Hierarchical Matrices

Wolfgang Hackbusch

Max-Planck-Institut für *Mathematik in den Naturwissenschaften*

Inselstr. 22-26, D-04103 Leipzig, Germany

wh@mis.mpg.de

http://www.mis.mpg.de/scicomp/hackbusch_e.html

Fribourg, April 25, 2008

1 Introduction

- Treatment of large-scale linear systems of equations is a common need in modern computations

- The use of matrices leads in general to difficulties

Large-scale systems: size $n = 10^5, 10^6$ or larger, depending on the storage size.

Fully populated matrices have n^2 entries; storage of $O(n^2)$ is usually not available.

Standard remedy: Restrict computations to sparse matrices ($O(n)$ entries) and use only matrix-vector multiplications (cost in computer time $O(n)$).

Goal of the hierarchical matrix technique: all matrix operations, in particular for full matrices.

Typical fields of application:

■ Boundary Element Method (BEM):
Formulation of homogeneous elliptic boundary value problems by integral equation formulations
$\Rightarrow$ System matrices are fully populated

■ Finite Element Method (FEM):
Elliptic boundary value problems lead to sparse matrices A, but for instance A^{-1} is full. LU-factors are partially filled.
Sometimes Schur complements

$$A_{11} - A_{12}\, A_{22}^{-1}\, A_{21}$$

are needed, which are also full.

■ Further Applications

The costs in standard matrix approaches are:

- storage, $A * x$, $A + B$: $O(n^2)$
- $A * B$, A^{-1}, LU decomposition: $O(n^3)$.

The technique of hierarchical matrices tries to perform all matrix operations with a computational cost of

$$O(n \log^* n).$$

!! The results are only approximate (only $A * x$ is exact).

Already existing discretisation error $\varepsilon = O(n^{-\alpha}) \Rightarrow \log(1/\varepsilon) = O(\log n)$.

Preview: How do $\mathcal{H}$-matrices look like?

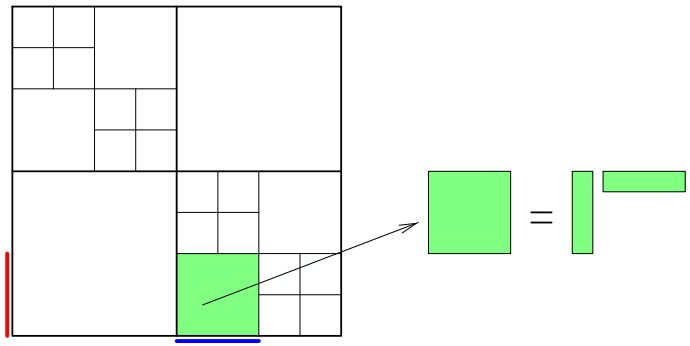
- Decompose the matrix into suitable subblocks.

- Approximate the matrix in each subblock by a rank-k-matrix*

$$block = \sum_{i=1}^{k} a_i b_i^\top$$

(for suitably small local rank k).

Illustration:



*k is upper bound. The true rank may be smaller.

Two Questions:

- How large is the representation error?
  More precisely: How does the local rank k correspond to the error of the matrix representation?

- How can the (approximate) matrix operations performed such that

$$cost = O(n * \log^* n) \text{ ?}$$

Side Remark: About Rk-Matrices

Let the Rk-matrix $\sum_{i=1}^{k} a_i b_i^\top$ be of size $n \times m$.

REMARK: (a) The amount of storage is $(n+m)\,k$ (a_i and b_i to be stored).
(b) The amount of work for the matrix-vector multiplication $A * c$ ($c \in \mathbb{R}^m$) are

$$2k\,(m+n) - k - n \quad \text{operations.}$$

Sums of Rk-Matrices, Truncation to Rank k
In general, the sum of Rk-matrices is of rank $2k$. Apply truncation to rank k by means of the singular-value decomposition:

$$A = U * D * V^\top, \quad (U, V \text{ unitary, } D \text{ diagonal with } d_1 \geq ... \geq d_{2k} \geq 0).$$

Truncation to rank k:

$$A' = U * D' * V^\top \quad \text{with } D' := \text{diag}\{d_1, ..., d_k, 0, ...\}$$

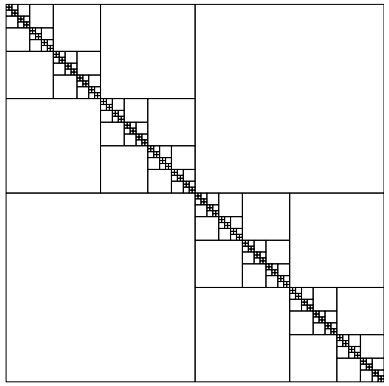
is of rank k and has the smallest Frobenius norm $\|A - A'\|_F$.

NOTATION: $A \oplus_{R1} B$ or, in the general case of rank-k-matrices, $A \oplus_{Rk} B$.

REMARK: The *R1*-addition $\oplus_{R1}$ of two $n \times m$-matrices costs $9(n+m) + O(1)$ operations.

2 Example for Demonstration

Let $n = 2^p$, $p = 0, 1, \ldots$



The construction of the $\mathcal{H}$-matrix format is recursive:
For $n = 1$, A is a rank-1-matrix. Otherwise the format of an $n \times n$ matrix of level p ($n = 2^p$) is

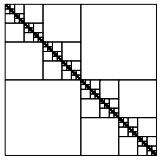
$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

with

- A_{ij} are blocks of the size $\frac{n}{2} \times \frac{n}{2}$,
- A_{ii} ($i = 1, 2$) are $\mathcal{H}$-matrices (of level $p - 1$),
- A_{12}, A_{21} are rank-1-matrix (abbreviation: $R1$).

2.1 Complexity of the $\mathcal{H}$-Matrix Arithmetic

2.1.1 Storage

Dimension: $n = 2^p$, p : hierarchy level: 

The construction yields

$$N_{storage}(p) = 2n + 2N_{storage}(p-1) \quad \text{for } p > 1.$$

Proof: The 2 off-diagonal blocks contain 4 vectors of size $n/2$.

Together with the induction start

$$N_{storage}(0) = 1 \text{ (case of } n = 1 = 2^0),$$

this leads to

LEMMA: The storage requirement for an $n \times n$ $\mathcal{H}$-matrix with $n = 2^p$ is

$$N_{storage}(p) = (2p+1)n = n(1 + 2\log_2 n).$$

2.1.2 Addition

LEMMA: The R1-addition of two $n \times n$ $\mathcal{H}$-matrices or an $\mathcal{H}$-matrix and an R1-matrix requires $18n \log_2 n + O(n)$ operations.

2.1.3 Matrix-Vector Multiplication

$A : n \times n$ $\mathcal{H}$-matrix, $x : n$-vector, $(A, x) \longmapsto A * x$.

LEMMA: The matrix-vector multiplication of an $n \times n$ $\mathcal{H}$-matrix by a vector requires

$$4n \log_2 n - n + 2 \text{ operations.}$$

2.1.4 Matrix-Matrix Multiplication

Three types of products are to be distinguished:

. 1) $R * R$ (R1-matrix times R1-matrix)
. 2) $R * H$ ($\mathcal{H}$-matrix times R1-matrix) or $H * R$
. 3) $H * H$ ($\mathcal{H}$-matrix times $\mathcal{H}$-matrix)

Type 1: $\left(ab^\top\right)\left(cd^\top\right) = (\alpha * a)\, d^\top$, with $\alpha = b^\top c$.

LEMMA: $N_{R1*R1}(p) = 3n - 1$ operations.

Type 2: $H * \left(ab^\top\right) = (H * a)\, b^\top$ requires only a matrix-vector multiplication.

LEMMA: $N_{H*R1}(p) = 4n \log_2 n - n + 2$ operations. Same for $R * H$.

Type 3: $H * H$ is computed recursively by

$$\begin{aligned} H * H &= \begin{bmatrix} H & R \\ R & H \end{bmatrix} * \begin{bmatrix} H & R \\ R & H \end{bmatrix} \\ &= \begin{bmatrix} \underline{H * H} + R * R & H * R + R * H \\ R * H + H * R & \underline{H * H} + R * R \end{bmatrix}. \end{aligned}$$

This leads to the recursion

$$\begin{aligned} N_{H*H}(p) &= 2N_{H\cdot H}(p-1) + 2N_{R\cdot R}(p-1) + 2N_{H\cdot R}(p-1) \\ &\quad + 2N_{R\cdot H}(p-1) + 2N_{H+R}(p-1) + 2N_{R+R}(p-1) \end{aligned}$$

with the starting value $N_{H*H}(0) = 1$.

LEMMA: The multiplication of two $\mathcal{H}$-matrices requires

$$13n \log_2^2 n + 65n \log_2 n - 51n + 52 \text{ operations.}$$

2.1.5 Matrix Inversion

Approximation of the inverse A^{-1} by an $\mathcal{H}$-matrix $Inv_{R1}(A)$.

Recursion with respect to p ($n = 2^p$):

For $p = 0$, $Inv_{R1}(A) := A^{-1}$.
Having defined Inv_{R1} on level $p-1$, the (exact) inverse of A is

$$\begin{bmatrix} A_{11}^{-1} + A_{11}^{-1} A_{12} S^{-1} A_{21} A_{11}^{-1} & -A_{11}^{-1} A_{12} S^{-1} \\ -S^{-1} A_{21} A_{11}^{-1} & S^{-1} \end{bmatrix}$$

with the Schur complement $S = A_{22} - A_{21} A_{11}^{-1} A_{12}$.

Recursion for the cost $N_{inv}(p)$:

$$\begin{aligned} N_{inv}(p) &= 2N_{inv}(p-1) + 4N_{H*R1}(p-1) \\ &\quad + 2N_{H+R1}(p-1) + 2N_{R1*R1}(p-1). \end{aligned}$$

LEMMA: The approximate inversion of an $\mathcal{H}$-matrix requires

$$13n \log_2^2 n + 47n \log_2 n - 109n + 110 \text{ operations.}$$

2.1.6 LU-Decomposition

A is to be represented by

$$A \approx LU,$$

where L is a lower triangular matrix and U a upper triangular matrix of the $\mathcal{H}$-format.

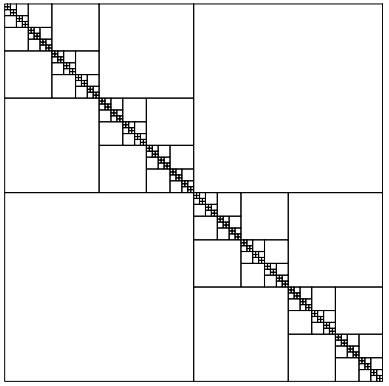
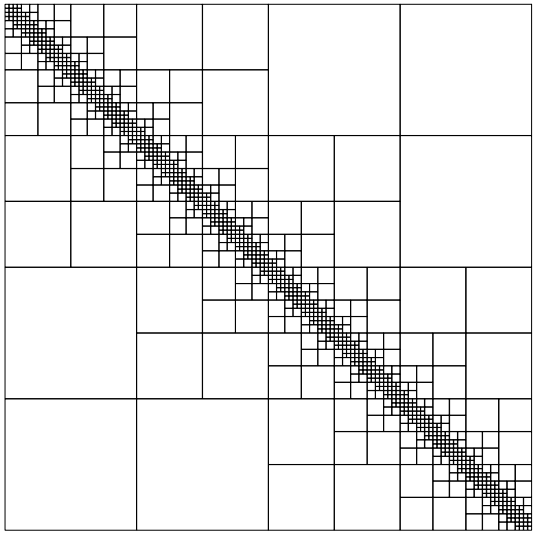
LEMMA: The approximate LU-decomposition costs

$$N_{LU}(p) = \frac{11}{2} n \log_2^2 n + 25 n \log_2 n - 28n + 28.$$

operations.

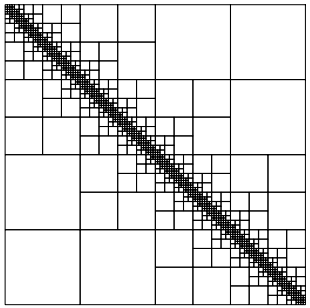
2.1.7 Concluding Remarks to the Introductory Case

At least, the rank 1 is to be replaced by a larger rank k.

Moreover, in general, the simple format  is to be replaced by a more refined format like 

3 General Construction of Hierarchical Matrices

Partition of the Matrix



How to partition the matrix in subblocks?

I: index set of matrix rows, J: index set of matrix columns.
Block: $b = \tau \times \sigma$ with $\tau \subset I$, $\sigma \subset J$.

Cluster Tree:
The cluster tree $T(I)$ contains a collection of subsets $\tau \subset I$ (similarly: $T(J)$).

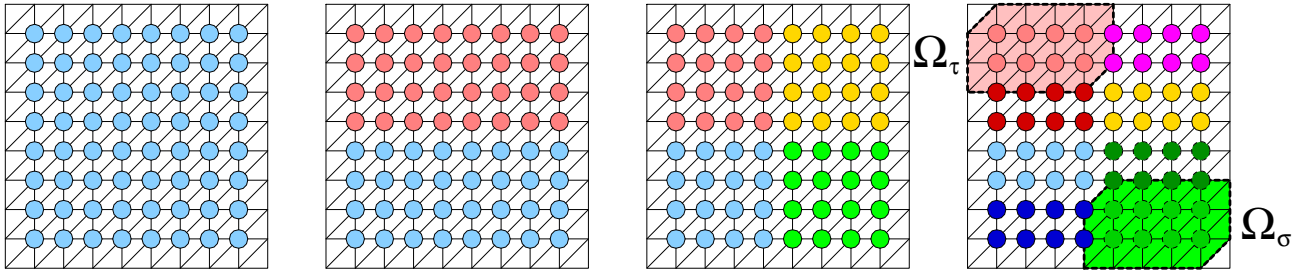
Block Cluster Tree $T(I \times J)$:
Collection of (small and large) blocks $b = \tau \times \sigma$ with $\tau \in T(I)$, $\sigma \in T(J)$.
Criterion for selection: b as large as possible and admissible, i.e.,

$$\min\{\mathrm{diam}(\tau), \mathrm{diam}(\sigma)\} \geq \eta\, \mathrm{dist}(\tau, \sigma).$$

Cluster Tree

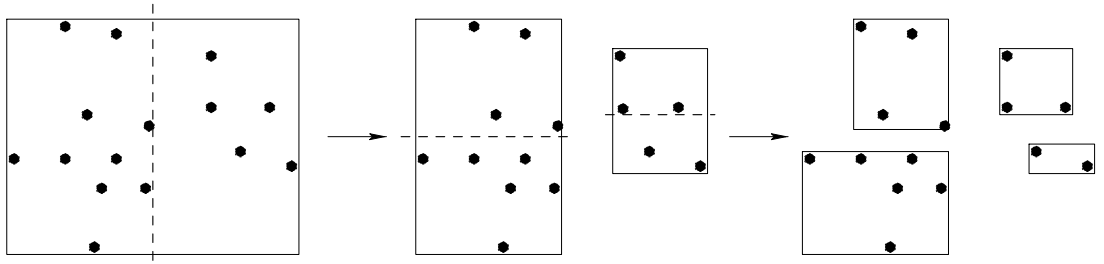
Let I (J) be the index set containing the row (column) indices i (j) of the matrix $A = \left(A_{ij}\right)$. Partition I recursively into (e.g.) two subsets. This process ends if the subsets of I have a sufficiently small cardinality. (Similarly for J).

The resulting tree $T(I)$ is called the cluster tree.



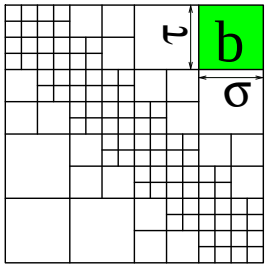
REMARK: For usual discretisations, an index $i \in I$ is associated to an nodal point $x_i \in \mathbb{R}^d$ or the support $\text{supp}(\phi_i) \subset \mathbb{R}^d$ of a basis function ϕ_i.

The practical performance uses bounding boxes:

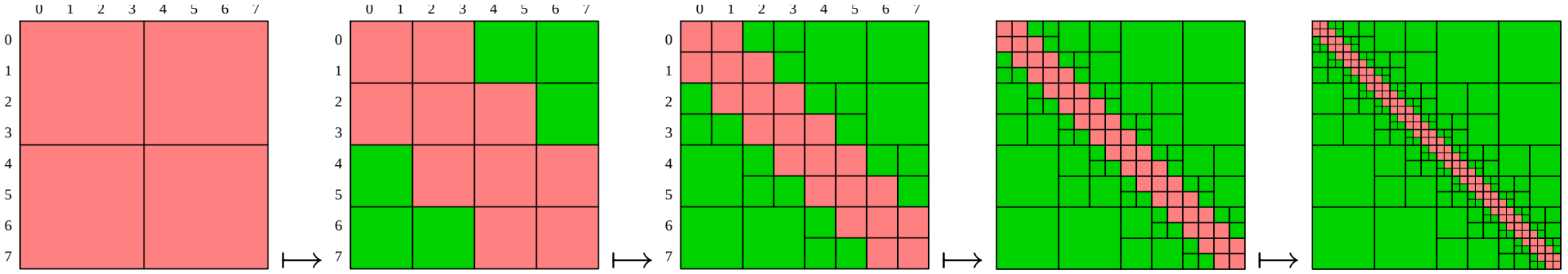


Block-Cluster Tree

NOTATION: $T(I \times J)$ is the block-cluster tree. Elements: blocks $b = \tau \times \sigma$.



Let $\tau \times \sigma \in T(I \times J)$ be a block ($\tau \in T(I)$, $\sigma \in T(J)$).
$\tau', \tau'' \in T(I)$ sons of τ, i.e., $\tau = \tau' \cup \tau''$. Similarly, $\sigma', \sigma'' \in T(J)$ sons of $\sigma \in T(J)$. Then the four sons of $\tau \times \sigma \in T(I \times J)$ are $\tau' \times \sigma'$, $\tau' \times \sigma''$, $\tau'' \times \sigma'$, $\tau'' \times \sigma'' \in T(I \times J)$. If either τ or σ is a leaf, $\tau \times \sigma$ is not further partitioned.

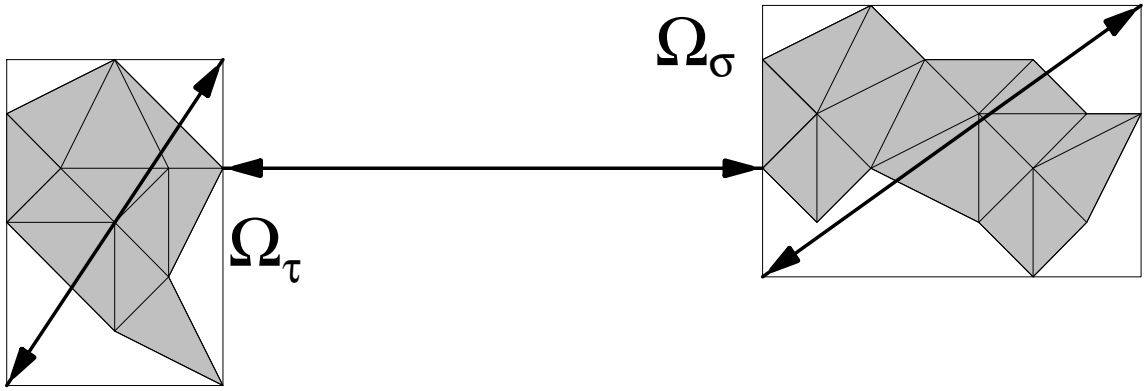


green blocks: admissible, red: non-admissible

DEFINITION (admissible block) Fix some $\eta > 0$. A block $\tau \times \sigma \in T(I \times J)$ is called admissible if

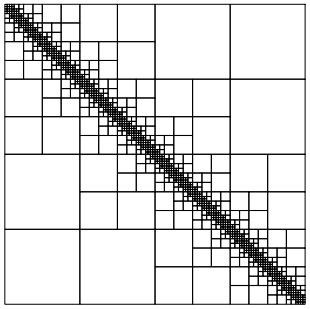
$$\min\{\text{diam}(\Omega_\tau), \text{diam}(\Omega_\sigma)\} \geq \eta \, \text{dist}(\Omega_\tau, \Omega_\sigma)$$

or $\tau \times \sigma$ is a leaf. $\tau \times \sigma \in T(I \times J)$ is called maximally admissible if the father of $\tau \times \sigma$ is non-admissible.



DEFINITION (Partition P): $P \subset T(I \times J)$ is defined by:

1) different $b \in P$ are disjoint,
2) their union $\bigcup_{b \in P} p = I \times J$ is complete,
3) they are maximally admissible.



4 Application to BEM

Example: $(\mathcal{A}u)(x) := \int_0^1 \log|x-y|\, u(y)dy$ for $x \in [0,1]$.

Discretisation: collocation with piecewise constant elements in

$$[x_{i-1}, x_i],\ x_i = ih, \qquad i = 1, \ldots, n,\ h = 1/n,$$

Midpoints $x_{i-1/2} = (i-1/2)h$ are the collocation points:

$$A = (a_{ij})_{i,j=1,\ldots,n} \qquad \text{with } a_{ij} = \int_{x_{j-1}}^{x_j} \log\left|x_{i-1/2} - y\right| dy.$$

Replace the kernel function $\kappa(x,y) = \log(x-y)$ in a certain range of x, y by an approximation $\tilde{\kappa}(x,y)$ of separable form

$$\tilde{\kappa}(x,y) = \sum\nolimits_{\iota \in J} X_\iota(x) Y_\iota(y).$$

$$\tilde{\kappa}(x,y) = \sum_{\iota \in J} X_\iota(x) Y_\iota(y).$$

Simple choice: Taylor's formula applied with respect to y:

$$\begin{aligned} J &= \{0, 1, \ldots, k-1\}, \\ X_\iota(x) &= \text{derivatives of } \kappa(x,\cdot) \text{ evaluated at } y = y^*, \\ Y_\iota(y) &= (y - y^*)^\iota. \end{aligned}$$

The kernel $\kappa(x,y) = \log|x-y|$ leads to the error estimate

$$|\kappa(x,y) - \tilde{\kappa}(x,y)| \le \frac{|y-y^*|^k/k}{(|x-y^*| - |y-y^*|)^k} \quad \text{for} \quad |x-y^*| \ge |y-y^*|.$$

If κ is replaced by $\tilde{\kappa}$, the integral $a_{ij} = \int_{x_{j-1}}^{x_j} \kappa(x_{i-1/2}, y)dy$ becomes

$$\tilde{a}_{ij} = \sum_{\iota \in J} X_\iota(x_{i-1/2}) \int_{x_{j-1}}^{x_j} Y_\iota(y)dy. \qquad (*)$$

Let b be a block and restrict i, j in $(*)$ to b. Then $(*)$ describes a block matrix $\tilde{A}|_b$. Each term of the sum in $(*)$ is an *R1*-matrix $ab^\top$ with

$$a_i = X_\iota(x_{i-1/2}), \quad b_j = \int_{x_{j-1}}^{x_j} Y_\iota(y)dy.$$

Since $\#J = k$, the block $\tilde{A}|_b$ is of *Rk*-type.

Furthermore, one can check that

$$|\kappa(x,y) - \tilde{\kappa}(x,y)| \leq \frac{1}{k}\left(\frac{1}{2}\right)^k, \qquad \|A - \tilde{A}\|_\infty \leq 2^{-k}/k.$$

Discretisation error $h^\varkappa$, where the step size h is related to $n = \#I$ by $h \sim \frac{1}{n}$. Hence k should be chosen such that

$$2^{-k} \sim \left(\frac{1}{n}\right)^\varkappa.$$

Hence,

$$k = O(\log n)$$

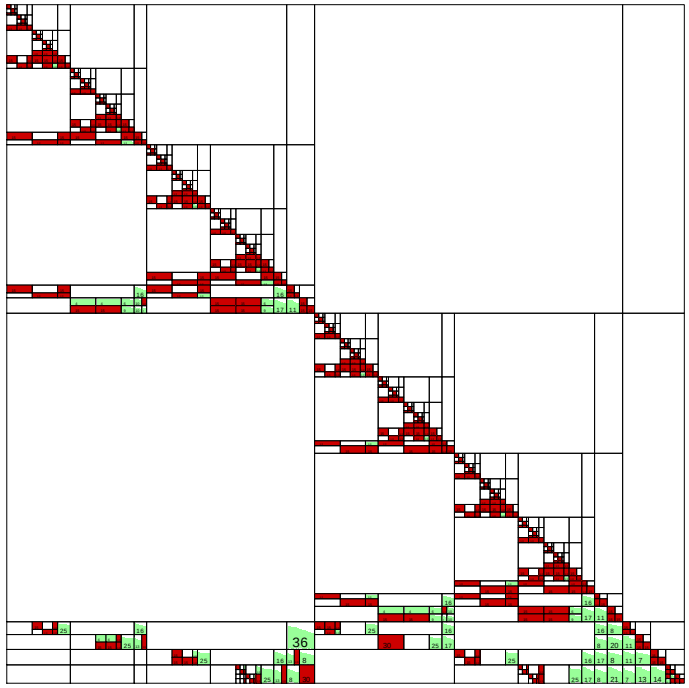
is the required rank.

NOTE: a) The construction of the cluster and block-cluster tree is automatic (block-box) and fast. Even refinements with form-regular elements are allowed. b) Similarly, the construction of the approximation $\tilde{A}$ is block-box like (usually by interpolation instead of Taylor expansion).

5 Application to FEM

REMARK: a) A FEM system matrix is an $\mathcal{H}$-matrix. Non-trivial blocks $= 0$.
b) For a uniformly elliptic differential operator with L^∞-coefficients, the inverse of the FEM-matrix can exponentially well be approximated by an $\mathcal{H}$-matrix [Bebendorf - Hackbusch 2003].

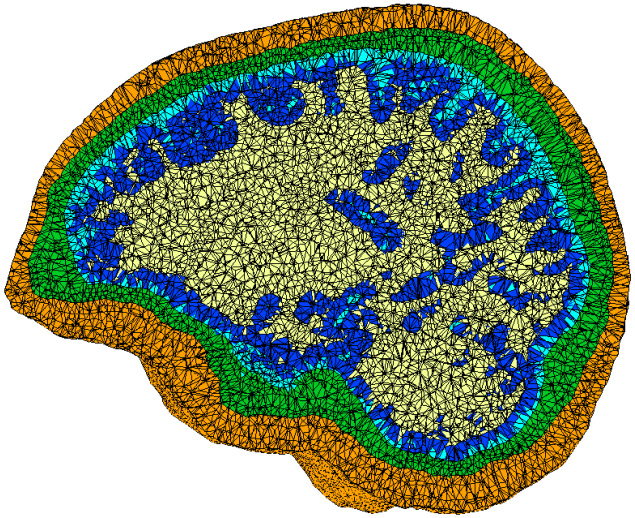
When solving a linear system of equations $Ax = b$, one can make use of the LU decomposition. The particular advantage of the LU decomposition for sparse matrices A is that the factors L and U contain many zero blocks (fill-in is not complete!).



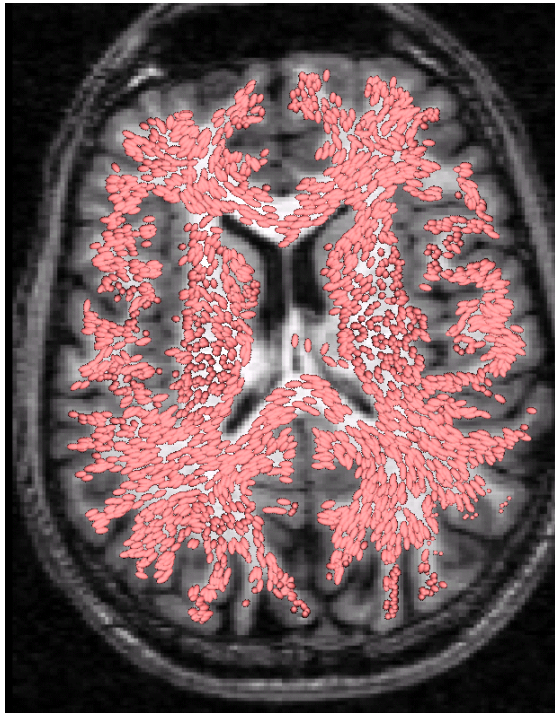
Example of an factor L:

EXAMPLE (inverse Problem, Wolters-Grasedyck-Hackbusch, 2004):
Given: electric/magnetic field (EEG,MEG) at $\approx$ 400 sensor positions on the head surface.
What is the current distribution in the brain ? Where are the sources ?

PDE: $-\text{div}\,\sigma(x)\nabla u(x) = f(x), \quad x \in \Omega \subset \mathbb{R}^3$, $\partial_n u = 0$ on $\partial\Omega$.
The boundary value has to be solved for $\approx$ 400 right-hand sides



Triangulation with
$N = 147287$ tetraeder



conductivity σ

- Galerkin discretisation $\rightsquigarrow Ax = b$
- The system has to be solved for ≈ 400 right-hand sides b
- Stopping criterion: $\|Ax - b\| / \|b\| \leq 10^{-8}$
- Machine: SUNFire, 900 MHz, single processor

	Pardiso†	$\mathrm{LU}_{\mathcal{H}}$, $\varepsilon = 10^{-6}$	PEBBLES‡
Setup	237	468	13
Solve	2.4	1.0	10
Total	1197	868	4013

†Pardiso (Schenk & Co)

‡PEBBLES (Langer/Haase)

6 $\mathcal{H}^2$-Matrices

Two hierarchies are involved:

1. Hierarchy given by the cluster tree T.

2. The involved rank-k-matrices do not use arbitrary row and column vectors, but vectors from special subspaces V_τ ($\tau \in T$) , i.e., the matrix blocks belong to tensor spaces $V_\tau \otimes V_\sigma$

3. The basis of V_τ is connected with the bases of $V_{\tau'}$ for $\tau' \in S(\tau)$. This leads to hierarchically defined bases: $V_\tau|_{\tau'} \subset V_{\tau'}$.

Since, in the end, the bases need not be stored directly, the log-factor disappears:

$$storage(A),\ cost(A * x),\ cost(A + B),\ cost(A * B) = O(n)$$

and smaller constants (see S. Börm 2004ff).

7 Matrix Equations

$$\begin{array}{lrcl} \text{Lyapunov:} & AX + XA^\top & = & C \\ \text{Sylvester} & AX - XB & = & C \\ \text{Riccati:} & AX + XA^\top - XFX & = & C \end{array}$$

Given: A, B, C, F; desired matrix-valued solution: X.

Applications: optimal control problems for elliptic / parabolic pdes.

- Low rank $C, F \Rightarrow$ low rank X

- $\mathcal{H}$-matrix C, low rank $F \Rightarrow \mathcal{H}$-matrix X

Computation via $\mathcal{H}$-arithmetic, possibly combined with multi-grid methods.

Matrix-Riccati Equation

$$A^\top X + XA - XFX + G = O \qquad (A < O).$$

Lemma 7.1 *The solution X satisfies*

$$X = -(M^\top M)^{-1} M^\top N,$$

where

$$\begin{bmatrix} M & N \end{bmatrix} := \mathsf{sign}\left(\begin{bmatrix} A^\top & G \\ F & -A \end{bmatrix}\right) - \begin{bmatrix} I & O \\ O & I \end{bmatrix}.$$

Lemma 7.2 *Assume that $\Re e\lambda \neq 0$ for all eigenvalues $\lambda \in \sigma(S)$.*

Start: $S^{(0)} := S$. Then the iteration

$$S^{(i+1)} := \frac{1}{2}\left(S^{(i)} + \left(S^{(i)}\right)^{-1}\right)$$

converges quadratically to $\mathsf{sign}(S)$.

Example of a matrix-Riccati equation by L. Grasedyck

Choice of A by $A = \Delta_h$ (1D-Laplacian).

The following table shows the relative error $\left\| \tilde{X} - X \right\|_2 / \|X\|_2$.

	$n = 101$	256	1024	65 536
$k = 1$	8.8_{10}-3	1.5_{10}-1	1.3_{10}-1	-
$k = 2$	2.4_{10}-4	2.6_{10}-4	4.2_{10}-4	6.7_{10}-4
$k = 4$	7.7_{10}-8	9.1_{10}-8	1.1_{10}-7	6.2_{10}-7
$k = 6$	1.9_{10}-10	3.7_{10}-10	2.4_{10}-10	1.7_{10}-9
Number of iterations	12	14	17	26
time* [sec]	2.2	8.5	67	18263

*) k=2, Sun Quasar 450 MHz

In the last case, the (full) matrix X has $4,294,967,296$ entries.

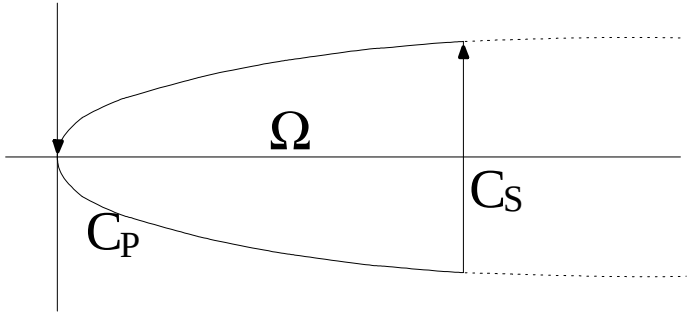
8 Matrix-Valued Functions $f(A)$

EXAMPLE: Matrix-exponential function e^{-tA}.

Cauchy-Dunford representation

$$e^{-tA} = \frac{1}{2\pi i} \int_\Gamma e^{-zt} (zI - A)^{-1} \, dt$$

using a parabola Γ :



After parametrisation and quadrature:

$$T_N(t) := \sum_{\ell=-N}^{N} \gamma_\ell e^{-\alpha_\ell t} (z_\ell I - A)^{-1}, \qquad z_\ell \in \Gamma.$$

Error estimate for $t \geq t_0 > 0$:

$$\left\| T_N(t) - e^{-tA} \right\| \lesssim e^{-cN^{2/3}}.$$

$\Rightarrow N \sim \log n \Rightarrow$ Total cost: $O(n \log^* n)$.

See: Gavrilyuk-Hackbusch-Khoromskij, 2002.

9 Higher dimensional-analogue: Tensor systems

The analogue of rank-k-matrices are sums of k tensor products.

Tensor space:

$$\mathbb{V} := V_1 \otimes V_2 \otimes \ldots \otimes V_d.$$

Example: $V_i = \mathbb{R}^{I_i}$ for index sets I_i. Then the entries of $v \in \mathbb{V}$ are

$$v_{i_1,i_2,\ldots,i_d} \qquad \text{with } i_j \in I_j.$$

$\mathbb{V}$ is isomorphic to $\mathbb{R}^I$ with the product index set $I := I_1 \times I_2 \times \ldots \times I_d$.

DEFINITION: A rank-k-tensor is of the form

$$\sum_{\mu=1}^{k} v_1^{(\mu)} \otimes v_2^{(\mu)} \otimes \ldots \otimes v_d^{(\mu)} \qquad \text{with } v_j^{(\mu)} \in V_j.$$

DEFINITION: A rank-k-tensor is of the form

$$\sum_{\mu=1}^{k} v_1^{(\mu)} \otimes v_2^{(\mu)} \otimes \ldots \otimes v_d^{(\mu)} \qquad \text{with } v_j^{(\mu)} \in V_j. \tag{*}$$

QUESTION: Given $v \in \mathbb{V}$, are there best rank-k-approximations (*) ?
How can they be computed?

REMARK: Tools like the singular-value decomposition do not exist for $d \geq 3$.
Non-existence of best-approximations and numerical instability possible.

A trust-region Newton method for determining the best rank-k-approximation (after a regularisation) is described by Espig (Diss. 2008).

Example from the electronic Schrödinger equation

Hartree-Fock equation $F_\psi\,\psi_b(\mathbf{y}) = \epsilon_b\,\psi_b(\mathbf{y})$ involves the Hartree potential

$$V_H(\mathbf{x}) = 2\sum_{b=1}^{N/2}\int \frac{\psi_b^*(\mathbf{y})\psi_b(\mathbf{y})}{|\mathbf{x}-\mathbf{y}|}d\mathbf{y} = \int \frac{\rho(\mathbf{y})}{|\mathbf{x}-\mathbf{y}|}\,d\mathbf{y}, \tag{1}$$

where $\rho(\mathbf{y}) = 2\sum_{b=1}^{N/2}\psi_b(\mathbf{y})\psi_b^*(\mathbf{y})$ is the electron density.

Standard approach uses Gaussians $g_k^{(j)}(y_j) = (y_j - A_k^{(j)})^{\ell_k}\,e^{-\alpha_k(y_j-A_k^{(j)})^2}$ to represent the orbital (wavefunction) by

$$\psi_b(\mathbf{y}) \approx \sum_{k=1}^{K_\psi} g_k^{(1)}(y_1)\,g_k^{(2)}(y_2)\,g_k^{(3)}(y_3). \tag{2}$$

Here, K_ψ = tensor rank. We start with a representation (2) produced by the MOLPRO program package using the MATROP program for matrix operations. Eq. (2) yields $\rho(\mathbf{y}) = \psi_b^*(\mathbf{y})\psi_b(\mathbf{y})$ with $K := K_\psi(K_\psi+1)/2$ terms.

Optimising the tensor representation reduces the tensor rank to a much smaller rank κ while almost keeping the same order of accuracy:

$$\rho(\mathbf{y}) \approx \sum_{k=1}^{\kappa} \varrho_k^{(1)}(y_1)\, \varrho_k^{(2)}(y_2)\, \varrho_k^{(3)}(y_3), \qquad \kappa \ll K.$$

The computational work for evaluating the Hartree potential (1) depends essentially on the tensor rank.

EXAMPLE CH_4: The MOLPRO program yields $K = 1540$, which can be reduced by our approach to $\kappa = 45$. The computing time for evaluating V_H for the tensor representation with $\kappa = 45$ is 8 hours, while the estimated time for $K = 1540$ is 190 hours.

molecule	initial rank $K$of $\rho(y)$	final rank κ	relative error	error in energy (hartree)
CH_4	1540	45	9.0×10^{-6}	6.0×10^{-5}
C_2H_2	2346	50	1.3×10^{-4}	5.0×10^{-4}
C_2H_6	4656	55	8.8×10^{-5}	4.0×10^{-4}

see Rao Chinnamsetty - Espig - Khoromskij - Hackbusch - Flad: J. Chem. Physics 127 (2007) and Rao Chinnamsetty, Diss. 2008.

Kronecker-Tensor Products

$V_j = \mathbb{R}^{I_j \times J_j}$ vector spaces of matrices. Then

$$\mathbb{V} := V_1 \otimes V_2 \otimes \ldots \otimes V_d \cong \mathbb{R}^{I \times J}$$

with $I := I_1 \times I_2 \times \ldots \times I_d$ and $J := J_1 \times J_2 \times \ldots \times J_d$.

Notation for $d = 2$: $A \otimes B = \begin{bmatrix} A_{11}B & A_{12}B & \ldots \\ A_{21}B & \ddots & \\ \vdots & & \end{bmatrix}$

REMARK: a) For $d = 2$ the approximation of a matrix M by a Kronecker-rank-k expression $\sum_{\mu=1}^{k} A^{(\mu)} \otimes B^{(\mu)}$ is equivalent to a certain standard rank-k approximation of a related matrix $\tilde{M}$.

b) For $d \geq 3$ the search for rank-k approximations is more involved.

If the matrix is the discretisation of a continuous operator with a kernel function $\varkappa(\mathbf{x}, \mathbf{y})$, $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, analytical methods may help (see next example).

Separable PDE in $[0,1]^d$, d large

Let $\Omega = (0,1)^d \subset \mathbb{R}^d$.
Equidistant grid: $\Omega_h = (h, 2h, \ldots, nh)$ with

$$(n+1)\,h = 1 \qquad \text{(here } n = 1024\text{)}.$$

Separable PDE: $L = \sum_{\nu=1}^{d} a_\nu(x_\nu)\frac{\partial^2}{\partial x_\nu^2}$, e.g., $L = \Delta$.

Discretisation of $-L$ by usual difference formula:

$$A = -L_h = -\sum_{\nu=1}^{d} a_\nu(x_\nu) D^h_{x_\nu x_\nu} \qquad (D^h_{x_\nu x_\nu}\text{: 2nd difference})$$
$$= A_1 \otimes I \otimes \ldots \otimes I + I \otimes A_2 \otimes \ldots \otimes I + \ldots + I \otimes I \otimes \ldots \otimes A_d$$

Goal: Approximation of L_h^{-1}.

Numerical result (Grasedyck 2004):
For $d = 2048$, accuracy 10^{-5} to 10^{-6}: 5 min computer time

Related dimension:

$$N = 1024^{2048} = 1.24 \times 10^{6165}.$$

Underlying method

$1/x$ can be approximated by exponential sums $\sum_{\nu=1}^{k} \omega_\nu \exp(\alpha_\nu x)$:

$$\min_{\omega_\nu,\alpha_\nu} \max_{x\in[x_0,x_1]} \left|\frac{1}{x} - \sum_{\nu=1}^{k} \omega_\nu \exp(\alpha_\nu x)\right| \leq O(e^{-ck}), \quad c > 0,$$

$$\min_{\omega_\nu,\alpha_\nu} \max_{x\in[x_0,\infty)} \left|\frac{1}{x} - \sum_{\nu=1}^{k} \omega_\nu \exp(\alpha_\nu x)\right| \leq O(e^{-ck^{1/2}}), \quad c > 0.$$

Let $[x_0, x_1]$ or $[x_0, \infty)$ contain the spectrum of L_h. Then

$$L_h^{-1} \approx \sum_{\nu=1}^{k} \omega_\nu \exp(\alpha_\nu L_h).$$

The special tensor structure

$$L_h = \sum_{\mu=1}^{d} I \otimes \ldots \otimes I \otimes L_{h,\mu} \otimes I \otimes \ldots \otimes I$$

implies $\exp(\alpha_\nu L_h) = \bigotimes_{\mu=1}^{d} \exp(\alpha_\nu L_{h,\mu})$.
Approximation of $\exp(\alpha_\nu L_{h,\mu})$ by $\mathcal{H}$-matrices (see above). Finally:

$$L_h^{-1} \approx \sum_{\nu=1}^{k} \omega_\nu \bigotimes_{\mu=1}^{d} \exp_{\mathcal{H}}(\alpha_\nu L_{h,\mu}) \qquad \text{(rank-}k\text{-tensor)}.$$

10 Final Remarks

- Concerning papers about the subject "hierarchical matrices" see

  `http://www.mis.mpg.de` ($\rightarrow$ institute reports) or

  `http://www.mis.mpg.de/scicomp/hackbusch_e.html`

- For scientific purpose the software library HLib is freely available (ask for a licence form)

- Every year we organise a winter school on this subject.

- For commercial applications: HLibPro (distributed via the Fraunhofer-Institute SCAI in St.Augustin)