Let us consider the boundary value problem

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

where Ω is a domain in two dimensions with boundary Γ.

How would you numerically determine v?

Boundary equation representation of a PDE

Let us represent the solution of the boundary value problem

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

as a double layer potential,

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

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

$$\text{(BIE)} \qquad \frac{1}{2}u(x) + \int_\Gamma \big(n(y) \cdot \nabla_y \log|x-y|\big) u(y)\, ds(y) = f(x), \qquad x \in \Gamma.$$

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

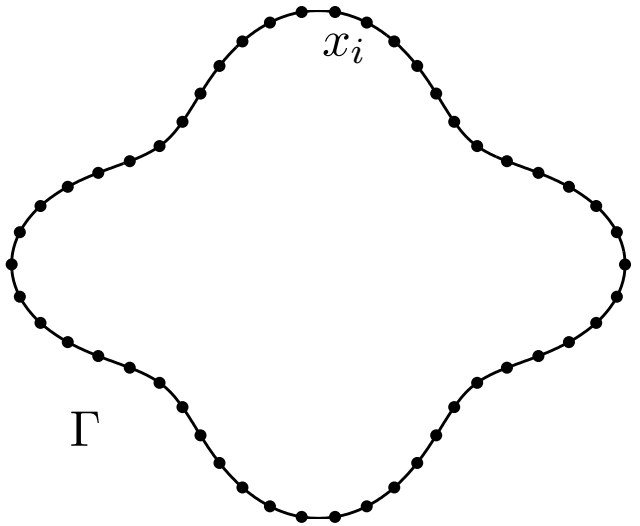
BIE AS FOUNDATION FOR NUMERICS

Suppose that we wish to numerically solve the integral equation

$$u(x) + \int_\Gamma K(x,y)u(y)\,ds(y) = f(x), \qquad x \in \Gamma.$$

We first discretize the contour into n points

$$\Gamma \sim [x_1, \ldots, x_n].$$



Then the operator

$$\int_\Gamma K(x,y)u(y)\,ds(y)$$

turns into a matrix A with entries (sort of)

$$A_{ij} = K(x_i, x_j), \qquad i,j = 1, \ldots, n.$$

Since A is **dense**, it appears that

the cost for constructing A is $O(n^2)$ (with a large constant),

the cost for solving $(I + A)u = f$ is $O(n^3)$.

Fast solution of boundary integral equations

We let A denote the dense $n \times n$ matrix discretizing the operator

$$\int_\Gamma K(x,y)u(y)\,ds(y).$$

There exist $O(n \log^q n)$ algorithms ($q = 0, 1, 2$) that evaluate the map

$$u \mapsto Au.$$

These include the Fast Multipole Method, Panel Clustering, multigrid, wavelets,...
Developed circa 1980 – 1985.

Using iterative (Krylov) methods, the equation

$$(I + A)u = f$$

can be solved using $O(\sqrt{\kappa} \cdot n \log^q n)$ operations, where κ is the condition number of $I + A$.

BIE formulations exist for many classical BVPs

Laplace $\quad -\Delta u = f,$

Elasticity $\quad \frac{1}{2} E_{ijkl} \left(\frac{\partial^2 u_k}{\partial x_l \partial x_j} + \frac{\partial^2 u_l}{\partial x_k \partial x_j} \right) = f_i,$

Stokes $\quad \Delta \mathbf{u} = \nabla p, \qquad \nabla \cdot \mathbf{u} = 0,$

Helmholtz $\quad (-\Delta - k^2) u = f,$

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

Maxwell $\quad \begin{cases} \nabla \cdot \mathbf{E} = \rho \qquad \nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \\ \nabla \cdot \mathbf{B} = 0 \qquad \nabla \times \mathbf{B} = \mathbf{J} + \frac{\partial \mathbf{E}}{\partial t} \end{cases}$

Ideally, we want numerical techniques that are ...

Accurate: The computational error should be roughly $\kappa\,\varepsilon$, where ε is the machine precision and κ is the actual condition number.

Efficient: The CPU time requirement should be roughly proportional to N, the actual complexity of the problem.

Robust: The computation should be black-box with no need for fine-tuning, parameter-selection, etc. In particular, delicate mesh-requirements currently form a major obstacle.

Under some conditions, methods exist that satisfy these criteria — FEM+multigrid, FFT, BIE + FMM, *et c.*

The technique of solving boundary value problems (BVP) via fast iterative techniques for solving boundary integral equations (BIE) works best when

(1) there is no body force,

(2) the differential operator has constant coefficients,

(3) the BVP is linear,

(4) the BIE that is used is well-conditioned.

Over the last 20 years, much progress has been made in extending the technique to overcome the apparent obstreperousness of problems violating conditions (1), (2) and (3).

However, to overcome (4), we need a ...

direct solver.

The primary research goal is to develop an $O(N)$ direct solver for boundary integral operators.

To be precise, we seek to construct an algorithm that computes the *inverse* of a boundary integral operator in $O(N)$ operations.

Benefits of a direct solver:

- **ill-conditioned problems**,
- multiple right-hand sides,
- up-dating a known solution to find the solution of another problem that is "close",
- constructing the SVD and other factorizations of the matrix.

Very practical aspect: People who develop general purpose software tend to shy away from iterative methods. They are perceived to be too delicate.

(Iterative methods sometimes require the use of pre-conditioners that need to be customized to particular problems.)

Direct methods are preferred, even when they are significantly slower, simply because of their robustness and versatility.

The work has two distinctly different components:

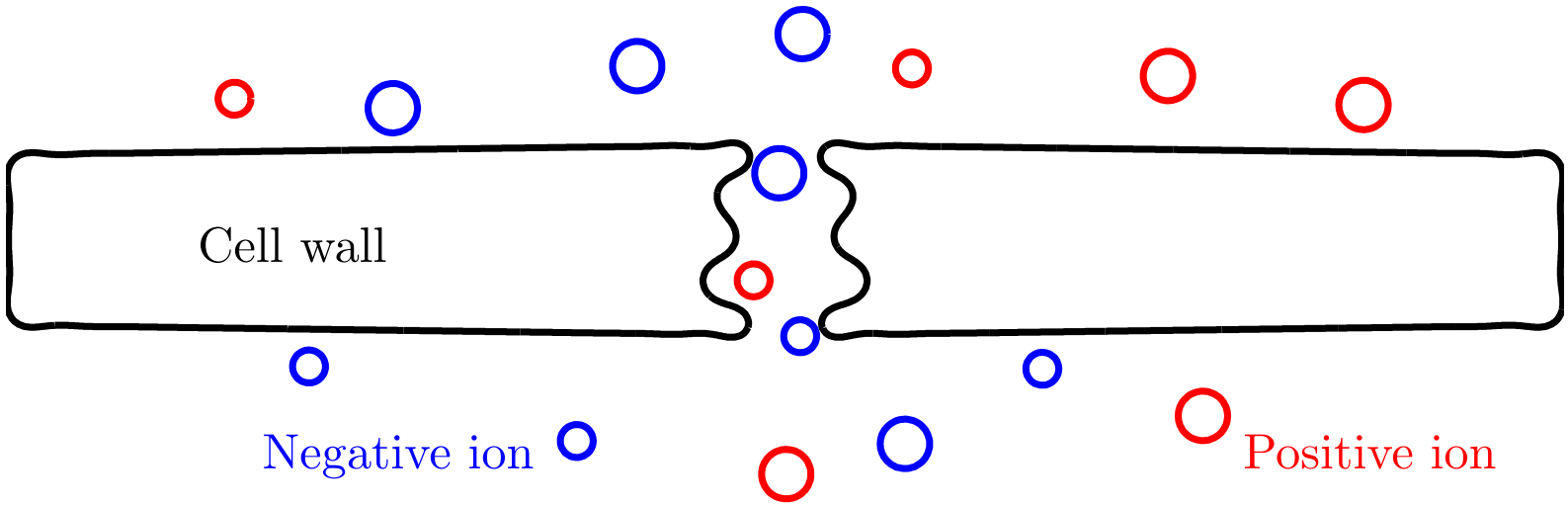
(A) Development of numerical methods – **direct inverter**, spectral decompositions, representation of functions, *etc*

(B) Adaptation of new numerical methods to application areas.
New tools $\Rightarrow$ we can solve problems that could not previously be solved.

1. Simulations of ionic solutions. Modelling of ion channels.
2. Multiscale modelling – composite materials, emulsions, ...
3. Electro-magnetic scattering problems.
4. Modelling of lattices – atomic crystals, large truss structures, ...

Application 1: Simulating fatty molecules in ionic solutions:

Modelling of an ion channel in a cell wall:



The cell walls have a **low** dielectric constant ($\varepsilon \approx 2 - 5$).

The water has a **large** dielectric constant ($\varepsilon \approx 80$).

The circles represent ions (blue are positive, red are negative).

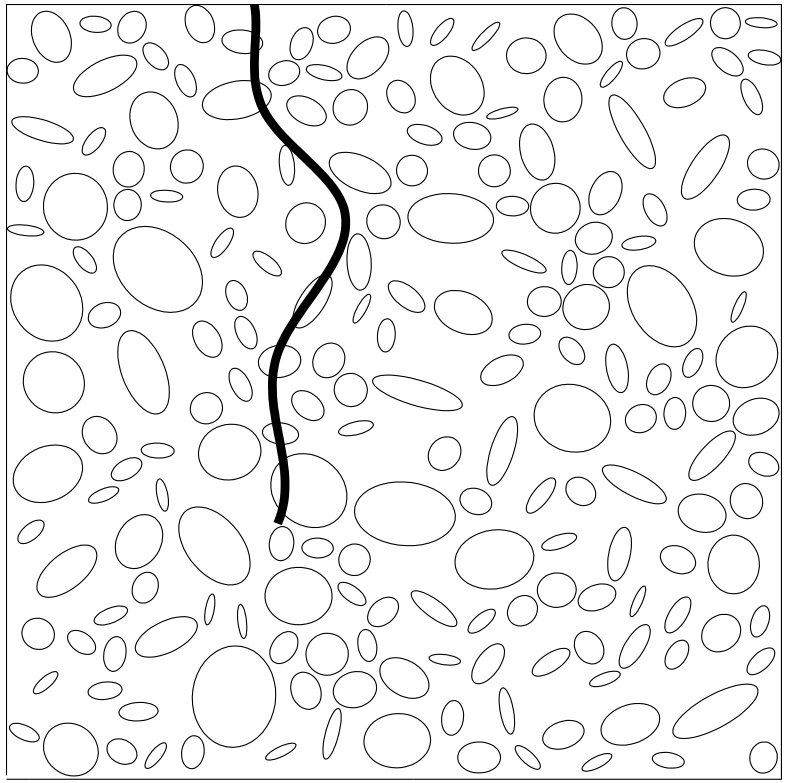
Our task: Determine the electro-static forces on all particles.

The object is to perform a molecular dynamics simulation of the ion channel.

A large number of time-steps (10^4 – 10^8) is required in a simulation.

This task is very well suited for direct methods.

Application 2: Modelling of composite materials / multiscale modelling.



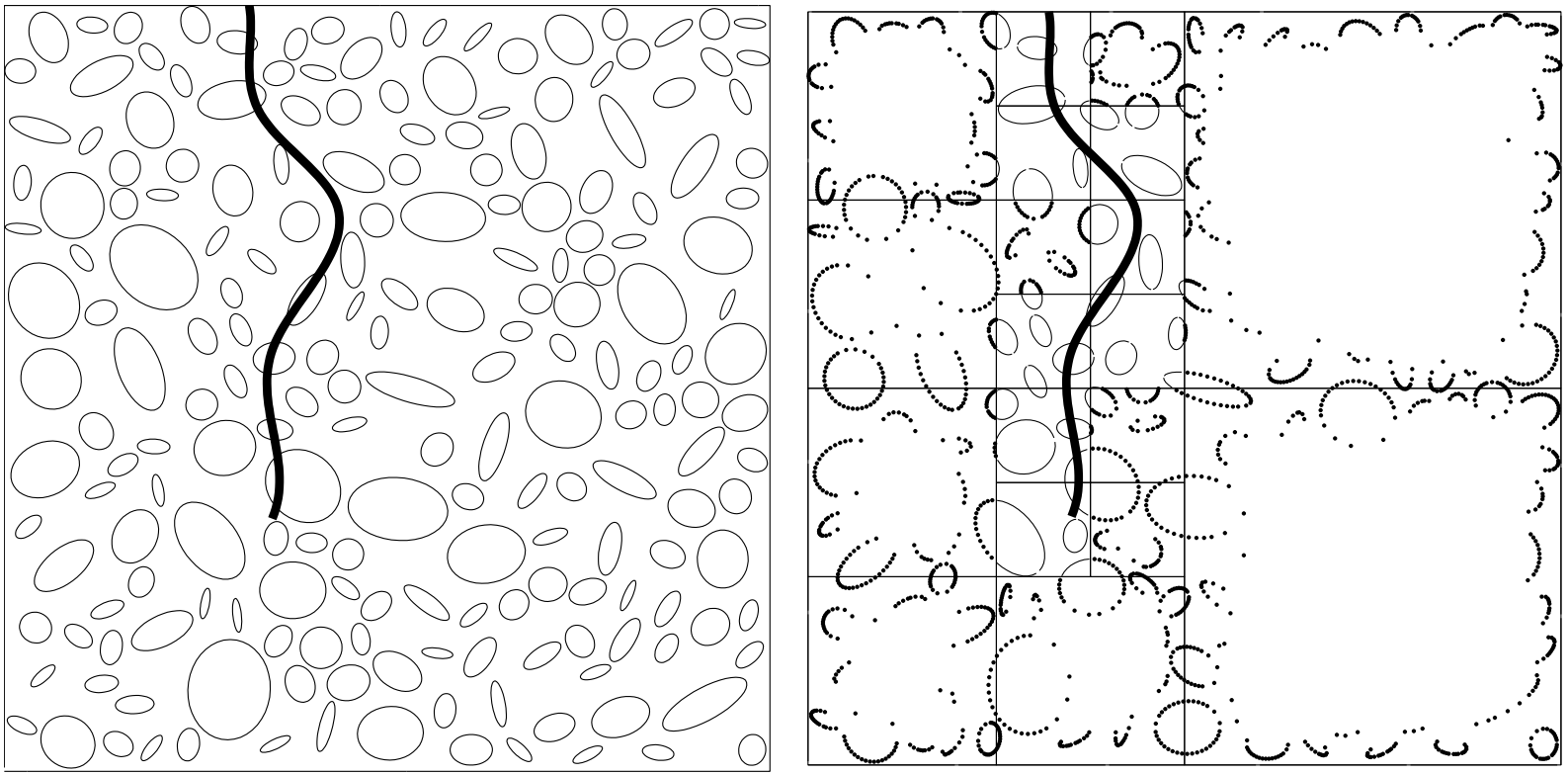
A composite material with a crack is loaded at the edges.

The task here is *model reduction*.

We do not want to fully resolve the micro-structure away from the crack.

Can we replace it with some averaged, equivalent, model?

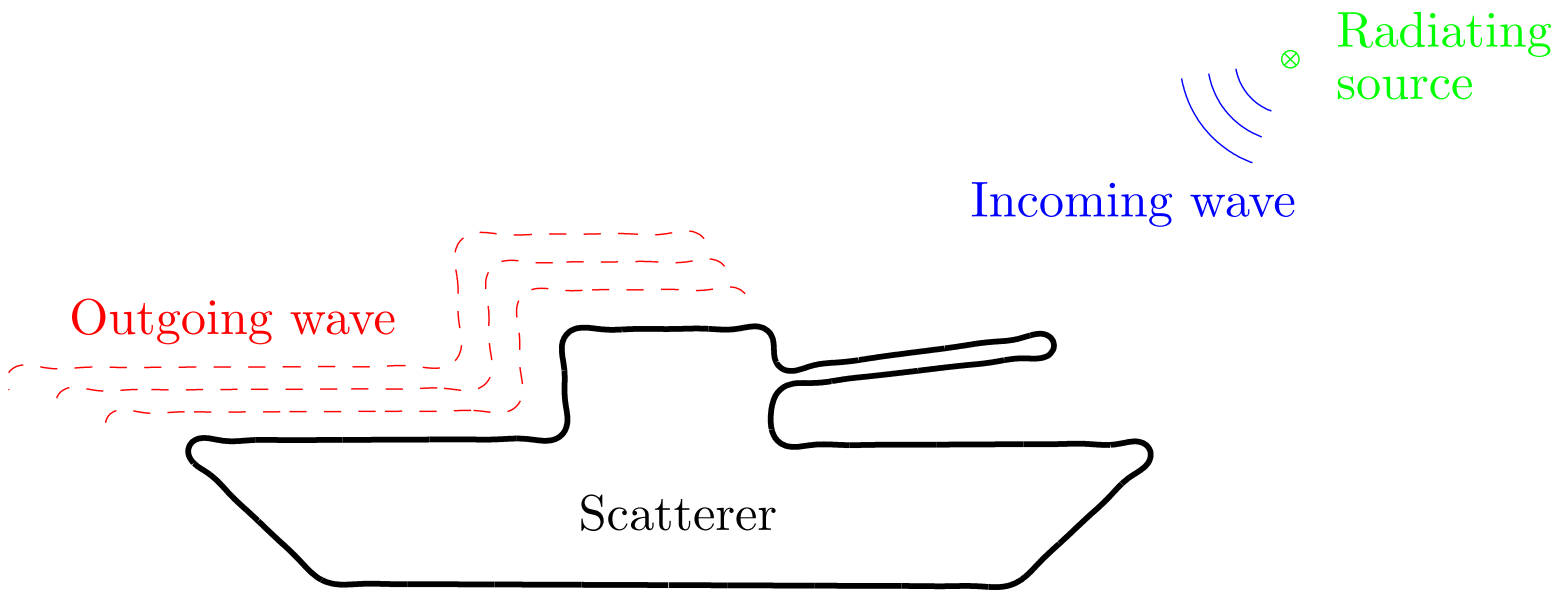
Yes, the direct solver does it automatically!



We need to develop a systematic method for constructing libraries of pre-computed patches.

Statistical methods must be used.

Application 3: The forwards scattering problem:
Given a scatterer, and an incoming wave, determine the reflected wave.



This problem arises in a range of applications: Determination of the radar cross section of an aircraft. Medical imaging. Particle accelerators.

This problem really should be solved using boundary integral equations. For electrically large objects, iterative methods do not work well.

Much harder problem:

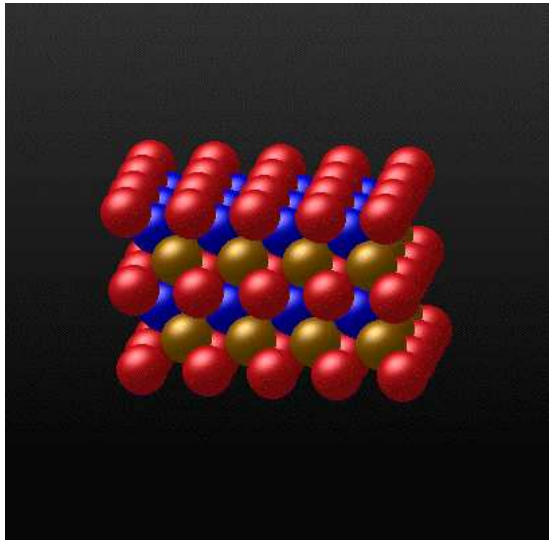
Inverse scattering:
Determine the shape of the scatterer given the outgoing wave.

Solving this problem almost surely involves solving the forwards scattering problem first.

Benefits of solving this problem include large amounts of money and various prizes.

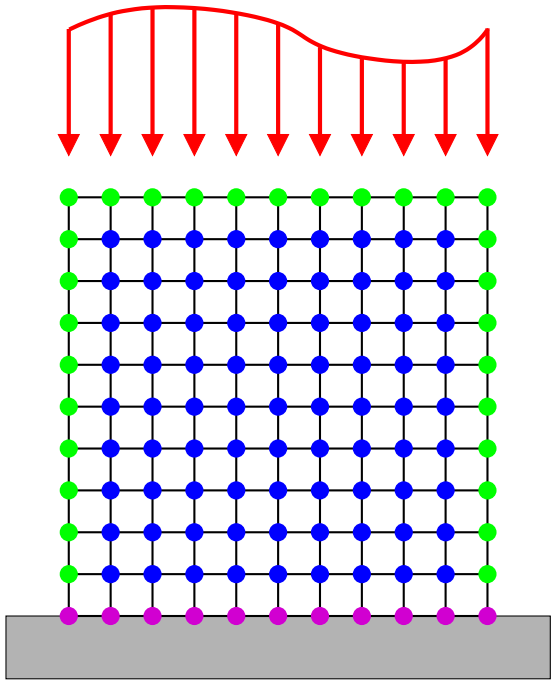
Application 4: Modelling of periodic lattices.

Some problems are naturally modelled by difference equations on $\mathbb{R}^n$:



Examples include atomic lattices, large mechanical truss structures, and many natural materials.

Boundary value problems on lattices:



The domain is $\Omega \subset \mathbb{Z}^2$.
Split $\Omega = \Omega_\mathrm{i} \cup \Gamma_\mathrm{D} \cup \Gamma_\mathrm{N}$, where

Ω_i are the interior nodes,

Γ_D is the Dirichlet boundary,

Γ_N is the Neumann boundary.

We can now formulate the discrete boundary value problem

$$\begin{cases} \boldsymbol{A}\boldsymbol{u} = \boldsymbol{f}, & \text{on } \Omega_\mathrm{i}, \\ \boldsymbol{u} = \boldsymbol{g}, & \text{on } \Gamma_\mathrm{D}, \\ \partial_\nu \boldsymbol{u} = \boldsymbol{h}, & \text{on } \Gamma_\mathrm{N}, \end{cases}$$

where $\boldsymbol{A}$ is the discrete Laplace operator,

$$\begin{aligned} [\boldsymbol{A}\boldsymbol{u}](m) = \quad & 4\boldsymbol{u}(m_1, m_2) - \boldsymbol{u}(m_1 - 1, m_2) - \boldsymbol{u}(m_1 + 1, m_2) \\ & - \boldsymbol{u}(m_1, m_2 - 1) - \boldsymbol{u}(m_1, m_2 + 1). \end{aligned}$$

It turns out that much of the machinery we have discussed for the continuum case can be adapted to the case of discrete structures.

- The lattice equation has a fundamental solution.
- It is possible to rewrite the BVP as a boundary equation.
- The boundary equation is very well conditioned.
- There exist $O(N)$ solvers for the boundary equation.
- ...

(This was my dissertation.)

To summarize:

The main goal is to construct an $O(N)$ algorithm for inverting boundary integral operators. This may turn out to be the reference method for solving linear boundary value problems.

We have new tools, these can be applied to a number of application areas:

- Electro-magnetic scattering problems.
- Modelling of lattices – atomic crystals, large truss structures, ...
- Multiscale modelling – composite materials, emulsions, ...
- Simulations of ionic solutions. Modelling of ion channels.
- ???

There are also possibilities for doing research on modelling of financial markets – in particular, statistical simulations of bond portfolios.