

Fast direct solution techniques for elliptic partial differential equations

Adrianna Gillman

Dartmouth College

Collaborators: Per-Gunnar Martinsson (University of Colorado, Boulder)
Alex Barnett (Dartmouth College)

April 21, 2012

Definition of a “fast” method:

A numerical method is *fast* if its execution time scales as $O(N \log^k N)$ as the problem size N grows where $k = 0, 1$, or 2 .

Our goal is to develop methods whose complexity is $O(N)$.

Definition of a “direct solver:”

Given a computational tolerance ε , and a linear system

$$\mathbf{A} \mathbf{u} = \mathbf{b}, \quad (1)$$

a *direct solver* constructs an operator \mathbf{T} such that

$$\|\mathbf{A}^{-1} - \mathbf{T}\| \leq \varepsilon.$$

Then an approximate solution to (1) is obtained by simply evaluating

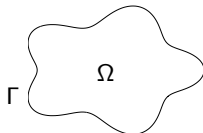
$$\mathbf{u}_{\text{approx}} = \mathbf{T} \mathbf{b}.$$

The matrix \mathbf{T} is typically constructed in a compressed format that allows the matrix-vector product $\mathbf{T} \mathbf{b}$ to be evaluated rapidly.

Variation: Find factors \mathbf{B} and \mathbf{C} such that $\|\mathbf{A} - \mathbf{B} \mathbf{C}\| \leq \varepsilon$, and linear solves involving the matrices \mathbf{B} and \mathbf{C} are fast.

Linear boundary value problems

We consider Laplace's equation with Dirichlet boundary condition:



$$\begin{cases} -\Delta u(\mathbf{x}) = 0, & \mathbf{x} \in \Omega, \\ u(\mathbf{x}) = f(\mathbf{x}), & \mathbf{x} \in \Gamma. \end{cases}$$

However, the solution techniques can be extended to linear boundary value problems of the form

$$\begin{cases} A u(\mathbf{x}) = g(\mathbf{x}), & \mathbf{x} \in \Omega, \\ B u(\mathbf{x}) = f(\mathbf{x}), & \mathbf{x} \in \Gamma, \end{cases} \quad (\text{BVP})$$

where Ω is a domain in \mathbb{R}^2 or \mathbb{R}^3 with boundary Γ . For instance:

- 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).

Discretization of linear boundary value problems



Direct discretization of the differential operator via Finite Elements, Finite Differences, ...



Very large $N \times N$ linear system that is sparse, and **ill-conditioned**.



Fast solvers:
iterative (multigrid), $O(N)$,
direct (nested dissection),
 $O(N^{3/2})$.



Conversion of the BVP to a Boundary Integral Operator (BIE).



Discretization of (BIE) using Nyström, collocation, BEM,



Dense $N \times N$ linear system that is moderate in size and (often) well-conditioned.



Iterative solver accelerated by fast matrix-vector multiplier, $O(N)$.

Discretization of linear boundary value problems



Direct discretization of the differential operator via Finite Elements, Finite Differences, ...



Very large $N \times N$ linear system that is sparse, and **ill-conditioned**.



Fast solvers:
iterative (multigrid), $O(N)$,
direct (nested dissection),
 $O(N^{3/2})$.
 $O(N)$ direct solvers.



Conversion of the BVP to a Boundary Integral Operator (BIE).



Discretization of (BIE) using Nyström, collocation, BEM,



Dense $N \times N$ linear system that is moderate in size and (often) well-conditioned.



Iterative solver accelerated by fast matrix-vector multiplier, $O(N)$.
 $O(N)$ direct solvers.

"Iterative" versus "direct" solvers

Two classes of methods for solving an $N \times N$ linear algebraic system

$$\mathbf{A} \mathbf{u} = \mathbf{b}.$$

Iterative methods:

Examples: GMRES, conjugate gradients, Gauss-Seidel, etc.

Construct a sequence of vectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \dots$ that converge to the solution.

Many iterative methods only need to know \mathbf{A} action on vectors.

Often require problem specific preconditioners.

In some cases, these are $O(N)$ solvers.

Direct methods:

Examples: Gaussian elimination, LU factorizations, matrix inversion, etc.

Deterministic. Always returns the solution.

Robust.

Great for multiple right hand sides.

Have often been considered too slow for high performance computing.

(Directly access elements of \mathbf{A} .)

(Exact except for rounding errors.)

Incomplete literature review — direct solvers based on data-sparsity:

- 1991 Data-sparse matrix algebra / wavelets, *Beylkin, Coifman, Rokhlin, et al*
- 1993 Fast inversion of 1D operators *V. Rokhlin and P. Starr*
- 1996 scattering problems, *E. Michielssen, A. Boag and W.C. Chew,*
- 1998 factorization of non-standard forms, *G. Beylkin, J. Dunn, D. Gines,*
- 1998 \mathcal{H} -matrix methods, *W. Hackbusch, B. Khoromskijet, S. Sauter, . . . ,*
- 2000 Cross approximation, matrix skeletons, etc., *E. Tyrtyshnikov.*
- 2002 $O(N^{3/2})$ inversion of Lippmann-Schwinger equations, *Y. Chen,*
- 2002 “Hierarchically Semi-Separable” matrices, *M. Gu, S. Chandrasekharan.*
- 2002 (1999?) \mathcal{H}^2 -matrix methods, *S. Börm, W. Hackbusch, B. Khoromskijet, S. Sauter.*
- 2004 Inversion of “FMM structure,” *S. Chandrasekharan, T. Pals.*
- 2004 Proofs of compressibility, *M. Bebendorf, S. Börm, W. Hackbusch,*
- 2006 Accelerated nested diss. via \mathcal{H} -mats, *L. Grasedyck, R. Kriemann, S. LeBorne*
[2007] *S. Chandrasekharan, M. Gu, X.S. Li, J. Xia.* [2010], *P. Schmitz and L. Ying.*
- 2010 construction of A^{-1} via randomized sampling, *L. Lin, J. Lu, L. Ying.*

Overview

- Linear inversion scheme
- One dimensional boundary integral equations
- Finite element solver
- Quasiperiodic scattering

Data sparse formats

Most “fast direct methods” exploit rank-deficiencies in the off-diagonal blocks of large dense matrices.

The most well-known formats are the \mathcal{H} -matrix and \mathcal{H}^2 -matrix formats of Hackbusch and co-workers.

Data sparse formats

Most “fast direct methods” exploit rank-deficiencies in the off-diagonal blocks of large dense matrices.

The most well-known formats are the \mathcal{H} -matrix and \mathcal{H}^2 -matrix formats of Hackbusch and co-workers.

This talk will describe methods based on the so called *Hierarchically Semi-Separable (HSS)* matrix format:

Data sparse formats

Most “fast direct methods” exploit rank-deficiencies in the off-diagonal blocks of large dense matrices.

The most well-known formats are the \mathcal{H} -matrix and \mathcal{H}^2 -matrix formats of Hackbusch and co-workers.

This talk will describe methods based on the so called *Hierarchically Semi-Separable (HSS)* matrix format:

- The HSS format is conceptually similar to \mathcal{H} -matrix and \mathcal{H}^2 -matrix formats in many ways: There is a tree structure on the index vector, a tessellation of the coefficient matrix, low-rank approximations to certain off-diagonal blocks, etc.
- Out of the box, the HSS format is more restrictive than the \mathcal{H} / \mathcal{H}^2 formats. However, when it works, it achieves very high performance in terms of both speed and accuracy.
- With certain modifications, the HSS format can be used for most problems that can be handled using \mathcal{H} / \mathcal{H}^2 matrices.

What does it mean for a matrix to be low rank?

Let M be an $m \times n$ matrix where $m \leq n$.

The **Singular Value Decomposition** (SVD) of M is a matrix factorization

$$M = U \Sigma V^*$$

where U and V are square unitary matrices and Σ is an $m \times n$ matrix with only positive real diagonal entries σ_j , $j = 1, \dots, m$.

What does it mean for a matrix to be low rank?

Let M be an $m \times n$ matrix where $m \leq n$.

The **Singular Value Decomposition** (SVD) of M is a matrix factorization

$$M = U \Sigma V^*$$

where U and V are square unitary matrices and Σ is an $m \times n$ matrix with only positive real diagonal entries σ_j , $j = 1, \dots, m$.

The values σ_j for $j = 1, \dots, m$ are called the **singular values**.

What does it mean for a matrix to be low rank?

Let M be an $m \times n$ matrix where $m \leq n$.

The **Singular Value Decomposition** (SVD) of M is a matrix factorization

$$M = U \Sigma V^*$$

where U and V are square unitary matrices and Σ is an $m \times n$ matrix with only positive real diagonal entries σ_j , $j = 1, \dots, m$.

The values σ_j for $j = 1, \dots, m$ are called the **singular values**.

The **ϵ -rank** of a matrix is the number k of singular values greater than ϵ .

A matrix is numerically low rank if $k \ll m$.

Intuition for inversion of Hierarchically Semi-Separable matrices

Consider a linear system

$$\mathbf{A} \mathbf{q} = \mathbf{f},$$

where \mathbf{A} is a “block-separable” matrix consisting of $p \times p$ blocks of size $n \times n$:

$$\mathbf{A} = \begin{bmatrix} \mathbf{D}_{11} & \mathbf{A}_{12} & \mathbf{A}_{13} & \mathbf{A}_{14} \\ \mathbf{A}_{21} & \mathbf{D}_{22} & \mathbf{A}_{23} & \mathbf{A}_{24} \\ \mathbf{A}_{31} & \mathbf{A}_{32} & \mathbf{D}_{33} & \mathbf{A}_{34} \\ \mathbf{A}_{41} & \mathbf{A}_{42} & \mathbf{A}_{43} & \mathbf{D}_{44} \end{bmatrix}. \quad (\text{Shown for } p = 4.)$$

Core assumption: Each off-diagonal block \mathbf{A}_{ij} admits the factorization

$$\begin{array}{ccccc} \mathbf{A}_{ij} & = & \mathbf{U}_i & \tilde{\mathbf{A}}_{ij} & \mathbf{V}_j^* \\ n \times n & & n \times k & k \times k & k \times n \end{array}$$

where the rank k is significantly smaller than the block size n . (Say $k \approx n/2$.)

The critical part of the assumption is that all off-diagonal blocks in the i 'th row use the same basis matrices \mathbf{U}_i for their column spaces (and analogously all blocks in the j 'th column use the same basis matrices \mathbf{V}_j for their row spaces).

Intuition for inversion of Hierarchically Semi-Separable matrices

We get $A = \begin{bmatrix} D_{11} & U_1 \tilde{A}_{12} V_2^* & U_1 \tilde{A}_{13} V_3^* & U_1 \tilde{A}_{14} V_4^* \\ U_2 \tilde{A}_{21} V_1^* & D_{22} & U_2 \tilde{A}_{23} V_3^* & U_2 \tilde{A}_{24} V_4^* \\ U_3 \tilde{A}_{31} V_1^* & U_3 \tilde{A}_{32} V_2^* & D_{33} & U_3 \tilde{A}_{34} V_4^* \\ U_4 \tilde{A}_{41} V_1^* & U_4 \tilde{A}_{42} V_2^* & U_4 \tilde{A}_{43} V_3^* & D_{44} \end{bmatrix}.$

Then A admits the factorization:

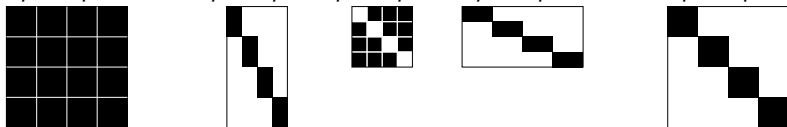
$$A = \underbrace{\begin{bmatrix} U_1 & & & \\ & U_2 & & \\ & & U_3 & \\ & & & U_4 \end{bmatrix}}_{=U} \underbrace{\begin{bmatrix} 0 & \tilde{A}_{12} & \tilde{A}_{13} & \tilde{A}_{14} \\ \tilde{A}_{21} & 0 & \tilde{A}_{23} & \tilde{A}_{24} \\ \tilde{A}_{31} & \tilde{A}_{32} & 0 & \tilde{A}_{34} \\ \tilde{A}_{41} & \tilde{A}_{42} & \tilde{A}_{43} & 0 \end{bmatrix}}_{=\tilde{A}} \underbrace{\begin{bmatrix} V_1^* & & & \\ & V_2^* & & \\ & & V_3^* & \\ & & & V_4^* \end{bmatrix}}_{=V^*} +$$

$$\underbrace{\begin{bmatrix} D_{11} & & & \\ & D_{22} & & \\ & & D_{33} & \\ & & & D_{44} \end{bmatrix}}_{=D},$$

Intuition for inversion of Hierarchically Semi-Separable matrices

$$\text{We get } A = \begin{bmatrix} D_{11} & U_1 \tilde{A}_{12} V_2^* & U_1 \tilde{A}_{13} V_3^* & U_1 \tilde{A}_{14} V_4^* \\ U_2 \tilde{A}_{21} V_1^* & D_{22} & U_2 \tilde{A}_{23} V_3^* & U_2 \tilde{A}_{24} V_4^* \\ U_3 \tilde{A}_{31} V_1^* & U_3 \tilde{A}_{32} V_2^* & D_{33} & U_3 \tilde{A}_{34} V_4^* \\ U_4 \tilde{A}_{41} V_1^* & U_4 \tilde{A}_{42} V_2^* & U_4 \tilde{A}_{43} V_3^* & D_{44} \end{bmatrix}.$$

Then A admits the factorization:

$$\begin{array}{c} A \\ p n \times p n \end{array} = \begin{array}{c} U \\ p n \times p k \end{array} \begin{array}{c} \tilde{A} \\ p k \times p k \end{array} \begin{array}{c} V^* \\ p k \times p n \end{array} + \begin{array}{c} D, \\ p n \times p n \end{array}$$


Intuition for inversion of Hierarchically Semi-Separable matrices

Lemma: [Variation of Woodbury] If an $N \times N$ matrix A admits the factorization

$$A = U \tilde{A} V^* + D,$$

then

$$A^{-1} = E (\tilde{A} + \hat{D})^{-1} F^* + G,$$

Diagram illustrating the matrix factorization and inversion:

- A^{-1} is an $pn \times pn$ matrix represented by a 4x4 grid of black squares.
- E is a $pn \times pk$ matrix represented by a 4x2 grid of black squares, with a staircase pattern of black squares along the diagonal.
- $(\tilde{A} + \hat{D})^{-1}$ is a $pk \times pk$ matrix represented by a 4x4 grid of black squares.
- F^* is a $pk \times pn$ matrix represented by a 4x4 grid of black squares, with a staircase pattern of black squares along the diagonal.
- G is a $pn \times pn$ matrix represented by a 4x4 grid of black squares, with a staircase pattern of black squares along the diagonal.

where (provided all intermediate matrices are invertible)

$$\hat{D} = (V^* D^{-1} U)^{-1}, \quad E = D^{-1} U \hat{D}, \quad F = (\hat{D} V^* D^{-1})^*, \quad G = D^{-1} - D^{-1} U \hat{D} V^* D^{-1}$$

Note: All matrices set in blue are block diagonal.

Intuition for inversion of Hierarchically Semi-Separable matrices

The Woodbury formula inverts a $p k \times p k$ matrix instead of a $p n \times p n$ matrix.

The cost is reduced from $(p n)^3$ to $(p k)^3$.

This is not “fast” yet.

But, \tilde{A} admits a compressed representation so we can create a telescoping factorization.

(Recall: A has $p \times p$ blocks, each of size $n \times n$ and of rank k .)

Intuition for inversion of Hierarchically Semi-Separable matrices

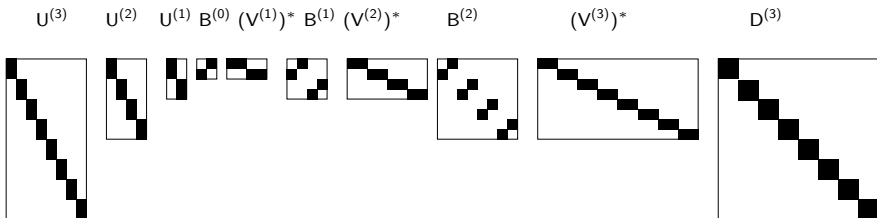
Using a telescoping factorization of A :

$$A = U^{(3)}(U^{(2)}(U^{(1)} B^{(0)} (V^{(1)})^* + B^{(1)})(V^{(2)})^* + B^{(2)})(V^{(3)})^* + D^{(3)},$$

we have a formula

$$A^{-1} = E^{(3)}(E^{(2)}(E^{(1)} \hat{D}^{(0)} F^{(1)} + \hat{D}^{(1)})(F^{(2)} + \hat{D}^{(2)})(V^{(3)})^* + \hat{D}^{(3)}).$$

Block structure of factorization:



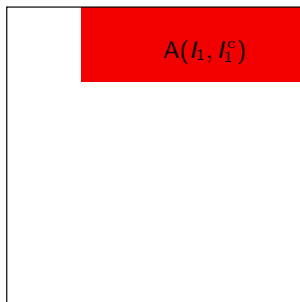
All matrices are now block diagonal except $\hat{D}^{(0)}$, which is small.

What is the role of the basis matrices U_i and V_j ?

Recall our toy example:

$$A = \begin{bmatrix} D_{11} & \mathbf{U}_1 \tilde{A}_{12} V_2^* & \mathbf{U}_1 \tilde{A}_{13} V_3^* & \mathbf{U}_1 \tilde{A}_{14} V_4^* \\ U_2 \tilde{A}_{21} V_1^* & D_{22} & U_2 \tilde{A}_{23} V_3^* & U_2 \tilde{A}_{24} V_4^* \\ U_3 \tilde{A}_{31} V_1^* & U_3 \tilde{A}_{32} V_2^* & D_{33} & U_3 \tilde{A}_{34} V_4^* \\ U_4 \tilde{A}_{41} V_1^* & U_4 \tilde{A}_{42} V_2^* & U_4 \tilde{A}_{43} V_3^* & D_{44} \end{bmatrix}.$$

We see that the columns of \mathbf{U}_1 must span the column space of the matrix $A(l_1, l_1^c)$ where l_1 is the index vector for the first block and $l_1^c = I \setminus l_1$.



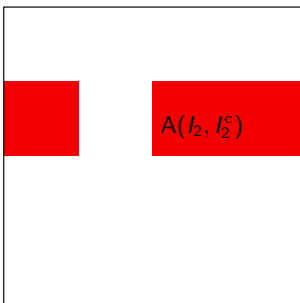
The matrix A

What is the role of the basis matrices U_i and V_j ?

Recall our toy example:

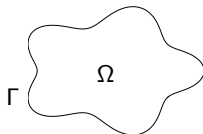
$$A = \begin{bmatrix} D_{11} & U_1 \tilde{A}_{12} V_2^* & U_1 \tilde{A}_{13} V_3^* & U_1 \tilde{A}_{14} V_4^* \\ \textcolor{red}{U}_2 \tilde{A}_{21} V_1^* & D_{22} & \textcolor{red}{U}_2 \tilde{A}_{23} V_3^* & \textcolor{red}{U}_2 \tilde{A}_{24} V_4^* \\ U_3 \tilde{A}_{31} V_1^* & U_3 \tilde{A}_{32} V_2^* & D_{33} & U_3 \tilde{A}_{34} V_4^* \\ U_4 \tilde{A}_{41} V_1^* & U_4 \tilde{A}_{42} V_2^* & U_4 \tilde{A}_{43} V_3^* & D_{44} \end{bmatrix}.$$

We see that the columns of $\textcolor{red}{U}_2$ must span the column space of the matrix $A(l_2, l_2^c)$ where l_2 is the index vector for the first block and $l_2^c = I \setminus l_2$.



The matrix A

Model Problem



Consider the problem

$$\begin{aligned} -\Delta u(\mathbf{x}) &= 0, & \mathbf{x} \in \Omega, \\ u(\mathbf{x}) &= g(\mathbf{x}), & \mathbf{x} \in \Gamma. \end{aligned}$$

The solution can be represented as a double layer potential

$$u(\mathbf{x}) = \int_{\Gamma} \partial_{\boldsymbol{\nu}} G(\mathbf{x}, \mathbf{y}) \phi(\mathbf{y}) ds(\mathbf{y}), \quad \mathbf{x} \in \Omega,$$

where $\boldsymbol{\nu}$ is the outward normal and $G(\mathbf{x}, \mathbf{y})$ is the fundamental solution

$$G(\mathbf{x}, \mathbf{y}) = -\frac{1}{2\pi} \log |\mathbf{x} - \mathbf{y}|.$$

Then the boundary charge distribution ϕ satisfies the boundary integral equation

$$\frac{1}{2} \phi(\mathbf{x}) + \int_{\Gamma} \partial_{\boldsymbol{\nu}} G(\mathbf{x}, \mathbf{y}) \phi(\mathbf{y}) ds(\mathbf{y}) = g(\mathbf{x})$$

How do you discretize integral equations?

Recall: An integral can be approximated via quadrature by

$$\int_a^b f(x) dx \sim \sum_{j=1}^N f(x_j) w_j$$

where $a \leq x_1 < \dots < x_N \leq b$ are called the quadrature nodes and $\{w_j\}_{j=1}^N$ are called the quadrature weights.

How do you discretize integral equations?

Plugging this into the integral equation, we get

$$\begin{aligned} g(\mathbf{x}) &= \frac{1}{2}\phi(\mathbf{x}) + \int_{\Gamma} \partial_{\nu} G(\mathbf{x}, \mathbf{y}) \phi(\mathbf{y}) ds(\mathbf{y}) \\ &\sim \frac{1}{2}\phi(\mathbf{x}) + \sum_{j=1}^N \partial_{\nu} G(\mathbf{x}, \mathbf{x}_j) \phi(\mathbf{x}_j) w_j \end{aligned}$$

Looking for the solution at the quadrature nodes leads to a linear system where the i^{th} row is given by

$$g(\mathbf{x}_i) = \frac{1}{2}\phi(\mathbf{x}_i) + \sum_{j=1}^N \partial_{\nu} G(\mathbf{x}_i, \mathbf{x}_j) \phi(\mathbf{x}_j) w_j$$

Model problem

So we want to solve

$$A\phi = \left(\frac{1}{2}I + D\right)\phi = \mathbf{g},$$

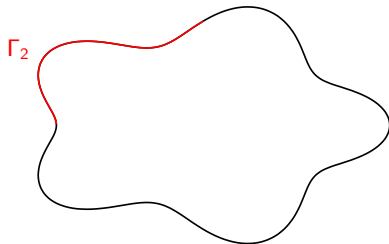
where D is a matrix that approximates the integral operator

$$\int_{\Gamma} \partial_{\nu} G(\mathbf{x}, \mathbf{y}) \phi(\mathbf{y}) ds(\mathbf{y}).$$

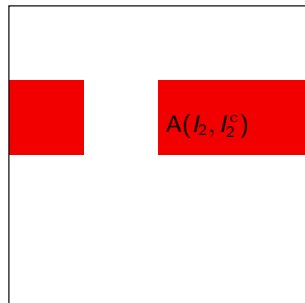
Properties of A :

- Dense matrix.
- Size is determined by the number of discretization points.
- Data-sparse/structured matrix.

Model problem



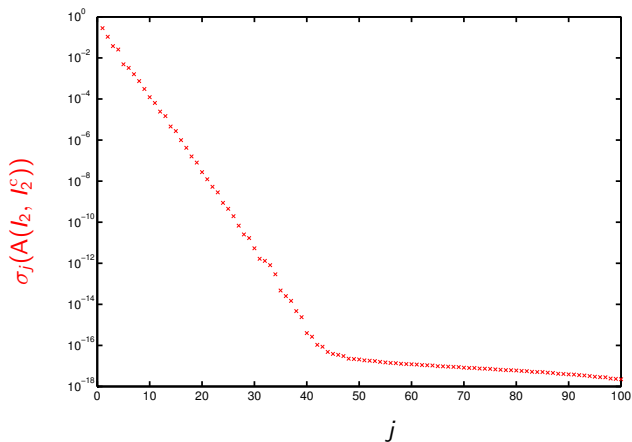
The contour Γ .



The matrix A .

Model problem

Singular values of $A(l_2, l_2^c)$

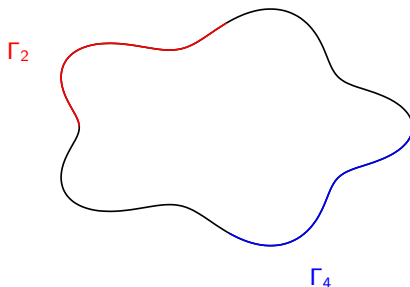


To precision 10^{-10} , the matrix $A(l_2, l_2^c)$ has rank 29.

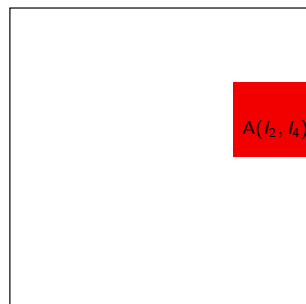
Model problem

Remark: In an HSS representation, the ranks are typically higher than in an \mathcal{H} -matrix representation.

Specifically, the block $A(l_2, l_2^c)$ would typically be considered “inadmissible.” Instead, in an \mathcal{H} -matrix representation, you would compress blocks such as



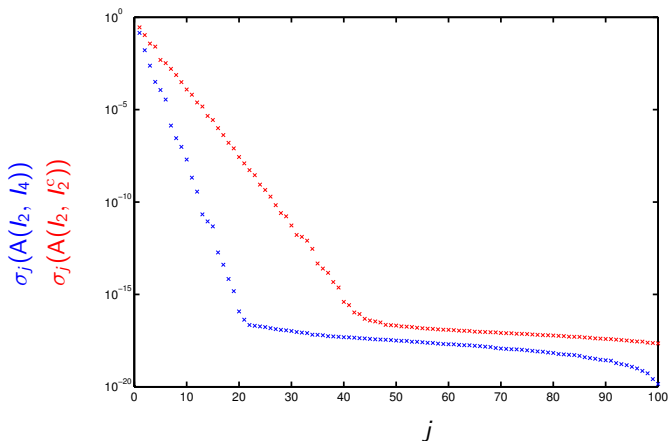
The contour Γ .



The matrix A .

Model problem

Singular values of $A(l_2, l_2^c)$ and $A(l_2, l_4)$:



To precision 10^{-10} , the matrix $A(l_2, l_2^c)$ has rank 29.

To precision 10^{-10} , the matrix $A(l_2, l_4)$ has rank 13.

Model problem

Choice of basis matrices (our approach is non-standard):

Recall: The HSS structure relies on factorizations such as (for $k < n$)

$$\begin{array}{ccccc} A_{\sigma,\tau} & = & U_{\sigma} & \tilde{A}_{\sigma,\tau} & V_{\tau}^* \\ n \times n & & n \times k & k \times k & k \times n \end{array}$$

For HSS matrix algebra to be numerically stable, it is critical that the basis matrices U_{τ} and V_{τ} be well-conditioned.

The gold-standard is to have U_{τ} and V_{τ} be orthonormal (i.e. $\sigma_j(U_{\tau}) = \sigma_j(V_{\tau}) = 1$ for $j = 1, 2, \dots, k$), and this is commonly enforced.

We have decided to instead use *interpolatory decompositions* in which:

1. U_{τ} and V_{τ} each contain the $k \times k$ identity matrix as a submatrix.
2. U_{τ} and V_{τ} are “reasonably” well-conditioned.
3. $\tilde{A}_{\sigma,\tau}$ is a submatrix of A for all σ, τ .

Our choice leads to some loss of accuracy, but vastly simplifies the task of computing compressed representations in the context of integral equations. For instance, if the original A represents a Nyström discretization, then the HSS representation on each level is also a Nyström discretization, only with modified diagonal blocks, and on coarser discretizations.

Numerical examples

All numerical examples were run on standard office desktops.

Most of the programs are written in Matlab (some in Fortran 77).

The reported CPU times have two components:

(1) Pre-computation (inversion, LU-factorization, constructing a Schur complement)

(2) Time for a single solve once pre-computation is completed

Numerical examples

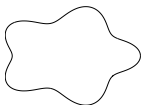
We invert a matrix approximating the operator

$$[A\phi](\mathbf{x}) = \frac{1}{2} \phi(\mathbf{x}) - \int_{\Gamma} D(\mathbf{x}, \mathbf{y}) \phi(\mathbf{y}) ds(\mathbf{y}), \quad \mathbf{x} \in \Gamma,$$

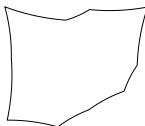
where D is the double layer kernel associated with Laplace's equation,

$$D(\mathbf{x}, \mathbf{y}) = \frac{1}{2\pi} \frac{\mathbf{n}(\mathbf{y}) \cdot (\mathbf{x} - \mathbf{y})}{|\mathbf{x} - \mathbf{y}|^2},$$

and where Γ is either one of the contours:



Smooth star



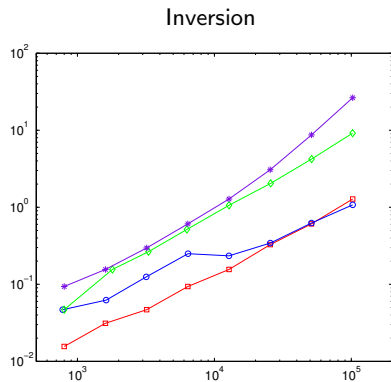
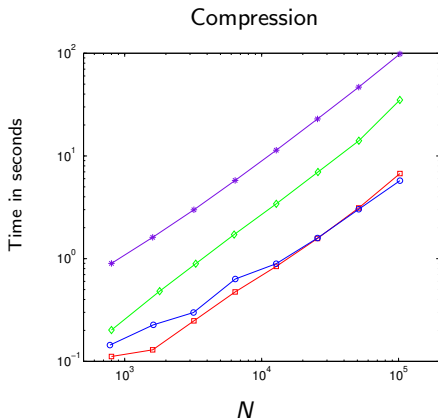
Star with corners
(local refinements at corners)



Snake
(# oscillations $\sim N$)

Examples from "A direct solver with $O(N)$ complexity for integral equations on one-dimensional domains," A. Gillman, P. Young, P.G. Martinsson, 2012.

Numerical examples



The graphs give the times required for:

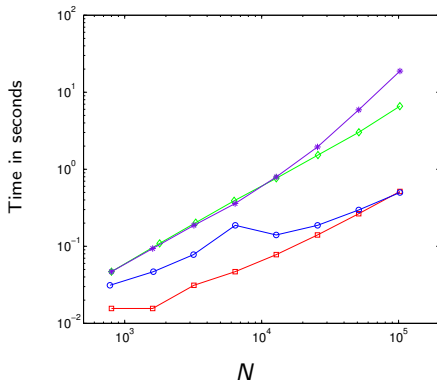
- Computing the HSS representation of the coefficient matrix.
- Inverting the HSS matrix.

Within each graph, the four lines correspond to the four examples considered:

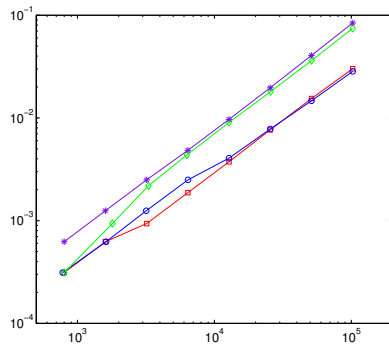
□ Smooth star
 ○ Star with corners
 ◇ Snake
 * Smooth star (Helmholtz)

Numerical examples

Transform inverse



Matrix vector multiply



The graphs give the times required for:

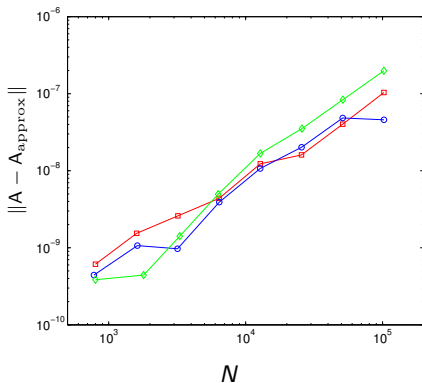
- Transforming the computed inverse to standard HSS format.
- Applying the inverse to a vector (i.e. solving a system).

Within each graph, the four lines correspond to the four examples considered:

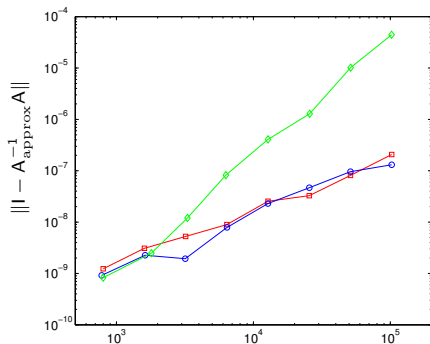
□ Smooth star ○ Star with corners ◇ Snake * Smooth star (Helmholtz)

Numerical examples

Approximation errors



Forwards error in inverse



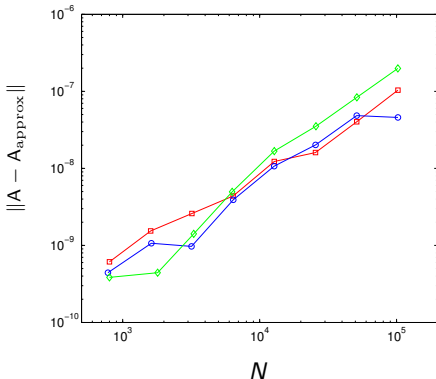
The graphs give the error in the approximation, and the forwards error in the inverse.

Within each graph, the four lines correspond to the four examples considered:

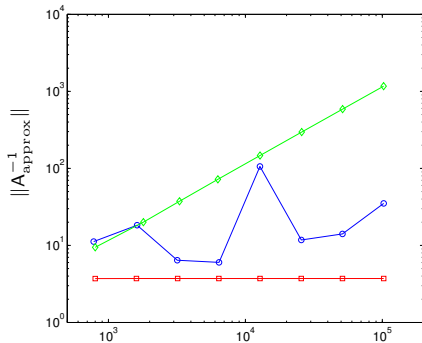
□ Smooth star ○ Star with corners ◇ Snake

Numerical examples

Approximation errors



Forwards error in inverse



The graphs give the error in the approximation, and the norm of the inverse.

Within each graph, the four lines correspond to the four examples considered:

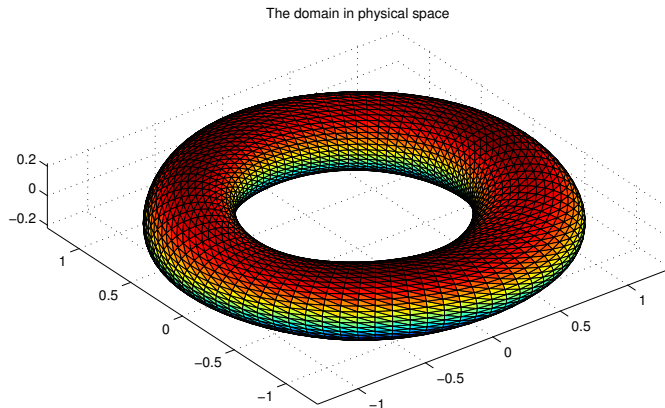
□ Smooth star ○ Star with corners ◇ Snake

Performance of direct solver for a torus domain

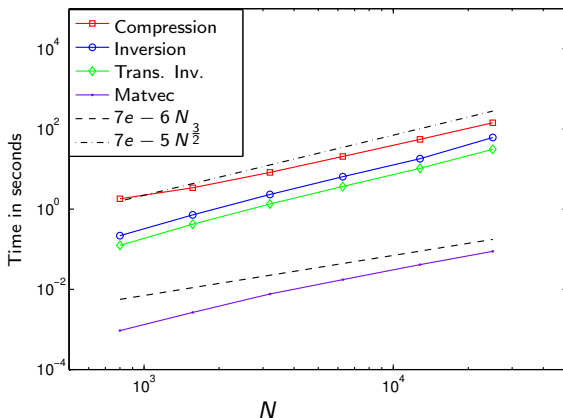
As a model problem we consider a single layer potential on a torus:

$$[A\sigma](x) = \sigma(x) + \int_{\Gamma} \log |x - y| \sigma(y) dA(y), \quad y \in \Gamma,$$

where Γ is the domain



Performance of direct solver for a torus domain



Observe that for a BIE with $N = 25\,600$, the inverse can be applied in 0.09 seconds.

The asymptotic complexity is:

Inversion step: $O(N^{1.5})$ (with small scaling constant)

Application of the inverse: $O(N)$

Current state of “fast” direct solvers for 2D BIEs

Applying the computed inverse is good... Cost $O(N)$.

The cost of storing the inverse is acceptable. $O(N)$ with a modest constant.

The cost of computing the inverse is less than desirable. $O(N^{1.5})$

In the case of 1D BIEs, the ranks of an $m \times m$ off-diagonal block scale as $O(\log m)$ as m grows.

In the case of 2D BIEs, the ranks of an $m \times m$ off-diagonal block scale as $O(m^{0.5})$ as m grows. Thus matrices of size $m^{0.5} \times m^{0.5}$ are beginning inverted.

How are we going to fix this?

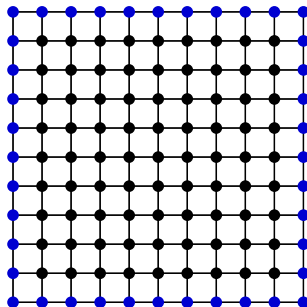
It turns out that these matrices are HSS.

We are currently developing a linear scaling technique that utilizes this fact.

Finite element matrices

Let $\Omega = \{m\}_{i=1}^N \subset \mathbb{Z}^2$ be a square lattice with N nodes and $\mathbf{u}(i)$ denote the temperature of node i and let $\mathbf{f}(i)$ denote an external heat source. Then the equilibrium equations read

$$[\mathbf{A}\mathbf{u}](i) = \mathbf{f}(i), \quad \forall i \in \Omega,$$



The operator A is an $N \times N$ sparse matrix corresponding to a *5 point stencil* .

Finite element matrices

Constant coefficient example for a 5×5 grid

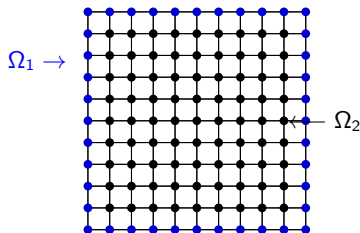
$$\mathbf{A} = \begin{bmatrix} B & -I & 0 & 0 & 0 \\ -I & B & -I & 0 & 0 \\ 0 & -I & B & -I & 0 \\ 0 & 0 & -I & B & -I \\ 0 & 0 & 0 & -I & B \end{bmatrix}$$

where

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

and I is the 5×5 identity matrix.

Definition of the Schur Complement:



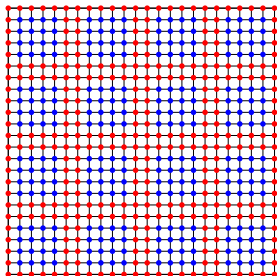
$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix}$$

The **Schur complement** of the matrix \mathbf{A} is the operator \mathbf{S} defined by

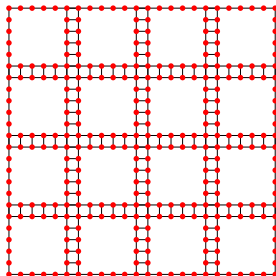
$$\mathbf{S} = \mathbf{A}_{11} - \mathbf{A}_{12} \mathbf{A}_{22}^{-1} \mathbf{A}_{21}.$$

Nested Dissection

Step 1: Partition the box into small boxes. For each box, identify the internal nodes (marked in blue) and eliminate them by computing the Schur complement for each box.

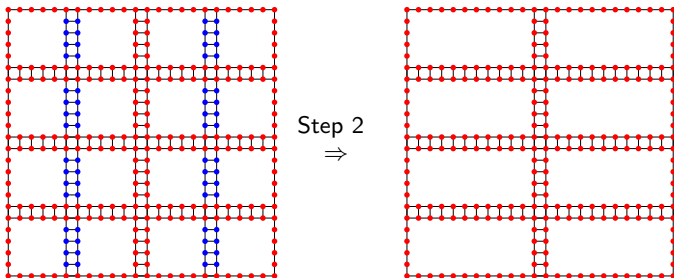


Step 1
 \Rightarrow



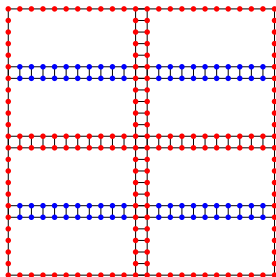
Nested Dissection

Step 2: Merge boxes by eliminating a series of vertical interior connections creating rectangular boxes.

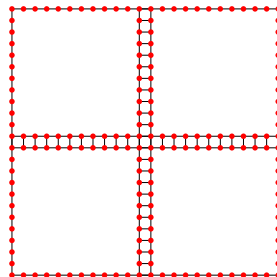


Nested Dissection

Step 3: Merge rectangular boxes by eliminating a series of horizontal interior connections.

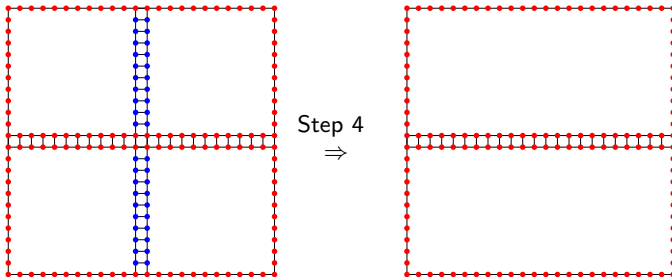


Step 3
 \Rightarrow



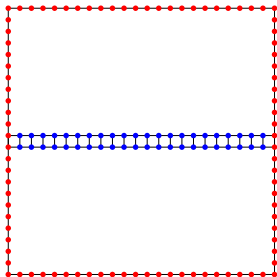
Nested Dissection

Step 4: Repeat step 2.

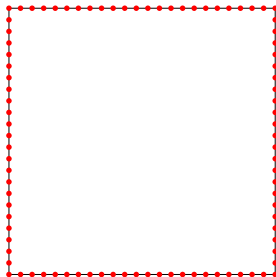


Nested Dissection

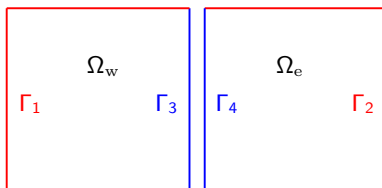
Step 5: Repeat step 3.



Step 5
 \Rightarrow



Merge two Schur Complements

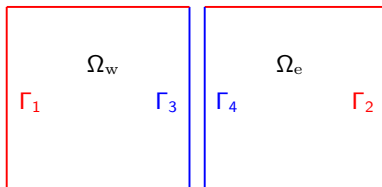


Supposing that the interior edges are unloaded, the global equilibrium equation now reads

$$\left[\begin{array}{cc|cc} \mathbf{S}_{11} & \mathbf{A}_{12} & \mathbf{S}_{13} & 0 \\ \mathbf{A}_{21} & \mathbf{S}_{22} & 0 & \mathbf{S}_{24} \\ \hline \mathbf{S}_{31} & 0 & \mathbf{S}_{33} & \mathbf{A}_{34} \\ 0 & \mathbf{S}_{24} & \mathbf{A}_{43} & \mathbf{S}_{44} \end{array} \right] \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \\ \mathbf{u}_4 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ 0 \\ 0 \end{bmatrix},$$

where \mathbf{A}_{ij} are the relevant sub-matrices \mathbf{A} .

Merge two Schur Complements



The Schur complement of the large box is

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{S}_{22} \end{bmatrix} - \begin{bmatrix} \mathbf{S}_{13} & 0 \\ 0 & \mathbf{S}_{24} \end{bmatrix} \begin{bmatrix} \mathbf{S}_{33} & \mathbf{A}_{34} \\ \mathbf{A}_{43} & \mathbf{S}_{44} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{S}_{31} & 0 \\ 0 & \mathbf{S}_{42} \end{bmatrix}.$$

Nested Dissection

Nested dissection as presented has complexity $O(N^{1.5})$.

It can be accelerated to $O(N)$ since the dense matrices have structure.

Numerical example

N	T_{solve} (sec)	T_{apply} (sec)	M (MB)	e_1	e_2
512^2	7.98	0.007	8.4	$5.56e-7$	$6.04e-7$
1024^2	26.49	0.014	18.6	$4.72e-7$	$4.98e-7$
2048^2	98.46	0.020	33.1	$2.89e-7$	$2.90e-7$
4096^2	435.8	0.039	65.6	-	-

- T_{solve} Time required to build the solution operator
 T_{apply} Time required to apply the solution operator (of size $4\sqrt{N} \times 4\sqrt{N}$)
 M Memory required to store the solution operator
 e_3 The l^2 -error in the vector $S^{-1}r$ where r is a unit vector of random direction.
 e_4 The l^2 -error in the first column of S^{-1} .

"An $O(N)$ algorithm for constructing the solution operator to elliptic boundary value problems in the absence of body loads," A. Gillman, P.G. Martinsson.

Related work:

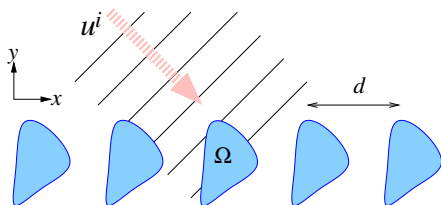
Solvers of this type have attracted much attention recently, including:

- \mathcal{H} -LU factorization of coefficient matrices by L. Grasedyck, S. LeBorne, S.Börm, et al. (2006)
- Multifrontal methods accelerated by HSS-matrix algebra: J. Xia, S. Chandrasekaran, S. Li. (2009)

Currently large effort at Purdue in this direction. (J. Xia, M. V. de Hoop, et al). Massive computations on seismic wave propagation.

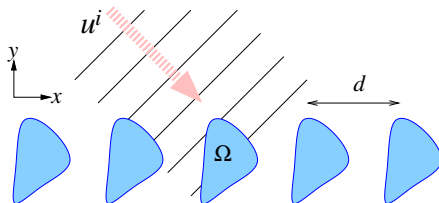
- L. Ying & P. Schmitz — general meshes in 2D, Cartesian meshes in 3D, etc. (2010).

Quasiperiodic scattering



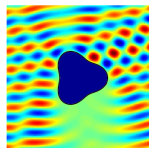
- Let $\Omega \subset \mathbb{R}^2$ denote one obstacle. Then the collection of obstacles is expressed as $\Omega_{\mathbb{Z}} = \{\mathbf{x} : (\mathbf{x} + n\mathbf{d}, y) \in \Omega \text{ for some } n \in \mathbb{Z}\}$.
- The obstacles are hit by an incident plane wave $u^{\text{inc}} = e^{i\mathbf{k} \cdot \mathbf{x}}$ where $|k| = \omega$.
- Our goal is to find the total field $u^{\text{total}} = u^{\text{inc}} + u$.
- Utilize the fact that each part of the field is quasiperiodic:
ie. $u(\mathbf{x} + \mathbf{d}, y) = \alpha u(\mathbf{x}, y)$ where $\alpha = e^{i\kappa \cdot \mathbf{d}}$ denotes the Bloch phase.

Quasiperiodic scattering



$$\begin{aligned}
 (\Delta + \omega^2)u(\mathbf{x}) &= 0 & \mathbf{x} \in \mathbb{R} \setminus \Omega_{\mathbb{Z}} \\
 u(\mathbf{x}) &= u^{\text{inc}}(\mathbf{x}) & \mathbf{x} \in \partial\Omega_{\mathbb{Z}} \\
 u &\text{ 'radiative' as } y \rightarrow \pm\infty
 \end{aligned}$$

Single object scattering



Consider the problem

$$\begin{aligned} (\Delta + \omega^2)u(\mathbf{x}) &= 0 & \mathbf{x} \in \mathbb{R} \setminus \Omega \\ u(\mathbf{x}) &= u^{\text{inc}}(\mathbf{x}) & \mathbf{x} \in \partial\Omega \\ u &\text{ 'radiative' far from } \Omega \end{aligned}$$

The solution can be represented as a double layer potential

$$u(\mathbf{x}) = \int_{\Gamma} \partial \nu G_{\omega}(\mathbf{x}, \mathbf{y}) \tau(\mathbf{y}) ds(\mathbf{y}), \quad \mathbf{x} \in \Omega,$$

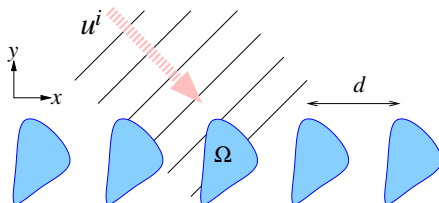
where ν is the outward normal and $G_{\omega}(\mathbf{x}, \mathbf{y})$ is the fundamental solution

$$G_{\omega}(\mathbf{x}, \mathbf{y}) = \frac{i}{4} H_0^{(1)}(\omega |\mathbf{x} - \mathbf{y}|).$$

Then the boundary charge distribution τ satisfies the boundary integral equation

$$-\frac{1}{2}\tau(\mathbf{x}) + \int_{\Gamma} \partial \nu G_{\omega}(\mathbf{x}, \mathbf{y}) \tau(\mathbf{y}) ds(\mathbf{y}) = u^{\text{inc}}(\mathbf{x})$$

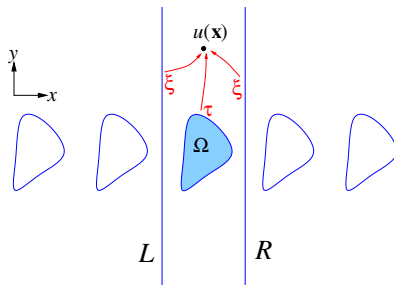
Quasiperiodic scattering



Replace $G_\omega(\mathbf{x}, \mathbf{y})$ by $G_{\omega, \text{QP}}(\mathbf{x}) := \sum_{m \in \mathbb{Z}} \alpha^m G_\omega(\mathbf{x} - m\mathbf{d})$.

This has some problems...

Quasiperiodic scattering



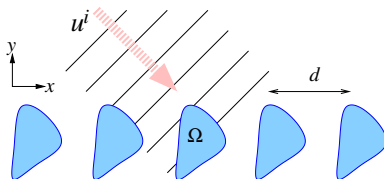
By using $G_\omega(\mathbf{x}, \mathbf{y})$ and enforcing periodicity by introducing “boundaries”, the problems can be avoided. The result is the following $(N + M) \times (N + M)$ linear system

$$\begin{bmatrix} A & B \\ C & Q \end{bmatrix} \begin{bmatrix} \tau \\ \xi \end{bmatrix} = \begin{bmatrix} -\mathbf{u}^{\text{inc}} \\ \mathbf{0} \end{bmatrix},$$

where A is $N \times N$ and Q is $M \times M$. Typically, $M < 200$.

L. Greengard and A. Barnett (2010)

A fast quasiperiodic solver



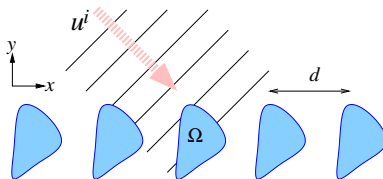
$$\begin{bmatrix} A & B \\ C & Q \end{bmatrix} \begin{bmatrix} \tau \\ \xi \end{bmatrix} = \begin{bmatrix} -\mathbf{u}^{\text{inc}} \\ \mathbf{0} \end{bmatrix}$$

By using HSS algebra and the Schur complement, we construct a fast technique for applying the inverse of the system.

$$\xi = (Q - CA^{-1}B)^{-1}A^{-1}\mathbf{u}^{\text{inc}}$$

$$\tau = A^{-1}\mathbf{u}^{\text{inc}} - A^{-1}B\xi$$

A fast quasiperiodic solver



$$\begin{bmatrix} A & B \\ C & Q \end{bmatrix} \begin{bmatrix} \tau \\ \xi \end{bmatrix} = \begin{bmatrix} -\mathbf{u}^{\text{inc}} \\ \mathbf{0} \end{bmatrix}$$

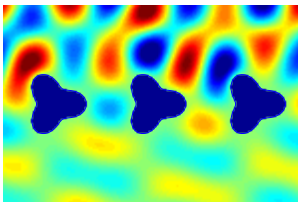
By using HSS algebra and the Schur complement, we construct a fast technique for applying the inverse of the system.

$$\xi = (Q - CA^{-1}B)^{-1}A^{-1}\mathbf{u}^{\text{inc}}$$

$$\tau = A^{-1}\mathbf{u}^{\text{inc}} - A^{-1}B\xi$$

Numerical example

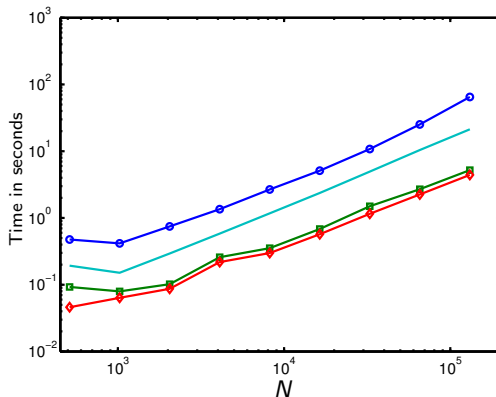
$$\omega = 10$$



$$\begin{bmatrix} A & B \\ C & Q \end{bmatrix} \begin{bmatrix} \tau \\ \xi \end{bmatrix} = \begin{bmatrix} -\mathbf{u}^{\text{inc}} \\ \mathbf{0} \end{bmatrix}$$

The size of A varies while $M = 90$ is fixed.

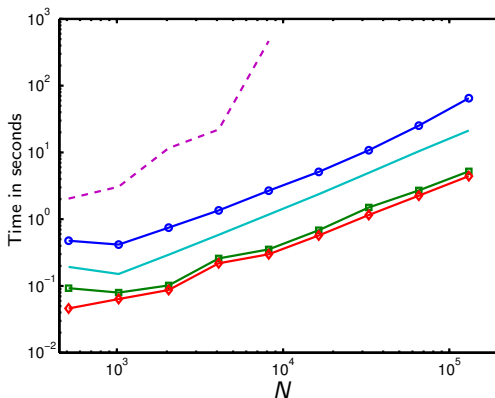
Numerical example



The four lines correspond to:

○ Compress □ Invert ◇ Transform Inv. — Block solve

Numerical example



The four lines correspond to:

○ Compress □ Invert ◇ Transform Inv. — Block solve - - - Dense

For $N = 8192$, the new technique can do 394 solves in the time it takes the dense solver to do one.

Concluding remarks and comments

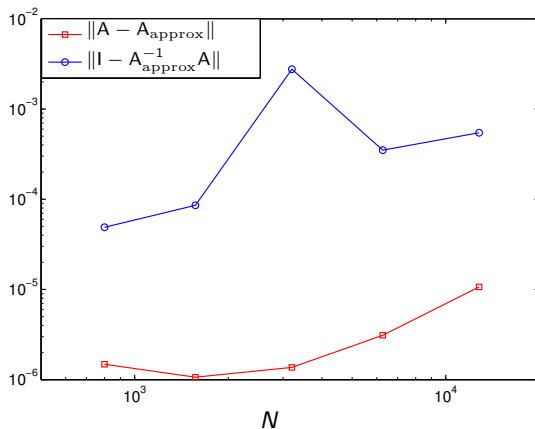
Assertions:

- Fast direct solvers excel for problems on 1D domains.
 - Integral operators on the line.
 - Boundary Integral Equations in \mathbb{R}^2 .
 - Boundary Integral Equations on rotationally symmetric surfaces in \mathbb{R}^3 .
 - (Low frequency) Quasiperiodic Scattering in \mathbb{R}^2 .
- Existing fast direct solvers for “finite element matrices” associated with elliptic PDEs in \mathbb{R}^2 work very well. In \mathbb{R}^3 , they can be game-changing in specialized environments.

Predictions:

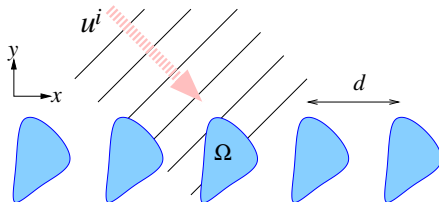
- For BIEs associated with non-oscillatory problems on surfaces in \mathbb{R}^3 , an $O(N)$ complexity (with a modest constant) solver will exist.
- *Randomized methods* will be extremely helpful.
- Direct solvers for *scattering problems* will find users, even if expensive. $O(N^{1.5})$ or $O(N^2)$ flop counts may be OK, provided parallelization is possible.
- Direct solvers will provide a fantastic tool for *numerical homogenization*.

Performance of direct solver for a torus domain



Errors for the same problem as the previous slide.

Quasiperiodic scattering



Replace $G_\omega(\mathbf{x}, \mathbf{y})$ by $G_{\omega, \text{QP}}(\mathbf{x}) := \sum_{m \in \mathbb{Z}} \alpha^m G_\omega(\mathbf{x} - m\mathbf{d})$.

This has some problems...

- Wood's anomalies
- Difficulty in evaluation
- Converges in disk