Hierarchical matrices

Steffen Börm

Christian-Albrechts-Universität, Kiel

BeTSSi II Benchmark Workshop, Kiel
2nd of September, 2014

Overview

1. What are hierarchical matrices?
2. How can we construct hierarchical matrices?
3. How can we solve linear systems efficiently?
4. What are current research priorities?

Overview

Model problem

Goal: Solve a integral equation

$$\lambda u(x) + \int_\Omega g(x,y)u(y)\,dy = f(x)$$

with non-local kernel functions g.

Discretization: Translate into a linear system $Gz = b$.

Challenges:

- The matrix G is usually dense (almost all entries are non-zero).
  $\rightarrow$ Very high storage requirements.
- The matrix G is usually ill-conditioned.
  $\rightarrow$ Standard Krylov methods take very long.

Low-rank matrices

Observation: Local degenerate approximations of g exist, i.e.,

$$g(x,y) \approx \tilde{g}_{\tau\sigma}(x,y) = \sum_{\nu=1}^{k} a_\nu(x) b_\nu(y) \qquad \text{for all } x \in \tau,\ y \in \sigma.$$

Examples: Taylor expansion, multipole expansion, interpolation, ...

Discretization of the approximation yields

$$g_{ij} \approx \int_\Omega \varphi_i(x) \int_\Omega \tilde{g}_{\tau\sigma}(x,y) \psi_j(y)\, dy\, dx$$

Low-rank matrices

Observation: Local degenerate approximations of g exist, i.e.,

$$g(x,y) \approx \tilde{g}_{\tau\sigma}(x,y) = \sum_{\nu=1}^{k} a_\nu(x) b_\nu(y) \qquad \text{for all } x \in \tau,\ y \in \sigma.$$

Examples: Taylor expansion, multipole expansion, interpolation, ...

Discretization of the approximation yields

$$g_{ij} \approx \sum_{\nu=1}^{k} \underbrace{\int_\Omega \varphi_i(x) a_\nu(x)\,dx}_{=:a_{i\nu}} \underbrace{\int_\Omega \psi_j(y) b_\nu(y)\,dy}_{=:b_{j\nu}}$$

Low-rank matrices

Observation: Local degenerate approximations of g exist, i.e.,

$$g(x,y) \approx \tilde{g}_{\tau\sigma}(x,y) = \sum_{\nu=1}^{k} a_\nu(x) b_\nu(y) \qquad \text{for all } x \in \tau,\ y \in \sigma.$$

Examples: Taylor expansion, multipole expansion, interpolation, ...

Discretization of the approximation yields

$$g_{ij} \approx \sum_{\nu=1}^{k} a_{i\nu} b_{j\nu}$$

Low-rank matrices

Observation: Local degenerate approximations of g exist, i.e.,

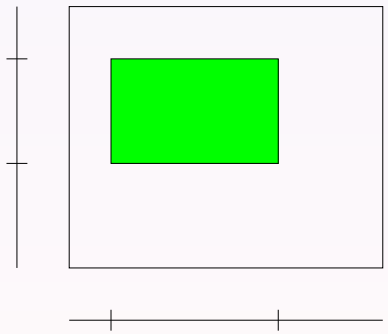
$$g(x,y) \approx \tilde{g}_{\tau\sigma}(x,y) = \sum_{\nu=1}^{k} a_\nu(x) b_\nu(y) \qquad \text{for all } x \in \tau,\ y \in \sigma.$$

Examples: Taylor expansion, multipole expansion, interpolation, ...

Discretization of the approximation yields low-rank approximation of matrix blocks.

$$G|_{t\times s} \approx A_{ts} B_{ts}^*$$

Factorized representation takes only $\mathcal{O}(k(\#t + \#s))$ units of storage.



Low-rank matrices

Observation: Local degenerate approximations of g exist, i.e.,

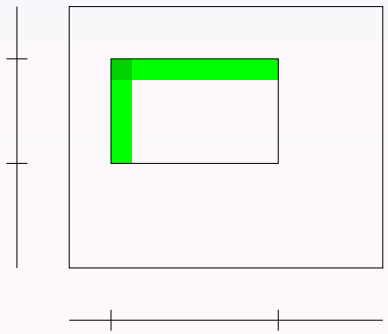
$$g(x,y) \approx \tilde{g}_{\tau\sigma}(x,y) = \sum_{\nu=1}^{k} a_\nu(x) b_\nu(y) \qquad \text{for all } x \in \tau,\ y \in \sigma.$$

Examples: Taylor expansion, multipole expansion, interpolation, ...

Discretization of the approximation yields low-rank approximation of matrix blocks.

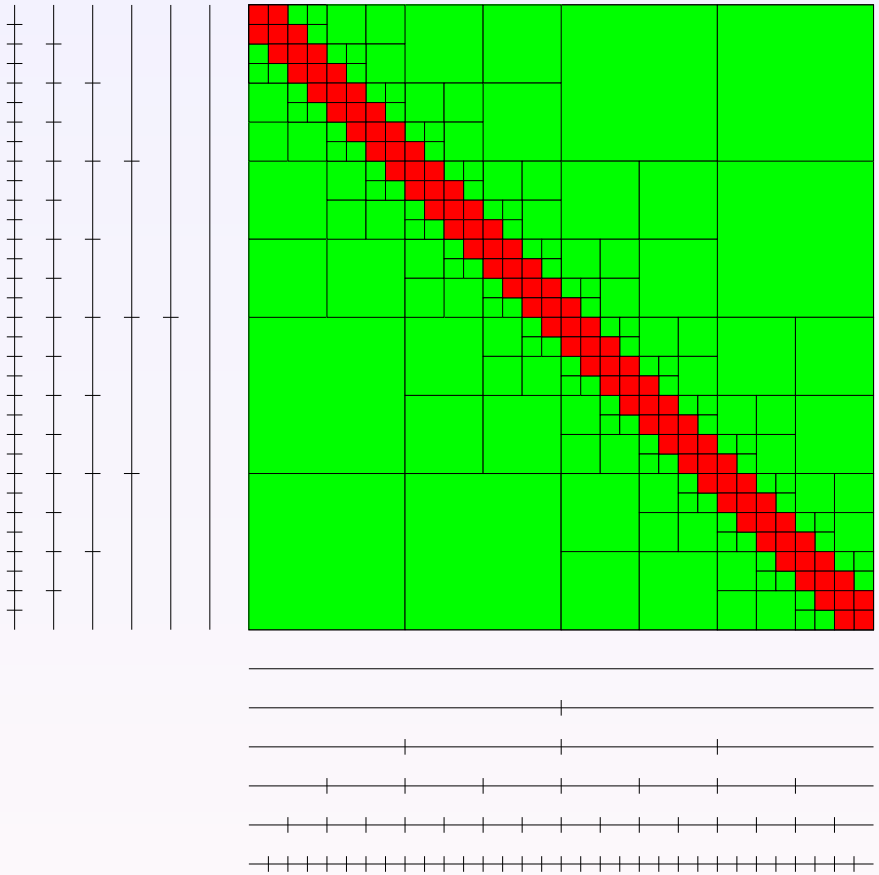
$$G|_{t\times s} \approx A_{ts} B_{ts}^*$$

Factorized representation takes only $\mathcal{O}(k(\#t + \#s))$ units of storage.



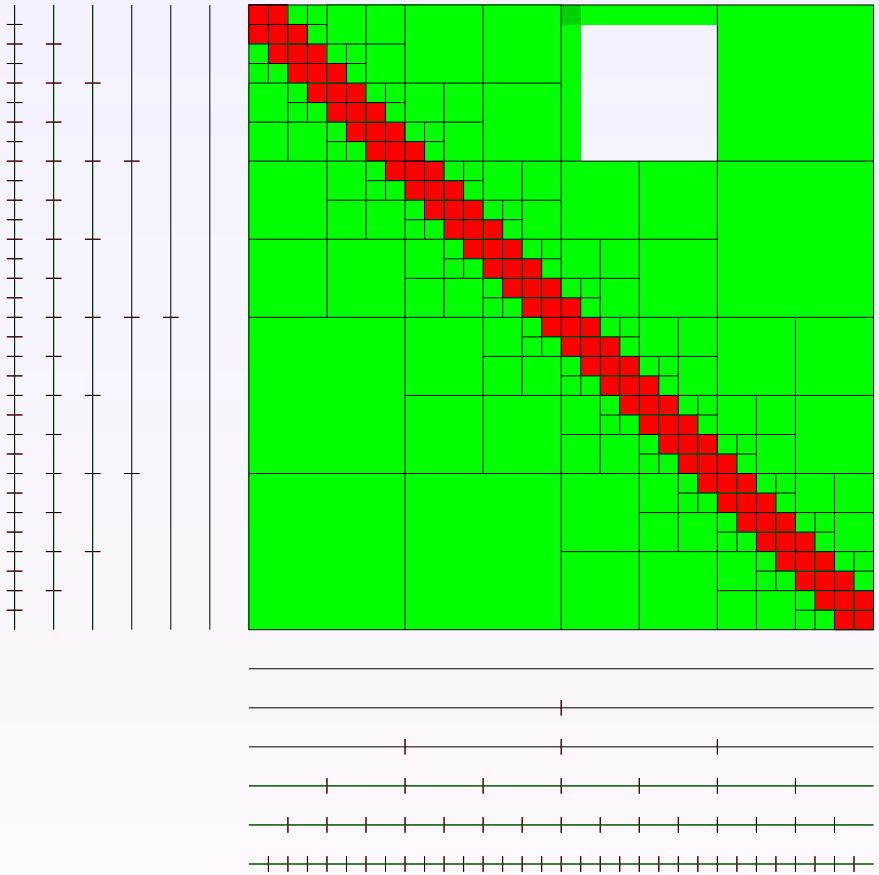
Hierarchical matrix

Idea: Split G into submatrices that can be approximated by low rank.



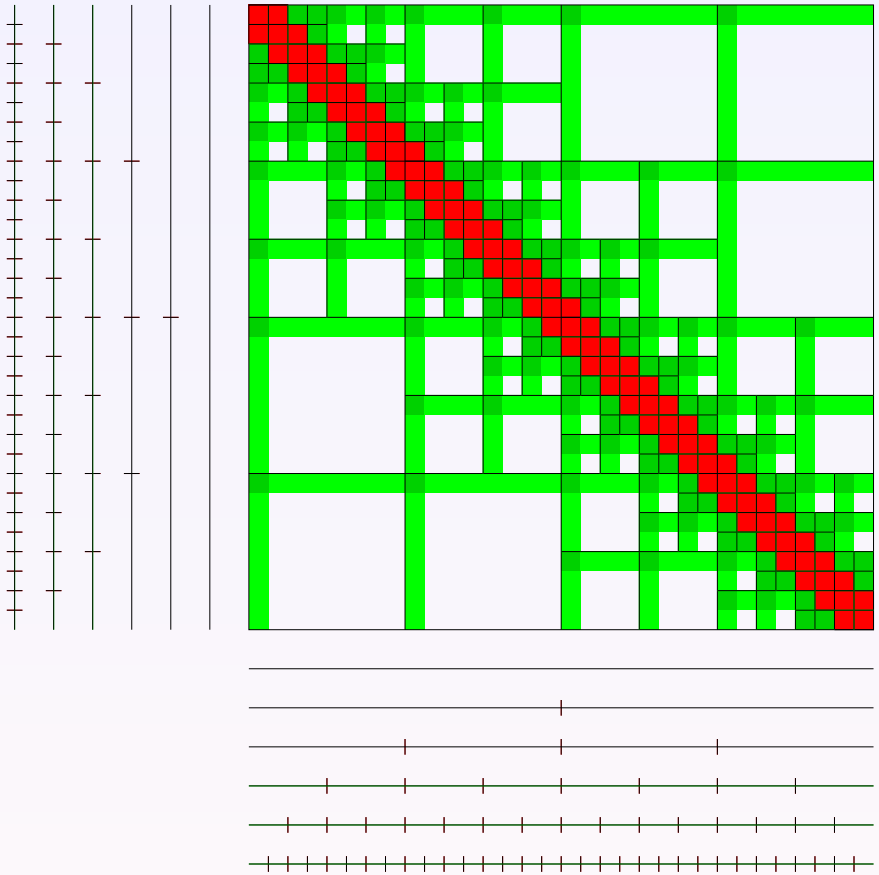
Hierarchical matrix

Idea: Split G into submatrices that can be approximated by low rank.



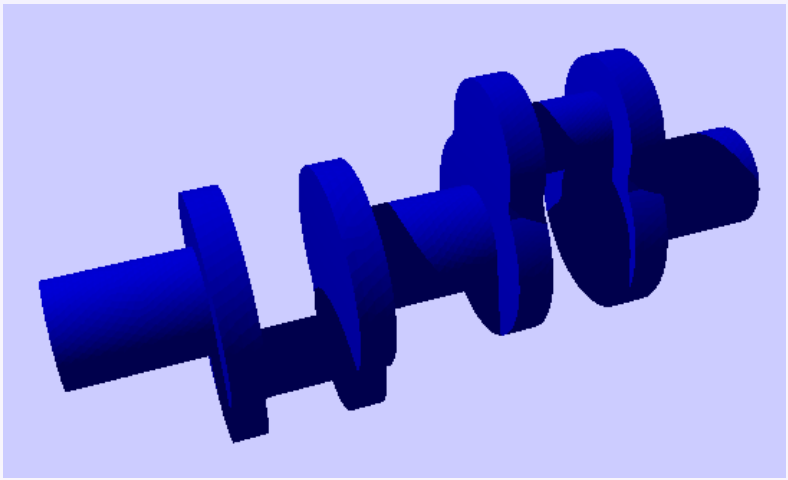
Hierarchical matrix

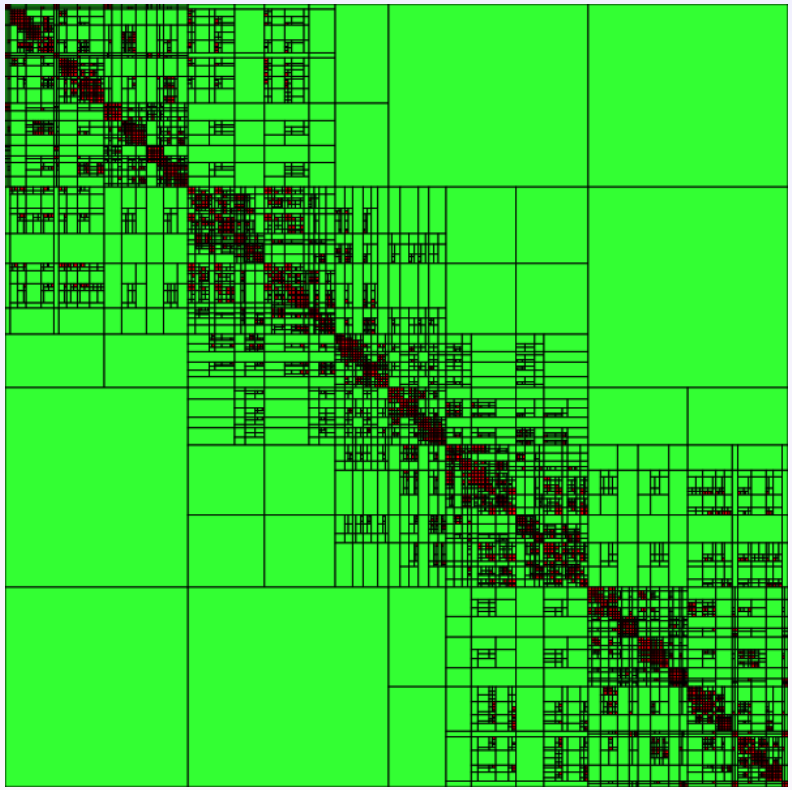
Idea: Split G into submatrices that can be approximated by low rank.



Result: Requires $\mathcal{O}(nk \log n)$ units of storage instead of $\mathcal{O}(n^2)$.

General block structure





General geometries can be handled by adapting the block structure.

Complexity remains in $\mathcal{O}(nk \log n)$.

Overview

Analytical approximation

Taylor expansion of the kernel function

$$g(x,y) \approx \sum_{\nu} \frac{(x-x_t)^{\nu}}{\nu!} \frac{\partial^{\nu} g}{\partial x^{\nu}}(x_t,y)$$

Lagrange interpolation of the kernel function

$$g(x,y) \approx \sum_{\nu} \mathcal{L}_{t,\nu}(x) g(\xi_{t,\nu},y)$$

Quadrature and a representation (e.g., Helmholtz-Kirchhoff) formula

$$g(x,y) \approx \sum_{\nu} w_{\nu} \left(g(x,\xi_{s,\nu}) \frac{\partial g}{\partial n(\xi_{s,\nu})}(\xi_{s,\nu},y) - \frac{\partial g}{\partial n(\xi_{s,\nu})}(x,\xi_{s,\nu}) g(\xi_{s,\nu},y) \right)$$

Algebraic approximation

Idea: Find low-rank approximation based only on matrix entries.

Cross approximation based on rank-revealing LR decomposition

$$G|_{t\times s} = \begin{pmatrix} L_{11} & \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix}$$

Algebraic approximation

Idea: Find low-rank approximation based only on matrix entries.

Cross approximation based on rank-revealing LR decomposition

$$G|_{t\times s} \approx \begin{pmatrix} L_{11} \\ L_{21} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \end{pmatrix}$$

Algebraic approximation

Idea: Find low-rank approximation based only on matrix entries.

Cross approximation based on rank-revealing LR decomposition

$$G|_{t\times s} \approx \begin{pmatrix} L_{11} \\ L_{21} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \end{pmatrix}$$

Best approximation based on singular value decomposition

$$G|_{t\times s} = U \begin{pmatrix} \sigma_1 & & & & \\ & \ddots & & & \\ & & \sigma_k & & \\ & & & \sigma_{k+1} & \\ & & & & \ddots \end{pmatrix} V^*$$

Algebraic approximation

Idea: Find low-rank approximation based only on matrix entries.

Cross approximation based on rank-revealing LR decomposition

$$G|_{t\times s} \approx \begin{pmatrix} L_{11} \\ L_{21} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \end{pmatrix}$$

Best approximation based on singular value decomposition

$$G|_{t\times s} \approx U \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{pmatrix} V^*$$

Algebraic approximation

Idea: Find low-rank approximation based only on matrix entries.

Cross approximation based on rank-revealing LR decomposition

$$G|_{t\times s} \approx \begin{pmatrix} L_{11} \\ L_{21} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \end{pmatrix}$$

Best approximation based on singular value decomposition

$$G|_{t\times s} \approx U \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{pmatrix} V^*$$

Advantage: Can be applied to arbitrary matrices.

Overview

Preconditioners

Challenge: Solve linear system

$$Gz = b,$$

where G is an ill-conditioned matrix.

Idea: Multiply by preconditioner M to obtain

$$MGz = Mb, \qquad \widehat{G}z = \widehat{b},$$

where $\widehat{G} := MG$ is well-conditioned. Apply efficient Krylov methods.

Hierarchical matrices: Compute $M \approx G^{-1}$ directly or use an approximate LR factorization $G \approx LR$ and let $M := R^{-1}L^{-1}$.

Block inverse

Block LR factorization yields

$$G = \begin{pmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{pmatrix} = \begin{pmatrix} I & \\ G_{21} G_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} G_{11} & G_{12} \\ & G_{22} - G_{21} G_{11}^{-1} G_{12} \end{pmatrix}.$$

Denoting the Schur complement by $S := G_{22} - G_{21} G_{11}^{-1} G_{12}$, we find

$$G^{-1} = \begin{pmatrix} G_{11}^{-1} & -G_{11}^{-1} G_{12} S^{-1} \\ & S^{-1} \end{pmatrix} \begin{pmatrix} I & \\ -G_{21} G_{11}^{-1} & I \end{pmatrix}.$$

Result: Inverse can be represented by products and inverses of submatrices. The latter can be handled simply by recursion.

Algorithm: Use of the block structure of hierarchical matrices, approximate inverse recursively.

Algebraic operations

Matrix multiplication is currently the central algorithm.

$$XY = \begin{pmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{pmatrix} \begin{pmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{pmatrix}$$

Algebraic operations

Matrix multiplication is currently the central algorithm.

$$XY = \begin{pmatrix} X_{11} Y_{11} + X_{12} Y_{21} & X_{12} Y_{22} + X_{11} Y_{12} \\ X_{21} Y_{11} + X_{22} Y_{21} & X_{22} Y_{22} + X_{21} Y_{12} \end{pmatrix}$$

Use recursion until low-rank matrices are reached, then take advantage of factorized representation.

Algebraic operations

Matrix multiplication is currently the central algorithm.

$$XY = \begin{pmatrix} X_{11} Y_{11} + X_{12} Y_{21} & X_{12} Y_{22} + X_{11} Y_{12} \\ X_{21} Y_{11} + X_{22} Y_{21} & X_{22} Y_{22} + X_{21} Y_{12} \end{pmatrix}$$

Use recursion until low-rank matrices are reached, then take advantage of factorized representation.

Matrix inverse can be approximated using multiplication and recursion.

LR factorization can also be reduced to multiplication and recursion.

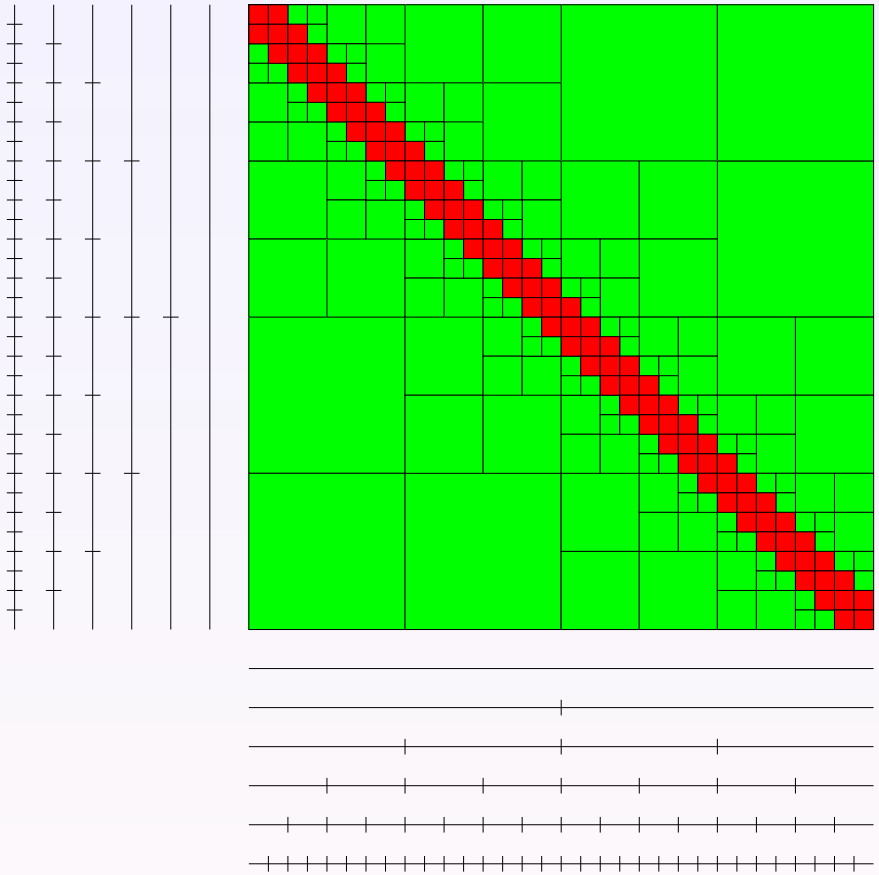
Complexity $\mathcal{O}(nk^2 \log^2 n)$ in time and $\mathcal{O}(nk \log^2 n)$ in storage.

Overview

1. What are hierarchical matrices?
2. How can we construct hierarchical matrices?
3. How can we solve linear systems efficiently?
4. What are current research priorities?

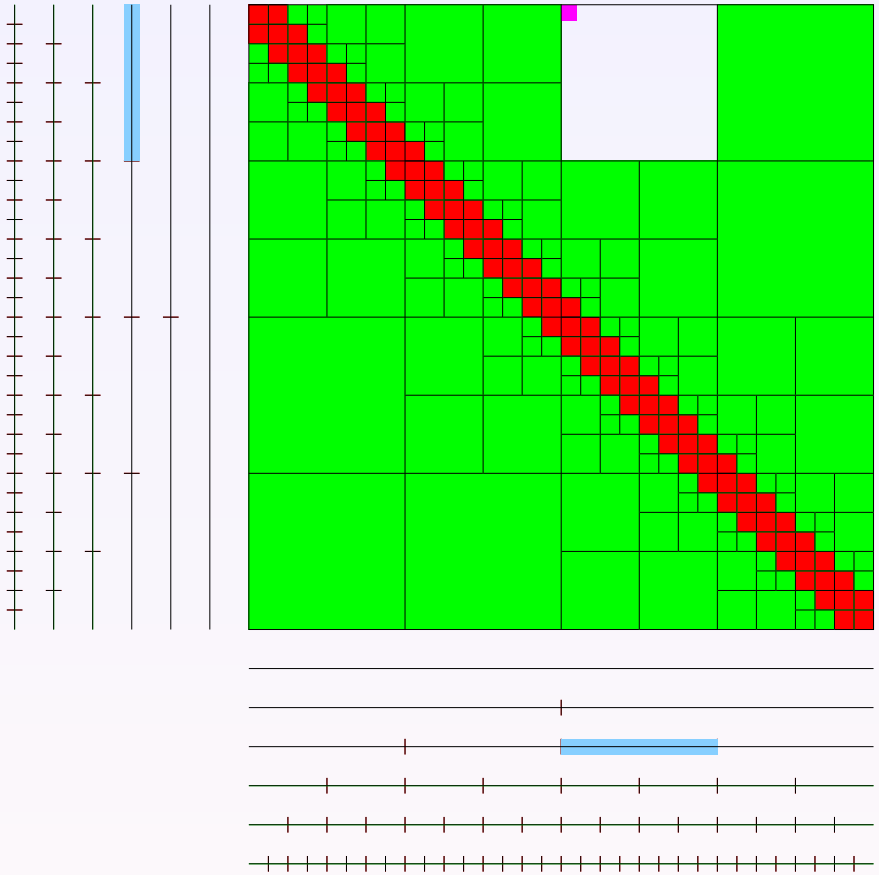
$\mathcal{H}^2$-matrices

Idea: Represent submatrices of G in hierarchical bases.



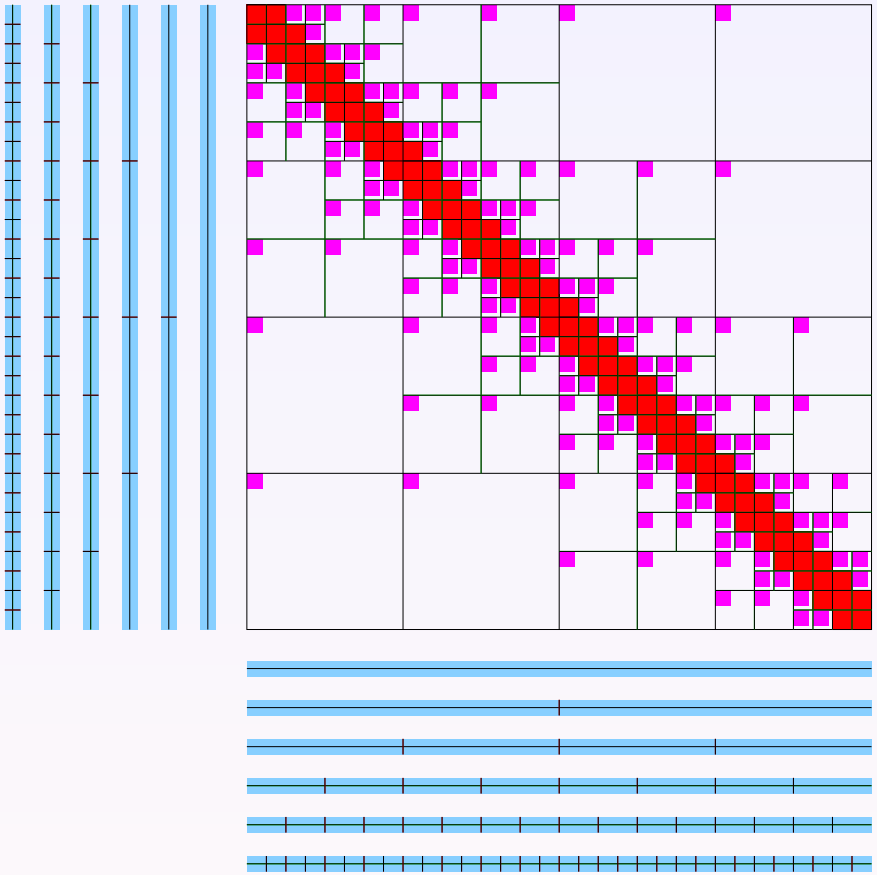
$\mathcal{H}^2$-matrices

Idea: Represent submatrices of G in hierarchical bases.



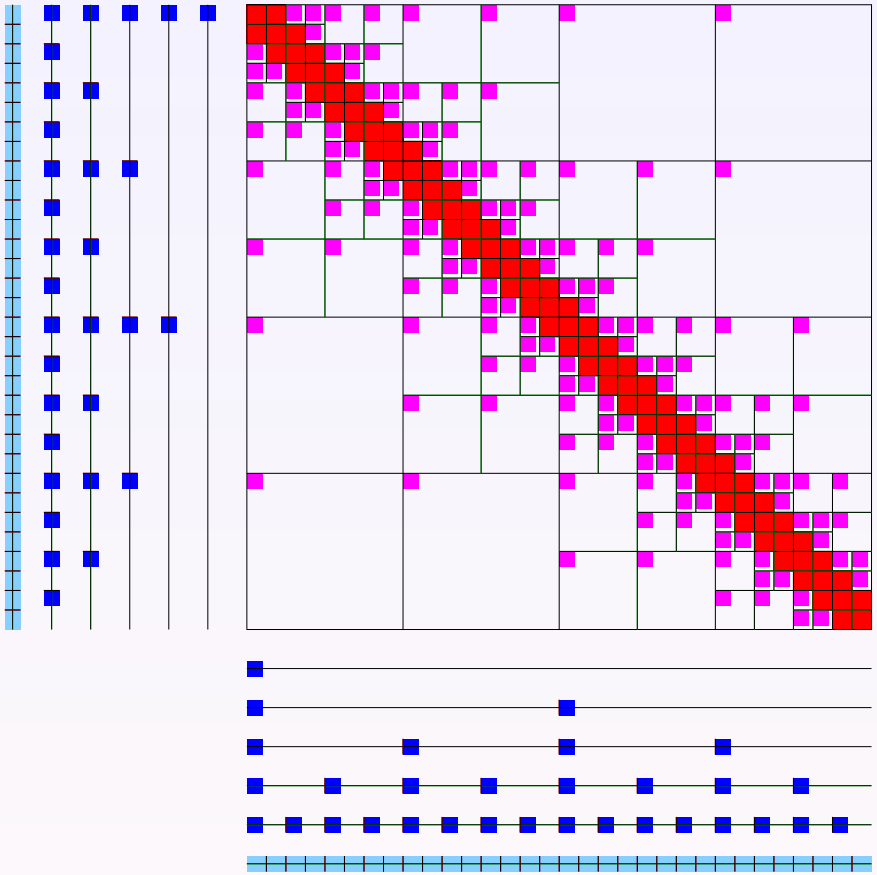
$\mathcal{H}^2$-matrices

Idea: Represent submatrices of G in hierarchical bases.



$\mathcal{H}^2$-matrices

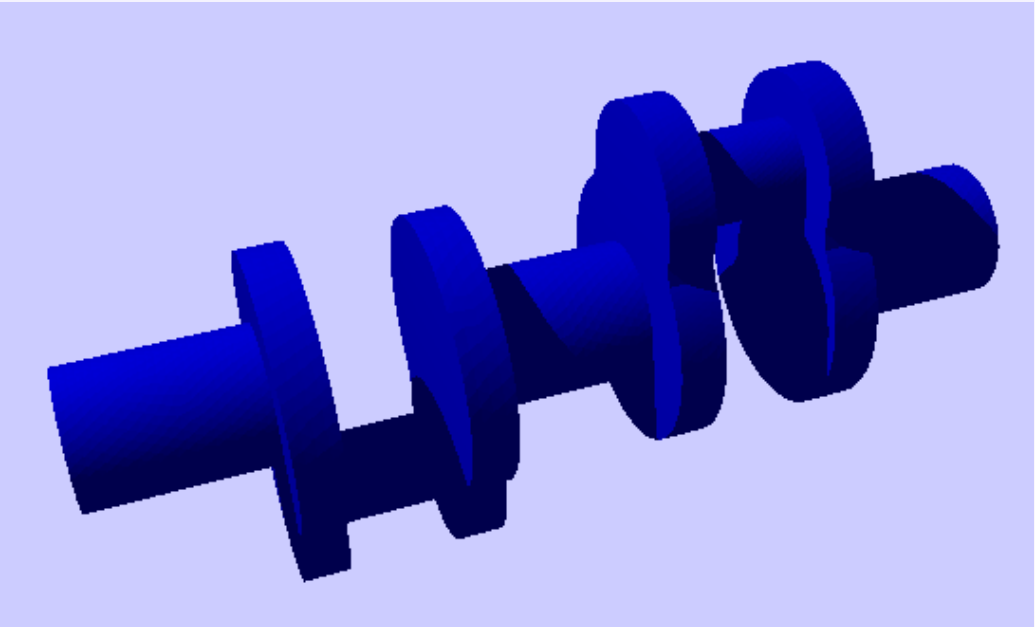
Idea: Represent submatrices of G in hierarchical bases.



Result: Requires $\mathcal{O}(nk)$ units of storage instead of $\mathcal{O}(n^2)$.

Comparison of $\mathcal{H}$- and $\mathcal{H}^2$-matrices

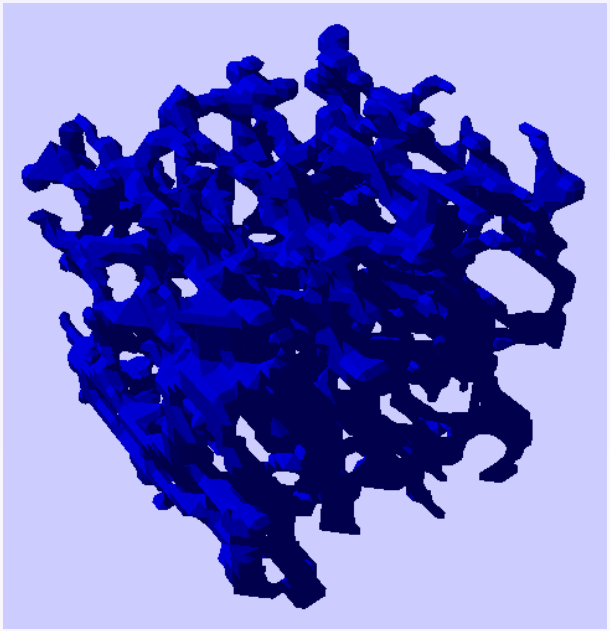
Model problem: Approximate the single layer potential operator.

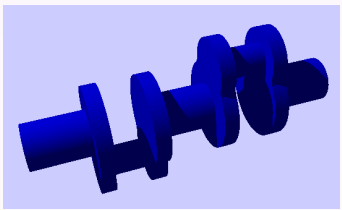


Crank shaft,
taken from
Joachim Schöberl's
NetGen package

Comparison of $\mathcal{H}$- and $\mathcal{H}^2$-matrices

Model problem: Approximate the single layer potential operator.



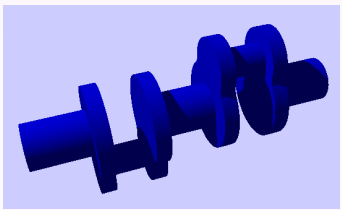


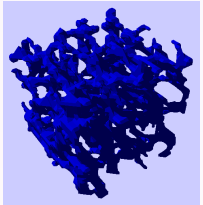
Nickle foam,
courtesy of
von Heiko Andrä
and Günther Of

Comparison of $\mathcal{H}$- and $\mathcal{H}^2$-matrices

Model problem: Approximate the single layer potential operator.

	$\mathcal{H}$-matrix			$\mathcal{H}^2$-matrix		
n	Build	Mem	Error	Build	Mem	Error
25 744	67	181	1.9_{-8}	68	135	1.4_{-8}
102 976	283	1 136	1.2_{-9}	289	551	8.4_{-10}
411 904	1 416	6 757	5.9_{-11}	1 464	2 225	5.1_{-11}
1 647 616	6 248	37 196	3.8_{-12}	6 627	9 795	3.0_{-12}
6 590 464	33 982	198 867	2.2_{-13}	36 224	41 611	1.8_{-13}
28 952	178	294	8.8_{-8}	172	387	7.0_{-8}
115 808	682	1 841	5.8_{-9}	700	1 228	3.7_{-9}
463 232	3 145	11 365	2.8_{-10}	3 176	4 933	2.2_{-10}
1 852 928	17 384	69 571	1.2_{-11}	17 629	15 905	1.3_{-11}





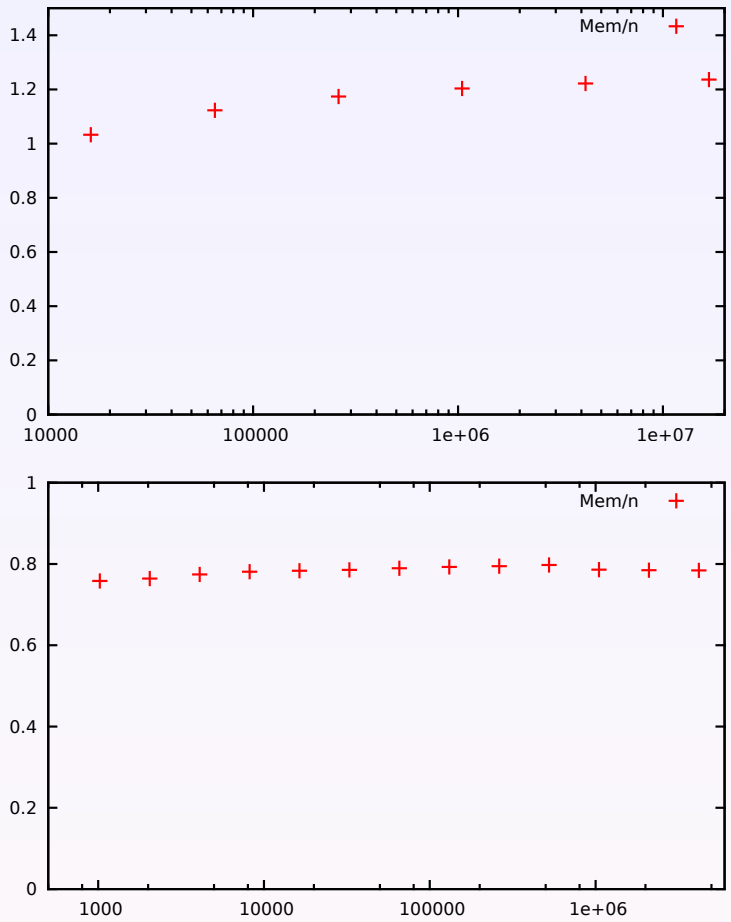
Algebraic operations with $\mathcal{H}^2$-matrices

Goal: Multiplication of $\mathcal{H}^2$-matrices.

Challenge: How to find good bases?

Complexity: $\mathcal{O}(nk^2 \log n)$ in time, $\mathcal{O}(nk)$ in storage.

Applications: Preconditioners for elliptic FEM and BEM problems.



(joint work with Knut Reimer)

Conclusion

Hierarchical matrices take advantage of local low-rank structures to compress dense matrices.

Algebraic operations can be carried out efficiently, leading to robust and fast preconditioners.

Research:

- $\mathcal{H}^2$-matrices for large problems.
- Parallelization of $\mathcal{H}$- and $\mathcal{H}^2$-matrix algorithms.
- Directional $\mathcal{H}^2$-matrices for high-frequency problems.
- Tensor approximation schemes for higher-dimensional problems, e.g., stochastic PDEs.