Hierarchical interpolative factorization

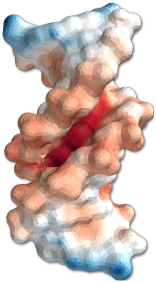
Kenneth L. Ho (Stanford)

Joint work with Lexing Ying

CNA Seminar, CMU, Oct. 2014

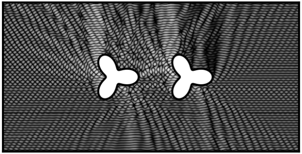
Introduction

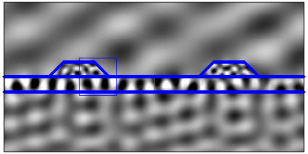
Elliptic PDEs in differential or integral form:



$$-\nabla \cdot (a(x)\nabla u(x)) + b(x)u(x) = f(x)$$
$$a(x)u(x) + \int_{\Omega} K(x,y)u(y)\, d\Omega(y) = f(x)$$

- Fundamental to science and engineering
- Interested in 2D/3D, complex geometry
- Discretize $\rightarrow$ structured linear system $Au = f$





Goal: fast, accurate, and robust algorithms to compute $u = A^{-1}f$

How to solve? Let $A \in \mathbb{C}^{N \times N}$.

- If N is small, use direct methods (Gaussian elimination, matrix factorization)
  - Stable, robust, rapid updates
  - $O(N^3)$ complexity, infeasible for $N \gtrsim 10^4$
- If N is large, use iterative methods (CG, GMRES, multigrid)
  - Can be accelerated to $O(n_{\text{iter}} N)$ complexity, highly scalable
  - Extremely successful, workhorse of modern scientific computing

How to solve? Let $A \in \mathbb{C}^{N \times N}$.

- If N is small, use direct methods (Gaussian elimination, matrix factorization)
  - Stable, robust, rapid updates
  - $O(N^3)$ complexity, infeasible for $N \gtrsim 10^4$
- If N is large, use iterative methods (CG, GMRES, multigrid)
  - Can be accelerated to $O(n_{\text{iter}} N)$ complexity, highly scalable
  - Extremely successful, workhorse of modern scientific computing

But ...

- What if n_{iter} is large (high contrasts, geometric singularities)?
- What if there are many RHS's (time stepping, inverse problems)?

How to solve? Let $A \in \mathbb{C}^{N\times N}$.

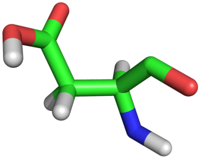
- If N is small, use direct methods (Gaussian elimination, matrix factorization)
  - Stable, robust, rapid updates
  - $O(N^3)$ complexity, infeasible for $N \gtrsim 10^4$
- If N is large, use iterative methods (CG, GMRES, multigrid)
  - Can be accelerated to $O(n_{\mathrm{iter}}N)$ complexity, highly scalable
  - Extremely successful, workhorse of modern scientific computing

But . . .

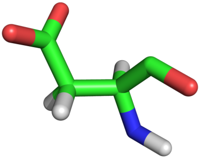
- What if n_{iter} is large (high contrasts, geometric singularities)?
- What if there are many RHS's (time stepping, inverse problems)?

In certain important environments, there is a need for fast direct solvers.

Example: protein pK_a calculations



$\rightleftharpoons$

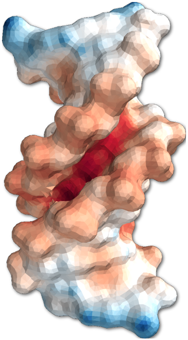


$$\mathrm{p}K_\mathrm{a} \equiv -\log_{10}\frac{[\mathrm{A}]\,[\mathrm{H}]}{[\mathrm{AH}]} = \frac{\beta}{\ln 10}\Delta G^{\mathrm{p}}_{\mathrm{AH}\to\mathrm{A+H}}$$

$$\begin{aligned}\Delta G^{\mathrm{p}}_{\mathrm{AH}\to\mathrm{A+H}} &= \Delta G^{\mathrm{s}}_{\mathrm{AH}\to\mathrm{A+H}} + \Delta G^{\mathrm{s}\to\mathrm{p}}_{\mathrm{A}} - \Delta G^{\mathrm{s}\to\mathrm{p}}_{\mathrm{AH}} \\ &= \underbrace{\Delta G^{\mathrm{s}}_{\mathrm{AH}\to\mathrm{A+H}}}_{\text{experiment}} + \underbrace{\Delta G^{\mathrm{s}}_{\mathrm{A}\to\mathrm{AH}} - \Delta G^{\mathrm{p}}_{\mathrm{A}\to\mathrm{AH}}}_{\text{electrostatic only}}\end{aligned}$$

- Ionization behavior is important for enzymatic and structural properties

Example: protein pK_a calculations



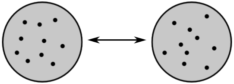
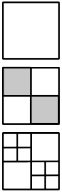
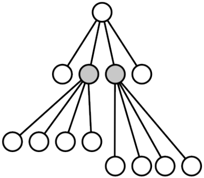
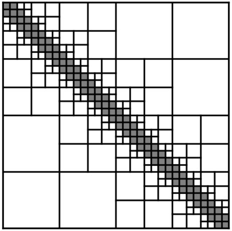
- Linearized Poisson-Boltzmann equation
- Discretize: $A(\Sigma)u = f(q)$
  - Σ: molecular geometry
  - q: atomic partial charges
- Fixed matrix, one solve for each of n_{titr} titrating sites
- Conformational flexibility: $O((n_{\text{titr}} n_{\text{rot}})^p)$ perturbed solves

Potential for massive acceleration using fast direct methods.

Related problems:

- Protein structure prediction, protein-protein docking, protein design
- Inverse scattering, time stepping, materials design

- **PDEs**: exploit sparsity (multifrontal), reduce to IE-like systems
- **IEs**: exploit FMM-type hierarchical low-rank structure
- $\mathcal{H}$-matrices: $O(N \log^{\alpha} N)$ but with a large constant
- HSS matrices/recursive skeletonization: $O(N)$ in 1D, $O(N^{3/2})$ in 2D, $O(N^2)$ in 3D
- HSS/RS/MF with structured matrix algebra: $O(N)$ in 2D, $O(N^{4/3})$ in 3D

[Ambikasaran, Bebendorf, Börm, Bremer, Chandrasekaran, Chen, Corona, Darve, Gillman, Greengard, Gu, Hackbusch, Ho, Li, Martinsson, Rokhlin, Schmitz, Starr, Xia, Ying, Young, Zorin]

Overview

Hierarchical interpolative factorization

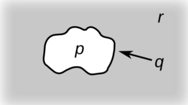
- RS/MF + recursive dimensional reduction
- Same idea as using structured algebra but much simpler
- Explicit matrix sparsification, generalized LU decomposition
- Linear or quasilinear complexity, small constants
- Unified formalism for IEs and PDEs
- Works for 2D/3D, adaptive and complex geometry

Tools: sparse elimination, interpolative decomposition, skeletonization

Sparse elimination

Let

$$A = \begin{bmatrix} A_{pp} & A_{pq} & \\ A_{qp} & A_{qq} & A_{qr} \\ & A_{rq} & A_{rr} \end{bmatrix}.$$



(Think of A as a sparse matrix.) If A_{pp} is nonsingular, define

$$R_p^* = \begin{bmatrix} I & & \\ -A_{qp}A_{pp}^{-1} & I & \\ & & I \end{bmatrix}, \quad S_p = \begin{bmatrix} I & -A_{pp}^{-1}A_{pq} & \\ & I & \\ & & I \end{bmatrix}$$

so that

$$R_p^* A S_p = \begin{bmatrix} A_{pp} & & \\ & * & A_{qr} \\ & A_{rq} & A_{rr} \end{bmatrix}.$$

- DOFs p have been eliminated
- Interactions involving r are unchanged

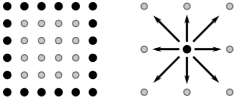
Interpolative decomposition

If $A_{:,q}$ is numerically low-rank, then there exist

- skeleton ($\hat{q}$) and redundant ($\check{q}$) columns partitioning $q = \hat{q} \cup \check{q}$
- an interpolation matrix T_q

such that

$$A_{:,\check{q}} \approx A_{:,\hat{q}} T_q.$$



- Essentially a pivoted QR written slightly differently
- Rank-revealing to any specified precison $\epsilon > 0$

Interactions between separated regions are low-rank.

[Cheng/Gimbutas/Martinsson/Rokhlin, Gu/Eisenstat]

Skeletonization

- Let $A = \begin{bmatrix} A_{pp} & A_{pq} \\ A_{qp} & A_{qq} \end{bmatrix}$ with A_{pq} and A_{qp} low-rank
- Apply ID to $\begin{bmatrix} A_{qp} \\ A_{pq}^* \end{bmatrix}$: $\begin{bmatrix} A_{q\check{p}} \\ A_{\check{p}q}^* \end{bmatrix} \approx \begin{bmatrix} A_{q\hat{p}} \\ A_{\hat{p}q}^* \end{bmatrix} T_p \implies \begin{array}{c} A_{q\check{p}} \approx A_{q\hat{p}} T_p \\ A_{\check{p}q} \approx T_p^* A_{\hat{p}q} \end{array}$
- Reorder $A = \begin{bmatrix} A_{\check{p}\check{p}} & A_{\check{p}\hat{p}} & A_{\check{p}q} \\ A_{\hat{p}\check{p}} & A_{\hat{p}\hat{p}} & A_{\hat{p}q} \\ A_{q\check{p}} & A_{q\hat{p}} & A_{qq} \end{bmatrix}$, define $Q_p = \begin{bmatrix} I & & \\ -T_p & I & \\ & & I \end{bmatrix}$
- **Sparsify** via ID: $Q_p^* A Q_p \approx \begin{bmatrix} * & * & \\ * & A_{\hat{p}\hat{p}} & A_{\hat{p}q} \\ & A_{q\hat{p}} & A_{qq} \end{bmatrix}$
- Eliminate: $R_{\check{p}}^* Q_p^* A Q_p S_{\check{p}} \approx \begin{bmatrix} * & & \\ & * & A_{\hat{p}q} \\ & A_{q\hat{p}} & A_{qq} \end{bmatrix}$
- Reduces to a subsystem involving skeletons only

[Ho/Ying, Xia/Xi/Gu]

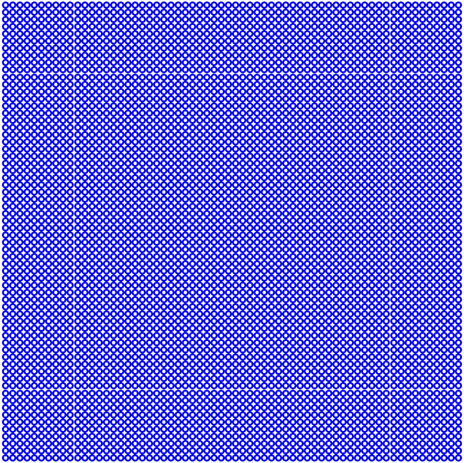
Integral equations

- Old algorithm (RS) in new factorization form
- New algorithm: HIF-IE

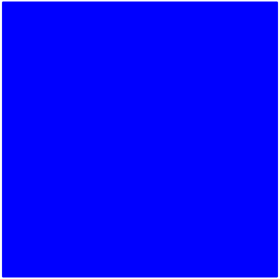
Algorithm: recursive skeletonization factorization

Build quadtree/octree.
for each level $\ell = 0, 1, 2, \ldots, L$ from finest to coarsest **do**
    Let C_ℓ be the set of all cells on level ℓ.
    for each cell $c \in C_\ell$ **do**
        Skeletonize remaining DOFs in c.
    end for
end for

RSF in 2D: level 0

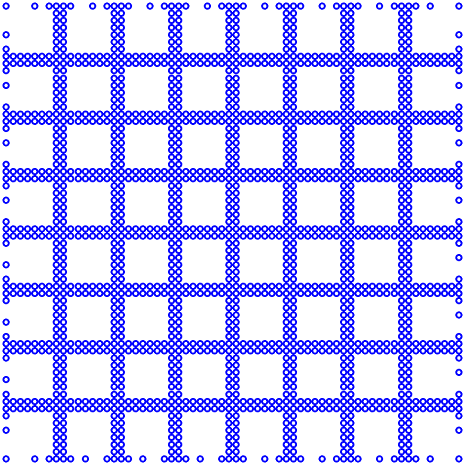


domain

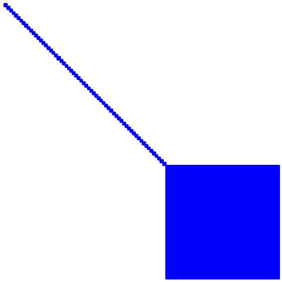


matrix

RSF in 2D: level 1

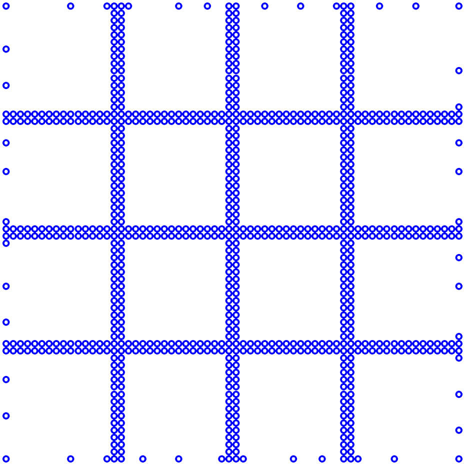


domain

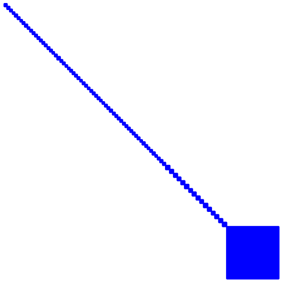


matrix

RSF in 2D: level 2

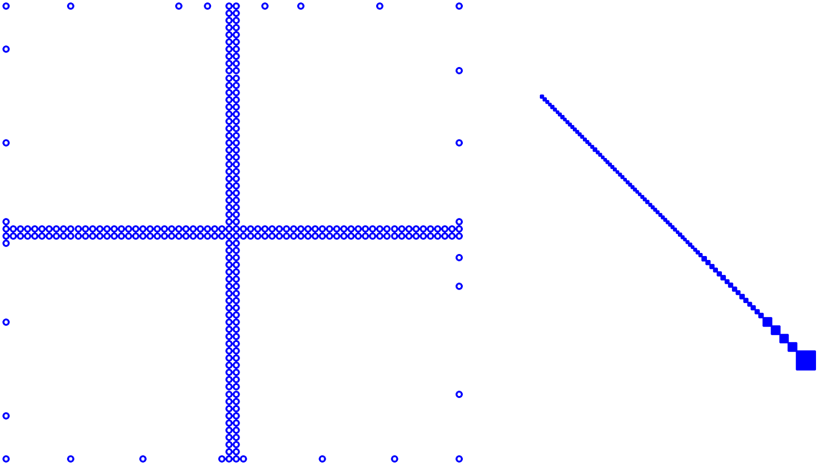


domain



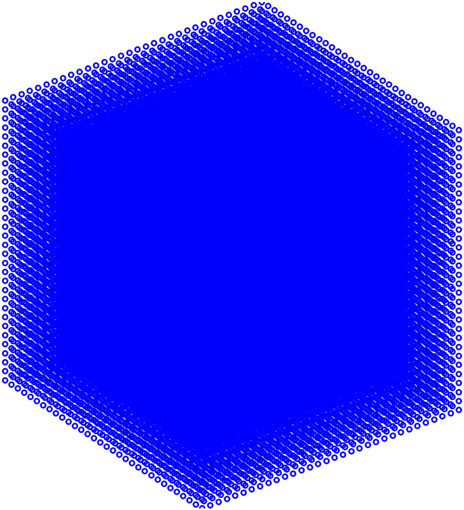
matrix

RSF in 2D: level 3



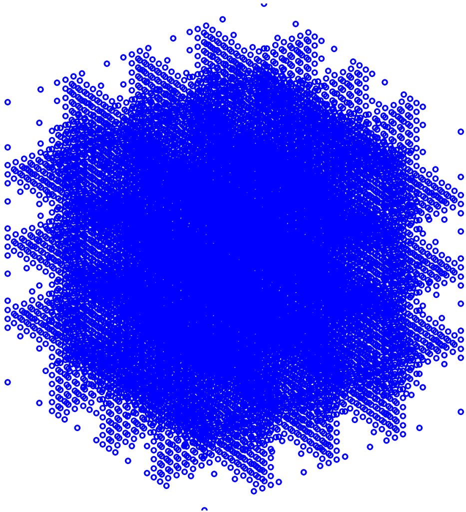
domain matrix

RSF in 3D: level 0



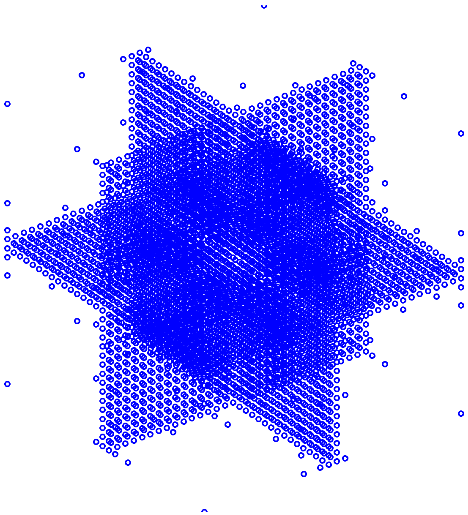
domain

RSF in 3D: level 1



domain

RSF in 3D: level 2



RSF analysis

- Skeletonization operators:

$$U_\ell = \prod_{c \in C_\ell} Q_c R_c, \quad V_\ell = \prod_{c \in C_\ell} Q_c S_c$$

$$Q_c = \begin{bmatrix} I & & \\ * & I & \\ & & I \end{bmatrix}, \quad R_c, S_c = \begin{bmatrix} I & * & \\ & I & \\ & & I \end{bmatrix}$$

- Block diagonalization:

$$D \approx U_{L-1}^* \cdots U_0^* A V_0 \cdots V_{L-1}$$

- Generalized LU decomposition:

$$A \approx U_0^{-*} \cdots U_{L-1}^{-*} D V_{L-1}^{-1} \cdots V_0^{-1}$$
$$A^{-1} \approx V_0 \cdots V_{L-1} D^{-1} U_L^* \cdots U_0^*$$

- Fast direct solver or preconditioner

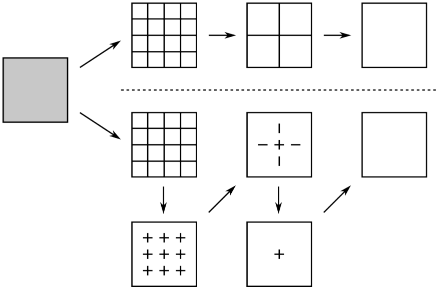
RSF analysis

The cost is determined by the skeleton size.

	1D	2D	3D
Skeleton size	$O(\log N)$	$O(N^{1/2})$	$O(N^{2/3})$
Factorization cost	$O(N)$	$O(N^{3/2})$	$O(N^2)$
Solve cost	$O(N)$	$O(N \log N)$	$O(N^{4/3})$

Question: How to reduce the skeleton size in 2D and 3D?

RSF analysis



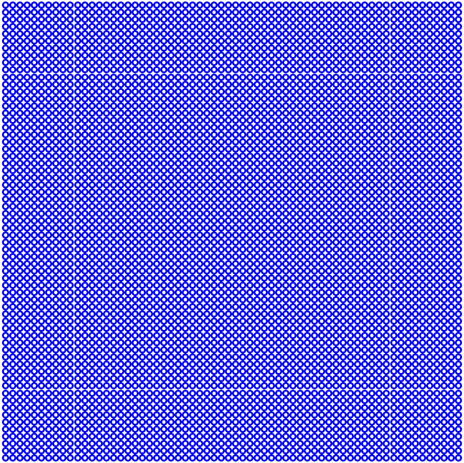
Question: How to reduce the skeleton size in 2D and 3D?

- Skeletons cluster near cell interfaces (Green's theorem)
- Exploit skeleton geometry by further skeletonizing along interfaces
- Dimensional reduction

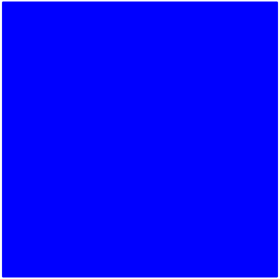
Algorithm: hierarchical interpolative factorization for IEs in 2D

Build quadtree.
for each level $\ell = 0, 1, 2, \dots, L$ from finest to coarsest **do**
  Let C_ℓ be the set of all cells on level ℓ.
  for each cell $c \in C_\ell$ **do**
    Skeletonize remaining DOFs in c.
  end for
  Let $C_{\ell+1/2}$ be the set of all edges on level ℓ.
  for each cell $c \in C_{\ell+1/2}$ **do**
    Skeletonize remaining DOFs in c.
  end for
end for

HIF-IE in 2D: level 0

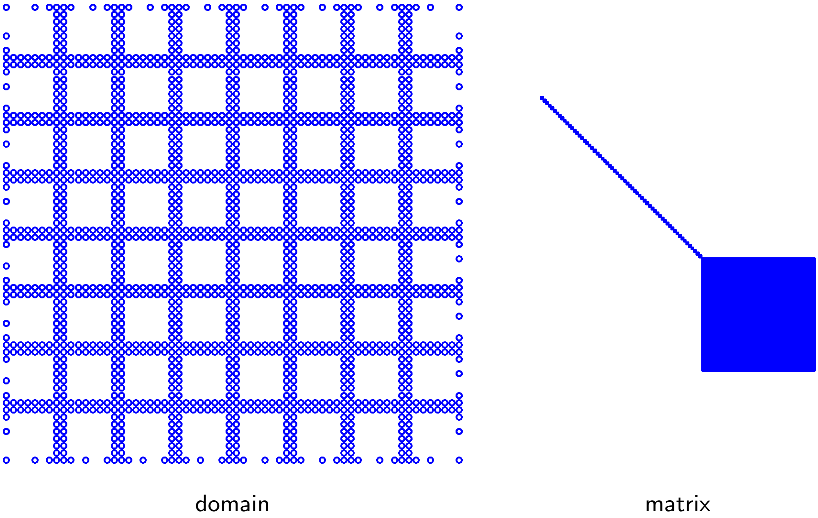


domain

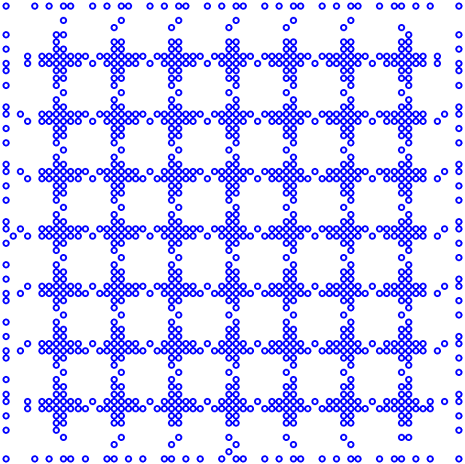


matrix

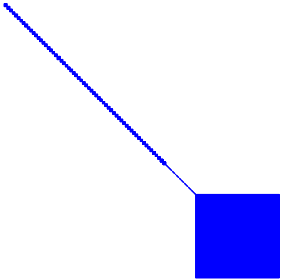
HIF-IE in 2D: level 1/2



HIF-IE in 2D: level 1

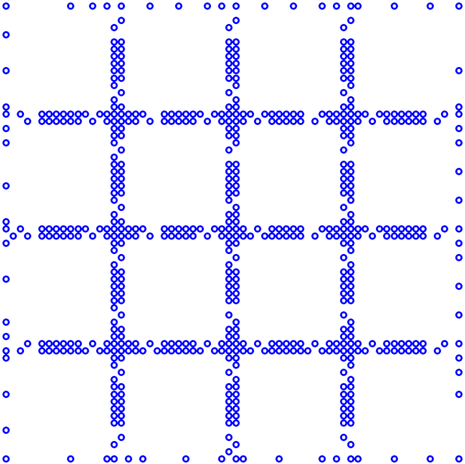


domain

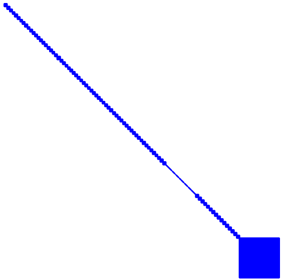


matrix

HIF-IE in 2D: level 3/2

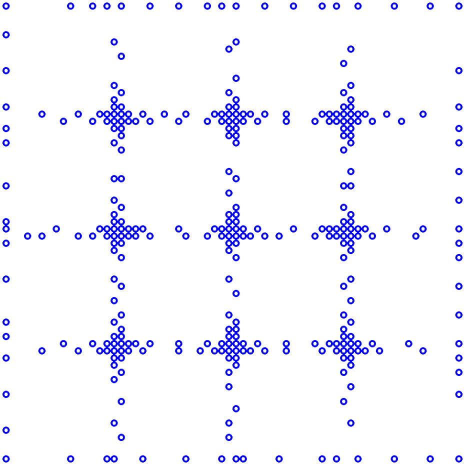


domain

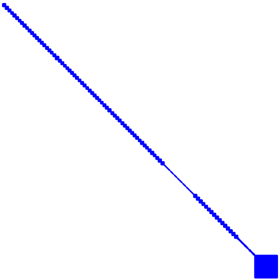


matrix

HIF-IE in 2D: level 2

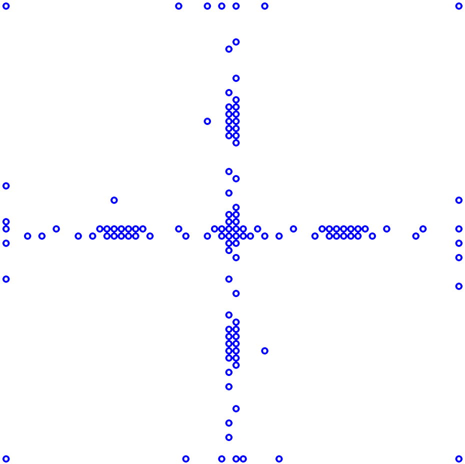


domain

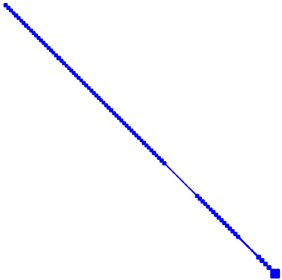


matrix

HIF-IE in 2D: level 5/2

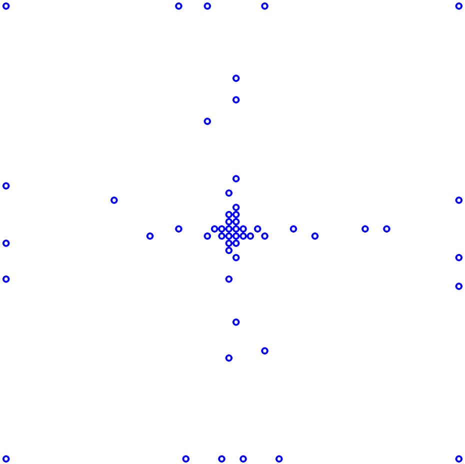


domain

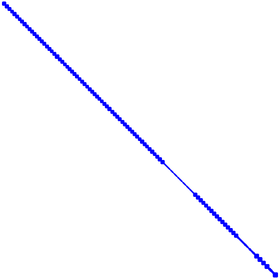


matrix

HIF-IE in 2D: level 3

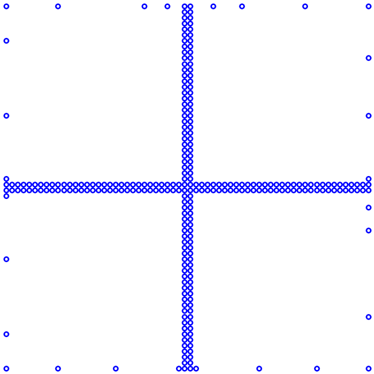


domain

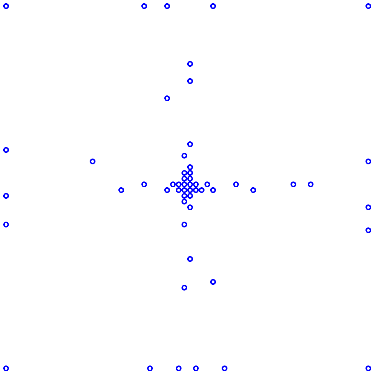


matrix

RSF vs. HIF-IE in 2D

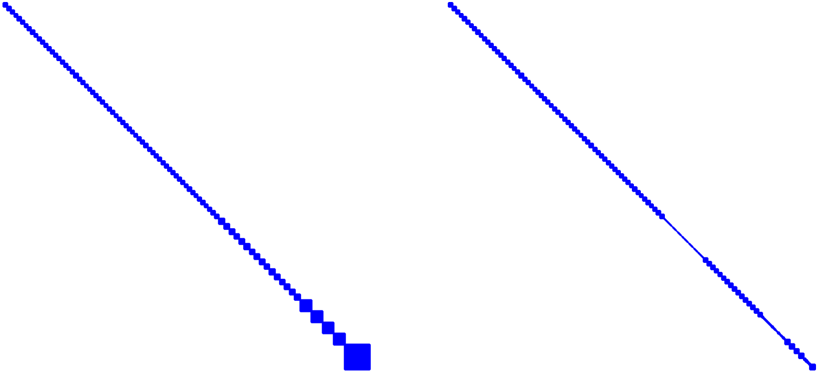


RSF



HIF-IE

RSF vs. HIF-IE in 2D

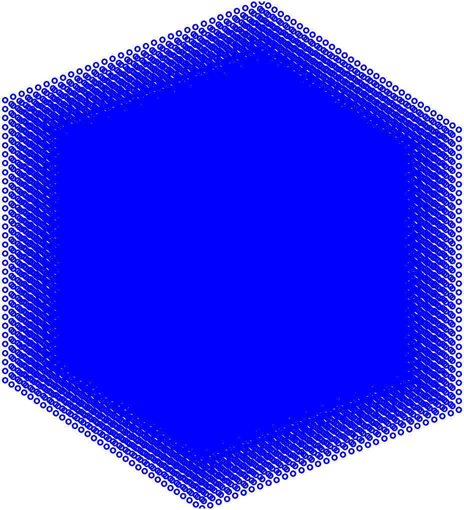


RSF HIF-IE

Algorithm: hierarchical interpolative factorization for IEs in 3D

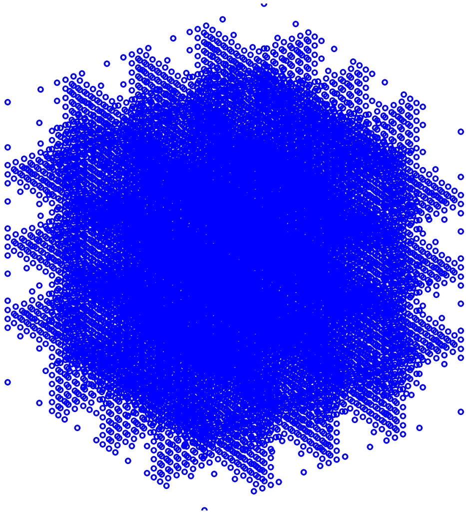
Build octree.
for each level $\ell = 0, 1, 2, \ldots, L$ from finest to coarsest **do**
  Let C_ℓ be the set of all cells on level ℓ.
  for each cell $c \in C_\ell$ **do**
    Skeletonize remaining DOFs in c.
  end for
  Let $C_{\ell+1/3}$ be the set of all faces on level ℓ.
  for each cell $c \in C_{\ell+1/3}$ **do**
    Skeletonize remaining DOFs in c.
  end for
  Let $C_{\ell+2/3}$ be the set of all edges on level ℓ.
  for each cell $c \in C_{\ell+2/3}$ **do**
    Skeletonize remaining DOFs in c.
  end for
end for

HIF-IE in 3D: level 0



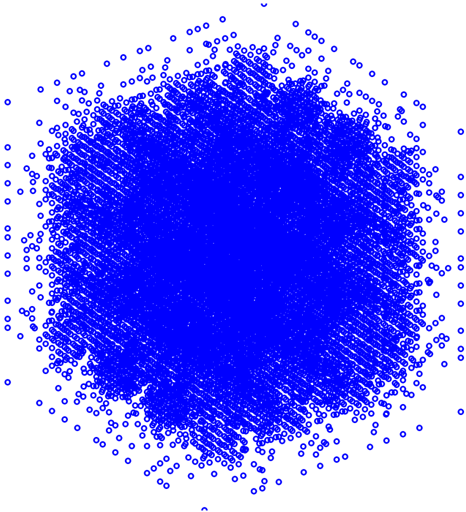
domain

HIF-IE in 3D: level 1/3



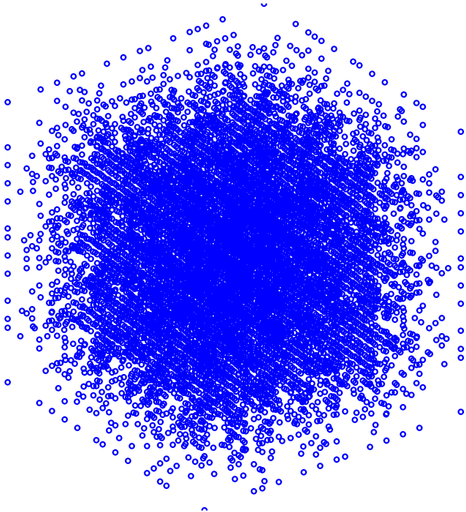
domain

HIF-IE in 3D: level 2/3



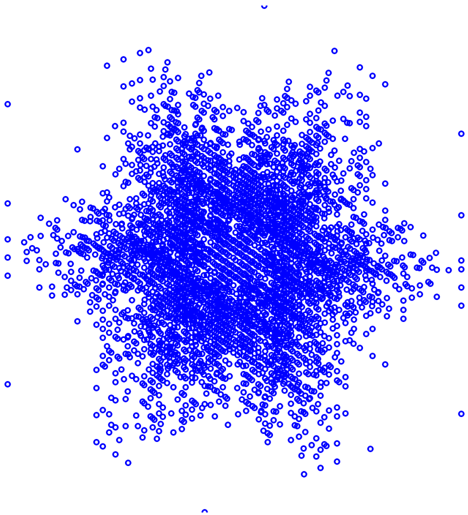
domain

HIF-IE in 3D: level 1



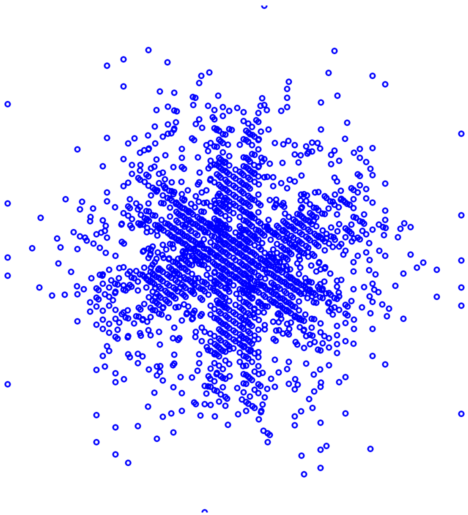
domain

HIF-IE in 3D: level 4/3



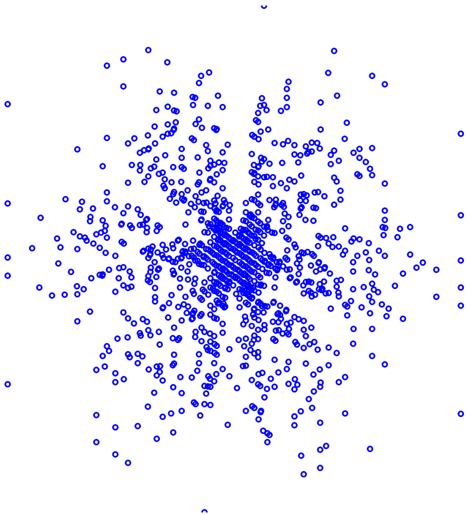
domain

HIF-IE in 3D: level 5/3



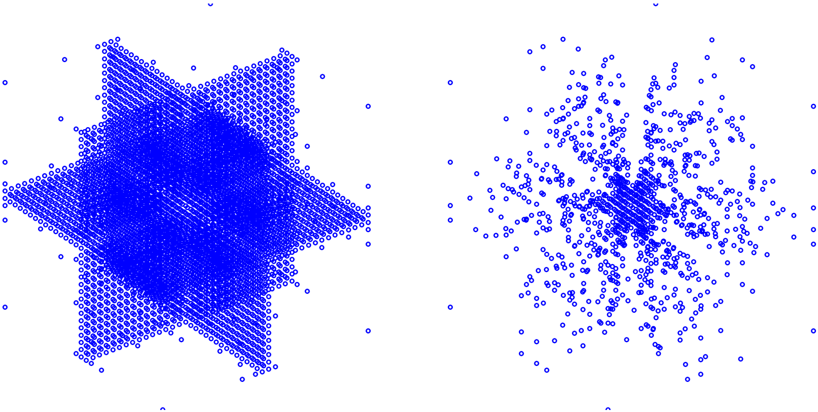
domain

HIF-IE in 3D: level 2



domain

RSF vs. HIF-IE in 3D



RSF HIF-IE

HIF-IE analysis

- 2D:

$$A \approx U_0^{-*} U_{1/2}^{-*} \cdots U_{L-1/2}^{-*} D V_{L-1/2}^{-1} \cdots V_{1/2}^{-1} V_0^{-1}$$
$$A^{-1} \approx V_0 V_{1/2} \cdots V_{L-1/2} D^{-1} U_{L-1/2}^{*} \cdots U_{1/2}^{*} U_0^{*}$$

- 3D:

$$A \approx U_0^{-*} U_{1/3}^{-*} U_{2/3}^{-*} \cdots U_{L-1/3}^{-*} D V_{L-1/3}^{-1} \cdots V_{2/3}^{-1} V_{1/3}^{-1} V_0^{-1}$$
$$A^{-1} \approx V_0 V_{1/3} V_{2/3} \cdots V_{L-1/3} D^{-1} U_{L-1/3}^{*} \cdots U_{2/3}^{*} U_{1/3}^{*} U_0^{*}$$

Conjecture:

Skeleton size:	$O(\log N)$
Factorization cost:	$O(N)$
Solve cost:	$O(N)$

HIF-IE analysis

- 2D:
$$A \approx U_0^{-*} U_{1/2}^{-*} \cdots U_{L-1/2}^{-*} D V_{L-1/2}^{-1} \cdots V_{1/2}^{-1} V_0^{-1}$$
$$A^{-1} \approx V_0 V_{1/2} \cdots V_{L-1/2} D^{-1} U_{L-1/2}^{*} \cdots U_{1/2}^{*} U_0^{*}$$

- 3D:
$$A \approx U_0^{-*} U_{1/3}^{-*} U_{2/3}^{-*} \cdots U_{L-1/3}^{-*} D V_{L-1/3}^{-1} \cdots V_{2/3}^{-1} V_{1/3}^{-1} V_0^{-1}$$
$$A^{-1} \approx V_0 V_{1/3} V_{2/3} \cdots V_{L-1/3} D^{-1} U_{L-1/3}^{*} \cdots U_{2/3}^{*} U_{1/3}^{*} U_0^{*}$$

Conjecture:

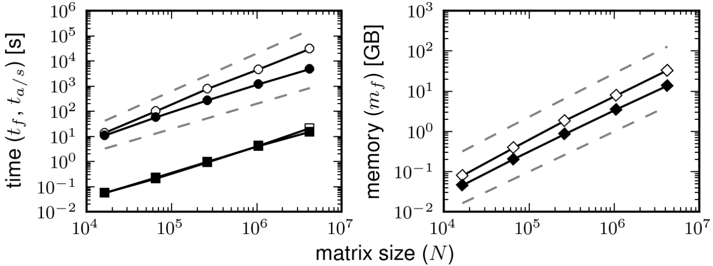
Skeleton size:	$O(\log N)$
Factorization cost:	$O(N)$
Solve cost:	$O(N)$

Actually slightly more complicated . . .

Numerical results in 2D

First-kind volume IE on the unit square with

$$a(x) \equiv 0, \quad K(x,y) = -\frac{1}{2\pi} \log \|x - y\|.$$

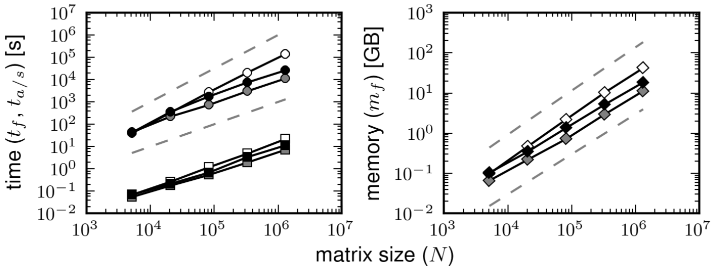


- **rskelf2** (white), **hifie2** (black)
- Factorization time (○), solve time (□), memory (◇) at precision $\epsilon = 10^{-6}$
- Reference scalings (gray dashes):
  - Left: $O(N)$ and $O(N^{3/2})$
  - Right: $O(N)$ and $O(N \log N)$

Numerical results in 3D

Second-kind boundary IE on the unit sphere with

$$a(x) \equiv -\frac{1}{2}, \quad K(x,y) = \frac{\partial}{\partial \nu(y)} \frac{1}{4\pi \|x-y\|}.$$

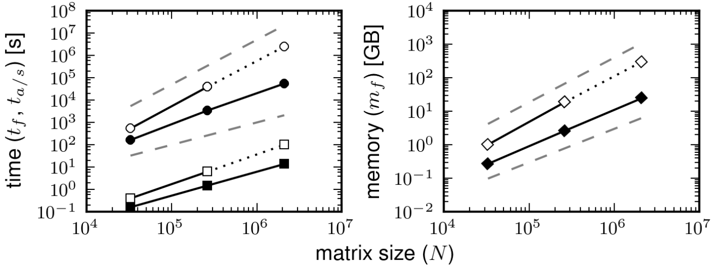


- **rskelf3** (white), **hifie3** (gray), **hifie3x** (black)
- Factorization time (○), solve time (□), memory (◇) at precision $\epsilon = 10^{-3}$
- Reference scalings (gray dashes):
  - Left: $O(N)$ and $O(N^{3/2})$
  - Right: $O(N)$ and $O(N \log N)$

Numerical results in 3D

First-kind volume IE on the unit cube with

$$a(x) \equiv 0, \quad K(x,y) = \frac{1}{4\pi \|x-y\|}.$$



- **rskelf3** (white), **hifie3** (black)
- Factorization time (○), solve time (□), memory (◇) at precision $\epsilon = 10^{-3}$
- Reference scalings (gray dashes):
  - Left: $O(N)$ and $O(N^2)$
  - Right: $O(N)$ and $O(N^{4/3})$

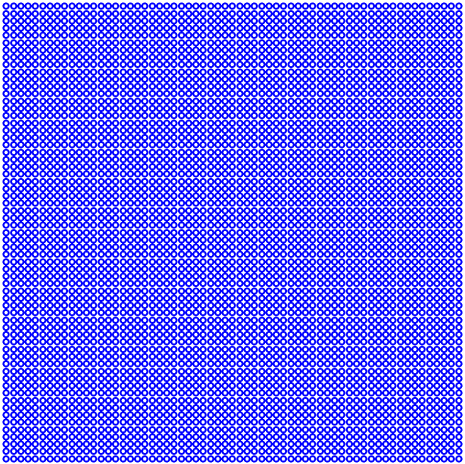
Differential equations

- Old algorithm (MF)
- New algorithm: HIF-DE
- Exploit sparsity: trivial “skeletonization”, thin separators

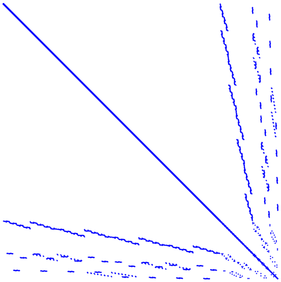
Algorithm: multifrontal

Build quadtree/octree.
for each level $\ell = 0, 1, 2, \ldots, L$ from finest to coarsest **do**
    Let C_ℓ be the set of all cells on level ℓ.
    for each cell $c \in C_\ell$ **do**
        Eliminate remaining interior DOFs in c.
    end for
end for

MF in 2D: level 0

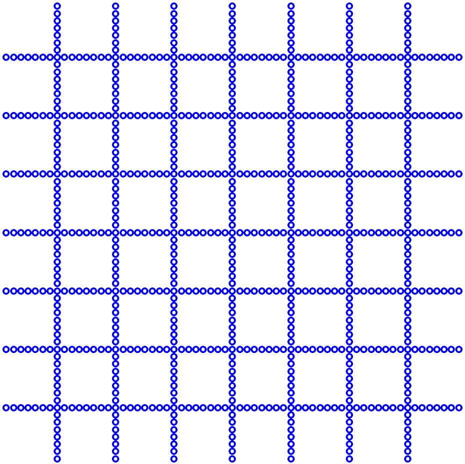


domain

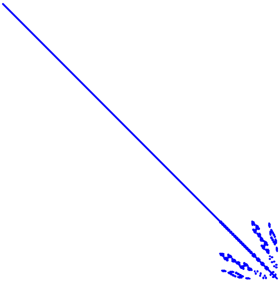


matrix

MF in 2D: level 1

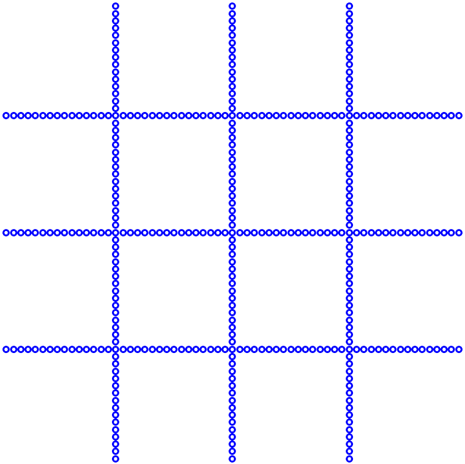


domain

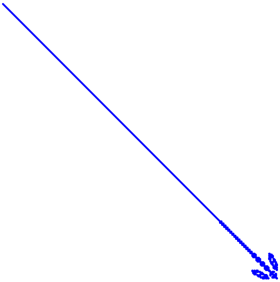


matrix

MF in 2D: level 2

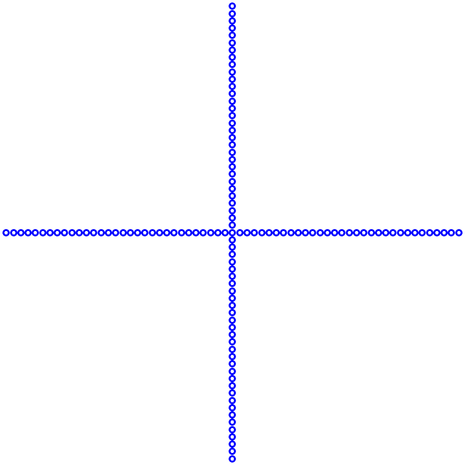


domain

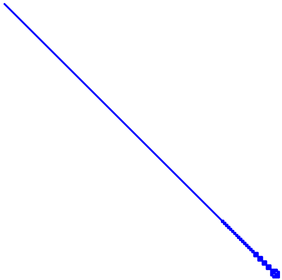


matrix

MF in 2D: level 3

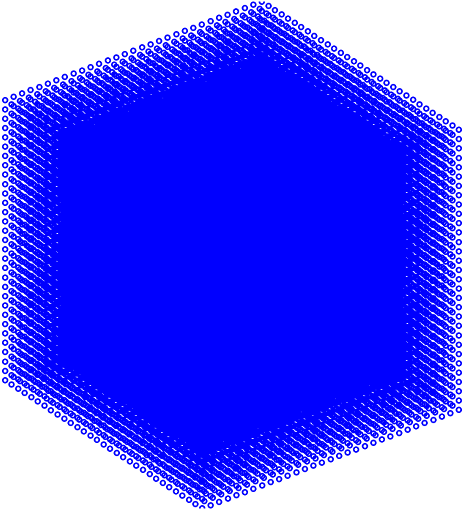


domain



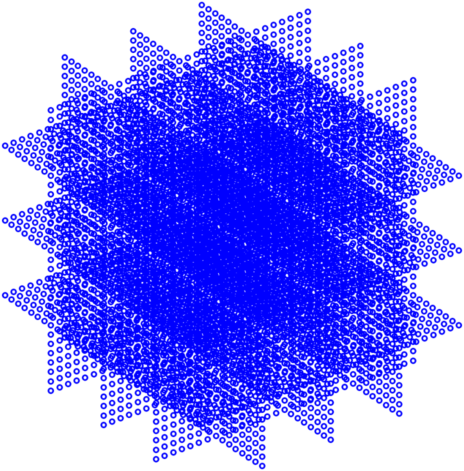
matrix

MF in 3D: level 0



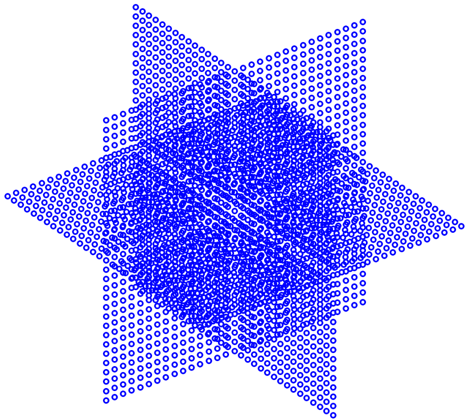
domain

MF in 3D: level 1



domain

MF in 3D: level 2

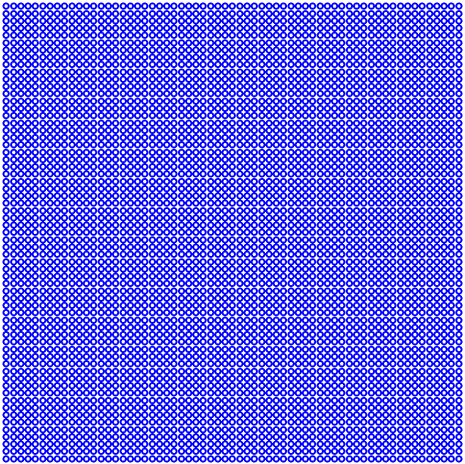


domain

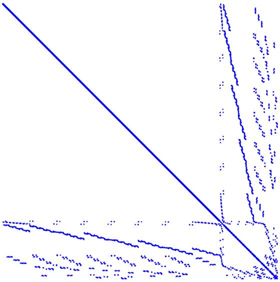
Algorithm: hierarchical interpolative factorization for PDEs in 2D

Build quadtree.
for each level $\ell = 0, 1, 2, \ldots, L$ from finest to coarsest **do**
  Let C_ℓ be the set of all cells on level ℓ.
  for each cell $c \in C_\ell$ **do**
    Eliminate remaining interior DOFs in c.
  end for
  Let $C_{\ell+1/2}$ be the set of all edges on level ℓ.
  for each cell $c \in C_{\ell+1/2}$ **do**
    Skeletonize remaining DOFs in c.
  end for
end for

HIF-DE in 2D: level 0

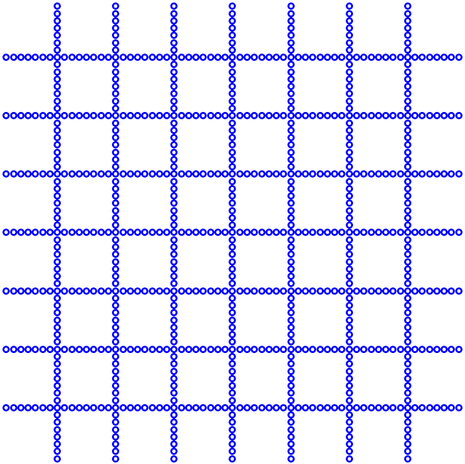


domain

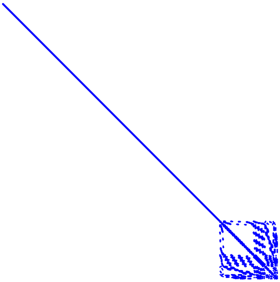


matrix

HIF-DE in 2D: level 1/2

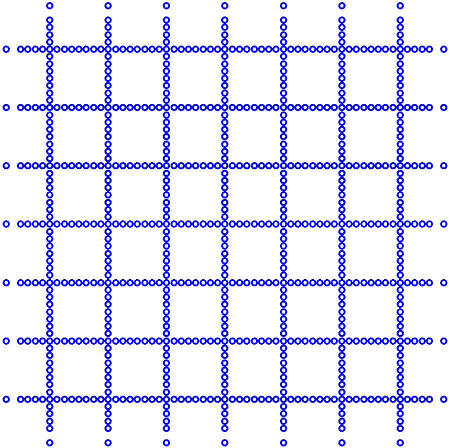


domain

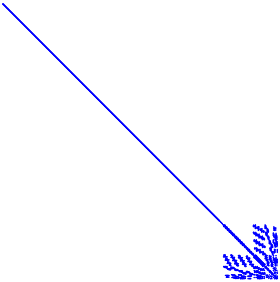


matrix

HIF-DE in 2D: level 1

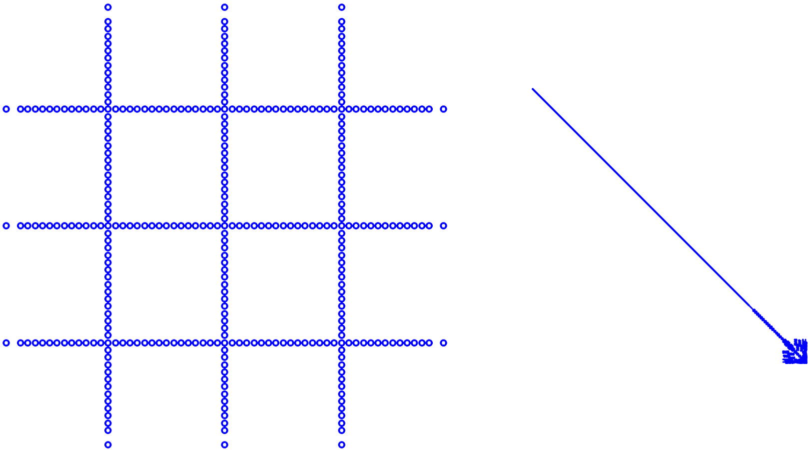


domain



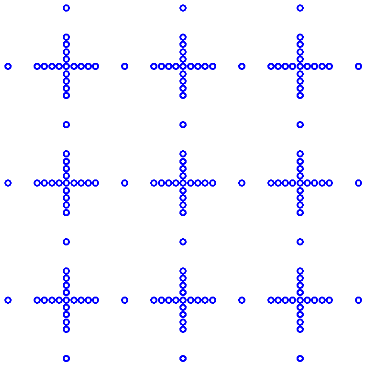
matrix

HIF-DE in 2D: level 3/2

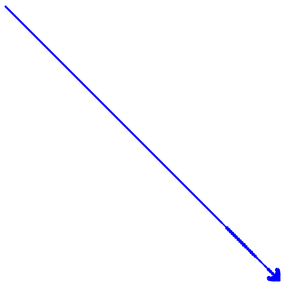


domain matrix

HIF-DE in 2D: level 2

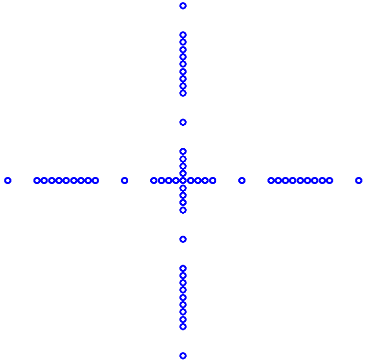


domain

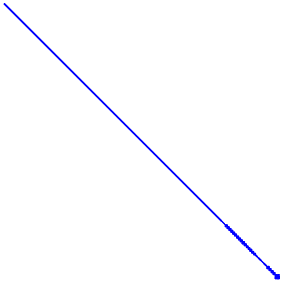


matrix

HIF-DE in 2D: level 5/2

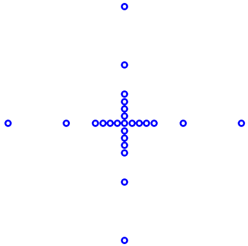


domain

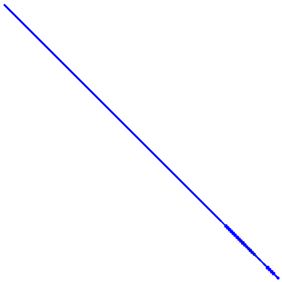


matrix

HIF-DE in 2D: level 3

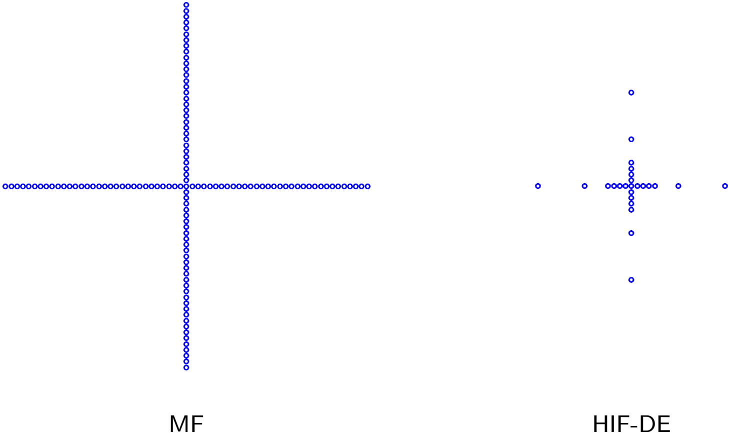


domain

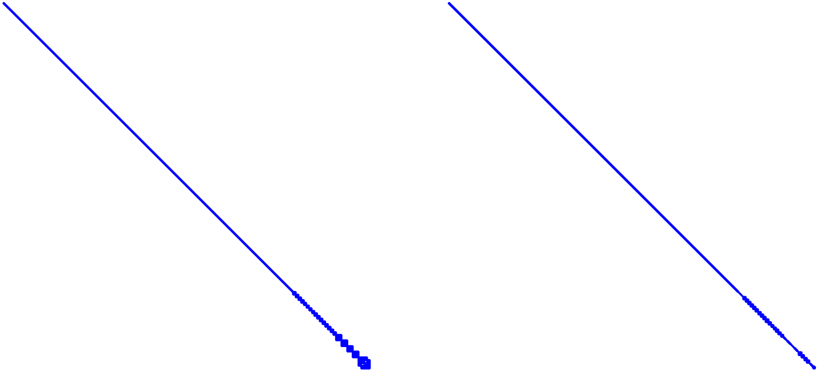


matrix

MF vs. HIF-DE in 2D



MF vs. HIF-DE in 2D

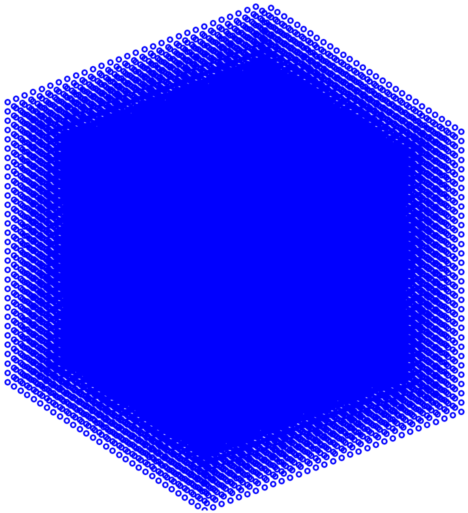


MF HIF-DE

Algorithm: hierarchical interpolative factorization for PDEs in 3D

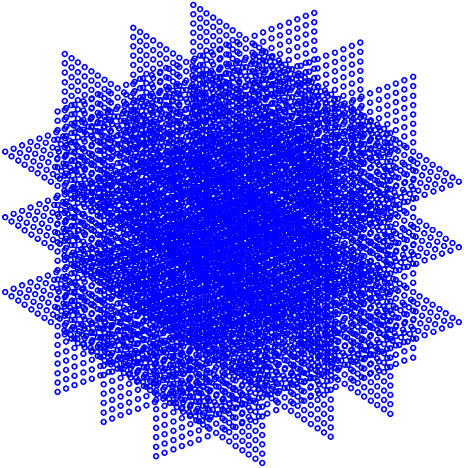
Build octree.
for each level $\ell = 0, 1, 2, \ldots, L$ from finest to coarsest **do**
    Let C_ℓ be the set of all cells on level ℓ.
    for each cell $c \in C_\ell$ **do**
        Eliminate remaining interior DOFs in c.
    end for
    Let $C_{\ell+1/3}$ be the set of all faces on level ℓ.
    for each cell $c \in C_{\ell+1/3}$ **do**
        Skeletonize remaining DOFs in c.
    end for
    Let $C_{\ell+2/3}$ be the set of all edges on level ℓ.
    for each cell $c \in C_{\ell+2/3}$ **do**
        Skeletonize remaining DOFs in c.
    end for
end for

HIF-DE in 3D: level 0



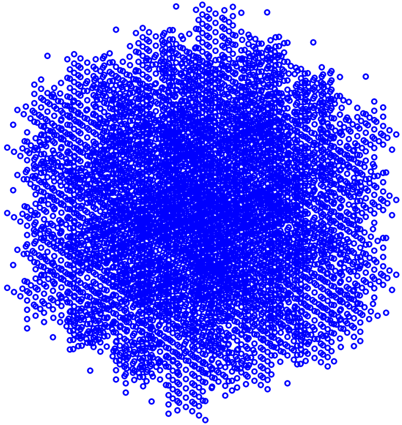
domain

HIF-DE in 3D: level 1/3



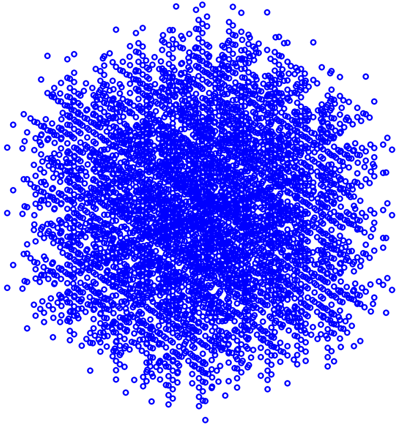
domain

HIF-DE in 3D: level 2/3



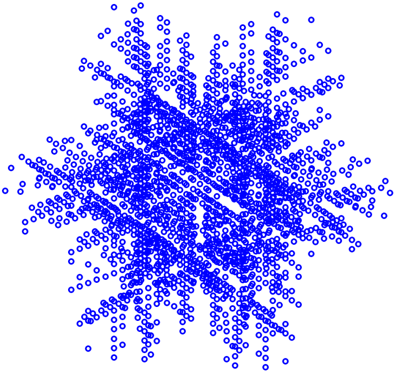
domain

HIF-DE in 3D: level 2



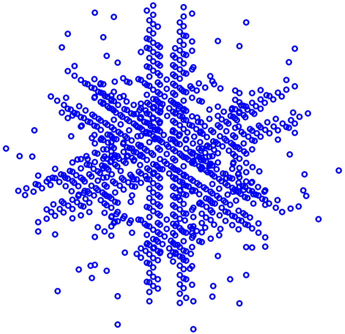
domain

HIF-DE in 3D: level 4/3



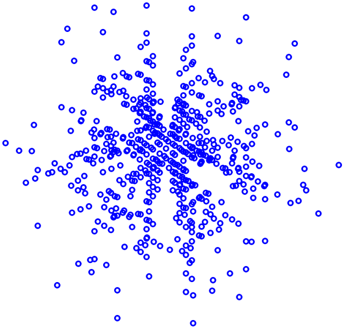
domain

HIF-DE in 3D: level 5/3



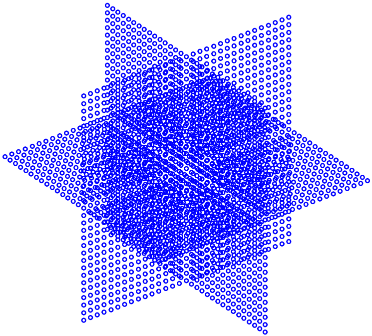
domain

HIF-DE in 3D: level 2

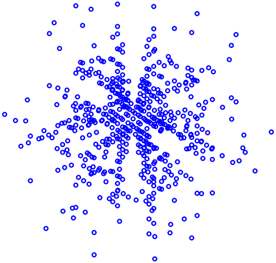


domain

MF vs. HIF-DE in 3D



MF

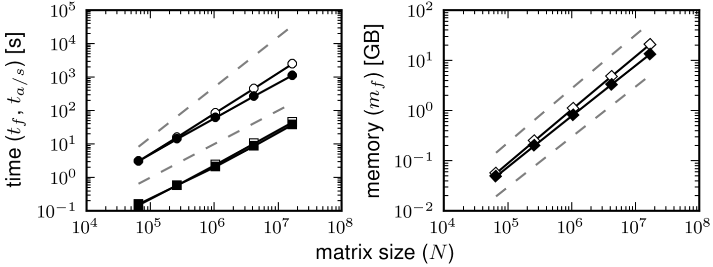


HIF-DE

Numerical results in 2D

Five-point stencil on the unit square with

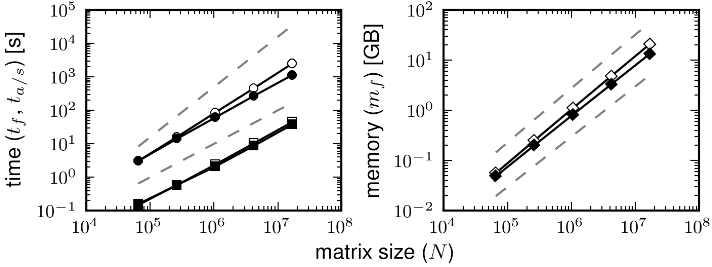
$$a(x) \equiv 1, \quad b(x) \equiv 0.$$



- **mf2** (white), **hifde2** (black)
- Factorization time ($\circ$), solve time ($\square$), memory ($\diamond$) at precision $\epsilon = 10^{-9}$
- Reference scalings (gray dashes):
  - Left: $O(N)$ and $O(N^{3/2})$
  - Right: $O(N)$ and $O(N \log N)$

Numerical results in 2D

Five-point stencil on the unit square with $a(x)$ a quantized high-contrast random field, $b(x) \equiv 0$.

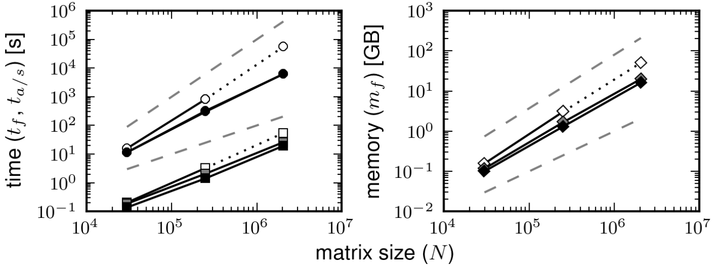


- **mf2** (white), **hifde2** (black)
- Factorization time ($\circ$), solve time ($\square$), memory ($\diamond$) at precision $\epsilon = 10^{-9}$
- Reference scalings (gray dashes):
  - Left: $O(N)$ and $O(N^{3/2})$
  - Right: $O(N)$ and $O(N \log N)$

Numerical results in 3D

Seven-point stencil discretization on the unit cube with

$$a(x) \equiv 1, \quad b(x) \equiv 0.$$



- **mf3** (white), **hifde3** (gray), **hifde3x** (black)
- Factorization time ($\circ$), solve time ($\Box$), memory ($\diamond$) at precision $\epsilon = 10^{-6}$
- Reference scalings (gray dashes):
  - Left: $O(N)$ and $O(N^2)$
  - Right: $O(N)$ and $O(N^{4/3})$

Conclusions

- Efficient factorization of structured operators in 2D and 3D
  - Fast matrix-vector multiplication
  - Fast direct solver at high accuracy, preconditioner otherwise
  - Empirical **linear complexity** but no proof yet
- Sparsification and elimination (skeletonization) via the ID
- Dimensional reduction by alternating between cells, faces, and edges
- Can be viewed as adaptive numerical coarsening
- **Extensions**: $A^{1/2}$, $\log\det A$, $\operatorname{diag} A^{-1}$
- Naturally parallelizable, block-sweep structure

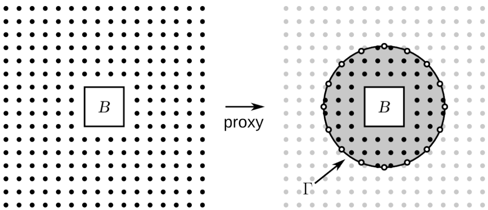
- *Perspective*: structured dense matrices can be sparsified very efficiently
- Can borrow directly from sparse algorithms, e.g., RSF $=$ MF
- What other features of sparse matrices can be exploited?

MATLAB codes available at `https://github.com/klho/FLAM/`.

Additional slides

Proxy compression

- Main cost of algorithm is computing IDs of tall-and-skinny matrices
- **Global** operation can be reduced to local operation using Green's theorem
- Suffices to compress against neighbors plus "proxy" surface
- Crucial for beating $O(N^2)$ complexity



Second-kind IEs

- IEs of the form $u(x) + \int_\Omega K(x, y)u(y)\, d\Omega(y) = f(x)$
- High contrast in diagonal vs. off-diagonal entries
- Mixing of cell, face, edge in HIF-IE leads to error
- Need to use effective precision $O(\epsilon/N)$
- Quasilinear complexity estimates:

	2D	3D
Factorization cost	$O(N \log N)$	$O(N \log^6 N)$
Solve cost	$O(N \log \log N)$	$O(N \log^2 N)$