Lecture 7

Low Rank Approximate Factorizations

Xiaoye Sherry Li
Lawrence Berkeley National Laboratory, USA
xsli@lbl.gov

Introduction

Consider solving large sparse linear system $A\mathbf{u} = \mathbf{b}$ with Gaussian elimination: $A = LU$

- Deliver reliable solution, error bounds, condition estimation, efficient for many RHS, . . .

- **Complexity wall** ... not linear time
  [George '73] For model problems, (exact) sparse LU with best ordering Nested Dissection gives optimal complexity:
  - 2D ($kxk = n$ grids): $\mathcal{O}(n \log n)$ Fill, $\mathcal{O}(n^{3/2})$ Flops

  Fill: adding up the dense submatrices of all the "+" separators:

$$k^2 + 4\left(\frac{k}{2}\right)^2 + 4^2\left(\frac{k}{4}\right)^2 + \ldots = \sum_{i=0} 4^i \left(\frac{k}{2^i}\right)^2 = O(k^2 \log k)$$

  Flops: dominated by cubic term of factorizing top-level separator: $O(k^3)$

Approximation

Exploit "data-sparseness" structure in separators

- data-sparse: matrix may be dense, but has a compressed representation smaller than N^2

Low-rank matrices as basic building blocks

- If B has exact rank at most k:
  - Outer-product form: $B_{m\times n} = U_{m\times k} V^T_{k\times n}, k \leq n$
  - Orthonormal outer-product form:
    $B_{m\times n} = U_{m\times k} X_{k\times k} V^T_{k\times n}, \quad U^T U = V^T V = I_k$
- If A has numerical low rank k (called ε-rank):
  $A = U\Sigma V^T \approx A_k := U\Sigma_k V^T, \Sigma = diag(\sigma_1, \ldots, \sigma_k, \sigma_{k+1}, \ldots, \sigma_n)$
  $\Sigma_k = diag(\sigma_1, \ldots, \sigma_k, 0, \ldots, 0)$, with $\sigma_k > \varepsilon$

Algorithms:

- truncated SVD
- rank-revealing QR
- randomized sampling, ...

Approximations by LR matrices

- Singular Value Decomposition (SVD)
  $A = U\Sigma V^T \approx A_k := U\Sigma_k V^T$
  $\Sigma = diag(\sigma_1, \ldots, \sigma_k, \sigma_{k+1}, \ldots, \sigma_n)$
  $\Sigma_k = diag(\sigma_1, \ldots, \sigma_k, 0, \ldots, 0)$
  - **accuracy**: $\|A - A_k\|_2 = \sigma_{k+1}$
  - **cost**: $O(m^2 n) \quad (m \leq n)$

- Rank-Revealing QR decomposition (RRQR)
  $A\Pi = QR, \quad R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}, \quad \Pi$ permutation matrix
  Choose $U = Q(:, 1:k), V = \Pi[R_{11} \quad R_{12}]^T$
  - **accuracy**: $\|A - UV^T\|_2 = \|R_{22}\|_2 \leq \mathbf{c}\, \sigma_{k+1}$
  - **cost**: $O(kmn) \quad (m \leq n,\ k \approx m)$

LR Matrices (con't)

- Randomized sampling
  1. Pick random matrix $\Omega_{n\times(k+p)}$, p small, e.g. 10
  2. Sample matrix $S = A\Omega$, with slight oversampling p
  3. Compute $Q = \text{ON-basis}(S)$

  - **accuracy:** with *probability* $\geq 1 - 6 \cdot p^{-p}$,
    $\|A - QQ^*A\| \leq [1 + 11\sqrt{k+p} \cdot \sqrt{\min\{m,n\}}]\sigma_{k+1}$
  - **cost:** $O(kmn)$

- **Remarks**
  - Kernel: All have same asymptotic cost with explicit matrix
    - ⋆ RS can be faster when fast matrix-vector available
    - ⋆ RS useful when only matrix-vector available
  - Putting in sparse solver: costs will be different ...

LR Matrices (con't)

- Randomized sampling
  1. Pick random matrix $\Omega_{n\times(k+p)}$, p small, e.g. 10
  2. Sample matrix $S = A\Omega$, with slight oversampling p
  3. Compute $Q = \text{ON-basis}(S)$

  - **accuracy:** with *probability* $\geq 1 - 6 \cdot p^{-p}$,
    $\|A - QQ^*A\| \leq [1 + 11\sqrt{k+p} \cdot \sqrt{\min\{m,n\}}]\sigma_{k+1}$
  - **cost:** $O(kmn)$

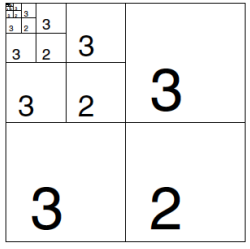
- **Remarks**
  - Kernel: All have same asymptotic cost with explicit matrix
    - ★ RS can be faster when fast matrix-vector available
    - ★ RS useful when only matrix-vector available
  - Putting in sparse solver: costs will be different ...

Data-sparse representations

Hierarchical matrices: $\mathcal{H}$-matrix, $\mathcal{H}^2$-matrix

[Bebendorf, Borm, Grasedyck, Hackbusch, Le Borne, Martinsson, Tygert, ...]

- allow **Fast** matrix-vector multiplication, factorization, inversion, ...
- $\mathcal{H}$-matrix : Given a "suitable" partition $P : I \times J$ of row and column dimensions, ranks of all blocks $A_b \leq k$. (low-rank blocks chosen independently from each other)
  - Example [Bebendorf 2008]: Hilbert matrix $h_{ij} = \frac{1}{i+j-1}$ and the blockwise ranks:
  - Flops of matrix-vector multiplication: $O(k(|I| \log |I| + |J| \log |J|))$



- $\mathcal{H}^2$-matrix is a uniform $\mathcal{H}$-matrix with **nested** cluster bases
  - more restrictive but faster than $\mathcal{H}$-matrix
  - Flops of matrix-vector multiplication: $O(k(|I| + |J|))$ (algebraic generalization of the **Fast Multipole** method)

Data-sparse representations

(Hierarchically) Semi-Separable matrices

[Bini, Chandrasekaran, Dewilde, Eidelman, Gemignani, Gohberg, Gu, Kailath, Olshevsky, van der Veen, Van Barel, Vandebril, White, et al.]

- SS matrix: $S = triu(UV^T) + tril(WZ^T)$, where $U, V, W,$ and Z are rank-k matrices.
  - Example: can be used to represent the inverse of a banded matrix
- HSS matrix: the bases are required to be **nested**
  - special case of $\mathcal{H}^2$-matrix

Other low-rank factorization ideas:

- BLR (Block LR) (Amestoy et al.)
- MLR (Multilevel LR) (Saad et al.)

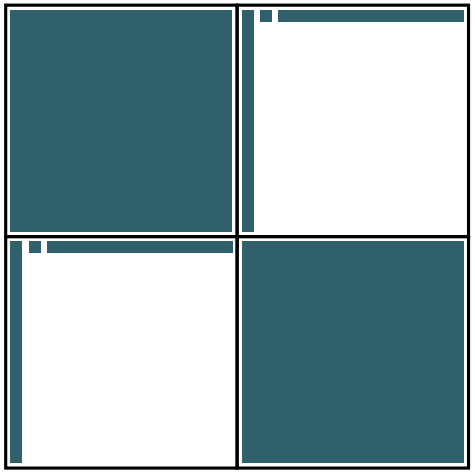
Outline

- How it works operationally?
  - Hierarchical matrix representation, factorization
  - HSS-embedded multifrontal factorization
    - ★ targeting at nonsymetric systems (with PDE behind)
- Theory
  - Schur monotonicity
  - conditioning analysis
  - rank analysis for discretized PDEs
- Practice
  - ordering within separator
  - parallelization
  - preconditioning
- Summary

Hierarchically Semi-Separable matrices

An HSS matrix A is a dense matrix whose off-diagonal blocks are low-rank.
High-level structure: 2×2 blocks

$$A = \left[\begin{array}{c|c} D_1 & U_1 B_1 V_1^T \\ \hline U_2 B_2 V_1^T & D_2 \end{array}\right]$$



Fundamental property required for efficiency: nested bases

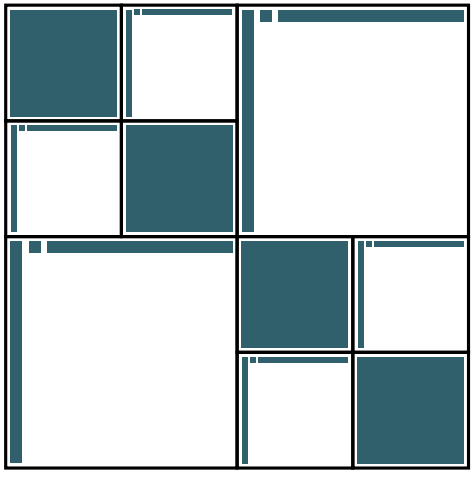
$$U_3 = \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} U_3^{small}, \; U_3^{small} : 2k \times k$$

Same for U_3, U_6, V_6 and recursively at subsequent levels.

Hierarchically Semi-Separable matrices

An HSS matrix A is a dense matrix whose off-diagonal blocks are low-rank.
Recursion

$$A = \left[\begin{array}{cc|cc} D_1 & U_1 B_1 V_2^T & \multicolumn{2}{c}{\multirow{2}{*}{$U_3 B_3 V_6^T$}} \\ U_2 B_2 V_1^T & D_2 & & \\ \hline \multicolumn{2}{c|}{\multirow{2}{*}{$U_6 B_6 V_3^T$}} & D_4 & U_4 B_4 V_5^T \\ & & U_5 B_5 V_4^T & D_5 \end{array}\right]$$



Fundamental property required for efficiency: nested bases

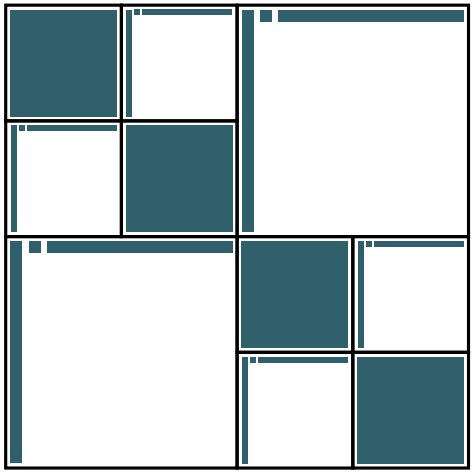
$$U_3 = \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} U_3^{small}, U_3^{small} : 2k \times k$$

Same for U_3, U_6, V_6 and recursively at subsequent levels.

Hierarchically Semi-Separable matrices

An HSS matrix A is a dense matrix whose off-diagonal blocks are low-rank.
Recursion

$$A = \left[\begin{array}{cc|cc} D_1 & U_1 B_1 V_2^T & \multicolumn{2}{c}{\multirow{2}{*}{$U_3 B_3 V_6^T$}} \\ U_2 B_2 V_1^T & D_2 & & \\ \hline \multicolumn{2}{c|}{\multirow{2}{*}{$U_6 B_6 V_3^T$}} & D_4 & U_4 B_4 V_5^T \\ & & U_5 B_5 V_4^T & D_5 \end{array}\right]$$

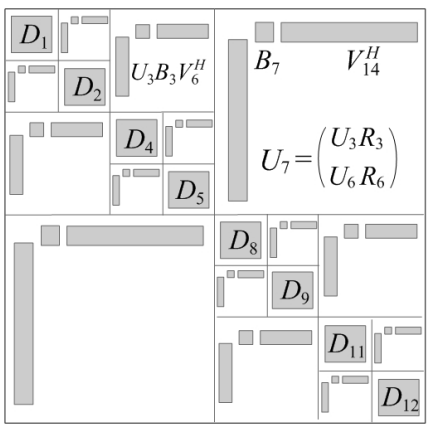


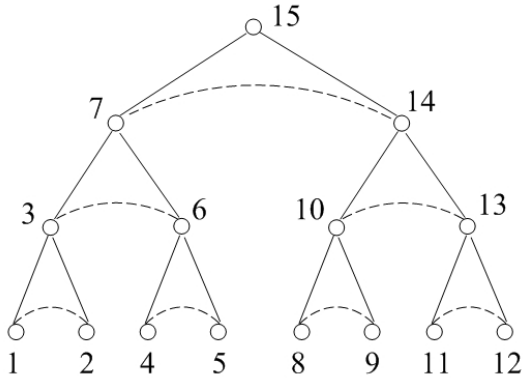
Fundamental property required for efficiency: nested bases

$$U_3 = \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} U_3^{small}, \; U_3^{small} : 2k \times k$$

Same for U_3, U_6, V_6 and recursively at subsequent levels.

Hierarchical bases, HSS tree





For efficiency, require:

$$U_3 = \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} U_3^{small}, \; U_3^{small} : 2k \times k, \quad U_6 = \begin{bmatrix} U_4 & 0 \\ 0 & U_5 \end{bmatrix} U_6^{small}, \quad U_6^{small} : 2k \times k$$

$$U_7 = \begin{bmatrix} U_3 & 0 \\ 0 & U_6 \end{bmatrix} U_7^{small}, \quad U_7^{small} : 2k \times k$$

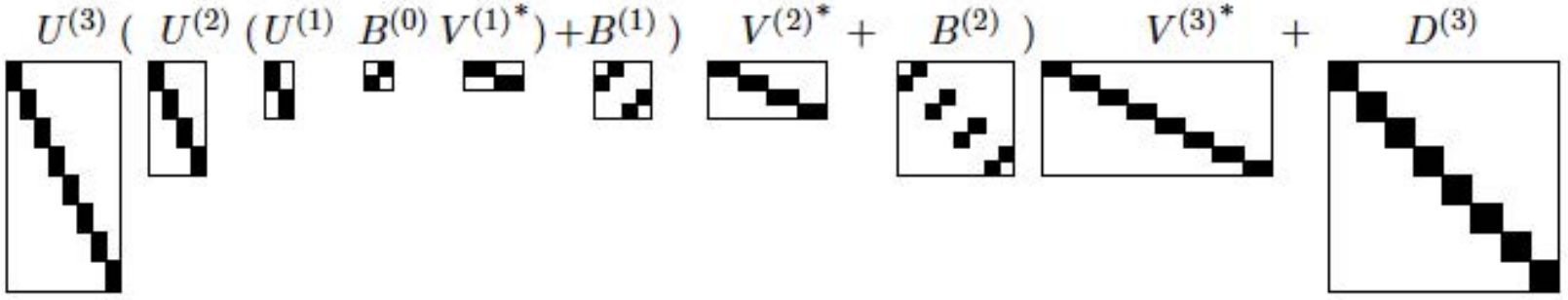
Each basis is a product of descendents' bases:

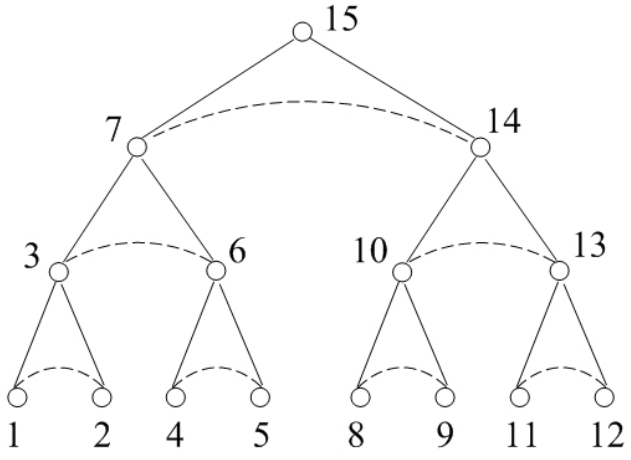
$$U_7 = \begin{bmatrix} U_1 & 0 & 0 & 0 \\ 0 & U_2 & 0 & 0 \\ 0 & 0 & U_4 & 0 \\ 0 & 0 & 0 & U_5 \end{bmatrix} \begin{bmatrix} U_3^{small} & 0 \\ 0 & U_6^{small} \end{bmatrix} U_7^{small},$$

Not to multiply out!

HSS explicit representation (construction)

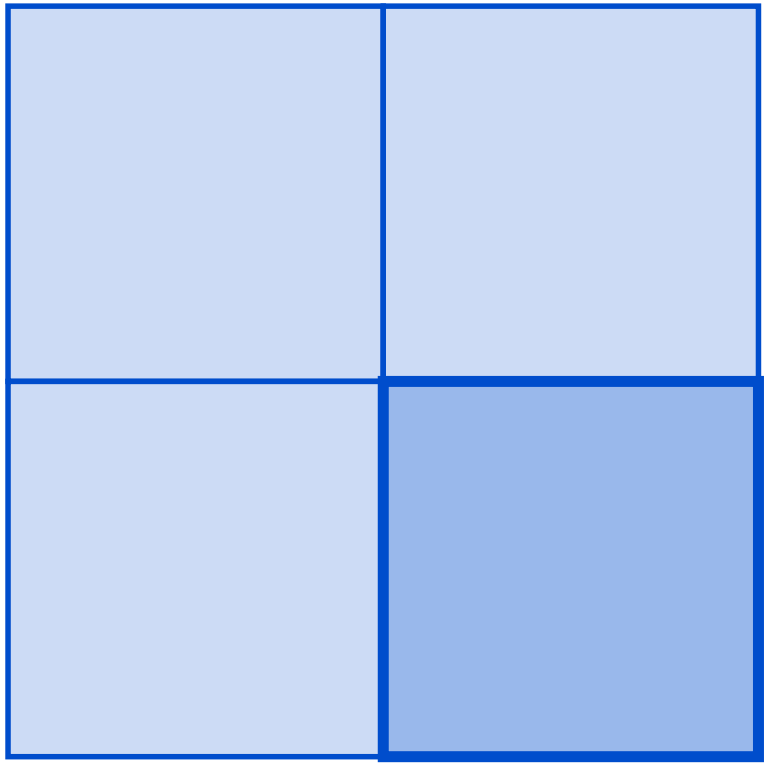
[Martinsson]



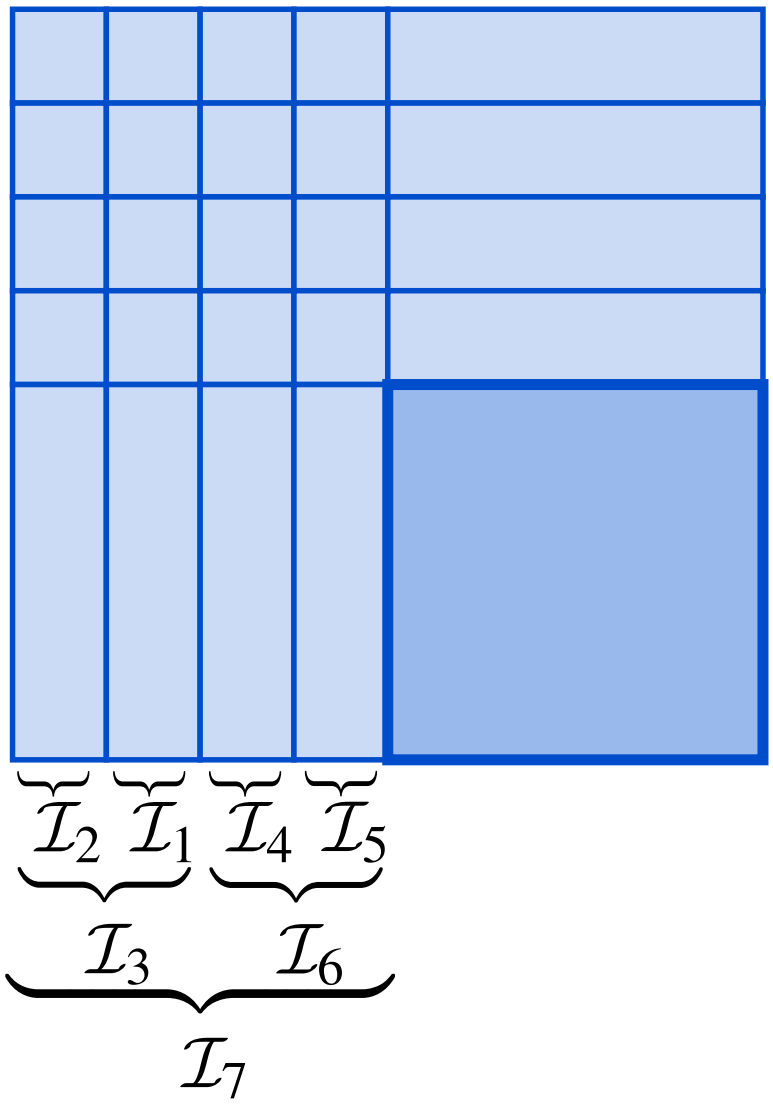


- keep it as an unevaluated product & sum
- operations going up / down the HSS tree

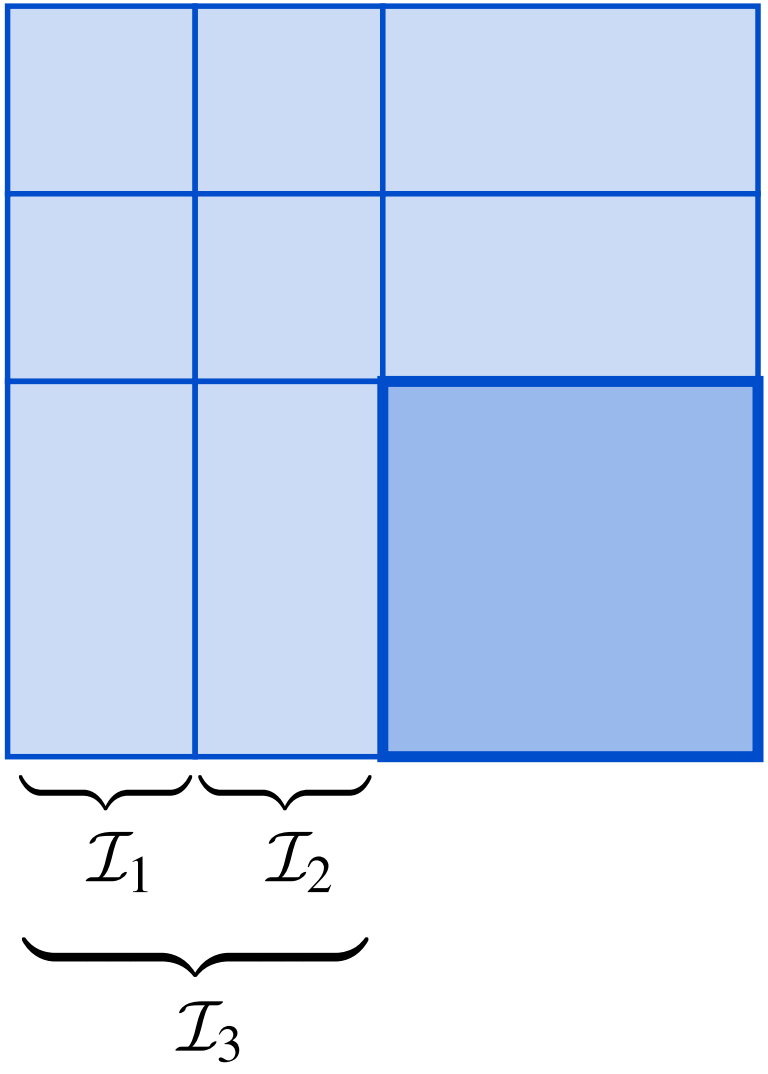
Structured factorization



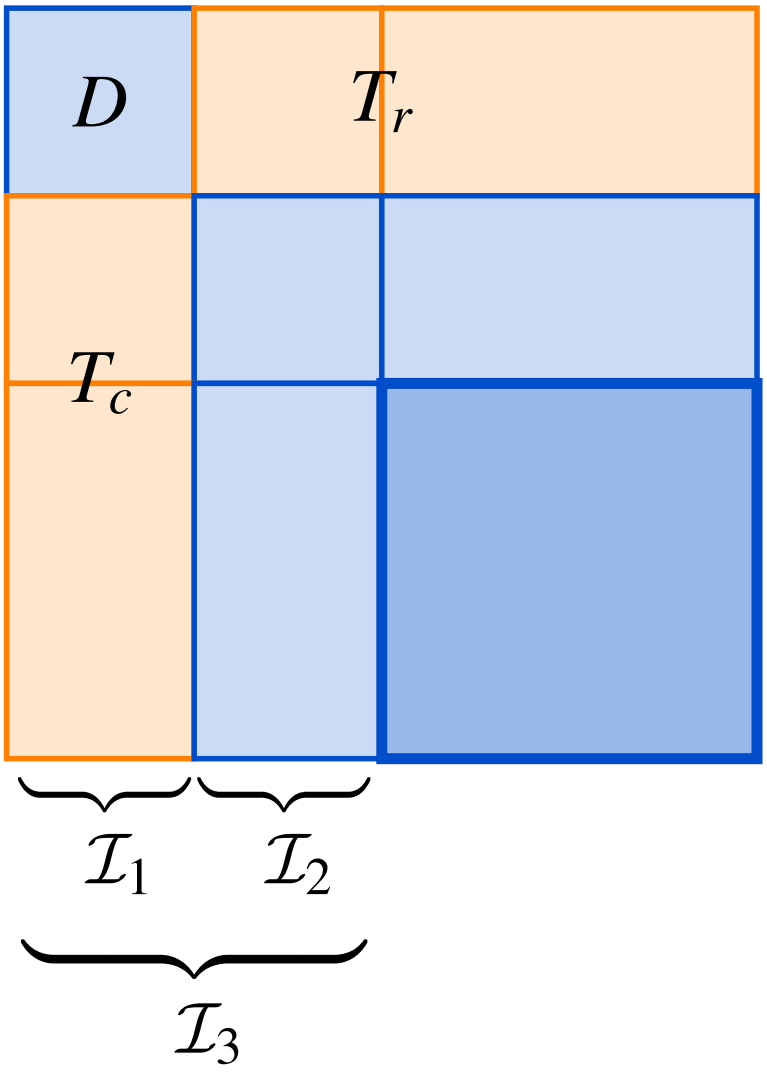
Structured factorization



Structured factorization



Structured factorization



HSS node 1:

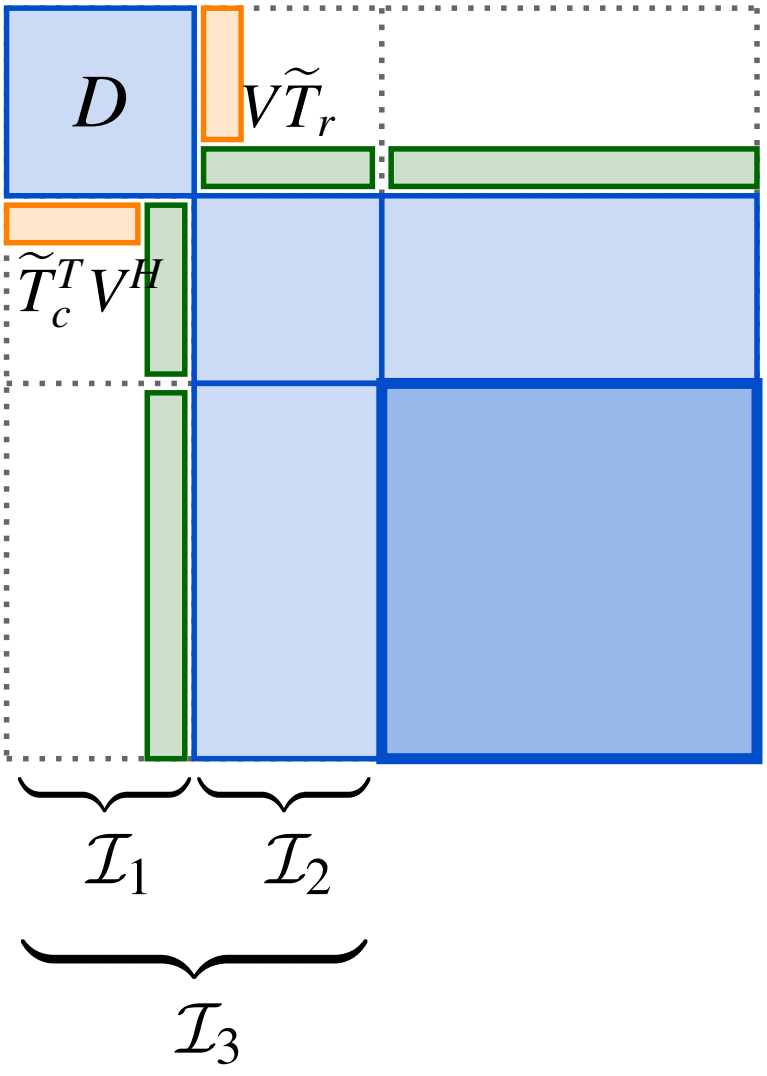
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$

Structured factorization



HSS node 1:

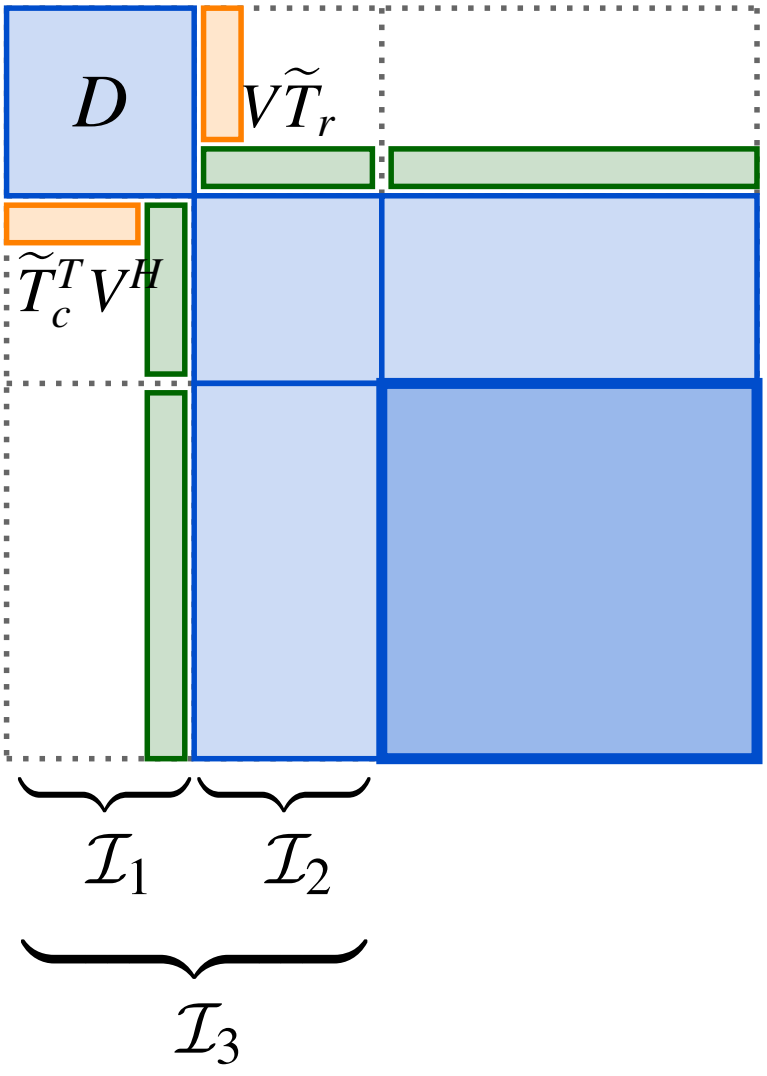
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$$

Structured factorization



HSS node 1:

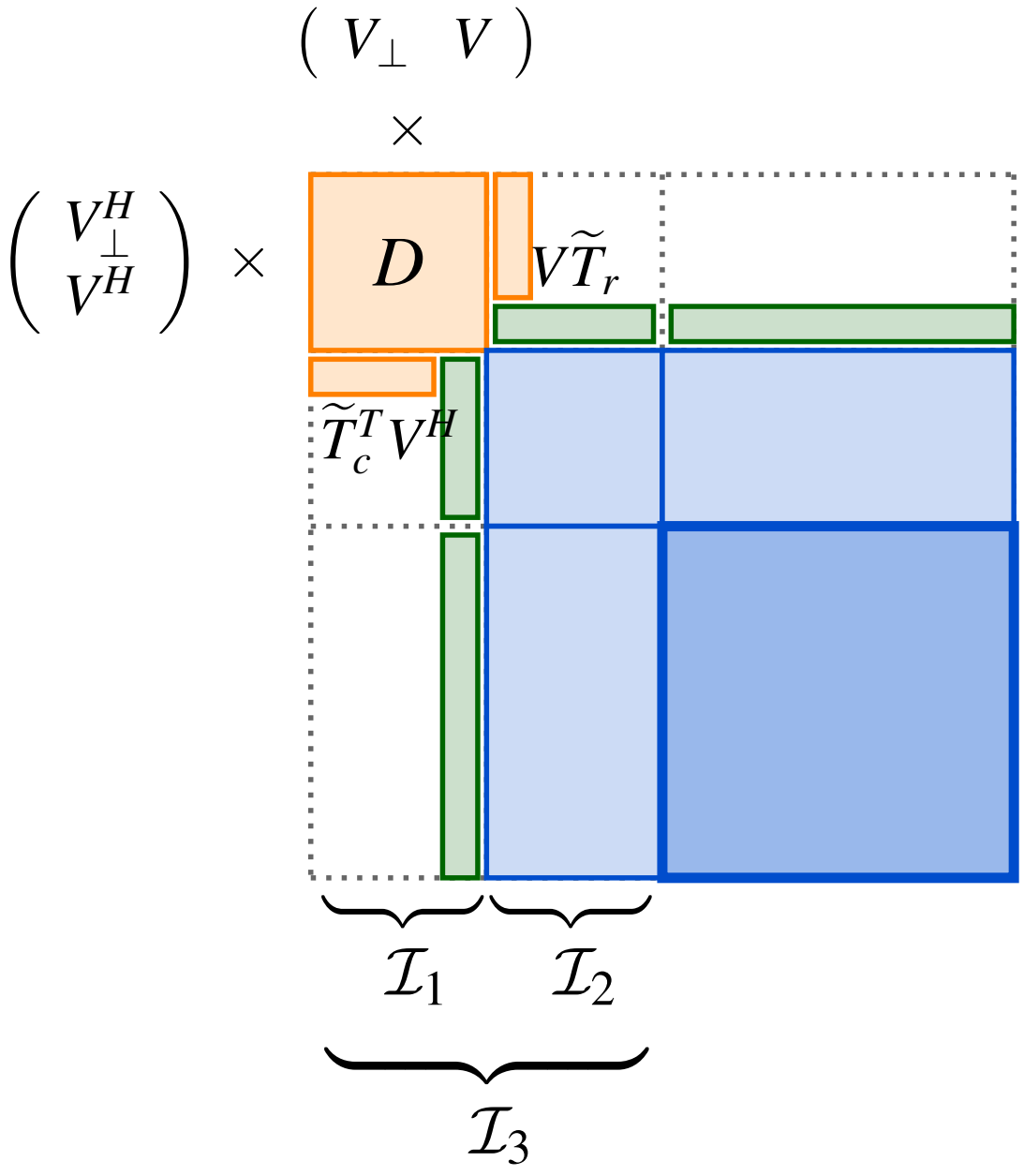
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$

Structured factorization



HSS node 1:

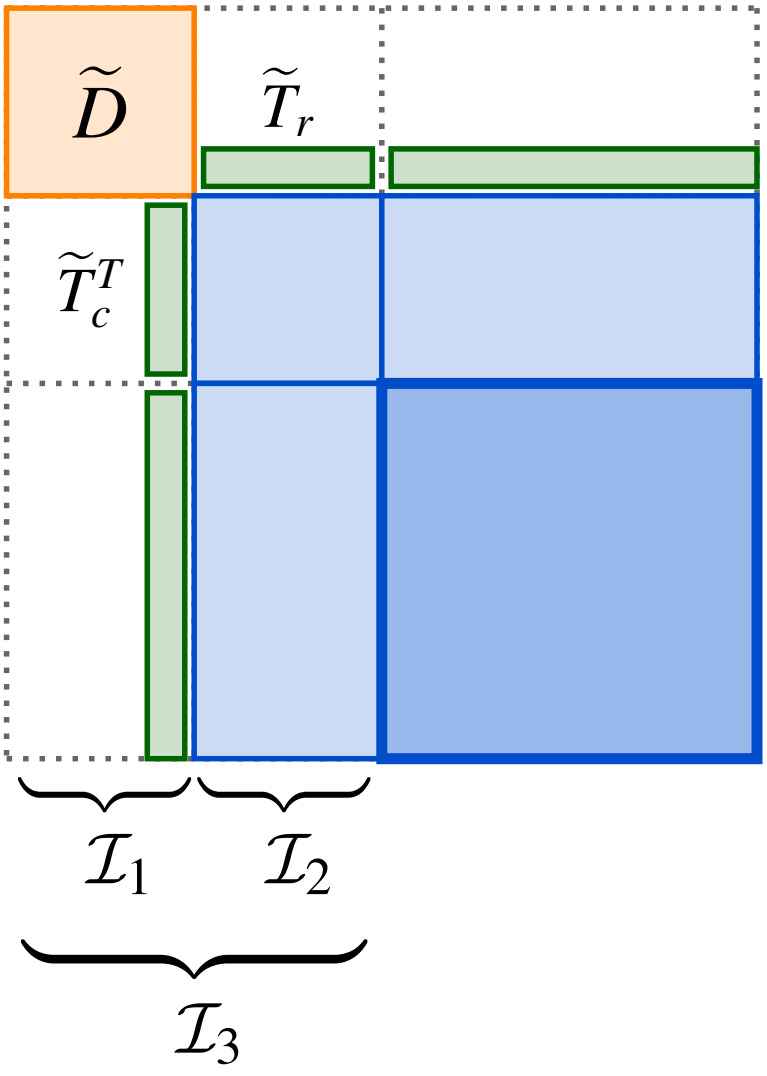
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H D (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$

Structured factorization



HSS node 1:

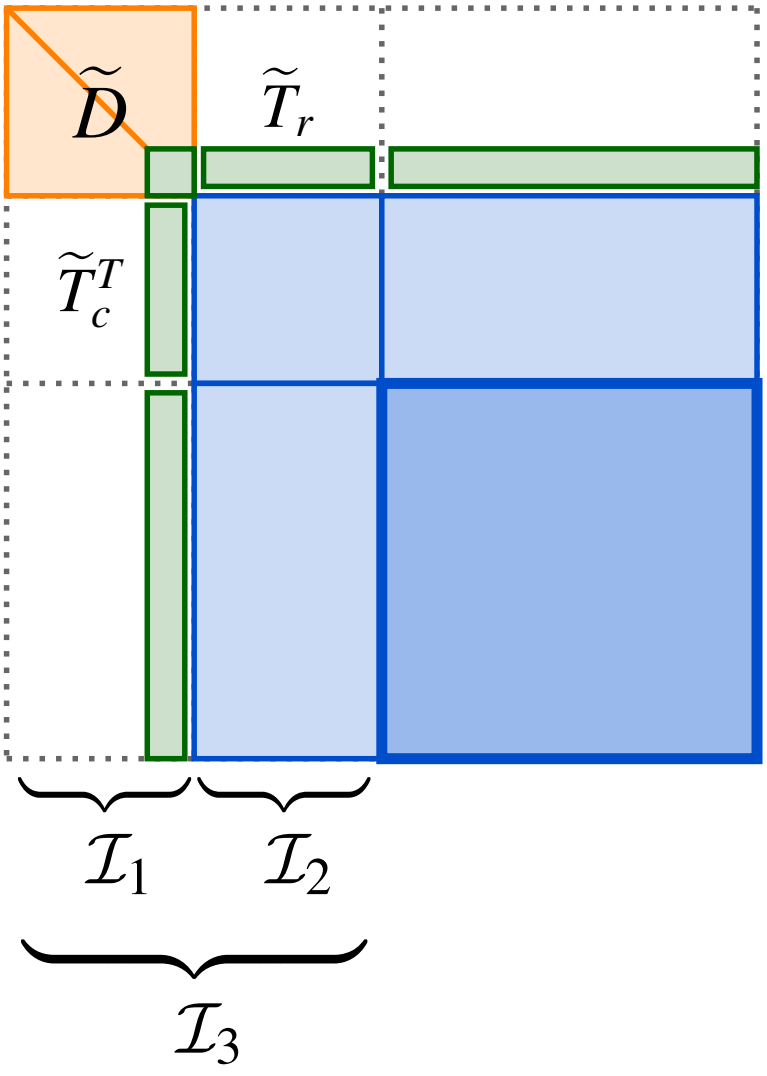
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H D (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$$\begin{pmatrix} D_L & \\ D_C & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$$

Structured factorization



HSS node 1:

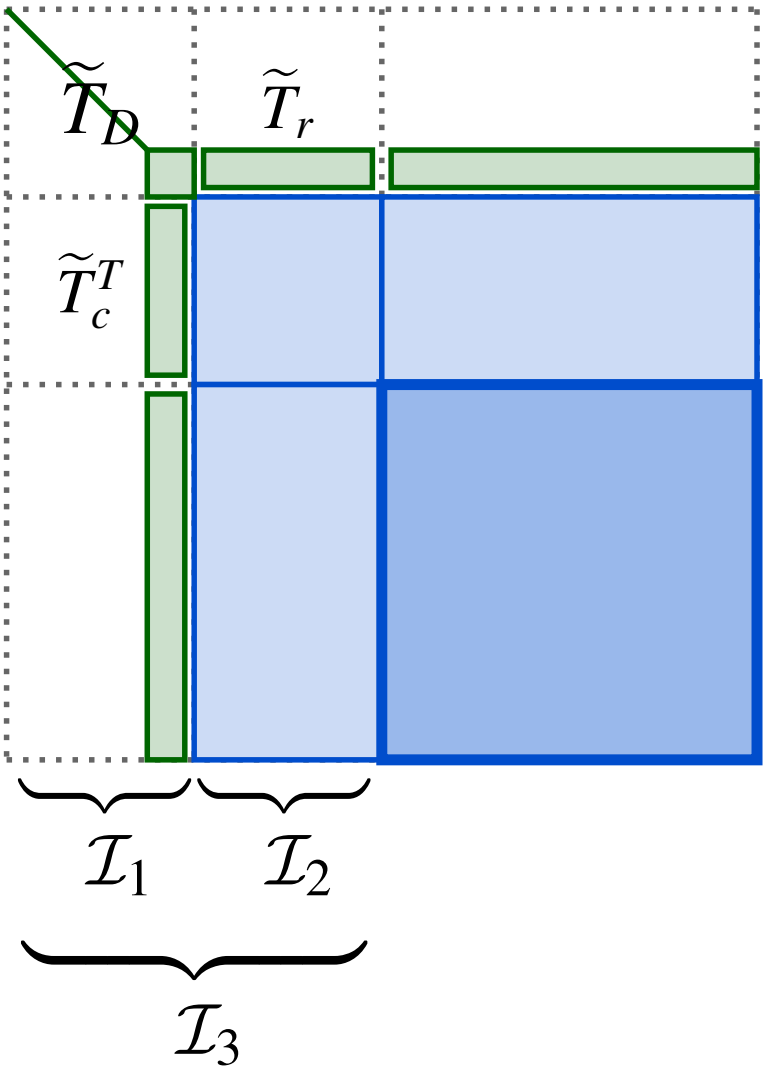
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H D (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$$

Structured factorization



HSS node 1:

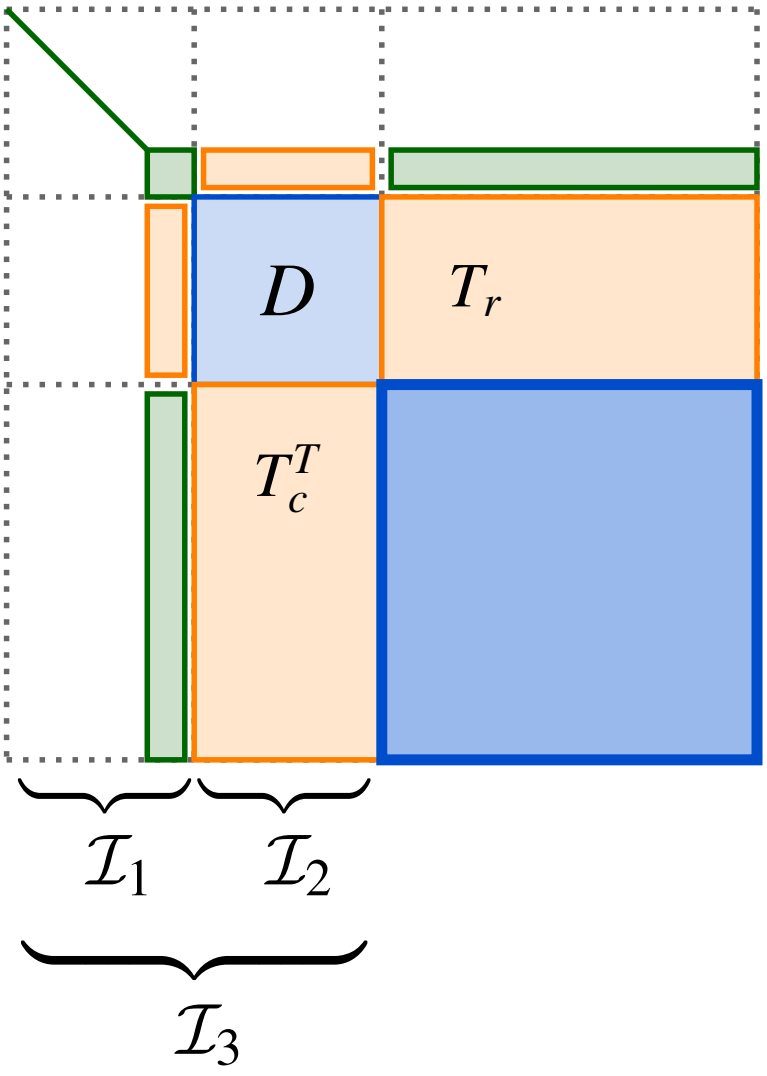
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H D (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$$

Structured factorization



HSS node 2:

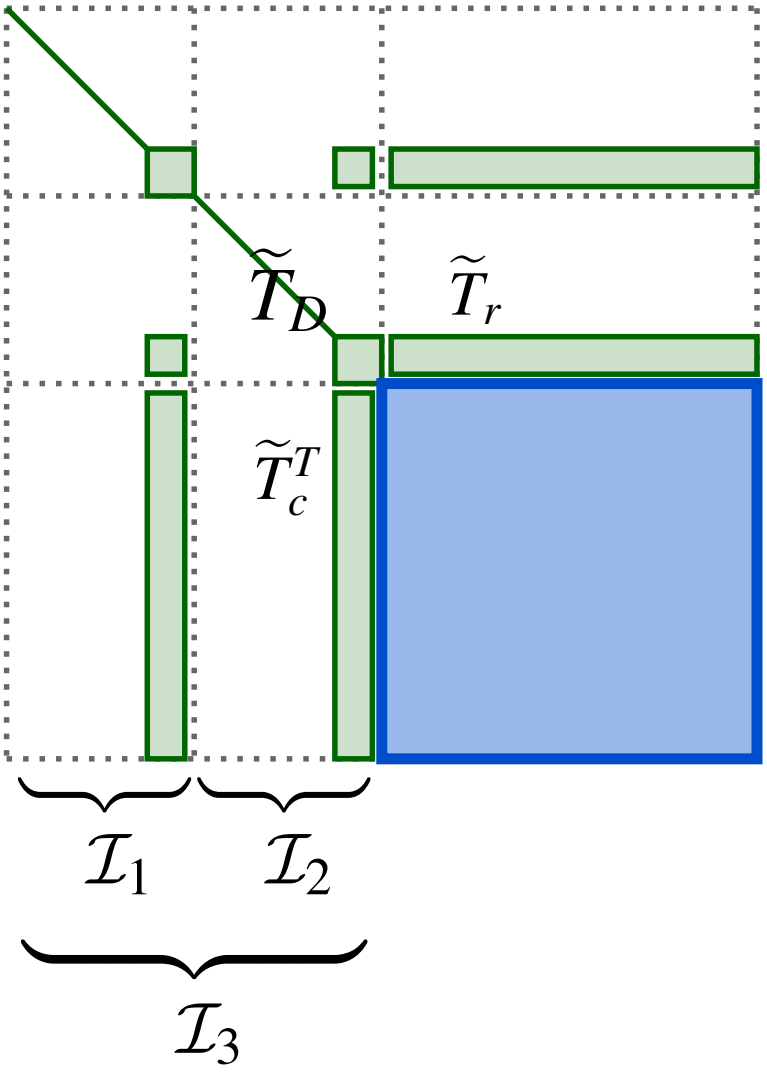
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H D (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$$

Structured factorization



HSS node 2:

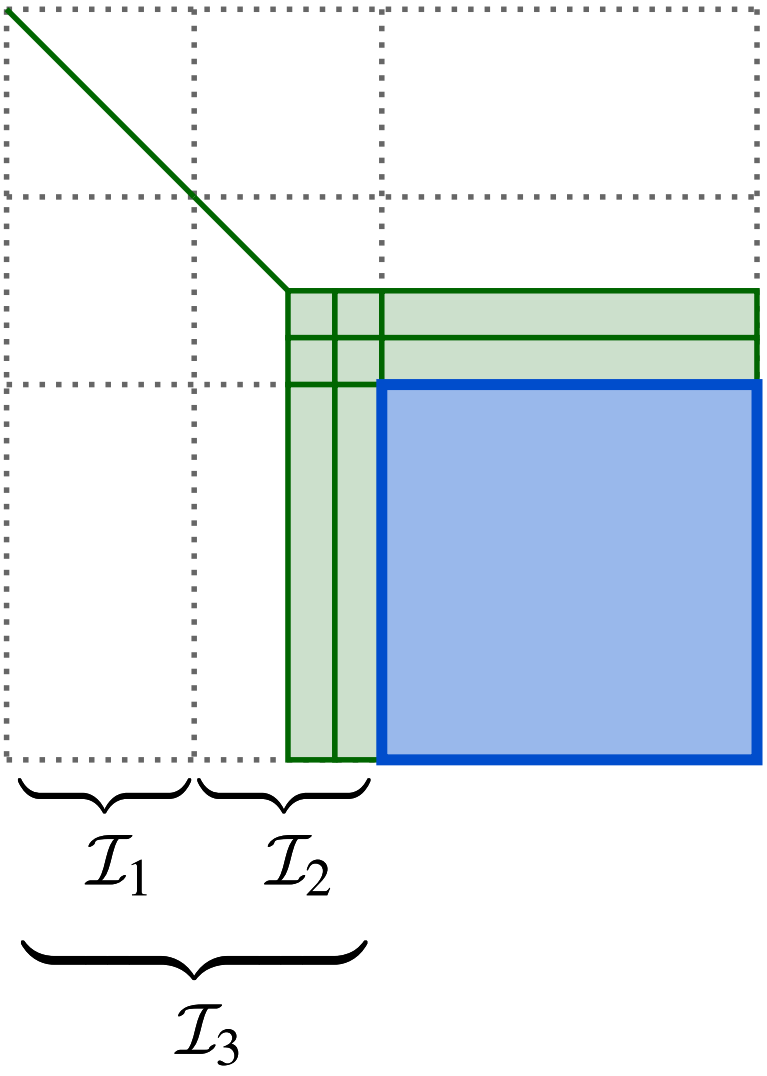
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H D (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$$

Structured factorization



HSS node 2:

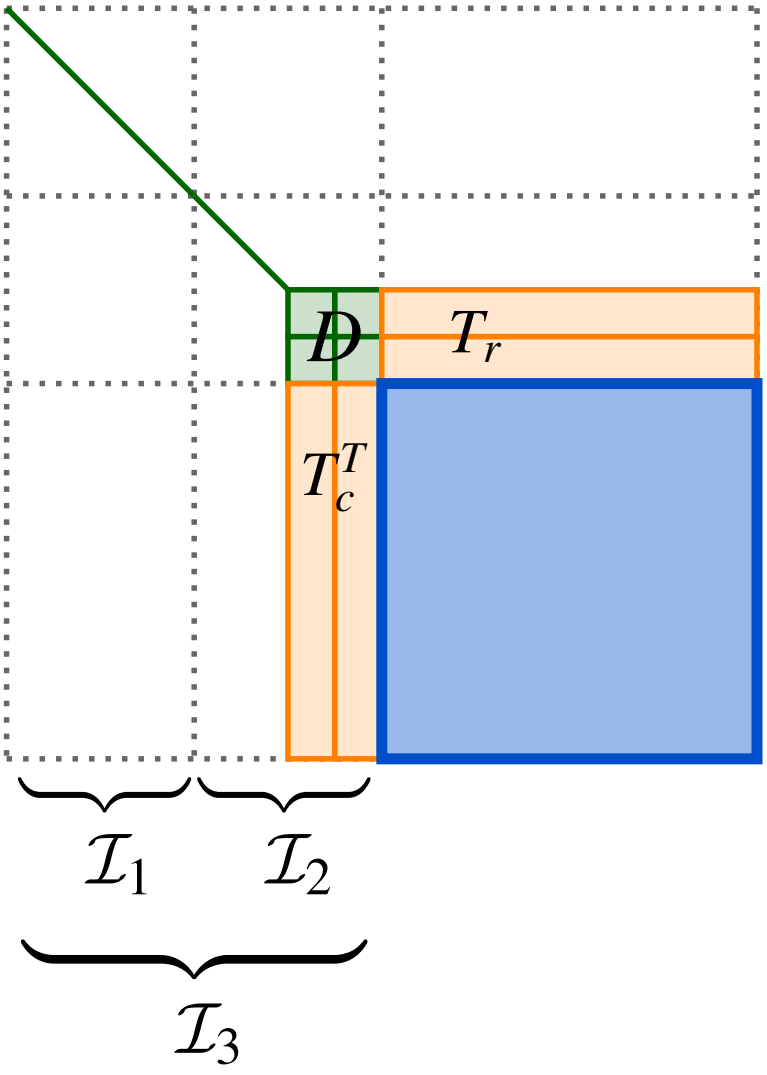
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H D (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$$

Structured factorization



HSS node 3:

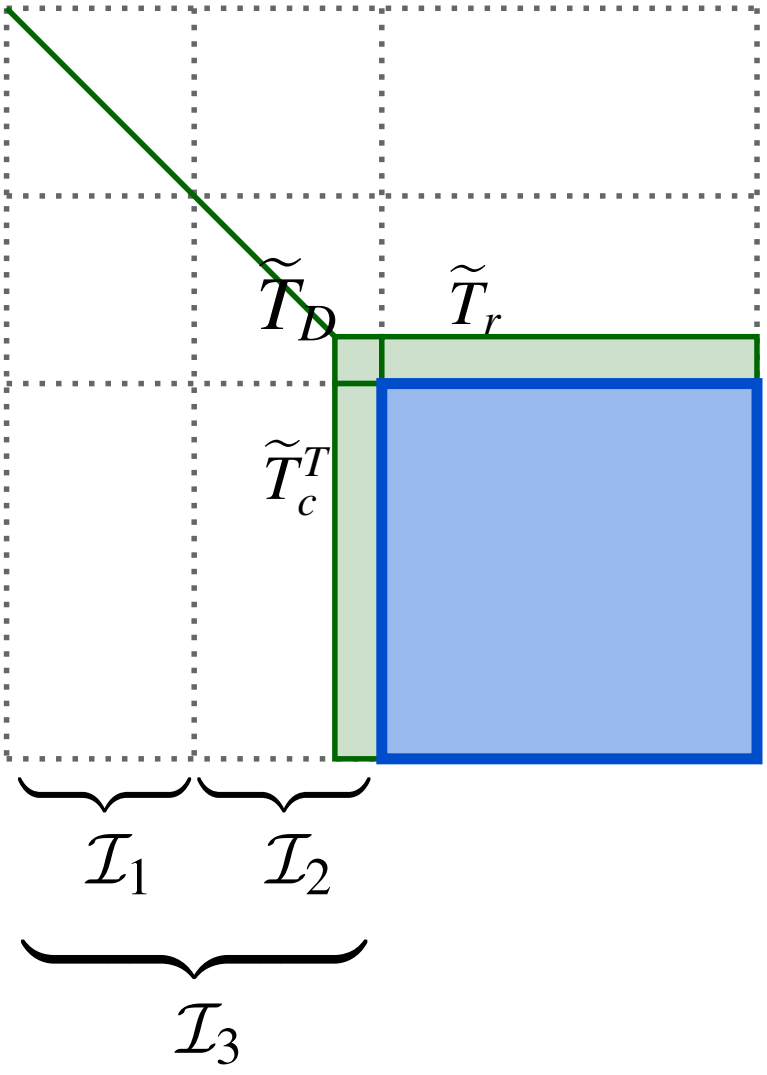
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H D (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$$

Structured factorization



HSS node 3:

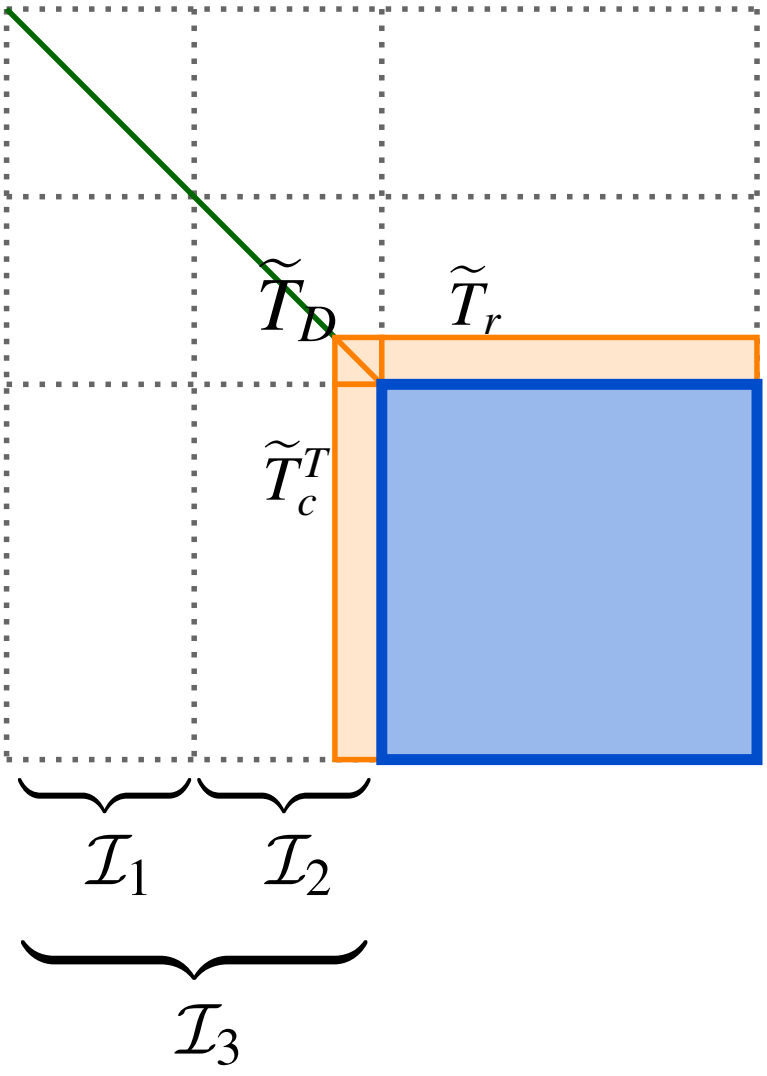
1. Approximate:
$(T_r , T_c^H) \approx V(\widetilde{T}_r , \widetilde{T}_c^H)$

2. Compl. basis with $V_\perp$:
$(V_\perp , V)$ is unitary

3. Change basis:
$\widetilde{D} = (V_\perp , V)^H D (V_\perp , V)$

4. Partially factorize $\widetilde{D}$:
$$\begin{pmatrix} D_L & \\ D_c & \widetilde{T}_D \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \widetilde{D}$$

Structured factorization



HSS node 3:

5. Partially factorize:

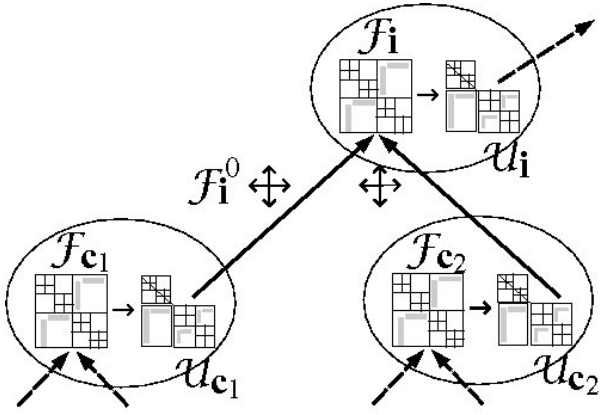
$$\begin{pmatrix} D_L & \\ D_c & \mathcal{U} \end{pmatrix} \begin{pmatrix} D_U & D_r \\ & I \end{pmatrix} = \begin{pmatrix} \widetilde{T}_D & \widetilde{T}_r \\ \widetilde{T}_c & \widetilde{T}_U \end{pmatrix}$$

Embedding HSS in multifrontal

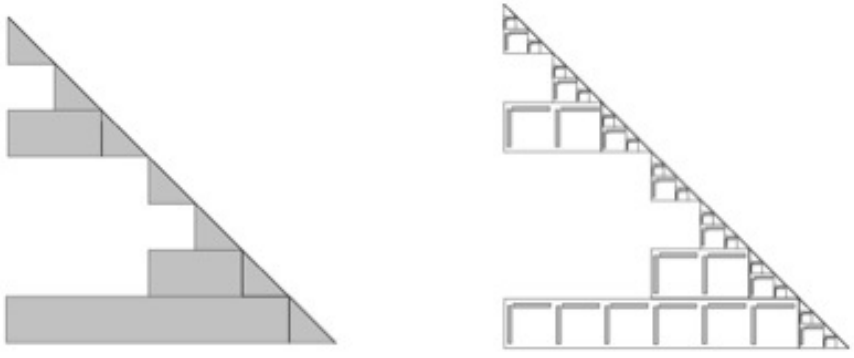
Approximate **Frontal** & **Update** matrices by HSS

Need following operations:

- **frontal HSS factorization of** F_i
- **extend-add of two HSS update matrices** U_i **and** U_j



Final Cholesky factor: Classical vs HSS-embedded



Theory

- Schur monotonicity for approximate Cholesky factorization
- conditioning analysis
- rank analysis for discretized PDEs

Schur monotonicity for approximate Cholesky $A = R^T R$

$$R = \begin{pmatrix} R_{1,1} & R_{1,2} & \cdots & R_{1,n} \\ & R_{2,2} & \cdots & R_{2,n} \\ & & \ddots & \vdots \\ & & & R_{n,n} \end{pmatrix}, \quad R \approx \widetilde{R} = \begin{pmatrix} R_1 & \tilde{R}_{1,2} & \cdots & \tilde{R}_{1,n} \\ & R_2 & \cdots & \tilde{R}_{2,n} \\ & & \ddots & \vdots \\ & & & R_n \end{pmatrix}$$

First approximation step: $R_1^T R_1 = A_{11}$ and $H_1 = D_1^{-T} A_{1,2:n}$

$$H_1 = \begin{pmatrix} U_1 & \widehat{U}_1 \end{pmatrix} \begin{pmatrix} \mathcal{Q}_1 & \widehat{\mathcal{Q}}_1 \end{pmatrix}^T, \quad H_1^T H_1 = \mathcal{Q}_1 \mathcal{Q}_1^T + \widehat{\mathcal{Q}}_1 \widehat{\mathcal{Q}}_1 e^T, \quad \|\widehat{\mathcal{Q}}_1\|_2 \leq \tau$$

Orthogonal dropping: $\widetilde{H}_1 = U_1 \mathcal{Q}_1 \longrightarrow \widetilde{H}_1^T (H_1 - \widetilde{H}_1) = 0$

Schur complement: $\mathcal{A}_1 = A_{2:n,2:n} - H_1^T H_1 = A_{2:n,2:n} - \mathcal{Q}_1 \mathcal{Q}_1^T - \widehat{\mathcal{Q}}_1 \widehat{\mathcal{Q}}_1^T$.

Approximate $\mathcal{A}_1$ by $\widetilde{\mathcal{A}}_1 = A_{2:n,2:n} - \mathcal{Q}_1 \mathcal{Q}_1^T = \mathcal{A}_1 + \widehat{\mathcal{Q}}_1 \widehat{\mathcal{Q}}_1^T = \mathcal{A}_1 + O(\tau^2)$

Nice Property: Successive Schur complements do not decrease in SPD sense
$\Rightarrow$ factorization is breakdown free

1. M. Gu, X.S. Li, P. Vassilevski, "Direction-Preserving and Schur-Monotonic Semiseparable Approximations of Symmetric Positive Definite Matrices", SIMAX, 31 (5), 2650-2664, 2010.
2. J. Xia, M. Gu, "Robust approximate Cholesky factorization of rank-structured symmetric positive definite matrices", SIMAX, 31 (5), 2899-2920, 2010.

Conditioning analysis when $R^T R$ as preconditioner (Napov)

- Goal: analyze spectral condition number $\kappa(R^{-T}AR^{-1})$
- Sketch: look at approximation after each step k of total l step; capture different approximation order:

$$B_k = \begin{pmatrix} \mathcal{R}_{11}^{(k)^T} & \\ \widetilde{\mathcal{R}}_{12}^{(k)^T} & \widetilde{S}_B^{(k)} \end{pmatrix} \begin{pmatrix} \mathcal{R}_{11}^{(k)} & \widetilde{\mathcal{R}}_{12}^{(k)} \\ & I \end{pmatrix}, \quad \widetilde{S}_B^{(k)} = A_{i_k+1:n,i_k+1:n} - \widetilde{\mathcal{R}}_{12}^{(k)^T}\widetilde{\mathcal{R}}_{12}^{(k)}$$

- SSS bound (sequential order):

$$\kappa(R^{-T}AR^{-1}) \leq \prod_{k=1}^{l} \frac{1+\gamma_k}{1-\gamma_k}, \quad \text{where } \gamma_k = \|(\mathcal{R}_{12}^{(k)} - \widetilde{\mathcal{R}}_{12}^{(k)})\widetilde{S}_B^{(k)^{-1/2}}\| < 1$$

- HSS bound: can be computed numerically using good estimates
- γ_k estimate: $\gamma_k \leq \left\|(\mathcal{R}_{12}^{(k)} - \widetilde{\mathcal{R}}_{12}^{(k)}\right\| \|A^{-1}\|^{1/2}$
  - can estimate $\|A^{-1}\|$ with a few iterations of Conjugate Gradient
- Adaptive threshold strategy based on γ_k estimate

A. Napov, "Conditioning analysis of incomplete Cholesky factorizations with orthogonal dropping", SIMAX, Vol. 34, No. 3, 1148-1173, 2013.

Rank analysis for some PDEs

Consider $n \times m$ grid, lexicographical (layer-by-layer) order gives rise to block tridiagonal matrix:

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & & & \\ A_{2,1} & A_{2,2} & \ddots & & \\ & \ddots & \ddots & A_{m-1,m} & \\ & & A_{m,m-1} & A_{m,m} \end{pmatrix}$$

, each $n \times n$ Schur compl. $S_{i+1} = A_{i,i} - A_{i,i-1} S_i^{-1} A_{i,i-1}$

Model problem (Chandrasekaran et al.):

$A_{i,i} = A_{j,j}, A_{i-1,i} = A_{j-1,j}, A_{i,i-1} = A_{j,j-1}$

- In 2D, ε-rank of the off-diagonal Hankel block is constant, for $n \to \infty$
- In 3D (k^3), ε-rank of the strip Hankel block is bounded by $O(k)$

Helmholtz equations with constant velocity (Engquist,Ying):

look at the Green's function of the Helmholtz operator.

- In 2D (k^2), ε-rank of the off-diagonal block bounded by $O(\log k)$
- In 3D (k^3), $O(k)$

1. S. Chandrasekaran, P. Dewilde, M. Gu, and N. Somasunderam, "On the Numerical Rank of the Off-Diagonal Blocks of Schur Complements of Discretized Elliptic PDEs", SIMAX, 31 (5), 2261-2290, 2010.
2. B. Engquist, L. Ying, "Sweeping preconditioner for the Helmholtz equation: Hierarchical matrix representation", Communications in Pure and Applied Mathematics 64 (2011).

Complexity of HSS-embedded multifrontal factorization

- With ND order, the intermediate Schur complements my have slightly higher ranks, but no more than twice:

$$A = \begin{pmatrix} A_{11} & & A_{33} \\ & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix}, \quad \text{Schur compl. } S = A_{33} - A_{31}A_{11}^{-1}A_{13} - A_{32}A_{22}^{-1}A_{32}$$

  Each contribution $-A_{31}A_{11}^{-1}A_{13}$ (or $-A_{32}A_{22}^{-1}A_{32}$) satisfies the off-diagonal rank bound, together, the off-diagonal rank bound of S is at most twice as that of layer-by-layer order.
- Given the rank bounds, can show the following cost of the HSS-MF factorization algorithm:

Problem		r	MF		HSS-MF	
			flops	fill	flops	fill
2D (k^2)	Elliptic	$O(1)$	$O(n^{3/2})$	$O(n \log n)$	$O(n \log n)$	$O(n \log \log n)$
	Helmholtz	$O(\log k)$				
3D (k^3)	Elliptic	$O(k)$	$O(n^2)$	$O(n^{4/3})$	$O(n^{4/3} \log n)$	$O(n \log n)$
	Helmholtz	$O(k)$				

J. Xia, "Efficient structured multifrontal factorization for general large sparse matrices", SISC, 35 (2), A832-A860, 2012.

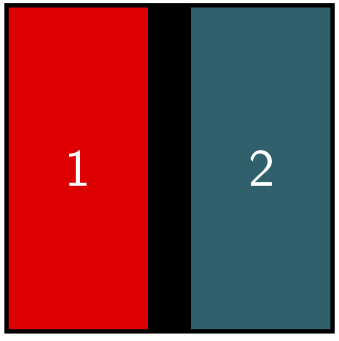
Warning: The constant prefactor may be large: $\sim O(100s)$ ($\sim$ 10 for classical)

Practice

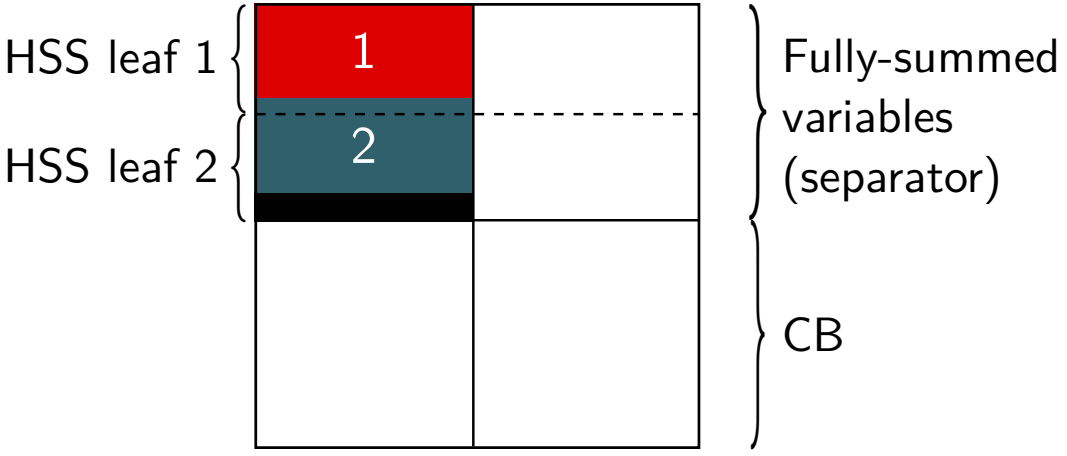
- ordering within separator
- parallelization
- preconditioning

Separator ordering: vertex-based vs edge-based

Vertex-based approach:

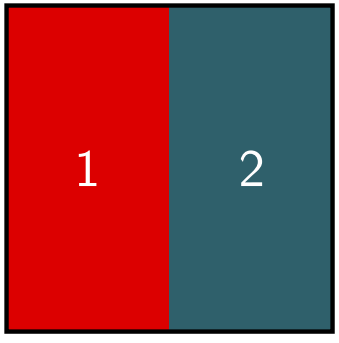


2D separator from a 3D domain.

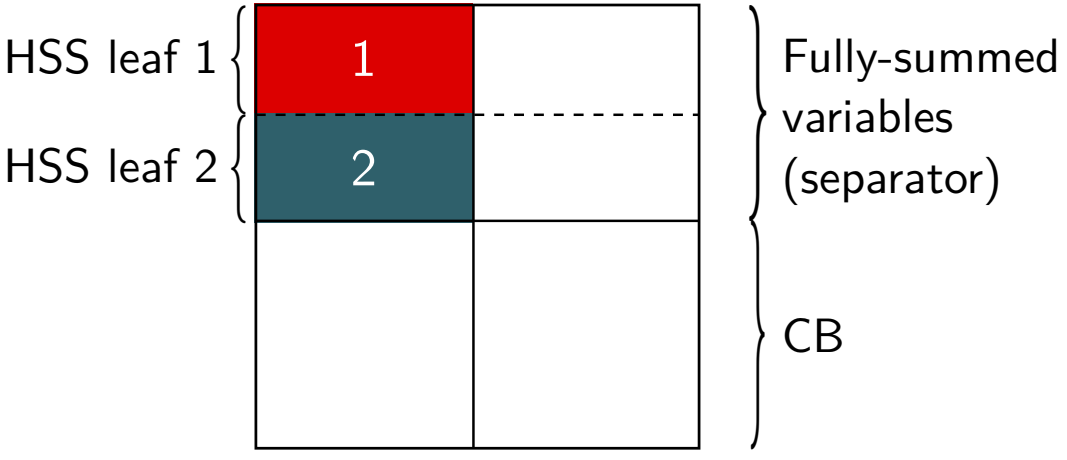


Frontal matrix.

Edge-based approach:



2D separator from a 3D domain.

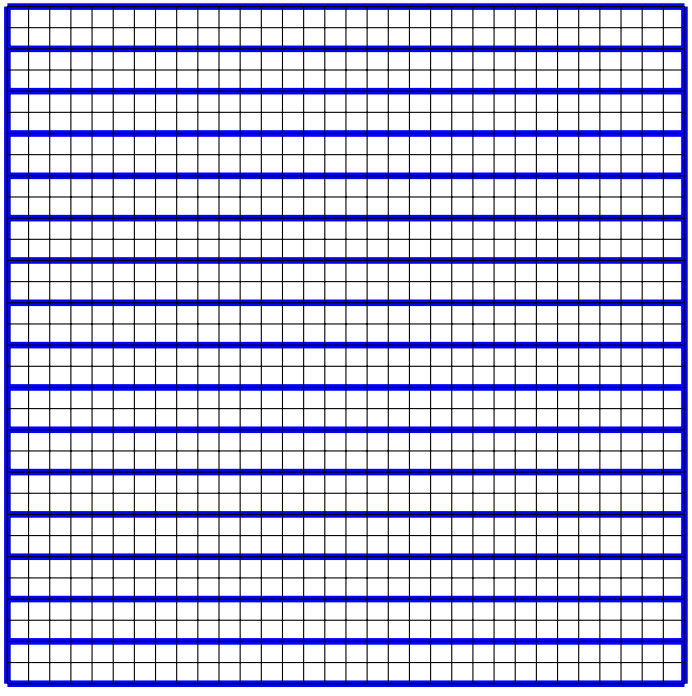


Frontal matrix.

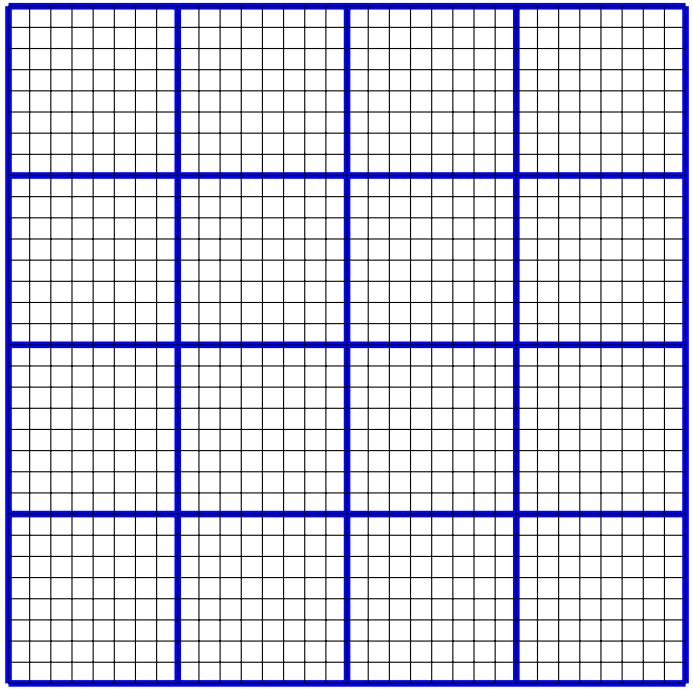
An edge-based ordering allows us to simply match parts of the separators with nodes of the HSS tree.

Ordering of separators: shape

In order to ensure some kind of admissibility condition, parts should have a small diameter.



Large diameters.



Small diameters.

For simplicity, we divide the separator into square blocks (chessboard).

Ordering of separators: ordering of blocks

In the HSS compression stage, blocks are merged two-by-two following a tree flow. Merged blocks should also have small diameter, thus the partioning should have some recursive property.

13	14	15	16
9	10	11	12
5	6	7	8
1	2	3	4

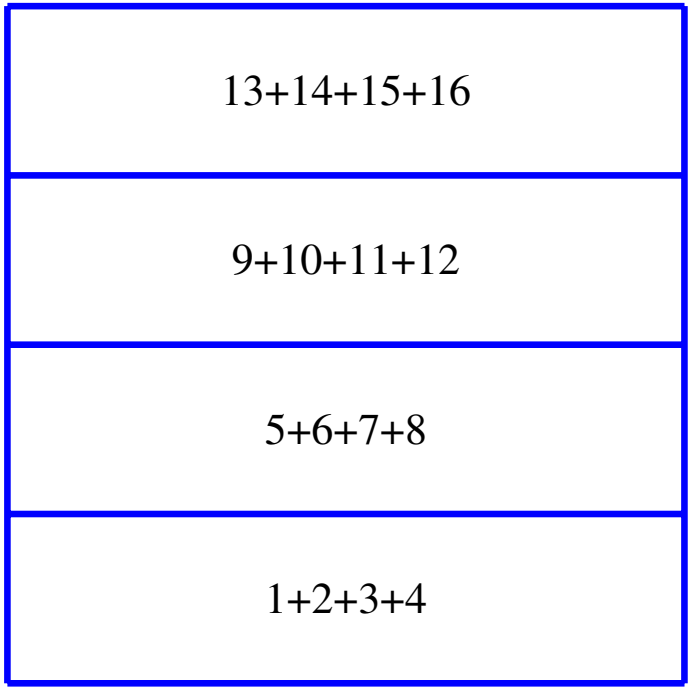
10	11	15	16
9	10	13	14
3	4	7	8
1	2	5	6

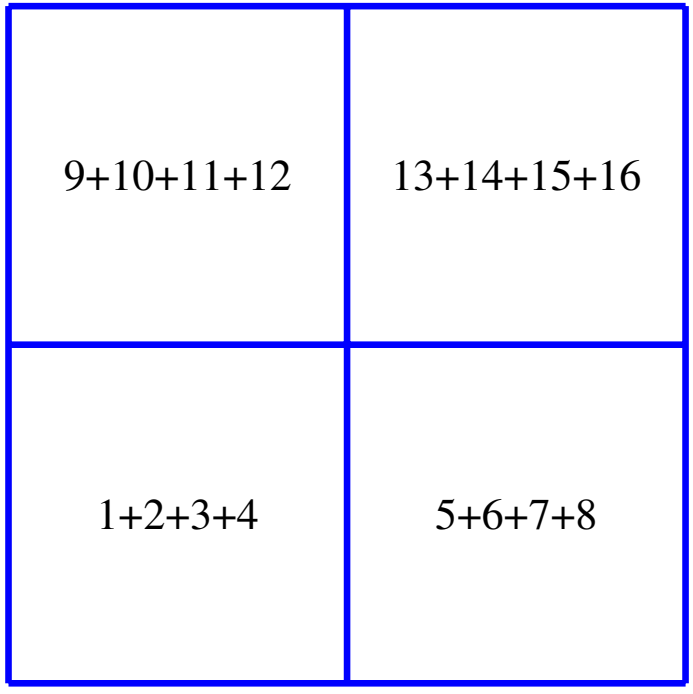
Leaf level.

We use an edge-based Nested Dissection (we cut the domain into squares, and order these squares using ND/Morton ordering).

Ordering of separators: ordering of blocks

In the HSS compression stage, blocks are merged two-by-two following a tree flow. Merged blocks should also have small diameter, thus the partioning should have some recursive property.





Two levels above leaves: blocks are merged four-by-four.

We use an edge-based Nested Dissection (we cut the domain into squares, and order these squares using ND/Morton ordering).

Results - topmost separator

Topmost separator of a 200^3 domain (200×200 plane). We compare:

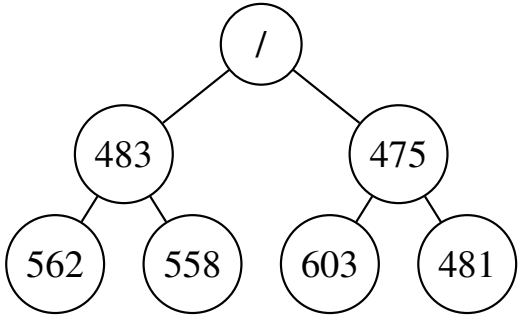
VND: vertex-based ND.

Nat: square blocks ordering in natural/lexicographic order.

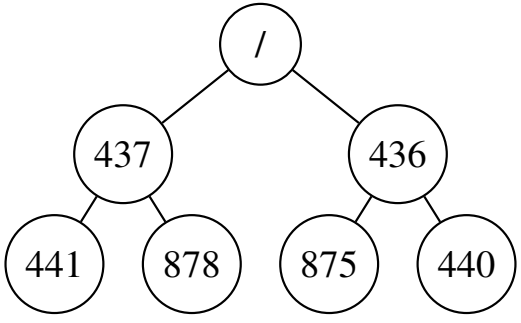
END: edge-based ND: square blocks ordered in ND.

	VND	Nat	END
Total HSS time (s)	55.0	51.8	32.3
Max rank	731	893	646
Min time in RRQR (s)	15.2	20.3	11.0
Max time in RRQR (s)	53.0	50.2	30.7

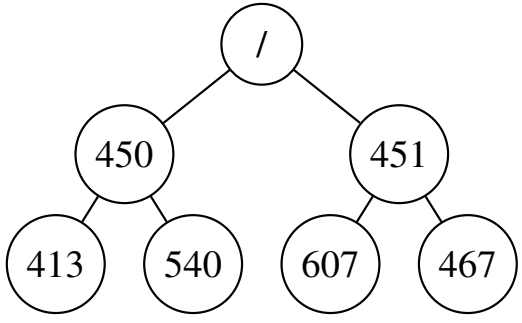
Ranks at the top of HSS trees:



VND.



Nat.



END.

END yields lower rank and better balance of ranks.

Results - complete problem

Helmholtz equations with PML boundary

$$\left(-\Delta - \frac{\omega}{v(x)^2}\right) u(x,\omega) = s(x,\omega)$$

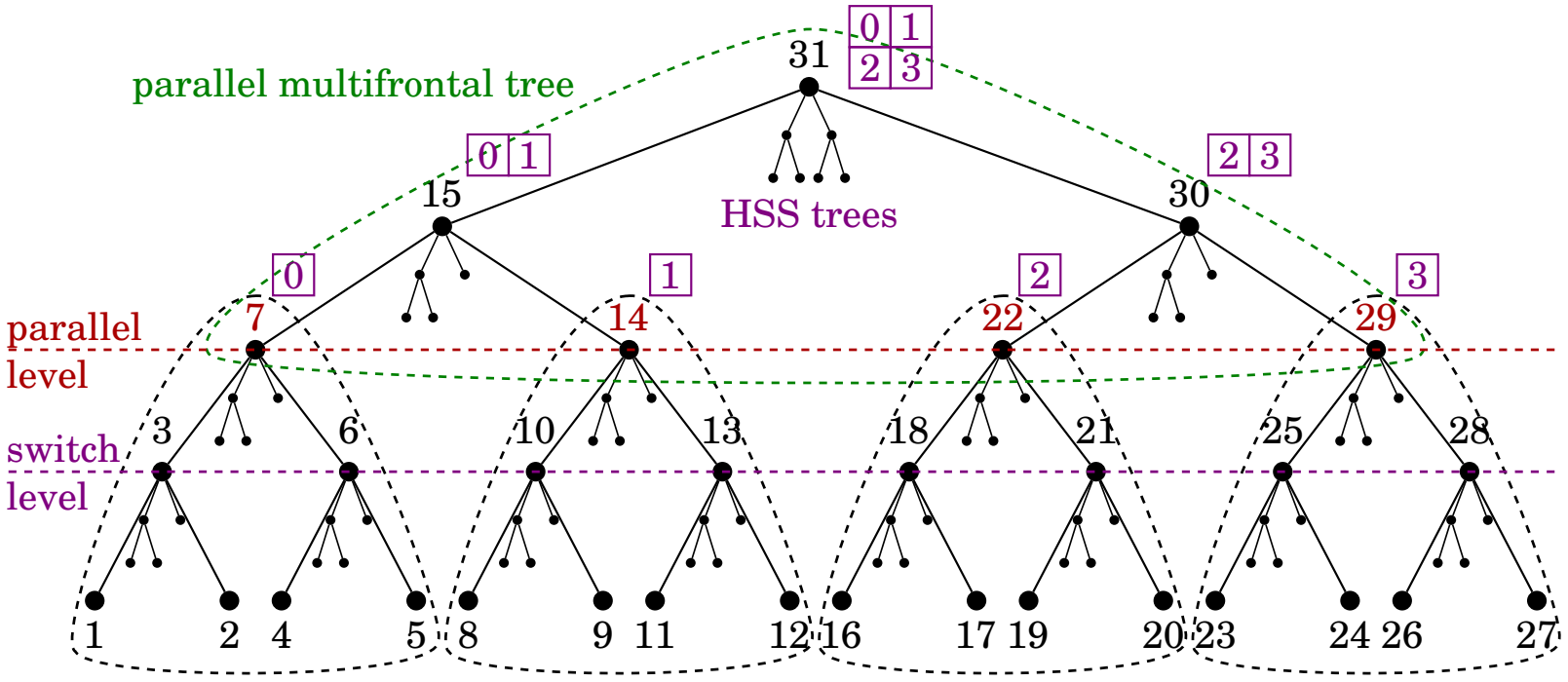
On the complete problem, with 256 cores and HSS compression at the 8 topmost levels of the tree:

	VND	Nat	END
Total factorization time (s)	984.8	978.5	938.0
Max rank	865	893	868
Min time in RRQR (s)	304.8	322.1	310.9
Max time in RRQR (s)	674.9	683.8	654.7

END: marginal (5%) improvement in run time but better workload balance, so hopefully more potential for strong scaling.

Parallelization: two types of tree-based parallelism

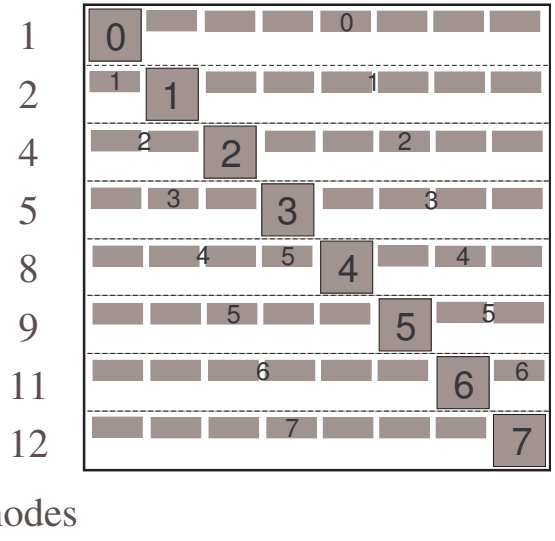
- **Outer tree**: separator tree for multifrontal factorization
- **Inner tree**: HSS tree at each internal separator node



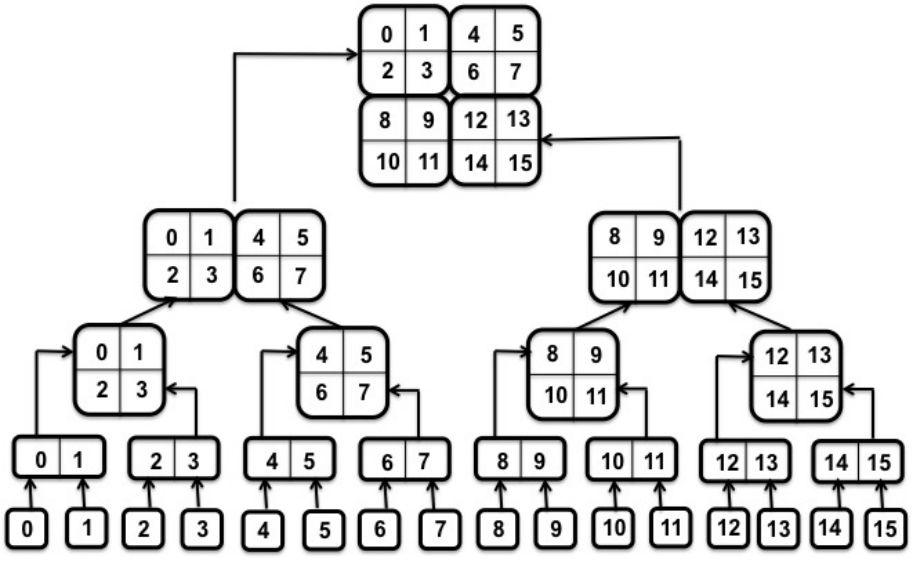
Parallelization strategy for HSS

- Work along the HSS tree level-wise, bottom up.
- 2D block-cyclic distribution at each tree node ($\#Levels = \log P$)
  - each P_i works on the bottom level leaf node i
  - every 2 processes cooperate on a Level 2 node
  - every 4 processes cooperate on a Level 3 node

Level 1: local $F_i = U_i \tilde{F}_i$

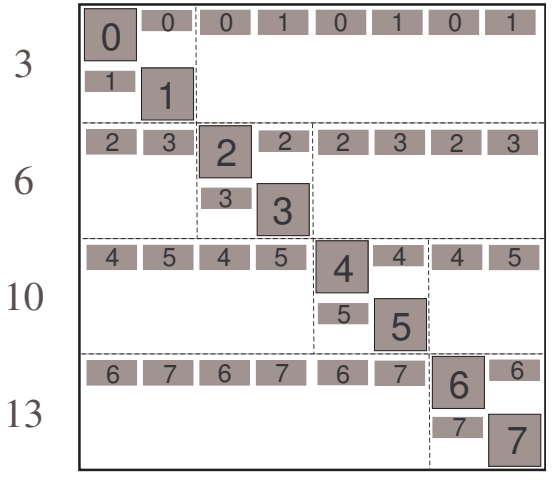


2D-procs mapped to HSS tree nodes

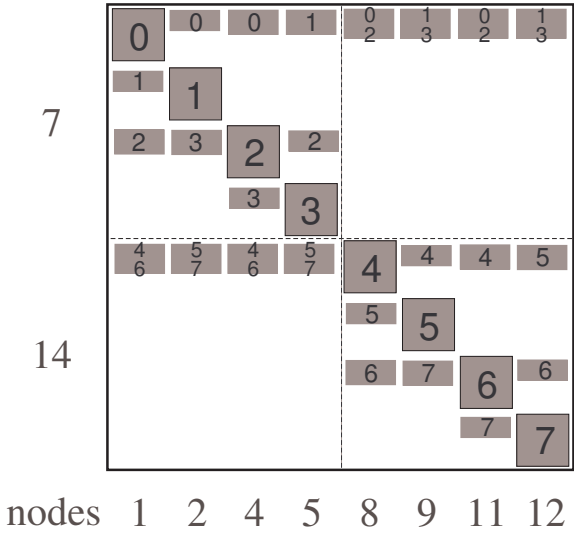


Parallel row compression (cont)

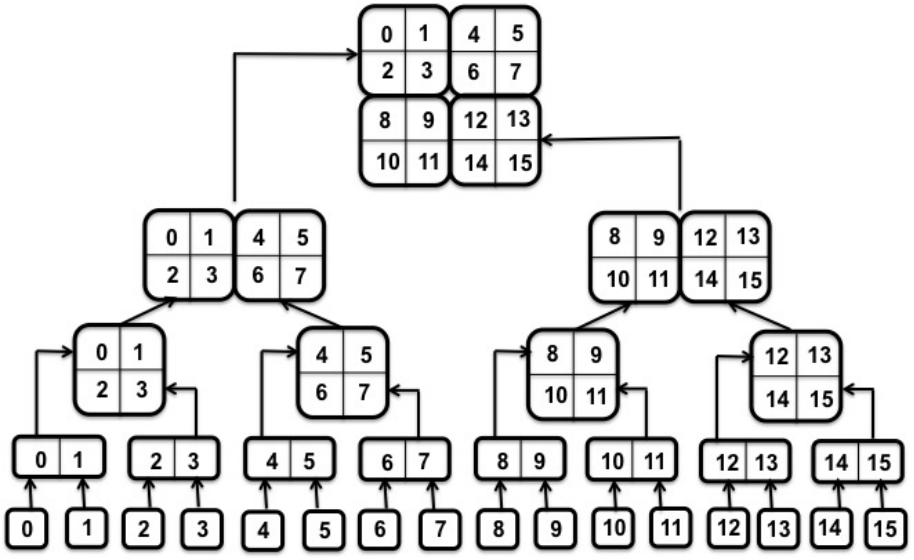
- Level 2 (2-cores/group)



- Level 3 (4-cores/group)



2D-procs mapped to HSS tree nodes



Summary of parallel row compression & complexity

- Each step involves RRQR and redistribution
  - pairwise exchange
- Flop count: $\mathcal{O}(\frac{r N^2}{P})$
- Communication in row compression:

  $\#msg = \mathcal{O}(r \log^2 P)$

  $\#words = \mathcal{O}(r N \log P)$

  (assume no overlap between comm. and comp.)

Arithmetic Intensity: $\mathcal{O}(\frac{N}{P \log P})$

(c.f. ScaLAPACK dense LU: $\mathcal{O}(\frac{N}{\sqrt{P}})$)

Parallel test

- Cray XE6 (hopper at NERSC)
- Example: Helmholtz equation with PML boundary

$$\left(-\Delta - \frac{\omega^2}{v(x)^2}\right) u(x,\omega) = s(x,\omega), \tag{1}$$

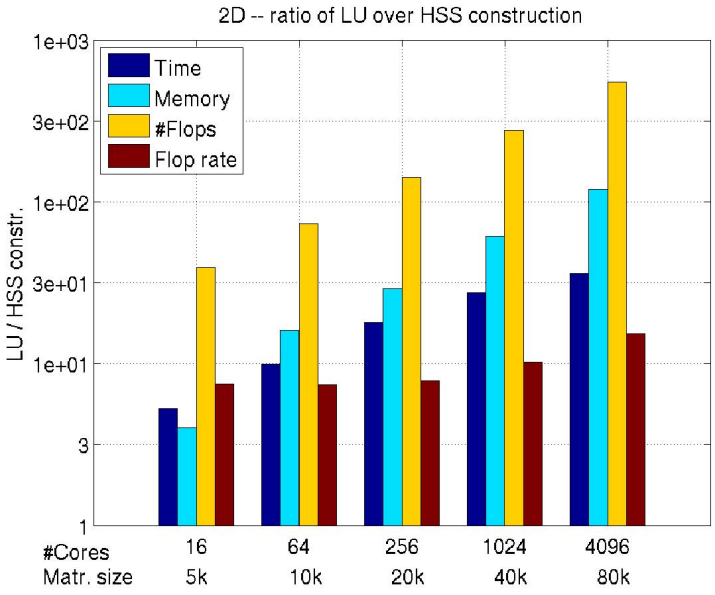
Δ: Laplacian
ω: angular frequency
$v(x)$: seismic velocity field
$u(x,\omega)$: time-harmonic wavefield solution

- FD discretized linear system:
  - Complex, pattern-symmetric, non-Hermitian,
  - Indefinite, ill-conditioned

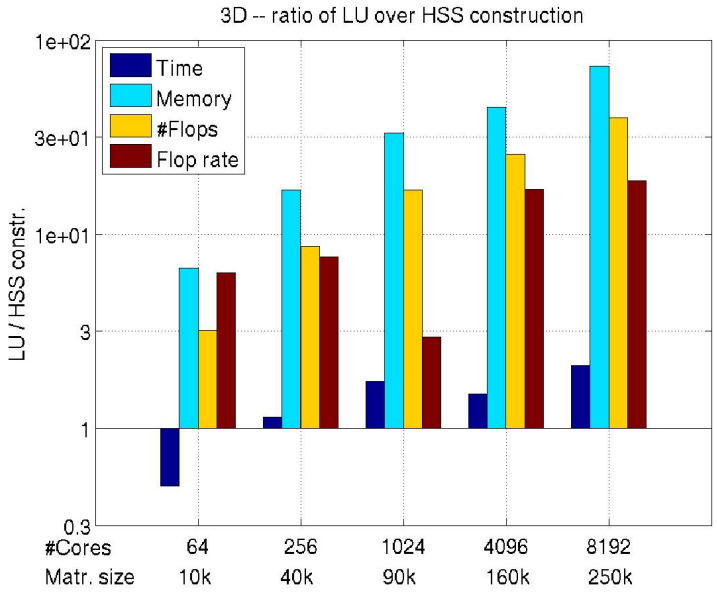
Parallel HSS performance

- HSS constrcution on the last Schur complement corresp. to the top separator.

Performance ratio of LU over HSS:



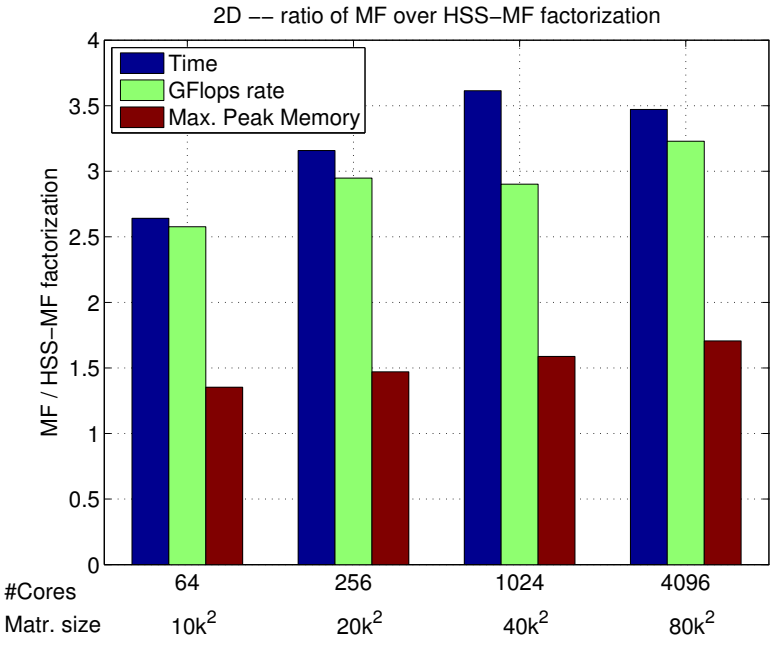
(a) 2D, max_rank=7



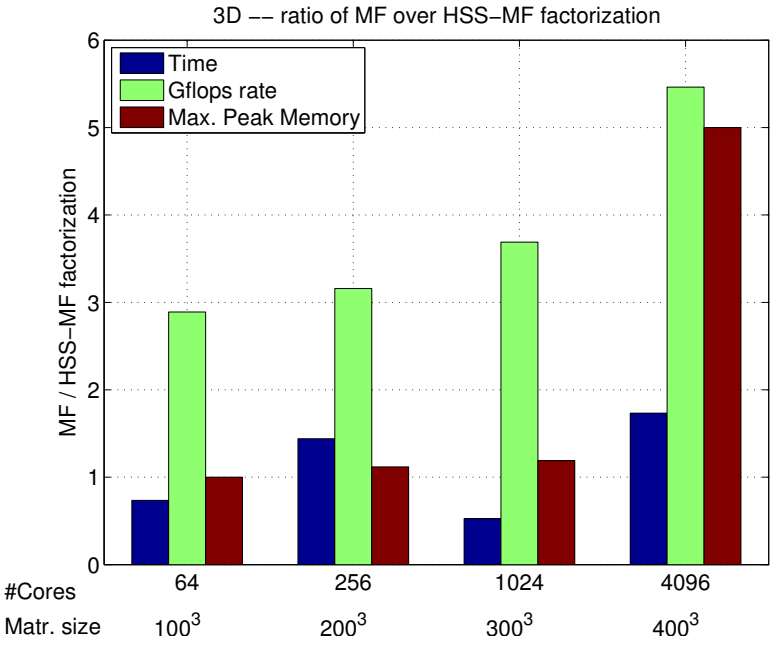
(b) 3D, max_rank=848

S. Wang, X.S. Li, J. Xia, and M.V. de Hoop, "Efficient scalable algorithms for solving linear systems with hierarchically semiseparable matrices", SISC, Nov. 2012. (revised)

Parallel HSS-MF performance



(c) 2D Helmholtz, 10Hz



(d) 3D Helmholtz, 5Hz

HSS-MF succeeded with 600^3 on 16,384 cores, while MF failed.

S. Wang, X.S. Li, F.-H. Rouet, J. Xia, and M. de Hoop, "A Parallel Geometric Multifrontal Solver Using Hierarchically Semiseparable Structure", ACM TOMS, June 2013. (in submission)

Sparse results - 2D problems

2D Helmholtz problems on square grids (mesh size k, $N = k^2$), 10 Hz.

k		10,000	20,000	40,000	80,000
P		64	256	1,024	4,096
MF	Factorization (s)	258.6	544.8	1175.8	2288.5
	Gflops/s	507.3	2109.3	8185.6	31706.9
	Solution+refinement (s)	10.4	10.8	11.5	11.6
	Factors size (GB)	120.1	526.7	2291.2	9903.7
	Max. peak (GB)	2.3	2.5	2.7	2.9
	Communication volume (GB)	136.2	1202.5	9908.1	79648.4
HSS	HSS+ULV (s)	97.9	172.5	325.3	659.3
	Gflops/s	196.9	715.6	2820.7	9820.6
	Solution+refinement (s)	20.2	55.4	61.4	115.8
	Steps	3	3	9	9
	Factors size (GB)	66.2	267.7	1333.2	4572.3
	Max. peak (GB)	1.7	1.7	1.7	1.7
	Communication volume (GB)	74.2	573.8	4393.4	41955.8
	HSS rank	258	503	1013	2015
	$\|\|x - x_{\text{MF}}\|\|/\|\|x_{\text{MF}}\|\|$	1.5×10^{-5}	2.2×10^{-5}	3.1×10^{-5}	3.5×10^{-6}
	$\max_i \frac{\lvert Ax-b\rvert_i}{(\lvert A\rvert\lvert x\rvert+\lvert b\rvert)_i}$	7.1×10^{-6}	1.0×10^{-5}	2.0×10^{-6}	3.5×10^{-6}

Results - 3D problems

3D Helmholtz problems on cubic grids (mesh size k, $N = k^3$), 5 Hz.

k		100	200	300	400
P		64	256	1,024	4,096
MF	Factorization (s)	88.4	1528.0	1175.8	6371.6
	Gflops/s	600.6	2275.7	9505.6	35477.3
	Solution+refinement (s)	0.6	2.2	3.5	4.8
	Factors size (GB)	16.6	280.0	1450.1	4636.1
	Max. peak (GB)	0.5	1.9	2.5	2.0
	Communication volume (GB)	83.1	2724.7	26867.8	165299.3
HSS	HSS+ULV (s)	120.4	1061.3	2233.8	3676.5
	Gflops/s	207.8	720.4	2576.6	6494.8
	Solution+refinement (s)	2.3	8.2	31.5	182.8
	Steps	4	5	10	6
	Factors size (GB)	10.7	112.9	434.3	845.3
	Max. peak (GB)	0.5	1.7	2.1	0.4
	Communication volume (GB)	93.6	2241.2	18621.1	143300.0
	HSS rank	481	925	1391	1860
	$\|\|x - x_{\text{MF}}\|\|/\|\|x_{\text{MF}}\|\|$	6.2×10^{-6}	9.4×10^{-7}	1.1×10^{-6}	1.7×10^{-6}
	$\max_i \frac{\lvert Ax-b\rvert_i}{(\lvert A\rvert\lvert x\rvert+\lvert b\rvert)_i}$	1.5×10^{-7}	5.7×10^{-7}	9.7×10^{-7}	3.7×10^{-6}

Preconditioning results

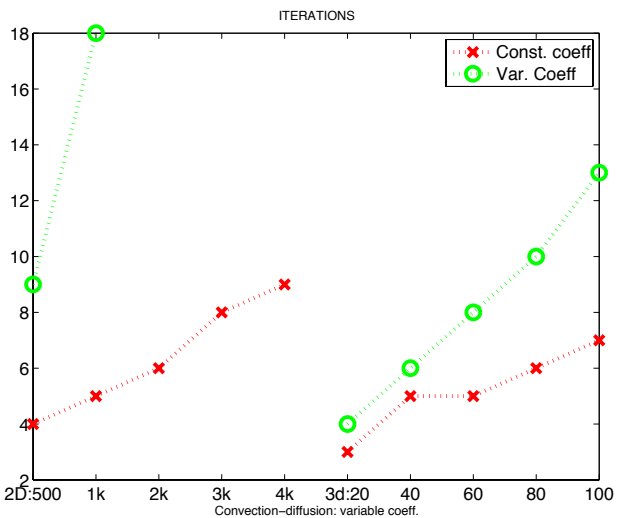
- Test matrices: 2D & 3D
  - model problems
  - convection-diffusion: constant coefficient, variable coefficient
  - curl-curl edge elements (Nedelec elements)
  - Helmholtz
  - general matrices
- RHS $= (1, 1, \ldots)^T$
- GMRES(30)
  - right preconditioner
  - initial $x_0 = (0, 0, \ldots)^T$
  - stopping criterion: $\frac{\|r_k\|_2}{\|b\|_2} \leq 10^{-6}$

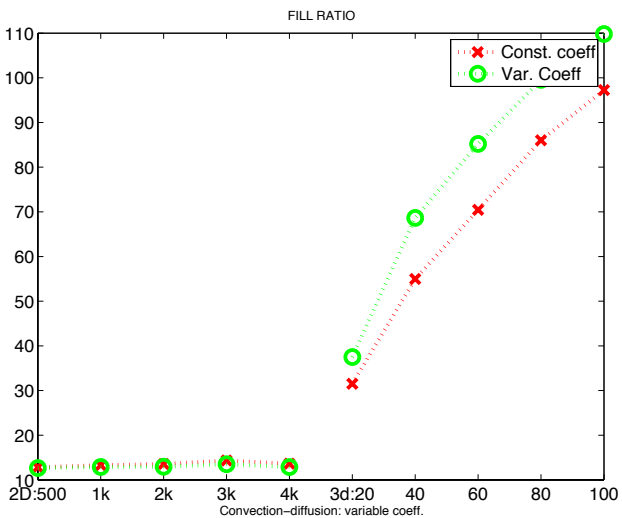
Convection-diffusion

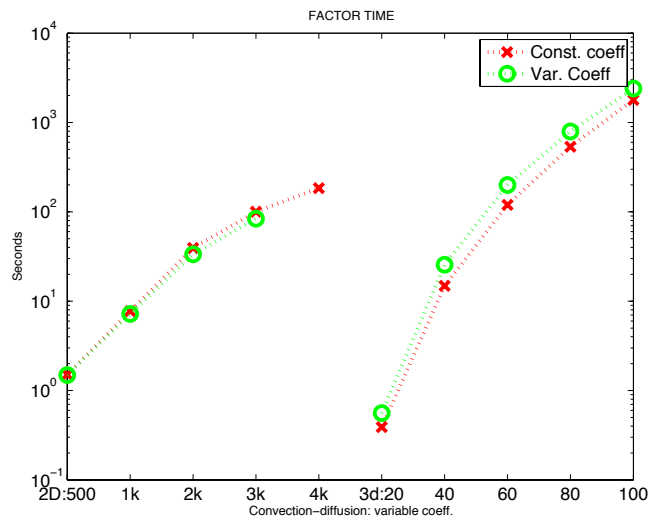
$$-\nu\Delta u + \mathbf{v}\cdot\nabla u = \mathbf{b} \quad \text{on } \Omega\,.$$

$\mathbf{v} = \ldots$

r	constant coeff.	variable coeff.
2D	$(1/\sqrt{2} \quad 1/\sqrt{2})$	$(\ x(1-x)(2y-1) \quad y(1-y)(2x-1)\ )$
3D	$(1/2 \quad 1/2 \quad 1/\sqrt{2})$	$(\ x(1-x)(2y-1)z \quad y(y-1)(2x-1) \quad (2x-1)(2y-1)z(z-1)\ )$

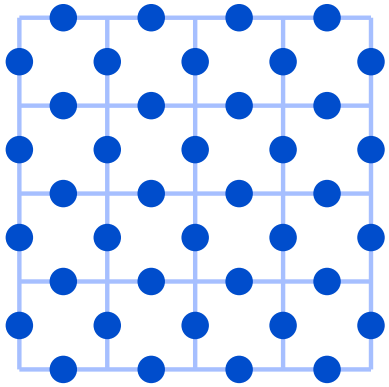






Curl-curl edge element (Nedelec element)

$\nabla \times \nabla \times u + \beta \ = \mathbf{b}$ on Ω
$\tau = 10^{-8}$



mesh size	HSS-rank	fill-ratio	factor (s)	Its	GMRES (s)
500^2	59	14.8e	5.8	2	0.5
1000^2	52	14.9	25.0	3	3.3
2000^2	60	14.9	106.1	3	13.3
3000^2	50	14.4	114.6	5	48.7
20^3	388	78.9	6.7	2	0.1
40^3	824	125.6	261.5	3	1.7
60^3	804	156.1	2055.8	3	7.4

Helmholtz

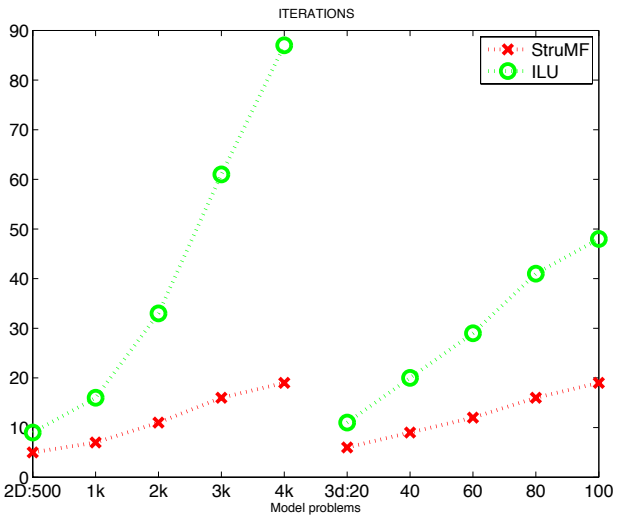
$$\left(-\Delta - \frac{\omega}{v(x)^2}\right) u(x,\omega) = s(x,\omega)$$

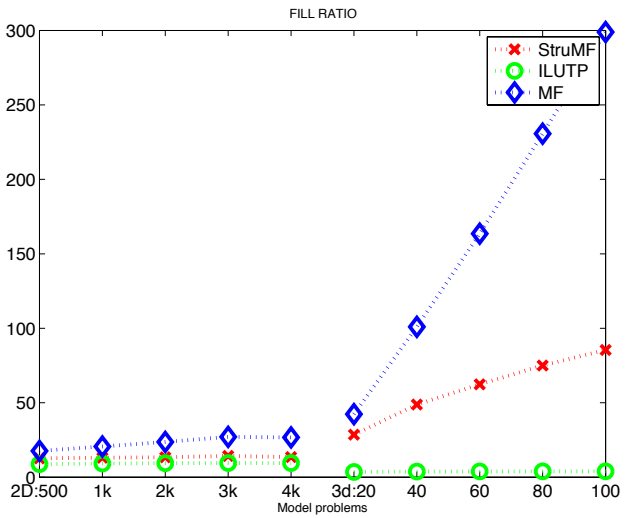
$\tau = 10^{-4}$

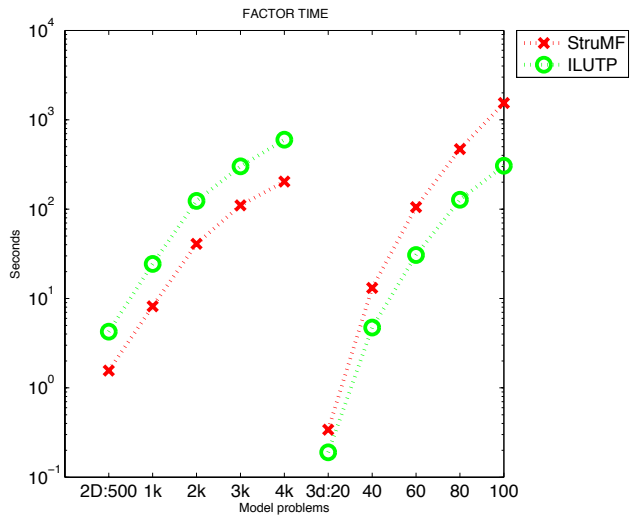
mesh size	HSS-rank	fill-ratio	factor (s)	Its	GMRES (s)
500^2	85	8.8	8.6	4	1.6
1000^2	210	9.5	53.1	4	6.5
2000^2	229	9.7	307.1	71	500.1
3000^2	380	10.0	950.2	139	2464.1
20^3	275	13.3	2.4	3	0.1
40^3	533	27.0	151.9	3	1.1
60^3	1039	38.8	1434.6	3	5.3
80^3	1167	47.3	7708.1	3	16.8

Model problems: $\Delta u = f$

- Compare to ILU in SuperLU (Li/Shao 10)
  - supernode-based ILUTP, threshold, partial pivoting
  - 10^{-4} for HSS trunction, and ILU threshold





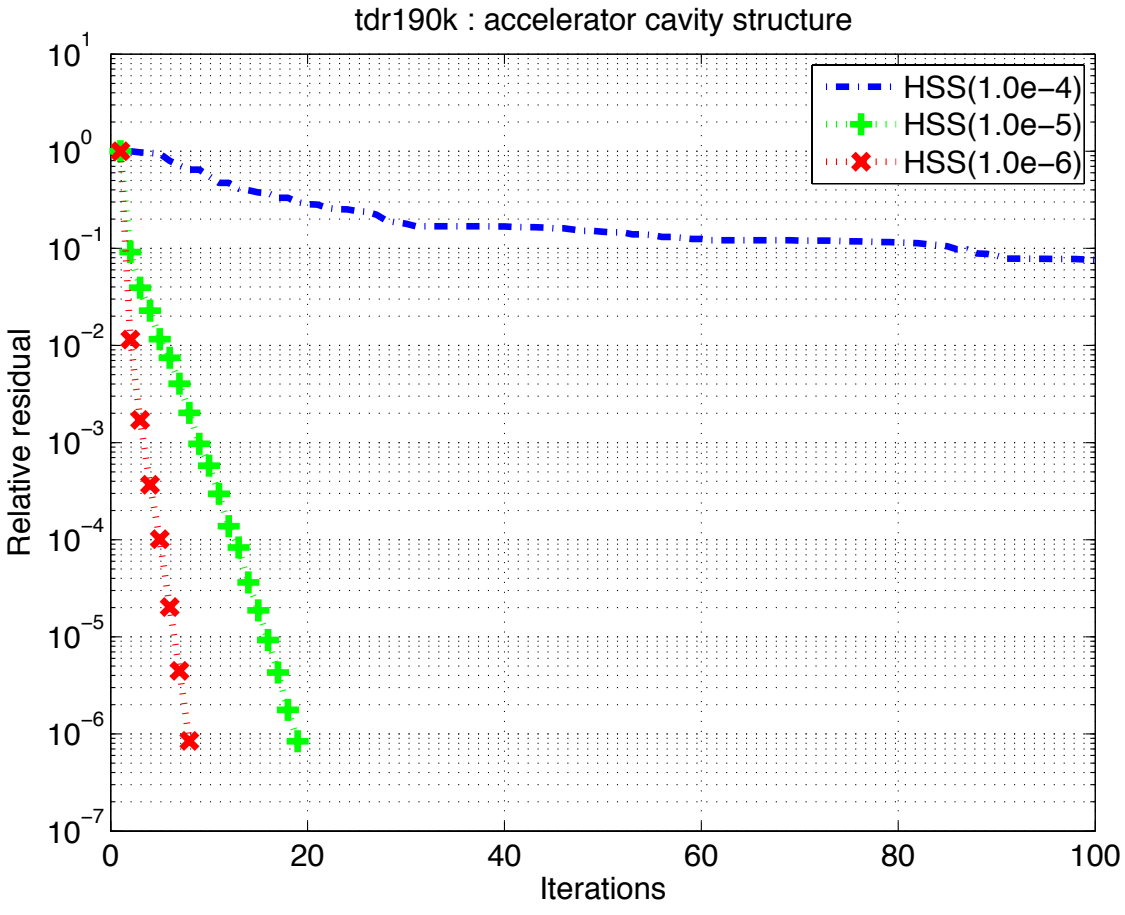


General matrices

Matr.	Descr.	N	HSS-rank	Fill-ratio		Factor (s)		Its	
				HSS	ILU	HSS	ILU	HSS	ILU
add32	circuit	4,690	0	2.1	1.3	0.01	0.01	1	2
mchln85ks17	car tire	84,180	948	13.5	12.3	133.8	216.1	4	39
mhd500	plasma	250,000	100	11.6	15.6	2.5	7.9	2	8
poli_large	economic	15,575	64	4.8	1.6	0.04	0.02	1	2
stomach	bio eng.	213,360	92	12.1	2.9	13.8	18.7	2	2
tdr190k	accelerator	1,100242	596	14.1	–	629.2	–	7	–
torso3	bio eng.	259,156	136	22.6	2.4	86.7	63.7	2	2
utm5940	TOKAMAK	5,940	123	6.7	8.0	0.1	0.16	3	15
wang4	device	26,068	385	45.3	23.1	4.4	6.4	3	4

HSS truncation tolerance

tdr190k – Maxwell equations in frequency domain, eigenvalue problem



Summary

- More theory has been developed
- In practice: very promising for large problems, large machines
  - demonstrated that it is implementable in parallel, with reduced communication
- Compare to ILU preconditioner
  - breakdown free
  - More parallel
  - Dropping operation may be more expensive (row/col vs. entry-wise in ILU)

Future work

- Parallel low-rank factorization using randomized sampling
- Analyze communication lower bound for HSS-structured sparse factorization
  - Classical sparse Cholesky (Gupta et al.'97):
    3D model problem: $\mathcal{O}(\frac{N^{4/3}}{\sqrt{P}})$ COMM-Volume
- Black-box preconditioner?
  - Apply to broader simulation problems: accelerator, fusion, etc.
  - compare to other preconditioners, e.g., ILU, multigrid
- Precondition the Communication-Avoiding Krylov algorithms [with Demmel's group]
- Compare to sparse solvers using $\mathcal{H}$-matrix [Weisbecker et al., Ying et al.]
- Resilience at extreme scale

Rank-revealing via randomized sampling

1. Pick random matrix $\Omega_{n\times(k+p)}$, p small, e.g. 10
2. Sample matrix $S = A\Omega$, with slight oversampling p
3. Compute $Q = \text{ON-basis}(S)$

- **accuracy:** with *probability* $\geq 1 - 6 \cdot p^{-p}$,
  $\|A - QQ^*A\| \leq [1 + 11\sqrt{k+p} \cdot \sqrt{\min\{m,n\}}]\sigma_{k+1}$
- **cost:** $O(kmn)$

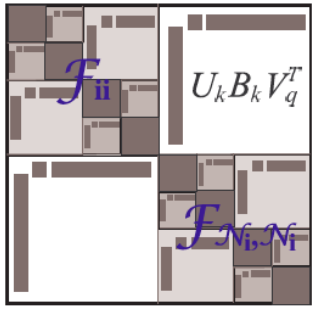
Randomized sampling simplies extend-add

- HSS construction via RS [Martinsson'11]
  - SRRQR repeatedly applied to matrices with **reduced column dimention**

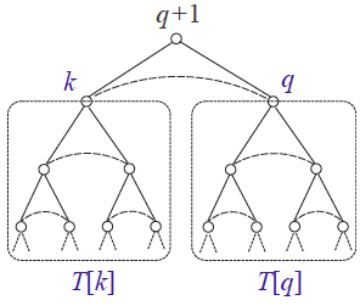
- Multifrontal HSS via RS
  1. Compress frontal $\mathcal{F}_i$:
     Form sample matrix $Y_i = \mathcal{F}_i X_i$, where $X_i = (X_i^{(1)} \quad X_i^{(2)})^T$ random,
     Construct HSS of $\mathcal{F}_i$ with help of Y_i
  2. ULV factorize $\mathcal{F}_i(1,1)$
  3. HSS approximation of $\mathcal{U}_i$
  4. Form sample matrix $Z_i = \mathcal{U}_i X_i^{(2)}$, where $X_i^{(2)}$ is a submatrix of X_i corresponding to $\mathcal{U}_i$
  5. **extend-add of sample matrices to parent:**
     $Y_p \equiv \mathcal{F}_p X_p = (A_p X_p) \oplus Z_i \oplus Z_j$

At Child

- Compression of F_i

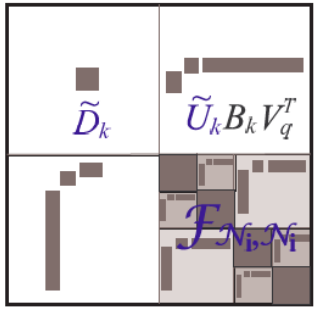


(i) HSS form for $\mathcal{F}_i$

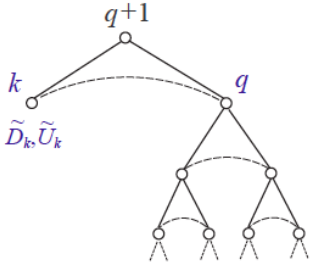


(ii) HSS tree T for $\mathcal{F}_i$ and subtrees $T[k]$ and $T[q]$

- Partial elimination of F_i



(i) After the partial ULV factorization of $\mathcal{F}_{\mathrm{ii}}$

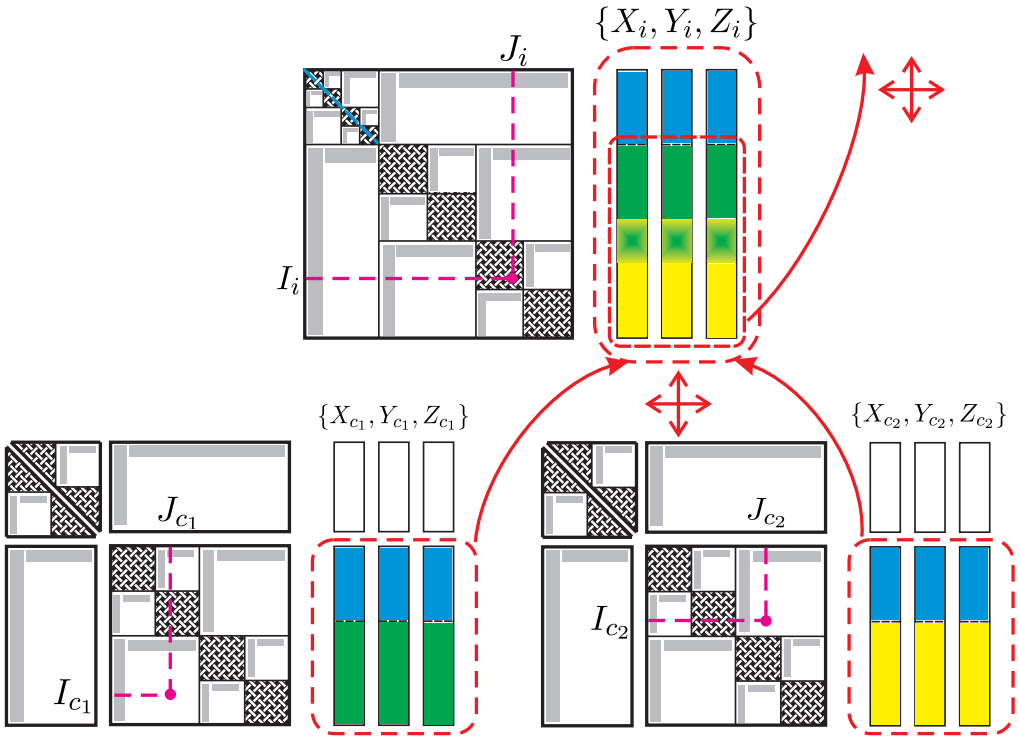


(ii) The resulting tree

- Compute update matrix U_i: $\mathcal{U}_i = \mathcal{F}_i(2,2) - U_q B_k^T (\tilde{U}_k^T \tilde{D}_k^{-1} \tilde{U}_k) B_k U_q^T$
  - fast low-rank update: obtain $\mathcal{U}_i$ generators **directly** from $\mathcal{F}_i(2,2)$ generators

At Parent: extend-add of sample matrices from children

RS simplies extend-add: $Y_p \equiv F_p X_p = (A_p X_p) \oplus Z_i \oplus Z_j$



$$
\begin{aligned}
Z_i &= \mathcal{U}_i X_i^{(2)} = \mathcal{F}_i(2,2) X_i^{(2)} - U_q B_k^T (\tilde{U}_k^T \tilde{D}_k^{-1} \tilde{U}_k) B_k U_q^T X_i^{(2)} \\
&= Y_i^{(2)} - U_q B_k^T U_k^T X_i^{(1)} - U_q B_k^T (\tilde{U}_k^T \tilde{D}_k^{-1} \tilde{U}_k) B_k U_q^T X_i^{(2)} \\
&= Y_i^{(2)} - U_q B_k^T \left[U_k^T X_i^{(1)} + (\tilde{U}_k^T \tilde{D}_k^{-1} \tilde{U}_k) B_k U_q^T X_i^{(2)} \right]
\end{aligned}
$$

References

- M. Bebendorf, “Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems”, Lecture Notes in Computational Science and Engineering, Springer, 2008.
- M. Gu, X.S. Li, P. Vassilevski, “Direction-Preserving and Schur-Monotonic Semiseparable Approximations of Symmetric Positive Definite Matrices”, SIMAX, 31 (5), 2650-2664, 2010.
- J. Xia, M. Gu, “Robust approximate Cholesky factorization of rank-structured symmetric positive definite matrices”, SIMAX, 31 (5), 2899-2920, 2010.
- J. Xia, “Efficient structured multifrontal factorization for general large sparse matrices”, SISC, 35 (2), A832-A860, 2012.
- A. Napov, “Conditioning analysis of incomplete Cholesky factorizations with orthogonal dropping”, to appear in SIMAX.
- Martinsson, “A Fast Randomized Algorithm for Computing A Hierarchically Semiseparable Representation of A Matrix”, SIMAX, Vol.32, No.4, 1251-1274, 2011.
- S. Wang, X.S. Li, J. Xia, and M.V. de Hoop, “Efficient scalable algorithms for solving linear systems with hierarchically semiseparable matrices”, SISC, Nov. 2012. (revised)
- S. Wang, X.S. Li, F.-H. Rouet, J. Xia, and M. de Hoop, “A Parallel Geometric Multifrontal Solver Using Hierarchically Semiseparable Structure”, ACM TOMS, June 2013. (in submission)
- P.R. Amestoy, C. Ashcraft, O. Boiteau, A. Buttari, J.-Y. L’Excellent, C. Weisbecker, “Improving Multifrontal Methods by Means of Block Low-Rank Representations”, SISC, submitted, 2012. Tech report, RT/APO/12/6, ENSEEIHT, Toulouse, France.