$\mathcal{H}^2$-Matrices

Steffen Börm
with S. Christophersen, J. Gördes, K. Reimer, D. Boysen

Christian-Albrechts-Universität, Kiel

Kolloquium für Angewandte Mathematik
TU Hamburg-Harburg, 5.12.2014

Overview

Overview

Model problem

Goal: Given $g(x, y) = -\log |x - y|$, solve the integral equation

$$\int_0^1 g(x, y) u(y)\, dy = f(x) \qquad \text{für } x \in [0, 1].$$

Discretization leads to system $Gu = b$ with a matrix $G \in \mathbb{R}^{n \times n}$, given by

$$G_{ij} = \int_{(i-1)/n}^{i/n} \int_{(j-1)/n}^{j/n} g(x, y)\, dy\, dx \qquad \text{for } 1 \leq i, j \leq n.$$

Problem: High accuracy requires $n \gg 1$, storage requirement $\sim n^2$.

Idea: Approximate G by a matrix that can be represented efficiently.

Degenerate approximation

Analysis: Let $\tau \subseteq [0,1]$ be an interval, $\sigma \subseteq [0,1]$, and

$$\tilde{g}_k(x,y) := \sum_{\nu=1}^{k} \mathcal{L}_{\tau,\nu}(x) g(\xi_{\tau,\nu}, y) \qquad \text{for } x \in \tau,\ y \in \sigma$$

the interpolating polynomial in $\xi_{\tau,1}, \ldots, \xi_{\tau,k} \in \tau$. We find

$$\|g - \tilde{g}_k\|_{\infty,\tau\times\sigma} \leq 2 \left(\frac{\operatorname{diam}(\tau)}{4 \operatorname{dist}(\tau,\sigma)} \right)^k$$

Degenerate approximation

Analysis: Let $\tau \subseteq [0,1]$ be an interval, $\sigma \subseteq [0,1]$, and

$$\tilde{g}_k(x,y) := \sum_{\nu=1}^{k} \mathcal{L}_{\tau,\nu}(x) g(\xi_{\tau,\nu}, y) \qquad \text{for } x \in \tau,\ y \in \sigma$$

the interpolating polynomial in $\xi_{\tau,1}, \ldots, \xi_{\tau,k} \in \tau$. We find

$$\|g - \tilde{g}_k\|_{\infty,\tau\times\sigma} \leq 2 \left(\frac{\operatorname{diam}(\tau)}{4 \operatorname{dist}(\tau,\sigma)} \right)^k \lesssim 4^{-k}$$

if we have $\operatorname{diam}(\tau) \leq \operatorname{dist}(\tau,\sigma)$.

Result: g can be approximated by degenerate kernel function, approximation converges exponentially as k grows.

Approximate factorization

Idea: Use approximation of g to construct approximation of G.

$$G_{ij} = \int_{(i-1)/n}^{i/n} \int_{(j-1)/n}^{j/n} g(x,y)\,dy\,dx$$

Approximate factorization

Idea: Use approximation of g to construct approximation of G.

$$G_{ij} \approx \int_{(i-1)/n}^{i/n} \int_{(j-1)/n}^{j/n} \sum_{\nu=1}^{k} \mathcal{L}_{\tau,\nu}(x) g(\xi_{\tau,\nu}, y) \, dy \, dx$$

Approximate factorization

Idea: Use approximation of g to construct approximation of G.

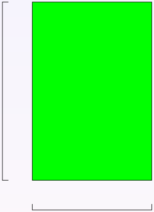
$$G_{ij} \approx \sum_{\nu=1}^{k} \int_{(i-1)/n}^{i/n} \mathcal{L}_{\tau,\nu}(x)\,dx \int_{(j-1)/n}^{j/n} g(\xi_{\tau,\nu}, y)\,dy$$

Approximate factorization

Idea: Use approximation of g to construct approximation of G.

$$G_{ij} \approx \sum_{\nu=1}^{k} \underbrace{\int_{(i-1)/n}^{i/n} \mathcal{L}_{\tau,\nu}(x)\,dx}_{=:A_{i\nu}} \underbrace{\int_{(j-1)/n}^{j/n} g(\xi_{\tau,\nu}, y)\,dy}_{=:B_{j\nu}} = (AB^*)_{ij}.$$

Block approximation: Choosing sets $\hat{\tau}, \hat{\sigma} \subseteq \{1, \ldots, n\}$ with



$$[(i-1)/n, i/n] \subseteq \tau, \qquad \text{für alle } i \in \hat{\tau},$$
$$[(j-1)/n, j/n] \subseteq \sigma \qquad \text{für alle } j \in \hat{\sigma},$$

we find $A \in \mathbb{R}^{\hat{\tau} \times k}$, $B \in \mathbb{R}^{\hat{\sigma} \times k}$ such that

$$\|G|_{\hat{\tau} \times \hat{\sigma}} - AB^*\|_2 \lesssim 4^{-k}.$$

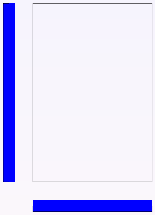
Approximate factorization

Idea: Use approximation of g to construct approximation of G.

$$G_{ij} \approx \sum_{\nu=1}^{k} \underbrace{\int_{(i-1)/n}^{i/n} \mathcal{L}_{\tau,\nu}(x)\,dx}_{=:A_{i\nu}} \underbrace{\int_{(j-1)/n}^{j/n} g(\xi_{\tau,\nu}, y)\,dy}_{=:B_{j\nu}} = (AB^*)_{ij}.$$

Block approximation: Choosing sets $\hat{\tau}, \hat{\sigma} \subseteq \{1, \ldots, n\}$ with

$$[(i-1)/n, i/n] \subseteq \tau, \qquad \text{für alle } i \in \hat{\tau},$$
$$[(j-1)/n, j/n] \subseteq \sigma \qquad \text{für alle } j \in \hat{\sigma},$$



we find $A \in \mathbb{R}^{\hat{\tau} \times k}$, $B \in \mathbb{R}^{\hat{\sigma} \times k}$ such that

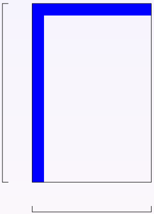
$$\|G|_{\hat{\tau}\times\hat{\sigma}} - AB^*\|_2 \lesssim 4^{-k}.$$

Approximate factorization

Idea: Use approximation of g to construct approximation of G.

$$G_{ij} \approx \sum_{\nu=1}^{k} \underbrace{\int_{(i-1)/n}^{i/n} \mathcal{L}_{\tau,\nu}(x)\,dx}_{=:A_{i\nu}} \underbrace{\int_{(j-1)/n}^{j/n} g(\xi_{\tau,\nu}, y)\,dy}_{=:B_{j\nu}} = (AB^*)_{ij}.$$

Block approximation: Choosing sets $\hat{\tau}, \hat{\sigma} \subseteq \{1, \ldots, n\}$ with



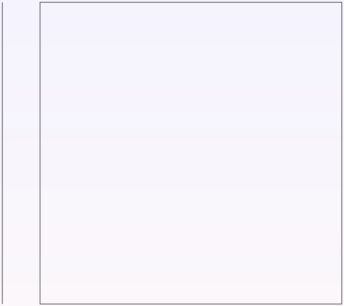
$$[(i-1)/n, i/n] \subseteq \tau, \qquad \text{für alle } i \in \hat{\tau},$$
$$[(j-1)/n, j/n] \subseteq \sigma \qquad \text{für alle } j \in \hat{\sigma},$$

we find $A \in \mathbb{R}^{\hat{\tau} \times k}$, $B \in \mathbb{R}^{\hat{\sigma} \times k}$ such that

$$\|G|_{\hat{\tau}\times\hat{\sigma}} - AB^*\|_2 \lesssim 4^{-k}.$$

Hierarchical matrix

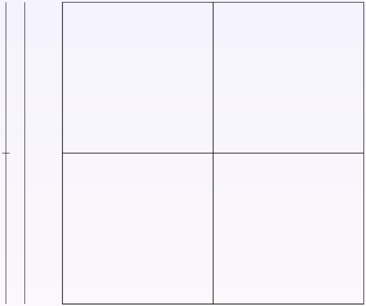
Idea: Split G into submatrices, that can be approximated ...



Approximation of $\tau \times \sigma$ admissible, if $\operatorname{diam}(\tau) \leq \operatorname{dist}(\tau, \sigma)$.

Hierarchical matrix

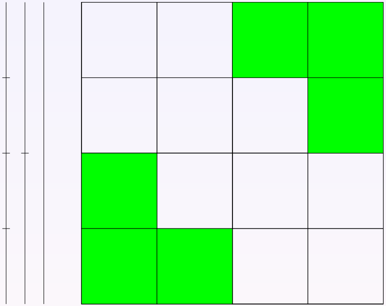
Idea: Split G into submatrices, that can be approximated ...



Approximation of $\tau \times \sigma$ admissible, if $\mathrm{diam}(\tau) \leq \mathrm{dist}(\tau, \sigma)$.

Hierarchical matrix

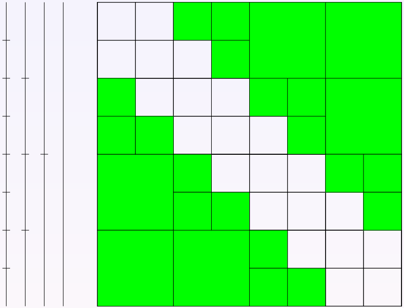
Idea: Split G into submatrices, that can be approximated ...



Approximation of $\tau \times \sigma$ admissible, if $\operatorname{diam}(\tau) \leq \operatorname{dist}(\tau, \sigma)$.

Hierarchical matrix

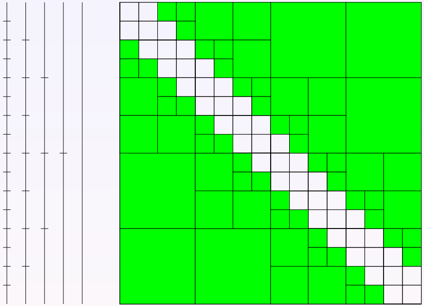
Idea: Split G into submatrices, that can be approximated ...



Approximation of $\tau \times \sigma$ admissible, if $\operatorname{diam}(\tau) \leq \operatorname{dist}(\tau, \sigma)$.

Hierarchical matrix

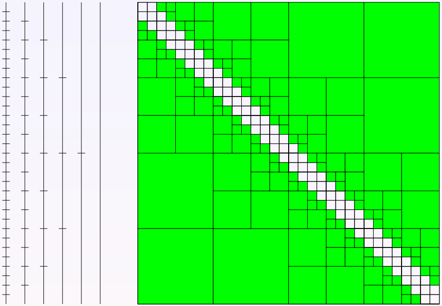
Idea: Split G into submatrices, that can be approximated ...



Approximation of $\tau \times \sigma$ admissible, if $\operatorname{diam}(\tau) \leq \operatorname{dist}(\tau, \sigma)$.

Hierarchical matrix

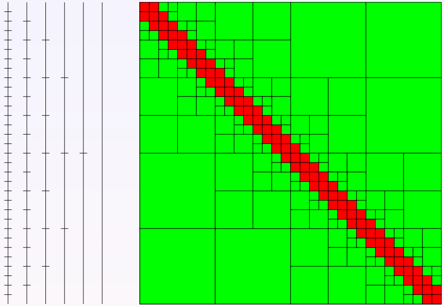
Idea: Split G into submatrices, that can be approximated ...



Approximation of $\tau \times \sigma$ admissible, if $\operatorname{diam}(\tau) \leq \operatorname{dist}(\tau, \sigma)$.

Hierarchical matrix

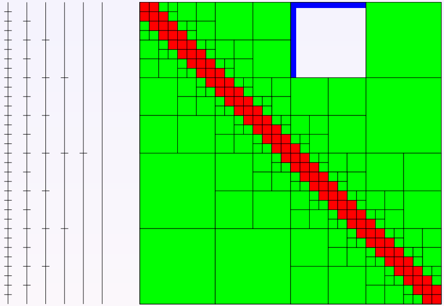
Idea: Split G into submatrices, that can be approximated and a remainder that is sparse.



Matrix split into blocks $b = (\tau, \sigma)$, constructed from clusters τ, σ.

Hierarchical matrix

Idea: Split G into submatrices, that can be approximated and a remainder that is sparse.



Matrix split into blocks $b = (\tau, \sigma)$, constructed from clusters τ, σ.

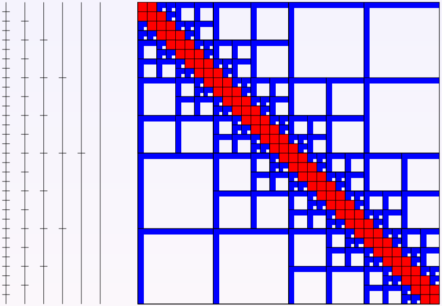
Admissible blocks stored in factorized form

$$G|_{\hat{\tau} \times \hat{\sigma}} \approx A_b B_b^*$$

with $A_b \in \mathbb{R}^{\hat{\tau} \times k}$, $B_b \in \mathbb{R}^{\hat{\sigma} \times k}$.

Hierarchical matrix

Idea: Split G into submatrices, that can be approximated and a remainder that is sparse.



Matrix split into blocks $b = (\tau, \sigma)$, constructed from clusters τ, σ.

Admissible blocks stored in factorized form

$$G|_{\hat{\tau}\times\hat{\sigma}} \approx A_b B_b^*$$

with $A_b \in \mathbb{R}^{\hat{\tau}\times k}$, $B_b \in \mathbb{R}^{\hat{\sigma}\times k}$.

Result: Hierarchical matrix, storage $\sim nk \log_2 n$.

Overview

Uniform $\mathcal{H}$-matrix

Idea: Interpolation in both variables.

$$\tilde{g}_k(x,y) \approx \sum_{\nu=1}^{k} \sum_{\mu=1}^{k} \mathcal{L}_{\tau,\nu}(x) g(\xi_{\tau,\nu}, \xi_{\sigma,\mu}) \mathcal{L}_{\sigma,\mu}(y)$$

Uniform $\mathcal{H}$-matrix

Idea: Interpolation in both variables.

$$G_{ij} \approx \sum_{\nu=1}^{k} \sum_{\mu=1}^{k} \underbrace{\int_{(i-1)/n}^{i/n} \mathcal{L}_{\tau,\nu}(x)\,dx}_{=:V_{\tau,i\nu}} \underbrace{g(\xi_{\tau,\nu}, \xi_{\sigma,\mu})}_{=:S_{b,\nu\mu}} \underbrace{\int_{(j-1)/n}^{j/n} \mathcal{L}_{\sigma,\mu}(y)\,dy}_{=:W_{\sigma,j\mu}}$$

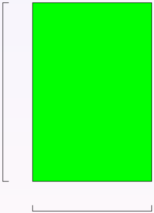
Uniform $\mathcal{H}$-matrix

Idea: Interpolation in both variables.

$$G_{ij} \approx \sum_{\nu=1}^{k} \sum_{\mu=1}^{k} \underbrace{\int_{(i-1)/n}^{i/n} \mathcal{L}_{\tau,\nu}(x)\,dx}_{=:V_{\tau,i\nu}} \underbrace{g(\xi_{\tau,\nu}, \xi_{\sigma,\mu})}_{=:S_{b,\nu\mu}} \underbrace{\int_{(j-1)/n}^{j/n} \mathcal{L}_{\sigma,\mu}(y)\,dy}_{=:W_{\sigma,j\mu}}$$

Block approximation by three-term factorization

$$G|_{\hat\tau \times \hat\sigma} \approx V_\tau S_{\tau\sigma} W_\sigma^*.$$



Cluster bases: V_τ depends only on τ, W_σ only on σ.

Coupling matrices $S_{\tau\sigma}$ are $k \times k$-matrices.

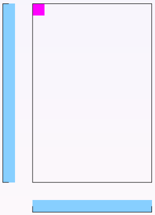
Uniform $\mathcal{H}$-matrix

Idea: Interpolation in both variables.

$$G_{ij} \approx \sum_{\nu=1}^{k} \sum_{\mu=1}^{k} \underbrace{\int_{(i-1)/n}^{i/n} \mathcal{L}_{\tau,\nu}(x)\,dx}_{=:V_{\tau,i\nu}} \underbrace{g(\xi_{\tau,\nu}, \xi_{\sigma,\mu})}_{=:S_{b,\nu\mu}} \underbrace{\int_{(j-1)/n}^{j/n} \mathcal{L}_{\sigma,\mu}(y)\,dy}_{=:W_{\sigma,j\mu}}$$

Block approximation by three-term factorization

$$G|_{\hat\tau\times\hat\sigma} \approx V_\tau S_{\tau\sigma} W_\sigma^*.$$

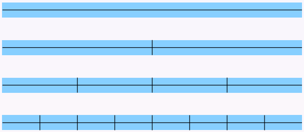


Cluster bases: V_τ depends only on τ, W_σ only on σ.

Coupling matrices $S_{\tau\sigma}$ are $k \times k$-matrices.

Cluster basis

Challenge: Matrices V_τ require storage $\sim nk \log n$.

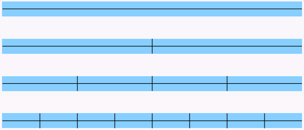


Cluster basis

Challenge: Matrices V_τ require storage $\sim nk \log n$.

Idea: Interpolation in τ' reproduces $\mathcal{L}_{t,\nu}$.

$$\mathcal{L}_{\tau,\nu}(x) = \sum_{\mu=1}^{k} \mathcal{L}_{\tau',\mu}(x)\mathcal{L}_{\tau,\nu}(\xi_{\tau',\mu}),$$

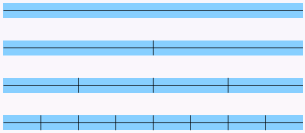


Cluster basis

Challenge: Matrices V_τ require storage $\sim nk \log n$.

Idea: Interpolation in τ' reproduces $\mathcal{L}_{t,\nu}$.

$$\mathcal{L}_{\tau,\nu}(x) = \sum_{\mu=1}^{k} \mathcal{L}_{\tau',\mu}(x)\mathcal{L}_{\tau,\nu}(\xi_{\tau',\mu}), \qquad V_\tau|_{\hat{\tau}'\times k} = V_{\tau'} E_{\tau'}.$$



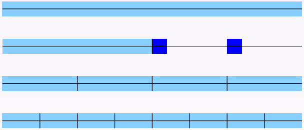
Cluster basis

Challenge: Matrices V_τ require storage $\sim nk \log n$.

Idea: Interpolation in τ' reproduces $\mathcal{L}_{t,\nu}$.

$$\mathcal{L}_{\tau,\nu}(x) = \sum_{\mu=1}^{k} \mathcal{L}_{\tau',\mu}(x)\mathcal{L}_{\tau,\nu}(\xi_{\tau',\mu}), \qquad V_\tau|_{\hat{\tau}'\times k} = V_{\tau'}E_{\tau'}.$$

Store transfer matrices $E_{\tau'}$ instead of V_τ, keep V_τ only in leaf clusters.



Cluster basis

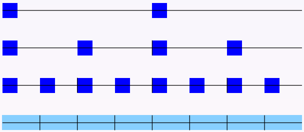
Challenge: Matrices V_τ require storage $\sim nk \log n$.

Idea: Interpolation in τ' reproduces $\mathcal{L}_{t,\nu}$.

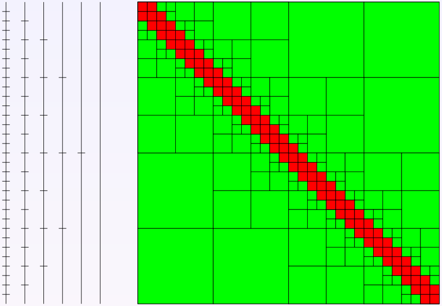
$$\mathcal{L}_{\tau,\nu}(x) = \sum_{\mu=1}^{k} \mathcal{L}_{\tau',\mu}(x)\mathcal{L}_{\tau,\nu}(\xi_{\tau',\mu}), \qquad V_\tau|_{\hat\tau' \times k} = V_{\tau'} E_{\tau'}.$$

Store transfer matrices $E_{\tau'}$ instead of V_τ, keep V_τ only in leaf clusters.

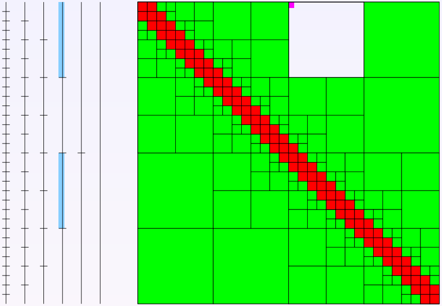
Result: Storage $\sim nk$.



$\mathcal{H}^2$-matrix



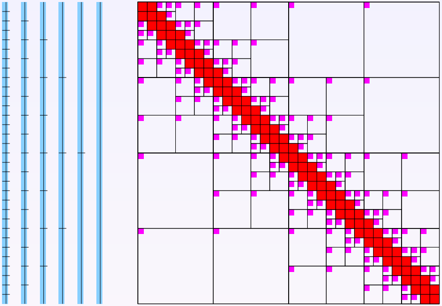
$\mathcal{H}^2$-matrix



Admissible blocks represented by coupling matrices

$$G|_{\hat{\tau}\times\hat{\sigma}} \approx V_\tau S_{\tau\sigma} W_\sigma^*.$$

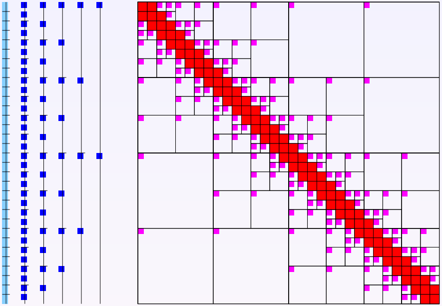
$\mathcal{H}^2$-matrix



Admissible blocks represented by coupling matrices

$$G|_{\hat{\tau}\times\hat{\sigma}} \approx V_\tau S_{\tau\sigma} W_\sigma^*.$$

$\mathcal{H}^2$-matrix



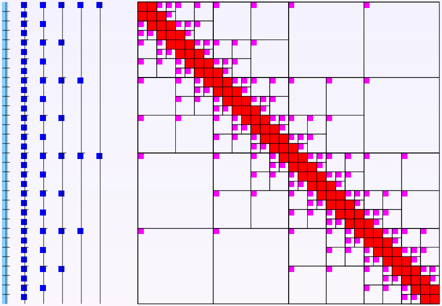
Admissible blocks represented by coupling matrices

$$G|_{\hat{\tau}\times\hat{\sigma}} \approx V_\tau S_{\tau\sigma} W_\sigma^*.$$

Cluster bases represented by transfer matrices

$$V_\tau|_{\hat{\tau}'\times k} = V_{\tau'} E_{\tau'}.$$

$\mathcal{H}^2$-matrix



Admissible blocks represented by coupling matrices

$$G|_{\hat{\tau}\times\hat{\sigma}} \approx V_\tau S_{\tau\sigma} W_\sigma^*.$$

Cluster bases represented by transfer matrices

$$V_\tau|_{\hat{\tau}'\times k} = V_{\tau'} E_{\tau'}.$$

Result: $\mathcal{H}^2$-matrix, storage $\sim nk$.

Experiment: How important is one logarithm?

Problem: One-dimensional model problem, approximated by $\mathcal{H}$- and $\mathcal{H}^2$-matrices using interpolation.

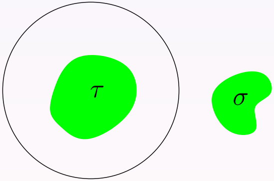
	$\mathcal{H}$-matrix			$\mathcal{H}^2$-matrix			$\mathcal{H}^2$-matrix (var.)		
n	Bld	M/n	Err	Bld	M/n	Err	Bld	M/n	Err
128	0.00	0.6	2.8_{-5}	0.00	0.5	3.0_{-5}	0.00	0.5	1.9_{-4}
256	0.01	0.8	2.7_{-5}	0.00	0.5	3.2_{-5}	0.00	0.5	1.1_{-4}
512	0.02	1.0	2.6_{-5}	0.01	0.5	3.2_{-5}	0.01	0.5	5.7_{-5}
1024	0.04	1.2	2.6_{-5}	0.01	0.5	3.2_{-5}	0.01	0.5	2.9_{-5}
2048	0.10	1.4	2.6_{-5}	0.03	0.5	3.3_{-5}	0.03	0.5	1.4_{-5}
4096	0.22	1.6	2.6_{-5}	0.05	0.5	3.3_{-5}	0.05	0.5	7.3_{-6}
8192	0.50	1.8	2.6_{-5}	0.10	0.5	3.3_{-5}	0.10	0.5	3.6_{-6}
⋮	⋮	⋮		⋮	⋮		⋮	⋮	
524288	52.97	2.9		6.62	0.5		6.62	0.5	

Overview

Approximation by Green's equation

Representation formula: Kernels appearing in BIEs satisfy

$$g(x,y) = \int_{\partial\omega} g(x,z) \frac{\partial g}{\partial n_z}(z,y)\, dz - \int_{\partial\omega} \frac{\partial g}{\partial n_z}(x,z) g(z,y)\, dz.$$



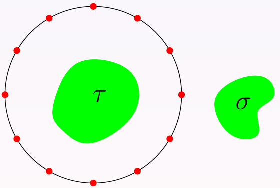
Approximation by Green's equation

Representation formula: Kernels appearing in BIEs satisfy

$$g(x,y) = \int_{\partial\omega} g(x,z) \frac{\partial g}{\partial n_z}(z,y)\, dz - \int_{\partial\omega} \frac{\partial g}{\partial n_z}(x,z) g(z,y)\, dz.$$

Idea: Approximate integrals by quadrature.

$$g(x,y) \approx \sum_{\nu=1}^{k} w_\nu g(x,z_\nu) \frac{\partial g}{\partial n_z}(z_\nu,y) - \sum_{\nu=1}^{k} w_\nu \frac{\partial g}{\partial n_z}(x,z_\nu) g(z_\nu,y).$$



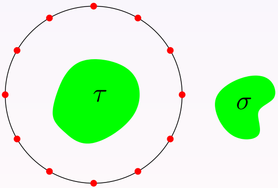
Approximation by Green's equation

Representation formula: Kernels appearing in BIEs satisfy

$$g(x,y) = \int_{\partial\omega} g(x,z)\frac{\partial g}{\partial n_z}(z,y)\,dz - \int_{\partial\omega} \frac{\partial g}{\partial n_z}(x,z)g(z,y)\,dz.$$

Idea: Approximate integrals by quadrature.

$$g(x,y) \approx \sum_{\nu=1}^{k} w_\nu g(x,z_\nu)\frac{\partial g}{\partial n_z}(z_\nu,y) - \sum_{\nu=1}^{k} w_\nu \frac{\partial g}{\partial n_z}(x,z_\nu)g(z_\nu,y).$$



Result: Rank $\sim m^{d-1}$,
while interpolation takes m^d.

Green hybrid method

Observation: Green quadrature method yields

$$G|_{\hat{\tau}\times\hat{\sigma}} \approx A_\tau B_{\tau\sigma}^*,$$

but rank is higher than expected.

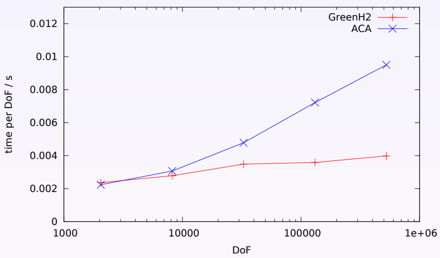
Idea: Apply cross approximation to A_τ to reduce rank:

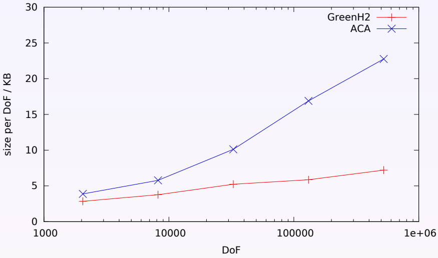
$$A_\tau \approx C_\tau D_\tau^*, \qquad G|_{\hat{\tau}\times\hat{\sigma}} \approx C_\tau D_\tau^* B_{\tau\sigma}^* \approx C_\tau \widehat{B}_{\tau\sigma}^*.$$

Result: Significantly improved efficiency.
Can even be used to construct $\mathcal{H}^2$-matrices.

Experiment: Boundary integral matrix

Goal: Approximate the stiffness matrix for a three-dimensional boundary integral equation.





Result: New method very efficient with regards to time and storage.

LR factorization

Goal: Given an $\mathcal{H}^2$-matrix G, compute its LR factorization $G = LR$.

Approach: If G is inadmissible, compute $G = LR$ directly.
Otherwise, use submatrices

$$\begin{pmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{pmatrix} = \begin{pmatrix} L_{11} & \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix}$$

LR factorization

Goal: Given an $\mathcal{H}^2$-matrix G, compute its LR factorization $G = LR$.

Approach: If G is inadmissible, compute $G = LR$ directly.
Otherwise, use submatrices

$$\begin{pmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{pmatrix} = \begin{pmatrix} L_{11} & \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix}$$

This is equivalent to

$$G_{11} = L_{11} R_{11},$$
$$G_{12} = L_{11} R_{12}, \qquad G_{21} = L_{21} R_{11},$$
$$G_{22} - L_{21} R_{12} = L_{22} R_{22}.$$

If we can compute $Z \leftarrow Z + \alpha XY$, we can find the LR factorization.

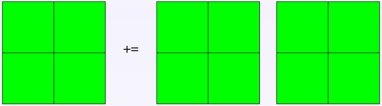
Multiplication

Goal: Perform update $Z|_{\hat{\tau}\times\hat{\rho}} \leftarrow Z|_{\hat{\tau}\times\hat{\rho}} + \alpha X|_{\hat{\tau}\times\hat{\sigma}} Y|_{\hat{\sigma}\times\hat{\rho}}$ efficiently.

Idea: If (τ, σ) and (σ, ρ) are subdivided, treat them by recursion.

Multiplication

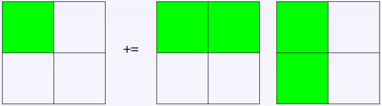
Goal: Perform update $Z|_{\hat\tau\times\hat\rho} \leftarrow Z|_{\hat\tau\times\hat\rho} + \alpha X|_{\hat\tau\times\hat\sigma} Y|_{\hat\sigma\times\hat\rho}$ efficiently.



Idea: If (τ,σ) and (σ,ρ) are subdivided, treat them by recursion.

Multiplication

Goal: Perform update $Z|_{\hat\tau\times\hat\rho} \leftarrow Z|_{\hat\tau\times\hat\rho} + \alpha X|_{\hat\tau\times\hat\sigma} Y|_{\hat\sigma\times\hat\rho}$ efficiently.

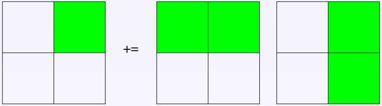


Idea: If (τ,σ) and (σ,ρ) are subdivided, treat them by recursion.

- $Z|_{\hat\tau_1\times\hat\rho_1} \leftarrow Z|_{\hat\tau_1\times\hat\rho_1} + \alpha X|_{\hat\tau_1\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_1} + \alpha X|_{\hat\tau_1\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_1}$

Multiplication

Goal: Perform update $Z|_{\hat\tau\times\hat\rho} \leftarrow Z|_{\hat\tau\times\hat\rho} + \alpha X|_{\hat\tau\times\hat\sigma} Y|_{\hat\sigma\times\hat\rho}$ efficiently.

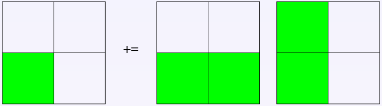


Idea: If (τ, σ) and (σ, ρ) are subdivided, treat them by recursion.

- $Z|_{\hat\tau_1\times\hat\rho_1} \leftarrow Z|_{\hat\tau_1\times\hat\rho_1} + \alpha X|_{\hat\tau_1\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_1} + \alpha X|_{\hat\tau_1\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_1}$
- $Z|_{\hat\tau_1\times\hat\rho_2} \leftarrow Z|_{\hat\tau_1\times\hat\rho_2} + \alpha X|_{\hat\tau_1\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_2} + \alpha X|_{\hat\tau_1\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_2}$

Multiplication

Goal: Perform update $Z|_{\hat{\tau}\times\hat{\rho}} \leftarrow Z|_{\hat{\tau}\times\hat{\rho}} + \alpha X|_{\hat{\tau}\times\hat{\sigma}} Y|_{\hat{\sigma}\times\hat{\rho}}$ efficiently.

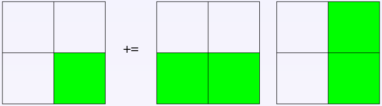


Idea: If (τ, σ) and (σ, ρ) are subdivided, treat them by recursion.

- $Z|_{\hat{\tau}_1\times\hat{\rho}_1} \leftarrow Z|_{\hat{\tau}_1\times\hat{\rho}_1} + \alpha X|_{\hat{\tau}_1\times\hat{\sigma}_1} Y|_{\hat{\sigma}_1\times\hat{\rho}_1} + \alpha X|_{\hat{\tau}_1\times\hat{\sigma}_2} Y|_{\hat{\sigma}_2\times\hat{\rho}_1}$
- $Z|_{\hat{\tau}_1\times\hat{\rho}_2} \leftarrow Z|_{\hat{\tau}_1\times\hat{\rho}_2} + \alpha X|_{\hat{\tau}_1\times\hat{\sigma}_1} Y|_{\hat{\sigma}_1\times\hat{\rho}_2} + \alpha X|_{\hat{\tau}_1\times\hat{\sigma}_2} Y|_{\hat{\sigma}_2\times\hat{\rho}_2}$
- $Z|_{\hat{\tau}_2\times\hat{\rho}_1} \leftarrow Z|_{\hat{\tau}_2\times\hat{\rho}_1} + \alpha X|_{\hat{\tau}_2\times\hat{\sigma}_1} Y|_{\hat{\sigma}_1\times\hat{\rho}_1} + \alpha X|_{\hat{\tau}_2\times\hat{\sigma}_2} Y|_{\hat{\sigma}_2\times\hat{\rho}_1}$

Multiplication

Goal: Perform update $Z|_{\hat\tau\times\hat\rho} \leftarrow Z|_{\hat\tau\times\hat\rho} + \alpha X|_{\hat\tau\times\hat\sigma} Y|_{\hat\sigma\times\hat\rho}$ efficiently.

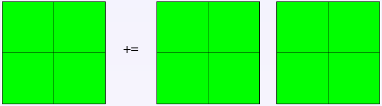


Idea: If (τ,σ) and (σ,ρ) are subdivided, treat them by recursion.

- $Z|_{\hat\tau_1\times\hat\rho_1} \leftarrow Z|_{\hat\tau_1\times\hat\rho_1} + \alpha X|_{\hat\tau_1\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_1} + \alpha X|_{\hat\tau_1\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_1}$
- $Z|_{\hat\tau_1\times\hat\rho_2} \leftarrow Z|_{\hat\tau_1\times\hat\rho_2} + \alpha X|_{\hat\tau_1\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_2} + \alpha X|_{\hat\tau_1\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_2}$
- $Z|_{\hat\tau_2\times\hat\rho_1} \leftarrow Z|_{\hat\tau_2\times\hat\rho_1} + \alpha X|_{\hat\tau_2\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_1} + \alpha X|_{\hat\tau_2\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_1}$
- $Z|_{\hat\tau_2\times\hat\rho_2} \leftarrow Z|_{\hat\tau_2\times\hat\rho_2} + \alpha X|_{\hat\tau_2\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_2} + \alpha X|_{\hat\tau_2\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_2}$

Multiplication

Goal: Perform update $Z|_{\hat\tau\times\hat\rho} \leftarrow Z|_{\hat\tau\times\hat\rho} + \alpha X|_{\hat\tau\times\hat\sigma} Y|_{\hat\sigma\times\hat\rho}$ efficiently.



Idea: If (τ,σ) and (σ,ρ) are subdivided, treat them by recursion.

- $Z|_{\hat\tau_1\times\hat\rho_1} \leftarrow Z|_{\hat\tau_1\times\hat\rho_1} + \alpha X|_{\hat\tau_1\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_1} + \alpha X|_{\hat\tau_1\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_1}$
- $Z|_{\hat\tau_1\times\hat\rho_2} \leftarrow Z|_{\hat\tau_1\times\hat\rho_2} + \alpha X|_{\hat\tau_1\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_2} + \alpha X|_{\hat\tau_1\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_2}$
- $Z|_{\hat\tau_2\times\hat\rho_1} \leftarrow Z|_{\hat\tau_2\times\hat\rho_1} + \alpha X|_{\hat\tau_2\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_1} + \alpha X|_{\hat\tau_2\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_1}$
- $Z|_{\hat\tau_2\times\hat\rho_2} \leftarrow Z|_{\hat\tau_2\times\hat\rho_2} + \alpha X|_{\hat\tau_2\times\hat\sigma_1} Y|_{\hat\sigma_1\times\hat\rho_2} + \alpha X|_{\hat\tau_2\times\hat\sigma_2} Y|_{\hat\sigma_2\times\hat\rho_2}$

Multiplication with low-rank block

Important case: What happens if one of the factors is not subdivided?

Multiplication with low-rank block

Important case: What happens if one of the factors is not subdivided?

Example: Let (σ, ρ) be an admissible block of Y.

$$X|_{\hat\tau\times\hat\sigma} Y|_{\hat\sigma\times\hat\rho} = X|_{\hat\tau\times\hat\sigma}(V_\sigma S_{\sigma\rho} W_\rho^*)$$

Multiplication with low-rank block

Important case: What happens if one of the factors is not subdivided?

Example: Let (σ, ρ) be an admissible block of Y.

$$X|_{\hat\tau\times\hat\sigma} Y|_{\hat\sigma\times\hat\rho} = X|_{\hat\tau\times\hat\sigma}(V_\sigma S_{\sigma\rho} W_\rho^*) = (X|_{\hat\tau\times\hat\sigma} V_\sigma S_{\sigma\rho}) W_\rho^*$$

Multiplication with low-rank block

Important case: What happens if one of the factors is not subdivided?

Example: Let (σ, ρ) be an admissible block of Y.

$$X|_{\hat{\tau}\times\hat{\sigma}} Y|_{\hat{\sigma}\times\hat{\rho}} = X|_{\hat{\tau}\times\hat{\sigma}} (V_\sigma S_{\sigma\rho} W_\rho^*) = (X|_{\hat{\tau}\times\hat{\sigma}} V_\sigma S_{\sigma\rho}) W_\rho^* = A_{\tau\rho} W_\rho^*.$$

Multiplication with low-rank block

Important case: What happens if one of the factors is not subdivided?

Example: Let (σ, ρ) be an admissible block of Y.

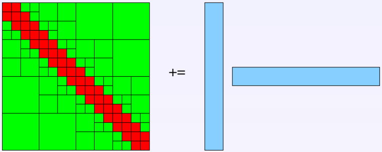
$$X|_{\hat{\tau}\times\hat{\sigma}} Y|_{\hat{\sigma}\times\hat{\rho}} = X|_{\hat{\tau}\times\hat{\sigma}} (V_\sigma S_{\sigma\rho} W_\rho^*) = (X|_{\hat{\tau}\times\hat{\sigma}} V_\sigma S_{\sigma\rho}) W_\rho^* = A_{\tau\rho} W_\rho^*.$$

Idea: $A_{\tau\rho}$ can be computed by fast matrix-vector multiplications.
We "only" need an efficient algorithm for low-rank updates

$$Z|_{\hat{\tau}\times\hat{\rho}} \leftarrow Z|_{\hat{\tau}\times\hat{\rho}} + \alpha A_{\tau\rho} W_\rho^*.$$

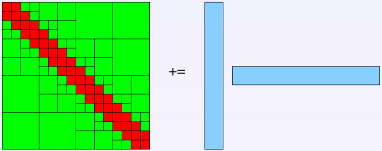
Low-rank update

Goal: Approximate $Z + AB^*$.



Low-rank update

Goal: Approximate $Z + AB^*$.

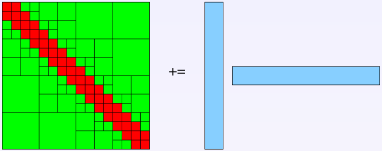


Idea: Result $Z + AB^*$ already is an $\mathcal{H}^2$-matrix:

$$(Z + AB^*)|_{\hat\tau\times\hat\sigma} = V_\tau S_{\tau\sigma} W_\sigma^* + A|_{\hat\tau\times k} B|_{\hat\sigma\times k}^*$$
$$= \underbrace{\begin{pmatrix} V_\tau & A|_{\hat\tau\times k} \end{pmatrix}}_{=:\widetilde{V}_\tau} \underbrace{\begin{pmatrix} S_{\tau\sigma} & \\ & I \end{pmatrix}}_{=:\widetilde{S}_{\tau\sigma}} \underbrace{\begin{pmatrix} W_\sigma & B|_{\hat\sigma\times k} \end{pmatrix}^*}_{=:\widetilde{W}_\sigma^*}$$

Low-rank update

Goal: Approximate $Z + AB^*$.



Idea: Result $Z + AB^*$ already is an $\mathcal{H}^2$-matrix:

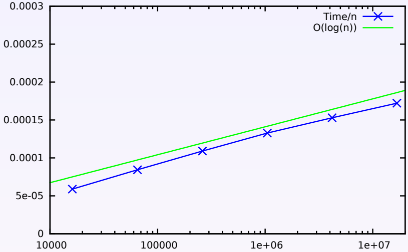
$$(Z + AB^*)|_{\hat\tau\times\hat\sigma} = V_\tau S_{\tau\sigma} W_\sigma^* + A|_{\hat\tau\times k} B|_{\hat\sigma\times k}^*$$

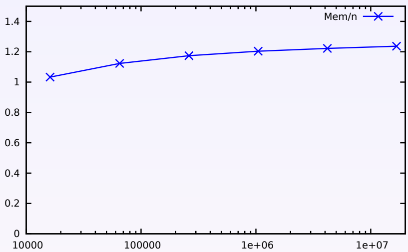
$$= \underbrace{\begin{pmatrix} V_\tau & A|_{\hat\tau\times k} \end{pmatrix}}_{=:\widetilde{V}_\tau} \underbrace{\begin{pmatrix} S_{\tau\sigma} & \\ & I \end{pmatrix}}_{=:\widetilde{S}_{\tau\sigma}} \underbrace{\begin{pmatrix} W_\sigma & B|_{\hat\sigma\times k} \end{pmatrix}^*}_{=:\widetilde{W}_\sigma^*}$$

Challenge: Each update increases storage requirements, although actual numerical rank may be low. $\rightarrow$ Recompression required.

Experiment: FEM Cholesky decomposition

Goal: Approximate Cholesky decomposition of a FEM stiffness matrix.





Results:

- Accuracy $\|I - \widetilde{L}^{-*}\widetilde{L}^{-1}A\|_2 \approx 0.1$.
- Factorization in $\sim n \log n$ operations.
- Storage requirements $\sim n$.

Matrix-Galerkin

Goal: Solve matrix equations $AX = B$, $AX + XB = C$, $AXA = B$.

Idea: Find approximation $\widetilde{X}$ in the vector space of $\mathcal{H}^2$-matrices, i.e.,

$$\widetilde{X} = \sum_{b=(\tau,\sigma)} V_\tau S_{\tau\sigma} W_\sigma^*.$$

Matrix-Galerkin

Goal: Solve matrix equations $AX = B$, $AX + XB = C$, $AXA = B$.

Idea: Find approximation $\widetilde{X}$ in the vector space of $\mathcal{H}^2$-matrices, i.e.,

$$\widetilde{X} = \sum_{b=(\tau,\sigma)} V_\tau S_{\tau\sigma} W_\sigma^*.$$

Approach: Galerkin's method in matrix space.

$$\langle A\widetilde{X}, V_\alpha S_{\alpha\beta} W_\beta^* \rangle_F = \langle B, V_\alpha S_{\alpha\beta} W_\beta^* \rangle_F \quad \text{for all } c = (\alpha, \beta),\ S_{\alpha\beta} \in \mathbb{R}^{k\times k}.$$

Matrix-Galerkin

Goal: Solve matrix equations $AX = B$, $AX + XB = C$, $AXA = B$.

Idea: Find approximation $\widetilde{X}$ in the vector space of $\mathcal{H}^2$-matrices, i.e.,

$$\widetilde{X} = \sum_{b=(\tau,\sigma)} V_\tau S_{\tau\sigma} W_\sigma^*.$$

Approach: Galerkin's method in matrix space.

$$\langle V_\alpha^* A \widetilde{X} W_\beta, S_{\alpha\beta} \rangle_F = \langle V_\alpha^* B W_\beta, S_{\alpha\beta} \rangle_F \quad \text{for all } c = (\alpha, \beta),\ S_{\alpha\beta} \in \mathbb{R}^{k\times k}.$$

Matrix-Galerkin

Goal: Solve matrix equations $AX = B$, $AX + XB = C$, $AXA = B$.

Idea: Find approximation $\widetilde{X}$ in the vector space of $\mathcal{H}^2$-matrices, i.e.,

$$\widetilde{X} = \sum_{b=(\tau,\sigma)} V_\tau S_{\tau\sigma} W_\sigma^*.$$

Approach: Galerkin's method in matrix space.

$$V_\alpha^* A \widetilde{X} W_\beta = V_\alpha^* B W_\beta \quad \text{for all } c = (\alpha, \beta)$$

Matrix-Galerkin

Goal: Solve matrix equations $AX = B$, $AX + XB = C$, $AXA = B$.

Idea: Find approximation $\widetilde{X}$ in the vector space of $\mathcal{H}^2$-matrices, i.e.,

$$\widetilde{X} = \sum_{b=(\tau,\sigma)} V_\tau S_{\tau\sigma} W_\sigma^*.$$

Approach: Galerkin's method in matrix space.

$$\sum_{b=(\tau,\sigma)} \underbrace{V_\alpha^* A V_\tau}_{=:Y_{\alpha\tau}} S_{\tau\sigma} \underbrace{W_\sigma^* W_\beta}_{=:Z_{\beta\sigma}^*} = \underbrace{V_\alpha^* B W_\beta}_{=:B_{\alpha\beta}} \qquad \text{for all } c = (\alpha, \beta)$$

Matrix-Galerkin

Goal: Solve matrix equations $AX = B$, $AX + XB = C$, $AXA = B$.

Idea: Find approximation $\widetilde{X}$ in the vector space of $\mathcal{H}^2$-matrices, i.e.,

$$\widetilde{X} = \sum_{b=(\tau,\sigma)} V_\tau S_{\tau\sigma} W_\sigma^*.$$

Approach: Galerkin's method in matrix space.

$$\sum_{b=(\tau,\sigma)} Y_{\alpha\tau} S_{\tau\sigma} Z_{\beta\sigma}^* = B_{\alpha\beta} \quad \text{for all } c = (\alpha, \beta)$$

Matrix-Galerkin

Goal: Solve matrix equations $AX = B$, $AX + XB = C$, $AXA = B$.

Idea: Find approximation $\widetilde{X}$ in the vector space of $\mathcal{H}^2$-matrices, i.e.,

$$\widetilde{X} = \sum_{b=(\tau,\sigma)} V_\tau S_{\tau\sigma} W_\sigma^*.$$

Approach: Galerkin's method in matrix space.

$$\sum_{b=(\tau,\sigma)} Y_{\alpha\tau} S_{\tau\sigma} Z_{\beta\sigma}^* = B_{\alpha\beta} \quad \text{for all } c = (\alpha, \beta)$$

Result: Linear system, coupling matrices are degrees of freedom.
Inherited from matrix equation: Symmetry, sparsity, spectral bounds.
$\rightarrow$ Use iterative solver.

Overview

Summary

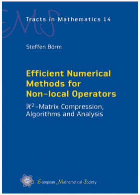
Goal: Efficient algorithms for dense matrices.

Hierarchical matrices: Blockwise low-rank approximation, storage $\sim nk \log n$.

$\mathcal{H}^2$-matrices: Blockwise approximation by cluster bases, storage $\sim nk$.

Arithmetic operations: Multiplication, inversion, factorization.

Applications: Integral equations, elliptic PDEs with discontinuous or anisotropic coefficients.



Software: `HLib` and new `H2Lib`.

Literature: S. Börm, Efficient Numerical Methods for Non-local Operators (2010)