Sparse factorization using low rank submatrices

Cleve Ashcraft
LSTC
cleve@lstc.com

2010 MUMPS User Group Meeting
April 15-16, 2010
Toulouse, FRANCE

ftp.lstc.com:outgoing/cleve/MUMPS10_Ashcraft.pdf

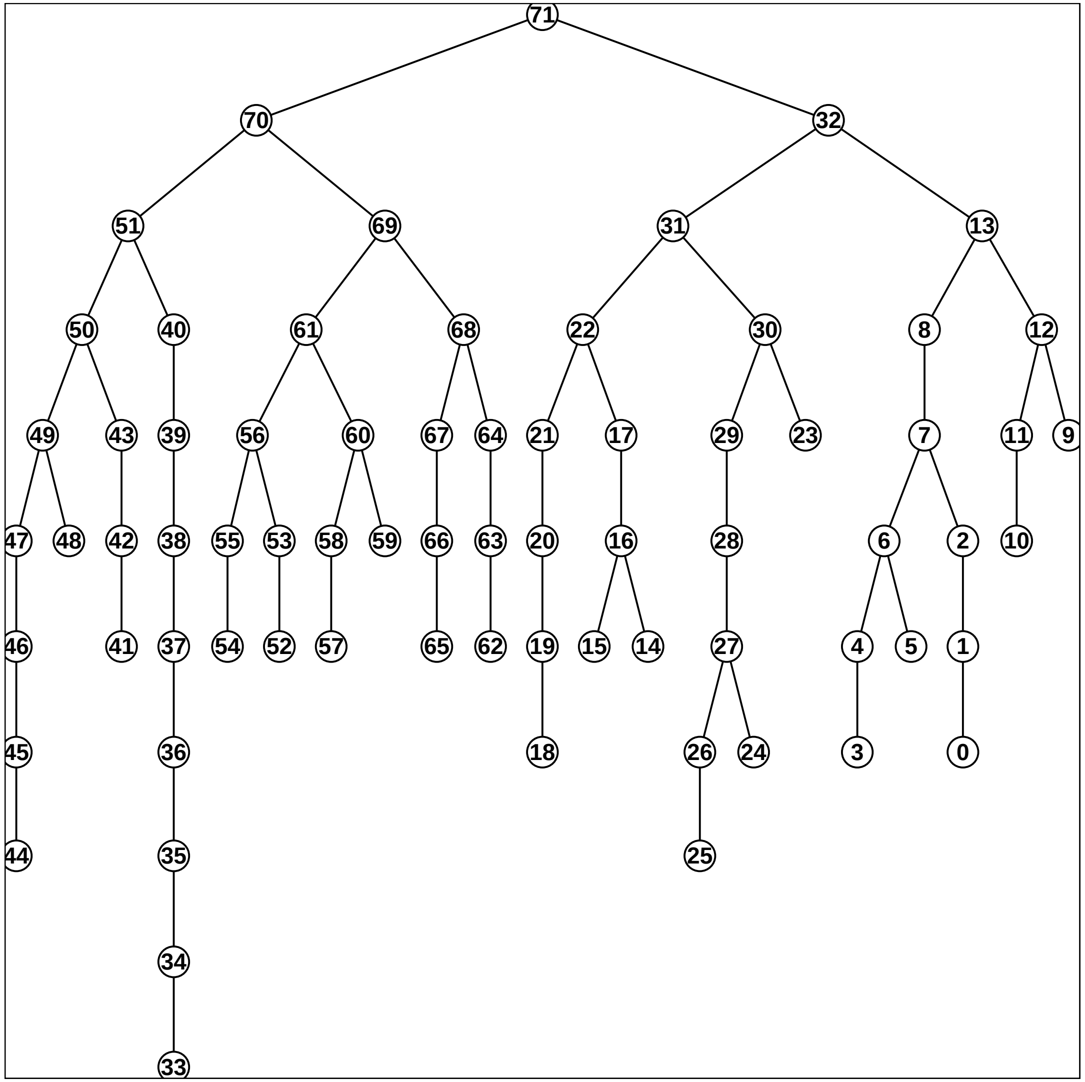
LSTC
Livermore Software Technology Corporation
Livermore, California 94550

- Founded by John Hallquist of LLNL, 1980's
- Public domain versions of DYNA and NIKE codes
- LS-DYNA : implicit/explicit, nonlinear finite element analysis code
- Multiphysics capabilities
  - Fluid/structure interaction
  - Thermal analysis
  - Acoustics
  - Electromagnetics

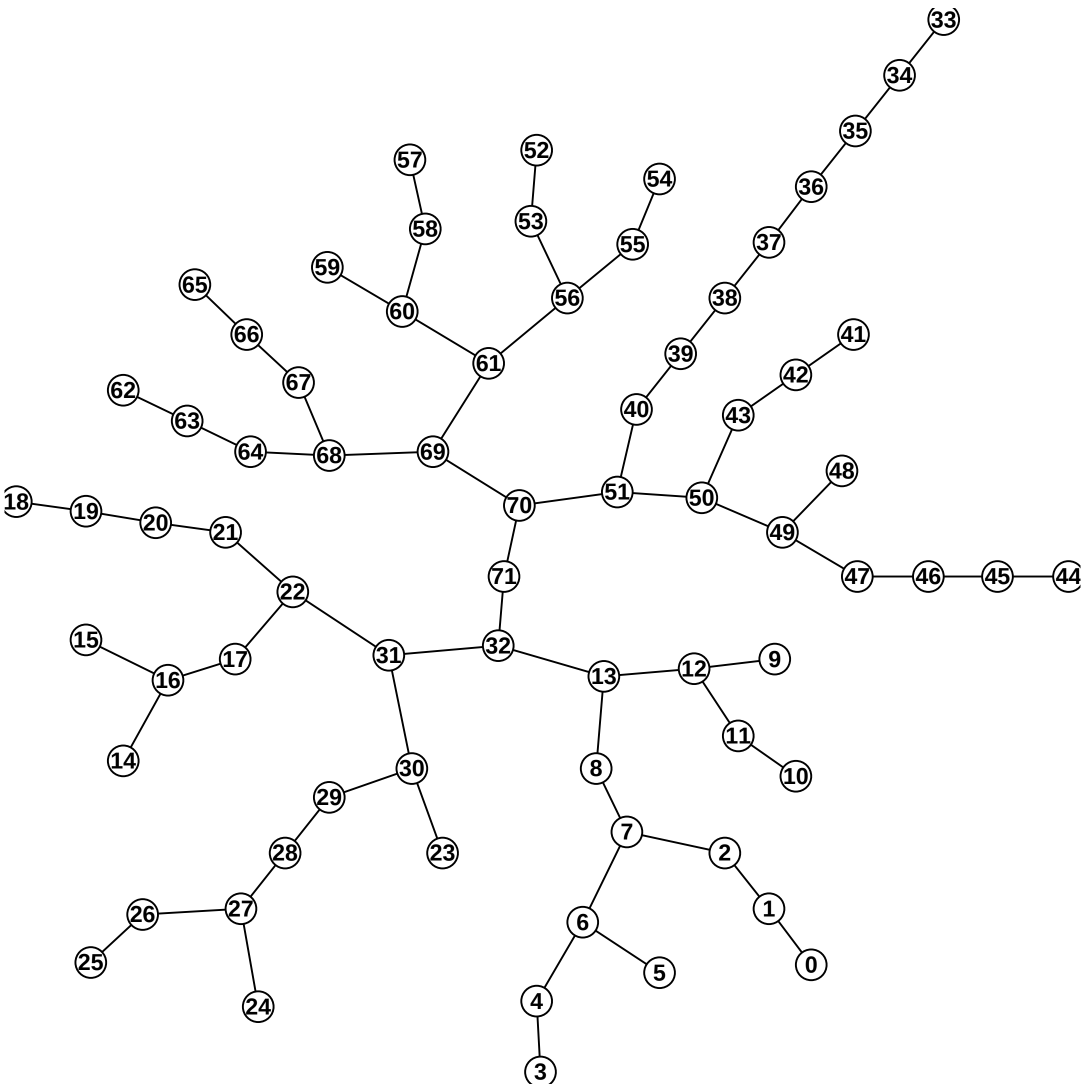
Multifrontal Algorithm

- very large, sparse LDL^T and LDU factorizations
- tree structure organizes factor storage, solve and factor operations
- medium-to-large sparse linear systems located at each leaf node of the tree
- medium-sized dense linear systems located at each interior node of the tree
- dense matrix-matrix operations at each interior node
- sparse matrix-matrix adds between nodes

Multifrontal tree



Multifrontal tree – polar representation

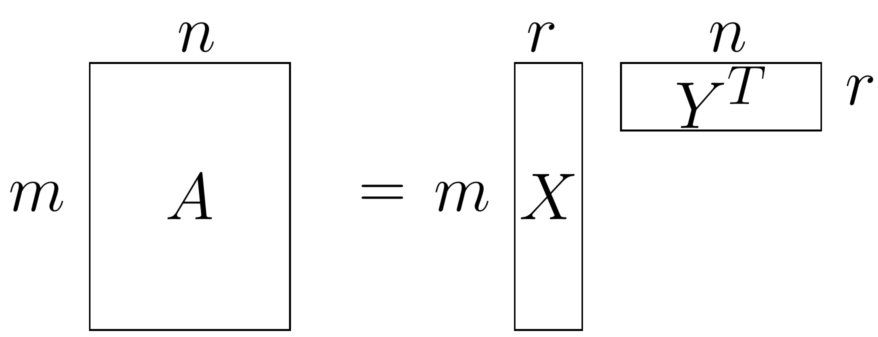


Multifrontal Algorithm

- 40M equations, present frontier at LSTC
- serial, SMP, MPP, hybrid, GPU
- large problems, $>$ 1M - 10M dof,
  require out-of-core storage of factor entries,
  even on distributed memory systems
- IO cost largely hidden during the factorization
- IO cost dominant during the solves
- eigensolver $\Longrightarrow$ several right hand sides
- many applications, e.g., Newton's method,
  have a single right hand side

Low Rank Approximations

- hierarchical matrices,
  Hackbusch, Bebendorf, Leborne, others
- semi-separable matrices, Gu, others
- submatrices are numerically rank deficient
- method of choice for Boundary Elements (BEM)
- now applied to Finite Elements (FEM)
- $A \approx XY^T$



- **storage** $= r(m+n)$ **vs** mn
- **reduction in ops** $= \dfrac{r}{\min(m,n)}$

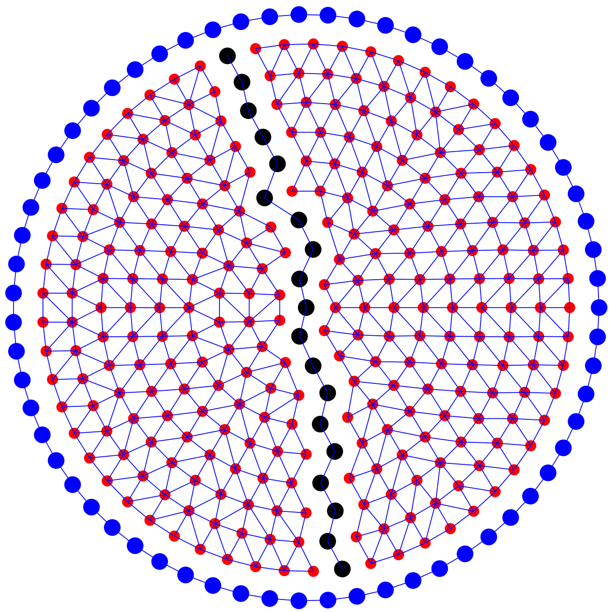
Multifrontal Algorithm + Low Rank Approximations

- **At each leaf node in the multifrontal tree — use standard multifrontal**
- **At each interior node in the multifrontal tree — low rank matrix-matrix multiplies and sums**
- **Between nodes — low rank matrix sums**
- **Dramatic reduction in storage**
- **Dramatic reduction in operations**
- **Excellent approximation properties for finite element operators**
- **Our experience is with potential equations, elasticity with solids and shells**

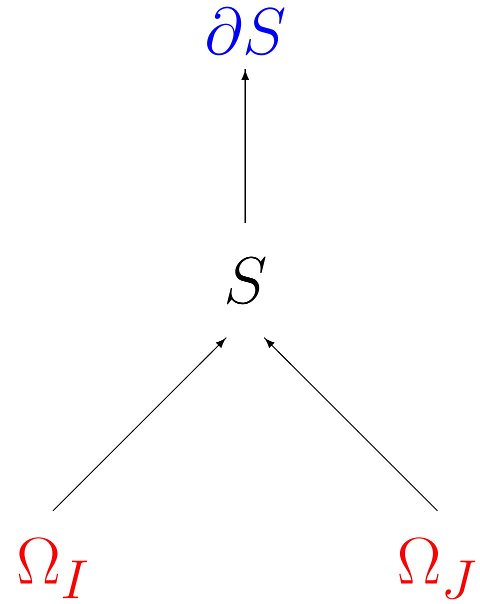
Outline

- graph, tree, matrix perspectives
- experiments – 2-D potential equation
- low rank computations
- blocking strategies
- summary

One subgraph



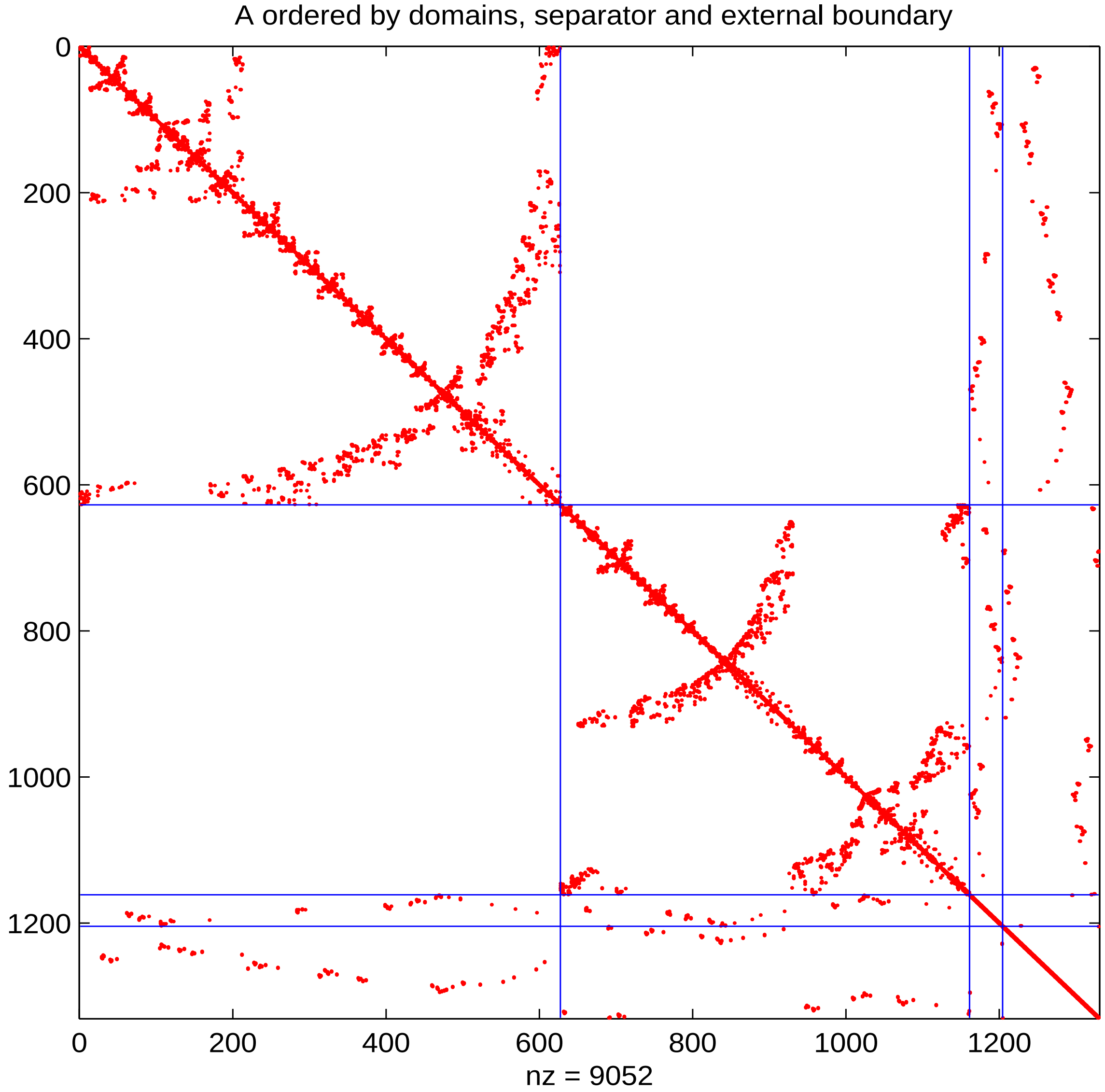
One subtree



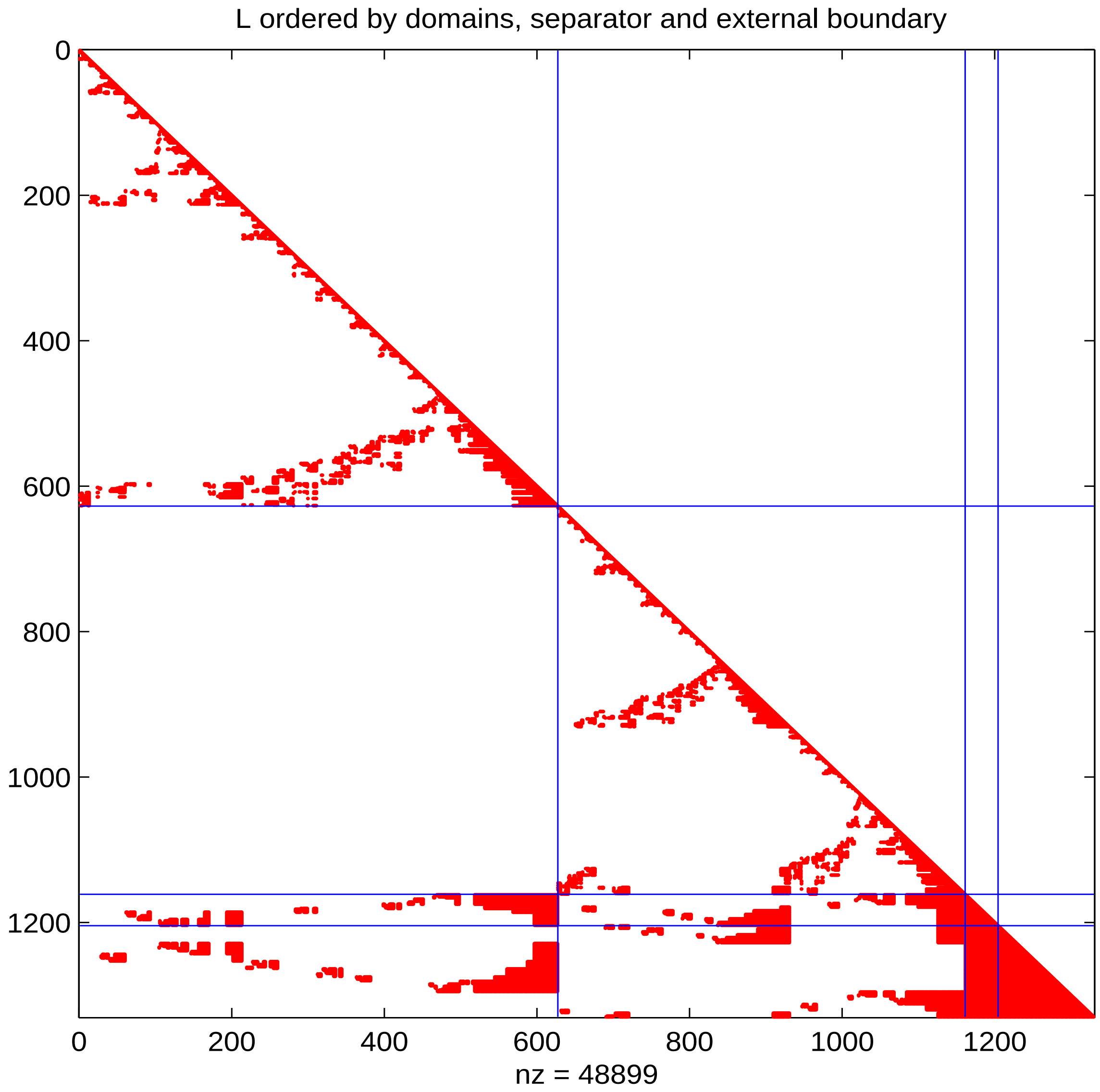
One submatrix

$$
\begin{bmatrix}
A_{\Omega_I,\Omega_I} & & A_{\Omega_I,S} & A_{\Omega_I,\partial S} \\
 & A_{\Omega_J,\Omega_J} & A_{\Omega_J,S} & A_{\Omega_J,\partial S} \\
A_{S,\Omega_I} & A_{S,\Omega_J} & A_{S,S} & A_{S,\partial S} \\
A_{\partial S,\Omega_I} & A_{\partial S,\Omega_J} & A_{\partial S,S} & A_{\partial S,\partial S}
\end{bmatrix}
$$

Portion of original matrix

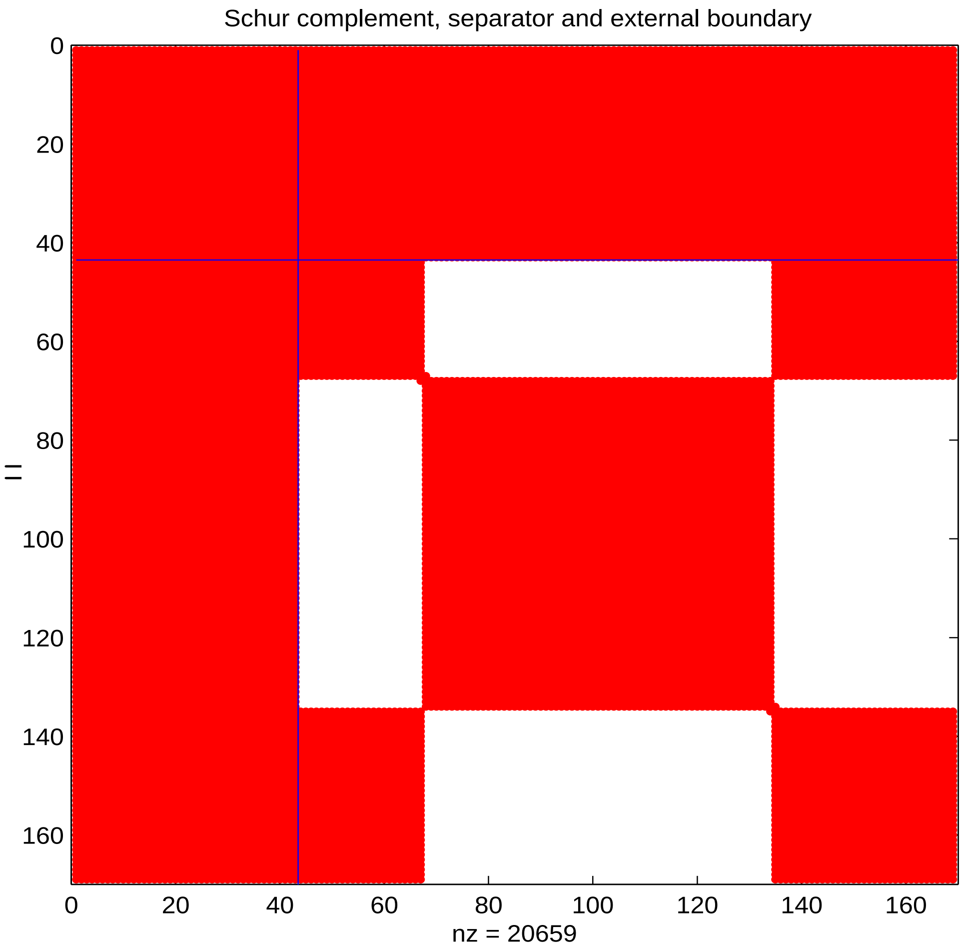


Portion of factor matrix



Schur complement matrix

$$\begin{bmatrix} \widehat{A}_{S,S} & \widehat{A}_{S,\partial S} \\ \widehat{A}_{\partial S,S} & \widehat{A}_{\partial S,\partial S} \end{bmatrix} =$$



Outline

- graph, tree, matrix perspectives
- experiments – 2-D potential equation
- low rank computations
- blocking strategies
- summary

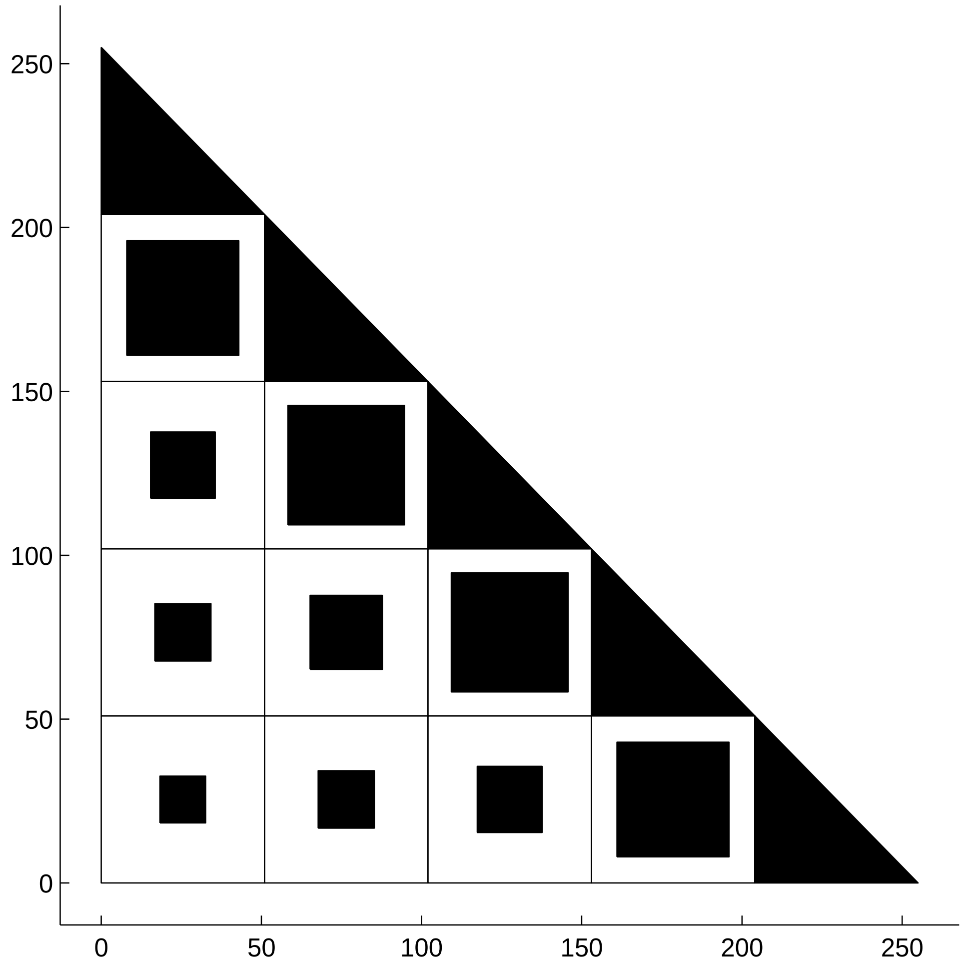
Computational experiments

- **Compute** $L_{S,S}$, $L_{\partial S,S}$ **and** $\widehat{A}_{\partial S,\partial S}$
- **Find domain decompositions of** S **and** ∂S
- **Form block matrices, e.g.,** $L_{S,S} = \sum_{K \geq J} L_{K,J}$
- **Find singular value decompositions of each** $L_{K,J}$
- **Collect all singular values**

$$\{\sigma\} = \sum_{K \geq J} \sum_{i=1}^{\min(|K|,|J|)} \sigma_i^{(K,J)}$$

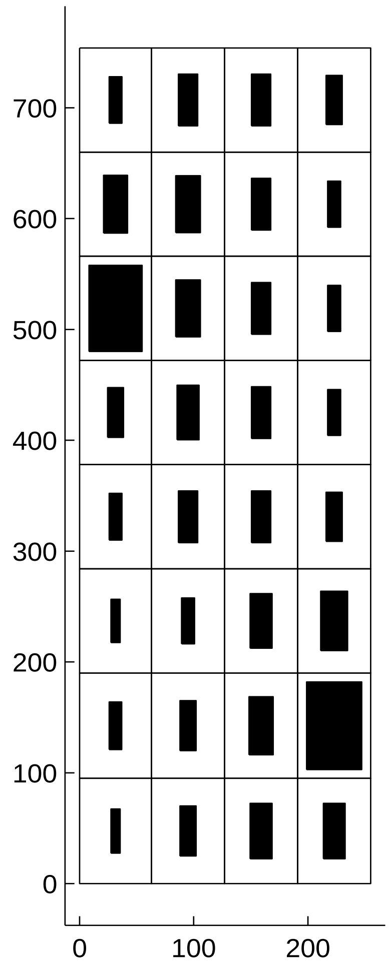
- **Split matrix** $L_{S,S} = M_{S,S} + N_{S,S}$ **using singular values** $\{\sigma\}$**.**
- **We want** $\|N_{S,S}\|_F \leq 10^{-14}\ \|L_{S,S}\|_F$

255×255 **diagonal block factor** $L_{S,S}$
43% **dense, relative accuracy** 10^{-14}

$L_{S,S} =$ 

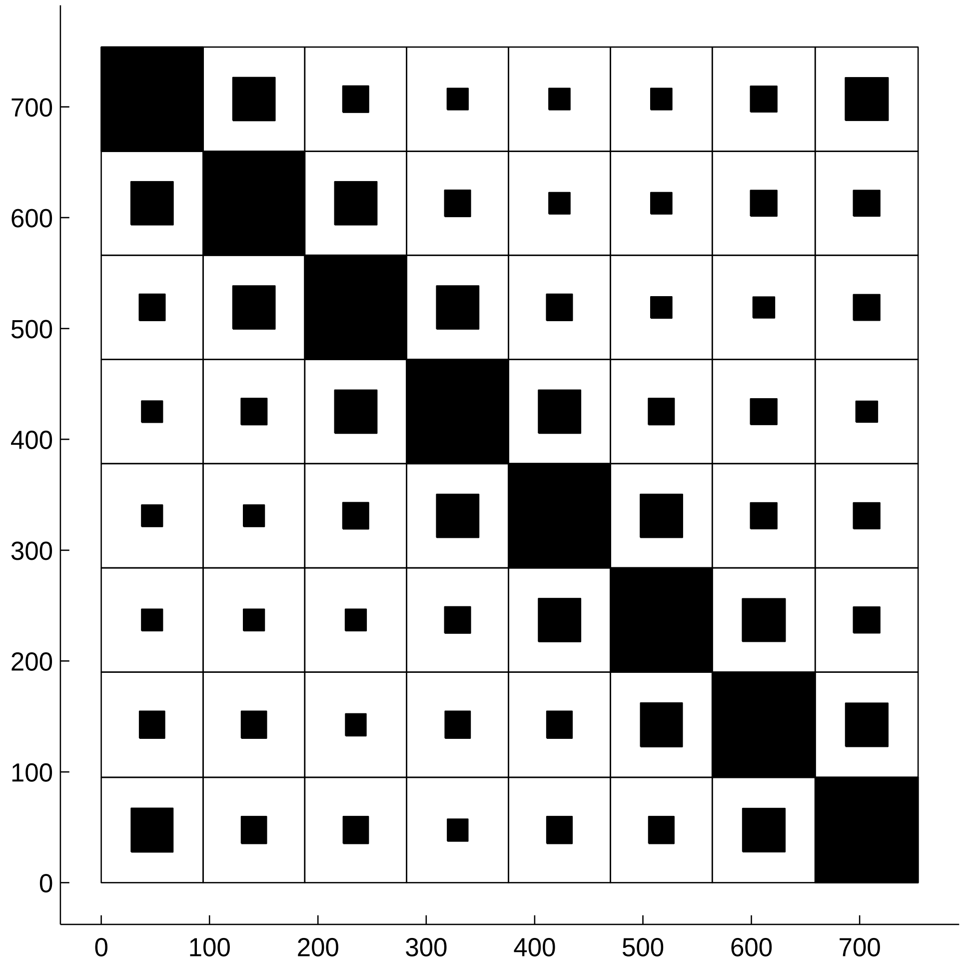
754×255 **lower block factor** $L_{\partial S,S}$
16% **dense, relative accuracy** 10^{-14}

$$L_{\partial S,S} =$$

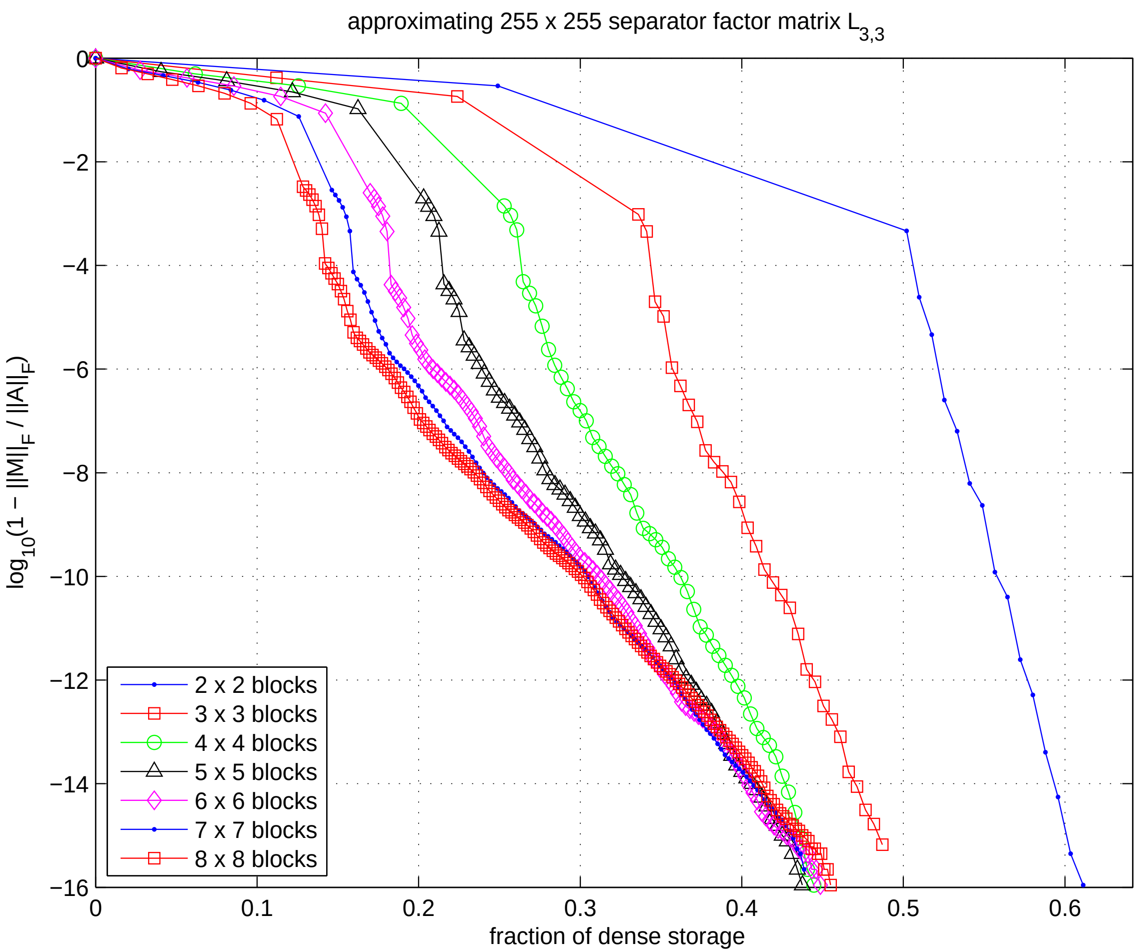


754×754 **update matrix** $\widehat{A}_{\partial S,\partial S}$
21% **dense, relative accuracy** 10^{-14}

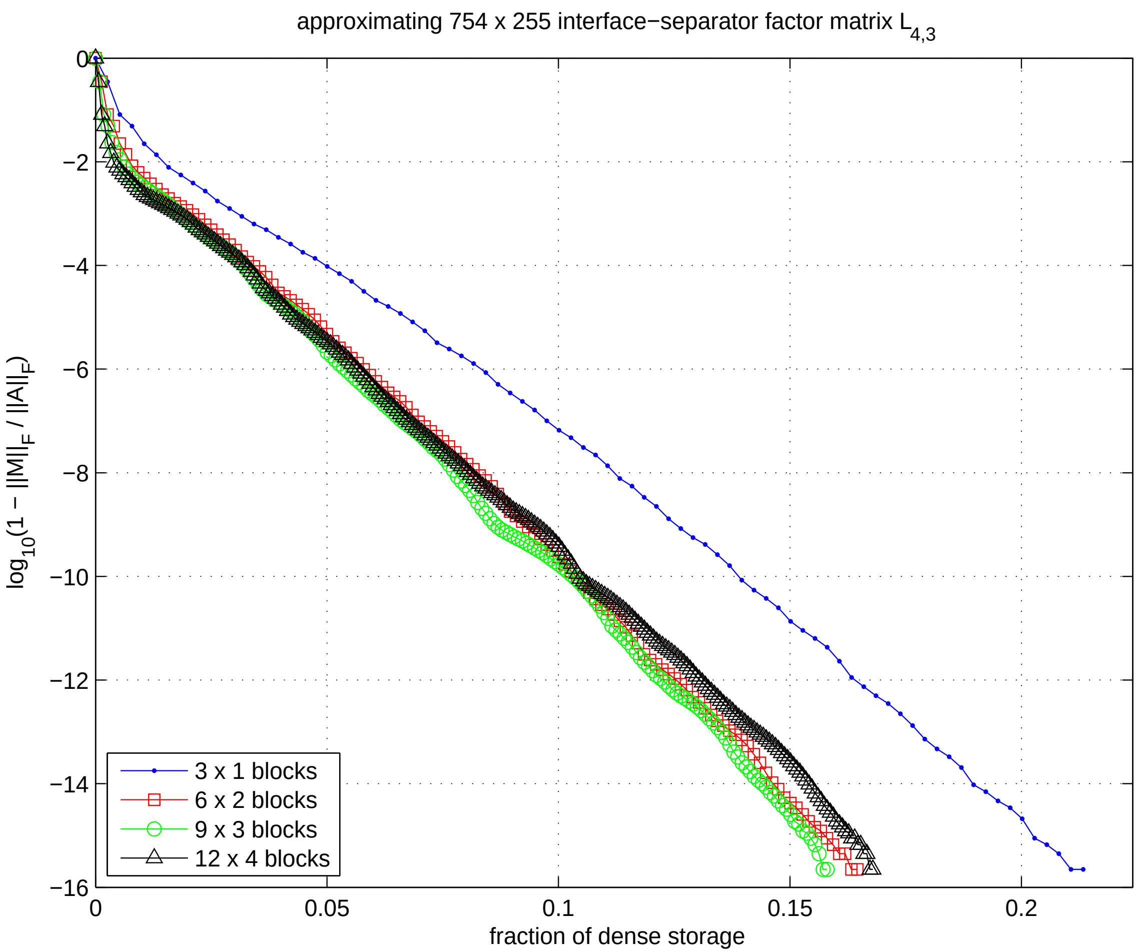
$$\widetilde{A}_{\partial S,\partial S} =$$



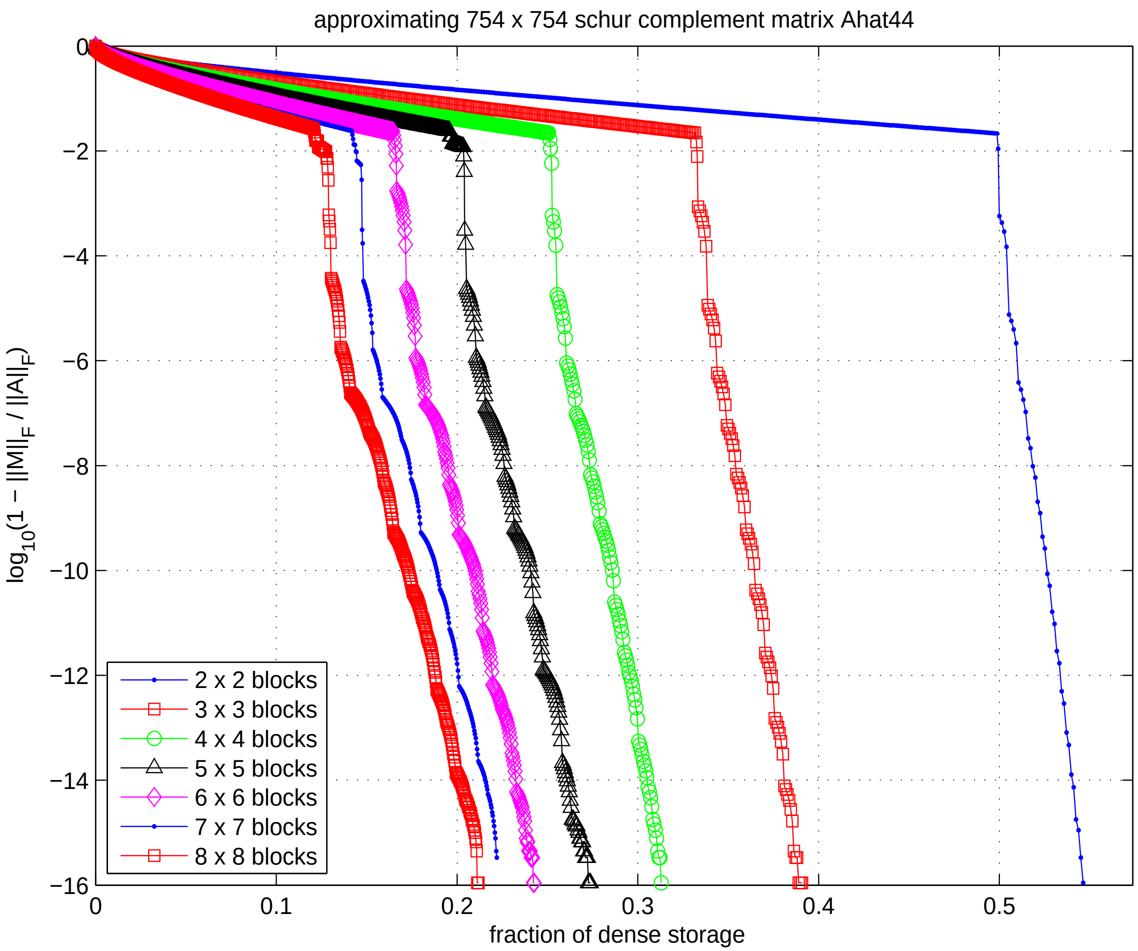
255×255 factor matrix $L_{S,S}$ storage vs accuracy



754×255 factor matrix $L_{\partial S,S}$ storage vs accuracy



754×754 update matrix $\widehat{A}_{\partial S, \partial S}$ storage vs accuracy



Outline

- graph, tree, matrix perspectives
- experiments – 2-D potential equation
- low rank computations
- blocking strategies
- summary

How to compute low rank submatrices ?

- **SVD – singular value decomposition**

$$A = U\Sigma U^T$$

where U and V are orthogonal and Σ is diagonal

- **Gold standard, expensive, $O(n^3)$ ops**

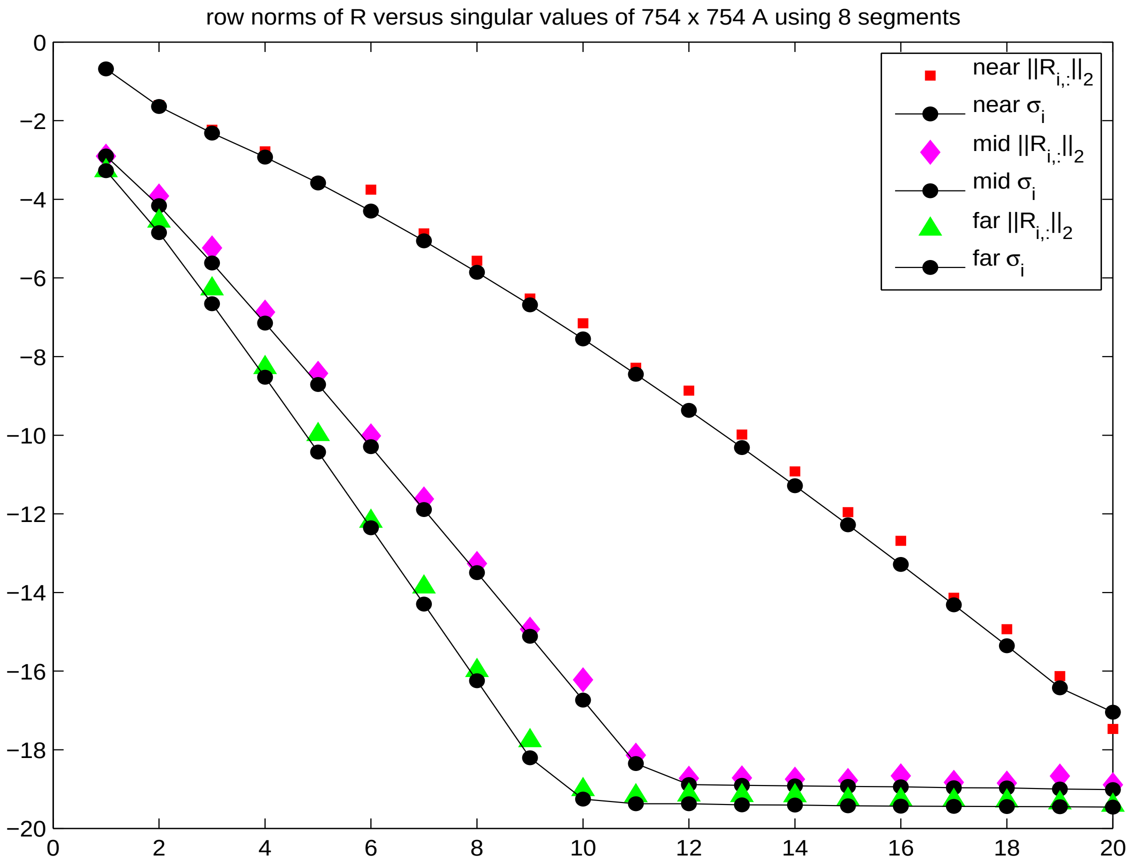
- **QR factorization**

$$AP = QR$$

where Q is orthogonal and R is upper triangular, and P is a permutation matrix

- **Silver standard, moderate cost, $O(rn^2)$ ops**

row norms of R vs singular values of A
754×754 update matrix $\widehat{A}_{\partial S, \partial S}$
94×94 submatrix size
near, mid and far interactions



SVD vs QR with column pivoting — Conclusions :

- **Column pivoting QR factorization does well.**
- **Row norms of R track singular values σ**
- **The numerical rank of R is greater than needed, but not that much greater**
- **For more accuracy/less storage two sided orthogonal factorizations**
  - $AP = ULV$, U **and** V **orthogonal,** L **triangular**
  - $PAQ = UBV$,
    U **and** V **orthogonal,** B **bidiagonal**

  track the singular values very closely.

Type of approximation of submatrices

- **submatrix** $L_{I,J}$ **of** $L_{\partial S,S}$, $\|L_{I,J}\|_F = 2.92 \times 10^{-2}$

factorization	numerical rank	total entries
none	—	**4900**
QR	**20**	**2681**
ULV	**18**	**2680**
SVD	**18**	**2538**

- **submatrix** $L_{I,J}$ **of** $L_{\partial S,S}$, $\|L_{I,J}\|_F = 2.04 \times 10^{-3}$

factorization	numerical rank	total entries
none	—	**4900**
QR	**14**	**1896**
ULV	**10**	**1447**
SVD	**10**	**1410**

Operations with low rank submatrices

$$A = U_A V_A^T, \qquad B = U_B V_B^T, \qquad C = U_C V_C^T$$

See Bebendorf, "Hierarchical Matrices", Chapter 1

- **Multiplication** $A = B\,C$, $\mathbf{rank}(A) \le \min(\mathbf{rank}(B), \mathbf{rank}(C))$

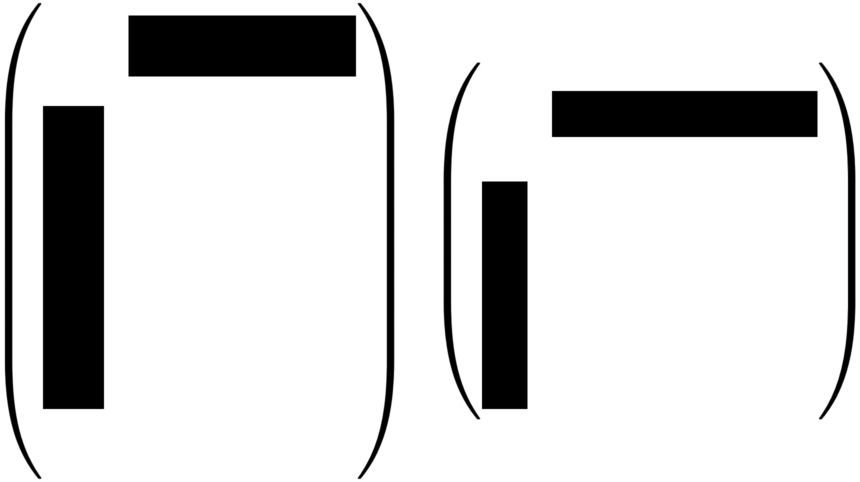
$$\begin{aligned} A = U_A V_A^T &= \left(U_B V_B^T\right)\left(U_C V_C^T\right) = BC \\ &= U_B \left(V_B^T U_C\right) V_C^T \\ &= U_B \left(\left(V_B^T U_C\right) V_C^T\right) \\ &= \left(U_B \left(V_B^T U_C\right)\right) V_C^T \end{aligned}$$

- **Addition** $A = B + C$, $\mathbf{rank}(A) \le \mathbf{rank}(B) + \mathbf{rank}(C)$

$$\begin{aligned} A = U_A V_A^T &= \left(U_B V_B^T\right) + \left(U_C V_C^T\right) = B + C \\ &= \left[U_B\ U_C\right]\left[V_B\ V_C\right]^T \end{aligned}$$

Multiplication $A = B\ C$

$$A = U_A V_A^T, \qquad B = U_B V_B^T, \qquad C = U_C V_C^T$$

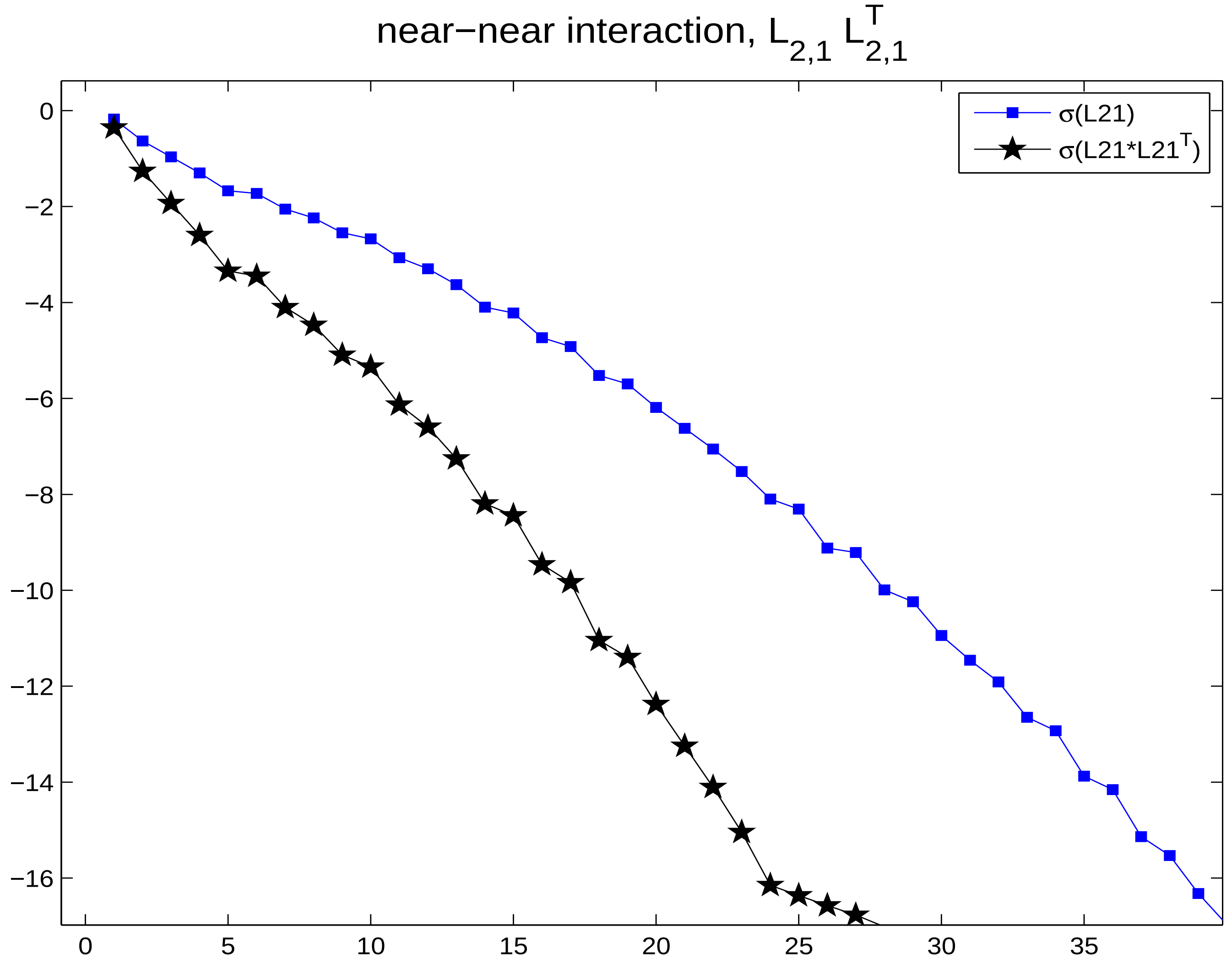
$A = B\ C =$ 

$$= \Big((m \times h)\,(h \times l) \Big) \Big((l \times k)\,(k \times n) \Big)$$

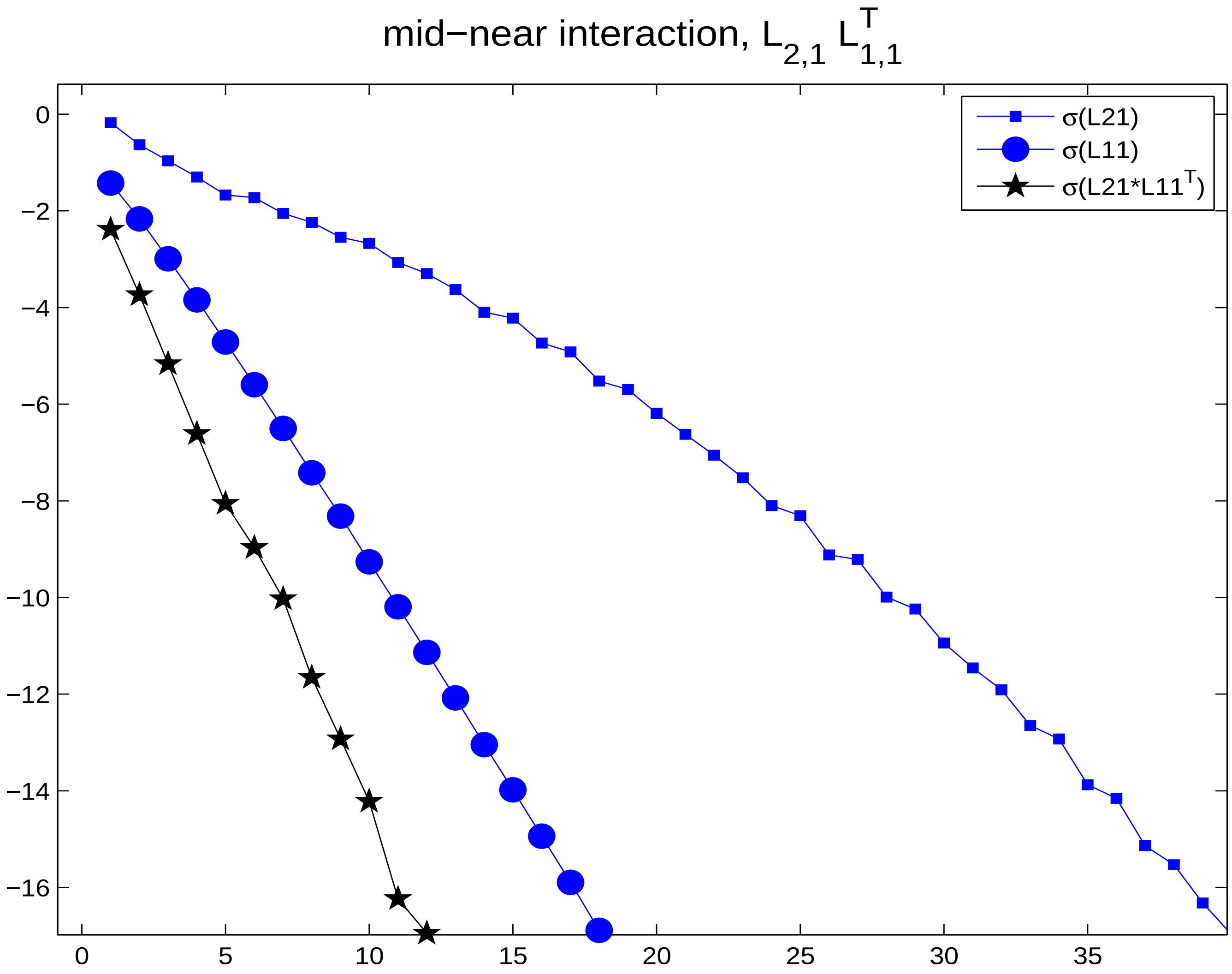
$=$  $=$ 

$$= (m \times \min(h,k))\,(\min(h,k) \times n)$$

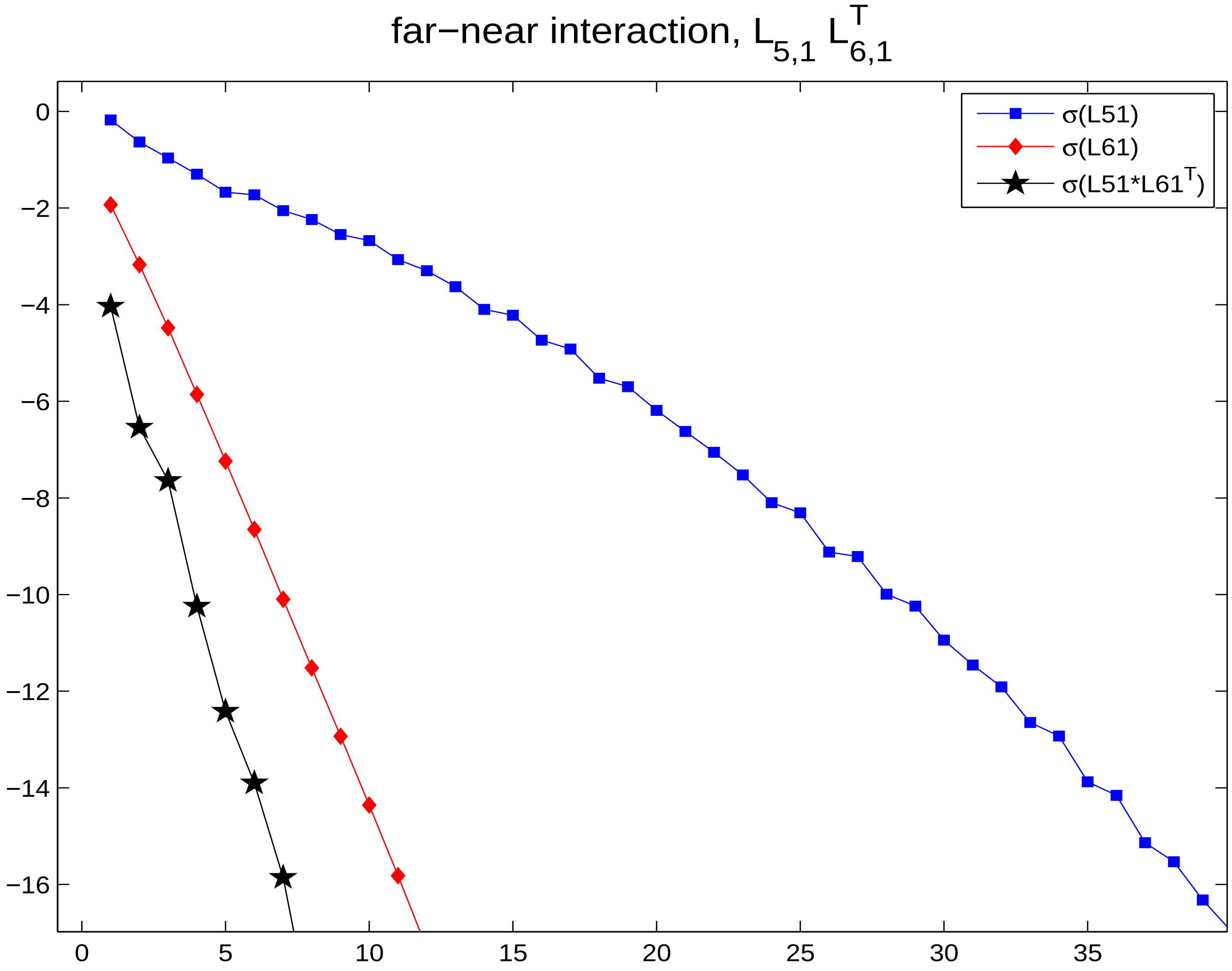
Near-near matrix product



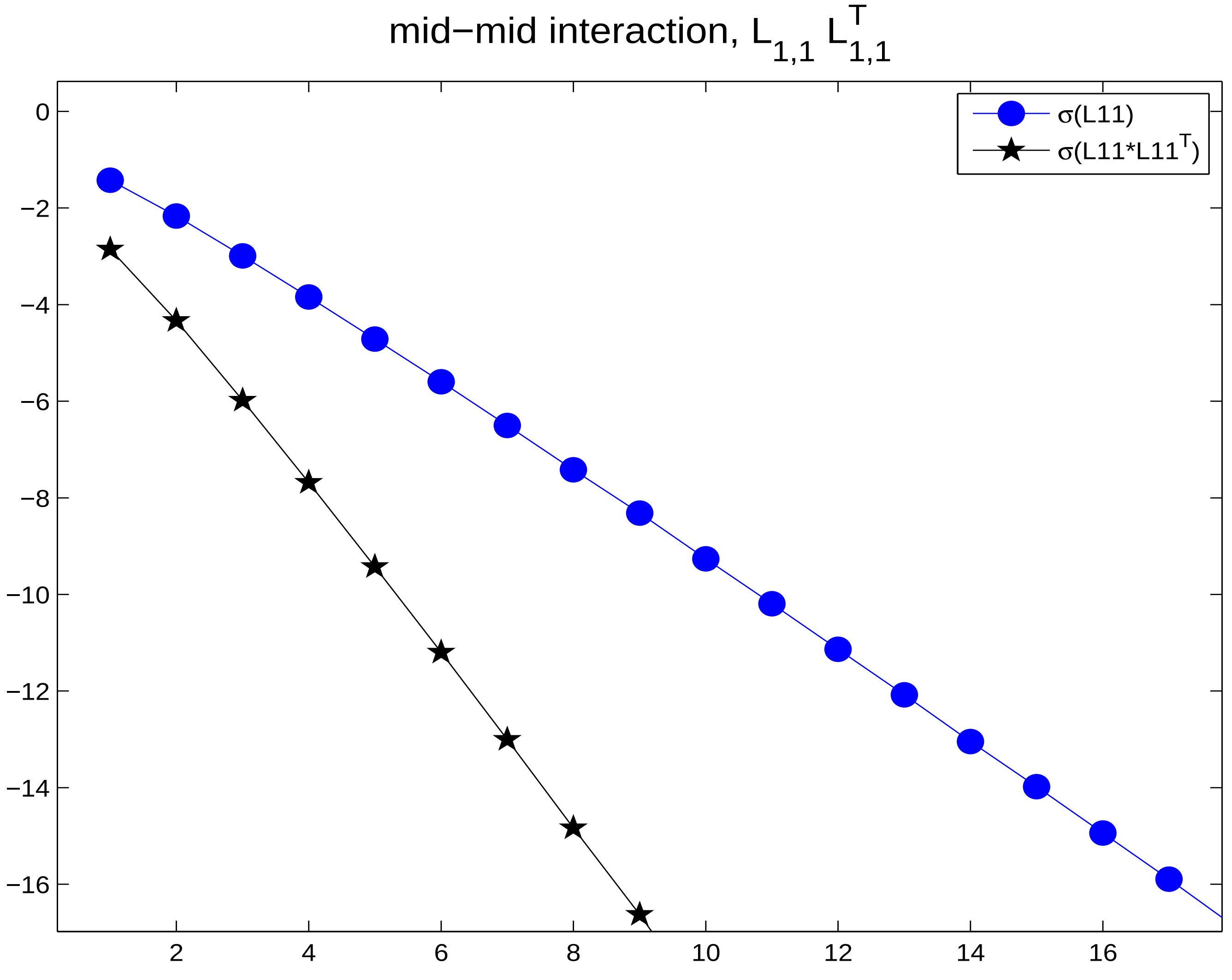
mid-near matrix product



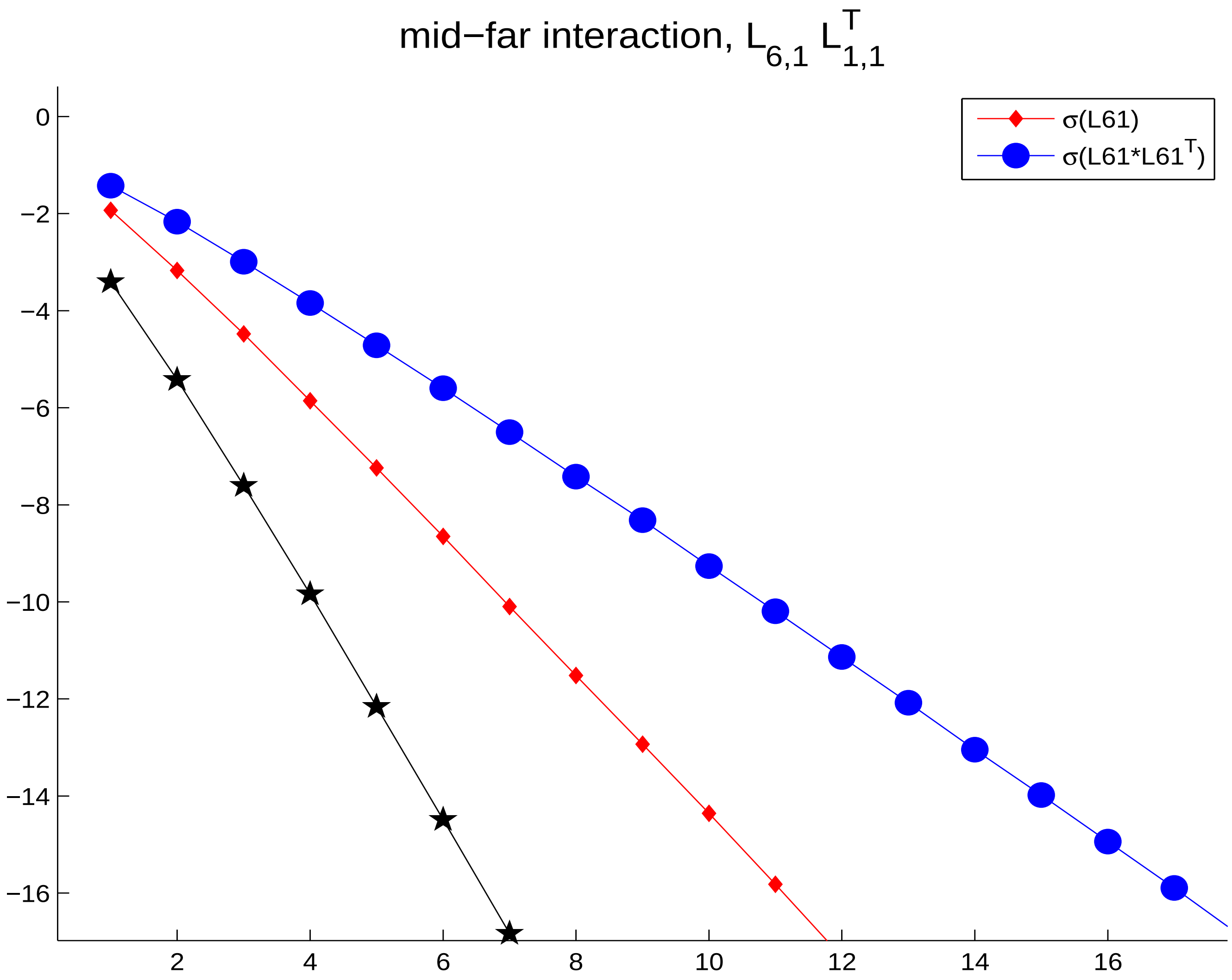
far-near matrix product



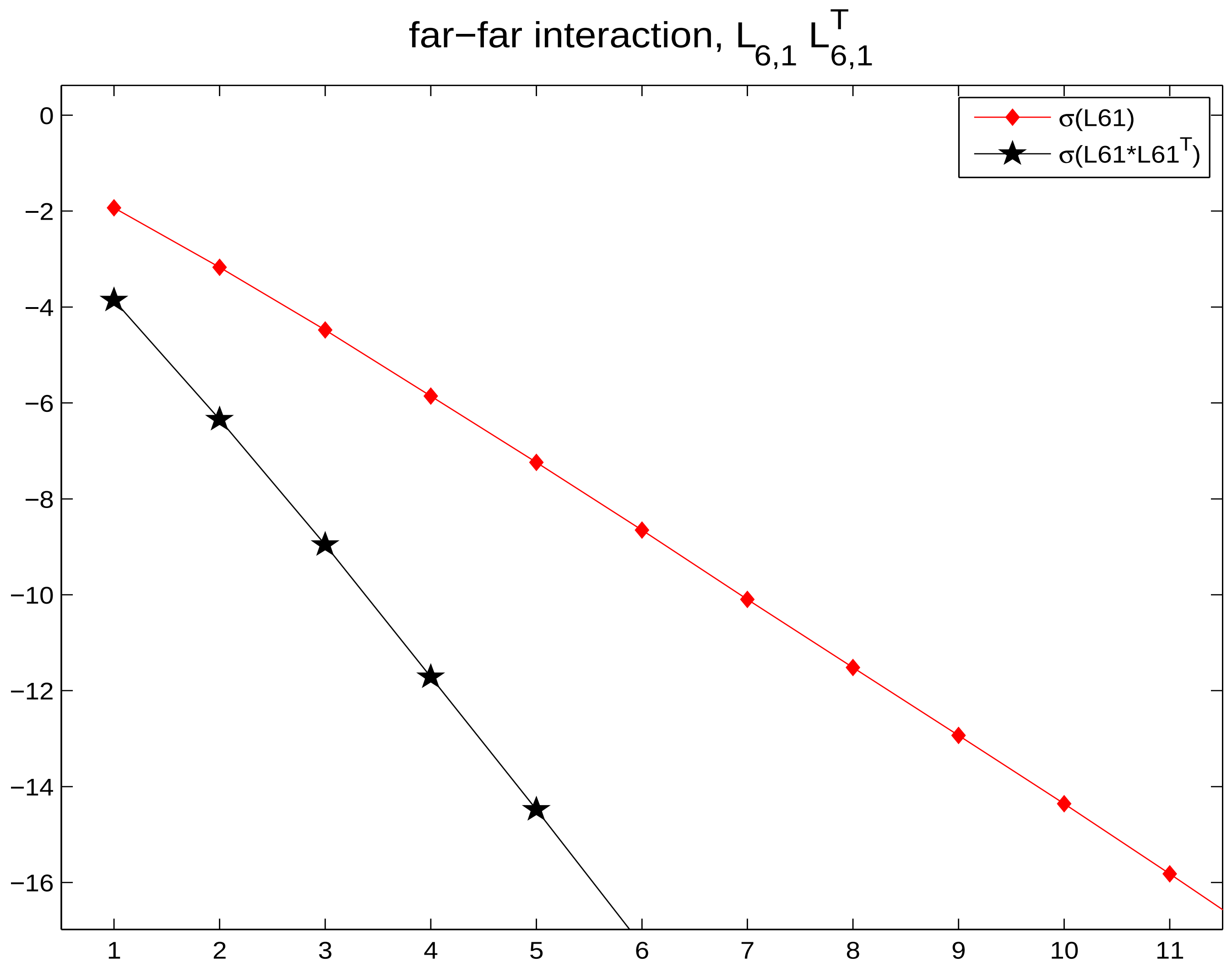
mid-mid matrix product



far-mid matrix product



far-far matrix product



Addition $A = B + C$

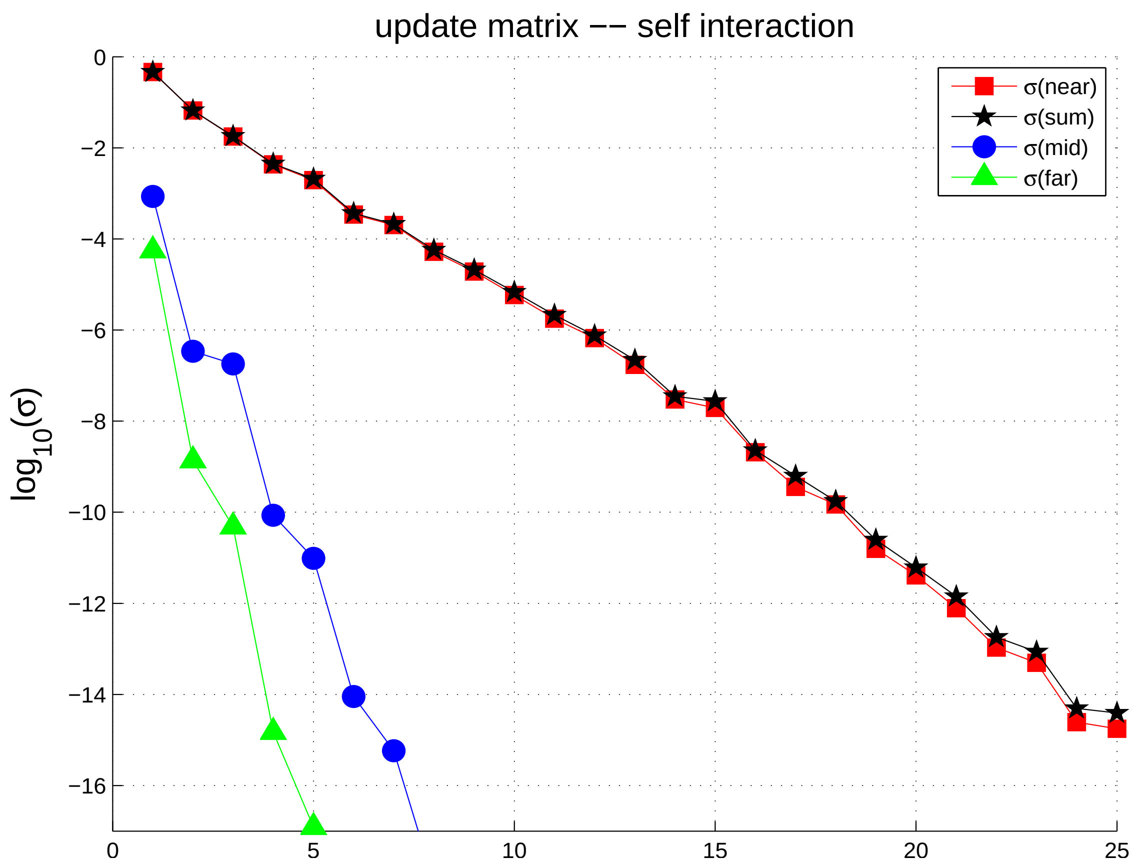
$\mathbf{rank}(A) \le \mathbf{rank}(B) + \mathbf{rank}(C)$

$$
\begin{aligned}
A = U_A V_A^T \; &= \; \left(U_B V_B^T\right) + \left(U_C V_C^T\right) = B + C \\
&= \; \left[U_B \; U_C\right] \left[V_B \; V_C\right]^T \\
&= \; (Q_1 R_1)\,(Q_2 R_2)^T \\
&= \; Q_1 \left(R_1 R_2^T\right) Q_2^T \\
&= \; Q_1\,(Q_3 R_3)\,Q_2^T \\
&= \; (Q_1 Q_3)\left(R_3 Q_2^T\right) \\
&= \; (Q_1 Q_3)\left(Q_2 R_3^T\right)^T
\end{aligned}
$$

- $R_1 R_2^T$ **usually has low numerical rank**
- **examples follow**

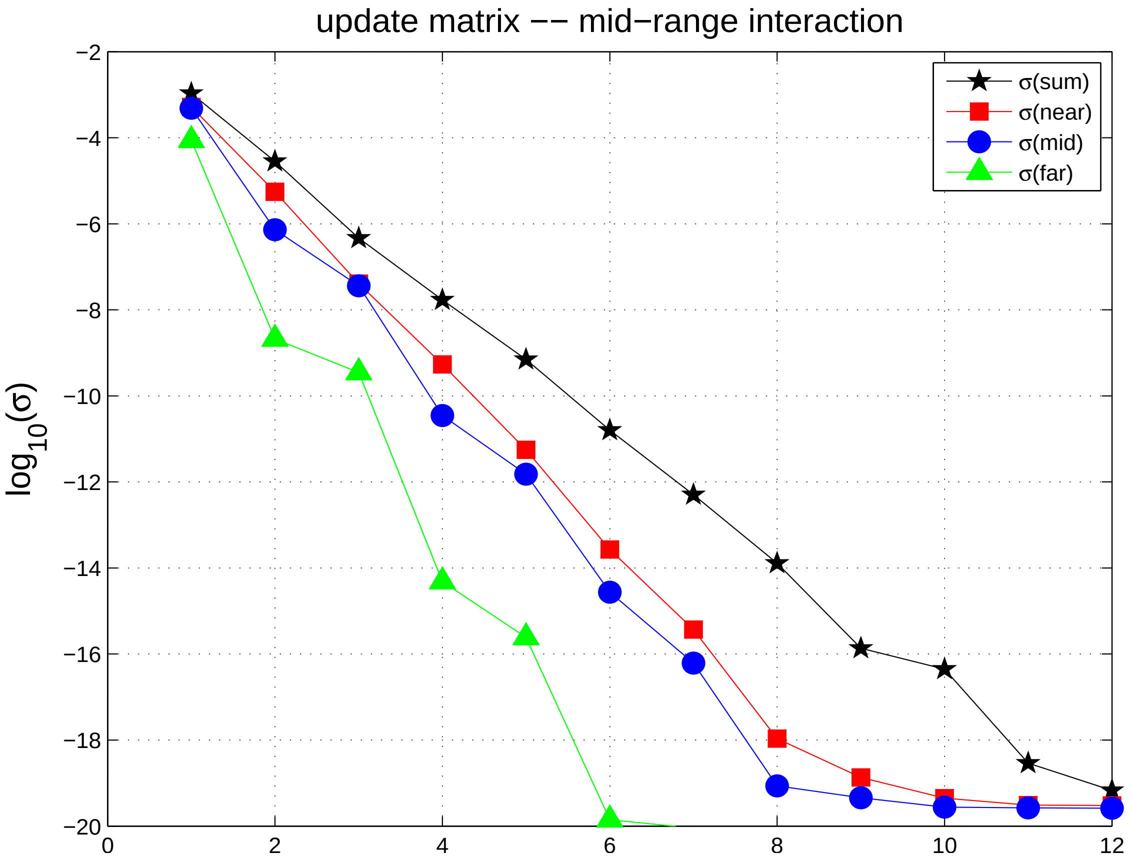
update matrix, diagonal block

$$\widehat{A}_{3,3} = L_{3,1}L_{3,1}^T + L_{3,2}L_{3,2}^T + L_{3,3}L_{3,3}^T$$



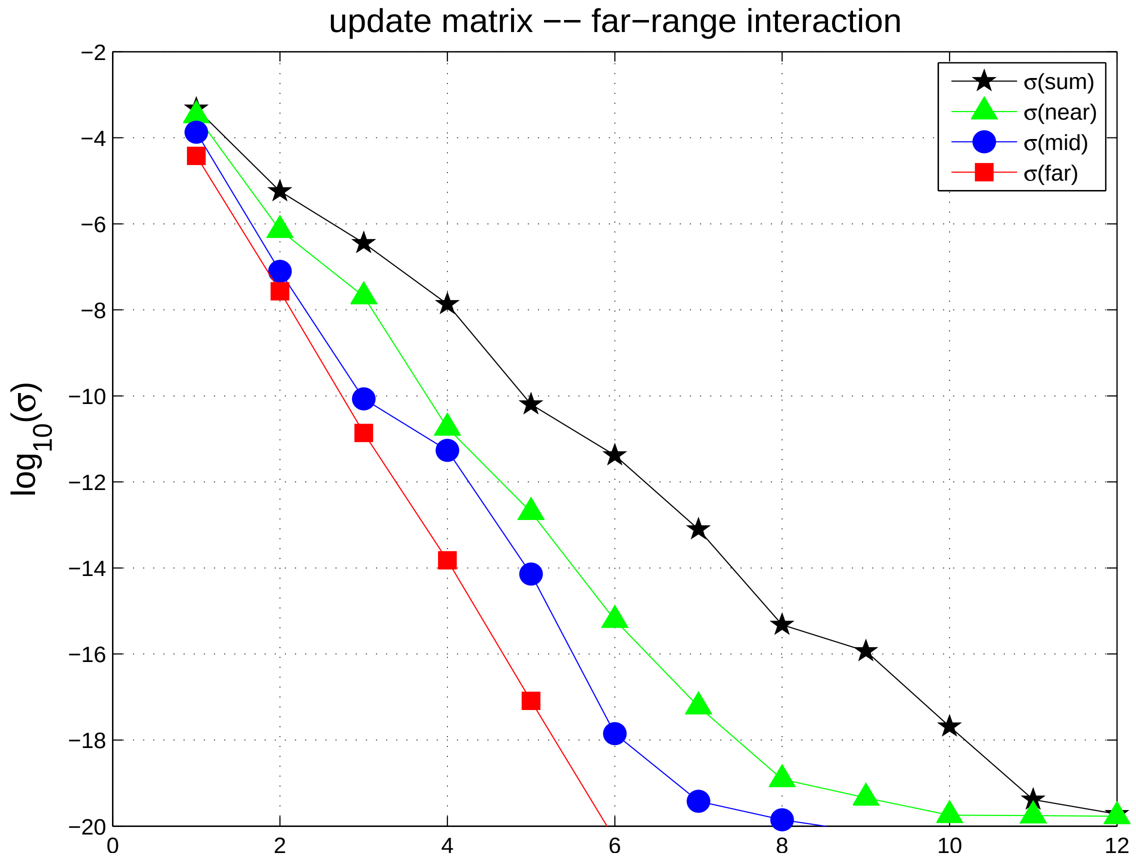
update matrix, mid-distance off-diagonal block

$$\widehat{A}_{5,3} = L_{5,1}L_{3,1}^T + L_{5,2}L_{3,2}^T + L_{5,3}L_{3,3}^T$$



update matrix, far-distance off-diagonal block

$$\widehat{A}_{7,3} = L_{7,1}L_{3,1}^T + L_{7,2}L_{3,2}^T + L_{7,3}L_{3,3}^T$$



Outline

- **graph, tree, matrix perspectives**
- **experiments – 2-D potential equation**
- **low rank computations**
- **blocking strategies**
- **summary**

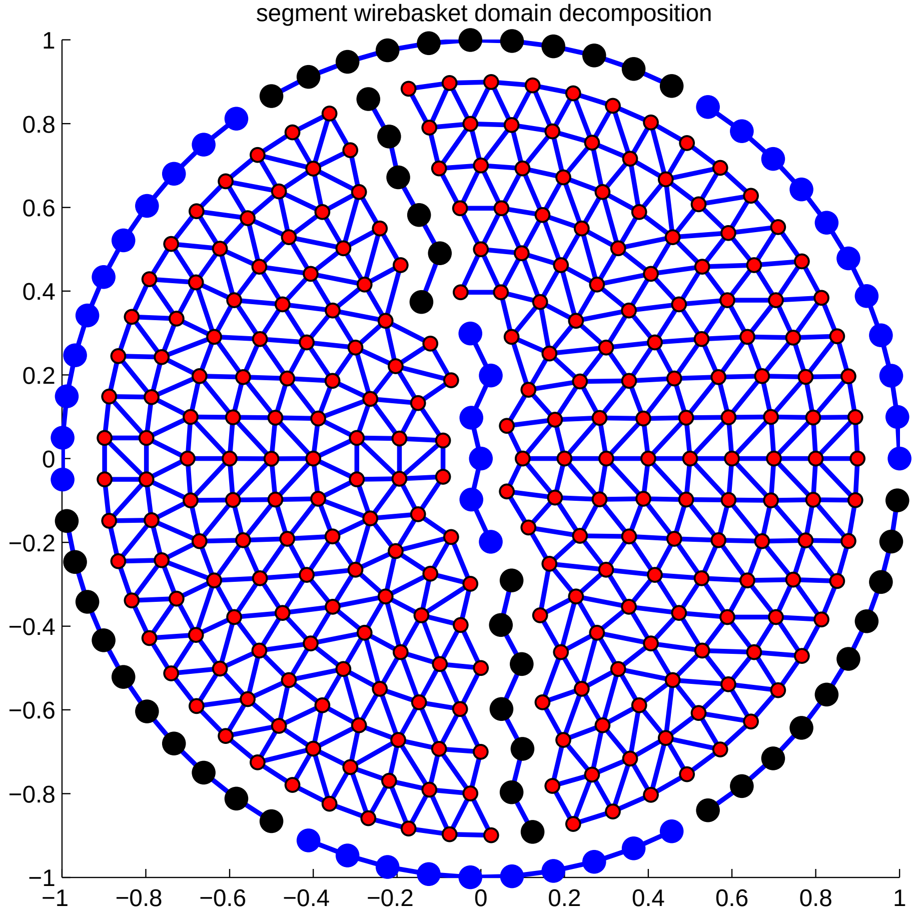
Blocking Strategies

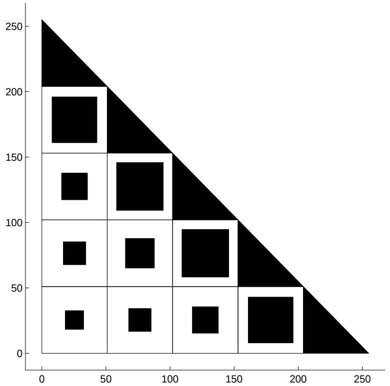
- **Active data structures**

$$\begin{bmatrix} L_{J,J} & \\ L_{\partial J,J} & \widehat{A}_{\partial J,\partial J} \end{bmatrix}$$

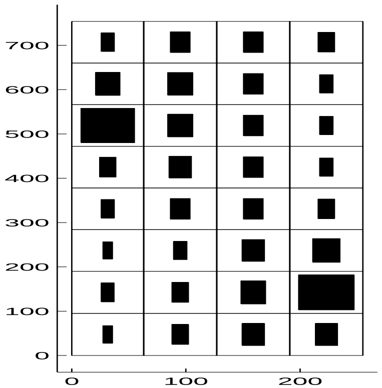
- **partition J and ∂J independently**
- **for 2-d problems, J and ∂J are 1-d manifolds**
- **for 3-d problems, J and ∂J are 2-d manifolds**
- **we need mesh partitioning of the separator and the boundary of a region J**
- **each index set of a partition is a <u>segment</u>**

Segment partition

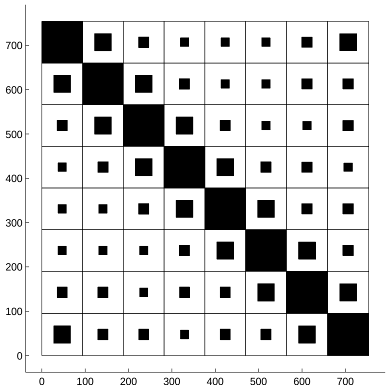




- $$L_{J,J} = \sum_{\sigma,\tau} L_{\sigma,\tau} \qquad \sigma \times \tau \subseteq J \times J$$



- $$L_{\partial J,J} = \sum_{\sigma,\tau} L_{\sigma,\tau} \qquad \sigma \times \tau \subseteq \partial J \times J$$



- $$\widehat{A}_{\partial J,\partial J} = \sum_{\sigma,\tau} \widehat{A}_{\sigma,\tau} \qquad \sigma\times\tau \subseteq \partial J\times\partial J$$

Strategy I

- **The partition of ∂J (local to node J) <u>conforms</u> to the partitions of ancestors K, $K \cap \partial J \neq \emptyset$**
- **Advantages :**
  - **Update assembly is simplified**

$$\widehat{A}^{(p(J))}_{\sigma_2,\tau_2} = \widehat{A}^{(p(J))}_{\sigma_2,\tau_2} + \widehat{A}^{(J)}_{\sigma_1,\tau_1}$$

    where $\sigma_1 \subseteq \sigma_2$, $\tau_1 \subseteq \tau_2$
  - **One destination for each $\widehat{A}^{(J)}_{\sigma_1,\tau_1}$**
- **Disadvantages :**
  - **Partition of ∂J can be fragmented, more segments, smaller size, less efficient storage**

Strategy II

- **The partition of ∂J (local to node J) <u>need not conform</u> to the partitions of ancestors**
- **Advantages :**
  - **Partition of ∂J can be optimized better since ∂J is small and localized**
- **Disadvantages :**
  - **Update assembly is more complex**

$$\widehat{A}^{(p(J))}_{\sigma_1\cap\sigma_2,\tau_1\cap\tau_2} = \widehat{A}^{(p(J))}_{\sigma_1\cap\sigma_2,\tau_1\cap\tau_2} + \widehat{A}^{(J)}_{\sigma_1\cap\sigma_2,\tau_1\cap\tau_2}$$

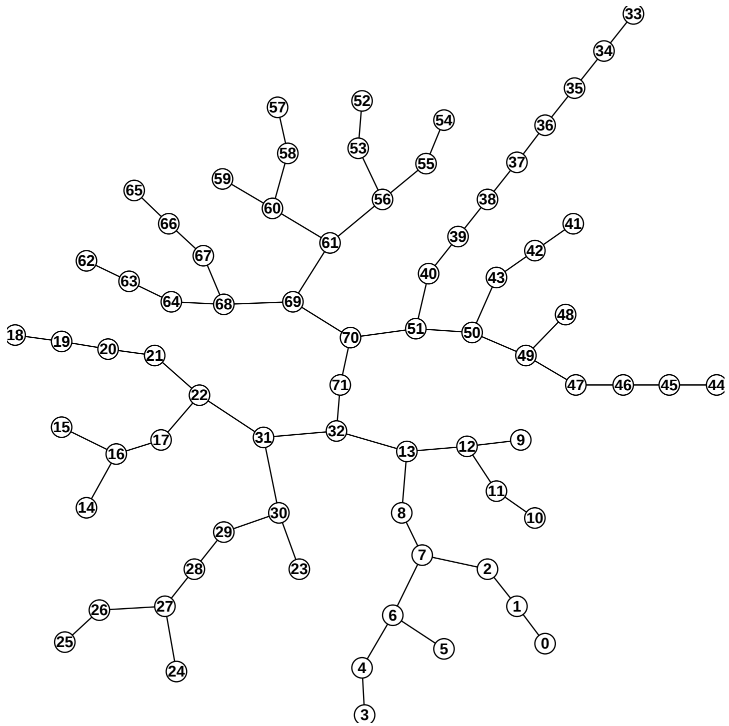
  where $\sigma_1 \cap \sigma_2 \neq \emptyset$, $\tau_1 \cap \tau_2 \neq \emptyset$

  - **Several destinations for a submatrix $\widehat{A}^{(J)}_{\sigma_1,\tau_1}$**

Outline

- **graph, tree, matrix perspectives**
- **experiments – 2-D potential equation**
- **low rank computations**
- **blocking strategies**
- **summary**

Multifrontal tree



Multifrontal Factorization

- each leaf node is a d-dimensional sparse FEM matrix
- each interior node is a $(d-1)$-dimensional dense BEM matrix
- use low rank storage and computation at each interior node

Call to action

- progress to date in low rank factorizations driven by iterative methods
- work needed from direct methods community
- start from industrial strength multifrontal code
  - serial, SMP, MPP, hybrid, GPU
  - pivoting for stability, out-of-core, singular systems, null spaces

Call to action

- **Many challenges**
  - **partition of space, partition of interface**
  - **added programming complexity of low rank matrices**
  - **challenges to pivot for stability**
  - **challenges/opportunities to implement in parallel**
- **Payoff will be huge**
  - **reduction in storage footprint**
  - **reduction in computational work**
  - **take direct methods to next level of problem size**