# Hierarchical interpolative factorization 

Kenneth L. Ho (Stanford)<br>Joint work with Lexing Ying

SIAM AN 2014

## Introduction

Integral equations associated with elliptic PDEs:

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

- Many applications in science and engineering
- Interested in 2D/3D, complex geometry
- Discretize $\rightarrow$ structured linear system $A u=f$

Goal: fast and accurate algorithms for the discrete operator

- Fast matrix-vector multiplication, fast direct solver, good preconditioner
- Ideally, fast matrix factorization
- Linear or quasilinear complexity, high practical efficiency


## Related work

Fast matrix-vector multiplication

- $O(N)$ or $O(N \log N)$ using FMM, treecode, $\mathcal{H} / \mathcal{H}^{2}$-matrices
- Fast iterative solver by combining with GMRES, CG, etc.


Fast direct solver

- HSS matrices/recursive skeletonization
- $O(N)$ in $1 \mathrm{D}, O\left(N^{3 / 2}\right)$ in 2D, $O\left(N^{2}\right)$ in 3 D
- H-matrices: $O\left(N \log ^{\alpha} N\right)$ but with a large constant
- HSS/RS with structured matrix algebra: $O(N)$ in 2D
- Corona, Martinsson, Zorin (2013)
- More recent ideas: Siva's talk later this session

Hierarchical interpolative factorization

- RS + recursive dimensional reduction
- Same idea as using structured algebra but much simpler
- New matrix sparsification framework, generalized LU decomposition
- Linear or quasilinear complexity, small constants
- Works for 2D/3D, adaptive geometry

Tools: sparse elimination, interpolative decomposition, skeletonization

## Sparse elimination

Let

$$
A=\left[\begin{array}{ccc}
A_{p p} & A_{p q} & \\
A_{q p} & A_{q q} & A_{q r} \\
& A_{r q} & A_{r r}
\end{array}\right] .
$$


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

$$
R_{p}^{*}=\left[\begin{array}{ccc}
I & & \\
-A_{q p} A_{p p}^{-1} & I & \\
& & I
\end{array}\right], \quad S_{p}=\left[\begin{array}{ccc}
I & -A_{p p}^{-1} A_{p q} & \\
& I & \\
& & I
\end{array}\right]
$$

so that

$$
R_{p}^{*} A S_{p}=\left[\begin{array}{ccc}
A_{p p} & & \\
& * & A_{q r} \\
& A_{r q} & A_{r r}
\end{array}\right] .
$$

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


## Interpolative decomposition

If $A_{i, q}$ 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:

$$
\begin{aligned}
A_{:,(\hat{q},, \tilde{q})}=\left[\begin{array}{ll}
Q_{1} & Q_{2}
\end{array}\right]\left[\begin{array}{ll}
R_{11} & R_{12} \\
& R_{22}
\end{array}\right] & \approx Q_{1}\left[\begin{array}{ll}
R_{11} & R_{12}
\end{array}\right] \\
& \Longrightarrow A_{:, \check{q}} \approx Q_{1} R_{12}=\underbrace{Q_{1} R_{11}}_{A_{:} ; \hat{q}} \underbrace{\left(R_{11}^{-1} R_{12}\right)}_{T_{q}}
\end{aligned}
$$

Interactions between separated regions are low-rank.

## Skeletonization

- Efficient elimination of redundant DOFs from dense matrices
- Let $A=\left[\begin{array}{ll}A_{p p} & A_{p q} \\ A_{q p} & A_{q q}\end{array}\right]$ with $A_{p q}$ and $A_{q p}$ low-rank
- Apply ID to $\left[\begin{array}{c}A_{q p} \\ A_{p q}^{*}\end{array}\right]:\left[\begin{array}{c}A_{q \check{p}} \\ A_{\hat{p} q}^{*}\end{array}\right] \approx\left[\begin{array}{c}A_{q \hat{p}} \\ A_{\hat{p} q}^{*}\end{array}\right] T_{p} \Longrightarrow \begin{gathered}A_{q \check{p}} \approx A_{q \hat{p}} T_{p} \\ A_{\check{p} q} \approx T_{p}^{*} A_{\hat{p} q}\end{gathered}$
- Reorder $A=\left[\begin{array}{lll}A_{\check{\rho} \check{\rho}} & A_{\check{\rho} \hat{\rho}} & A_{\check{\rho} q} \\ A_{\hat{\rho} \check{\rho}} & A_{\hat{\rho} \hat{\rho}} & A_{\hat{\rho} q} \\ A_{q \check{\rho}} & A_{q \hat{\rho}} & A_{q q}\end{array}\right]$, define $Q_{p}=\left[\begin{array}{ccc}1 & & \\ -T_{p} & 1 & \\ & & 1\end{array}\right]$
- Sparsify via ID: $Q_{p}^{*} A Q_{p} \approx\left[\begin{array}{ccc}* & * & \\ * & A_{\hat{\rho} \hat{p}} & A_{\hat{p} q} \\ & A_{q \hat{p}} & A_{q q}\end{array}\right]$
- Sparse eliminate: $R_{\stackrel{p}{*}}^{*} Q_{p}^{*} A Q_{p} S_{\check{p}} \approx\left[\begin{array}{ccc}* & & \\ & * & A_{\hat{\rho} q} \\ & A_{q \hat{p}} & A_{q q}\end{array}\right]$


## Algorithm: recursive skeletonization factorization

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

- Old algorithm (RS) in new factorization form


## RSF in 2D: level 0

|  |
| :---: |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

domain

## RSF in 2D: level 1


domain
matrix

## RSF in 2D: level 2



## RSF in 2D: level 3



RSF in 3D: level 0


## RSF in 3D: level 1


domain

## RSF in 3D: level 2


domain

## RSF analysis

- Skeletonization operators:

$$
\begin{gathered}
U_{\ell}=\prod_{c \in C_{\ell}} Q_{c} R_{\check{c}}, \quad V_{\ell}=\prod_{c \in C_{\ell}} Q_{c} S_{\check{c}} \\
Q_{c}=\left[\begin{array}{lll}
1 & & \\
* & 1 & \\
& & I
\end{array}\right], \quad R_{\check{c}}, S_{\check{c}}=\left[\begin{array}{lll}
1 & * & \\
& 1 & \\
& & I
\end{array}\right]
\end{gathered}
$$

- Block diagonalization:

$$
D \approx U_{L-1}^{*} \cdots U_{0}^{*} A V_{0} \cdots V_{L-1}
$$

- Generalized LU decomposition:

$$
\begin{aligned}
A & \approx U_{0}^{-*} \cdots U_{L-1}^{-*} D V_{L-1}^{-1} \cdots V_{0}^{-1} \\
A^{-1} & \approx V_{0} \cdots V_{L-1} D^{-1} U_{L}^{*} \cdots U_{0}^{*}
\end{aligned}
$$

- Fast direct solver or preconditioner


## RSF analysis

The cost is determined by the skeleton size.

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

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 $\ell=0,1,2, \ldots, L$ from finest to coarsest do
Let $C_{\ell}$ be the set of all cells on level $\ell$.
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 $\ell$.
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

## HIF-IE in 2D: level $1 / 2$


domain
matrix

## HIF-IE in 2D: level 1


domain
matrix

## HIF-IE in 2D: level $3 / 2$




## HIF-IE in 2D: level 2



HIF-IE in 2D: level $5 / 2$


## HIF-IE in 2D: level 3

0 00
0

0
0000
domain
0
0
-

a

matrix



## 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 $C_{\ell}$ be the set of all cells on level $\ell$.
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 $\ell$.
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 $\ell$.
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



HIF-IE in 3D: level $1 / 3$


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:

$$
\begin{aligned}
A & \approx U_{0}^{-*} U_{1 / 2}^{-*} \cdots U_{L-1 / 2}^{-*} D V_{L-1 / 2}^{-1} \cdots V_{1 / 2}^{-1} V_{0}^{-1} \\
A^{-1} & \approx V_{0} V_{1 / 2} \cdots V_{L-1 / 2} D^{-1} U_{L-1 / 2}^{*} \cdots U_{1 / 2}^{*} U_{0}^{*}
\end{aligned}
$$

- 3D:

$$
\begin{aligned}
A & \approx U_{0}^{-*} U_{1 / 3}^{-*} U_{2 / 3}^{-*} \cdots U_{L-1 / 3}^{-*} D V_{L-1 / 3}^{-1} \cdots V_{2 / 3}^{-1} V_{1 / 3}^{-1} V_{0}^{-1} \\
A^{-1} & \approx V_{0} V_{1 / 3} V_{2 / 3} \cdots V_{L-1 / 3} D^{-1} U_{L-1 / 3}^{*} \cdots U_{2 / 3}^{*} U_{1 / 3}^{*} U_{0}^{*}
\end{aligned}
$$

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

## 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 ( $\square$ ), memory ( $\diamond$ )
- Precision $\epsilon=10^{-6}$

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 ( $\square$ ), memory ( $\diamond$ )
- Precision $\epsilon=10^{-3}$

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 ( $\square$ ), memory ( $\diamond$ )
- Precision $\epsilon=10^{-3}$


## 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 upscaling
- Extensions: PDEs, $A^{1 / 2}, \log \operatorname{det} A, \operatorname{diag} A^{-1}$
- 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/.

## Proxy compression

- Main cost of algorithm is computing IDs
- Global operation can be reduced to local operation using Green's theorem
- Suffices to compress against neighbors plus "proxy" surface
- Crucial for beating $O\left(N^{2}\right)$ 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:

|  | 2 D | 3 D |
| :--- | :---: | :---: |
| Factorization cost | $O(N \log N)$ | $O\left(N \log ^{6} N\right)$ |
| Solve cost | $O(N \log \log N)$ | $O\left(N \log ^{2} N\right)$ |

## Second-kind results in 2D

Second-kind volume IE on the unit square with

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



- rskelf2 (white), hifie2 (gray), hifie2x (black)
- Factorization time (○), solve time ( $\square$ ), memory ( $\diamond$ )
- Precision $\epsilon=10^{-6}$


## Second-kind results in 3D

Second-kind volume IE on the unit cube with

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



- rskelf3 (white), hifie3 (gray), hifie3x (black)
- Factorization time (○), solve time ( $\square$ ), memory ( $\diamond$ )
- Precision $\epsilon=10^{-3}$

