November 5, 2010

Hierarchical Matrices:

Concept, Applications and Eigenvalues

Thomas Mach

Max Planck Institute for Dynamics of Complex Technical Systems
Computational Methods in Systems and Control Theory

HIERARCHICAL STRUCTURES

MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

November 5, 2010

Hierarchical Matrices:
Concept, Applications and Eigenvalues

Thomas Mach

Max Planck Institute for Dynamics of Complex Technical Systems
Computational Methods in Systems and Control Theory

HIERARCHICAL STRUCTURES

MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

November 5, 2010

Hierarchical Matrices:

Concept, Applications and Eigenvalues

Thomas Mach

Max Planck Institute for Dynamics of Complex Technical Systems
Computational Methods in Systems and Control Theory

HIERARCHICAL STRUCTURES

MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

November 5, 2010

Hierarchical Matrices:

Concept, Applications and Eigenvalues

Thomas Mach

Max Planck Institute for Dynamics of Complex Technical Systems
Computational Methods in Systems and Control Theory

HIERARCHICAL STRUCTURES

MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

Householder Notation

Scalar

$$\lambda \in \mathbb{R}$$

Householder Notation

Vector

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$$

Householder Notation

Matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix} \in \mathbb{R}^{n \times n}$$

Dense Matrix Format

Store n vectors (Fortran):

$$\begin{bmatrix} \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \vdots \\ a_{n1} \end{bmatrix} & \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \\ \vdots \\ a_{n2} \end{bmatrix} & \begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \\ \vdots \\ a_{n3} \end{bmatrix} & \dots & \begin{bmatrix} a_{1n} \\ a_{2n} \\ a_{3n} \\ \vdots \\ a_{nn} \end{bmatrix} \end{bmatrix}$$

- n^2 entries in the storage
- Ax costs $\mathcal{O}(n^2)$ flops
- AB costs $\mathcal{O}(n^\delta)$ flops, $\delta \geq 2.376$ usually $\delta = 3$
- A^{-1} costs $\mathcal{O}(n^3)$ flops

Dense Matrix Format

Store n ...

Landau Symbol

There is a constant c, so that Ax costs not more than cn^2 flops

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & & a_{1n} \\ a_{21} & a_{22} & a_{23} & & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & a_{n3} & & a_{nn} \end{bmatrix}$$

Flops

$1 \text{ flop} = (\alpha\beta + \gamma \rightarrow \gamma)$

- n^2 entries in the storage
- Ax costs $\mathcal{O}(n^2)$ flops
- AB costs $\mathcal{O}(n^\delta)$ flops, $\delta \geq 2.376$ usually $\delta = 3$
- A^{-1} costs $\mathcal{O}(n^3)$ flops

Dense Matrix Format

Store n vectors (Fortran):

$$\begin{bmatrix} \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \vdots \\ a_{n1} \end{bmatrix} & \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \\ \vdots \\ a_{n2} \end{bmatrix} & \begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \\ \vdots \\ a_{n3} \end{bmatrix} & \dots & \begin{bmatrix} a_{1n} \\ a_{2n} \\ a_{3n} \\ \vdots \\ a_{nn} \end{bmatrix} \end{bmatrix}$$

- n^2 entries in the storage
- Ax costs $\mathcal{O}(n^2)$ flops
- AB costs $\mathcal{O}(n^\delta)$ flops, $\delta \geq 2.376$ usually $\delta = 3$
- A^{-1} costs $\mathcal{O}(n^3)$ flops

expensive!

Sparse Matrix Format

$$\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix}$$

Sparse Matrix Format

$$\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix}$$

non-zero entries per row/column $= c \leq 5$

Sparse Matrix Format

$$\begin{bmatrix} 4 & -1 & -1 & -1 \\ -1 & -1 & -1 & 4 \\ -1 & 4 & -1 & -1 \\ -1 & -1 & 4 & -1 \\ 4 & -1 & -1 & -1 \\ -1 & -1 & -1 & 4 \\ -1 & -1 & 4 & \\ -1 & 4 & -1 & \\ 4 & -1 & -1 & \end{bmatrix}$$

non-zero entries per row/column $= c \leq 5$

Sparse Matrix Format (Matlab)

Store vector of tripel:

$$\begin{bmatrix} (1,1,\ \ 4) \\ (2,1,-1) \\ (4,1,-1) \\ (1,2,-1) \\ \vdots \end{bmatrix}$$

- $\mathcal{O}(n)$ entries in the storage (if #non-zeros/column $< c \ll n$)
- Ax costs $\mathcal{O}(n)$ flops
- AB may be much denser $\Rightarrow$ better use $A(Bx)$
- A^{-1} not possible, only solve $Ax = b$

Is there something in between?

Yes, e.g.

- Low rank matrices and Tucker tensor format
- Toeplitz/Hankel matrices
- Semiseparable matrices [Gantmacher, Krein 1937]
- Cauchy matrices
- Fast Multipole Method (FMM) [Greengard, Rokhlin '87]
- Mosaic-skeleton matrices [Tyrtyshnikov '96]
- ...

Is there something in between?

Yes, e.g.

- Low rank matrices and Tucker tensor format
- Toeplitz/Hankel matrices
- Semiseparable matrices [Gantmacher, Krein 1937]
- Cauchy matrices
- Fast Multipole Method (FMM) [Greengard, Rokhlin '87]
- Mosaic-skeleton matrices [Tyrtyshnikov '96]
- ...

Is there something in between?

Yes, e.g.

- Low rank matrices and Tucker tensor format
- Toeplitz/Hankel matrices
- Semiseparable matrices [Gantmacher, Krein 1937]
- Cauchy matrices
- Fast Multipole Method (FMM) [Greengard, Rokhlin '87]
- Mosaic-skeleton matrices [Tyrtyshnikov '96]
- Hierarchical ($\mathcal{H}$-) Matrices [Hackbusch '98]

Integral Equation

Fredholm equation of first kind:

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

Integral Equation

unknown

Fredholm equation of first kind:

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

Integral Equation

Fredholm equation of first kind:

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

- electrostatic problem:
  - $g(x,y) = \frac{1}{4\pi\epsilon\|x-y\|}$
  - $u(y)$ charge density
  - $f(x)$ electrostatic potential
- inverse of elliptic differential operators
- population dynamics [Koch, Hackbusch, Sundmacher]

$\mathcal{H}$-Matrices [Hackbusch '98]

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

Ritz-Galerkin method:

$$u(y) = \sum_i u_i\psi_i(y)$$

$$\Rightarrow \sum_i \underbrace{\int_\Omega \int_\Omega g(x,y)\psi_i(y)dy\psi_j(x)dx}_{:=M_{ji}} u_i = \underbrace{\int_\Omega f(x)\psi_j(x)dx}_{=f_j}$$

$\mathcal{H}$-Matrices [Hackbusch '98]

discretization error $\epsilon \sim \frac{1}{n^\kappa}$, $0 < \kappa < 1$

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

Ritz-Galerkin method:

$u(y) = \sum_i u_i \psi_i(y)$

$$\Rightarrow \sum_i \underbrace{\int_\Omega \int_\Omega g(x,y)\psi_i(y)dy\psi_j(x)dx}_{:=M_{ji}} u_i = \underbrace{\int_\Omega f(x)\psi_j(x)dx}_{=f_j}$$

$\mathcal{H}$-Matrices [Hackbusch '98]

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

Ritz-Galerkin method:

$$u(y) = \sum_i u_i \psi_i(y)$$

$$\Rightarrow \sum_i \underbrace{\int_\Omega \int_\Omega g(x,y)\psi_i(y)dy\psi_j(x)dx}_{:=M_{ji}} u_i = \underbrace{\int_\Omega f(x)\psi_j(x)dx}_{=f_j}$$

$$Mu = f$$

$\mathcal{H}$-Matrices [Hackbusch '98]

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

Ritz-Galerkin method:

$$u(y) = \sum_i u_i \psi_i(y)$$

$$\Rightarrow \sum_i \underbrace{\int_\Omega \int_\Omega g(x,y)\psi_i(y)dy\psi_j(x)dx}_{:=M_{ji}} u_i = \underbrace{\int_\Omega f(x)\psi_j(x)dx}_{=f_j}$$

$x \in \Omega_s, y \in \Omega_t$: $g(x,y) \approx \sum_{p=1}^k g_p^{x,s}(x) g_p^{y,t}(y)$

$\mathcal{H}$-Matrices [Hackbusch '98]

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

Ritz-Galerkin method:

$$u(y) = \sum_i u_i \psi_i(y)$$

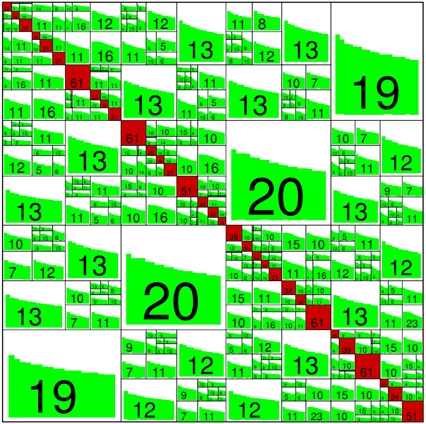
$$\Rightarrow \sum_i \underbrace{\int_\Omega \int_\Omega g(x,y)\psi_i(y)dy\psi_j(x)dx}_{:=M_{ji}} u_i = \underbrace{\int_\Omega f(x)\psi_j(x)dx}_{=f_j}$$

$x \in \Omega_s, y \in \Omega_t$: $g(x,y) \approx \sum_{p=1}^k g_p^{x,s}(x) g_p^{y,t}(y)$

$$M_{ji} = \int_\Omega \int_\Omega g(x,y)\psi_j(x)\psi_i(y)dydx$$

$$\approx \sum_{p=1}^k \int_\Omega g_p^x(x)\psi_j(x)dx \int_\Omega g_p^y(y)\psi_i(y)dy = A_{j\cdot} B_{i\cdot}^T$$

$\mathcal{H}$-Matrices [Hackbusch '98]

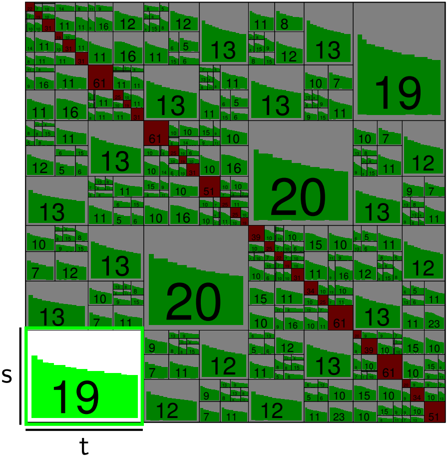


$$x \in \Omega_s, y \in \Omega_t : \quad g(x,y) \approx \sum_{p=1}^{k} g_p^{x,s}(x) g_p^{y,t}(y)$$

$$M_{ji} = \int_\Omega \int_\Omega g(x,y)\psi_j(x)\psi_i(y)dydx$$

$$\approx \sum_{p=1}^{k} \int_\Omega g_p^x(x)\psi_j(x)dx \int_\Omega g_p^y(y)\psi_i(y)dy = A_{j.} B_{i.}^T$$

$\mathcal{H}$-Matrices [Hackbusch '98]

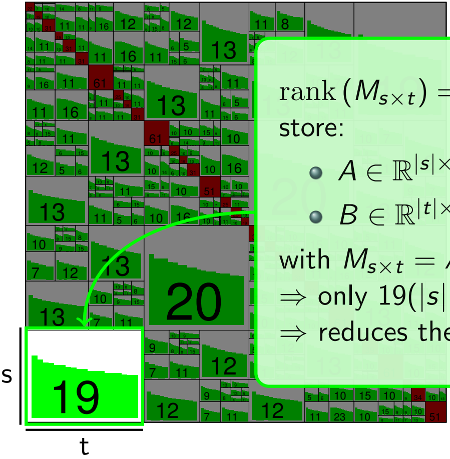


$$x \in \Omega_s, y \in \Omega_t : \quad g(x,y) \approx \sum_{p=1}^{k} g_p^{x,s}(x) g_p^{y,t}(y)$$

$$M_{ji} = \int_\Omega \int_\Omega g(x,y)\psi_j(x)\psi_i(y)\,dy\,dx$$

$$\approx \sum_{p=1}^{k} \int_\Omega g_p^x(x)\psi_j(x)\,dx \int_\Omega g_p^y(y)\psi_i(y)\,dy = A_{j.} B_{i.}^T$$

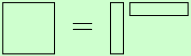
$\mathcal{H}$-Matrices [Hackbusch '98]



$\operatorname{rank}(M_{s\times t}) = 19$

store:

- $A \in \mathbb{R}^{|s|\times 19}$,
- $B \in \mathbb{R}^{|t|\times 19}$



with $M_{s\times t} = AB^T$

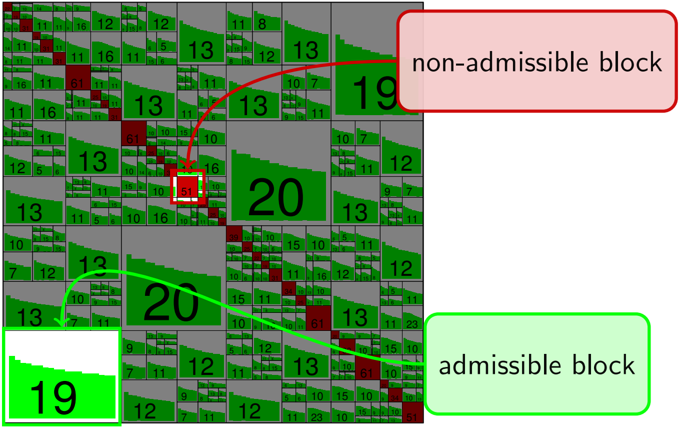
$\Rightarrow$ only $19(|s| + |t|)$ instead of $|s|\,|t|$ storage

$\Rightarrow$ reduces the required storage to 15%

$\mathcal{H}$-Matrices [Hackbusch '98]

if $x \approx y$ then $\frac{1}{\|x-y\|}$ is large
small matrices on the diagonal have no low rank approximation
$\Rightarrow$ use dense matrix format

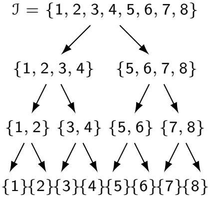
$\mathcal{H}$-Matrices [Hackbusch '98]



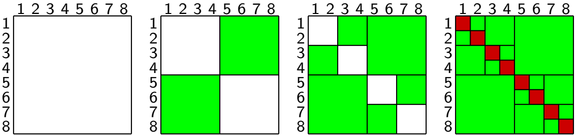
$\mathcal{H}$-Matrices [Hackbusch '98]

rank-k-matrix: $M_{s \times t} = AB^T,\ A \in \mathbb{R}^{n \times k}, B \in \mathbb{R}^{m \times k}\ (k \ll n, m)$

hierarchical tree $T_{\mathcal{I}}$



block $\mathcal{H}$-tree $T_{\mathcal{I} \times \mathcal{I}}$

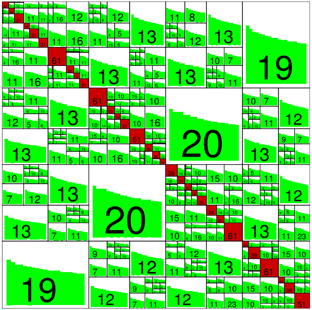


dense matrices, **rank-k-matrices**

$\mathcal{H}$-Matrices [Hackbusch '98]

Hierarchical matrices

$$\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k) = \left\{ M \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}} \middle| \operatorname{rank}(M_{s\times t}) \leq k \ \forall s \times t \text{ admissible} \right\}$$



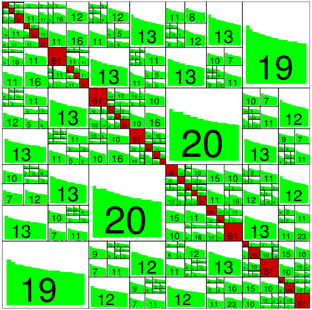
- adaptive rank $k(\varepsilon)$
- storage $N_{St,\mathcal{H}}(T, k) = \mathcal{O}(n \log n \ k(\varepsilon))$
- complexity of approximate arithmetic

$M_{\mathcal{H}} v$	$\mathcal{O}(n \log n \ k(\varepsilon))$
$+_{\mathcal{H}}, -_{\mathcal{H}}$	$\mathcal{O}(n \log n \ k(\varepsilon)^2)$
$*_{\mathcal{H}}, \mathcal{H}LU(\cdot), (\cdot)^{-1}_{\mathcal{H}}$	$\mathcal{O}(n (\log n)^2 \ k(\varepsilon)^2)$

$\mathcal{H}$-Matrices [Hackbusch '98]

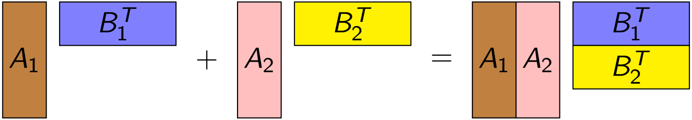
Hierarchical matrices

$$\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k) = \left\{ M \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}} \middle| \operatorname{rank}(M_{s\times t}) \leq k \ \forall s \times t \text{ admissible} \right\}$$



- adaptive rank $k(\varepsilon)$
- storage $N_{St,\mathcal{H}}(T, k) = \mathcal{O}(n \log n \; k(\varepsilon))$
- complexity of approximate arithmetic

$M_{\mathcal{H}} v$	$\mathcal{O}(n \log n \; k(\varepsilon))$
$+_{\mathcal{H}}, -_{\mathcal{H}}$	$\mathcal{O}(n \log n \; k(\varepsilon)^2)$
$*_{\mathcal{H}}, \mathcal{H}LU(\cdot), (\cdot)^{-1}_{\mathcal{H}}$	$\mathcal{O}(n (\log n)^2 \; k(\varepsilon)^2)$



Special Case: $\mathcal{H}_\ell$-Matrices

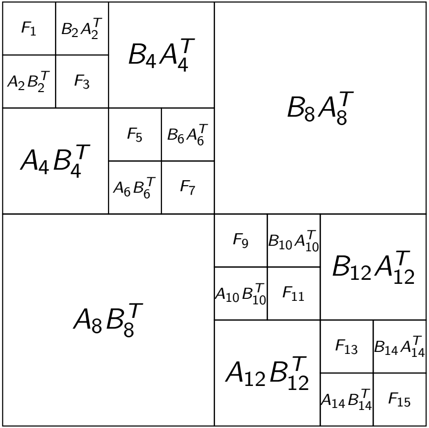


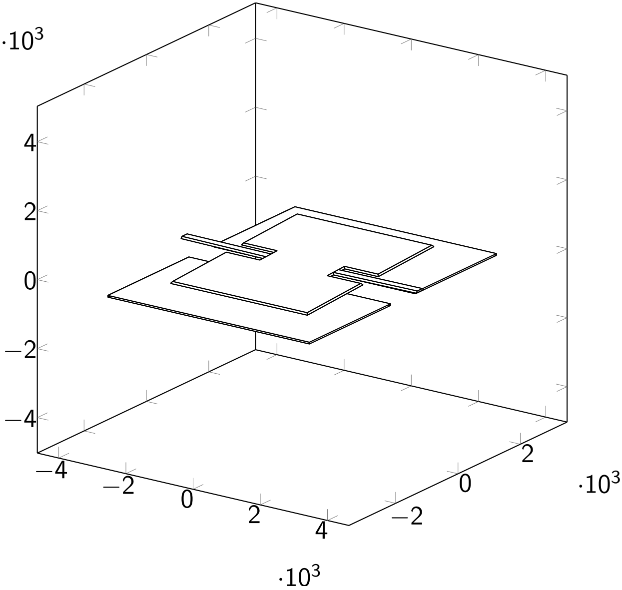
Figure: Structure of an $\mathcal{H}_3(k)$-matrix

Applications

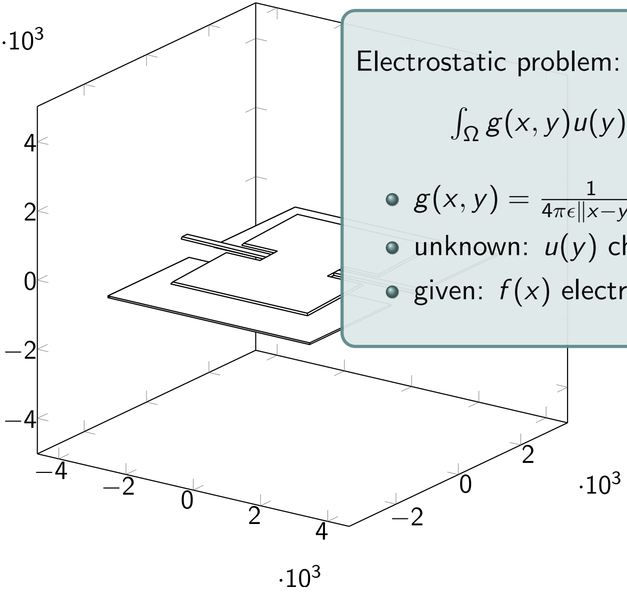
Applications

- Finite-Element-Method (FEM)
  - Inverse of FEM matrices
- Integral equations
- Solving matrix equations for model order reduction
  - $AX + XA^T + BB^T = 0$, $A \in \mathcal{H}, B \in \mathbb{R}^{n \times k}$
  - Sign function iteration
  - [Baur '07, Grasedyck '03, et al.]
- Boundary-Element-Method (BEM)

BEM for Electrostatic Computations



BEM for Electrostatic Computations

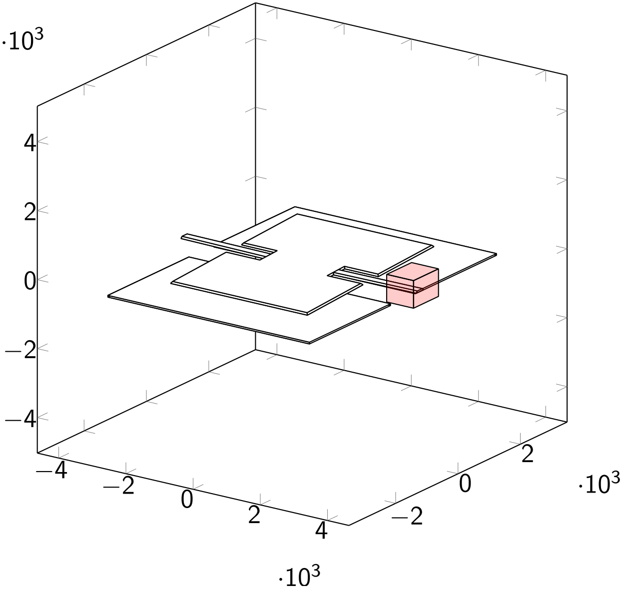


Electrostatic problem:

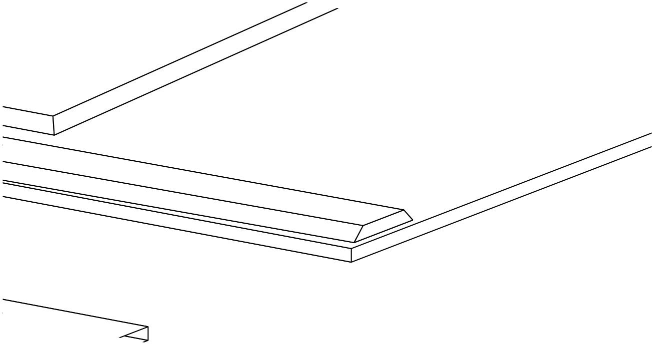
$$\int_\Omega g(x,y)u(y)dy = f(x)$$

- $g(x,y) = \frac{1}{4\pi\epsilon\|x-y\|}$
- unknown: $u(y)$ charge density
- given: $f(x)$ electrostatic potential

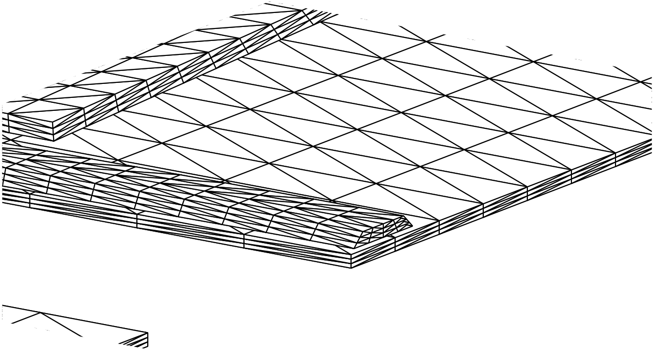
BEM for Electrostatic Computations



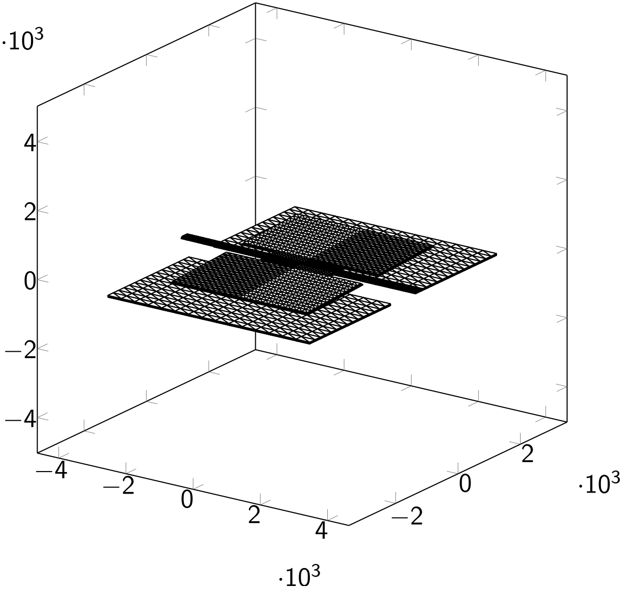
BEM for Electrostatic Computations



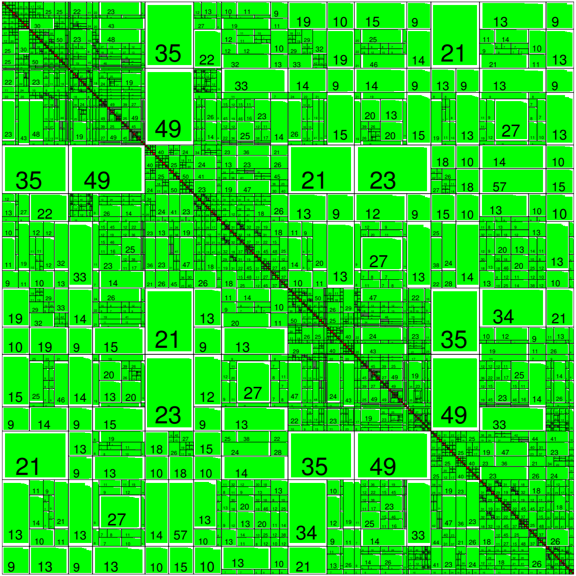
BEM for Electrostatic Computations



BEM for Electrostatic Computations

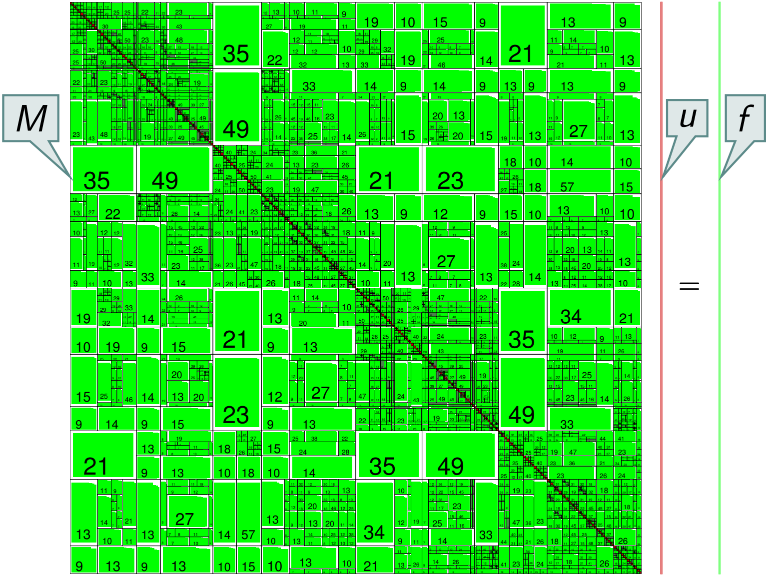


BEM for Electrostatic Computations

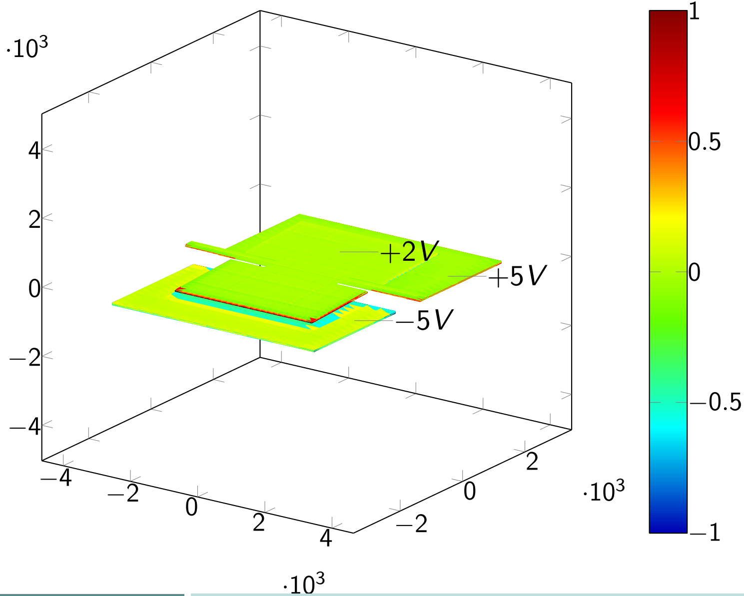


$\in \mathbb{R}^{7416 \times 7416}$

BEM for Electrostatic Computations



BEM for Electrostatic Computations



Eigenvalues

Definition

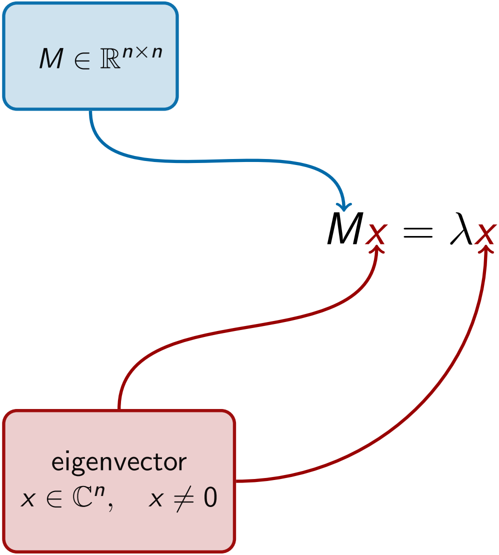
$$Mx = \lambda x$$

Definition

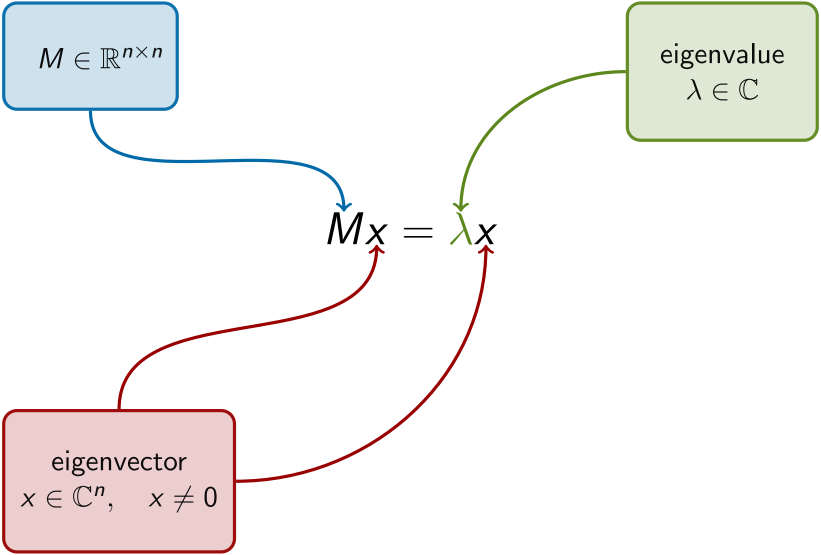
$M \in \mathbb{R}^{n \times n}$

$$Mx = \lambda x$$

Definition


$M \in \mathbb{R}^{n \times n}$
$Mx = \lambda x$
eigenvector
$x \in \mathbb{C}^n, \quad x \neq 0$

Definition


$M \in \mathbb{R}^{n \times n}$
eigenvalue
$\lambda \in \mathbb{C}$
$Mx = \lambda x$
eigenvector
$x \in \mathbb{C}^n, \quad x \neq 0$

Definition

$M = M^T \in \mathbb{R}^{n \times n}$

eigenvalue

$\lambda \in \mathbb{R}$

$$Mx = \lambda x$$

eigenvector

$x \in \mathbb{R}^n, \quad x \neq 0$

Application

Eigenvalue problems occur in many different applications:

- vibration analysis,
- molecular and quantum dynamics (e.g. Schrödinger or Kohn-Sham equations),
- finite state of a Markov chain,
- numerical mathematics (e.g. MOR, e^M, convergence theory),
- ...

Definition

$M = M^T \in \mathcal{H}(T, k)$

eigenvalue
$\lambda \in \mathbb{R}$

$$Mx = \lambda x$$

eigenvector
$x \in \mathbb{R}^n, \quad x \neq 0$

Preconditioned Inverse Iteration

[Knyazev, Neymeyr, et al.]

Definition

The function

$$\mu(x) = \mu(x, M) = \frac{x^T M x}{x^T x}$$

is called the *Rayleigh quotient*.

Preconditioned Inverse Iteration

[Knyazev, Neymeyr, et al.]

Definition

The function

$$\mu(x) = \mu(x, M) = \frac{x^T M x}{x^T x}$$

is called the *Rayleigh quotient*.

Minimize the Rayleigh quotient by a gradient method:

$$x_{i+1} := x_i - \alpha \nabla \mu(x_i), \quad \nabla \mu(x) = \frac{2}{x^T x}(Mx - x\mu(x)),$$

Preconditioned Inverse Iteration

[Knyazev, Neymeyr, et al.]

Residual $r(x) = Mx - x\mu(x)$.

Minimize the Rayleigh quotient by a gradient method:

$$x_{i+1} := x_i - \alpha \nabla \mu(x_i), \quad \nabla \mu(x) = \frac{2}{x^T x}(Mx - x\mu(x)),$$

Preconditioned Inverse Iteration

[Knyazev, Neymeyr, et al.]

Definition

The function

$$\mu(x) = \mu(x, M) = \frac{x^T M x}{x^T x}$$

is called the *Rayleigh quotient*.

Minimize the Rayleigh quotient by a gradient method:

$$x_{i+1} := x_i - \alpha \nabla \mu(x_i), \quad \nabla \mu(x) = \frac{2}{x^T x}(Mx - x\mu(x)),$$

$+$ preconditioning $\Rightarrow$ update equation:

$$x_{i+1} := x_i - T^{-1}\,(Mx_i - x_i \mu(x_i)).$$

Preconditioned Inverse Iteration

[Knyazev, Neymeyr '09]

$$x_{i+1} := x_i - T^{-1}\left(Mx_i - x_i\mu(x_i)\right)$$

If

- M symmetric positive definite and
- T^{-1} approximates the inverse of M, so that

$$\left\|\mathcal{I} - T^{-1}M\right\|_M \leq c < 1,$$

Preconditioned Inverse Iteration

[Knyazev, Neymeyr '09]

$$x_{i+1} := x_i - T^{-1}\left(Mx_i - x_i\mu(x_i)\right)$$

If

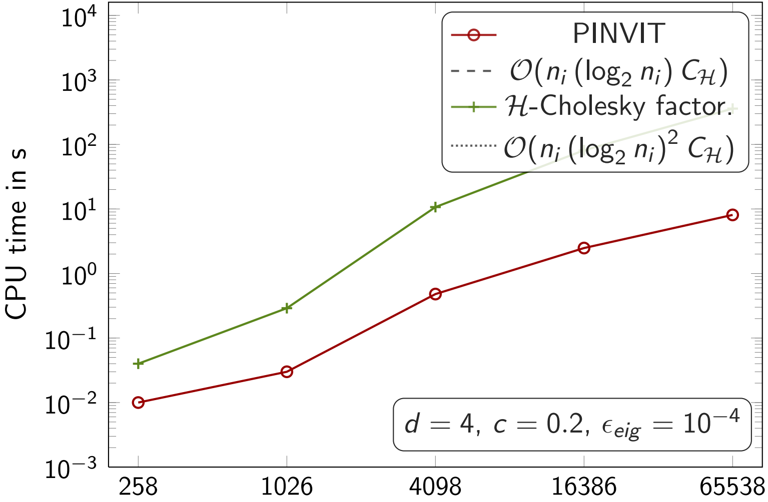
- M symmetric positive definite and
- T^{-1} approximates the inverse of M, so that

$$\left\|\mathcal{I} - T^{-1}M\right\|_M \leq c < 1,$$

then Preconditioned INVerse ITeration (PINVIT) converges.

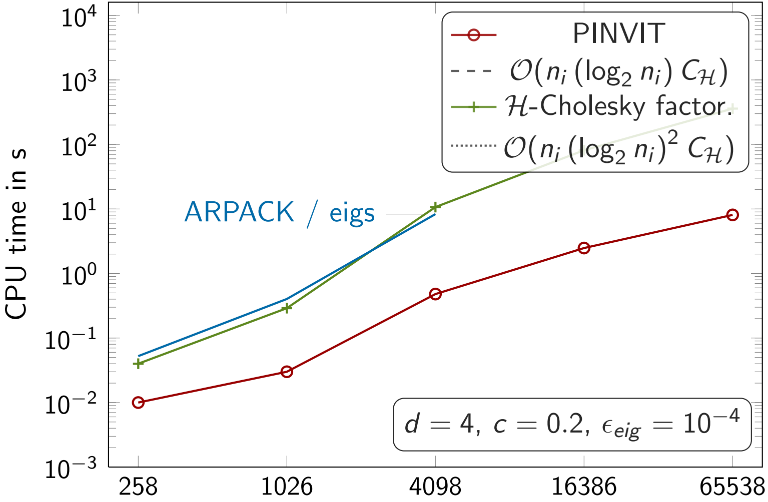
BEM, smallest eigenvalues

dense matrix, but approximable by an $\mathcal{H}$-matrix



BEM, smallest eigenvalues

dense matrix, but approximable by an $\mathcal{H}$-matrix



Concluding Remarks

- $\mathcal{H}$-arithmetic is cheap and applicable too many problems.
- Computing eigenvalues of $\mathcal{H}$-matrices in linear-polylogarithmic complexity is possible.
- See poster: *"Slicing the Spectrum of Symmetric $\mathcal{H}_\ell$-Matrices"*.

Concluding Remarks

- $\mathcal{H}$-arithmetic is cheap and applicable too many problems.
- Computing eigenvalues of $\mathcal{H}$-matrices in linear-polylogarithmic complexity is possible.
- See poster: *"Slicing the Spectrum of Symmetric $\mathcal{H}_\ell$-Matrices"*.

Concluding Remarks

- $\mathcal{H}$-arithmetic is cheap and applicable too many problems.
- Computing eigenvalues of $\mathcal{H}$-matrices in linear-polylogarithmic complexity is possible.
- See poster: *"Slicing the Spectrum of Symmetric $\mathcal{H}_\ell$-Matrices"*.

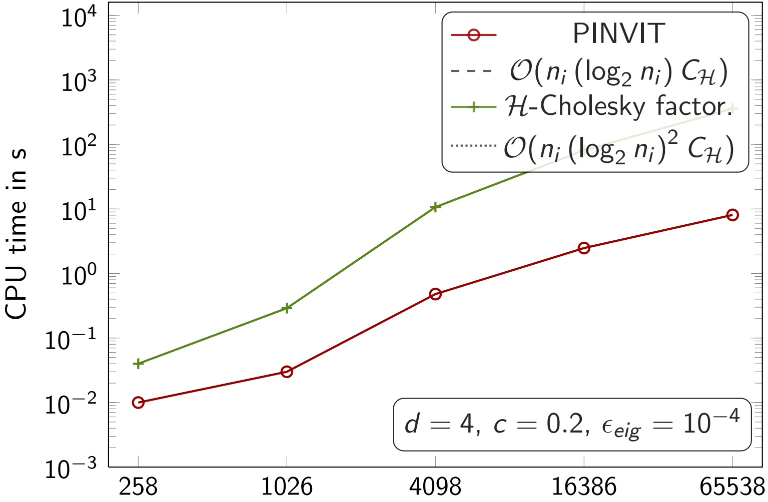
Concluding Remarks

- $\mathcal{H}$-arithmetic is cheap and applicable too many problems.
- Computing eigenvalues of $\mathcal{H}$-matrices in linear-polylogarithmic complexity is possible.
- See poster: *"Slicing the Spectrum of Symmetric $\mathcal{H}_\ell$-Matrices"*.

Thank you for your attention.

BEM, smallest eigenvalues

dense matrix, but approximable by an $\mathcal{H}$-matrix



BEM, smallest eigenvalues

dense matrix, but approximable by an $\mathcal{H}$-matrix