

August 27, 2026
Announcements

Goals

- PDE solvers from those pieces
- logistics
- operators in action

Review

- integral operators
- "fast" algorithms
- compact operators

Applications: (Elliptic) PDE solving (I)

$\nabla^2 = \partial_x^2 + \partial_y^2$

Poisson: $\Delta u = f$ (on Ω) $u = g$ (on $\partial\Omega$)

$\Delta G = \delta$ $\leftarrow$ fundamental solution
(+ BCs at ∞)

$\Delta G(x) = 0$ ($x \neq 0$)

$\int \delta \, 1 = 1$



$$\int \Delta G \varphi \, dx = \int \delta \varphi \, dx = \varphi(0) \quad \varphi \in C^2$$

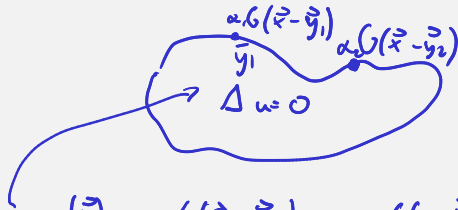
amenable to integration by parts

Applications: (Elliptic) PDE solving (I)

$$G(\vec{x}) = \frac{1}{4\pi} \frac{1}{\|\vec{x}\|_2}$$

$$\Delta u = \rho$$

Laplace: $\Delta u = 0$ (on Ω) $u = g$ (on $\partial\Omega$)



$$u(\vec{x}) = \alpha_1 G(\vec{x} - \vec{y}_1) + \alpha_2 G(\vec{x} - \vec{y}_2) + \dots$$

$$= \sum_i \alpha_i G(\vec{x} - \vec{y}_i)$$

point potential

layer potential

$$u(\vec{x}) = \int G(\vec{x} - \vec{y}) \alpha(y) d\vec{y}$$

Applications: (Elliptic) PDE solving (I)

$$\int \sigma(x) = \int G(x-y) \alpha(y) dS_y = g(x) \quad (x \in \partial \Omega)$$

$$\tilde{u}(x) = \int_{\Omega} G(x-\bar{y}) f(\bar{y}) d\bar{x}$$

volume potential

$$= G * f$$

$$\Delta(G * f) = f$$

$$\Delta \tilde{u}(x) = (\Delta G(x-\cdot)) * f$$

$$\Delta \tilde{u} = f$$

$$= (\delta(x-\cdot)) * f$$

$$u_{\text{poissol}} = \tilde{u} + \hat{u}$$

$$= f$$

$$\Delta \hat{u} = 0$$

$$g(x) = u_{\text{poissol}}(x) = \tilde{u} + \hat{u}$$

Applications: (Elliptic) PDE solving (II)



And one more thing...

Both multigrid and fast/IE schemes ultimately are $O(N)$ in the number of degrees of freedom N .



Survey

- ▶ Home dept
- ▶ Degree pursued
- ▶ Longest program ever written
 - ▶ in Python?
- ▶ Research area
- ▶ Interest in PDE solvers

Class web page

<https://tinyurl.com/fastalg-f26>

contains:

- ▶ Class outline
- ▶ Notes
- ▶ Demos
- ▶ Assignments
- ▶ Discussion forum
- ▶ Grading
- ▶ Video

Open Source <3

These notes (and the accompanying demos) are open-source!

Bug reports and pull requests welcome:

<https://github.com/inducer/fast-alg-ie-notes>

Copyright (C) 2013 – 26 Andreas Kloeckner

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Sources

- ▶ [Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions](#) by Halko/Martinsson/Tropp
- ▶ Carrier, Greengard, Rokhlin: [A Fast Adaptive Multipole Algorithm for Particle Simulations](#)
- ▶ Rainer Kress: [Linear integral equations](#). (second edition)
- ▶ David Colton and Rainer Kress: [Inverse Acoustic and Electromagnetic Scattering Theory](#). (3rd edition)

Outline

Introduction

Dense Matrices and Computation

Tools for Low-Rank Linear Algebra

Rank and Smoothness

Near and Far: Separating out High-Rank Interactions

Outlook: Building a Fast PDE Solver

Going Infinite: Integral Operators and Functional Analysis

Singular Integrals and Potential Theory

Boundary Value Problems

Back from Infinity: Discretization

Computing Integrals: Approaches to Quadrature

Going General: More PDEs

Matvec: A Slow Algorithm

Matrix-vector multiplication: our first 'slow' algorithm.
 $O(N^2)$ complexity.

$$\beta_i = \sum_{j=1}^N A_{ij} \alpha_j$$

Assume A dense.

Matrices and Point Interactions

$$A_{ij} = G(x_i, y_j)$$

Handwritten annotations above the equation:
- Above i : row
- Above j : col
- Above x_i : row
- Above y_j : col.

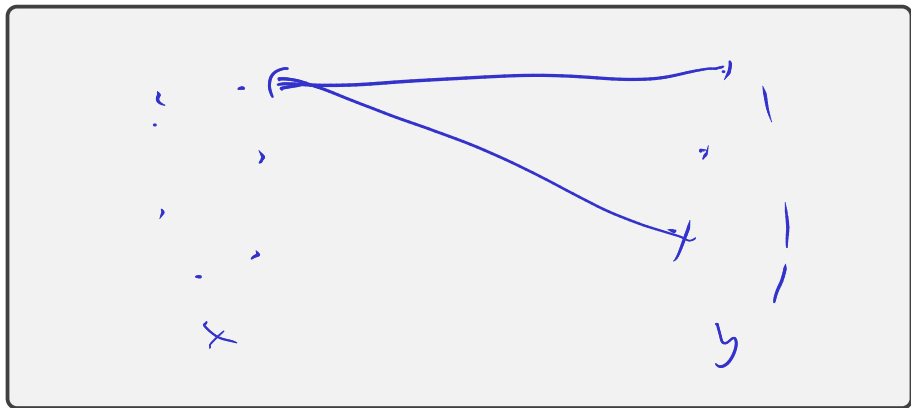
Does that actually change anything?

$\vec{x}_i$: targets
 $\vec{y}_j$: sources
 G : kernel

Matrices and Point Interactions

$$A_{ij} = G(x_i, y_j)$$

Graphically, too:



Matrices and point Interactions

$$\psi(x_i) = \sum_{j=1}^N G(x_i, y_j) \varphi(y_j)$$

This *feels* different.

Q: Are there enough matrices that come from globally defined G to make this worth studying?

Point Interaction Matrices: Examples (I)

- (Lagrange) interpolation

$$\chi(x) = \sum_{j=1}^N \ell_j(x) \varphi(y_j)$$

Interpolation error

- Numerical diff

$$\chi(x) = \sum_{j=1}^N \ell_j'(x) \varphi(y_j)$$

- Quadrature

$$\chi(x) = \dots$$

Point Interaction Matrices: Examples (II)

- Potential kernels
- Fourier
- Spectral fraction transforms
(e.g. spherical harmonic)
- Convolutions

Point Interaction Matrices: Examples (III)



So yes, there are indeed lots of these things.

Integral Operators

Why did we go through the trouble of rephrasing matvecs as

$$\psi(x_i) = \sum_{j=1}^N G(x_i, y_j) \varphi(y_j)?$$


$$\psi(x) = \int G(x, y) \varphi(y) dy$$

Cheaper Matvecs

$$\psi(x_i) = \sum_{j=1}^N G(x_i, y_j) \varphi(y_j)$$

So what can we do to make evaluating this cheaper?



Fast Dense Matvecs

Consider

$$A_{ij} = u_i v_j,$$

let $\mathbf{u} = (u_i)$ and $\mathbf{v} = (v_j)$.

Can we compute $A\mathbf{x}$ quickly? (for a vector $\mathbf{x}$)

$$A = \mathbf{u} \mathbf{v}^T$$

$$A\vec{x} = (\mathbf{u} \mathbf{v}^T) \vec{x} = \mathbf{u} (\mathbf{v}^T \vec{x})$$

$$A\varphi = u(\vec{x}) \int v(\vec{y}) \vec{\varphi}(\vec{y}) d\vec{y}$$