In this talk, we will discuss numerical methods for solving the equation

$$\text{(BVP)} \qquad \begin{cases} A\,u(x) = 0, & x \in \Omega, \\ B\,u(x) = f(x), & x \in \Gamma, \end{cases}$$

where A is a linear constant-coefficient partial differential operator, and B is some local linear boundary operator (*e.g.* Dirichlet B.C. $\Leftrightarrow B = I$).

We'll start by collecting some examples of practical interest.

Laplace's equation

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

where

$$\Delta\, u = \frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2},$$

and where Ω is a domain with boundary Γ, f is a given function on the boundary, v is the unknown function.

- Equilibrium equation for diffusive processes (heat conduction, *etc.*).
- Viscous fluid flows.
- Static electric and magnetic fields.
- Gravitational fields.
- Etc. etc. etc. etc.

The equations of linear elasticity

Let $u(x) = [u_1(x),\, u_2(x)]$ denote the displacement of a piece of elastic material in equilibrium. Then u satisfies

$$-\mu\,\Delta u_1 - \kappa\left(\frac{\partial^2 u_1}{\partial x_1^2} + \frac{\partial^2 u_2}{\partial x_1 x_2}\right) = 0, \quad \text{on } \Omega,$$

$$-\mu\,\Delta u_2 - \kappa\left(\frac{\partial^2 u_1}{\partial x_1 x_2} + \frac{\partial^2 u_2}{\partial x_2^2}\right) = 0, \quad \text{on } \Omega,$$

where μ and κ are material constants. Dirichlet boundary conditions read

$$(u_1,\, u_2) = (f_1,\, f_2), \qquad \text{on } \Gamma.$$

Closely related to the bi-harmonic equation,

$$(-\Delta)^2\, u = 0.$$

The wave equation

$$-\Delta u + \frac{\partial^2 u}{\partial t^2} = 0$$

Describes wave propagation in a range of different applications:

- Waves in fluids.
- (Certain) waves in elastic bodies.
- Electromagnetic waves.
- etc. etc. etc.

Closely related is the Helmholtz equation:

$$-\Delta u - \omega^2 u = 0.$$

The Maxwell Equations

$$\begin{cases} \nabla \cdot \mathbf{E} = \rho & \nabla \times \mathbf{E} = -\dfrac{\partial \mathbf{B}}{\partial t} \\ \nabla \cdot \mathbf{B} = 0 & \nabla \times \mathbf{B} = \mathbf{J} + \dfrac{\partial \mathbf{E}}{\partial t} \end{cases}$$

Models electro-magnetism.

The Maxwell equations in many environments simplify to either the Laplace equation, or to the wave equation (and thence to the Helmholtz equation).

The Shrödinger Equation

$$(-\Delta + V)\,\Psi = i\frac{\partial \Psi}{\partial t}$$

A related stationary problem is modelled by the Yukawa equation

$$(-\Delta + V + \omega)\,\Psi = 0.$$

These are the fundamental equations of quantum mechanics.

Finding numerical solutions is crucial in chemistry.

The Navier-Stokes Equations

$$\begin{cases} \rho\left(\dfrac{\partial}{\partial t}\mathbf{v} + (\mathbf{v}\cdot\nabla)\mathbf{v}\right) - \mu\Delta\mathbf{v} + \nabla p = 0, \\ \nabla\cdot\mathbf{v} = 0. \end{cases}$$

Models fluid flows.

In common time-stepping schemes, the iterated solution of the modified Stokes' equation is required. This equation reads

$$\begin{cases} \alpha\mathbf{v} - \mu\Delta\mathbf{v} + \nabla p = \mathbf{f}, \\ \nabla\cdot\mathbf{v} = 0. \end{cases}$$

Conclusion: In a broad range of applications, one is faced with the task of numerically solving a Boundary Value Problem of the form

$$\text{(BVP)} \qquad \begin{cases} A\,u(x) = 0, & x \in \Omega, \\ B\,u(x) = f(x), & x \in \Gamma, \end{cases}$$

where A is a linear constant-coefficient partial differential operator, and B is some local linear boundary operator (*e.g.* Dirichlet B.C. $\Leftrightarrow B = I$).

The numerical solution of linear BVP's is frequently the time-limiting step in computational models. Challenging environments include:

- Large-scale scattering problems.
- Equations on complicated domains:
  - The equations of elasticity in an engine-block.
  - Crack propagation in a fibre composite.
  - Wave-propagation in heterogeneous media.
- Environments where repeated solves are necessary:
  - Monte Carlo simulations in biochemistry,
  - Molecular dynamics,
  - Geometry optimization.
- etc. etc. etc.

For the equations we are interested in here, free-space equations have analytic solutions. As an example, the solution of

$$\begin{cases} -\Delta\, u(x) = f(x), & x \in \mathbb{R}^3, \\ \lim\limits_{|x|\to\infty} |u(x)| = 0, \end{cases}$$

is given by

$$u(x) = [G * f](x) = \int_{\mathbb{R}^3} G(x-y)\, f(y)\, dy,$$

where G is the fundamental solution of the Laplace operator,

$$G(x) = \frac{1}{4\pi|x|}.$$

Loosely speaking, the function G satisfies

$$-\Delta\, G(x) = \delta(x).$$

Examples of explicit fundamental solutions:

Laplace in 2D:	$-\Delta u = f$	$G(x) = -\dfrac{1}{2\pi} \log \lvert x \rvert$
Laplace in 3D:	$-\Delta u = f$	$G(x) = \dfrac{1}{4\pi \lvert x \rvert}$
Helmholtz in 2D:	$(-\Delta - k^2)\, u = f$	$G(x) = H_0^{(1)}(k\lvert x \rvert)$
Helmholtz in 3D:	$(-\Delta - k^2)\, u = f$	$G(x) = \dfrac{e^{ik\lvert x \rvert}}{4\pi \lvert x \rvert}$
Yukawa in 3D:	$\left(-\Delta + m^2\right)\, u = f$	$G(x) = \dfrac{e^{-m\lvert x \rvert}}{4\pi \lvert x \rvert}$
Elasticity in 3D:	$\sum_{j,k,l=1}^{3} C_{ijkl}(u_{k,jl} + u_{l,jk}) = f_i$	$G(x) =$ messy stuff

Remark: While it is in principle a trivial matter to write down the solution of a free-space equation,

$$\text{(INT)} \qquad u(x) = [G * f](x) = \int_{\mathbb{R}^3} G(x-y)\, f(y)\, dy,$$

it is not a trivial matter to rapidly compute a numerical approximation of the integral.

If you need N degrees of freedom to represent the function f, then in a wide range of environments, there exist $O(N)$ methods for evaluating (INT) to high accuracy. However, these are somewhat recent results.

The real difficulties arise when boundaries are present.

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

It is in principle simple to eliminate the body load. Set

$$u = u_{\text{hom}} + u_{\text{part}},$$

where

$$u_{\text{part}}(x) = \int_\Omega G(x-y)\, h(y)\, dy. \tag{4}$$

Then u_{hom} satisfies

$$\begin{cases} -\Delta\, u_{\text{hom}}(x) = 0, & x \in \Omega, \\ \quad u_{\text{hom}}(x) = f(x) - u_{\text{part}}(x), & x \in \Gamma, \end{cases}$$

and we obtain a problem with no body load.

In practice, it is not so easy to evaluate (4) to high accuracy.

Still, we will henceforth assume there are no body-loads.

For simple geometries, separation of variables does the trick.

Example: Set $\Omega = \{x \in \mathbb{R}^2 : \ |x| \leq 1\}$, and consider

$$\begin{cases} -\Delta\, u(x) = 0, & x \in \Omega, \\ \quad u(x) = f(x), & x \in \Gamma, \end{cases} \tag{5}$$

Use polar coordinates, $(x_1,\, x_2) = (r\cos\theta,\, r\sin\theta)$, and Fourier expand f,

$$f(\theta) = \alpha_0 + \sum_{n=1}^{\infty} \big(\alpha_n\, \cos(n\theta) + \beta_n\, \sin(n\theta)\big).$$

The solution u is given by

$$u(x) = \alpha_0 + \sum_{n=1}^{\infty} \big(\alpha_n\, r^n\, \cos(n\theta) + \beta_n\, r^n\, \sin(n\theta)\big).$$

Similar solutions exist for rectangles, half-planes, strips, pie-slices, spheres, cones, cylinders, etc.

In such environments, FFT-type methods produce extremely fast solvers.

(Recent result by M. Tygert: very fast algorithms for rapidly expanding functions in spherical harmonics, or other orthogonal systems.)

Core problem: How do you numerically solve the equation

$$\text{(BVP)} \qquad \begin{cases} A\,u(x) = 0, & x \in \Omega, \\ B\,u(x) = f(x), & x \in \Gamma. \end{cases}$$

where A is a linear constant-coefficient partial differential operator, and B is some local linear boundary operator.

Option 1: Discretize the differential operator directly: Instead of

$$\text{(BVP)} \qquad \begin{cases} A\,u(x) = 0, & x \in \Omega, \\ u(x) = f(x), & x \in \Gamma, \end{cases}$$

solve

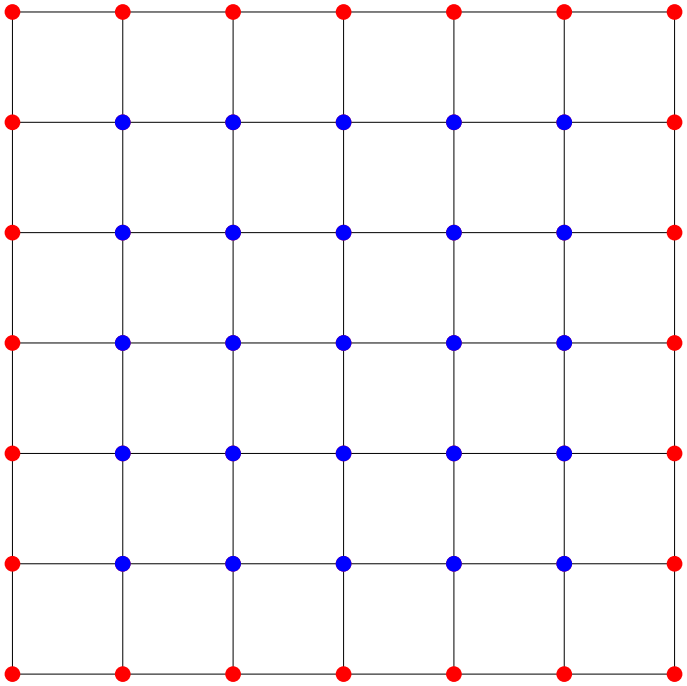
$$\text{(BVP-DISC)} \qquad A_N\,u_N = g_N,$$

where u_N is a function in an N-dimensional function space, A_N is an $N \times N$ matrix discretizing the operator A (obtained via a Finite Element / Finite Differencees / ... discretization), and g_N is a vector of data derived from f.

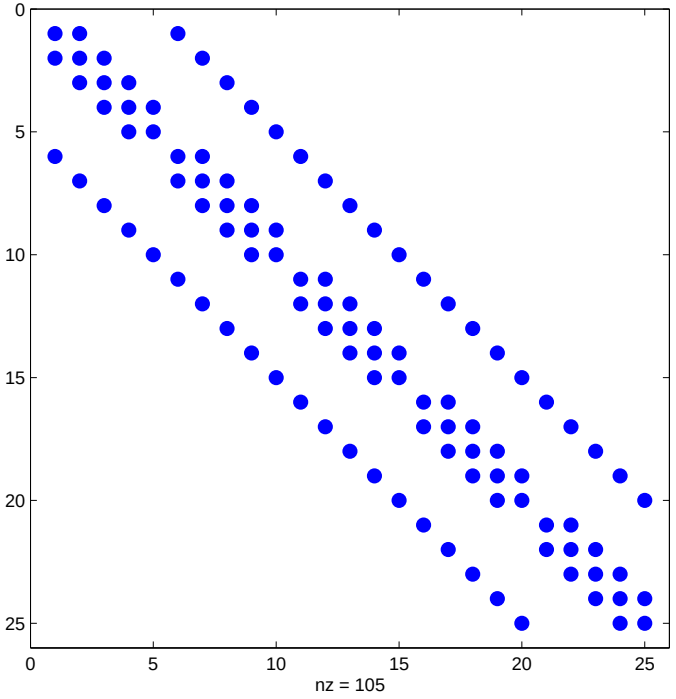
Example: Let Ω be a square, and let $A = -\Delta$. Discretize Ω into a grid, and let A_N denote the five-point stencil,

$$[A_N\varphi](i,j) = \frac{1}{h^2}\big(4\varphi(i,j) - \varphi(i-1,j) - \varphi(i+1,j) - \varphi(i,j-1) - \varphi(i,j+1)\big).$$

Then A_N is a sparse matrix.



The grid.



Sparsity pattern of A_N.

We next need to solve a linear system:

$$A_N \, u_N = g_N.$$

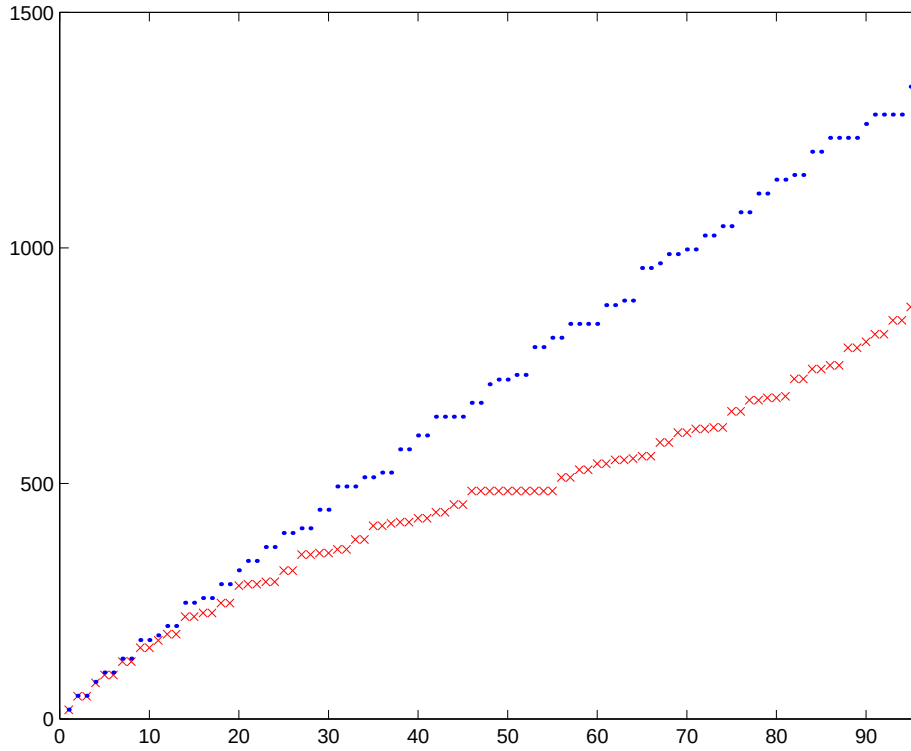
Since N is typically large, using Guassian elimination would be very expensive. However, since A_N is sparse, we can use an iterative solver (conjugate gradients/GMRES/...).

Problem:

A is an unbounded operator $\Rightarrow$

The matrix A_N is ill-conditioned $\Rightarrow$

The iterative solver converges slowly.



Pre-conditioners can help solving ill-conditioned linear systems.

A pre-conditioner is an operator P_N such that:

- It is cheap to apply P_N to a vector.
- The product $P_N\, A_N$ is well-conditioned.

Loosely speaking, $P_N \approx A_N^{-1}$.

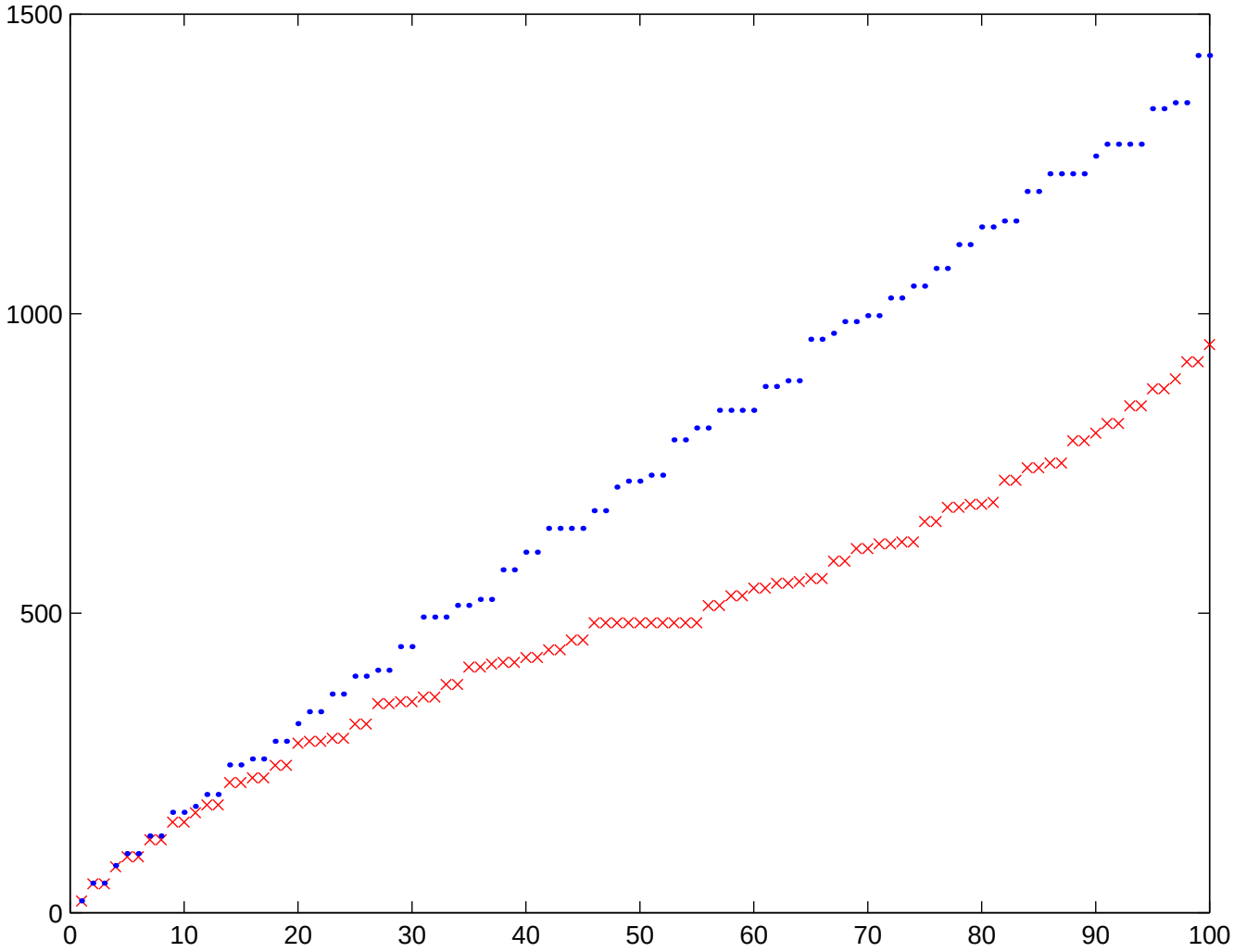
The idea is to use an iterative solver to solve

$$P_N\, A_N\, u_N = P_N\, g_N.$$

The popular multigrid algorithm is a form of a pre-conditioner.

However, many problems related to ill-conditioning remain.

The spectrum of the discrete Laplacian (the "five-point stencil") approximates the spectrum of the Laplacian:



Option 2 (for numerically solving a BVP):

Reformulate the BVP as a Boundary Integral Equation.

Example:

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

We make the following Ansatz:

$$u(x) = \int_\Gamma \big(n(y) \cdot \nabla_y \log|x-y|\big) v(y)\, ds(y), \qquad x \in \Omega,$$

where $n(y)$ is the outward pointing unit normal of Γ at y. Then the boundary charge distribution u satisfies the Boundary Integral Equation

$$\text{(BIE)} \qquad v(x) + 2\int_\Gamma \big(n(y) \cdot \nabla_y \log|x-y|\big) v(y)\, ds(y) = 2f(x), \qquad x \in \Gamma.$$

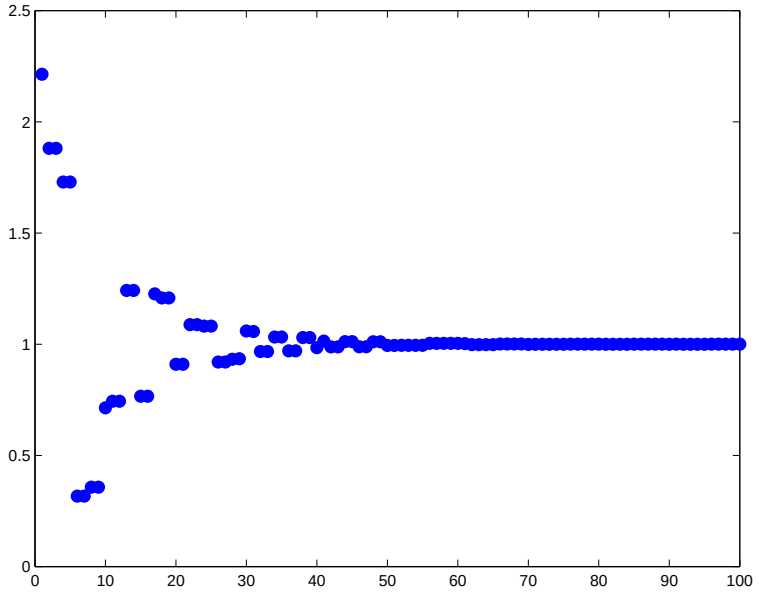
- (BIE) and (BVP) are in a strong sense equivalent.
- (BIE) is appealing mathematically (2^{nd} kind Fredholm equation).

Second kind Fredholm Eqn.

$$(I + K)\, u = f$$

K is a compact ("almost finite-dimensional") operator.

Typical spectrum of $I + K$:

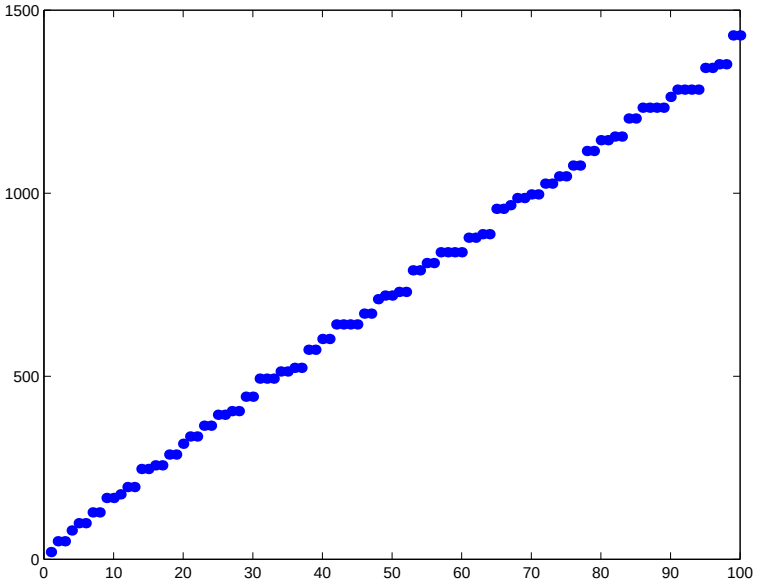


Partial Differential Equation

$$-\Delta\, u = g$$

$-\Delta$ is an unbounded operator.

Typical spectrum of $-\Delta$:



The condition numbers of the *discretized* operators.

h	Cond. nr. of discretized BIE		Cond. nr. of discrete Laplacian	
0.2	8.546837835256035	$(N = 25)$	2.1E1	$(N = 50)$
0.1	7.053618952378199	$(N = 50)$	7.0E1	$(N = 200)$
0.05	6.993154106860152	$(N = 100)$	2.6E2	$(N = 800)$
0.025	6.993012937976997	$(N = 200)$	9.9E2	$(N = 1\,600)$
0.0125	6.993012936936591	$(N = 400)$	3.9E3	$(N = 6\,400)$
0.00625	6.993012936936595	$(N = 800)$	1.5E4	$(N = 25\,600)$

Rewriting the BVP as a BIE can be viewed as analytic pre-conditioning.

We frequently have a choice between:

$$\text{PDE formulation} \quad \Leftrightarrow \quad \text{Integral Equation formulation}$$

There are compelling arguments in favor of the IE formulation:

Conditioning:

When there exists an IE formulation that is a Fredholm equation of the second kind, the mathematical equation itself is well-conditioned.

Dimensionality:

Frequently, an IE can be defined on the boundary of the domain.

Integral operators are benign objects:

It is (relatively) easy to implement high order discretizations of integral operators. Relative accuracy of 10^{-10} or better is often achieved.

Unfortunately, there is a fundamental drawback to basing numerical methods on integral operators:

Discretization of integral operators typically results in dense matrices.

In the 1950's when computers made numerical PDE solvers possible, researchers faced a grim choice:

PDE-based:	Ill-conditioned, N is too large, low accuracy.
Integral Equations:	Dense system.

The integral equations lost and were largely forgotten

— they were simply too expensive.

(Except in some scattering problems where there was no choice.)

The situation changed dramatically in the 1980's. It was discovered that while K_N is dense, it is possible to evaluate the matrix-vector product

$$v \mapsto K_N\, v$$

in $O(N)$ operators – to high accuracy and with a small constant.

The most successful such algorithm is the Fast Multipole Method by Rokhlin and Greengard.

Let

$$(I + K_N)\, v_N = g_N,$$

denote the discretized version of a second kind Fredholm equation, *e.g.*

$$u(x) + 2 \int_\Gamma \big(n(y) \cdot \nabla_y \log|x - y|\big) u(y)\, ds(y) = 2f(x), \qquad x \in \Gamma.$$

A prescription for rapidly solving BVPs:

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

Convert (BVP) to a second kind Fredholm equation:

(BIE)
$$u(x) + \int_\Gamma \big(n(y) \cdot \nabla_y \log|x-y|\big) u(y)\, ds(y) = f(x), \qquad x \in \Gamma.$$

Discretize (BIE) into the discrete equation

(DISC)
$$(I + K_N) u_N = f_N$$

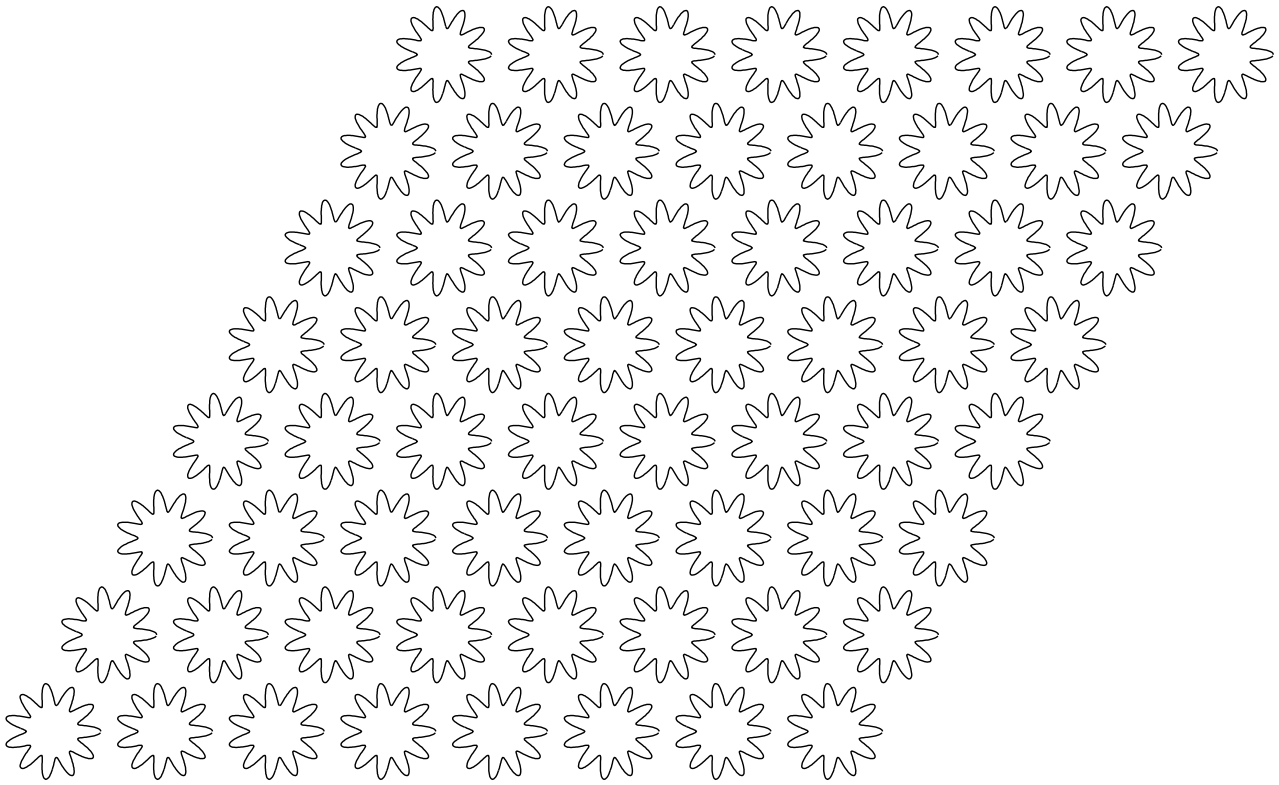
where K_N is a (typically dense) $N \times N$ matrix.

Fast Multipole Method — Can multiply K_N by a vector in $O(N)$ time.

Iterative solver — Solves (DISC) using $\sqrt{\kappa}$ matrix-vector multiplies, where κ is the condition number of $(I + K_N)$.

Total complexity — $O(\sqrt{\kappa}\, N)$. (Recall that κ is small. Like 14.)

Example:



External Laplace problem with Dirichlet boundary data.

The contour is discretized into 25 600 points.

A single matrix-vector multiply takes 0.2 sec on a 2.8 Ghz desktop PC.

Fifteen iterations required for 10^{-10} accuracy $\rightarrow$ total CPU time is 3 sec.

For the boundary value problems where the "boundary equation + iterative solver + FMM" methodology works, it is in a strong sense an *optimal* solver:

1. The condition number of the numerical equation is similar to the condition number of the original problem. As a consequence, very high accuracy can be obtained,

$$\text{achievable accuracy} \sim \kappa\, \varepsilon_{\text{machine}}.$$

2. Computational complexity is asymptotically optimal:

$$\text{number of flops} \sim N,$$

   where N is the number of discretization points *on the boundary*.

3. Ease of discretization of a curve – as opposed to an area.
   Ease of discretization of an integral operator – as opposed to a PDO.
   High order schemes are easily implemented.

Limitations:

These schemes are fundamentally linear.

Not all BVPs can be reformulated as boundary integral equations.
For those that cannot, there is no gain in dimensionality.

More research is needed:

1. Lack of systematic techniques for rewriting PDE's to second kind Fredholm equations on the boundary. In particular: The presence of corners, cracks, cusps, ridges, etc, make it harder to construct second kind Fredholm equations (the compactness of the integral operators is compromised).

2. Lack of discretization schemes for "difficult geometries", even in 2D.

3. Lack of systematic schemes for discretizing surfaces in 3D.

Dependence on iterative solvers is a problem:

As long as a numerical method is based on iterative solvers, its performance is held hostage to the number of iterations required.

This is a particular problem when it comes to problems that are inherently poorly conditioned, such as high-frequency scattering problems.

Another drawback of relying on iterative solvers is that they have a bad reputation among developers of general-purpose software. In commercial software development, robustness is a higher priority than performance.

New development:

- Derivation of direct solvers to complement, and hopefully replace, the iterative ones. (PGM, V. Rokhlin, M. Tygert)

Direct solvers

Recall that many BVPs can be cast in the following form:

$$\text{(BIE)} \qquad u(x) + \int_\Gamma g(x,y)u(y)\,ds(y) = f(x), \qquad x \in \Gamma.$$

Upon discretization, equation (BIE) turns into a discrete equation

$$\text{(DISC)} \qquad (I + K_N)u = f$$

where K_N is a (typically dense) $N \times N$ matrix.

A ***direct method*** computes a compressed representation for $(I + K_N)^{-1}$.

- Cost for pre-computing the inverse.
- Cost for applying the inverse to a vector.

In many environments, both of these costs can be made $O(N)$.

Direct methods are good for (1) ill-conditioned problems, (2) problems with multiple right-hand sides, (3) spectral decompositions, (4) updating, ...

Sampling of related work:

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, et al*,

2002 inversion of Lippmann-Schwinger equations, *Y. Chen*.

Current state of the research

The fast direct solvers we are developing exploit the fact that off-diagonal blocks of the matrix to be inverted have low rank.

This restricts the range of application to non-oscillatory, or moderately oscillatory problems. In other words, we **can** do:

- Laplace's equation, equations of elasticity, Yukawa's equation,...
- Helmholtz' and Maxwell's equations for low- and intermediate freqs.

(In special cases, high frequency problem can also be solved.)

Boundary integral equations in 2D are completely understood.

Lippmann-Schwinger eqns in 2D are well understood, implementation is under way.

In 3D, we "know" how to solve the problem, but much work remains.

Technical aspects:

Once you have a compressed inverse, applying it is very similar to applying the original operator using an FMM:

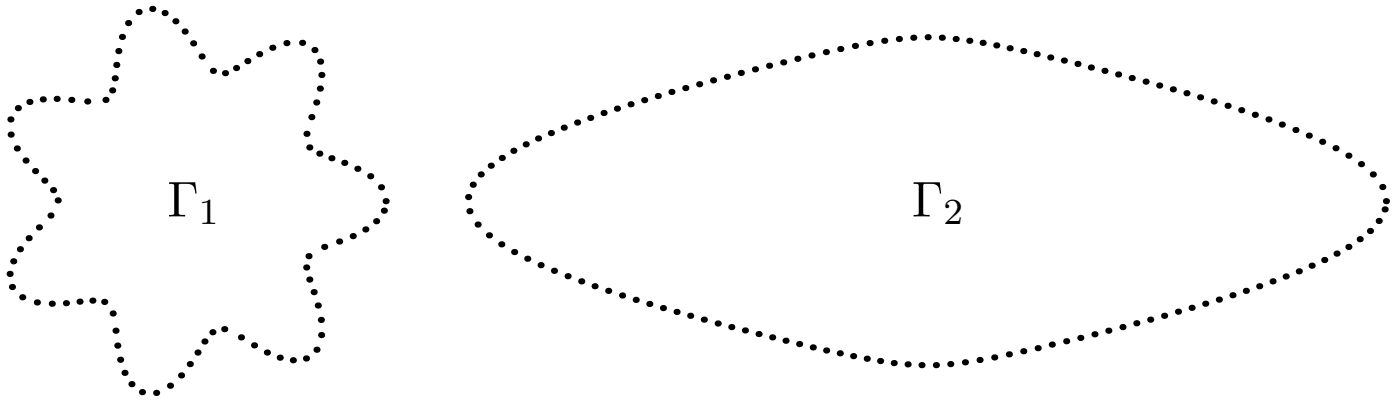
- Hierarchical (adaptive) subdivision of the computational domain.
- Outgoing fields are aggregated through an upwards pass.
- Incoming fields are aggregated through a downwards pass.

Pre-computation of the inverse operator is slightly different:

- There is only an upwards pass.
- For each subdomain, we compute operators, rather than fields. (Inverses, and "Schur complements").

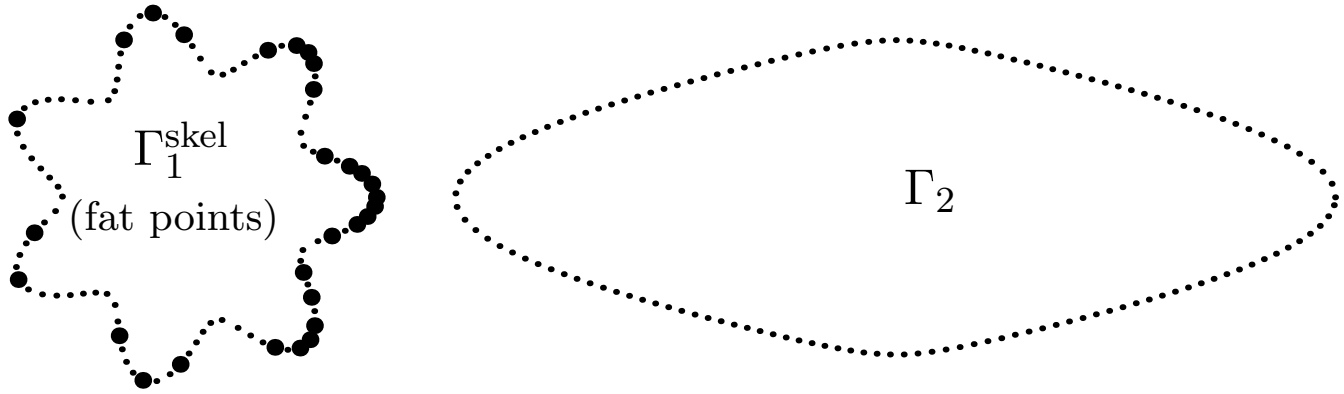
Important differences between the current methods and classical FMM:

- Incoming and outgoing potentials are represented via tabulation.
- Randomized algorithms are used to compress low-rank matrices.



100 charges on Γ_1 induce a potential v on Γ_2.

The same potential v can be reproduced by placing only 30 charges on the bold points in the figure below. (To within precision 10^{-10}.)



One can pick the 30 charge locations so that they work for ***any*** *distribution of charges.*

(In the classical FMM, a multipole expansion was used to reproduce v.)

Let A_{21} denote the matrix that maps charges on Γ_1 to potentials on Γ_2. (In other words, A_{21} is an off-diagonal block of the system matrix.)

Suppose that A_{21} is an $m \times n$ matrix and has ε-rank k.

Then A_{21} allows the factorization

$$\begin{array}{ccccc} A_{21} & = & A_{\text{col}} & P & +O(\varepsilon) \\ m \times n & & m \times k & k \times n & \end{array}$$

where

- A_{col} consists of k columns of A_{21}.
- The $k \times k$ identity matrix is a submatrix of P.
- No element of P has magnitude larger than one.

Let A_{21} denote the matrix that maps charges on Γ_1 to potentials on Γ_2. (In other words, A_{21} is an off-diagonal block of the system matrix.)

Suppose that A_{21} is an $m \times n$ matrix and has ε-rank k.

Then A_{21} allows the factorization

$$\begin{array}{ccccc} A_{21} & = & A_{\text{col}} & P & +O(\varepsilon) \\ m \times n & & m \times k & k \times n & \end{array}$$

where

- A_{col} consists of k columns of A_{21}.
- The $k \times k$ identity matrix is a submatrix of P.
- No element of P has magnitude larger than one.

Question: How do you compute the factorization above when m and n are much larger than k, but you can apply A_{21} rapidly to a vector?

Generalization — an interpolation result:

Theorem: *Let Ω be a compact set, and let V be a k-dimensional space of complex-valued continuous functions on Ω.*

Then there exist k points $\{x_j\}_{j=1}^k \subseteq \Omega$, and k continuous functions $\{\varphi_j\}_{j=1}^k$ on Ω such that for all $f \in V$,

$$f(x) = \sum_{j=1}^{k} f(x_j)\, \varphi_j(x).$$

Moreover, for $j = 1, \ldots, k$,

$$|\varphi_j(x)| \leq 1, \quad \forall\ x \in \Omega.$$

In many environments, the points $(x_n)_{n=1}^N$ and the functions $(\varphi_n)_{n=1}^N$ can be computed rapidly and accurately.

For the result regarding an $m \times n$ matrix A of rank k, simply set:

$$\Omega = \{1,\, 2,\, \ldots, n\}, \qquad V = \mathrm{Row}(A).$$

Randomized algorithms:

(Mark Tygert, Vladimir Rokhlin, PGM)

A rank-k approximation to a matrix A can be obtained via the application of A to $k + 20$ random vectors.

Theorem: *Let A be an $m \times n$ matrix and let k be an integer. Set $l = k+20$, and let G be an $n \times l$ matrix with i.i.d. Gaussian elements. Let $(q_j)_{j=1}^{k}$ denote the first k left singular vectors of $A\,G$ and set $Q = [q_1, \ldots, q_k]$. Then with probability at most 10^{-17},*

$$||A - Q\,Q^{\mathrm{t}}\,A||_2 \leq 10\,\sqrt{(k+20)\,m}\,\sigma_{k+1},$$

where σ_j denotes the j'th singular value of A.

This should be compared to Krylov-subspace methods (Lanczos etc.).

The Rokhlin-paradigm for solving PDEs:

1. Rewrite as a second kind Fredholm equation.
2. Use iterative solvers + FMM to solve the integral equation.

Work in progress:

- Replace [Iterative solver + FMM] by direct solvers. 2D is done.
- Interpolative representation of functions.
- Randomized algorithms.

Applications:

- Fast methods for anisotropic elasticity.
- Heterogeneous materials: percolation and wave-propagation.
- Computational strategies for modeling biochemical phenomena, in particular, ion channels and macro-molecules in ionic solutions.