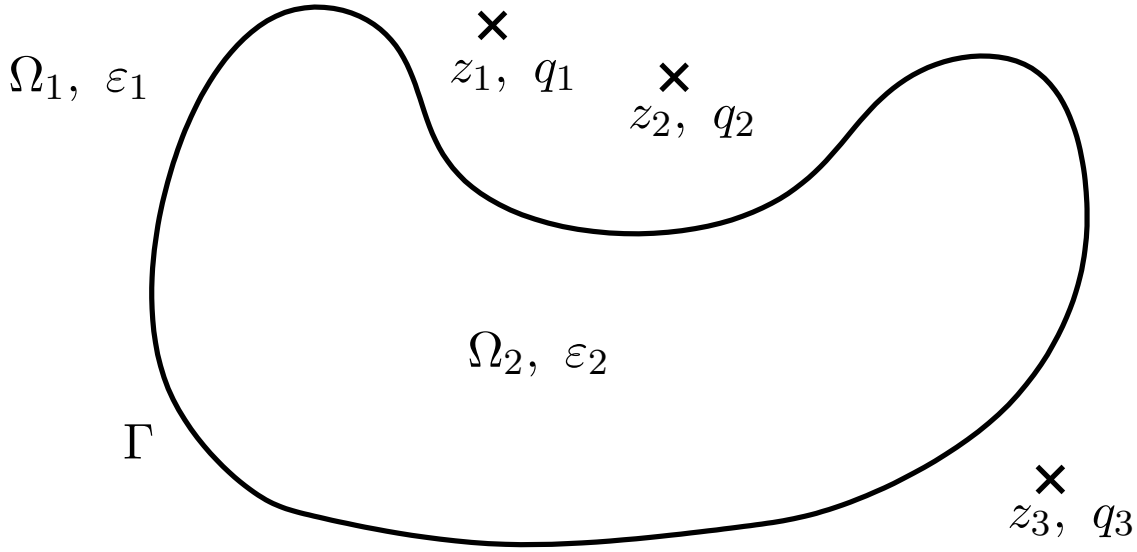
Rapid evaluation of electrostatic interactions in multi-phase media

P.G. Martinsson, The University of Colorado at Boulder

Acknowledgements:
Some of the work presented is joint work with Mark Tygert and Vladimir Rokhlin at Yale University.
Bob Eisenberg and Dirk Gillespie of the Rush University Medical Center have spent much effort explaining aspects of biochemical modeling to the author.

Problem formulation:



The domain $\Omega = \mathbb{R}^2 = \Omega_1 \cup \Omega_2$.

The interface is called Γ.

In the sub-domain Ω_i, the dielectric constant is ε_i.

In Ω_1, there are particles of charge q_i at positions z_i.

Let $u^{(i)}$ denote the electric potential in Ω_i. Then $u^{(i)}$ satisfies

$$-\varepsilon_1 \Delta u^{(1)}(x) = \sum_{i=1}^{M} q_i \delta(x - z_i), \qquad x \in \Omega_1, \tag{1}$$

$$-\varepsilon_2 \Delta u^{(2)}(x) = 0, \qquad x \in \Omega_2. \tag{2}$$

Letting n denote the outward unit normal of the interface $\Gamma = \partial\Omega_2$, the boundary conditions on Γ read

$$u^{(1)}(x) = u^{(2)}(x), \qquad x \in \Gamma, \tag{3}$$

$$\varepsilon_1 \frac{\partial u^{(1)}}{\partial n}(x) = \varepsilon_2 \frac{\partial u^{(2)}}{\partial n}(x), \qquad x \in \Gamma. \tag{4}$$

The final condition on u is a decay condition at infinity ensuring that the solution has finite total energy. In two dimensions, the condition is that there must exist a constant c such that

$$\lim_{|x| \to \infty} \left| u^{(1)}(x) - c \log |x| \right| = 0. \tag{5}$$

... pictures of examples ...

Our "target applications" are:

- Modelling of proteins (and other large molecules) in ionic solutions:
  - Simulations of ion channels.
  - Simulation of diffusional processes to determine ligand-protein and protein-protein binding kinetics.
  - Implicit solvent molecular dynamics of biomolecules.
  - Solvation and binding energy calculations to determine ligand-protein and protein-protein equilibrium binding constants and aid in rational drug design.
  - Biomolecular titration studies.

- Modelling of semiconducting devices.

Note: In applications involving high charge concentrations, one must sometimes solve the non-linear Poisson-Boltzmann equation. We will not consider such cases here.

In biochemical applications, it appears that such problems are most commonly solved using FFT-accelerated Poisson solvers such as **DelPhi** by Barry Honig.

Advantages:

- Very fast.
- Well-established.
- Can handle non-linear equations.

Limitations:

- Regular grids only - no adaptivity.
- It is not easy to correctly discretize boundaries where the dielectric sharply changes, and it is unclear how accurately such programs solve the actual PDE.
- Likewise, point-charges cannot easily be introduced.
- Periodic boundary data only.

There are of course many other programs as well, include FEM-based ones such as APBS (Adaptive Poisson-Boltzmann Solver) by Nathan Baker, J. Andrew McCammon, and Michael Holst. These are better equipped to accurately solve the equations, but appear to be considered slow.

We propose an algorithm based on a combination of three tools:

(1) Reformulate the problem as a Boundary Integral Equation (BIE).

(2) Solve the BIE using fast **direct** methods that pre-compute an inverse to the integral operator. This takes advantage of the fact that in Monte-Carlo and Molecular Dynamics simulations, the electro-statics problem is solved many times on the same geometry (millions of times, or more).

(3) Use the Fast Multipole Method to rapidly evaluate all electrostatic interactions.

The resulting scheme has been tested in both 2D and 3D – results later.

PDE-solvers based on Boundary Integral Equations:

The idea is to convert a partial differential equation

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

to an "equivalent" integral equation

$$\text{(BIE)} \qquad v(x) + \int_\Gamma k(x,y)\,v(y)\,ds(y) = h(x), \qquad x \in \Gamma.$$

- The kernel k is derived from the operator A.
- The data function h is derived from the data of (BVP).
- The conversion from (BVP) to (BIE) sometimes involves the evaluation of certain integrals over Γ and/or Ω.
- Sometimes the integral equation must be formulated on Ω.
- ...

When integral equation formulations are available, there are compelling arguments in their favor, these include:

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.

There is a fundamental difficulty with using integral operators in numerics:

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 (the discretized integral operator) is dense, it is possible to evaluate the matrix-vector product

$$v \mapsto K_N\, v$$

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

A very succesful such algorithm is the Fast Multipole Method by Rokhlin and Greengard (circa 1985).

Combining such methods with iterative solvers (GMRES / conjugate gradient / ...) leads to very fast solvers for the integral equations, especially when second kind Fredholm formulations are used.

In order to derive an integral formulation of the dielectrics problem, we first decompose the solution u into two parts:

$$u = v + w. \tag{8}$$

Here, v is the potential caused directly by the point-charges,

$$v(x) = -\frac{1}{2\pi\varepsilon_1} \sum_{i=1}^{M} q_i \log|x - z_i|, \tag{9}$$

and w is a potential caused by an "induced" charge distribution σ,

$$w(x) = \int_\Gamma \left(-\frac{1}{2\pi\varepsilon_1} \log|x - y| \right) \sigma(y)\, ds(y). \tag{10}$$

The function u defined by (8) automatically satisfies

- The PDE in the domain, $\Delta u = 0$.
- Continuity across the dielectric interface.
- "Decay" at infinity.

It only remains to enforce continuity in the electric flow across Γ.

It remains to determine σ so that

$$\varepsilon_1 \frac{\partial u^{(1)}}{\partial n}(x) = \varepsilon_2 \frac{\partial u^{(2)}}{\partial n}(x),$$

Recall that

$$u(x) = \underbrace{-\frac{1}{2\pi\varepsilon_1} \sum_{i=1}^{M} q_i \log|x - z_i|}_{=v(x)} + \underbrace{\int_\Gamma \left(-\frac{1}{2\pi\varepsilon_1} \log|x-y| \right) \sigma(y)\, ds(y)}_{=w(x)}.$$

It is a well-known result that under moderate smoothness conditions on Γ, we have

$$\varepsilon_1 \frac{\partial u^{(1)}}{\partial n}(x) = \lim_{y \to x,\ y \in \Omega_1} \varepsilon_1 \nabla u(y) \cdot n(x) = \varepsilon_1 \frac{\partial v}{\partial n}(x) - \frac{1}{2}\sigma(x) + [D\sigma](x),$$

where D denotes the "double layer operator",

$$[D\sigma](x) = -\frac{1}{2\pi} \int_\Gamma \frac{n(x) \cdot (x-y)}{|x-y|^2} \sigma(y)\, ds(y).$$

Now, inserting

$$\varepsilon_1 \frac{\partial u^{(1)}}{\partial n}(x) = \varepsilon_1 \frac{\partial v}{\partial n}(x) - \frac{1}{2}\sigma(x) + [D\sigma](x),$$

and

$$\varepsilon_2 \frac{\partial u^{(2)}(x)}{\partial n} = \varepsilon_2 \frac{\partial v}{\partial n}(x) + \frac{\varepsilon_2}{\varepsilon_1}\frac{1}{2}\sigma(x) + \frac{\varepsilon_2}{\varepsilon_1}[D\sigma](x).$$

into

$$\varepsilon_1 \frac{\partial u^{(1)}}{\partial n}(x) = \varepsilon_2 \frac{\partial u^{(2)}}{\partial n}(x),$$

we obtain the equation

$$-\frac{\varepsilon_1 + \varepsilon_2}{2(\varepsilon_1 - \varepsilon_2)}\sigma(x) + [D\sigma](x) = -\varepsilon_1 \frac{\partial v}{\partial n}(x), \qquad x \in \Gamma. \tag{11}$$

We have constructed the following algorithm:

1. Evaluate the flows across Γ induced by the point charges:

$$\frac{dv(x)}{dn} = -\frac{1}{2\pi\varepsilon_1}\sum_{i=1}^{M} q_i \frac{n(x)\cdot(x-z_i)}{|x-z_i|^2}.$$

2. Determine the induced charges on the interface from the equation

$$-\frac{\varepsilon_1+\varepsilon_2}{2(\varepsilon_1-\varepsilon_2)}\sigma(x) + [D\sigma](x) = -\varepsilon_1\frac{\partial v}{\partial n}(x), \qquad x\in\Gamma.$$

3. Evaluate the forces caused by the induced charges,

$$F_i^{(\text{induced})} = -q_i\nabla w(z_i) = \frac{q_i}{2\pi\varepsilon_1}\int_\Gamma \frac{z_i-y}{|z_i-y|^2}\sigma(y)\,ds(y).$$

4. Evaluate the forces caused by particle-particle interactions,

$$F_i^{(\text{particles})} = \sum_{i\neq j}\frac{q_iq_j}{2\pi\varepsilon_1}\,\frac{z_i-z_j}{|z_i-z_j|^2}.$$

Steps 1, 3, and 4, will be accelerated using the Fast Multipole Method.

Step 2 will be accelerated using a fast direct solver.

What is a direct solver?

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, ...

Practical considerations:

Direct methods tend to be more **robust** than iterative ones.

This makes them more suitable for "black-box" implementations.

Commercial software developers appear to avoid implementing iterative solvers whenever possible. (Sometimes for good reasons.)

The effort to develop direct solvers should be viewed as a step towards getting a LAPACK-type environment for solving the basic linear boundary value problems of mathematical physics.

Sampling of related work:

1991 Sparse matrix algebra / wavelets, *Beylkin, Coifman, Rokhlin*,

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 $O(N^{3/2})$ inversion of Lippmann-Schwinger equations, *Y. Chen*,

2002 inversion of "Hierarchically semi-separable" matrices, *M. Gu, S. Chandrasekharan, et al*,

2004 $O(N)$ inversion of boundary integral equations in 2D, *Martinsson, Rokhlin*,

2007 $O(N \log N)$ inversion of 2D finite element matrices, *Martinsson*.

Current state of the research

The fast direct solvers currently being developed 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, such methods currently **can** handle:

- Laplace's equation, equations of elasticity, Yukawa's equation,...
- Helmholtz' and Maxwell's equations for low and intermediate frequencies. (In special cases, high frequency problem can also be solved.)

Current status:

2D	Fast inversion
	Extremely fast application of inverse
3D	Sluggish inversion
	Fast application of the inverse

Holy grail: Fast inversion scheme for high-frequency problems.

How does the inversion scheme for 2D boundary integral equations work?

Consider the linear system

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}.$$

We suppose that for $i \neq j$, the blocks A_{ij} allow the factorization

$$\underbrace{A_{ij}}_{n_i \times n_j} = \underbrace{U_i}_{n_i \times k_i} \underbrace{\tilde{A}_{ij}}_{k_i \times k_j} \underbrace{U_j^{\mathrm{t}}}_{k_j \times n_j},$$

where the ranks k_i are significantly smaller than the block sizes n_i.

We then let

$$\underbrace{\tilde{q}_i}_{k_i \times 1} = U_i^{\mathrm{t}} \underbrace{q_i}_{n_i \times 1},$$

be the variables of the "reduced" system.

Recall: $A_{ij} = U_i \, \tilde{A}_{ij} \, U_j^{\mathrm{t}}$ and $\tilde{q}_i = U_i^{\mathrm{t}} \, q_i$.

The system $\sum_j A_{ij} \, q_j = v_i$ then takes the form

$$\left[\begin{array}{cccc|cccc} A_{11} & 0 & 0 & 0 & 0 & U_1\tilde{A}_{12} & U_1\tilde{A}_{13} & U_1\tilde{A}_{14} \\ 0 & A_{22} & 0 & 0 & U_2\tilde{A}_{21} & 0 & U_2\tilde{A}_{23} & U_2\tilde{A}_{24} \\ 0 & 0 & A_{33} & 0 & U_3\tilde{A}_{31} & U_3\tilde{A}_{32} & 0 & U_3\tilde{A}_{34} \\ 0 & 0 & 0 & A_{44} & U_4\tilde{A}_{41} & U_4\tilde{A}_{42} & U_4\tilde{A}_{43} & 0 \\ \hline -U_1^{\mathrm{t}} & 0 & 0 & 0 & I & 0 & 0 & 0 \\ 0 & -U_2^{\mathrm{t}} & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & -U_3^{\mathrm{t}} & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & -U_4^{\mathrm{t}} & 0 & 0 & 0 & I \end{array}\right] \left[\begin{array}{c} q_1 \\ q_2 \\ q_3 \\ q_4 \\ \tilde{q}_1 \\ \tilde{q}_2 \\ \tilde{q}_3 \\ \tilde{q}_4 \end{array}\right] = \left[\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ 0 \\ 0 \\ 0 \\ 0 \end{array}\right].$$

Now form the Schur complement to eliminate the q_j's.

After eliminating the "fine-scale" variables q_i, we obtain

$$\begin{bmatrix} I & U_1^{\mathrm{t}}\tilde{A}_{11}^{-1}U_1\tilde{A}_{12} & U_1^{\mathrm{t}}\tilde{A}_{11}^{-1}U_1\tilde{A}_{13} & U_1^{\mathrm{t}}\tilde{A}_{11}^{-1}U_1\tilde{A}_{14} \\ U_2^{\mathrm{t}}\tilde{A}_{22}^{-1}U_2\tilde{A}_{21} & I & U_2^{\mathrm{t}}\tilde{A}_{22}^{-1}U_2\tilde{A}_{23} & U_2^{\mathrm{t}}\tilde{A}_{22}^{-1}U_2\tilde{A}_{24} \\ U_3^{\mathrm{t}}\tilde{A}_{33}^{-1}U_3\tilde{A}_{31} & U_3^{\mathrm{t}}\tilde{A}_{33}^{-1}U_3\tilde{A}_{32} & I & U_3^{\mathrm{t}}\tilde{A}_{33}^{-1}U_3\tilde{A}_{34} \\ U_4^{\mathrm{t}}\tilde{A}_{44}^{-1}U_4\tilde{A}_{41} & U_4^{\mathrm{t}}\tilde{A}_{44}^{-1}U_4\tilde{A}_{42} & U_4^{\mathrm{t}}\tilde{A}_{44}^{-1}U_4\tilde{A}_{43} & I \end{bmatrix} \begin{bmatrix} \tilde{q}_1 \\ \tilde{q}_2 \\ \tilde{q}_3 \\ \tilde{q}_4 \end{bmatrix} = \begin{bmatrix} U_1^{\mathrm{t}}A_{11}^{-1}\,v_1 \\ U_2^{\mathrm{t}}A_{22}^{-1}\,v_2 \\ U_3^{\mathrm{t}}A_{33}^{-1}\,v_3 \\ U_4^{\mathrm{t}}A_{44}^{-1}\,v_4. \end{bmatrix}$$

We set

$$\tilde{A}_{ii} = \left(U_i^{\mathrm{t}}\,A_{ii}^{-1}\,U_i\right)^{-1},$$

and multiply line i by $\tilde{A}_{ii}$ to obtain the reduced system

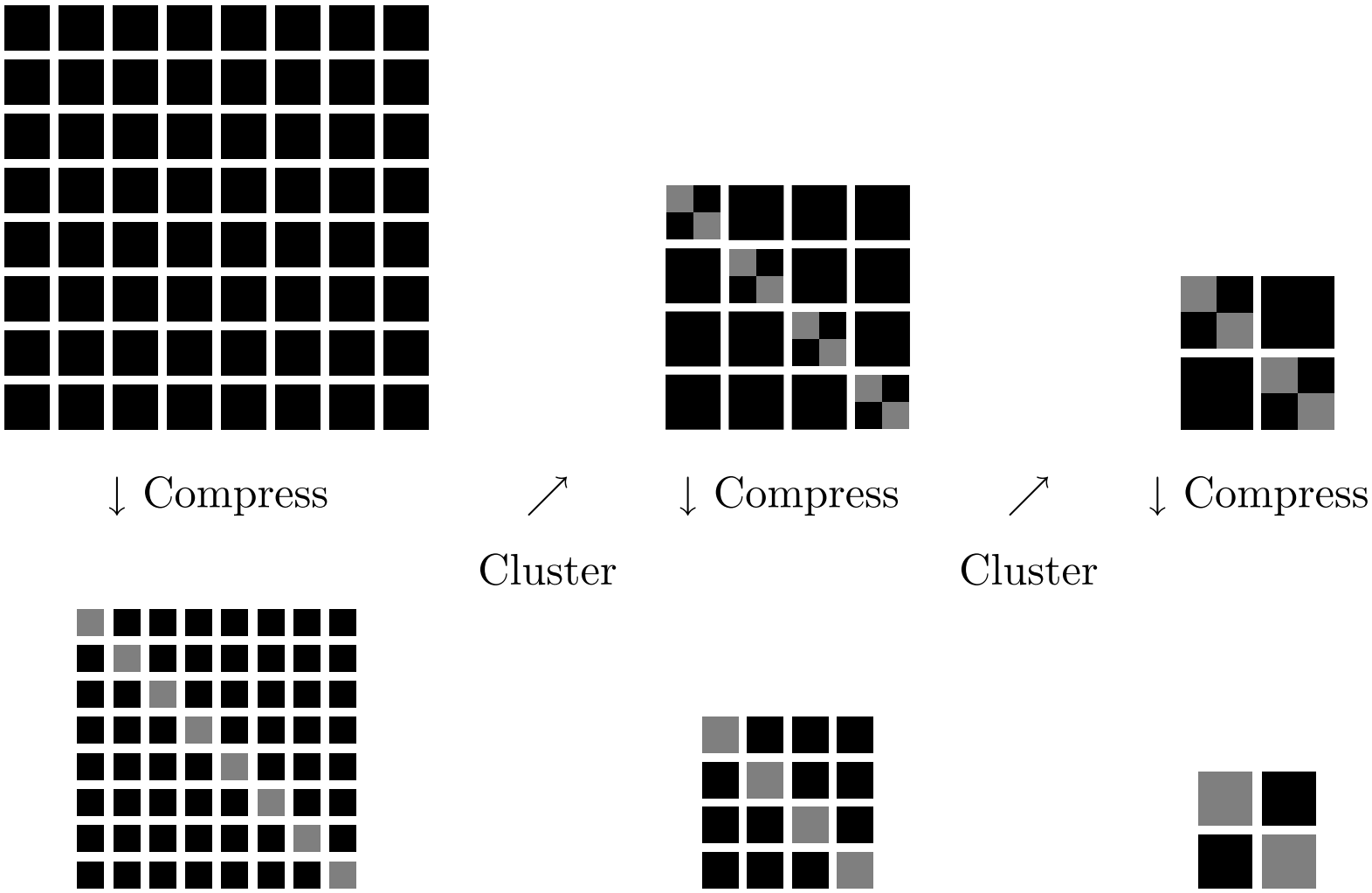
$$\begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} & \tilde{A}_{13} & \tilde{A}_{14} \\ \tilde{A}_{21} & \tilde{A}_{22} & \tilde{A}_{23} & \tilde{A}_{24} \\ \tilde{A}_{31} & \tilde{A}_{32} & \tilde{A}_{33} & \tilde{A}_{34} \\ \tilde{A}_{41} & \tilde{A}_{42} & \tilde{A}_{43} & \tilde{A}_{44} \end{bmatrix} \begin{bmatrix} \tilde{q}_1 \\ \tilde{q}_2 \\ \tilde{q}_3 \\ \tilde{q}_4 \end{bmatrix} = \begin{bmatrix} \tilde{v}_1 \\ \tilde{v}_2 \\ \tilde{v}_3 \\ \tilde{v}_4 \end{bmatrix}.$$

where

$$\tilde{v}_i = \tilde{A}_{ii}\,U_i^{\mathrm{t}}\,A_{ii}^{-1}\,v_i.$$

(This derivation was pointed out by Leslie Greengard.)

A globally $O(N)$ algorithm is obtained by hierarchically repeating the process:



The one-level coarse-graining involves the following steps:

- Compute U_i and $\tilde{A}_{ij}$ so that $A_{ij} = U_i \, \tilde{A}_{ij} \, U_j^{\mathrm{t}}$.

- Compute the new diagonal matrices

$$\tilde{A}_{ii} = \left(U_i^{\mathrm{t}} \, A_{ii}^{-1} \, U_i \right)^{-1},$$

- Compute the new loads

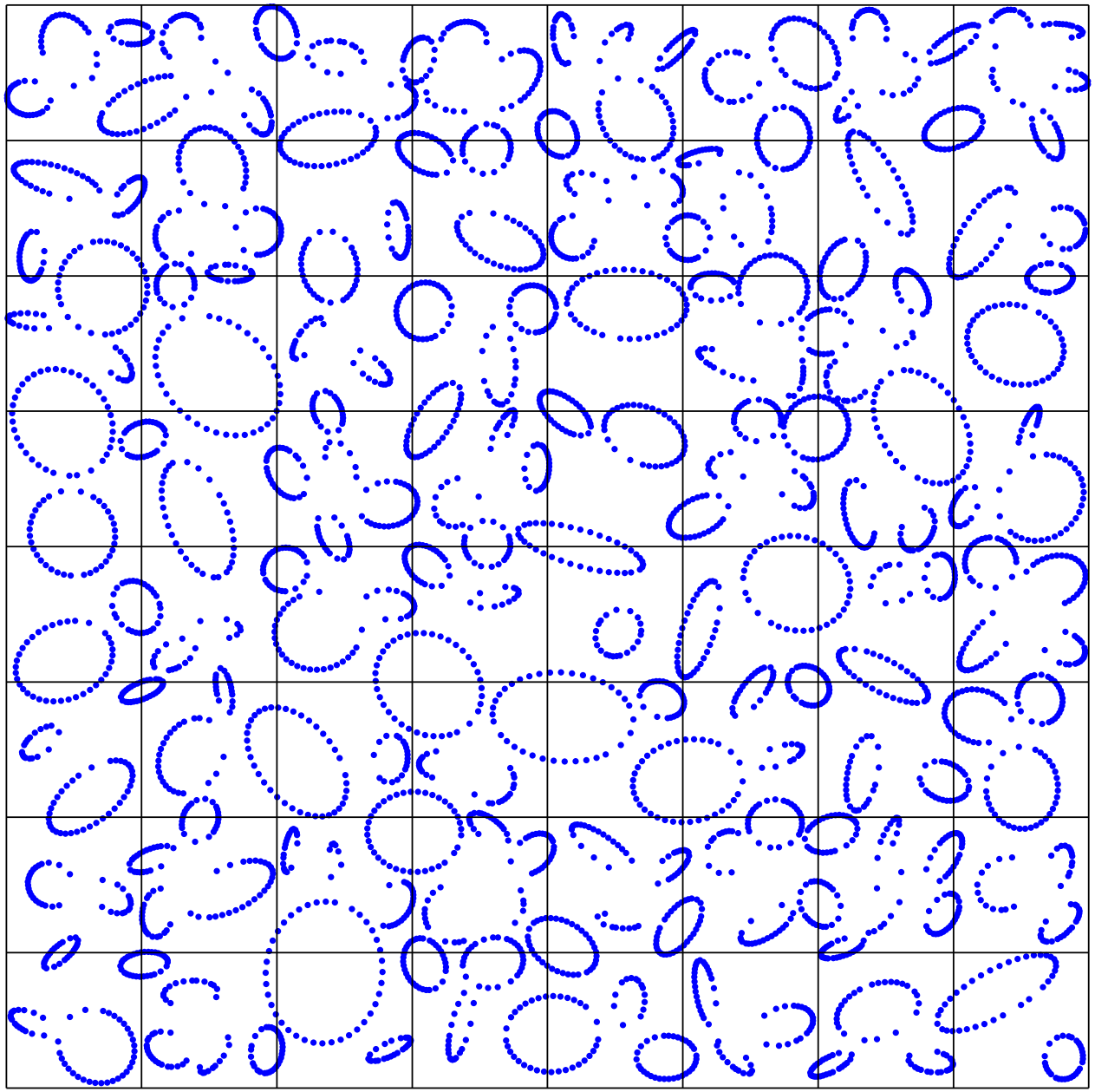
$$\tilde{v}_i = \tilde{A}_{ii} \, U_i^{\mathrm{t}} \, A_{ii}^{-1} \, v_i.$$

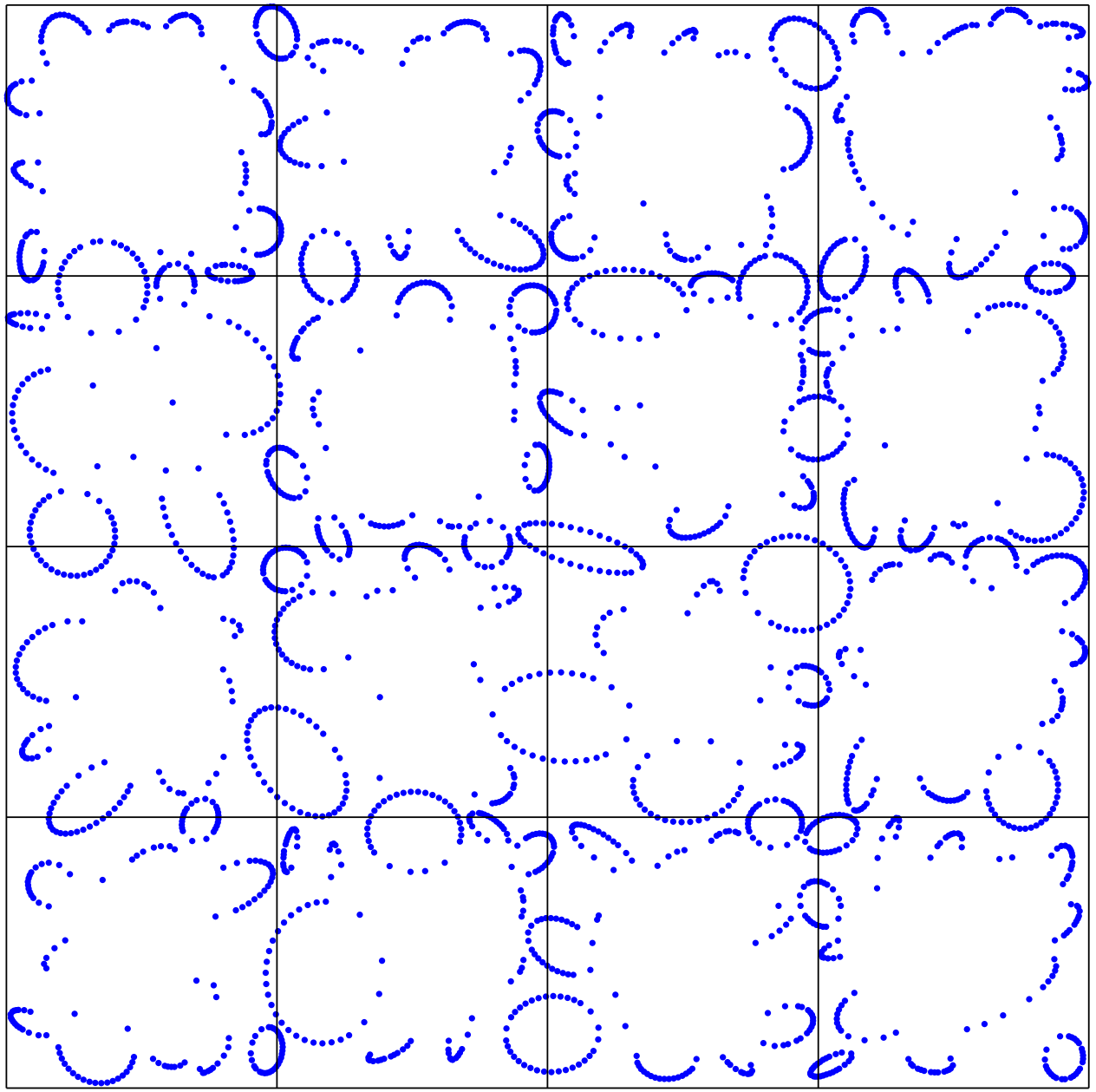
For the algorithm to be $O(N)$, it has to be able to carry out these steps *locally*.

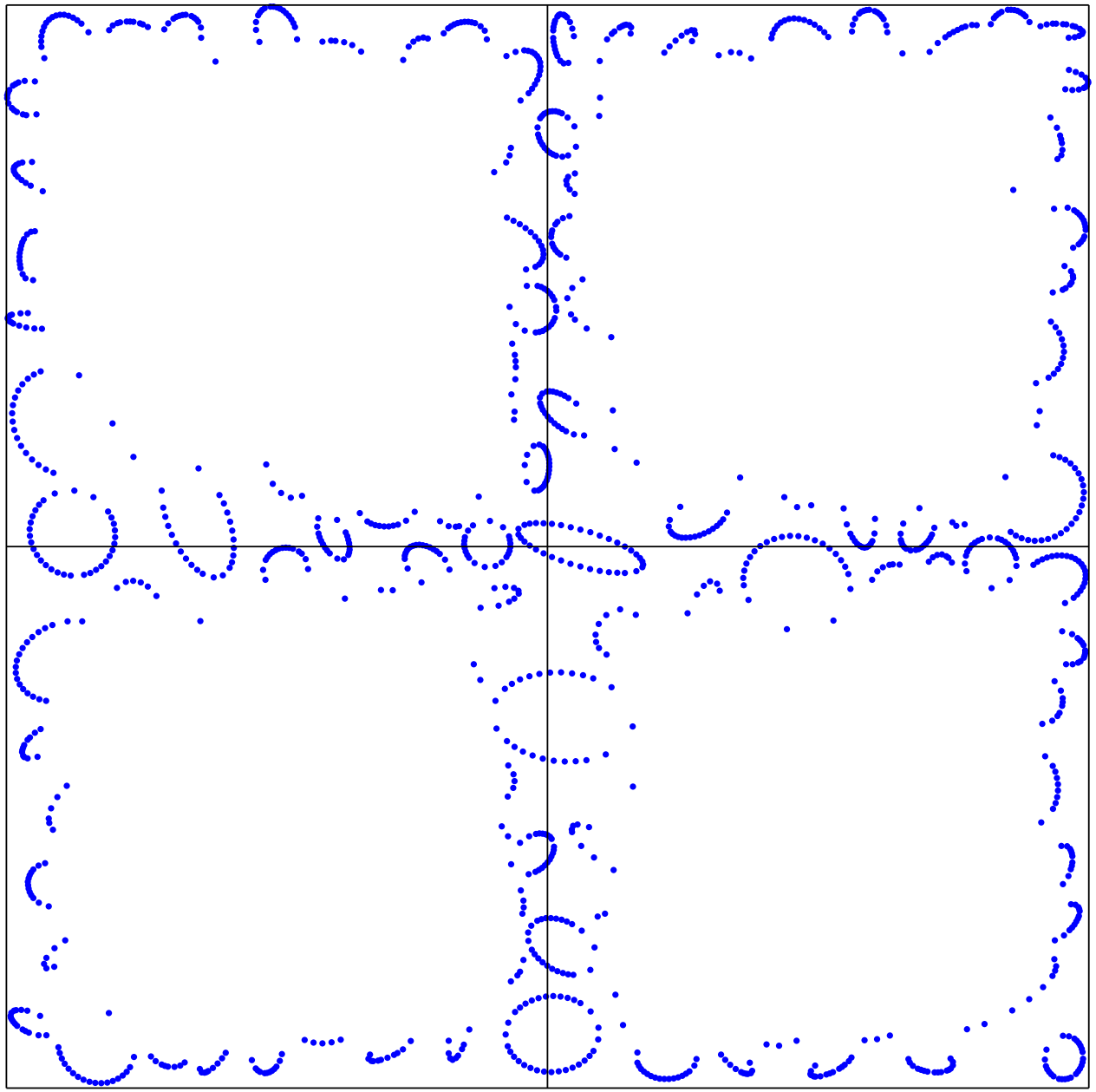
To achieve this, we use interpolative representations.

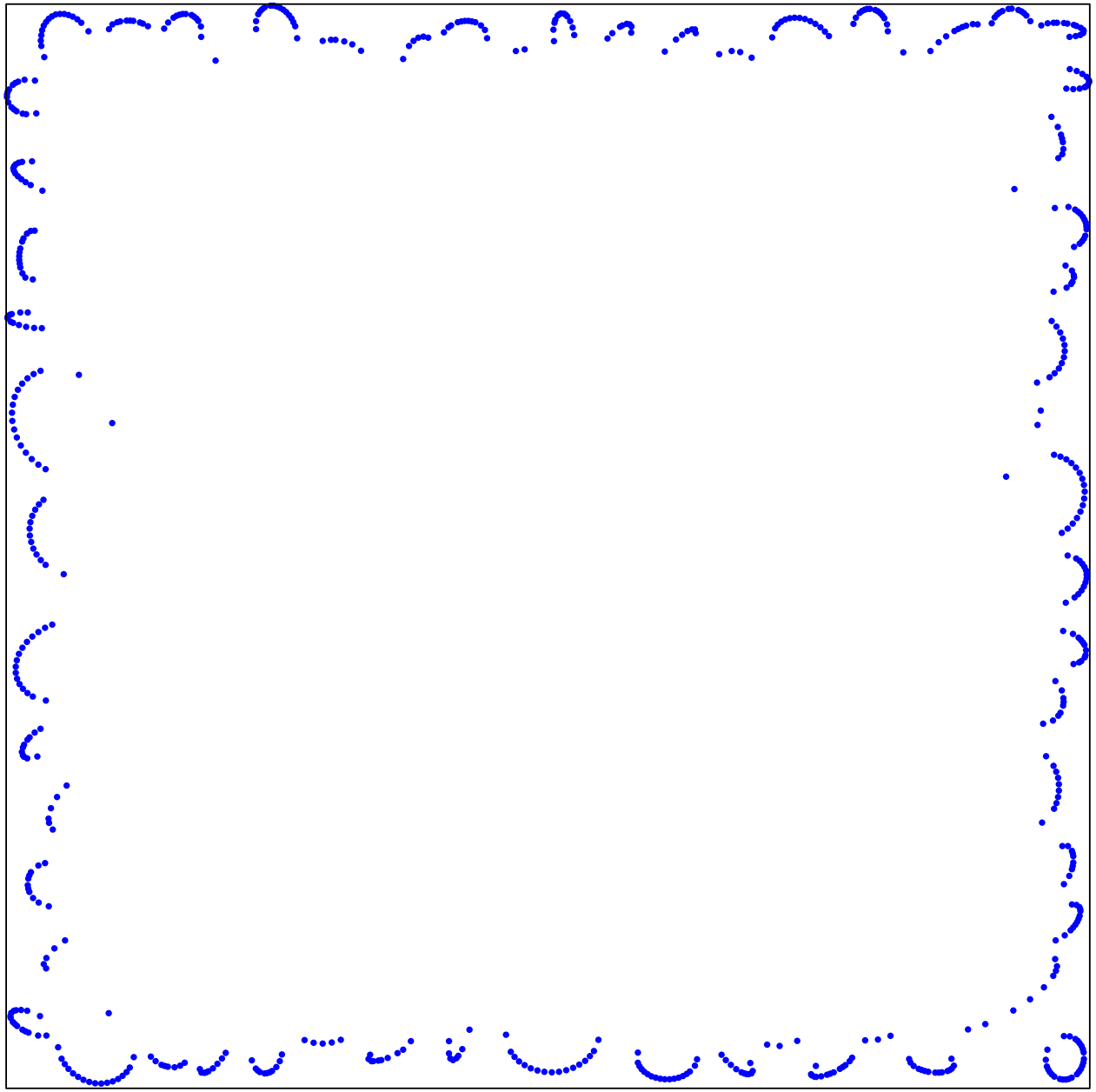
$\tilde{A}_{ij}$ will be a submatrix of A_{ij}, so it will not need to be computed.

To obtain a globally $O(N)$ scheme, we hierarchically merge proxy matrices.







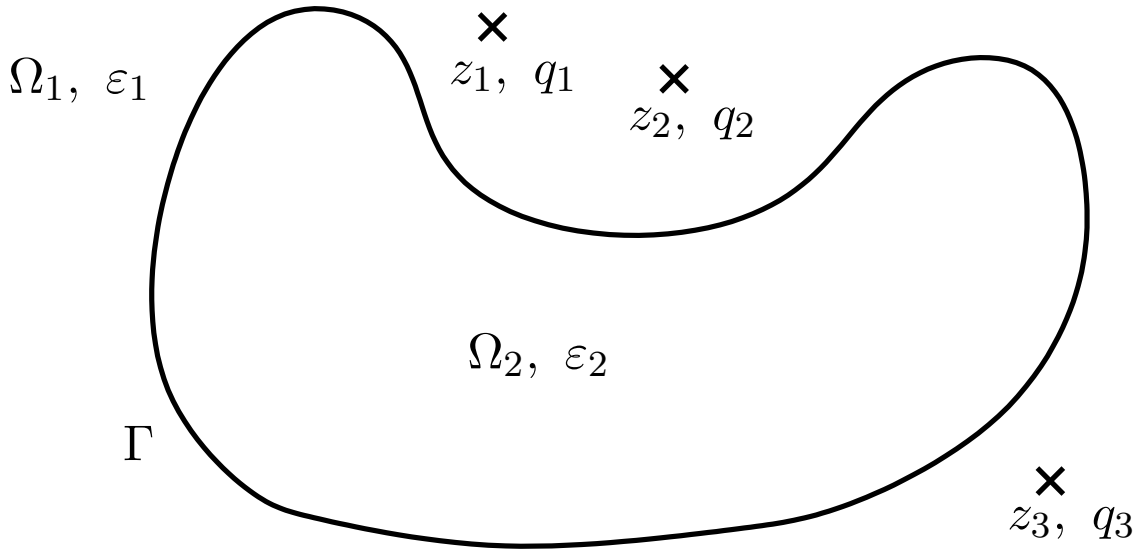


Numerical examples

In developing direct solvers, the "proof is in the pudding" — recall that from a theoretical point of view, the problem is already solved (by Hackbusch and others).

All computations were performed on standard laptops and desktop computers in the 2.0GHz - 2.8Ghz speed range, and with 512Mb of RAM.

Example 1: Electro-statics in a bi-phase dielectric medium



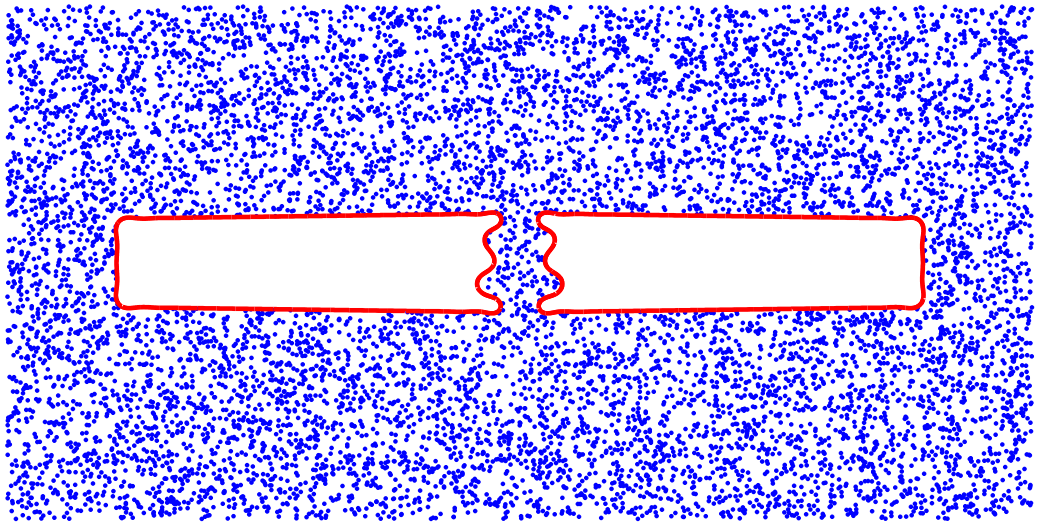
The domain Ω_2 has a different dielectric constant than the background.

There are electric charges at the points $z_i \in \Omega_1$.

Task: Evaluate all electro-static forces on the point-charges.

The "induced" charge distribution σ on Γ satisfies, for $x \in \Gamma$,

$$\frac{\varepsilon_1 + \varepsilon_2}{2(\varepsilon_1 - \varepsilon_2)}\sigma(x) - \frac{1}{2\pi}\int_\Gamma \frac{n(y)\cdot(x-y)}{|x-y|^2}\sigma(y)\,ds(y) = -\varepsilon_1 \frac{\partial v}{\partial n}(x),$$



Computational accuracy $= 10^{-10}$.

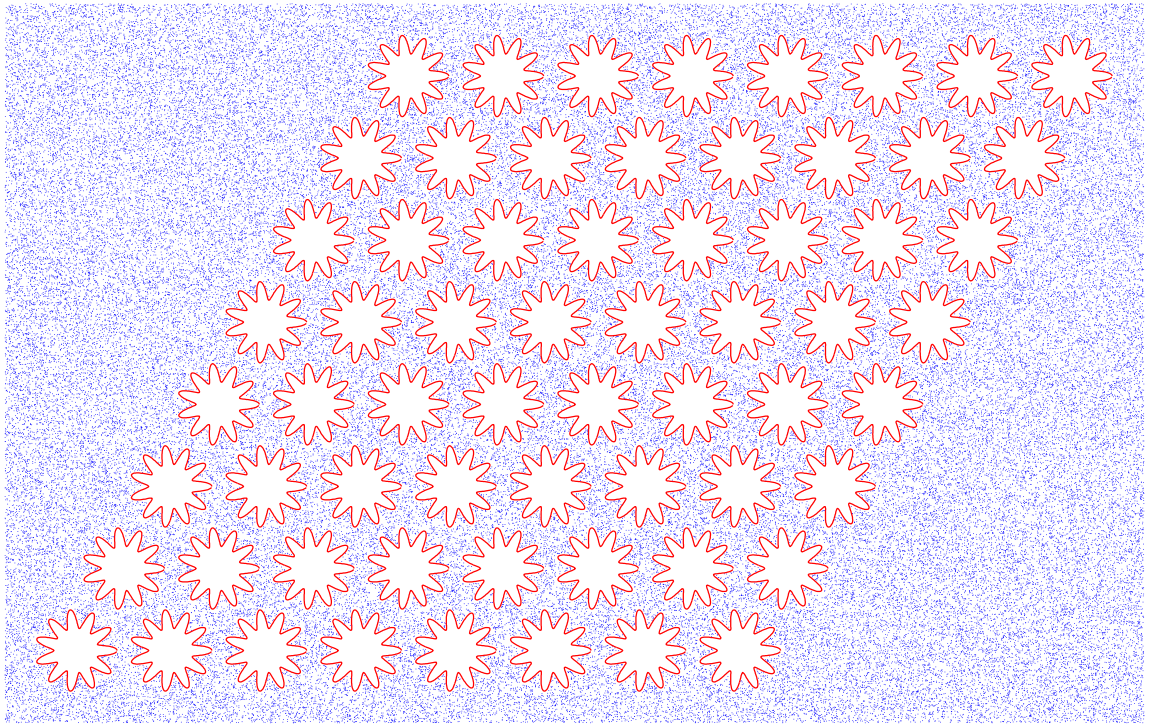
Nr. of points in the discretization of the contour $= 25\,600$.

Nr. of particles $= 10\,000$.

Time to invert the boundary integral equation $= 2.9$sec.

Time to compute the induced charges $= 0.034$sec. (2.0sec for the FMM)

Total time for the electro-statics problem $= 0.65$sec.

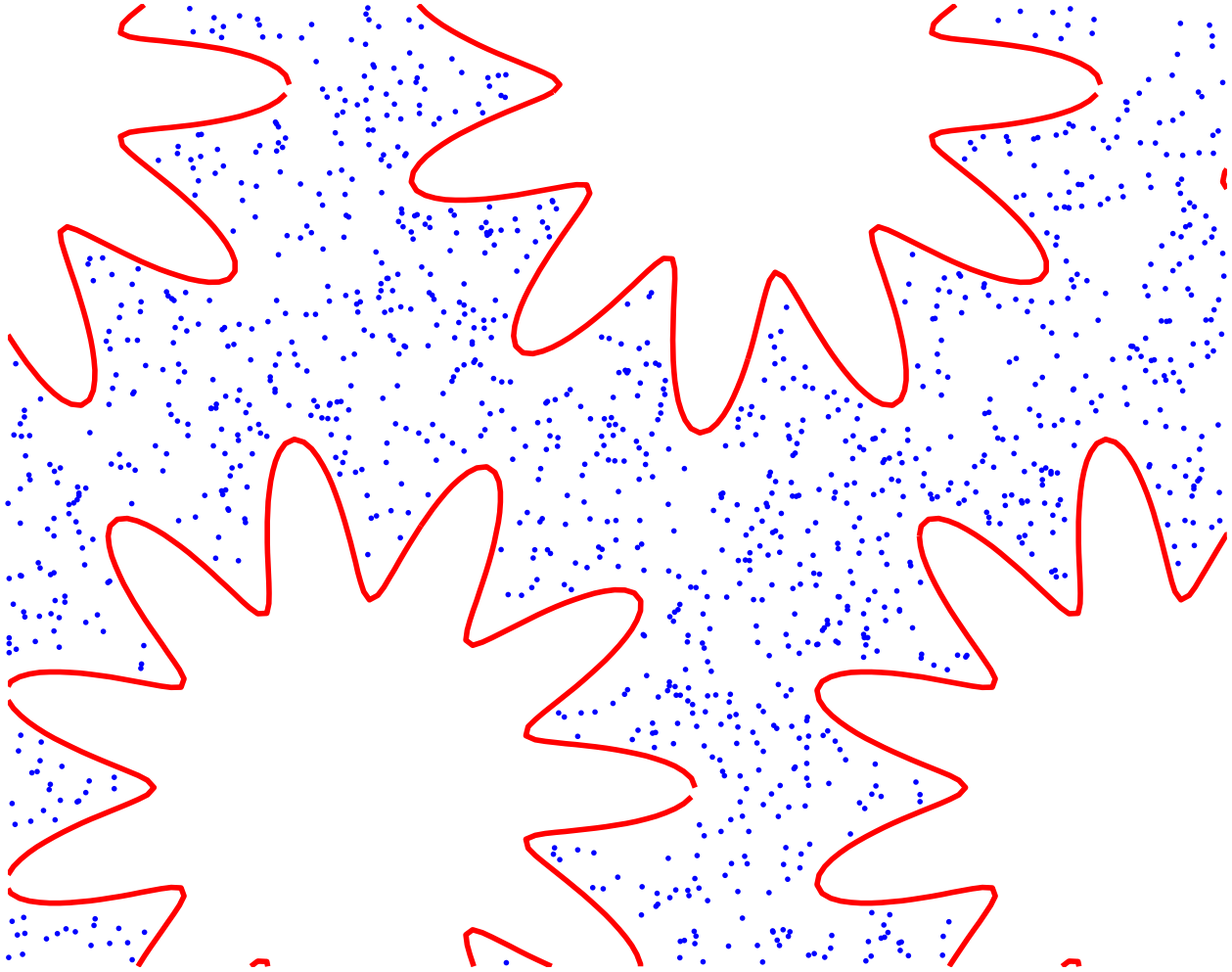


$\varepsilon = 10^{-5}$ $N_{\text{contour}} = 25\,600$ $N_{\text{particles}} = 100\,000$

Time to invert the boundary integral equation = 46sec.

Time to compute the induced charges = 0.42sec.(2.5sec for the FMM)

Total time for the electro-statics problem = 3.8sec.



A close-up of the particle distribution.

The algorithm has been implemented in 3D by Leslie Greengard and Denis Gueyffier at NYU.

Summary

Methods based on the Fast Multipole Method, and fast direct solvers for boundary integral equations are capable of solving electrostatics problems on heterogeneous dielectric domains rapidly *and accurately*.

For problems involving multiple right hand sides, the methods are very fast.

In 3D, the pre-computational cost is still somewhat high.
With luck, this will soon change.