An Introduction to Hierarchical Poincaré-Steklov (HPS) Solvers

Anna Yesypenko

Oden Institute for Computational Engineering and Sciences
The University of Texas at Austin

Problem Addressed

This talk concerns numerical methods for boundary value problems of the form

$$\begin{cases} \mathcal{A}u(\boldsymbol{x}) = g(\boldsymbol{x}), & \boldsymbol{x} \in \Omega, \\ \mathcal{B}u(\boldsymbol{x}) = f(\boldsymbol{x}), & \boldsymbol{x} \in \Gamma, \end{cases} \tag{BVP}$$

where Ω is a domain (2D or 3D) with boundary Γ, and where $\mathcal{A}$ is a linear elliptic differential operator; possibly with variable coefficients.

Examples of problems we are interested in:

- The equations of linear elasticity.
- Stokes' equation.
- Helmholtz' equation (at least at low and intermediate frequencies).
- Time-harmonic Maxwell (at least at low and intermediate frequencies).

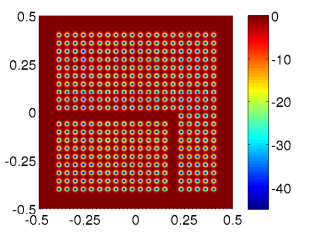
Helmholtz equation governs electomagnetic and acoustic scattering

In a varying medium Ω, the Helmholtz equation takes the form

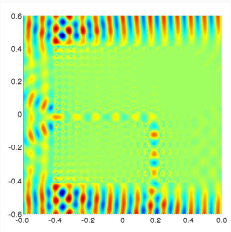
$$-\Delta u(x) - \kappa^2(1 - b(x))u(x) = f(x), \qquad x \in \Omega$$

with appropriate boundary conditions.

κ is roughly # wavelengths in unit domain and $b(x) \leq 1$ describes how the medium varies.



$-\kappa^2(1 - b(x))$



$u(x)$

Example: photonic crystal in free space.
The distance between crystals is set to filter certain light frequencies.

Image credit: Gillman, Martinsson, Barnett. *BIT* 2014.

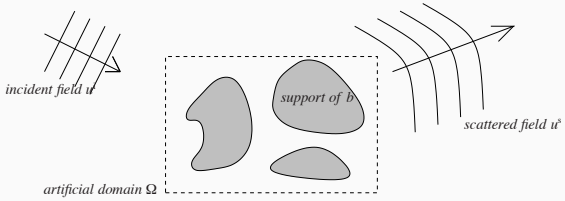


Image credit: Gillman, Martinsson, Barnett.

Standard numerical recipe for (BVP)

- Truncate the domain using absorbing boundary conditions.
- Discretize using FD/FEM.
- Solve the sparse system $\mathbf{Au} = \mathbf{f}$ with an iterative solver (and possibly a preconditioner).

Challenge 1: Discretization

Large systems $\mathbf{Au} = \mathbf{f}$

- Typically discretize to ~ 10 points per wavelength.
- As wavenumber κ increases, $N \sim \kappa^d$, $d = 2, 3$.

Pollution effect for large κ

- For low order discretization, we often need *more* points per wavelength as κ increases to avoid phase errors.
- Counteract pollution using local high order discretization.

Challenge 2: Iterative Solvers Struggle

- Because **A** is indefinite and ill-conditioned, a preconditioner is needed.

$$\underbrace{-\Delta u(x)}_{\substack{\text{positive} \\ \text{eigenvalues}}} - \underbrace{\kappa^2 u(x)}_{\text{negative shift}}$$

  Preconditioners are often problem-specific.

- For strong scattering (e.g. multiple reflections, cavities, trapped rays), no efficient preconditioners are known to exist, and number of iterations may scale with κ.

This workshop:

- Focus on boundary integral formulations on complicated interfaces.
- For variable coefficients PDEs, can formulate a volume integral equation.*

Hierarchical Poincaré Steklov scheme:

- Discretize the (BVP) on the volume with *local* high order discretization.
- Build an *approximate* direct solver that is fast to build and apply.

---

* A. Gopal, P.G. Martinsson. "An accelerated, high-order accurate direct solver for the Lippmann-Schwinger equation for acoustic scattering in the plane." *ACOM* 2022.

Outline

- Review classical algorithms for sparse direct solvers and their complexity.
- HPS discretization scheme.
- Applications of HPS, including BIE coupling for exterior scattering problems and time-stepping.
- Ongoing efforts to attain linear complexity solvers in 3D.

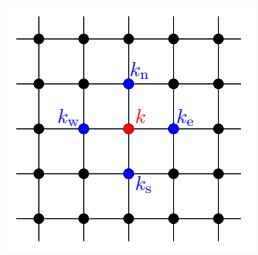
Model Problem

Let $\Omega = [0,1]^2$ and $\Gamma = \partial\Omega$. We seek to solve

$$\begin{cases} -\Delta u(\boldsymbol{x}) = g(\boldsymbol{x}), & \boldsymbol{x} \in \Omega, \\ u(\boldsymbol{x}) = f(\boldsymbol{x}), & \boldsymbol{x} \in \Gamma, \end{cases}$$

We introduce an $n \times n$ grid on Ω with nodes $\{\boldsymbol{x}_k\}_{k=1}^{N}$ where $N = n^2$ and discretize using the five point stencil. We need to solve the sparse system

$$\underset{N\times N}{\mathbf{A}}\ \mathbf{u} \ = \ \mathbf{b}$$



$h = \frac{1}{n+1}$

where $\mathbf{u}_k \approx u(\boldsymbol{x}_k)$

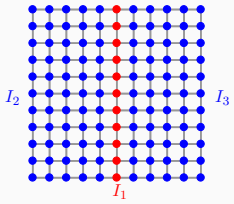
$$[\mathbf{Au}]_k = \frac{1}{h^2}(4\mathbf{u}_k - \mathbf{u}_{k_s} - \mathbf{u}_{k_e} - \mathbf{u}_{k_n} - \mathbf{u}(k_w)).$$

Divide and Conquer

To compute a sparse factorization

$$\underset{N\times N}{\mathbf{A}} = \mathbf{L}\,\mathbf{U},$$

we start by splitting the nodes $I = [I_1, I_2, I_3]$.



The resulting factorization then preserves some sparsity of $\mathbf{A}$

$$\mathbf{A} = \left[\begin{array}{c|c|c} \mathbf{A}_{33} & & \mathbf{A}_{31} \\ \hline & \mathbf{A}_{22} & \mathbf{A}_{21} \\ \hline \mathbf{A}_{13} & \mathbf{A}_{12} & \mathbf{A}_{11} \end{array}\right] = \left[\begin{array}{c|c|c} \mathbf{L}_{33} & & \\ \hline & \mathbf{L}_{22} & \\ \hline \mathbf{L}_{13} & \mathbf{L}_{12} & \mathbf{L}_{11} \end{array}\right] \left[\begin{array}{c|c|c} \mathbf{U}_{33} & & \mathbf{U}_{31} \\ \hline & \mathbf{U}_{22} & \mathbf{U}_{21} \\ \hline & & \mathbf{U}_{11} \end{array}\right],$$

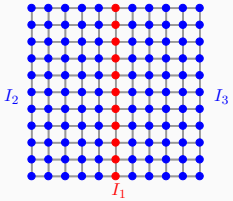
however, the Schur complement on I_1 is **dense**

$$\underset{n\times n}{\mathbf{L}_{11}}\ \underset{n\times n}{\mathbf{U}_{11}} = \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21} - \mathbf{A}_{13}\mathbf{A}_{33}^{-1}\mathbf{A}_{31}.$$

Divide and Conquer

To factorize $\mathbf{A}$, we do the following:

$$\mathbf{A} = \left[\begin{array}{c|c|c} \mathbf{A}_{33} & & \mathbf{A}_{31} \\ \hline & \mathbf{A}_{22} & \mathbf{A}_{21} \\ \hline \mathbf{A}_{13} & \mathbf{A}_{12} & \mathbf{A}_{11} \end{array}\right]$$



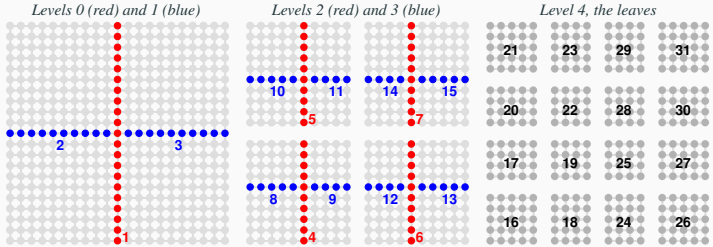
- Factorize *sparse* system $\mathbf{L}_{22}\mathbf{U}_{22}$ for volume I_2. size $\sim N/2 \times N/2$
- Factorize *sparse* system $\mathbf{L}_{33}\mathbf{U}_{33}$ for volume I_3. size $\sim N/2 \times N/2$
- Factorize **dense** system $\mathbf{T}_{11}$ on interface I_1. size $\sim \sqrt{N} \times \sqrt{N}$

$$\mathbf{T}_{11} = \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21} - \mathbf{A}_{13}\mathbf{A}_{33}^{-1}\mathbf{A}_{31}.$$

Note the obvious recursion!

Algorithm with Multiple Fronts

Nested Dissection Ordering



- The factorization is constructed using an upwards tree traversal.
- First, form sparse direct solvers for the "leaves".
- Then, factorize resulting dense Schur complements at each level of the tree.

We convert a *sparse* system on a volume into a **dense** system to solve on interfaces.

Nested Dissection Orderings

Typically, the ordering is more complicated.

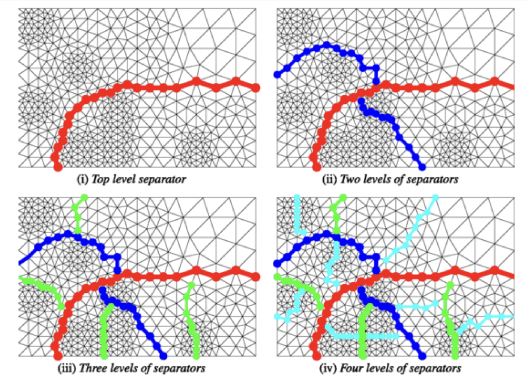
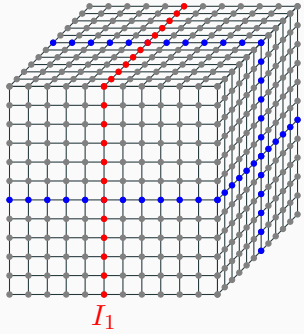


Image credit: Jianlin Xia, "Robust and Efficient Multifrontal Solver for Large Discretized PDEs," 2012.

Nested Dissection in 3D



In 3D, the algorithm transforms a *sparse* matrix on *3D volume* into **dense** matrix on **2D separators**.

The separator I_1 is a surface of size $\sim N^{2/3} \times N^{2/3}$.

We summarize the costs for $d = 2, 3$:

d	cost to compute	memory for solver
2	$O(N^{1.5})$	$O(N \log N)$
3	$O(N^2)$	$O(N^{4/3})$

Dimensionality Reduction as Workshop Theme

Key point: When faced with a BVP for a 3D volume, it is often beneficial to instead solve a **dense** problem on the **surface**.

- Constant-coefficient PDE: Reformulate as an integral equation on the boundary.
- Variable-coefficient PDE: Discretize the PDE, then compute a sparse direct solver involving dense Schur complements on surfaces.

Sparse Direct Solvers

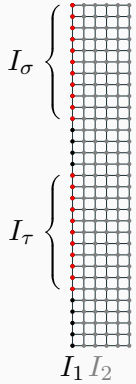
An *algebraic* means of reformulating a volume PDE to a *dense* operator on a surface.

Physical interpretation of Schur complement

Consider applying Dirichlet data $\mathbf{g}_1$ to the Schur complement:

$$\mathbf{T}_{11}\,\mathbf{g}_1 = \mathbf{A}_{11}\,\mathbf{g}_1 - \mathbf{A}_{12}\,\underbrace{\mathbf{A}_{22}^{-1}\,\mathbf{A}_{21}\,\mathbf{g}_1}_{\substack{\text{solution } \mathbf{u}_2 \\ \text{in volume}}}.$$

The matrix $\mathbf{T}_{11}$ can be viewed as a Dirichlet-to-Neumann map, also known as a Poincaré-Steklov operator.

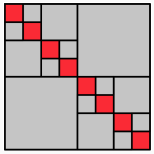


Rank Structure

The matrix $\mathbf{T}_{11}$ is compressible an HSS/HBS matrix.
For a dense matrix of size $n \times n$:

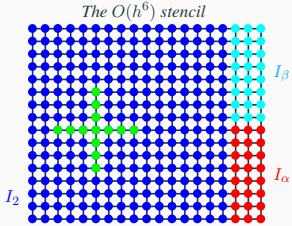
Matrix-vector multiply: $\mathcal{O}(n^2) \searrow \mathcal{O}(n)$

LU factorization: $\mathcal{O}(n^3) \searrow \mathcal{O}(n)$

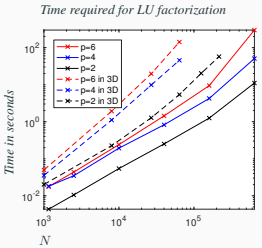


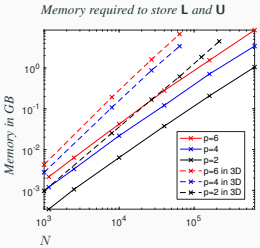
High order discretization and sparse direct solvers

- Need high order discretizations to resolve the PDE, especially for scattering problems.
- Need direct solvers to resolve ill-conditioned systems.
- **Issue**: Traditional high order discretizations (e.g. 4th or 6th order finite difference) cause performance of multifrontal solvers to plummet.



With increasing p, the separators grow wider.





As p increases, factorization and memory costs increase.

Hierarchical Poincaré-Steklov schemes

Instead of discretizing the domain Ω, we partition into subdomains $\Omega = \dot{\bigcup}_i \Omega_i$, then use a multidomain discretization.

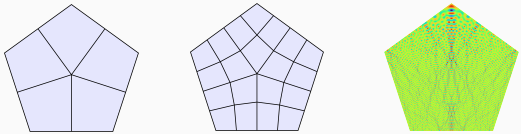


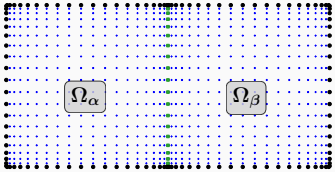
Image credit: Fortunato, Hale, Townsend.
"Ultraspherical spectral element method," *JCP* 2021.

On each subdomain Ω_i, enforce the PDE on subdomain interior.
Ensure that the computed solutions are consistent by enforcing:

- Continuity of solution across boundaries.
- Continuity of normal derivative across boundaries.

Related work: domain decomposition, non-overlapping Schwarz.

HPS for Two Subdomains



Consider the discretization of $\Omega = \Omega_\alpha \cup \Omega_\beta$ with labeling of the nodes $I = I_1 \cup I_2 \cup I_3$ so that

I_1 : shared boundary, where continuity of $\partial u/\partial x$ enforced

I_2 : collocation of (BVP) on Ω_α

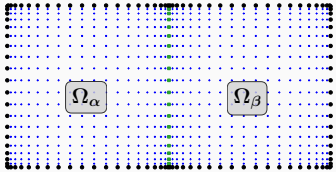
I_3 : collocation of (BVP) on Ω_β

The discretized system is

$$\left[\begin{array}{ccc} \mathbf{A}_{33} & & \mathbf{A}_{31} \\ \hline & \mathbf{A}_{22} & \mathbf{A}_{21} \\ \hline \mathbf{D}_{13}^{\alpha} & -\mathbf{D}_{12}^{\beta} & \mathbf{N} \end{array}\right] \left[\begin{array}{c} \mathbf{u}_3 \\ \hline \mathbf{u}_2 \\ \hline \mathbf{u}_1 \end{array}\right] = \left[\begin{array}{c} \mathbf{f}_3 \\ \hline \mathbf{f}_2 \\ \hline \mathbf{f}_1 \end{array}\right],$$

where $\mathbf{N} = \mathbf{D}_{11}^{\alpha} - \mathbf{D}_{11}^{\beta}$. The last equation enforces continuity of $\partial u/\partial x$ on I_1.

HPS for Two Subdomains



Consider the discretization of $\Omega = \Omega_\alpha \cup \Omega_\beta$ with labeling of the nodes $I = I_1 \cup I_2 \cup I_3$ so that

I_1 : shared boundary, where continuity of $\partial u/\partial x$ enforced

I_2 : collocation of (BVP) on Ω_α

I_3 : collocation of (BVP) on Ω_β

Factorizing the system to "eliminate" the subdomain interiors leads to a modified system on the interface:

$$\begin{bmatrix} \mathbf{A}_{33} & & \mathbf{A}_{31} \\ & \mathbf{A}_{22} & \mathbf{A}_{21} \\ \mathbf{D}_{13}^{\alpha} & -\mathbf{D}_{12}^{\beta} & \mathbf{N} \end{bmatrix} = \mathbf{L} \begin{bmatrix} \mathbf{A}_{33} & & \\ & \mathbf{A}_{22} & \\ & & \mathbf{T}_{11} \end{bmatrix} \mathbf{U},$$

where

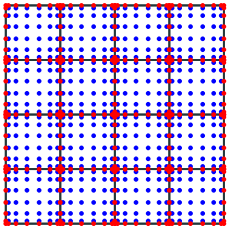
$$\mathbf{T}_{11} = \mathbf{N} - \underbrace{\mathbf{D}_{12}^{\alpha}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}}_{\mathrm{DtN}_\alpha} + \underbrace{\mathbf{D}_{13}^{\beta}\mathbf{A}_{33}^{-1}\mathbf{A}_{31}}_{\mathrm{DtN}_\beta}.$$

is a linear combination of DtN maps.

Model Problem for HPS

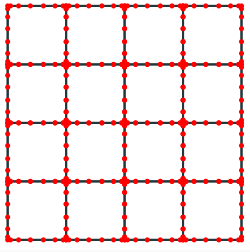
Consider a model problem on a square domain $\Omega = [0,1]^2$.

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega, \\ u(\boldsymbol{x}) = f(\boldsymbol{x}), & \boldsymbol{x} \in \Gamma, \end{cases}$$



Use spectral differentiation to discretize PDE on each patch and to enforce continuity of $\partial u/\partial x$ on interfaces.

$\Rightarrow$



Eliminating points in patch interiors leads to modified system on interfaces, involving Poincaré-Steklov operators.

To solve $\mathbf{Au} = \mathbf{f}$

Step 1: Compute equivalent body load $\tilde{\mathbf{f}}$ on interfaces.

Step 2: Solve *sparse* system $\mathbf{T}\tilde{\mathbf{u}} = \tilde{\mathbf{f}}$ for the solution on the interfaces using direct solver.

Step 3: Solve [locally] for the solution on patch interior, given the solution on the interface.

Complexity Scaling in p

Using dense spectral differentiation on the leaves, the cost to *build* a solution operator:

2D	$\sim \mathcal{O}(p^4 N + N^{3/2})$	with spectral differentiation*
3D	$\sim \mathcal{O}(p^6 N + N^2)$	with spectral differentiation.

P.G. Martinsson, "Fast Direct Solvers for Elliptic PDEs," SIAM 2019.

- Increasing p leads to increase in *linear*-scaling factor, but does not affect the *pre-factor* cost of computing a direct solver.
- Cost of direct solver can be reduced using $\mathcal{H}$-matrix algebra.

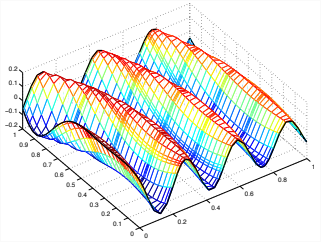
Using ultraspherical polynomials (instead of Chebyshev polynomials) for leaf discretization reduces complexity to $\mathcal{O}(p^2 N + N^{3/2})$ in 2D. See "Ultraspherical Method," 2021.

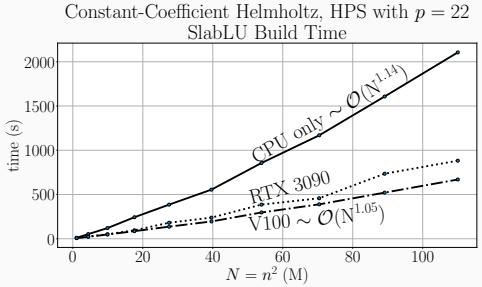
Benchmark problem

On a square domain $\Omega = [0,1]^2$, we scale κ as N increases to maintain 10 ppw:

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega, \\ u(\boldsymbol{x}) = f(\boldsymbol{x}), & \boldsymbol{x} \in \Gamma, \end{cases}$$

Let us consider how the discretization performs on the benchmark problem with prescribed Dirichlet data f as the restriction of wave from a point source $\boldsymbol{x} \to Y_0(\kappa|\boldsymbol{x} - \hat{\boldsymbol{x}}|)$.





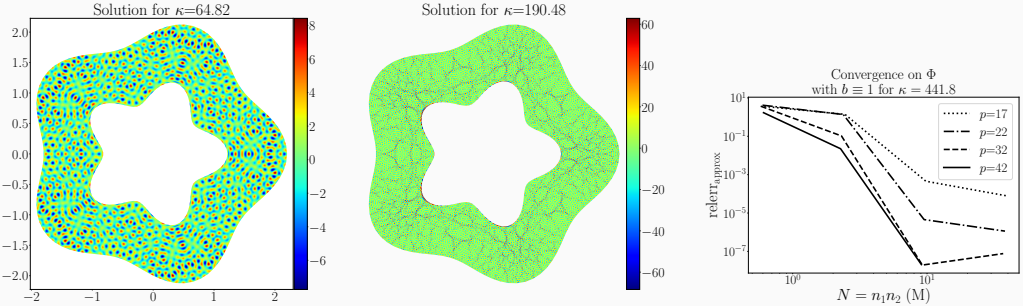
N	κ	M_{build}	T_{solve}	$\text{relerr}_{\text{true}}$
1.1 M	630.3	0.3 GB	1.0 s	2.4e-08
4.4 M	1258.6	1.8 GB	3.8 s	9.5e-08
17.6 M	2515.3	8.9 GB	13.9 s	4.0e-07
⋮	⋮	⋮	⋮	
110.0 M	6285.2	80.0 GB	83.9 s	7.5e-07
	~ 10 ppw			

Sparsity alone (and some GPU acceleration) is enough to scale to $1000\lambda \times 1000\lambda$ on a gaming desktop, with 7 digits of accuracy compared to known solution.

Yesypenko, Martinsson. "SlabLU: A Two-Level Sparse Direct Solver for Elliptic PDEs," *ACOM*, 2024.

High frequency Helmholtz in curved domain

The flexibility of a multi-domain scheme allows us to handle complicated geometries.



Interior Dirichlet problem with $\partial\Omega \equiv 1$.

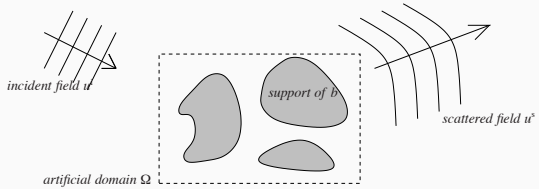
Choosing higher p helps avoid numerical pollution and achieve higher accuracy for fixed number of points.

Yesypenko, Martinsson. "SlabLU: A Two-Level Sparse Direct Solver for Elliptic PDEs."

Fast Direct Solvers and BIE Coupling

Consider the free-space acoustic scattering problem

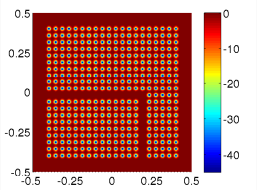
$$\begin{cases} -\Delta u^s(\boldsymbol{x}) - \kappa^2(1 - b(\boldsymbol{x}))u^s(\boldsymbol{x}) = -\kappa^2 b(\boldsymbol{x})u^i(\boldsymbol{x}), & \boldsymbol{x} \in \Omega, \\ \lim\limits_{|\boldsymbol{x}|\to\infty} \left(\partial_{|\boldsymbol{x}|} u^s(\boldsymbol{x}) - i\kappa u^s(\boldsymbol{x})\right) = 0 \end{cases}$$

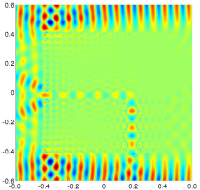


$b(x)$ is a smooth scattering potential with compact support
$u^i(x)$ is a given 'incoming potential'
$u^s(x)$ is the sought 'outgoing potential'

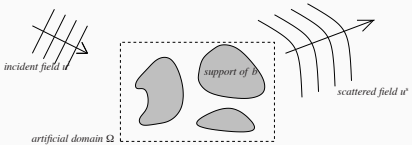
Gillman, Martinsson, Barnett. "A spectrally accurate direct solution technique for frequency-domain scattering problems with variable media," BIT 2015.

Impendence to Impedence operators





Left: Field $-\kappa^2(1 - b(\boldsymbol{x}))$
Right: Solution $u(\boldsymbol{x})$



- In the interior, discretize volume PDE.
- On the exterior, use BIE representation, with panel-based quadrature.
- Glue operators together at the interface $\partial\Omega$ using $\mathcal{H}$-matrix algebra.
- Instead of using Poincaré-Steklov operators, this work uses impedence-to-impedence maps, which avoid internal resonances.

Parabolic PDEs and HPS

The ability to reuse precomputed solution operators allows for efficient implicit time-stepping for parabolic PDEs. For a variable-coefficient operator $\mathcal{L}$, let

$$\frac{\partial u}{\partial t} = \mathcal{L}u \qquad u \in \Omega \times [0, T]$$

Define time step Δt and time points $t_n = n\Delta t$. Let u^n be the solution at time step t_n. Discretizing in time using backward Euler yields a steady-state PDE in u^{n+1}

$$(I - \Delta t\mathcal{L})u^{n+1} = u^n,$$

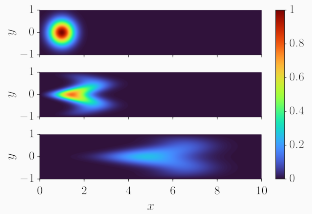
where the equation above much be solved to compute u^{n+1} given u^n. A direct solver can be computed and reused for *multiple right hand sides*.

HPS with time stepping schemes

Backward Euler (first order)

Time-dependent reaction-diffusion equation

$$\frac{\partial u}{\partial t} = \kappa \nabla^2 u - \nabla \cdot (\boldsymbol{b}(x,y)u)$$

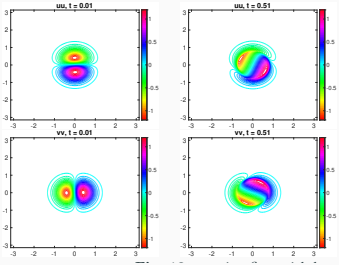


Fortunato, Hale, Townsend. "Ultraspherical spectral element method," *JCP* 2021.

Runge-Kutte (high order)

Regularized non-linear Schrodinger equation

$$i\frac{\partial u}{\partial t} + (1 - ia\epsilon)\Delta u + (1 + ic\epsilon)|u|^2 u = ib\epsilon u$$



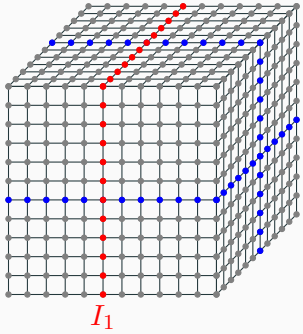
Ke Chen et al. "Fast and high-order approximation of parabolic equations," *arXiV* 2023.

Hierarchical Poincaré Steklov schemes

- Combines high order discretization with sparse direct solvers to solve elliptic PDEs to high accuracy.
- Particularly compelling for indefinite PDEs for combating the effect of pollution *and* solving the discretized system to high accuracy.
- Useful in applications to scattering and solving parabolic PDEs.

Finally, we discuss rank structure related to sparse direct solvers and how it can be used to build a fast (linear scaling for a constant PDE) solver.

Motivation for Using Rank Structure



Though we have reduced the dimensionality from a 3D volume to a 2D surface, the complexity costs for a direct solver are still prohibitively high in 3D.

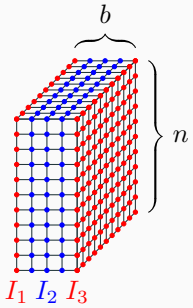
d	cost to compute	memory to store
2	$O(N^{1.5})$	$O(N \log N)$
3	$O(N^2)$	$O(N^{4/3})$

Limitations of 3D Sparse Direct Solvers

There is a large literature of fast direct solvers in 2D*, but 3D remains a challenge.

Memory consumption is a particular concern.

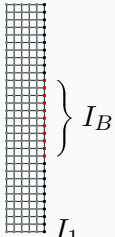
- For $N = 100$M in 2D, sub-block $\mathbf{T}_{11}$ needs <1GB to store.
- For $N = 100$M in 3D, sub-block $\mathbf{T}_{11}$ needs >300 GB to store.

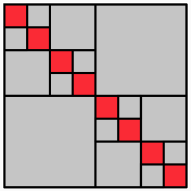


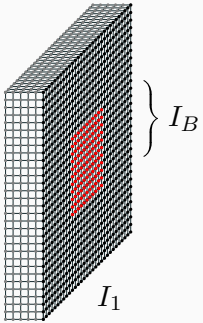
$$\mathbf{T}_{11} = \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}$$

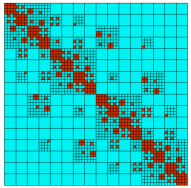
* Gillman, Martinsson. "A Direct Solver with $O(N)$ Complexity for Variable Coefficient Elliptic PDEs," *SIAM* 2014.
* Gillman, Martinsson, Barnett. "A spectrally accurate direct solution technique for frequency-domain scattering problems," *BIT* 2015.

$\mathcal{H}$-matrix structure in $\mathbf{T}_{11}$









$\mathcal{H}^2$-matrix structures on interfaces in 3D need more complicated data structure to achieve small memory footprint.

Problem Setting

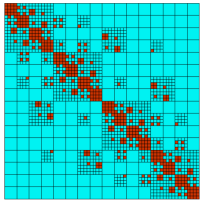


Image credit:
Ambikasaran & Darve, arxiv.org #1407.1572

$\mathcal{H}^2$-matrices with a *strong admissibility* condition are for achieving linear scaling direct solvers. For a dense matrix of size $m \times m$:

Matrix-vector product: $\mathcal{O}(m^2) \searrow \mathcal{O}(m)$

LU factorization: $\mathcal{O}(m^3) \searrow \mathcal{O}(m \text{ polylog}(m))$

In practice, there are challenges to using this format...

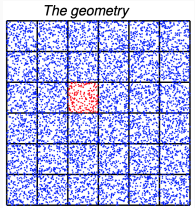
Challenge 1: Compression

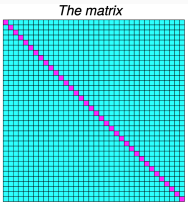
For many applications, a fast matrix-vector product $\mathbf{v} \to \mathbf{T}\mathbf{v}$ is available, but individual entries $\mathbf{T}_{ij}$ may be hard or complicated to access.

$$\mathbf{T}_{11} = \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}$$

Challenge 2: Computing $\mathbf{T}^{-1}$

Weak admissibility

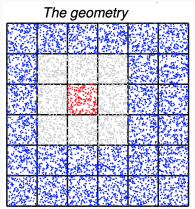


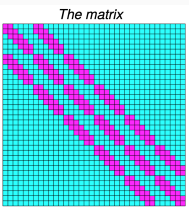


Neighbor blocks are compressed as low rank, leading to simpler algebraic structure.

If you can represent $\mathbf{T}$, algebraic algorithms for a factorization $\mathbf{T}^{-1}$ are available.

Strong admissibility (focus for 3D solvers)





Neighbor blocks are stored densely.

Even if you can represent $\mathbf{T}$, it is not so simple to construct $\mathbf{T}^{-1}$ (often recompression is needed).

Addressing Challenges

Randomized Sketching

For a *globally low rank* matrix $\mathbf{B} \in \mathbb{R}^{m \times n}$ with rank k, a factorization can be reconstructed matrix-free from sketches

$$\underset{m\times k}{\mathbf{Y}} = \underset{m\times n}{\mathbf{B}}\ \underset{n\times k}{\mathbf{\Omega}}, \qquad \underset{n\times k}{\mathbf{Z}} = \underset{n\times m}{\mathbf{B}^*}\ \underset{m\times k}{\mathbf{\Psi}},$$

where $\mathbf{\Omega}, \mathbf{\Psi}$ are Gaussian random matrices.

Cheng, Gimbutas, Martinsson, Rohklin. "On the compression of low-rank matrices," SIAM 2005.

Linear algorithm for computing $\mathbf{T}^{-1}$

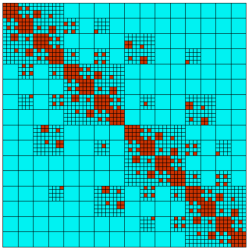
A promising approach is "strong recursive skeletonization," but the algorithm is *only* applicable when matrix entries are available.

Minden, Damle, Ho, Ying. "A recursive skeletonization factorization based on strong admissibility," SIAM 2017.

Sushnikova, Greengard, O'Neil, Rachh. "FMM-LU," SIAM 2023.

Randomized Strong Recursive Skeletonization (RSRS)

Often, a fast matrix-vector product $\mathbf{v} \to \mathbf{T}\mathbf{v}$ is available for $\mathbf{T}$ and its adjoint.



Suppose we are given $\{\mathbf{Y}, \mathbf{Z}, \mathbf{\Omega}, \mathbf{\Psi}\}$:

$$\underset{N\times p}{\mathbf{Y}} = \underset{N\times N}{\mathbf{T}}\ \underset{N\times p}{\mathbf{\Omega}}, \qquad \underset{N\times p}{\mathbf{Z}} = \underset{N\times N}{\mathbf{T}^*}\ \underset{N\times p}{\mathbf{\Psi}}$$

where $\mathbf{\Omega}, \mathbf{\Psi}$ are random matrices and $p \ll N$.

RSRS recovers the $\mathcal{H}^2$-matrix **and** computes an invertible factorization in-place. Requires *constant* number of samples and *linear* post-processing cost.

A. Yesypenko, P.G. Martinsson. arXiv preprint arXiv:2311.01451

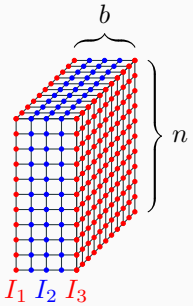
Extends on many previous works on randomized sampling for $\mathcal{H}^2$ matrices, notably the dissertation of James Levitt.

Numerical Experiments for 3D SlabLU

We compute $\mathbf{T}_{11}^{-1}$ for a Schur complement arising from a PDE discretization for 3D SlabLU*

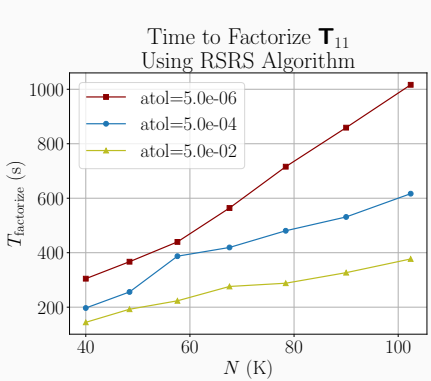
$$\mathbf{T}_{11} = \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}$$

- PDE is Poisson, discretized with 2nd order finite difference discretization.
- Slab width $b = 10$ fixed, which number of points on slab interface grows as $N = n^2$.
- User supplies compression tolerance 'atol.'

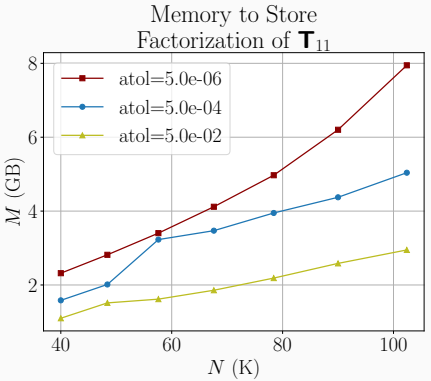


* Yesypenko, Martinsson. "SlabLU: A Two-Level Sparse Direct Solver for Elliptic PDEs." *ACOM*, 2024.

Linear-Time Construction

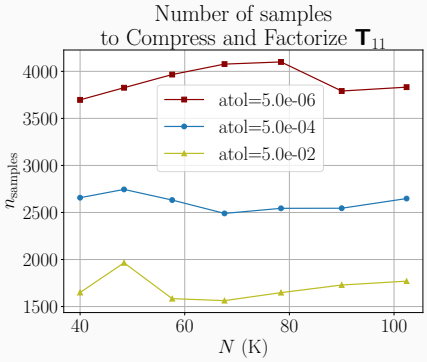


Time to compute $\mathbf{T}_{11}^{-1}$ scales linearly.

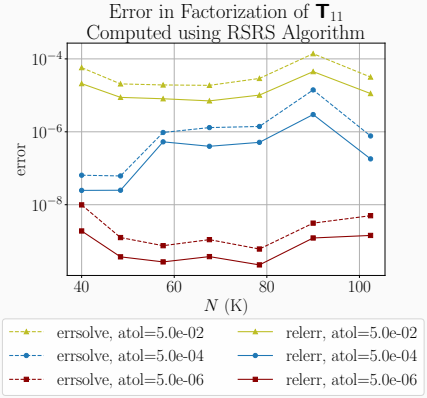


Memory to store $\mathbf{T}_{11}^{-1}$ scales linearly.

Constant Number of Samples



Number of samples needed is independent of N.



The computed factorization is accurate and does not deteriorate with increasing N.

Thank you! Concluding Remarks

The talk today featured:

- Review of sparse direct solvers for PDE discretizations.
- Domain decomposition techniques and efficient high order discretization.
- Applications of HPS to scattering and time-stepping.
- An overview of fast direct solvers which use rank-structure to achieve linear complexity.