I_1^1, I_2^1)x(I_2^1),$$
$$y(I_2^1) = A(I_2^1, I_1^1)x(I_1^1) + A(I_2^1, I_2^1)x(I_2^1).$$

- In off-diagonal blocks $A(I_1^1, I_2^1)$ and $A(I_2^1, I_1^1)$, need to multiply $n/2 \times n/2$ low-rank matrix with vector $\rightsquigarrow$ cost for each block

$$c_{LR\cdot x}(n/2) = 2nk.$$

- Diagonal blocks, are processed recursively $\rightsquigarrow$ total cost

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

Master theorem $\rightsquigarrow$

$$c_{A\cdot x}(n) = (4k + 1)\log_2(n)n.$$

HODLR matrices: Addition

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

$$C(I_i^\ell, I_j^\ell) := \mathcal{T}_k(A(I_i^\ell, I_j^\ell) + B(I_i^\ell, I_j^\ell))$$

  for off-diagonal block $I_i^\ell \times I_j^\ell$. Cost

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

  (c_{SVD} constant implied by method used for low-rank truncation)
- Total cost:

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

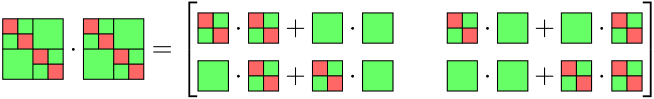
HODLR matrices: Matrix multiplication

Matrix-matrix multiplication performed recursively.

Set $A^{(\ell)}_{i,j} = A(I^\ell_i, I^\ell_j)$, $B^{(\ell)}_{i,j} = B(I^\ell_i, I^\ell_j)$. Then

$$\begin{aligned} AB &= \begin{bmatrix} A^{(1)}_{1,1} & A^{(1)}_{1,2} \\ A^{(1)}_{2,1} & A^{(1)}_{2,2} \end{bmatrix} \begin{bmatrix} B^{(1)}_{1,1} & B^{(1)}_{1,2} \\ B^{(1)}_{2,1} & B^{(1)}_{2,2} \end{bmatrix} \\ &= \begin{bmatrix} A^{(1)}_{1,1}B^{(1)}_{1,1} + A^{(1)}_{1,2}B^{(1)}_{2,1} & A^{(1)}_{1,1}B^{(1)}_{1,2} + A^{(1)}_{1,2}B^{(1)}_{2,2} \\ A^{(1)}_{2,1}B^{(1)}_{1,1} + A^{(1)}_{2,2}B^{(1)}_{2,1} & A^{(1)}_{2,1}B^{(1)}_{1,2} + A^{(1)}_{2,2}B^{(1)}_{2,2} \end{bmatrix}. \end{aligned}$$

Illustration of block structure:



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

HODLR matrices: Matrix multiplication

Four different types of multiplications involved in 2×2 block matrix-matrix product:

1.  $\cdot$ : multiplication of two HODLR matrices of size $n/2$,
2.  $\cdot$ : multiplication of two low-rank blocks,
3.  $\cdot$ : multiplication of a HODLR matrix with a low-rank block,
4.  $\cdot$ : multiplication of a low-rank block with a HODLR matrix.

Case 1 and addition require truncation!

Cost recursively:

$$\begin{aligned} c_{H\cdot H}(n) = 2(&c_{H\cdot H}(n/2) + c_{LR\cdot LR}(n/2) + c_{H\cdot LR}(n/2) + c_{LR\cdot H}(n/2) \\ &+ c_{H+LR}(n/2) + c_{LR+LR}(n/2)), \end{aligned}$$

where

$$\begin{aligned} c_{LR\cdot LR}(n) &= 4nk^2 \\ c_{H\cdot LR}(n) &= c_{LR\cdot H}(n) = kc_{H\cdot v}(n) = k(4k+1)\log_2(n)n \\ c_{H+LR}(n) &= c_{H+H}(n) = c_{SVD}(nk^3 + n\log(n)k^2) \end{aligned}$$

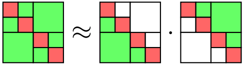
Total cost:

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

HODLR matrices: Solution of linear systems

Approximate solution of linear system $Ax = b$ with HODLR matrix A:

1. Approximate LU factorization $A \approx LU$ in HODLR format:



2. Forward/backward substitution to solve $Ly = b$, $Ux = y$.

HODLR matrices: Solution of linear systems

Forward substitution $Ly = b$ with lower-triangular HODLR L:

$$L = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix},$$

Low-rank matrix L_{21} and HODLR matrices L_{11}, L_{22}.

1. Solve
$$L_{11} y_1 = b_1.$$
2. Compute
$$\tilde{b}_2 := b_2 - L_{21} y_1$$
3. Solve
$$L_{22} y_2 = \tilde{b}_2.$$

Cost recursively:

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

On level $\ell = p$: Direct solution of $2^p = n/n_0$ linear systems of size $n_0 \times n_0$.

Total cost:

$$c_{\text{forw}}(n) \in O(kn\log(n)).$$

Backward substitution analogously.

HODLR matrices: Solution of linear systems

Approximate LU factorization. On level $\ell = 1$:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad L = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix}, \quad U = \begin{bmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{bmatrix},$$

$$\Updownarrow$$

$$A_{11} = L_{11} U_{11}, \quad A_{12} = L_{11} U_{12}, \quad A_{21} = L_{21} U_{11}, \quad A_{22} = L_{21} U_{12} + L_{22} U_{22}.$$

Algorithm:

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 $A_{22} - L_{21} U_{12}$.

Analysis of cost analogous to matrix-matrix mult:

$$c_{LU}(n) \lesssim c_{H \cdot H}(n) = O(k^3 n \log n + k^2 n \log^2 n).$$

Hierarchical matrices ($\mathcal{H}$ matrices)

- Clustering
- 3D Example

General clustering/partitioning philosophy

Find partitioning such that:

(a) Ranks of all matrix blocks are small.
(b) Total number of matrix blocks is small.

Main approaches to balance both goals:

- Geometric clustering.
- Algebraic clustering.

1D Example

1D integral equation: Find $u : \Omega \to \mathbb{R}$ such that

$$\int_0^1 \log|x-y|u(y)\mathrm{d}y = f(x), \qquad x \in \Omega = [0,1],$$

for $f : \Omega \to \mathbb{R}$.

For $n = 2^p$, subdivide interval $[0,1]$ into subintervals

$$\tau_i := [(i-1)h, ih], \quad 1 \le i \le n, \quad h = 1/n.$$

Galerkin discretisation with piecewise constant basis functions $\rightsquigarrow$ $A \in \mathbb{R}^{n \times n}$ defined by

$$A(i,j) := \int_0^1 \int_0^1 \varphi_i(x) \log|x-y| \varphi_j(y)\mathrm{d}y\mathrm{d}x = \int_{\tau_i} \int_{\tau_j} \log|x-y|\mathrm{d}y\mathrm{d}x.$$

1D Example

- $\log|x-y|$ has singularity at $x = y \rightsquigarrow$ can only expect good low-rank approximations for a subblock if *all* indices i, j contained in the subblock are sufficiently far apart.
- Cluster tree: Subdivide index set $I = \{1, \dots, n\}$ by binary tree T_I such that neighbouring indices are hierarchically grouped together. Driven by subdivision of the domain $\Omega = [0, 1]$:

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

$$[0,1]$$
$$[0,\tfrac{1}{2}] \quad [\tfrac{1}{2},1]$$
$$[0,\tfrac{1}{4}] \quad [\tfrac{1}{4},\tfrac{1}{2}] \quad [\tfrac{1}{2},\tfrac{3}{4}] \quad [\tfrac{3}{4},1]$$
$$[0,\tfrac{1}{8}[\tfrac{1}{8},\tfrac{1}{4}[\tfrac{1}{4},\tfrac{3}{8}[\tfrac{3}{8},\tfrac{1}{2}[\tfrac{1}{2},\tfrac{5}{8}[\tfrac{5}{8},\tfrac{3}{4}[\tfrac{3}{4},\tfrac{7}{8}[\tfrac{7}{8},1]$$

Admissibility condition

- Consider general domain $\Omega \in \mathbb{R}^d$ and consider integral equation with ‘diagonal” singularity at $x = y$.
- For $s \subset I$, Ω_s is part of the domain containing support of all basis functions associated with s.
- Admissibility condition motivated by interpolation error estimates: Let $\eta > 0$ be a given constant and let $s, t \subset I$. Matrix block (s, t) is called *admissible* if

$$\max\{\text{diam}(\Omega_s), \text{diam}(\Omega_t)\} \leq \eta \text{dist}(\Omega_s, \Omega_t),$$

where

$$\text{diam}(\Omega) := \max_{x,y\in\Omega} \|x - y\|_2,$$
$$\text{dist}(\Omega_s, \Omega_t) := \min_{x\in\Omega_s, y\in\Omega_t} \|x - y\|_2.$$

Partitioning algorithm

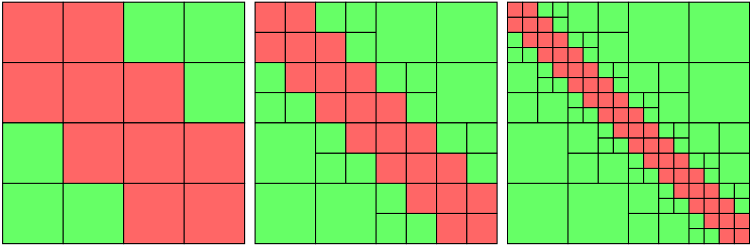
- Assume cluster tree T_I on $I = \{1, \ldots, n\}$
- Call BuildBlocks($\{1, \ldots, n\}, \{1, \ldots, n\}$)

BuildBlocks(s, t)

if (s, t) is admissible or both s and t have no sons **then**
    Fix block (s, t)
else
    for all sons s' of s and all sons t' of t **do**
        BuildBlocks(s', t')
    end for
end if

1D Example

Block subdivision with admissibility constant $\eta = 1$:



$p = 2$ $\qquad$ $p = 3$ $\qquad$ $p = 4$

Summary of $\mathcal{H}$ matrices

- Similar to HODLR but more general partitioning driven by cluster tree + admissibility condition.
- Generality offered by partitioning can greatly reduce ranks (but much more difficult to program).
- HODLR ideas for performing operations (addition, multiplication, factorization) extend to $\mathcal{H}$ matrices. Recursions derived from cluster tree.
- Complexity estimates for HODLR extend to $\mathcal{H}$ matrices under certain assumptions (e.g., balanced cluster tree).
- See `www.hlib.org` and `www.hlibpro.com` for software.

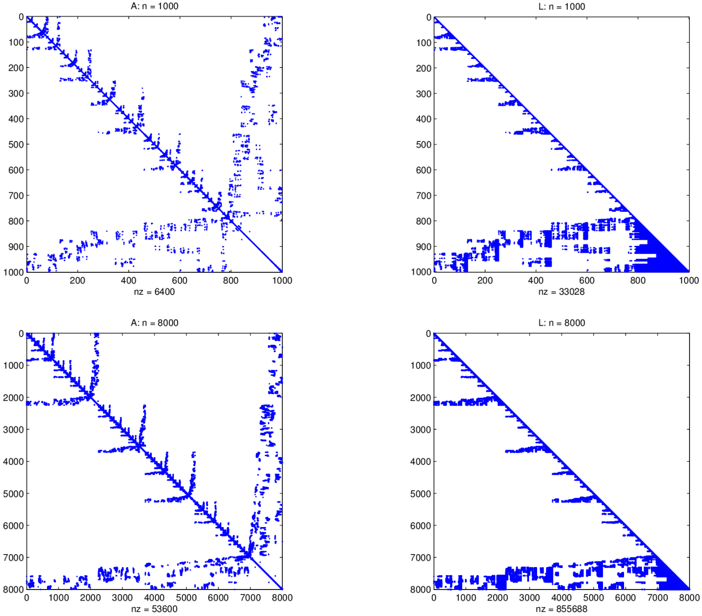
3D Example

Let $\Omega = (0,1)^3$ and consider finite difference discretization of

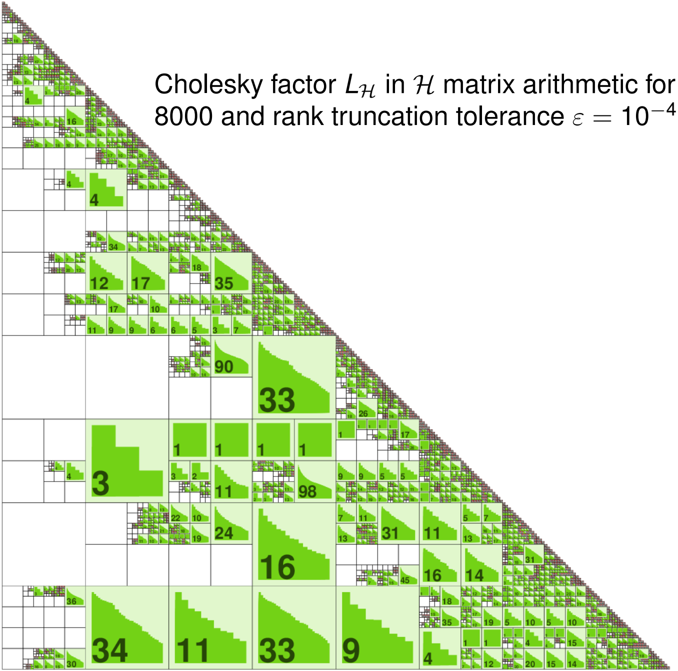
$$\begin{aligned} -\Delta u &= f \quad \text{in}\,\Omega, \\ u &= 0 \quad \text{on}\,\partial\Omega. \end{aligned}$$

- Nonzero entries in Cholesky factor of A (after reordering) $\sim n^{5/3}$.
- Limits use of sparse direct methods.

3D Example: Cholesky factors



Cholesky factor $L_{\mathcal{H}}$ in $\mathcal{H}$ matrix arithmetic for $n = 8000$ and rank truncation tolerance $\varepsilon = 10^{-4}$.



3D Example: Performance

- Use hierachical Cholesky factor as preconditioner in CG.
- Stop preconditioned CG when accuracy 10^{-8} is reached.

n	ε	$\|I - (L_{\mathcal{H}} L_{\mathcal{H}}^T)^{-1} A\|_2$	CG steps	stor($L_{\mathcal{H}}$) [MB]	time(chol) [s]	time(solve) [s]
27000	1e-01	7.82e-01	8	64	1.93	0.35
	1e-02	5.54e-02	3	85	3.28	0.21
	1e-03	3.88e-03	2	107	4.61	0.18
	1e-04	2.98e-04	2	137	6.65	0.22
	1e-05	2.32e-05	1	172	10.31	0.17
64000	1e-01	9.11e-01	8	174	5.82	0.88
	1e-02	9.66e-02	4	255	10.22	0.62
	1e-03	6.56e-03	2	330	15.15	0.48
	1e-04	5.51e-04	2	428	23.78	0.54
	1e-05	4.53e-05	1	533	34.81	0.46
125000	1e-01	1.15e+00	9	373	14.26	2.09
	1e-02	1.57e-01	4	542	25.21	1.32
	1e-03	1.19e-02	3	764	44.33	1.33
	1e-04	9.12e-04	2	991	65.86	1.19
	1e-05	7.37e-05	1	1210	97.62	1.01

For comparison: Sparse Cholesky factor for $n = 125\,000$ requires 964 MB memory and 8 seconds.

HSS matrices

- Definition
- Mat-vec product

HSS matrices: Definition

HSS matrix $=$ HODLR matrix + nestedness of low-rank factors in off-diagonal blocks.

Consider off-diagonal block on level ℓ:

$$A(I_i^\ell, I_j^\ell) = U_i^{(\ell)} S_{i,j}^{(\ell)} (V_j^{(\ell)})^T, \qquad S_{i,j}^{(\ell)} \in \mathbb{R}^{k\times k}.$$

HSS: There exist matrices $X_i^{(\ell)} \in \mathbb{R}^{2k\times k}$, $Y_j^{(\ell)} \in \mathbb{R}^{2k\times k}$ such that

$$U_i^{(\ell)} = \begin{bmatrix} U_{2i-1}^{(\ell+1)} & 0 \\ 0 & U_{2i}^{(\ell+1)} \end{bmatrix} X_i^{(\ell)}, \qquad V_j^{(\ell)} = \begin{bmatrix} V_{2j-1}^{(\ell+1)} & 0 \\ 0 & V_{2j}^{(\ell+1)} \end{bmatrix} Y_j^{(\ell)}.$$

- Only need to store low-rank factors on lowest level, $2k \times k$ matrices $X_i^{(\ell)}$, $Y_j^{(\ell)}$, and $k \times k$ matrices $S_{i,j}^{(\ell)}$ on each level.
- $O(kn)$ storage.

HSS matrices: Matrix-vector product

On level $\ell = p$:

$$x_i^p = (V_i^{(p)})^T x(I_i^p), \qquad i = 1, \ldots, 2^p$$

for level $\ell = p-1, \ldots, 1$ **do**

$$x_i^\ell = (Y_i^{(\ell)})^T \begin{bmatrix} x_{2i-1}^{\ell+1} \\ x_{2i}^{\ell+1} \end{bmatrix}, \qquad i = 1, \ldots, 2^\ell$$

end for

for level $\ell = 1, \ldots, p-1$ **do**

$$\begin{bmatrix} y_{2i-1}^{(\ell)} \\ y_{2i}^\ell \end{bmatrix} = \begin{bmatrix} 0 & S_{2i-1,2i}^{(\ell)} \\ S_{2i,2i-1}^{(\ell)} & 0 \end{bmatrix} \begin{bmatrix} x_{2i-1}^\ell \\ x_{2i}^\ell \end{bmatrix}, \qquad i = 1, \ldots, 2^{\ell-1}$$

end for

for level $\ell = 1, \ldots, p-1$ **do**

$$\begin{bmatrix} y_{2i-1}^{\ell+1} \\ y_{2i}^{\ell+1} \end{bmatrix} = \begin{bmatrix} y_{2i-1}^{\ell+1} \\ y_{2i}^{\ell+1} \end{bmatrix} + X_i^{(\ell)} y_i^\ell, \qquad i = 1, \ldots, 2^\ell$$

end for

On level $\ell = p$:

$$y(I_i^p) = U_i^{(p)} y_i^p + A(I_i^p, I_i^p) x(I_i^p), \qquad i = 1, \ldots, 2^p$$

- $O(n)$ operations
- Closely related to fast multipole method.

HSS matrices: Summary

- Operations and factorizations for HSS matrices in [Sheng/Dewilde/Chandrasekaran: Algorithms to Solve Hierarchically Semi-separable Systems, 2007].
- LU factorization / solving linear systems has complexity $O(n)$.
- $\mathcal{H}^2$-matrices

A kaleidoscope of applications

Applications related to discretizations of differential/integral equations

- $\mathcal{H}$ matrix based preconditioning for FE discretization of 3D Maxwell [Ostrowski et al.'2010].
- Matrix sign function iteration in $\mathcal{H}$-arithmetic for solving matrix Lyapunov and Riccati equations [Grasedyck/Hackbusch/Khoromskij'2004].
- HSS methods for integral equations [Martinsson, Rokhlin and collaborators'2005–2015].
- Contour integral+$\mathcal{H}$ matrices for matrix functions [Gavrilyuk et al.'2002].
- HODLR for approximating frontal matrices in sparse direct fact of 3D [Aminfar et al.'2014].
- HSS in sparse direct fact [Xiaoye Sherry Li and collaborators'2011–2015].
- $\mathcal{H}$ matrix approximation of BEM matrices [Hackbusch/Sauter/...'1990ies].
- ...

Other applications

- $\mathcal{H}$ matrices for fast sparse covariance matrix estimation [Ballani/DK'2014, Greengard et al.'2014].
- Block low-rank approximation of kernel matrices [Si/Hsieh/Dhillon'2014, Wang et al.'2015].
- $\mathcal{H}^2$ matrix approximations for ensemble Kalman filters [Li et al.'2014].
- Clustered low-rank approximation of graphs [Savas/Dhillon'2011].
- ...