MATH 393C: Fast Methods in Scientific Computing

Lecture on April 11, 2019

P.G. Martinsson

The University of Texas at Austin

In this lecture, we develop a direct solver for an integral equations such as

$$\alpha\, q(\boldsymbol{x}) + \int_\Gamma k(\boldsymbol{x},\boldsymbol{y})\, q(\boldsymbol{y})\, dS(\boldsymbol{y}) = f(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Gamma, \tag{1}$$

where Γ is a contour in $\mathbb{R}^2$ or a surface in $\mathbb{R}^3$. We'll do 2D first, and will then generalize.

Upon Nyström discretization (see Lecture 7), the BIE (1) turns into the linear system

$$\underset{N\times N}{\mathbf{A}} \quad \underset{N\times 1}{\mathbf{q}} \quad = \quad \underset{N\times 1}{\mathbf{f}},$$

where **A** is a dense $N \times N$ matrix.

Standard approach: Use an iterative solver (e.g. GMRES, CG), combined with an $O(N)$ method for evaluating $\mathbf{x} \mapsto \mathbf{A}\mathbf{x}$ such as the Fast Multipole Method (FMM) or panel clustering. When convergence is fast, optimal $O(N)$ complexity results.

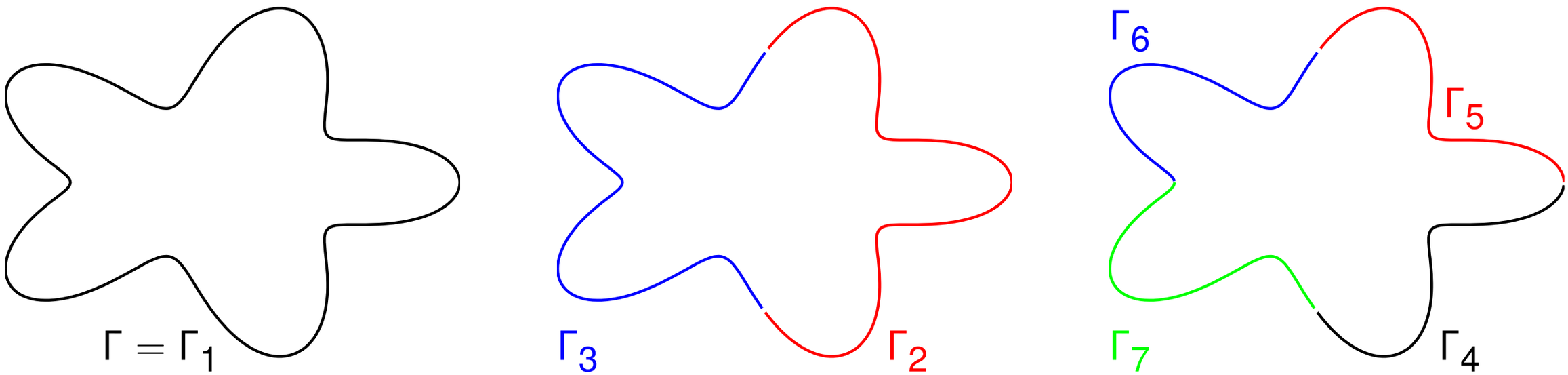
New approach: We seek to construct a direct solver which in a single sweep constructs a data-sparse representation of an operator **B** such that $\mathbf{B} \approx \mathbf{A}^{-1}$. Why?

- Can solve problems for which iterative methods converge slowly or not at all.
- Very fast when solving a sequence of equations with the same operator.
- Good for modern computers (low communication, memory and flops are cheap).

Key observation: The off-diagonal blocks of **A** tend to have low numerical rank.
(Note that for high-frequency problems, other structure in **A** is used.)

The direct solvers are (like the FMM, panel clustering, $\mathcal{H}$-matrices, ...) based on hierarchical partitioning of the physical domain.

Example: Consider a BIE defined on a contour $\Gamma \subset \mathbb{R}^2$.



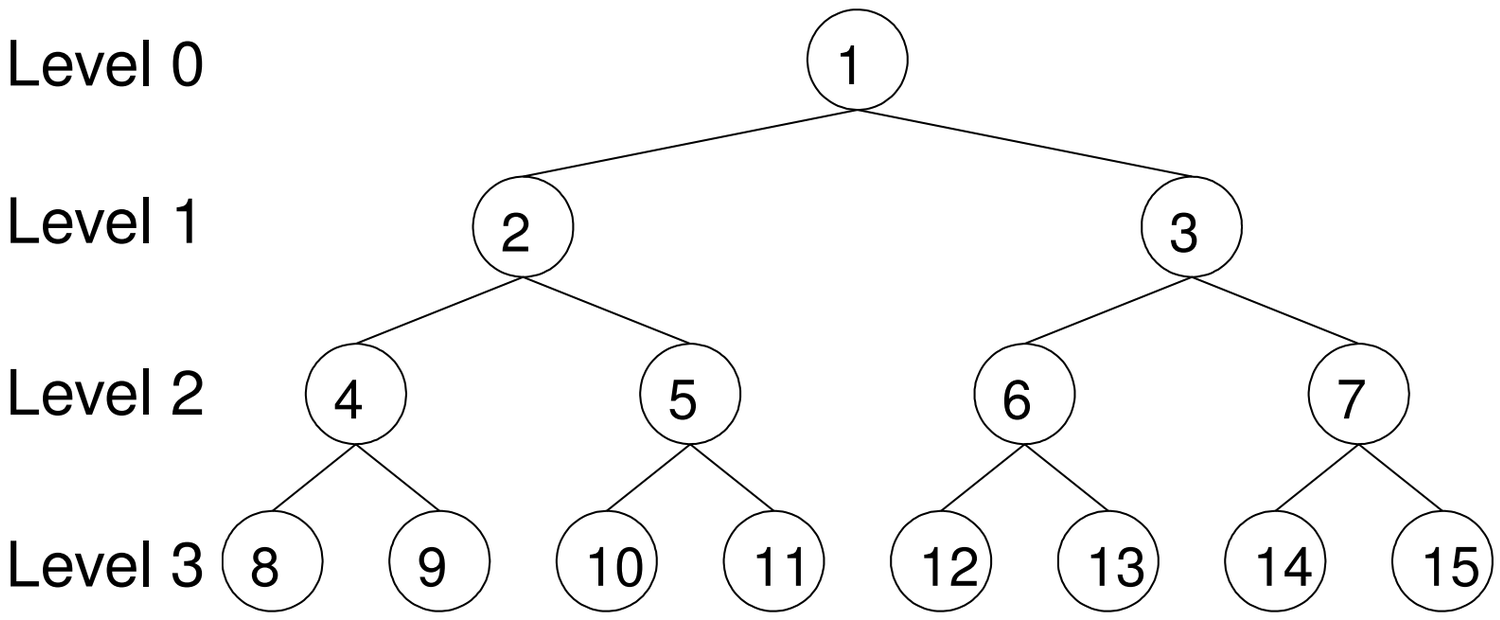
Let $\Gamma = \Gamma_1$ denote the root of a tree.

Partition Γ_1 into two pieces $\Gamma_1 = \Gamma_2 \cup \Gamma_3$.

Further partition $\Gamma_2 = \Gamma_4 \cup \Gamma_5$ and $\Gamma_3 = \Gamma_6 \cup \Gamma_7$.

The tree partitioning corresponds to a partitioning of the index vector $I = [1, 2, 3, \ldots, N]$.

For instance, if $N = 400$, and we use a tree with 4 levels, and split the index vector by halves each time, we get:



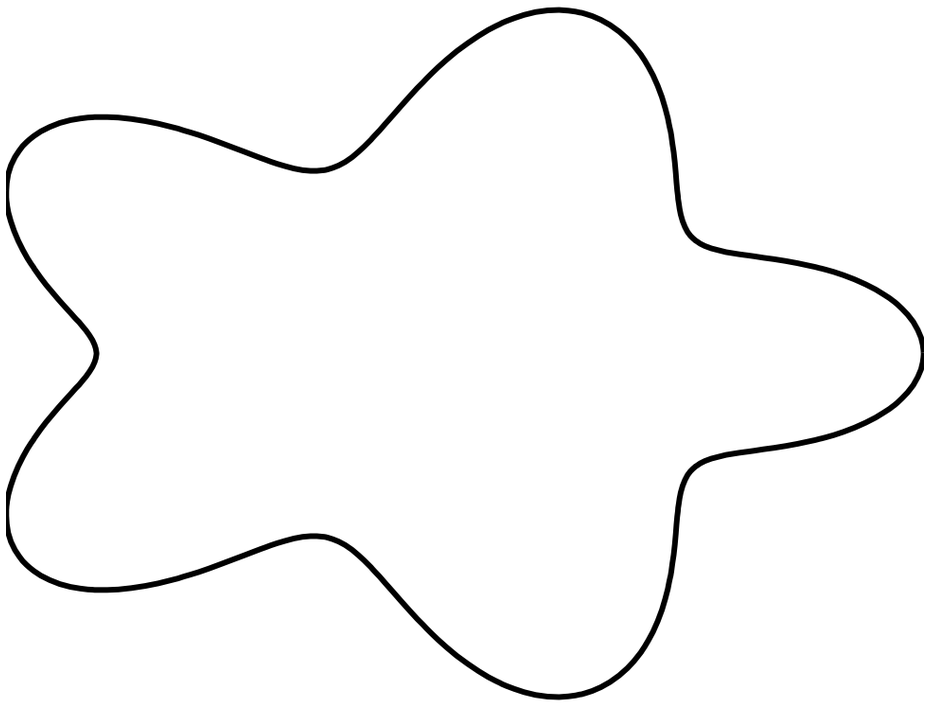
Note: *This simplistic illustration would be accurate for a simple curve.*

For complicated curves, for surfaces/volumes, etc, the index vectors are not contiguous.

The key is to subdivide based on locations $\{\boldsymbol{x}_i\}_{i=1}^{N}$ *in physical space.*

Claim: The matrix **A** resulting upon discretization of a BIE on a curve can often be represented as an "$\mathcal{S}$-matrix" with low or moderate ranks.

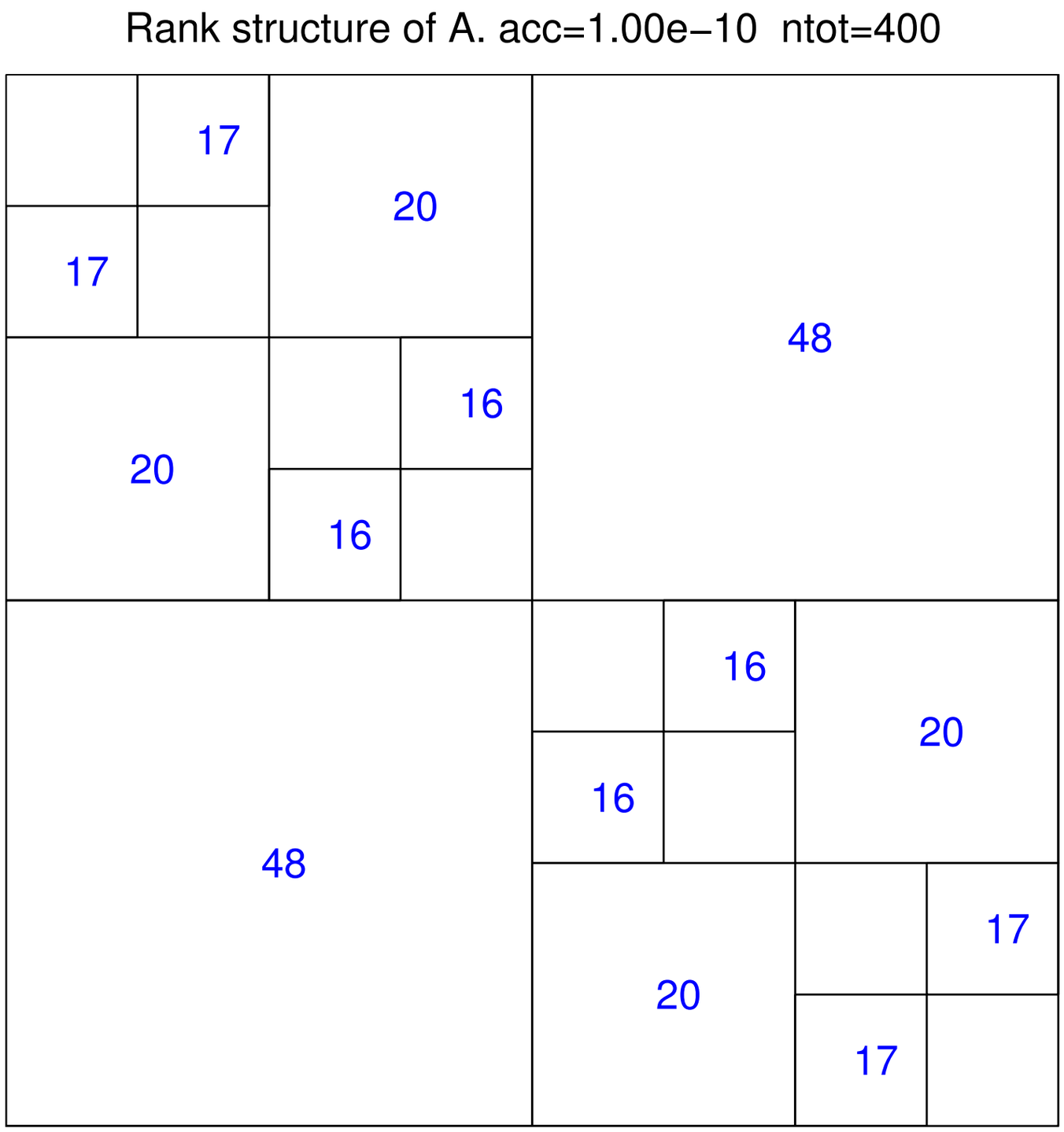
Example 1: Laplace problem discretized with Kolm-Rokhlin quadrature, $n = 400$.



The contour.

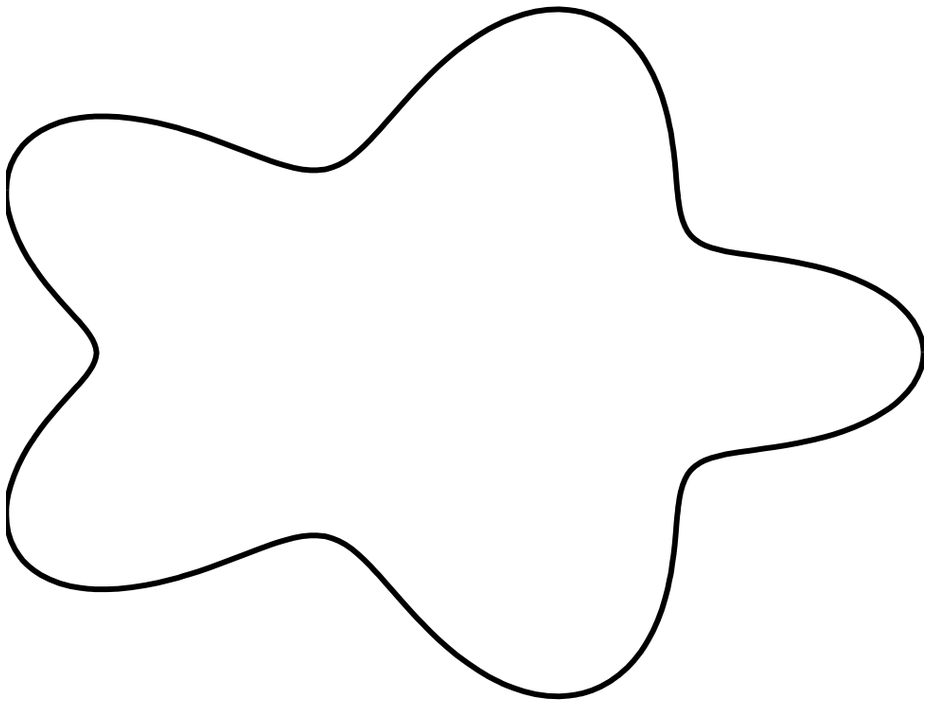
"Combined field" kernel:

$$K(\boldsymbol{x}, \boldsymbol{x}') = \log|\boldsymbol{x}-\boldsymbol{x}'| + \frac{\boldsymbol{n}' \cdot (\boldsymbol{x} - \boldsymbol{x}')}{|\boldsymbol{x} - \boldsymbol{x}'|}.$$



Ranks of off-diagonal blocks.

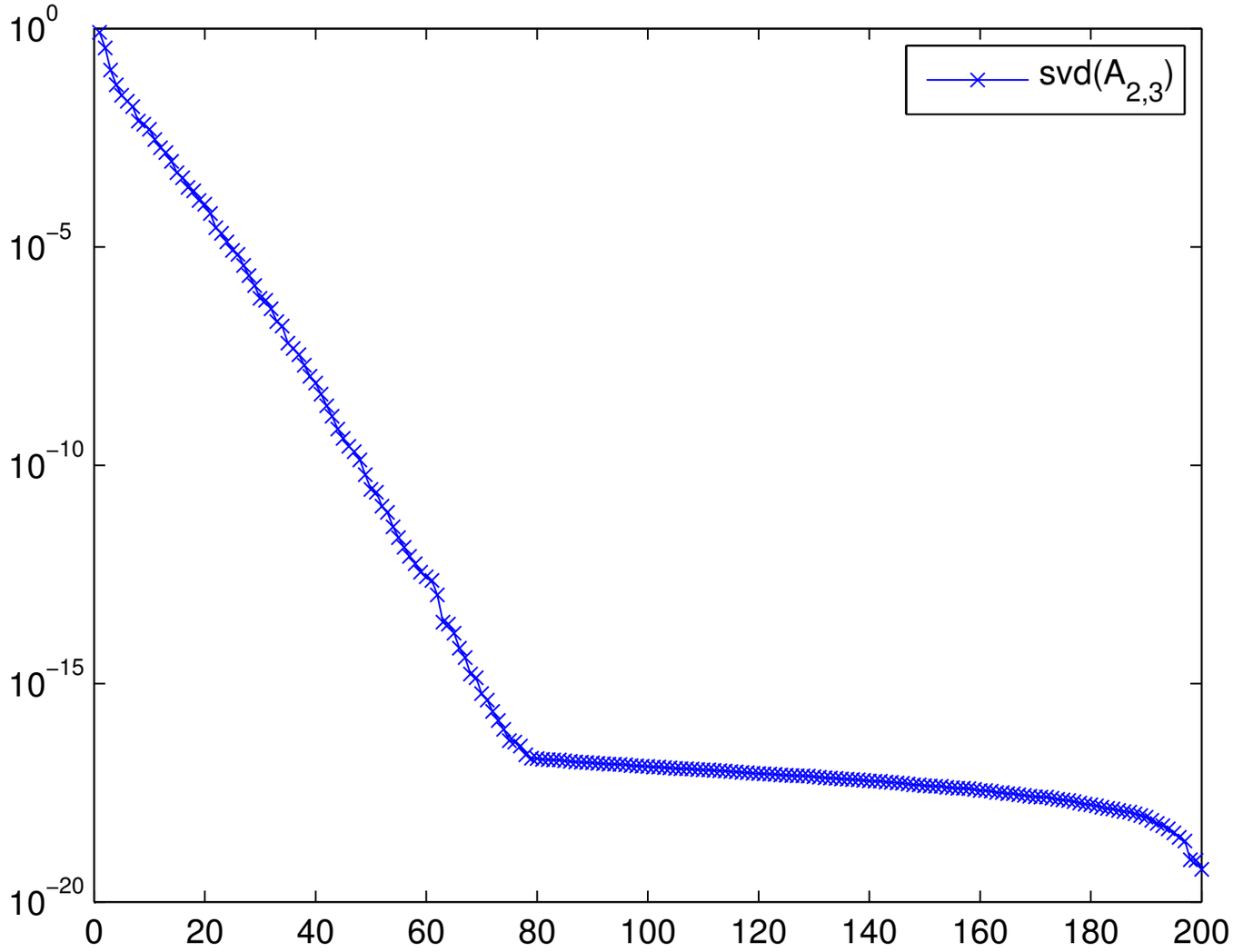
Example 1: Laplace problem discretized with Kolm-Rokhlin quadrature, $n = 400$.



The contour.

"Combined field" kernel:

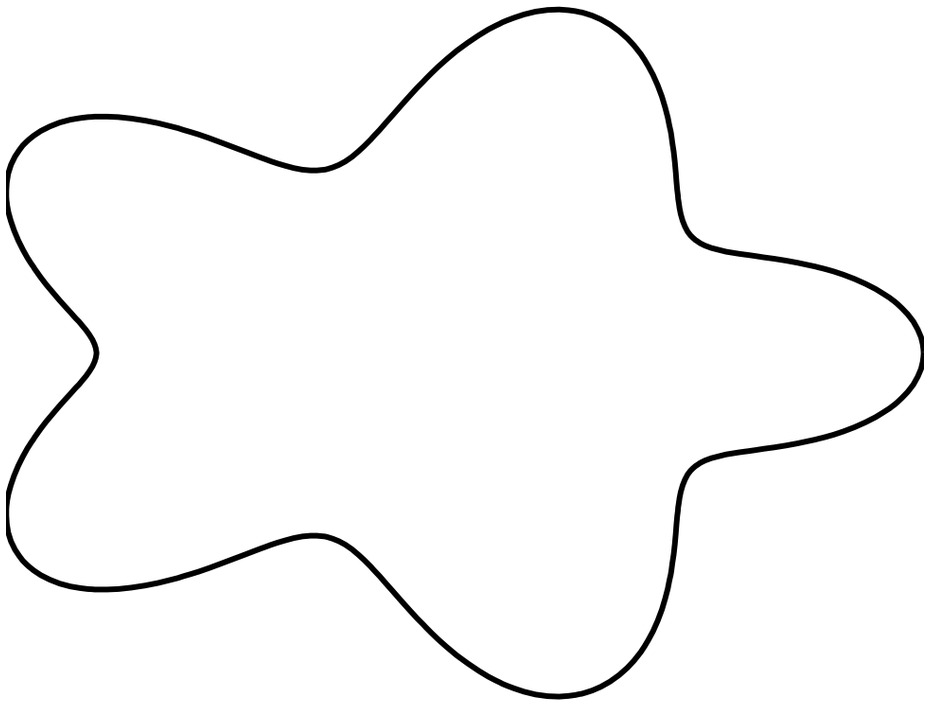
$$K(\boldsymbol{x}, \boldsymbol{x}') = \log|\boldsymbol{x}-\boldsymbol{x}'| + \frac{\boldsymbol{n}' \cdot (\boldsymbol{x} - \boldsymbol{x}')}{|\boldsymbol{x} - \boldsymbol{x}'|}.$$



Singular values of $\mathbf{A}_{2,3}$

(top right quadrant of $\mathbf{A}$*)*

Example 2: Helmholtz problem discretized with Kolm-Rokhlin quadrature, $n = 400$.

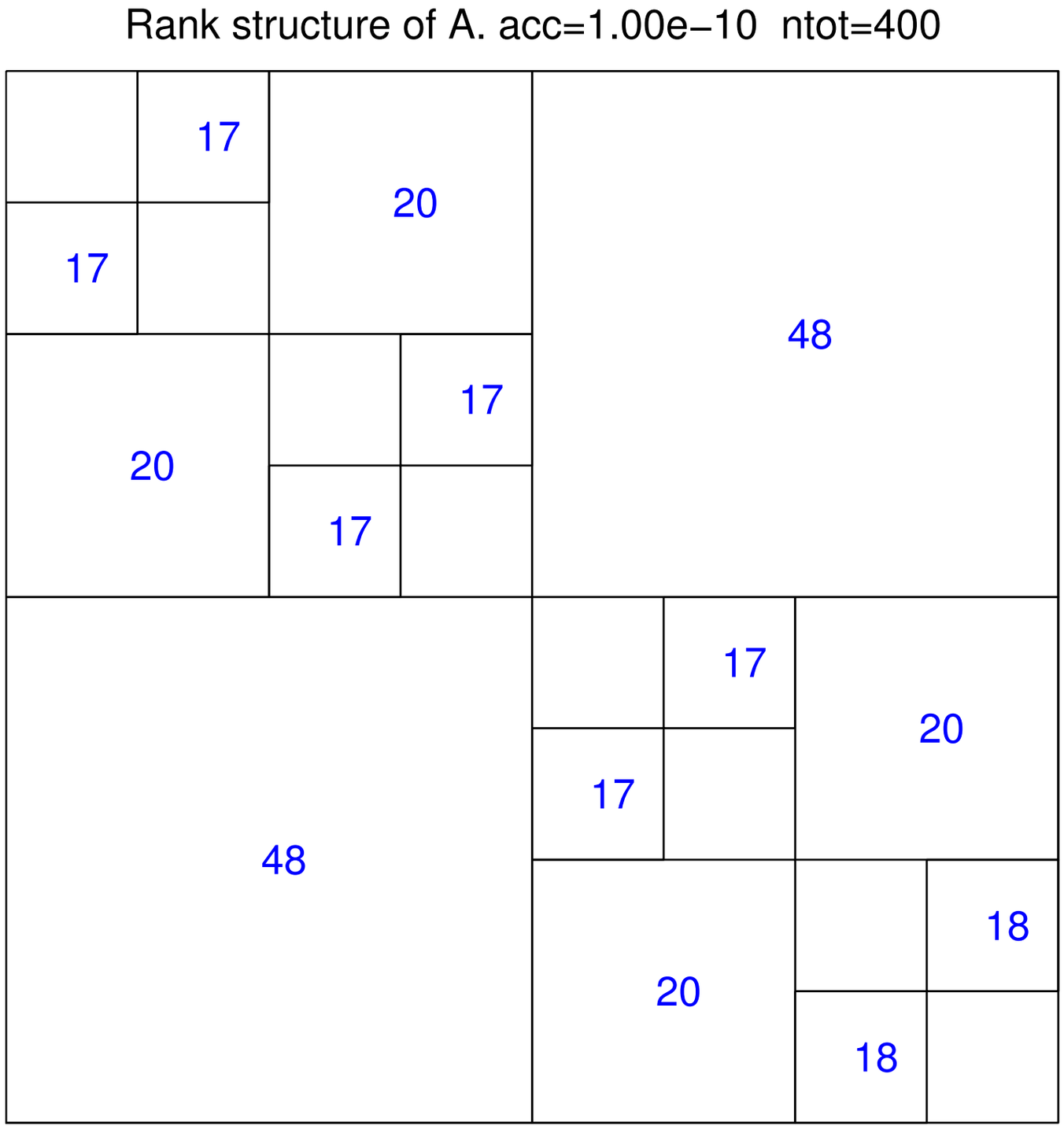


The contour, diameter = 1.2λ.

"Combined field" kernel:

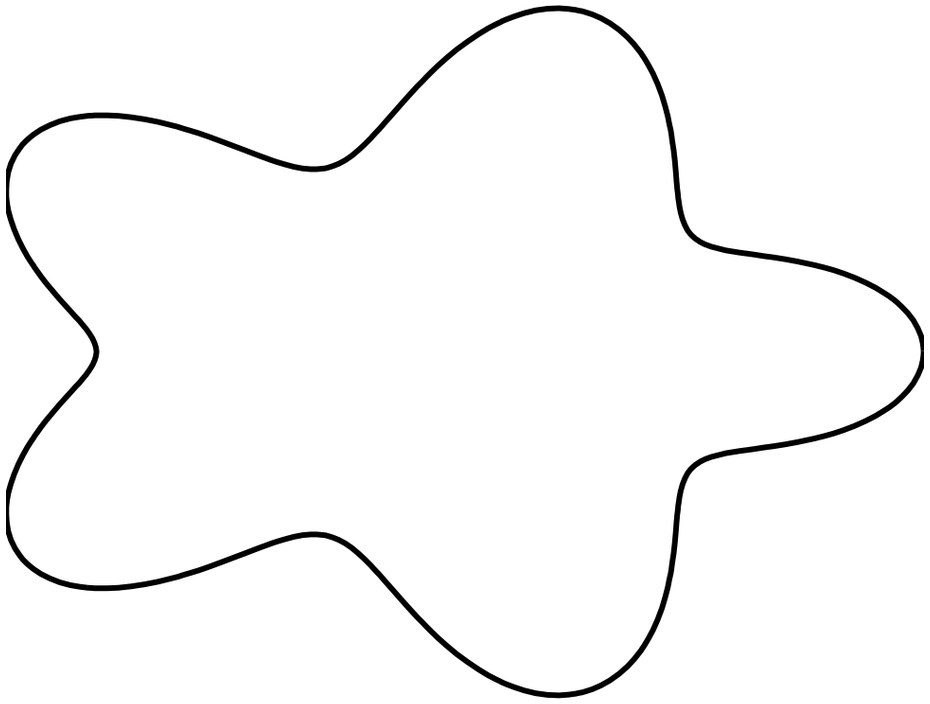
$$K(\boldsymbol{x}, \boldsymbol{x}') = i\,\kappa\, H_0(\kappa|\boldsymbol{x} - \boldsymbol{x}'|) + \partial_{\boldsymbol{n}'} H_0(\kappa|\boldsymbol{x} - \boldsymbol{x}'|).$$

(the weights might be off...)



Ranks of off-diagonal blocks.

Example 2: Helmholtz problem discretized with Kolm-Rokhlin quadrature, $n = 400$.

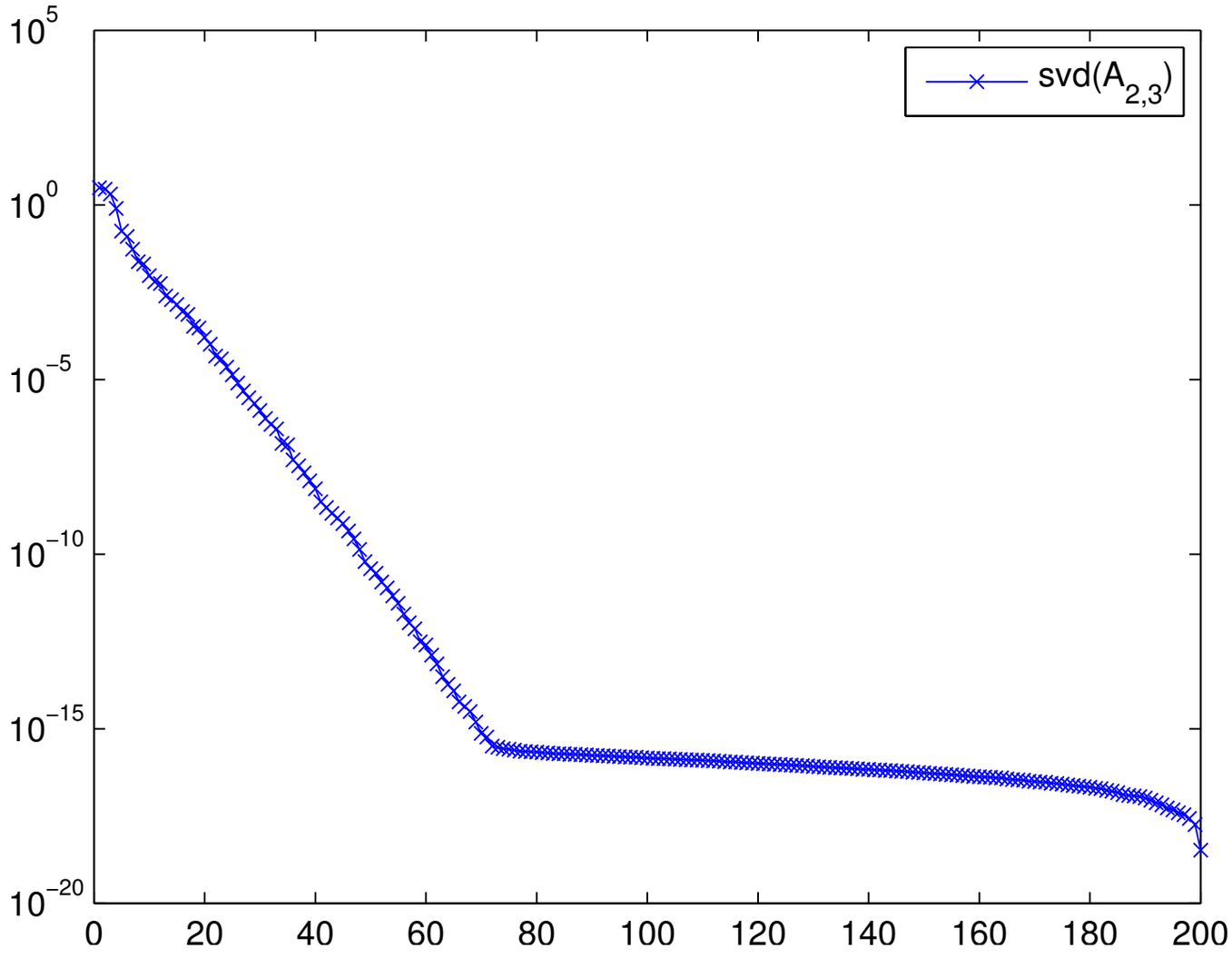


The contour, diameter = 1.2λ.

“Combined field” kernel:

$$K(\boldsymbol{x}, \boldsymbol{x}') = i\,\kappa\, H_0(\kappa|\boldsymbol{x} - \boldsymbol{x}'|) + \partial_{\boldsymbol{n}'} H_0(\kappa|\boldsymbol{x} - \boldsymbol{x}'|).$$

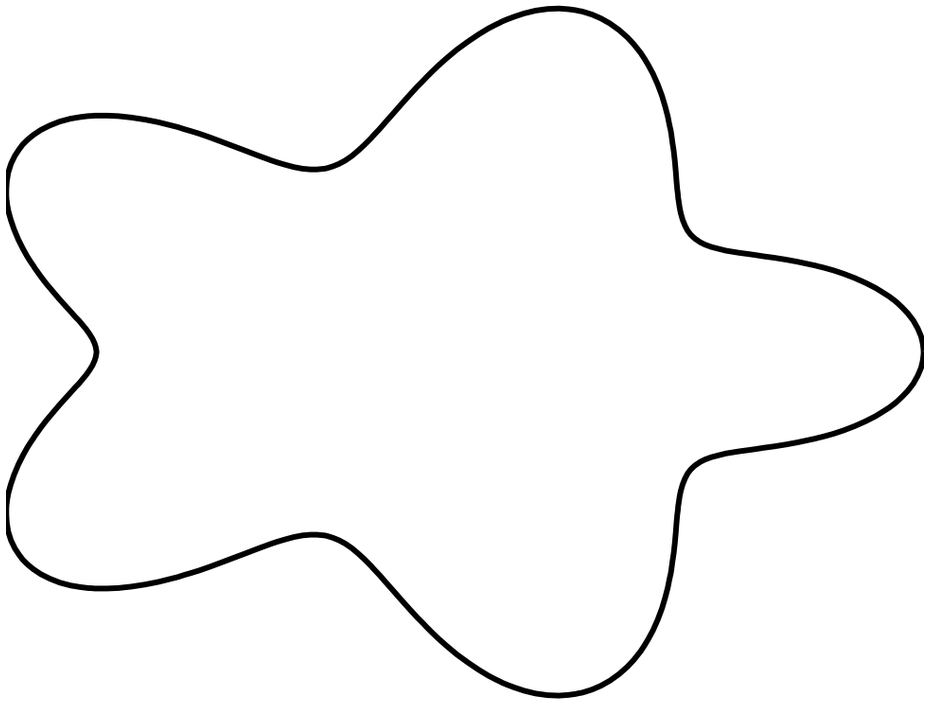
(the weights might be off...)



Singular values of $\mathbf{A}_{2,3}$

(top right quadrant of $\mathbf{A}$*)*

Example 3: *medium-frequency* Helmholtz, Kolm-Rokhlin quadrature, $n = 400$.

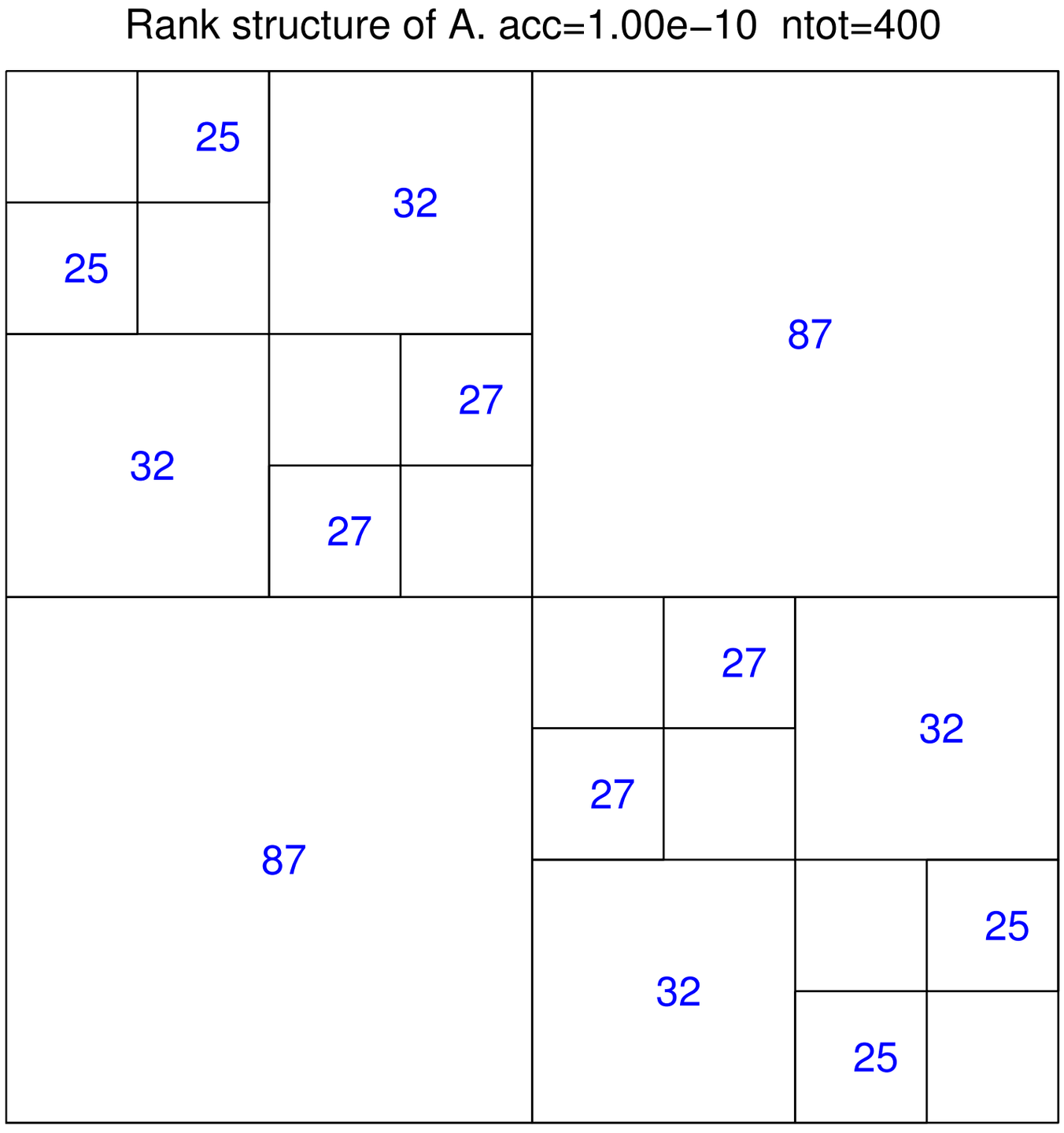


The contour, diameter = 29λ.

"Combined field" kernel:

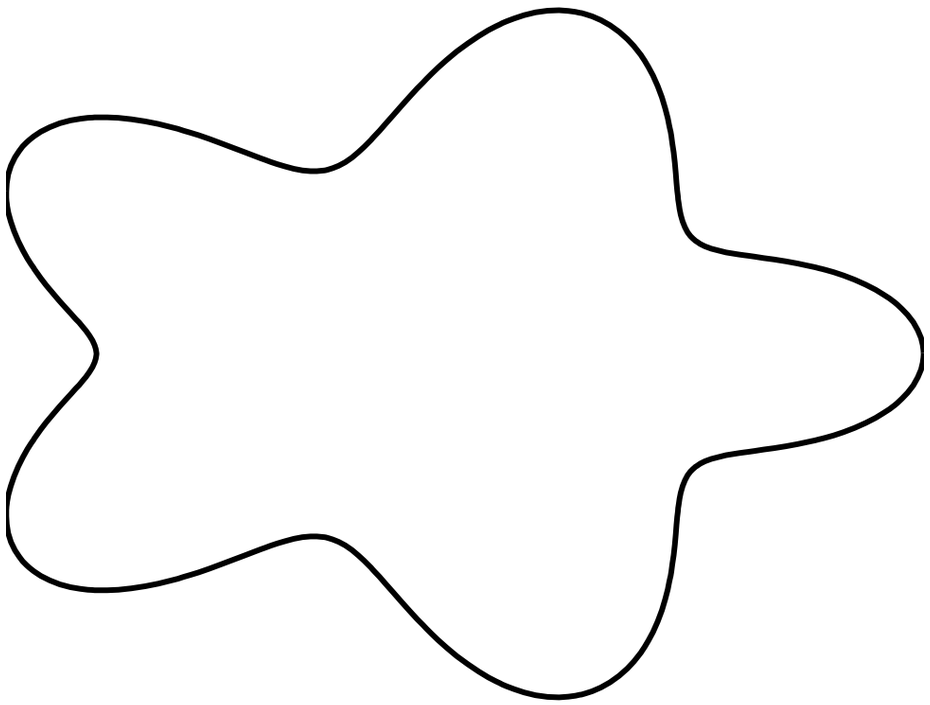
$$K(\boldsymbol{x}, \boldsymbol{x}') = i\,\kappa\, H_0(\kappa|\boldsymbol{x} - \boldsymbol{x}'|) + \partial_{\boldsymbol{n}'} H_0(\kappa|\boldsymbol{x} - \boldsymbol{x}'|).$$

(the weights might be off...)



Ranks of off-diagonal blocks.

Example 3: *medium-frequency* Helmholtz, Kolm-Rokhlin quadrature, $n = 400$.

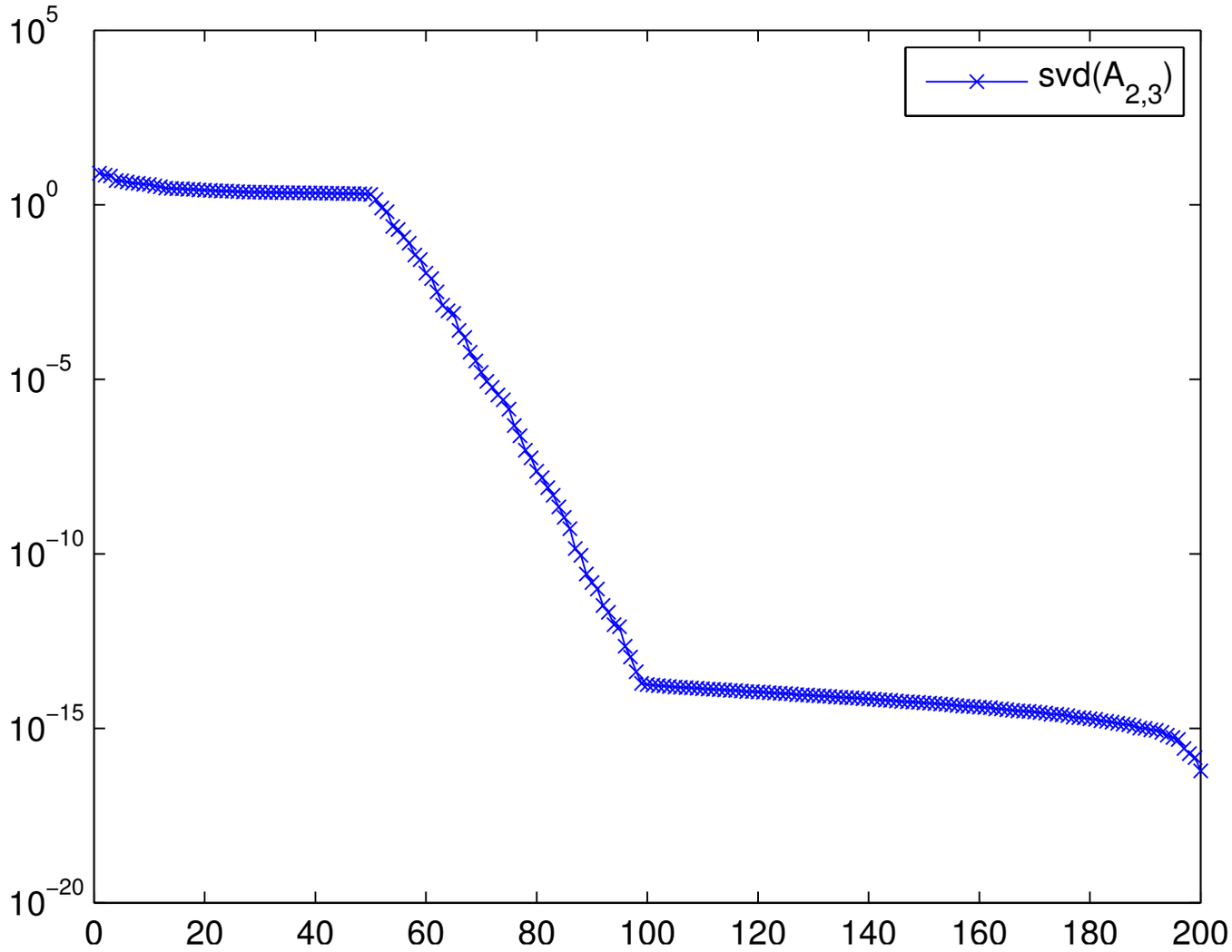


The contour, diameter = 29λ.

"Combined field" kernel:

$$K(\boldsymbol{x}, \boldsymbol{x}') = i\,\kappa\, H_0(\kappa|\boldsymbol{x} - \boldsymbol{x}'|) + \partial_{\boldsymbol{n}'} H_0(\kappa|\boldsymbol{x} - \boldsymbol{x}'|).$$

(the weights might be off...)



Singular values of $\mathbf{A}_{2,3}$

(top right quadrant of $\mathbf{A}$*)*

The "simple" $\mathcal{S}$-matrix format can be used to build direct solvers for BIEs, but we will use a more efficient format called the *Hierarchically Block Separable (HBS)* format (sometimes called "Hierarchically Semi Separable (HSS)" format).

First we introduce *block 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 $n \times n$:

$$\mathbf{A} = \begin{bmatrix} \mathbf{D}_4 & \mathbf{A}_{45} & \mathbf{A}_{46} & \mathbf{A}_{47} \\ \mathbf{A}_{54} & \mathbf{D}_5 & \mathbf{A}_{56} & \mathbf{A}_{57} \\ \mathbf{A}_{64} & \mathbf{A}_{65} & \mathbf{D}_6 & \mathbf{A}_{67} \\ \mathbf{A}_{74} & \mathbf{A}_{75} & \mathbf{A}_{76} & \mathbf{D}_7 \end{bmatrix}. \qquad \text{(Shown for } p = 4.\text{)}$$

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

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

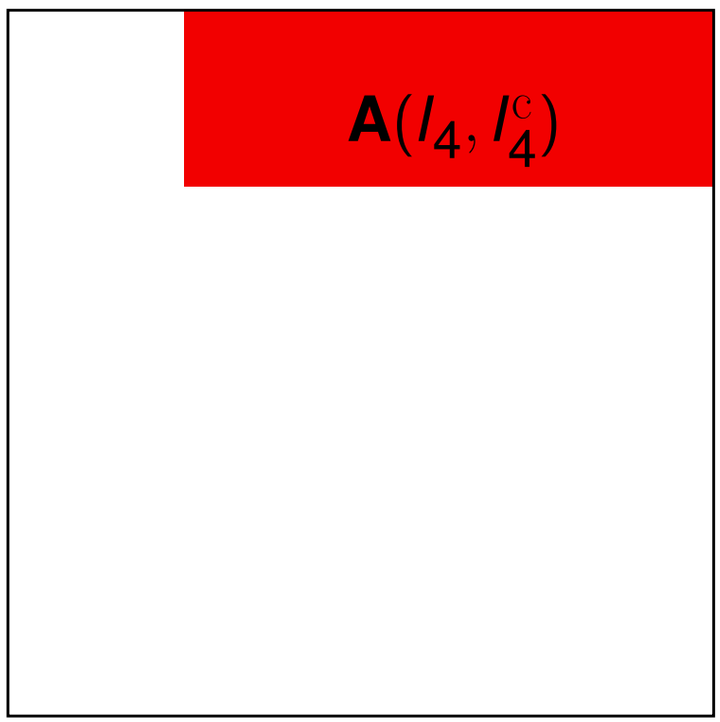
where the rank k is significantly smaller than the block size n.

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).

What is the role of the basis matrices $\mathbf{U}_\tau$ *and* $\mathbf{V}_\tau$*?*

Recall our toy example: $\mathbf{A} = \begin{bmatrix} \mathbf{D}_4 & \mathbf{U}_4 \tilde{\mathbf{A}}_{45} \mathbf{V}_5^* & \mathbf{U}_4 \tilde{\mathbf{A}}_{46} \mathbf{V}_6^* & \mathbf{U}_4 \tilde{\mathbf{A}}_{47} \mathbf{V}_7^* \\ \mathbf{U}_5 \tilde{\mathbf{A}}_{54} \mathbf{V}_4^* & \mathbf{D}_5 & \mathbf{U}_5 \tilde{\mathbf{A}}_{56} \mathbf{V}_6^* & \mathbf{U}_5 \tilde{\mathbf{A}}_{57} \mathbf{V}_7^* \\ \mathbf{U}_6 \tilde{\mathbf{A}}_{64} \mathbf{V}_4^* & \mathbf{U}_6 \tilde{\mathbf{A}}_{65} \mathbf{V}_5^* & \mathbf{D}_6 & \mathbf{U}_6 \tilde{\mathbf{A}}_{67} \mathbf{V}_7^* \\ \mathbf{U}_7 \tilde{\mathbf{A}}_{74} \mathbf{V}_4^* & \mathbf{U}_7 \tilde{\mathbf{A}}_{75} \mathbf{V}_5^* & \mathbf{U}_7 \tilde{\mathbf{A}}_{76} \mathbf{V}_6^* & \mathbf{D}_7 \end{bmatrix}$.

We see that the columns of $\mathbf{U}_4$ must span the column space of the matrix $\mathbf{A}(I_4, I_4^c)$ where I_4 is the index vector for the first block and $I_4^c = I \backslash I_4$.

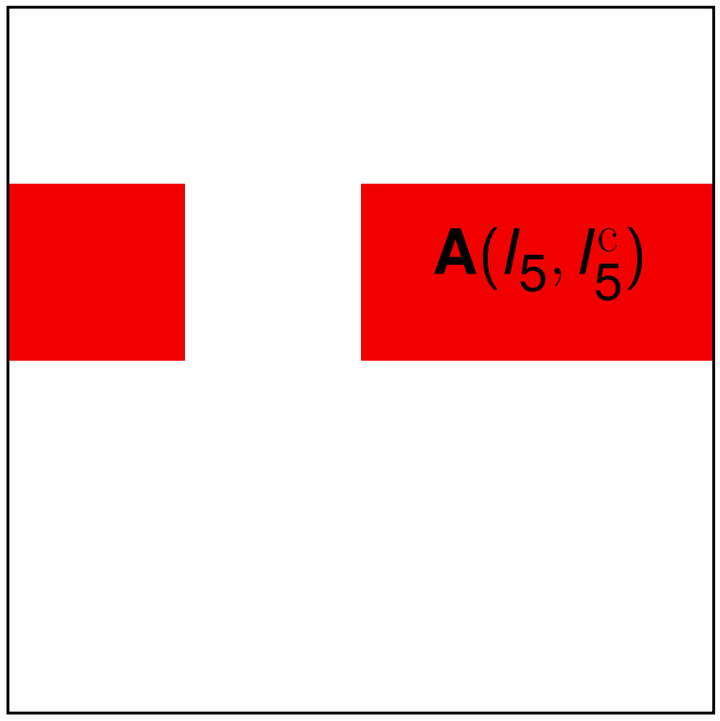


The matrix $\mathbf{A}$

What is the role of the basis matrices $\mathbf{U}_\tau$ and $\mathbf{V}_\tau$?

Recall our toy example: $\mathbf{A} = \begin{bmatrix} \mathbf{D}_4 & \mathbf{U}_4\tilde{\mathbf{A}}_{45}\mathbf{V}_5^* & \mathbf{U}_4\tilde{\mathbf{A}}_{46}\mathbf{V}_6^* & \mathbf{U}_4\tilde{\mathbf{A}}_{47}\mathbf{V}_7^* \\ \mathbf{U}_5\tilde{\mathbf{A}}_{54}\mathbf{V}_4^* & \mathbf{D}_5 & \mathbf{U}_5\tilde{\mathbf{A}}_{56}\mathbf{V}_6^* & \mathbf{U}_5\tilde{\mathbf{A}}_{57}\mathbf{V}_7^* \\ \mathbf{U}_6\tilde{\mathbf{A}}_{64}\mathbf{V}_4^* & \mathbf{U}_6\tilde{\mathbf{A}}_{65}\mathbf{V}_5^* & \mathbf{D}_6 & \mathbf{U}_6\tilde{\mathbf{A}}_{67}\mathbf{V}_7^* \\ \mathbf{U}_7\tilde{\mathbf{A}}_{74}\mathbf{V}_4^* & \mathbf{U}_7\tilde{\mathbf{A}}_{75}\mathbf{V}_5^* & \mathbf{U}_7\tilde{\mathbf{A}}_{76}\mathbf{V}_6^* & \mathbf{D}_7 \end{bmatrix}$.

We see that the columns of $\mathbf{U}_5$ must span the column space of the matrix $\mathbf{A}(I_5, I_5^{\mathrm{c}})$ where I_5 is the index vector for the first block and $I_5^{\mathrm{c}} = I \backslash I_5$.



The matrix $\mathbf{A}$

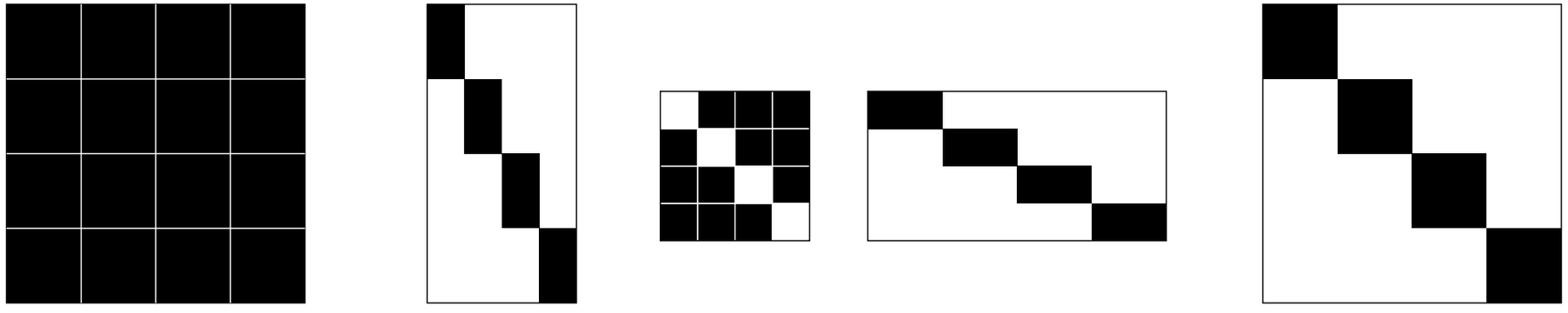
Recall $\mathbf{A} = \begin{bmatrix} \mathbf{D}_4 & \mathbf{U}_4\tilde{\mathbf{A}}_{45}\mathbf{V}_5^* & \mathbf{U}_4\tilde{\mathbf{A}}_{46}\mathbf{V}_6^* & \mathbf{U}_4\tilde{\mathbf{A}}_{47}\mathbf{V}_7^* \\ \mathbf{U}_5\tilde{\mathbf{A}}_{54}\mathbf{V}_4^* & \mathbf{D}_5 & \mathbf{U}_5\tilde{\mathbf{A}}_{56}\mathbf{V}_6^* & \mathbf{U}_5\tilde{\mathbf{A}}_{57}\mathbf{V}_7^* \\ \mathbf{U}_6\tilde{\mathbf{A}}_{64}\mathbf{V}_4^* & \mathbf{U}_6\tilde{\mathbf{A}}_{65}\mathbf{V}_5^* & \mathbf{D}_6 & \mathbf{U}_6\tilde{\mathbf{A}}_{67}\mathbf{V}_7^* \\ \mathbf{U}_7\tilde{\mathbf{A}}_{74}\mathbf{V}_4^* & \mathbf{U}_7\tilde{\mathbf{A}}_{75}\mathbf{V}_5^* & \mathbf{U}_7\tilde{\mathbf{A}}_{76}\mathbf{V}_6^* & \mathbf{D}_7 \end{bmatrix}.$

Then $\mathbf{A}$ admits the factorization:

$$\mathbf{A} = \underbrace{\begin{bmatrix} \mathbf{U}_4 & & & \\ & \mathbf{U}_5 & & \\ & & \mathbf{U}_6 & \\ & & & \mathbf{U}_7 \end{bmatrix}}_{=\mathbf{U}} \underbrace{\begin{bmatrix} \mathbf{0} & \tilde{\mathbf{A}}_{45} & \tilde{\mathbf{A}}_{46} & \tilde{\mathbf{A}}_{47} \\ \tilde{\mathbf{A}}_{54} & \mathbf{0} & \tilde{\mathbf{A}}_{56} & \tilde{\mathbf{A}}_{57} \\ \tilde{\mathbf{A}}_{64} & \tilde{\mathbf{A}}_{65} & \mathbf{0} & \tilde{\mathbf{A}}_{67} \\ \tilde{\mathbf{A}}_{74} & \tilde{\mathbf{A}}_{75} & \tilde{\mathbf{A}}_{76} & \mathbf{0} \end{bmatrix}}_{=\tilde{\mathbf{A}}} \underbrace{\begin{bmatrix} \mathbf{V}_4^* & & & \\ & \mathbf{V}_5^* & & \\ & & \mathbf{V}_6^* & \\ & & & \mathbf{V}_7^* \end{bmatrix}}_{=\mathbf{V}^*} + \underbrace{\begin{bmatrix} \mathbf{D}_4 & & & \\ & \mathbf{D}_5 & & \\ & & \mathbf{D}_6 & \\ & & & \mathbf{D}_7 \end{bmatrix}}_{=\mathbf{D}}$$

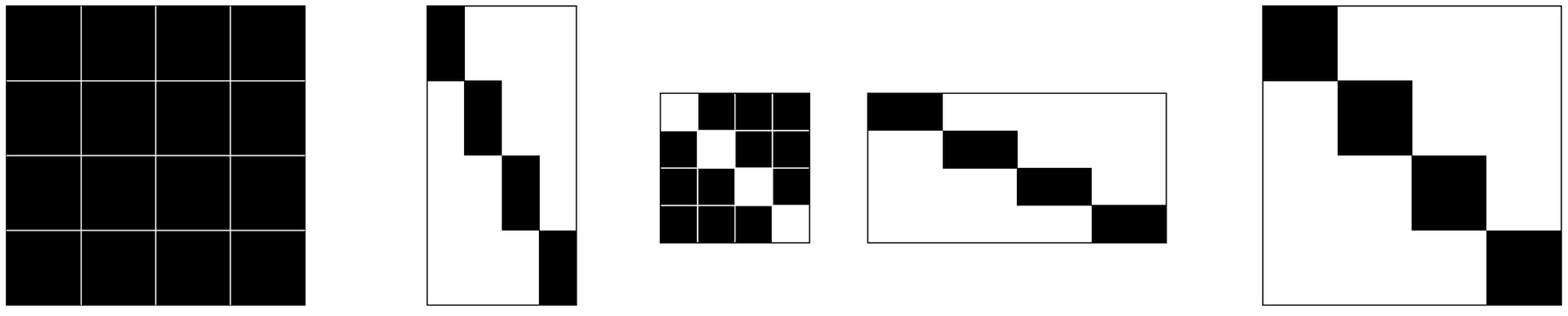
or

$$\underset{pn \times pn}{\mathbf{A}} = \underset{pn \times pk}{\mathbf{U}} \; \underset{pk \times pk}{\tilde{\mathbf{A}}} \; \underset{pk \times pn}{\mathbf{V}^*} + \underset{pn \times pn}{\mathbf{D}},$$



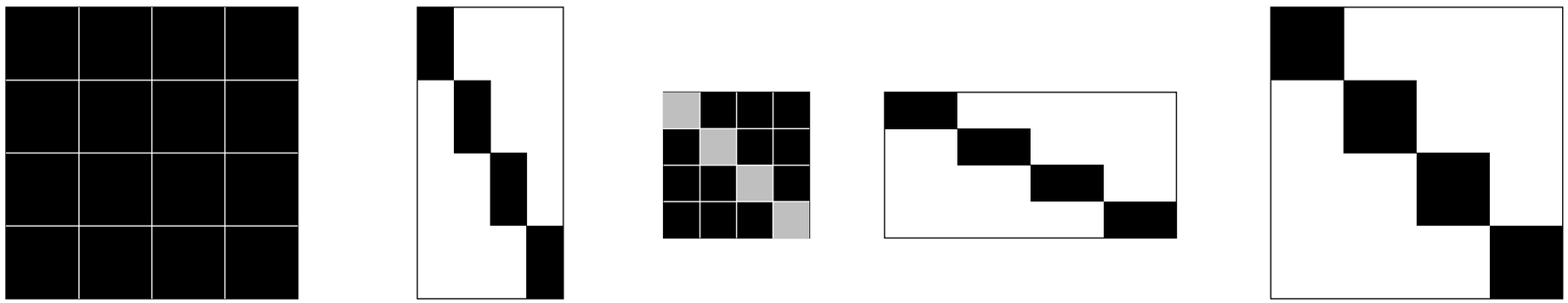
Lemma: [Variation of Woodbury] If an $N \times N$ matrix $\mathbf{A}$ admits the factorization

$$\underset{pn\times pn}{\mathbf{A}} = \underset{pn\times pk}{\mathbf{U}}\ \underset{pk\times pk}{\tilde{\mathbf{A}}}\ \underset{pk\times pn}{\mathbf{V}^*} + \underset{pn\times pn}{\mathbf{D}},$$



then

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



where (provided all intermediate matrices are invertible)

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

Note: All matrices set in blue are block diagonal.

Classical Woodbury: $\left(\mathbf{D}+\mathbf{U}\tilde{\mathbf{A}}\mathbf{V}^*\right)^{-1} = \mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{U}\left(\tilde{\mathbf{A}}+\mathbf{V}^*\mathbf{D}^{-1}\mathbf{U}\right)^{-1}\mathbf{V}^*\mathbf{D}^{-1}.$

Derivation of "our" Woodbury: We consider the linear system

$$\begin{bmatrix} \mathbf{D}_4 & \mathbf{U}_4\tilde{\mathbf{A}}_{45}\mathbf{V}_5^* & \mathbf{U}_4\tilde{\mathbf{A}}_{46}\mathbf{V}_6^* & \mathbf{U}_4\tilde{\mathbf{A}}_{47}\mathbf{V}_7^* \\ \mathbf{U}_5\tilde{\mathbf{A}}_{54}\mathbf{V}_4^* & \mathbf{D}_5 & \mathbf{U}_5\tilde{\mathbf{A}}_{56}\mathbf{V}_6^* & \mathbf{U}_5\tilde{\mathbf{A}}_{57}\mathbf{V}_7^* \\ \mathbf{U}_6\tilde{\mathbf{A}}_{64}\mathbf{V}_4^* & \mathbf{U}_6\tilde{\mathbf{A}}_{65}\mathbf{V}_5^* & \mathbf{D}_6 & \mathbf{U}_6\tilde{\mathbf{A}}_{67}\mathbf{V}_7^* \\ \mathbf{U}_7\tilde{\mathbf{A}}_{74}\mathbf{V}_4^* & \mathbf{U}_7\tilde{\mathbf{A}}_{75}\mathbf{V}_5^* & \mathbf{U}_7\tilde{\mathbf{A}}_{76}\mathbf{V}_6^* & \mathbf{D}_7 \end{bmatrix} \begin{bmatrix} \mathbf{q}_4 \\ \mathbf{q}_5 \\ \mathbf{q}_6 \\ \mathbf{q}_7 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_4 \\ \mathbf{f}_5 \\ \mathbf{f}_6 \\ \mathbf{f}_7 \end{bmatrix}.$$

Introduce *reduced variables* $\tilde{\mathbf{q}}_i = \mathbf{V}_i^* \mathbf{q}_i$.

The system $\sum_j \mathbf{A}_{ij}\mathbf{q}_j = \mathbf{f}_i$ then takes the form

$$\left[\begin{array}{cccc|cccc} \mathbf{D}_4 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{U}_4\tilde{\mathbf{A}}_{45} & \mathbf{U}_4\tilde{\mathbf{A}}_{46} & \mathbf{U}_4\tilde{\mathbf{A}}_{47} \\ \mathbf{0} & \mathbf{D}_5 & \mathbf{0} & \mathbf{0} & \mathbf{U}_5\tilde{\mathbf{A}}_{54} & \mathbf{0} & \mathbf{U}_5\tilde{\mathbf{A}}_{56} & \mathbf{U}_5\tilde{\mathbf{A}}_{57} \\ \mathbf{0} & \mathbf{0} & \mathbf{D}_6 & \mathbf{0} & \mathbf{U}_6\tilde{\mathbf{A}}_{64} & \mathbf{U}_6\tilde{\mathbf{A}}_{65} & \mathbf{0} & \mathbf{U}_6\tilde{\mathbf{A}}_{67} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{D}_7 & \mathbf{U}_7\tilde{\mathbf{A}}_{74} & \mathbf{U}_7\tilde{\mathbf{A}}_{75} & \mathbf{U}_7\tilde{\mathbf{A}}_{76} & \mathbf{0} \\ \hline -\mathbf{V}_4^* & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{V}_5^* & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{V}_6^* & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{V}_7^* & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{array}\right] \begin{bmatrix} \mathbf{q}_4 \\ \mathbf{q}_5 \\ \mathbf{q}_6 \\ \mathbf{q}_7 \\ \tilde{\mathbf{q}}_4 \\ \tilde{\mathbf{q}}_5 \\ \tilde{\mathbf{q}}_6 \\ \tilde{\mathbf{q}}_7 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_4 \\ \mathbf{f}_5 \\ \mathbf{f}_6 \\ \mathbf{f}_7 \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}.$$

Now form the Schur complement to eliminate the $\mathbf{q}_j$'s.

After eliminating the “fine-scale” variables $\mathbf{q}_i$, we obtain

$$\begin{bmatrix} \mathbf{I} & \mathbf{V}_4^* \tilde{\mathbf{A}}_{44}^{-1} \mathbf{U}_4 \tilde{\mathbf{A}}_{45} & \mathbf{V}_4^* \tilde{\mathbf{A}}_{44}^{-1} \mathbf{U}_4 \tilde{\mathbf{A}}_{46} & \mathbf{V}_4^* \tilde{\mathbf{A}}_{44}^{-1} \mathbf{U}_4 \tilde{\mathbf{A}}_{47} \\ \mathbf{V}_5^* \tilde{\mathbf{A}}_{55}^{-1} \mathbf{U}_5 \tilde{\mathbf{A}}_{54} & \mathbf{I} & \mathbf{V}_5^* \tilde{\mathbf{A}}_{55}^{-1} \mathbf{U}_5 \tilde{\mathbf{A}}_{56} & \mathbf{V}_5^* \tilde{\mathbf{A}}_{55}^{-1} \mathbf{U}_5 \tilde{\mathbf{A}}_{57} \\ \mathbf{V}_6^* \tilde{\mathbf{A}}_{66}^{-1} \mathbf{U}_6 \tilde{\mathbf{A}}_{61} & \mathbf{V}_6^* \tilde{\mathbf{A}}_{66}^{-1} \mathbf{U}_6 \tilde{\mathbf{A}}_{65} & \mathbf{I} & \mathbf{V}_6^* \tilde{\mathbf{A}}_{66}^{-1} \mathbf{U}_6 \tilde{\mathbf{A}}_{67} \\ \mathbf{V}_7^* \tilde{\mathbf{A}}_{77}^{-1} \mathbf{U}_7 \tilde{\mathbf{A}}_{74} & \mathbf{V}_7^* \tilde{\mathbf{A}}_{77}^{-1} \mathbf{U}_7 \tilde{\mathbf{A}}_{75} & \mathbf{V}_7^* \tilde{\mathbf{A}}_{77}^{-1} \mathbf{U}_7 \tilde{\mathbf{A}}_{76} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{q}}_4 \\ \tilde{\mathbf{q}}_5 \\ \tilde{\mathbf{q}}_6 \\ \tilde{\mathbf{q}}_7 \end{bmatrix} = \begin{bmatrix} \mathbf{V}_4^* \mathbf{D}_4^{-1} \mathbf{f}_4 \\ \mathbf{V}_5^* \mathbf{D}_5^{-1} \mathbf{f}_5 \\ \mathbf{V}_6^* \mathbf{D}_6^{-1} \mathbf{f}_6 \\ \mathbf{V}_7^* \mathbf{D}_7^{-1} \mathbf{f}_7. \end{bmatrix}.$$

After eliminating the "fine-scale" variables $\mathbf{q}_i$, we obtain

$$\begin{bmatrix} \mathbf{I} & \mathbf{V}_4^* \tilde{\mathbf{A}}_{44}^{-1} \mathbf{U}_4 \tilde{\mathbf{A}}_{45} & \mathbf{V}_4^* \tilde{\mathbf{A}}_{44}^{-1} \mathbf{U}_4 \tilde{\mathbf{A}}_{46} & \mathbf{V}_4^* \tilde{\mathbf{A}}_{44}^{-1} \mathbf{U}_4 \tilde{\mathbf{A}}_{47} \\ \mathbf{V}_5^* \tilde{\mathbf{A}}_{55}^{-1} \mathbf{U}_5 \tilde{\mathbf{A}}_{54} & \mathbf{I} & \mathbf{V}_5^* \tilde{\mathbf{A}}_{55}^{-1} \mathbf{U}_5 \tilde{\mathbf{A}}_{56} & \mathbf{V}_5^* \tilde{\mathbf{A}}_{55}^{-1} \mathbf{U}_5 \tilde{\mathbf{A}}_{57} \\ \mathbf{V}_6^* \tilde{\mathbf{A}}_{66}^{-1} \mathbf{U}_6 \tilde{\mathbf{A}}_{61} & \mathbf{V}_6^* \tilde{\mathbf{A}}_{66}^{-1} \mathbf{U}_6 \tilde{\mathbf{A}}_{65} & \mathbf{I} & \mathbf{V}_6^* \tilde{\mathbf{A}}_{66}^{-1} \mathbf{U}_6 \tilde{\mathbf{A}}_{67} \\ \mathbf{V}_7^* \tilde{\mathbf{A}}_{77}^{-1} \mathbf{U}_7 \tilde{\mathbf{A}}_{74} & \mathbf{V}_7^* \tilde{\mathbf{A}}_{77}^{-1} \mathbf{U}_7 \tilde{\mathbf{A}}_{75} & \mathbf{V}_7^* \tilde{\mathbf{A}}_{77}^{-1} \mathbf{U}_7 \tilde{\mathbf{A}}_{76} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{q}}_4 \\ \tilde{\mathbf{q}}_5 \\ \tilde{\mathbf{q}}_6 \\ \tilde{\mathbf{q}}_7 \end{bmatrix} = \begin{bmatrix} \mathbf{V}_4^* \mathbf{D}_4^{-1} \mathbf{f}_4 \\ \mathbf{V}_5^* \mathbf{D}_5^{-1} \mathbf{f}_5 \\ \mathbf{V}_6^* \mathbf{D}_6^{-1} \mathbf{f}_6 \\ \mathbf{V}_7^* \mathbf{D}_7^{-1} \mathbf{f}_7. \end{bmatrix}.$$

We set

$$\tilde{\mathbf{A}}_{ii} = \left(\mathbf{V}_i^* \, \mathbf{D}_{ii}^{-1} \, \mathbf{U}_i \right)^{-1},$$

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

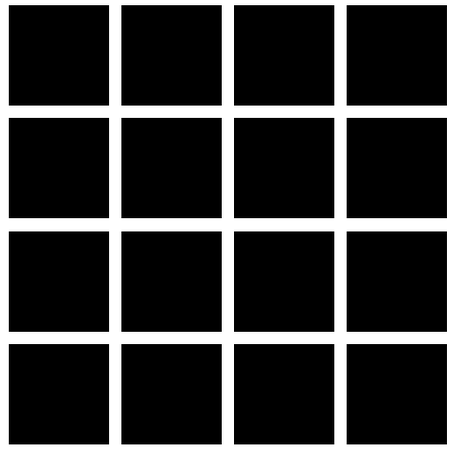
$$\begin{bmatrix} \tilde{\mathbf{A}}_{44} & \tilde{\mathbf{A}}_{45} & \tilde{\mathbf{A}}_{46} & \tilde{\mathbf{A}}_{47} \\ \tilde{\mathbf{A}}_{54} & \tilde{\mathbf{A}}_{55} & \tilde{\mathbf{A}}_{56} & \tilde{\mathbf{A}}_{57} \\ \tilde{\mathbf{A}}_{64} & \tilde{\mathbf{A}}_{65} & \tilde{\mathbf{A}}_{66} & \tilde{\mathbf{A}}_{67} \\ \tilde{\mathbf{A}}_{74} & \tilde{\mathbf{A}}_{75} & \tilde{\mathbf{A}}_{76} & \tilde{\mathbf{A}}_{77} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{q}}_4 \\ \tilde{\mathbf{q}}_5 \\ \tilde{\mathbf{q}}_6 \\ \tilde{\mathbf{q}}_7 \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{f}}_4 \\ \tilde{\mathbf{f}}_5 \\ \tilde{\mathbf{f}}_6 \\ \tilde{\mathbf{f}}_7 \end{bmatrix}.$$

where

$$\tilde{\mathbf{f}}_i = \tilde{\mathbf{A}}_{ii} \, \mathbf{V}_i^* \, \mathbf{D}_{ii}^{-1} \, \mathbf{f}_i.$$

Before compression, we have a $pn \times pn$ linear system

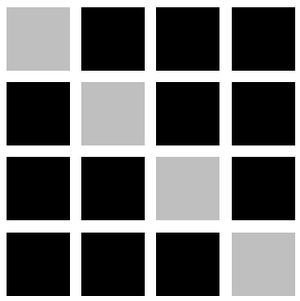
$$\sum_{j=1}^{p} \mathbf{A}_{ij}\mathbf{q}_j = \mathbf{f}_i, \quad i = 1, 2, \ldots, p.$$



The original matrix

After compression, we have a $pk \times pk$ linear system

$$\mathbf{D}_{ii}\tilde{\mathbf{q}}_i + \sum_{i \neq j} \tilde{\mathbf{A}}_{ij}\tilde{\mathbf{q}}_j = \tilde{\mathbf{f}}_i, \quad i = 1, 2, \ldots, p.$$



The reduced matrix

Recall that k is the ε-rank of $\mathbf{A}_{i,j}$ for $i \neq j$.

The point is that $k < n$.

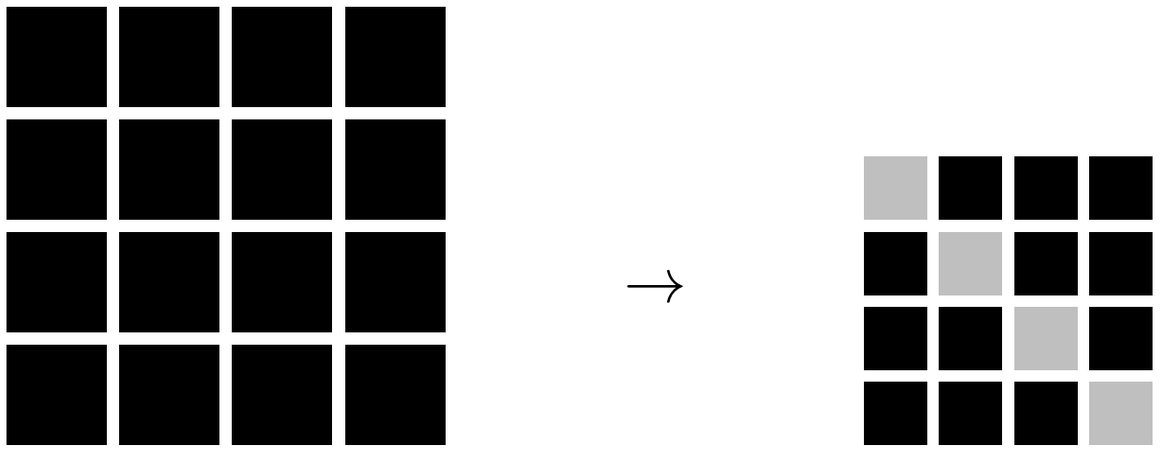
The compression algorithm needs to execute the following steps:

- Compute $\mathbf{U}_i$, $\mathbf{V}_i$, $\tilde{\mathbf{A}}_{ij}$ so that $\mathbf{A}_{ij} = \mathbf{U}_i \, \tilde{\mathbf{A}}_{ij} \, \mathbf{V}_j^*$.
- Compute the new diagonal matrices $\hat{\mathbf{D}}_{ii} = \left(\mathbf{V}_i^* \, \mathbf{A}_{ii}^{-1} \, \mathbf{U}_i\right)^{-1}$.
- Compute the new loads $\tilde{\mathbf{q}}_i = \hat{\mathbf{D}}_{ii} \, \mathbf{V}_i^* \, \mathbf{A}_{ii}^{-1} \, \mathbf{q}_i$.

For the algorithm to be efficient, it has to be able to carry out these steps *locally*.

To achieve this, we use interpolative representations, then $\tilde{\mathbf{A}}_{i,j} = \mathbf{A}(\tilde{I}_i, \tilde{I}_j)$.

We have built a scheme for reducing a system of size $pn \times pn$ to one of size $pk \times pk$.

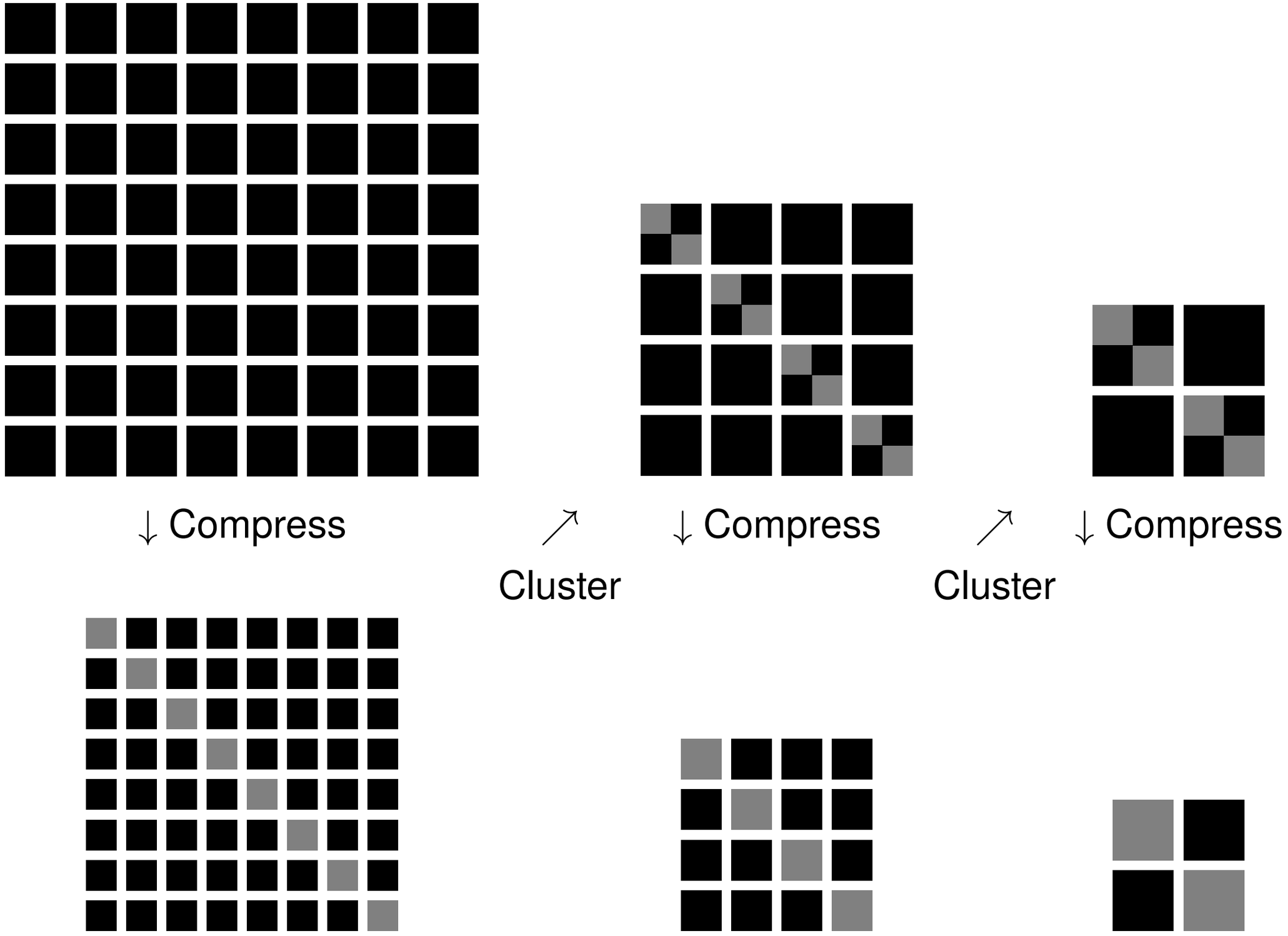


The computational gain is $(k/n)^3$. Good, but not earth-shattering.

Question: How do we get to $O(N)$?

Answer: It turns out that the reduced matrix is itself compressible. Recurse!

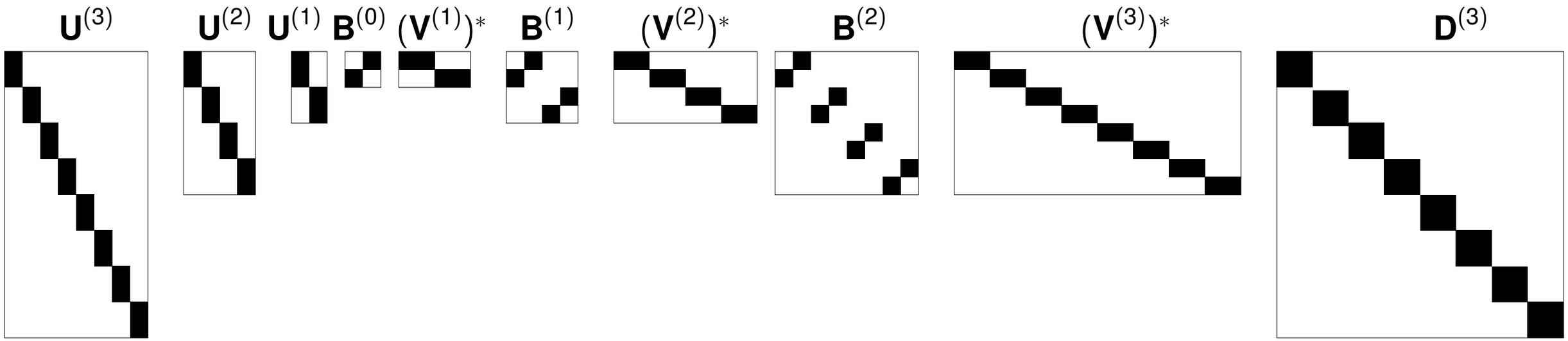
A globally $O(N)$ algorithm is obtained by hierarchically repeating the process:



Formally, one can view this as a telescoping factorization of $\mathbf{A}$:

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

Expressed pictorially, the factorization takes the form



The *inverse of A* then takes the form

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

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

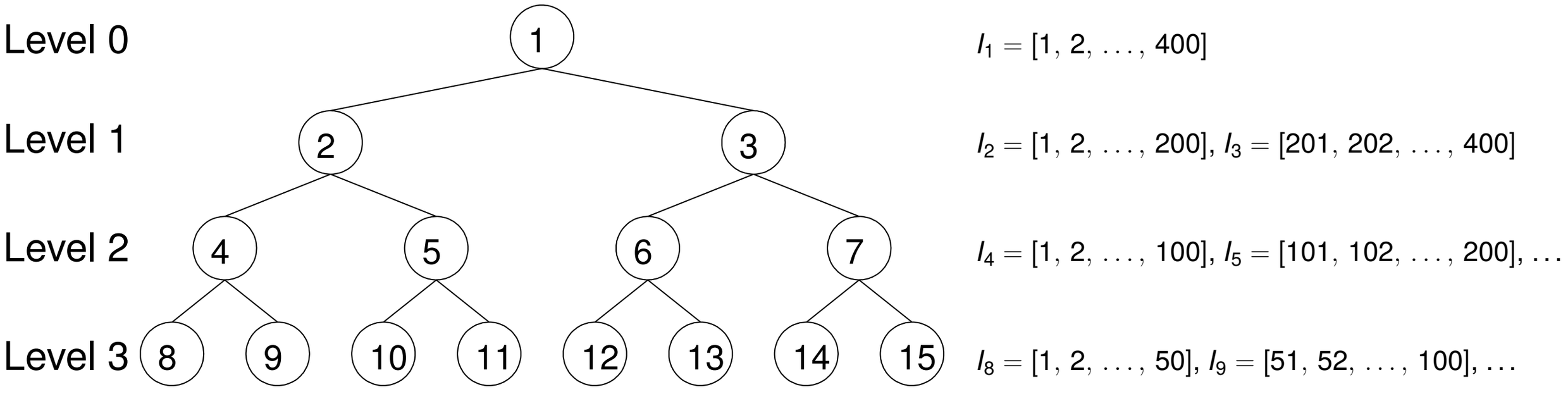
Formal definition of an HBS matrix

Let us first recall the concept of a binary tree on the index vector:

Let **A** be an $N \times N$ matrix.

Suppose $\mathcal{T}$ is a binary tree on the index vector $I = [1, 2, 3, \ldots, N]$.

For a node τ in the tree, let I_τ denote the corresponding index vector.



For nodes σ and τ on the same level, set $\mathbf{A}_{\sigma,\tau} = \mathbf{A}(I_\sigma, I_\tau)$.

Formal definition of an HBS matrix

Suppose $\mathcal{T}$ is a binary tree.

For a node τ in the tree, let I_τ denote the corresponding index vector.

For leaves σ and τ, set $\mathbf{A}_{\sigma,\tau} = \mathbf{A}(I_\sigma, I_\tau)$ and suppose that all off-diagonal blocks satisfy

$$\begin{array}{cccccc} \mathbf{A}_{\sigma,\tau} & = & \mathbf{U}_\sigma & \tilde{\mathbf{A}}_{\sigma,\tau} & \mathbf{V}_\tau^* & \sigma \neq \tau \\ n \times n & & n \times k & k \times k & k \times n & \end{array}$$

For non-leaves σ and τ, let $\{\sigma_1, \sigma_2\}$ denote the children of σ, and let $\{\tau_1, \tau_2\}$ denote the children of τ. Set

$$\mathbf{A}_{\sigma,\tau} = \begin{bmatrix} \tilde{\mathbf{A}}_{\sigma_1,\tau_1} & \tilde{\mathbf{A}}_{\sigma_1,\tau_2} \\ \tilde{\mathbf{A}}_{\sigma_2,\tau_1} & \tilde{\mathbf{A}}_{\sigma_2,\tau_2} \end{bmatrix}$$

Then suppose that the off-diagonal blocks satisfy

$$\begin{array}{cccccc} \mathbf{A}_{\sigma,\tau} & = & \mathbf{U}_\sigma & \tilde{\mathbf{A}}_{\sigma,\tau} & \mathbf{V}_\tau^* & \sigma \neq \tau \\ 2k \times 2k & & 2k \times k & k \times k & k \times 2k & \end{array}$$

An HBS matrix **A** *associated with a tree* $\mathcal{T}$ *is specified by the following factors:*

	Name:	Size:	Function:
For each leaf	$\mathbf{D}_\tau$	$n \times n$	The diagonal block $\mathbf{A}(I_\tau, I_\tau)$.
node τ:	$\mathbf{U}_\tau$	$n \times k$	Basis for the columns in the blocks in row τ.
	$\mathbf{V}_\tau$	$n \times k$	Basis for the rows in the blocks in column τ.
For each parent	$\mathbf{B}_\tau$	$2k \times 2k$	Interactions between the children of τ.
node τ:	$\mathbf{U}_\tau$	$2k \times k$	Basis for the columns in the (reduced) blocks in row τ.
	$\mathbf{V}_\tau$	$2k \times k$	Basis for the rows in the (reduced) blocks in column τ.

Inversion of an HBS matrix

loop over all levels, finer to coarser, $\ell = L, L-1, \ldots, 1$

  loop over all boxes τ on level ℓ,

    if τ is a leaf node

      $\mathbf{X} = \mathbf{D}_\tau$

    else

      Let σ_1 and σ_2 denote the children of τ.

      $$\mathbf{X} = \begin{bmatrix} \mathbf{D}_{\sigma_1} & \mathbf{B}_{\sigma_1,\sigma_2} \\ \mathbf{B}_{\sigma_2,\sigma_1} & \mathbf{D}_{\sigma_2} \end{bmatrix}$$

    end if

    $\mathbf{D}_\tau = \left(\mathbf{V}_\tau^* \mathbf{X}^{-1} \mathbf{U}_\tau\right)^{-1}$.

    $\mathbf{E}_\tau = \mathbf{X}^{-1} \mathbf{U}_\tau \mathbf{D}_\tau$.

    $\mathbf{F}_\tau^* = \mathbf{D}_\tau \mathbf{V}_\tau^* \mathbf{X}^{-1}$.

    $\mathbf{G}_\tau = \mathbf{X}^{-1} - \mathbf{X}^{-1} \mathbf{U}_\tau \mathbf{D}_\tau \mathbf{V}_\tau^* \mathbf{X}^{-1}$.

  end loop

end loop

$$\mathbf{G}_1 = \begin{bmatrix} \mathbf{D}_2 & \mathbf{B}_{2,3} \\ \mathbf{B}_{3,2} & \mathbf{D}_3 \end{bmatrix}^{-1}.$$

```
function EFG = OMNI_invert_HBS_nsym(NODES)
nboxes = size(NODES,2);
EFG = cell(3,nboxes);
ATD_VEC = cell(1,nboxes);
% Loop over all nodes, from finest to coarser.
for ibox = nboxes:(-1):2
   % Assemble the diagonal matrix.
   if (NODES{5,ibox}==0) % ibox is a leaf.
      AD = NODES{40,ibox};
   elseif (NODES{5,ibox}==2) % ibox has precisely two children
      ison1 = NODES{4,ibox}(1);
      ison2 = NODES{4,ibox}(2);
      AD = [ATD_VEC{ison1},NODES{46,ison1};NODES{46,ison2},ATD_VEC{ison2}];
   end
   % Extract the matrices U and V.
   U = NODES{38,ibox};
   V = NODES{39,ibox};
   % Construct the various projection maps.
   ADinv = inv(AD);
   ATD = inv(V'*ADinv*U);
   ATD_VEC{ibox} = ATD;
   EFG{1,ibox} = ADinv*U*ATD;
   EFG{2,ibox} = ATD*(V')*ADinv;
   EFG{3,ibox} = ADinv - EFG{1,ibox}*(V'*ADinv);
end
% Assemble the "top matrix" and invert it:
AT = [ATD_VEC{2},NODES{46,2};NODES{46,3},ATD_VEC{3}];
EFG{3,1} = inv(AT);
return
```

Now let us return to the question of how to compute a block-separable factorization of a matrix $\mathbf{A}$, where the low-rank factorization is based on an *interpolative decomposition.*

Example: Consider an $N \times N$ matrix $\mathbf{A}$, and a partitioning of the index vector

$$I = \{1, 2, 3, \ldots, N\} = I_4 \cup I_5 \cup I_6 \cup I_7.$$

We then seek to determine matrices $\{\mathbf{U}_\tau, \mathbf{V}_\tau\}_{\tau=4}^{7}$ and index vectors $\tilde{I}_\kappa \subset I_\kappa$ such that

$$\mathbf{A}(I_\tau, I_\sigma) = \mathbf{U}_\tau \, \tilde{\mathbf{A}}_{\tau,\sigma} \, \mathbf{V}_\sigma^*, \qquad \sigma \neq \tau,$$

where $\tilde{\mathbf{A}}_{\tau,\sigma} = \mathbf{A}(\tilde{I}_\tau, \tilde{I}_\sigma)$ is a submatrix of $\mathbf{A}_{\tau,\sigma}$.

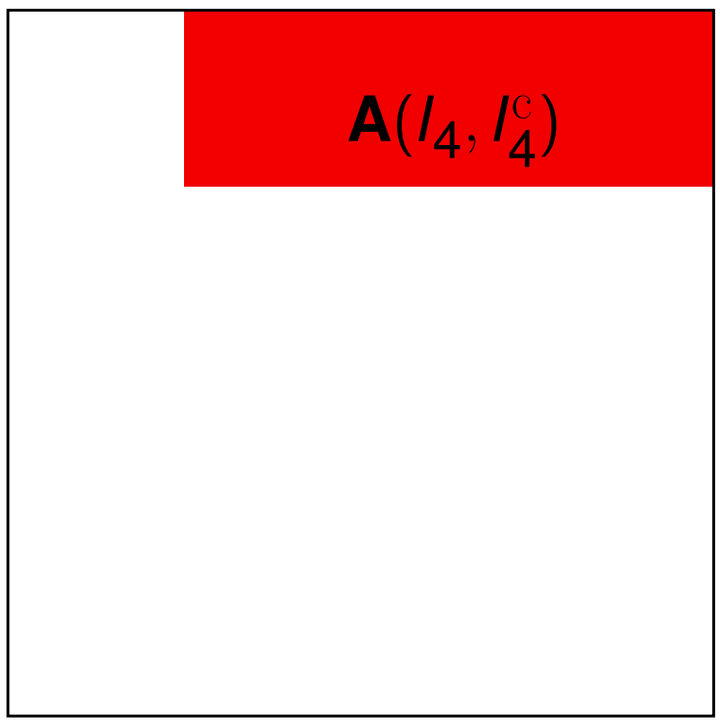
In other words, we seek a factorization

$$\mathbf{A} = \underbrace{\begin{bmatrix} \mathbf{U}_4 & & & \\ & \mathbf{U}_5 & & \\ & & \mathbf{U}_6 & \\ & & & \mathbf{U}_7 \end{bmatrix}}_{=\mathbf{U}} \underbrace{\begin{bmatrix} \mathbf{0} & \tilde{\mathbf{A}}_{45} & \tilde{\mathbf{A}}_{46} & \tilde{\mathbf{A}}_{47} \\ \tilde{\mathbf{A}}_{54} & \mathbf{0} & \tilde{\mathbf{A}}_{56} & \tilde{\mathbf{A}}_{57} \\ \tilde{\mathbf{A}}_{64} & \tilde{\mathbf{A}}_{65} & \mathbf{0} & \tilde{\mathbf{A}}_{67} \\ \tilde{\mathbf{A}}_{74} & \tilde{\mathbf{A}}_{75} & \tilde{\mathbf{A}}_{76} & \mathbf{0} \end{bmatrix}}_{=\tilde{\mathbf{A}}} \underbrace{\begin{bmatrix} \mathbf{V}_4^* & & & \\ & \mathbf{V}_5^* & & \\ & & \mathbf{V}_6^* & \\ & & & \mathbf{V}_7^* \end{bmatrix}}_{=\mathbf{V}^*} + \underbrace{\begin{bmatrix} \mathbf{D}_4 & & & \\ & \mathbf{D}_5 & & \\ & & \mathbf{D}_6 & \\ & & & \mathbf{D}_7 \end{bmatrix}}_{=\mathbf{D}}.$$

What is the role of the basis matrices $\mathbf{U}_\tau$ *and* $\mathbf{V}_\tau$*?*

Recall our toy example: $\mathbf{A} = \begin{bmatrix} \mathbf{D}_4 & \mathbf{U}_4 \tilde{\mathbf{A}}_{45} \mathbf{V}_5^* & \mathbf{U}_4 \tilde{\mathbf{A}}_{46} \mathbf{V}_6^* & \mathbf{U}_4 \tilde{\mathbf{A}}_{47} \mathbf{V}_7^* \\ \mathbf{U}_5 \tilde{\mathbf{A}}_{54} \mathbf{V}_4^* & \mathbf{D}_5 & \mathbf{U}_5 \tilde{\mathbf{A}}_{56} \mathbf{V}_6^* & \mathbf{U}_5 \tilde{\mathbf{A}}_{57} \mathbf{V}_7^* \\ \mathbf{U}_6 \tilde{\mathbf{A}}_{64} \mathbf{V}_4^* & \mathbf{U}_6 \tilde{\mathbf{A}}_{65} \mathbf{V}_5^* & \mathbf{D}_6 & \mathbf{U}_6 \tilde{\mathbf{A}}_{67} \mathbf{V}_7^* \\ \mathbf{U}_7 \tilde{\mathbf{A}}_{74} \mathbf{V}_4^* & \mathbf{U}_7 \tilde{\mathbf{A}}_{75} \mathbf{V}_5^* & \mathbf{U}_7 \tilde{\mathbf{A}}_{76} \mathbf{V}_6^* & \mathbf{D}_7 \end{bmatrix}$.

We see that the columns of $\mathbf{U}_4$ must span the column space of the matrix $\mathbf{A}(I_4, I_4^c)$ where I_4 is the index vector for the first block and $I_4^c = I \backslash I_4$.

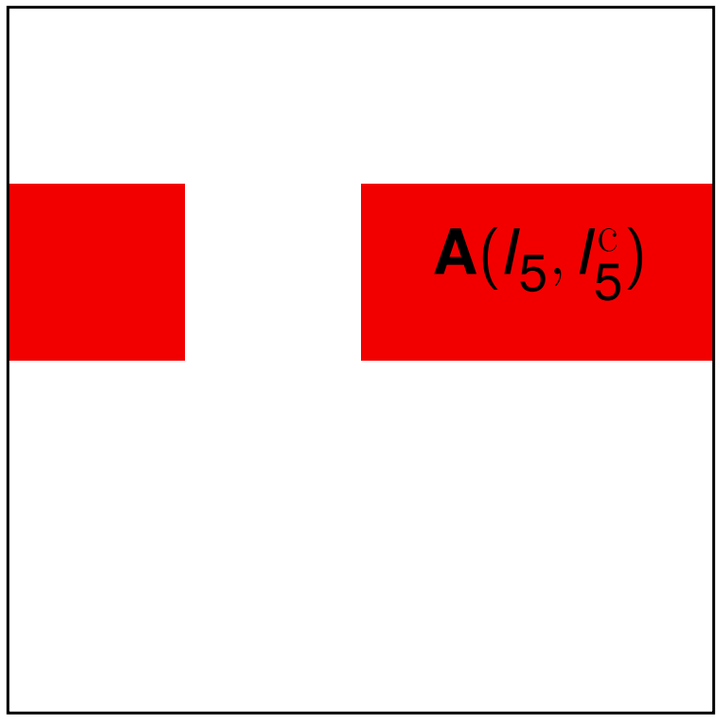


The matrix $\mathbf{A}$

What is the role of the basis matrices $\mathbf{U}_\tau$ and $\mathbf{V}_\tau$?

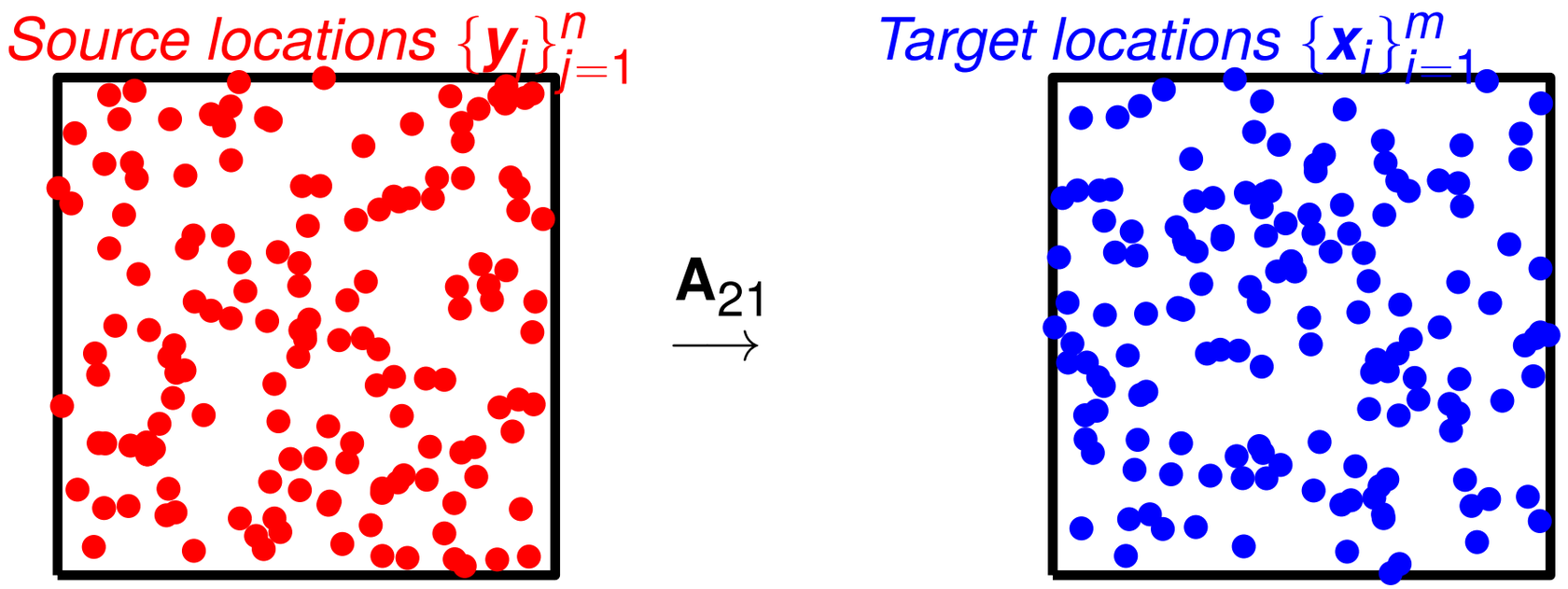
Recall our toy example: $\mathbf{A} = \begin{bmatrix} \mathbf{D}_4 & \mathbf{U}_4\tilde{\mathbf{A}}_{45}\mathbf{V}_5^* & \mathbf{U}_4\tilde{\mathbf{A}}_{46}\mathbf{V}_6^* & \mathbf{U}_4\tilde{\mathbf{A}}_{47}\mathbf{V}_7^* \\ \mathbf{U}_5\tilde{\mathbf{A}}_{54}\mathbf{V}_4^* & \mathbf{D}_5 & \mathbf{U}_5\tilde{\mathbf{A}}_{56}\mathbf{V}_6^* & \mathbf{U}_5\tilde{\mathbf{A}}_{57}\mathbf{V}_7^* \\ \mathbf{U}_6\tilde{\mathbf{A}}_{64}\mathbf{V}_4^* & \mathbf{U}_6\tilde{\mathbf{A}}_{65}\mathbf{V}_5^* & \mathbf{D}_6 & \mathbf{U}_6\tilde{\mathbf{A}}_{67}\mathbf{V}_7^* \\ \mathbf{U}_7\tilde{\mathbf{A}}_{74}\mathbf{V}_4^* & \mathbf{U}_7\tilde{\mathbf{A}}_{75}\mathbf{V}_5^* & \mathbf{U}_7\tilde{\mathbf{A}}_{76}\mathbf{V}_6^* & \mathbf{D}_7 \end{bmatrix}$.

We see that the columns of $\mathbf{U}_5$ must span the column space of the matrix $\mathbf{A}(I_5, I_5^{\mathrm{c}})$ where I_5 is the index vector for the first block and $I_5^{\mathrm{c}} = I \backslash I_5$.



The matrix $\mathbf{A}$

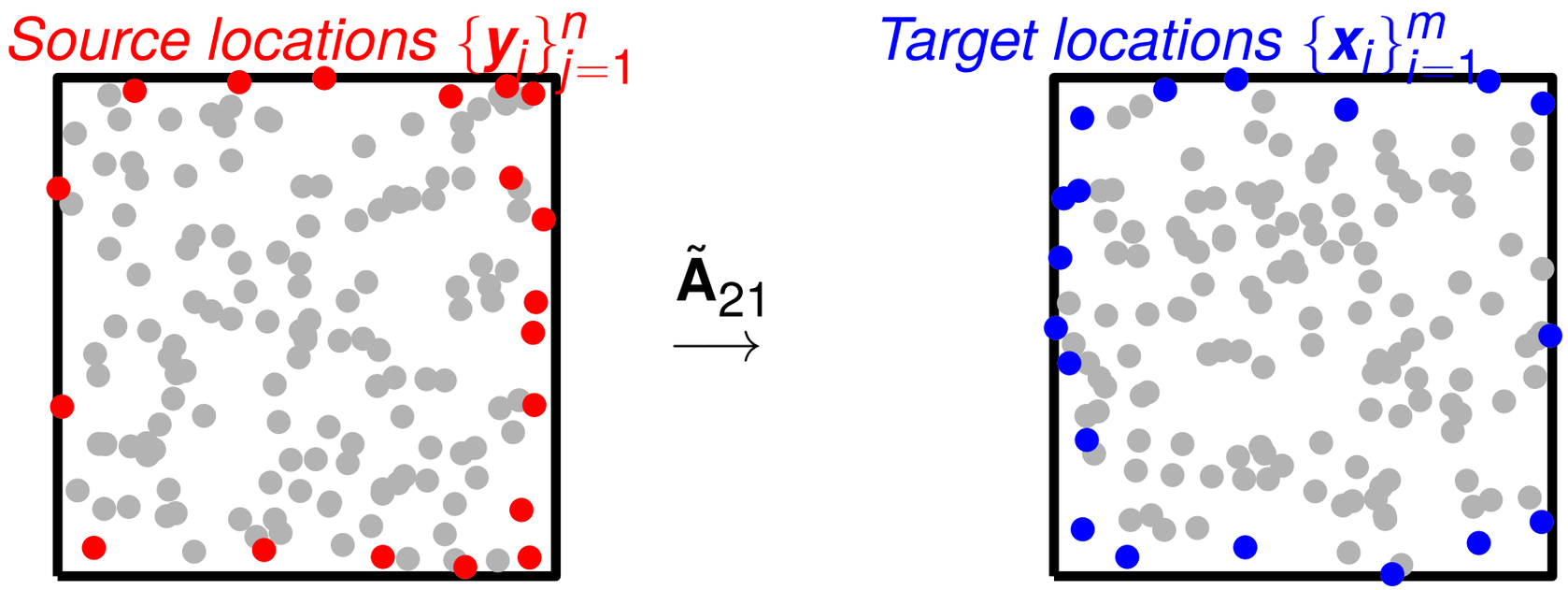
As mentioned earlier, it is handy to use the *interpolative decomposition (ID)*, in which $\mathbf{U}_\tau$ and $\mathbf{V}_\tau$ contain identity matrices. To review how this works, consider a situation with n sources in a domain Ω_1 inducing m potentials in a different domain Ω_2.



Let $\mathbf{A}_{21}$ denote the $m \times n$ matrix with entries $\mathbf{A}_{21}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{y}_j|$. Then

$$\begin{array}{ccccc} \mathbf{f} & = & \mathbf{A}_{21} & \mathbf{q} \\ m \times 1 & & m \times n & n \times 1 \end{array}$$

As mentioned earlier, it is handy to use the *interpolative decomposition (ID)*, in which $\mathbf{U}_\tau$ and $\mathbf{V}_\tau$ contain identity matrices. To review how this works, consider a situation with n sources in a domain Ω_1 inducing m potentials in a different domain Ω_2.



Let $\mathbf{A}_{21}$ denote the $m \times n$ matrix with entries $\mathbf{A}_{21}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{y}_j|$. Then

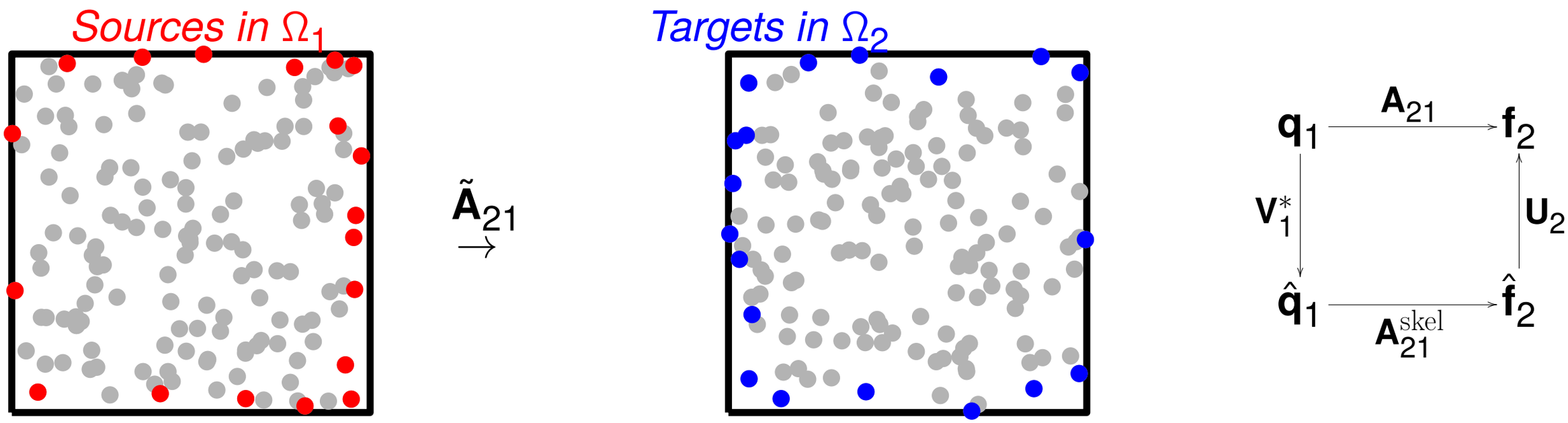
$$\underset{m\times 1}{\mathbf{f}} = \underset{m\times n}{\mathbf{A}_{21}}\ \underset{n\times 1}{\mathbf{q}} = \underset{m\times k}{\mathbf{U}_2}\ \underset{k\times k}{\tilde{\mathbf{A}}_{21}}\ \underset{k\times n}{\mathbf{V}_1^*}\ \underset{n\times 1}{\mathbf{q}}$$

where $\tilde{\mathbf{A}}_{21} = \mathbf{A}_{21}(\tilde{I}_2, \tilde{I}_1)$ is a $k \times k$ submatrix of $\mathbf{A}$.

The index vector $\tilde{I}_1 \subseteq \{1, 2, \ldots, n\}$ marks the chosen *skeleton source locations.*

The index vector $\tilde{I}_2 \subseteq \{1, 2, \ldots, m\}$ marks the chosen *skeleton target locations.*

Review of ID: Consider a rank-k factorization of an $m \times n$ matrix: $\mathbf{A}_{21} = \mathbf{U}_2 \, \tilde{\mathbf{A}}_{21} \, \mathbf{V}_1^*$

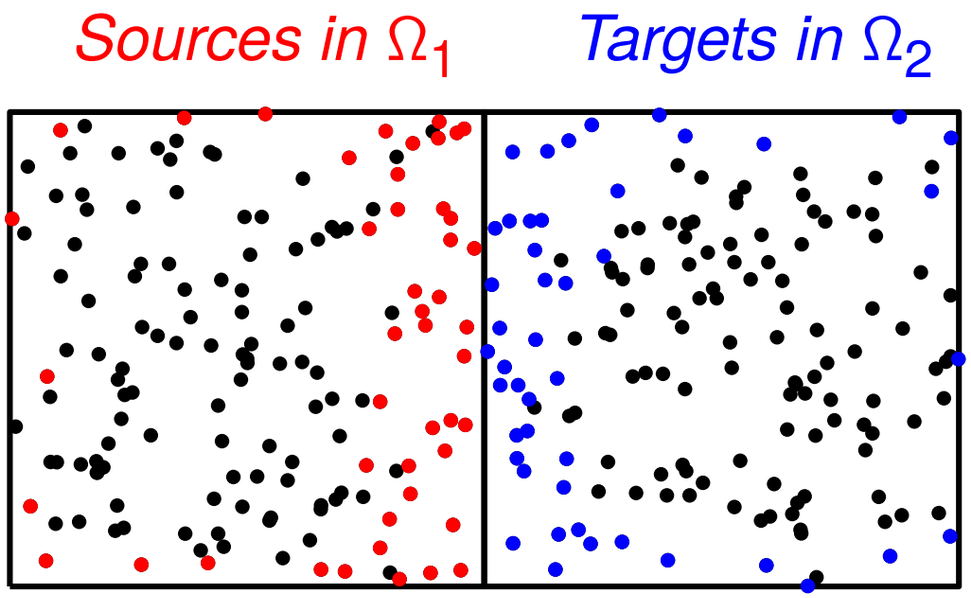


To precision 10^{-10}*, the rank is* 19.

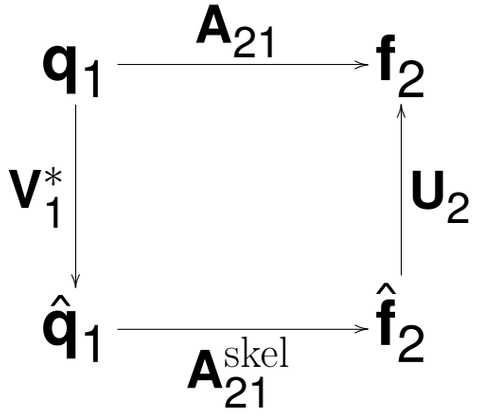
Advantages of the ID:

- The rank k is typically close to optimal.
- Applying $\mathbf{V}_1^*$ and $\mathbf{U}_2$ is cheap — they both contain $k \times k$ identity matrices.
- The matrices $\mathbf{V}_1^*$ and $\mathbf{U}_2$ are well-conditioned.
- Finding the k points is cheap — simply use Gaussian elimination.
- The map $\tilde{\mathbf{A}}_{12}$ is simply a restriction of the original map $\mathbf{A}_{12}$.
  (We loosely say that "the physics of the problem is preserved".)
- Interaction between **adjacent** boxes can be compressed (no buffering required).

Review of ID: Consider a rank-k factorization of an $m \times n$ matrix: $\mathbf{A}_{21} = \mathbf{U}_2 \, \tilde{\mathbf{A}}_{21} \, \mathbf{V}_1^*$



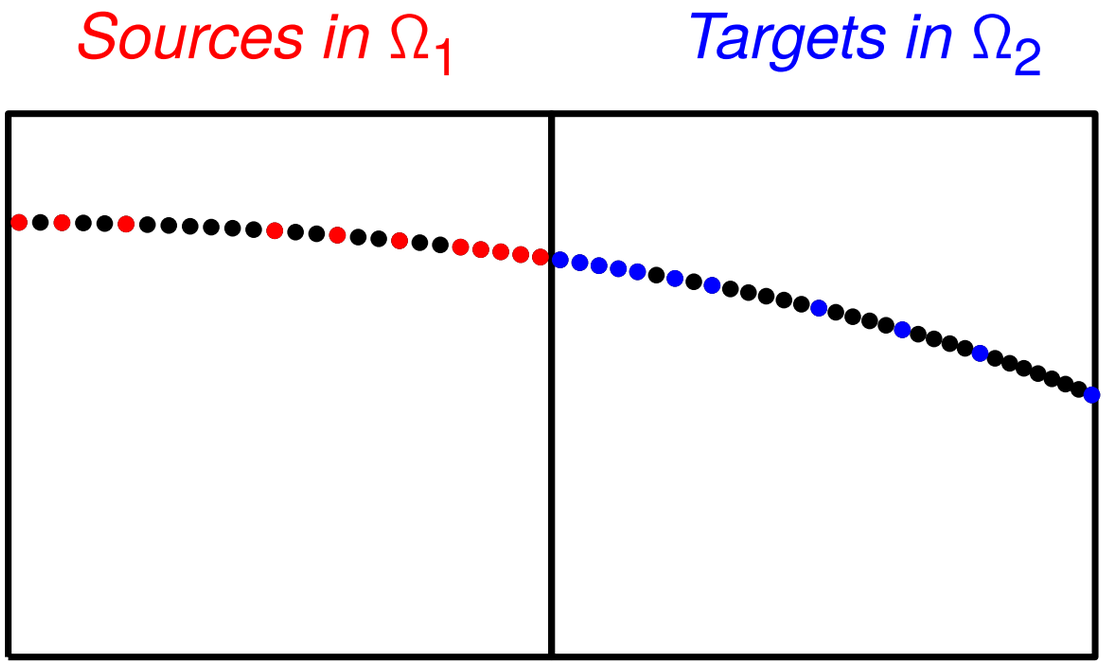
To precision 10^{-10}*, the rank is* 46.



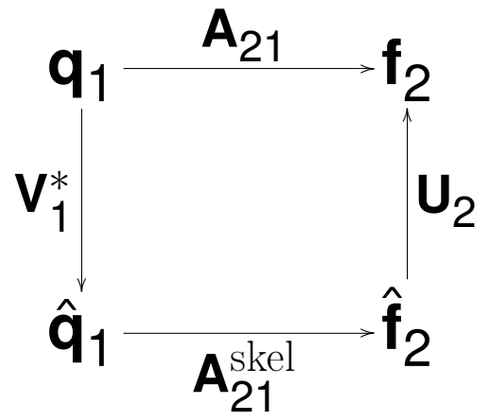
Advantages of the ID:

- The rank is k is typically close to optimal.
- Applying $\mathbf{V}_1^*$ and $\mathbf{U}_2$ is cheap — they both contain $k \times k$ identity matrices.
- The matrices $\mathbf{V}_1^*$ and $\mathbf{U}_2$ are well-conditioned.
- Finding the k points is cheap — simply use Gaussian elimination.
- The map $\tilde{A}_{12}$ is simply a restriction of the original map A_{12}.
  (We loosely say that "the physics of the problem is preserved".)
- Interaction between **adjacent** boxes can be compressed (no buffering required).

Review of ID: Consider a rank-k factorization of an $m \times n$ matrix: $\mathbf{A}_{21} = \mathbf{U}_2 \, \tilde{\mathbf{A}}_{21} \, \mathbf{V}_1^*$



To precision 10^{-10}*, the rank is* 11.

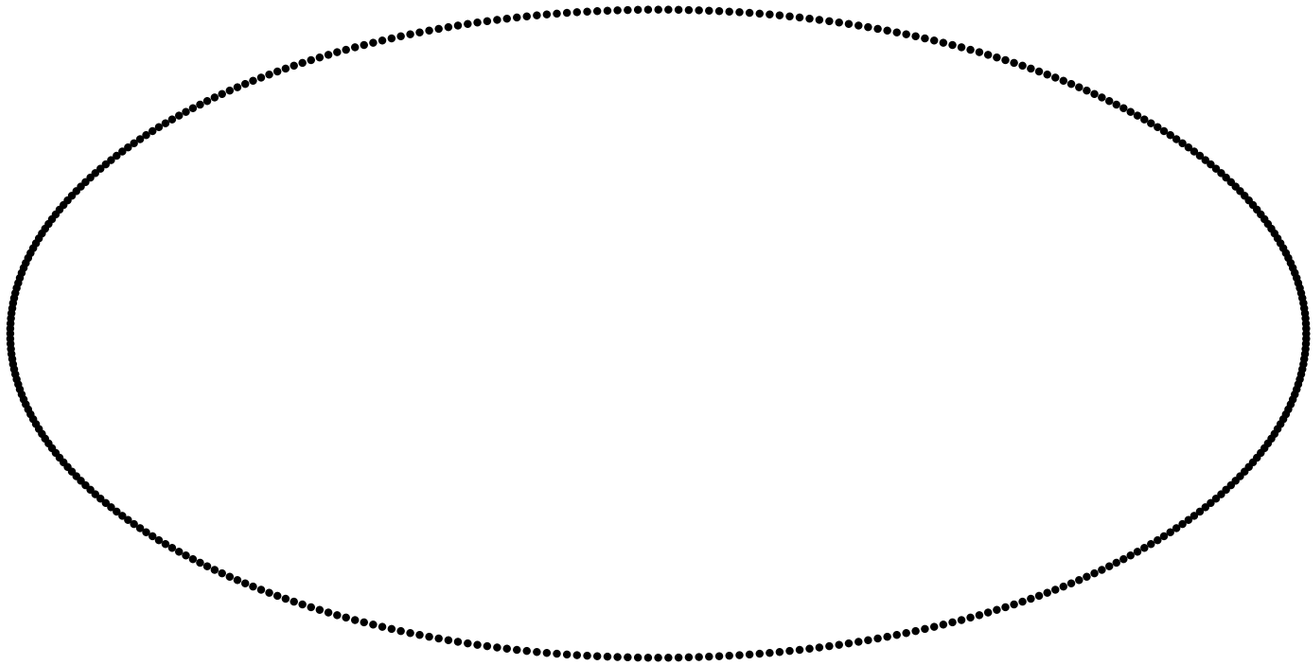


Advantages of the ID:

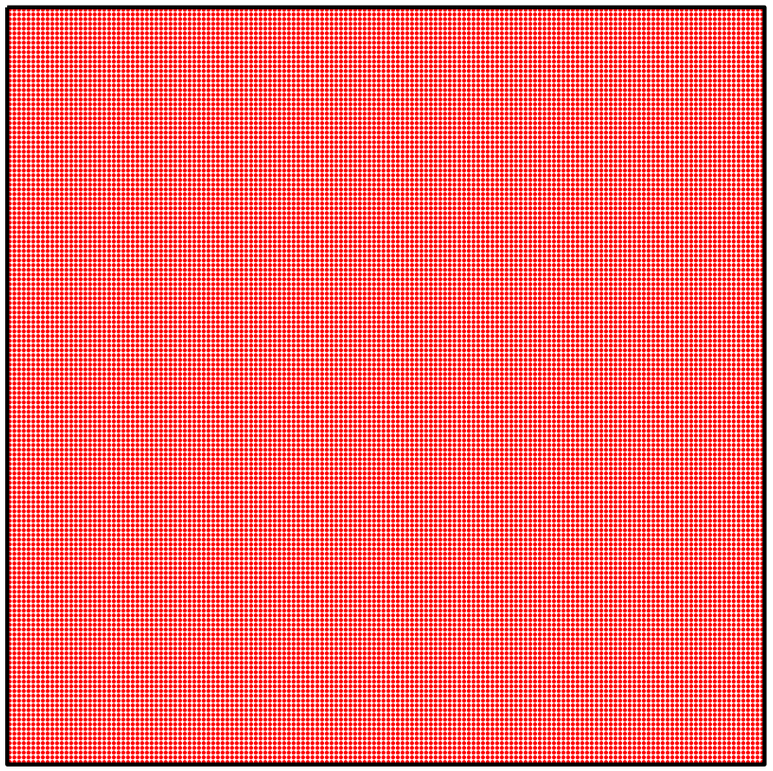
- The rank is k is typically close to optimal.
- Applying $\mathbf{V}_1^*$ and $\mathbf{U}_2$ is cheap — they both contain $k \times k$ identity matrices.
- The matrices $\mathbf{V}_1^*$ and $\mathbf{U}_2$ are well-conditioned.
- Finding the k points is cheap — simply use Gaussian elimination.
- The map $\tilde{A}_{12}$ is simply a restriction of the original map A_{12}.
  (We loosely say that "the physics of the problem is preserved".)
- Interaction between **adjacent** boxes can be compressed (no buffering required).

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



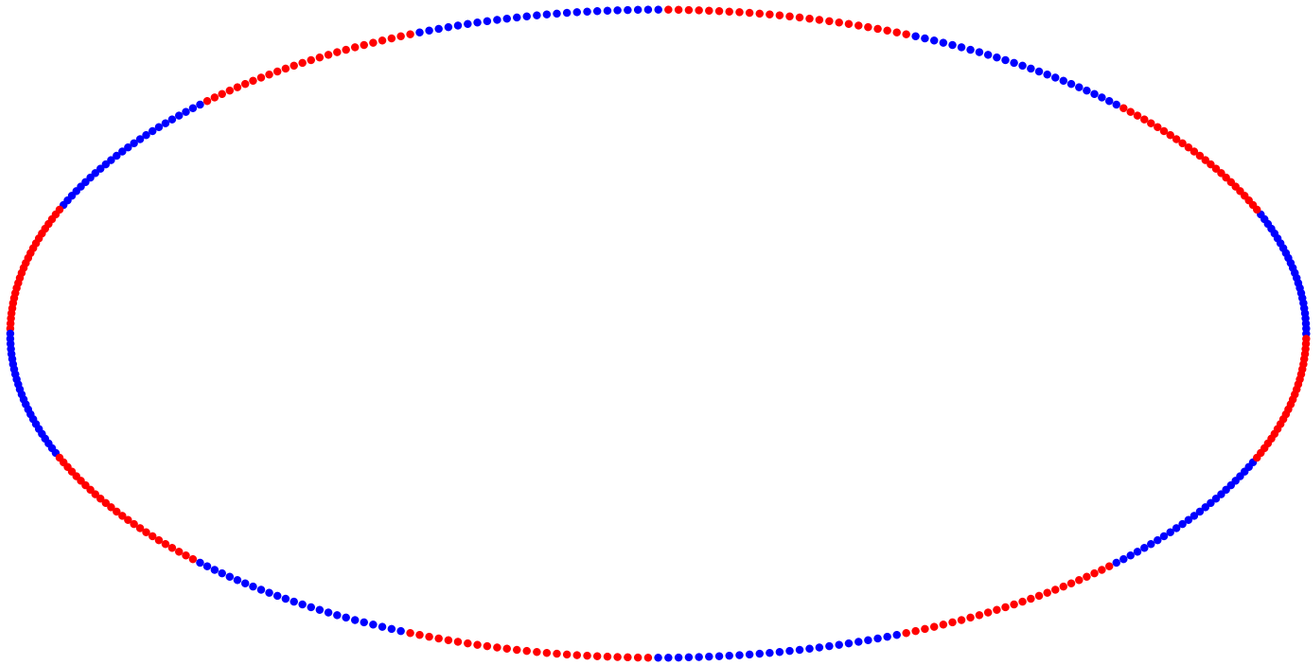
The contour



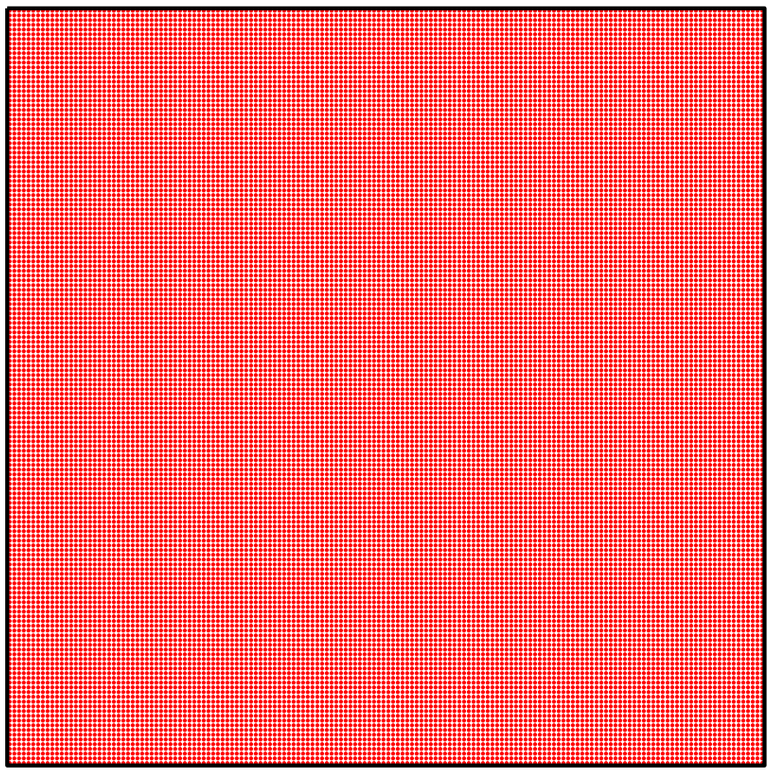
The matrix

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



The contour

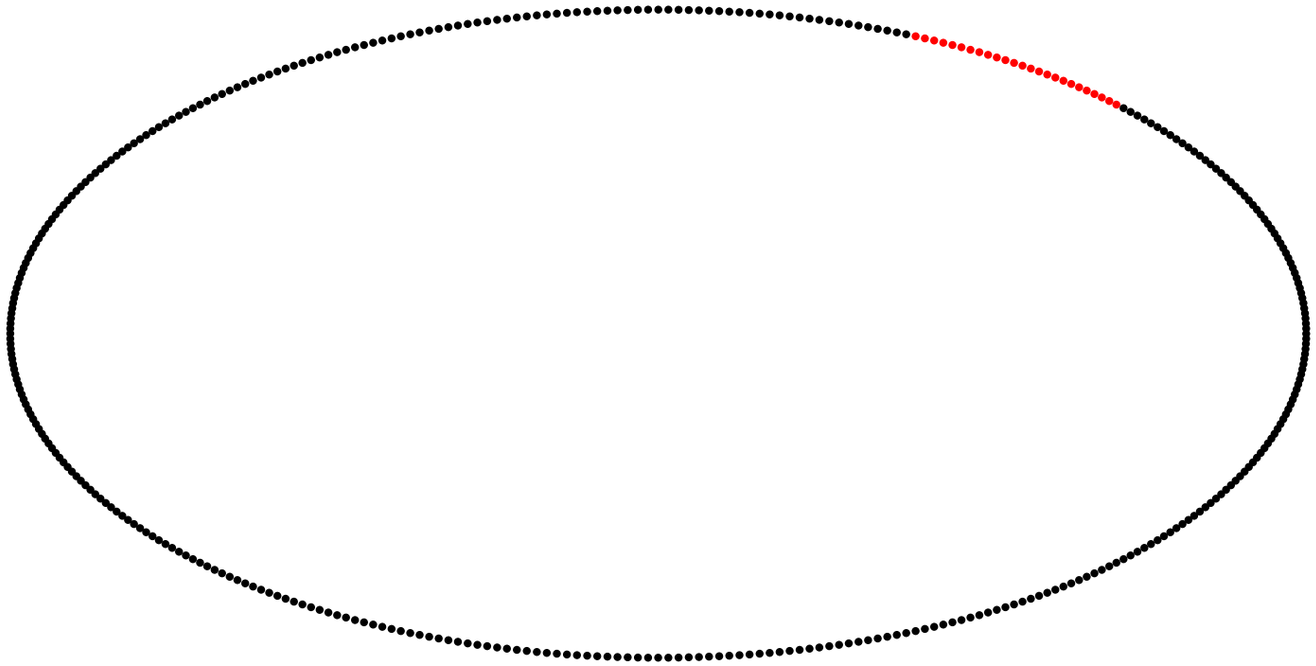


The matrix

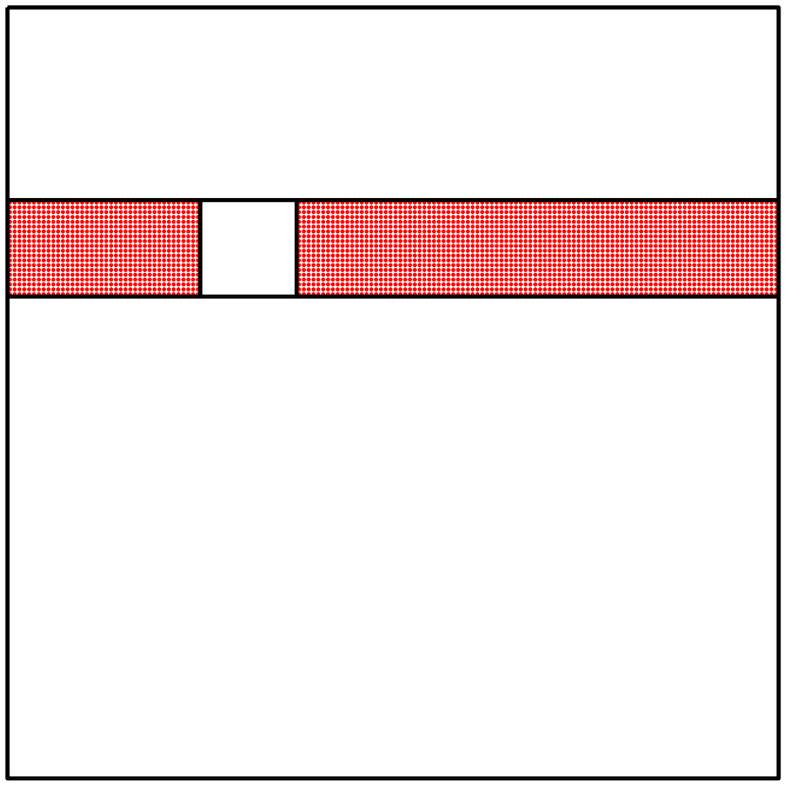
Partition the contour into 16 leaves.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



The contour



The block $\mathbf{A}(I_\tau, I_\tau^c)$ shown in red.

Now let us focus on a single panel Γ_τ associated with index vector I_τ.

Our task is to determine a basis matrix $\mathbf{U}_\tau$ and an index vector $\tilde{I}_\tau \subset I_\tau$ such that

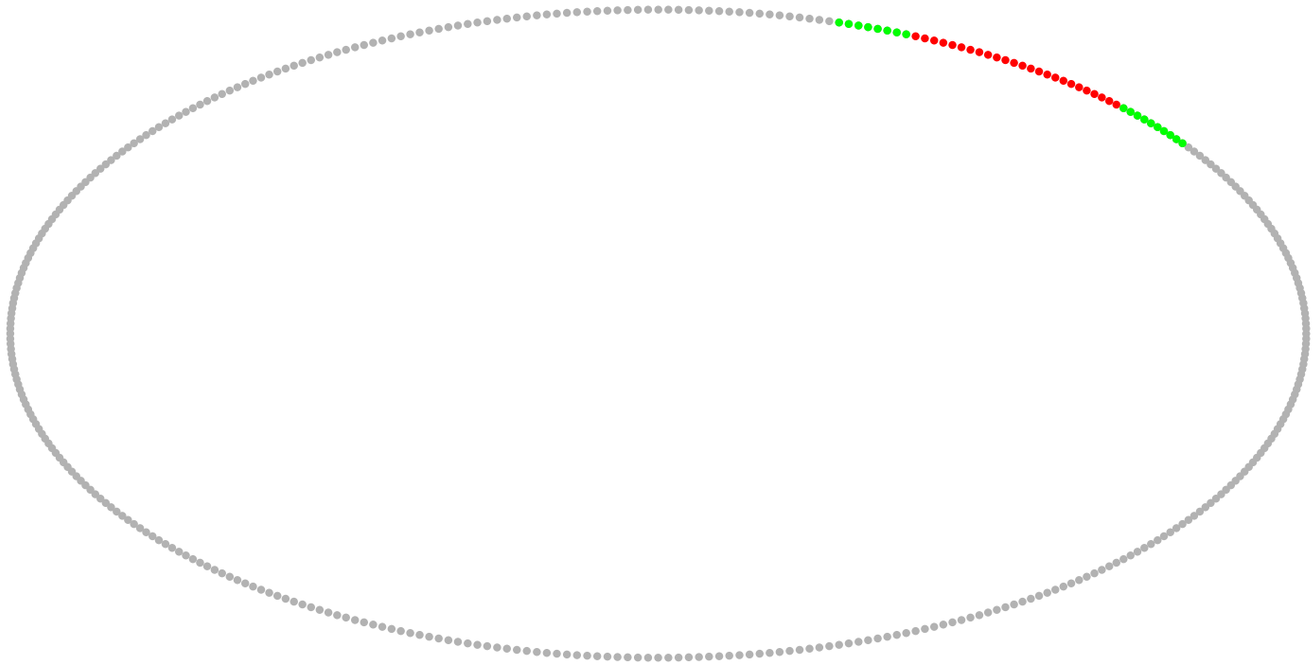
$$\begin{array}{ccccc} \mathbf{A}(I_\tau, I_\tau^c) & = & \mathbf{U}_\tau & & \mathbf{A}(\tilde{I}_\tau, I_\tau^c) \\ n \times (N-n) & & n \times k & & k \times (N-n) \end{array}$$

The most direct way of doing this is to perform Gram-Schmidt on the rows of $\mathbf{A}(I_\tau, I_\tau^c)$.

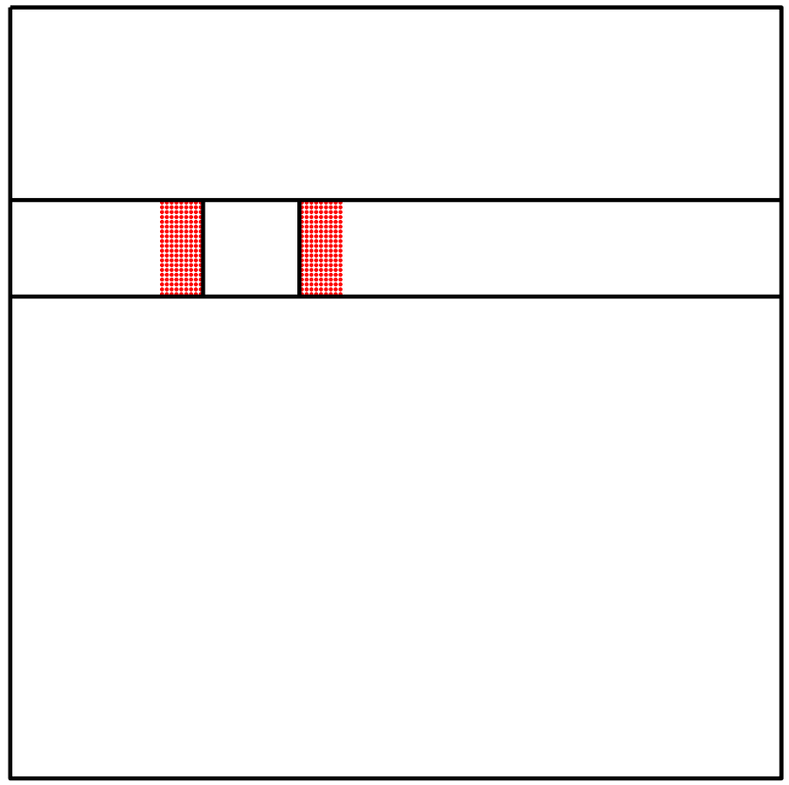
This works great, but it is expensive, since $\mathbf{A}(I_\tau, I_\tau^c)$ is big. We seek a *local* procedure.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



The contour



The block $\mathbf{A}(I_\tau, I_\tau^{(\text{near})})$ shown in red.

Idea (bad): Ignore all charges in the far-field!

Let $I_\tau^{(\text{near})}$ denote the near-field points.

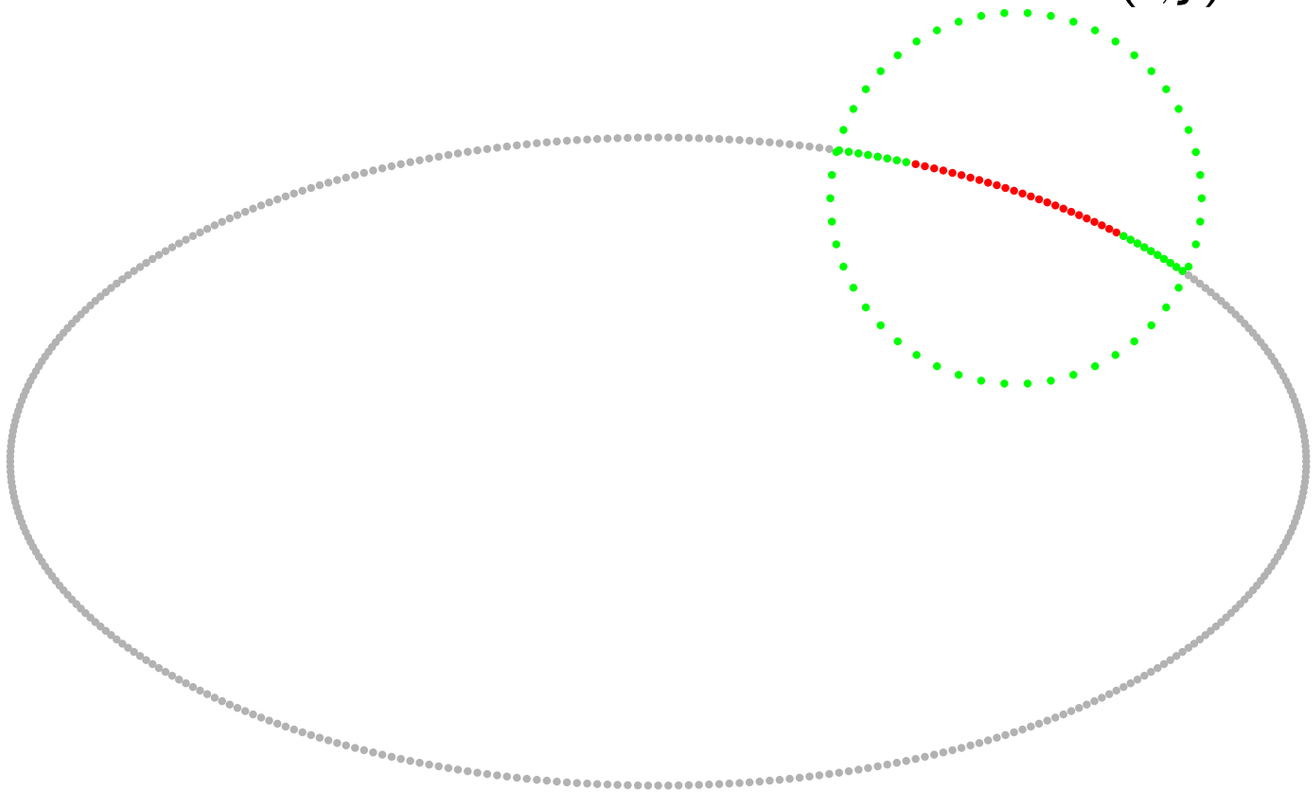
Then factor the smaller matrix $\mathbf{B} = \mathbf{A}(I_\tau, I_\tau^{(\text{near})})$:

$$\begin{array}{ccccc} \mathbf{B} & = & \mathbf{U}_\tau & & \mathbf{B}(J,:) \\ n \times n_{\text{near}} & & n \times k & & k \times n_{\text{near}} \end{array}$$

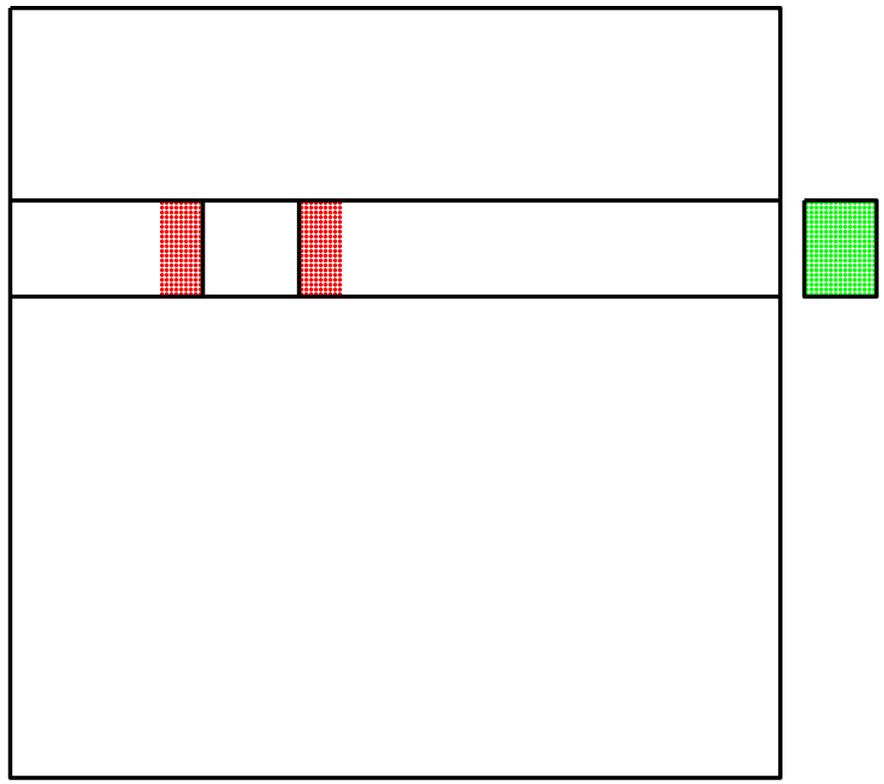
and set $\tilde{I}_\tau = I_\tau(J)$.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^{N}$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



The contour



The block $\mathbf{A}(I_\tau, I_\tau^{(\text{near})})$ shown in red. The block $\mathbf{G}$ shown in green.

Idea: Replace charges in the far-field by "proxy" charges.

Let $I_\tau^{(\text{near})}$ denote the near-field points and let $\mathbf{G}$ denote a matrix of size $n \times n_{\text{proxy}}$ that maps charges on the proxy locations to potentials on Γ_τ.

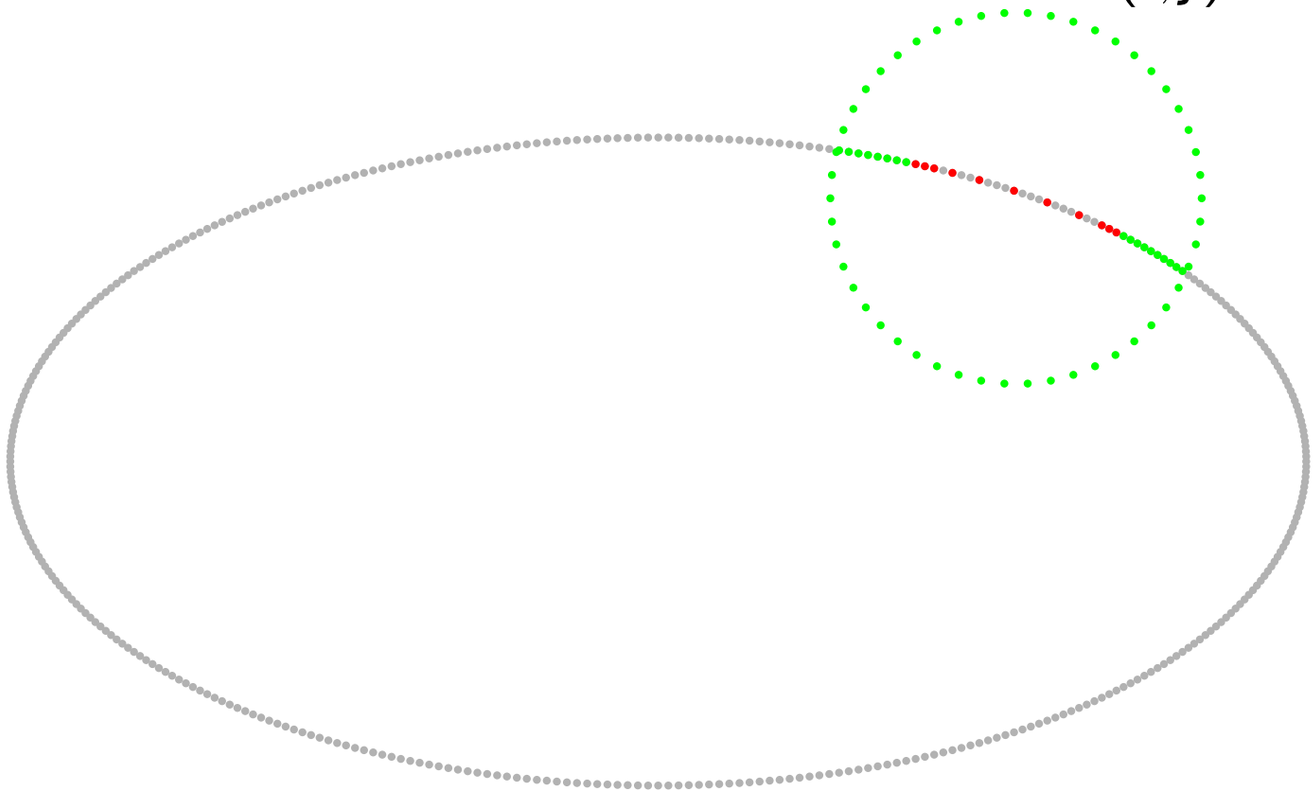
Then factor the smaller matrix $\mathbf{B} = [\mathbf{A}(I_\tau, I_\tau^{(\text{near})}),\ \mathbf{G}]$:

$$\underset{n \times (n_{\text{near}} + n_{\text{proxy}})}{\mathbf{B}} = \underset{n \times k}{\mathbf{U}_\tau} \quad \underset{k \times (n_{\text{near}} + n_{\text{proxy}})}{\mathbf{B}(J,:)}$$

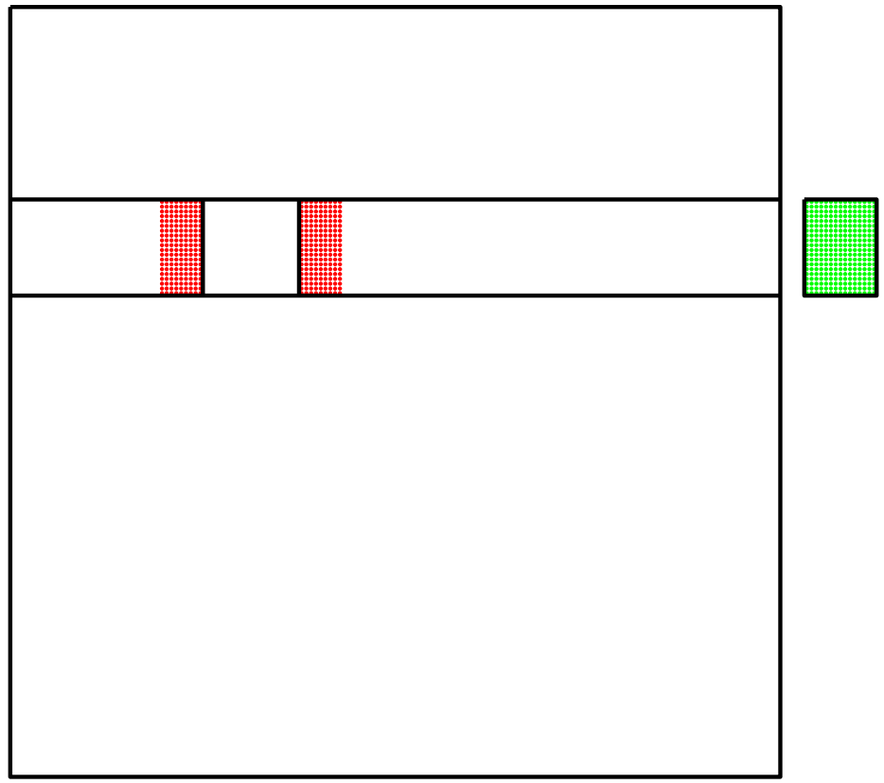
and set $\tilde{I}_\tau = I_\tau(J)$.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



The contour



The block $\mathbf{A}(I_\tau, I_\tau^{(\text{near})})$ *shown in red. The block* $\mathbf{G}$ *shown in green.*

Idea: Replace charges in the far-field by "proxy" charges.

Let $I_\tau^{(\text{near})}$ denote the near-field points and let $\mathbf{G}$ denote a matrix of size $n \times n_{\text{proxy}}$ that maps charges on the proxy locations to potentials on Γ_τ.

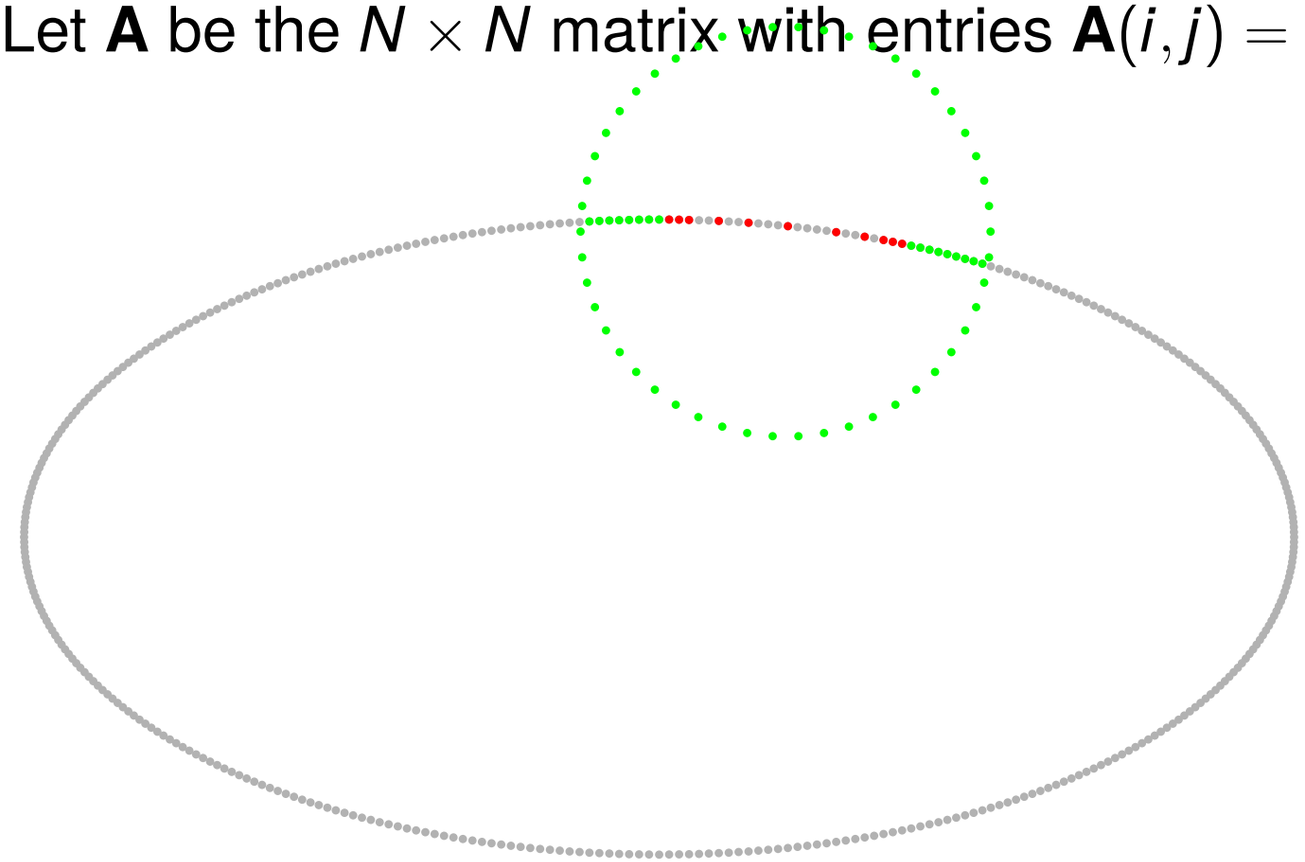
Then factor the smaller matrix $\mathbf{B} = [\mathbf{A}(I_\tau, I_\tau^{(\text{near})}),\ \mathbf{G}]$:

$$\underset{n \times (n_{\text{near}} + n_{\text{proxy}})}{\mathbf{B}} = \underset{n \times k}{\mathbf{U}_\tau} \quad \underset{k \times (n_{\text{near}} + n_{\text{proxy}})}{\mathbf{B}(J,:)}$$

and set $\tilde{I}_\tau = I_\tau(J)$.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^{N}$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



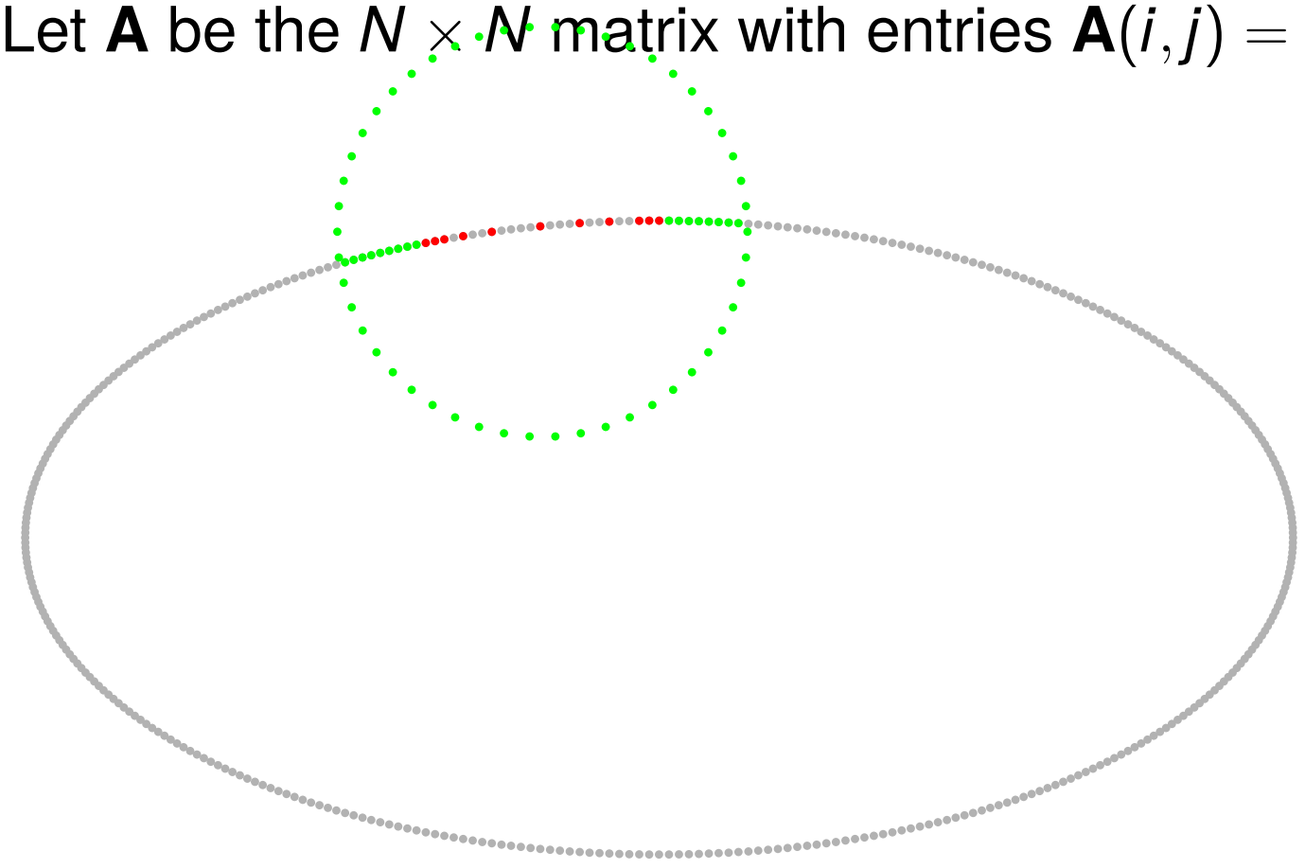
The contour

Idea: Replace charges in the far-field by "proxy" charges.

... execute the same steps for the next panel ...

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



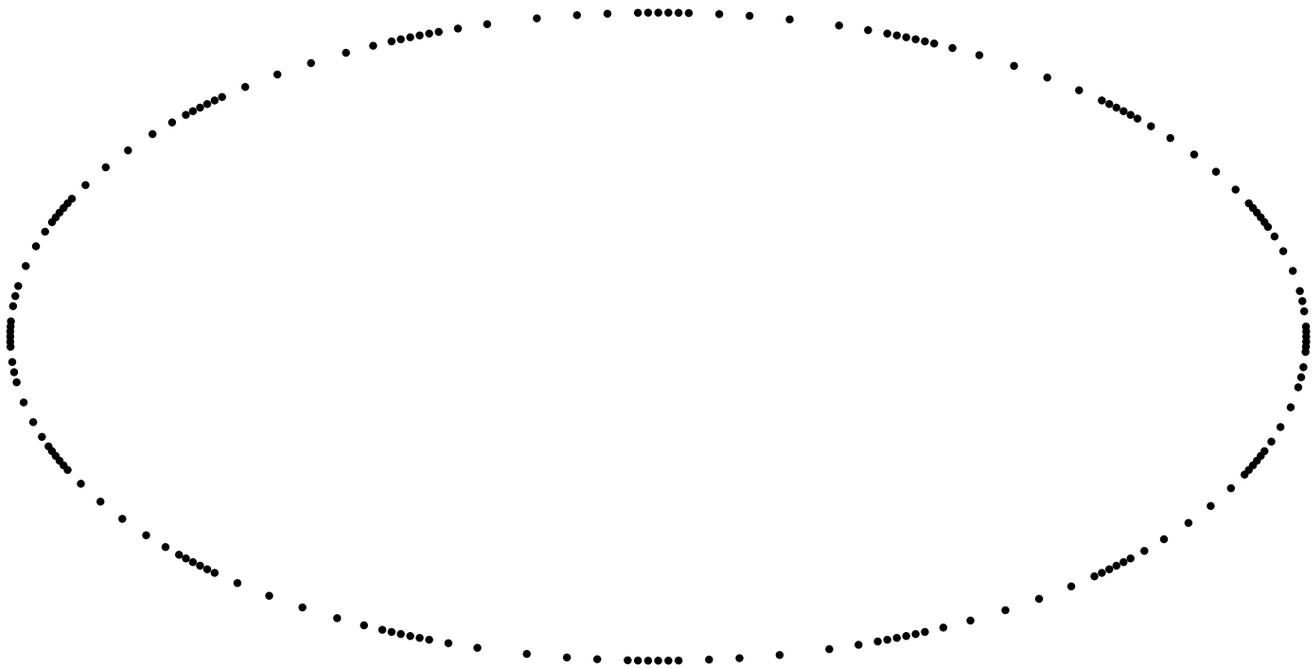
The contour

Idea: Replace charges in the far-field by “proxy” charges.

... and the next ...

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.

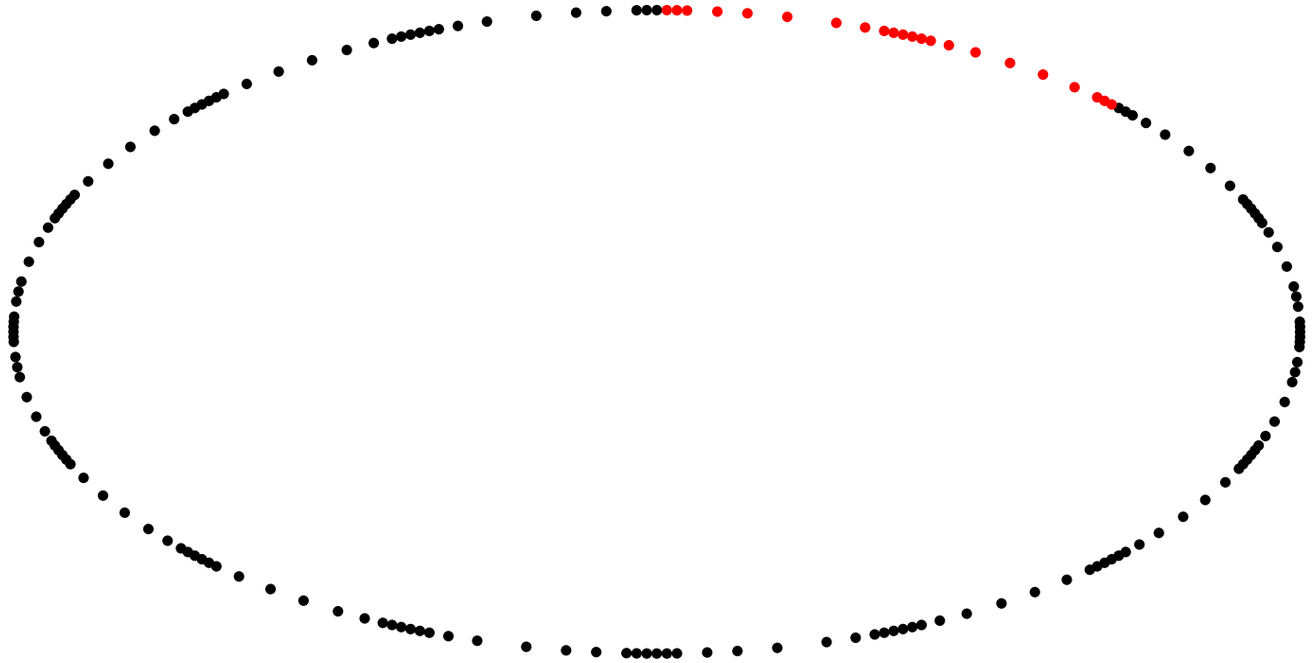


The contour

Once all leaves have been processed, we have in effect eliminated a bunch of points.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



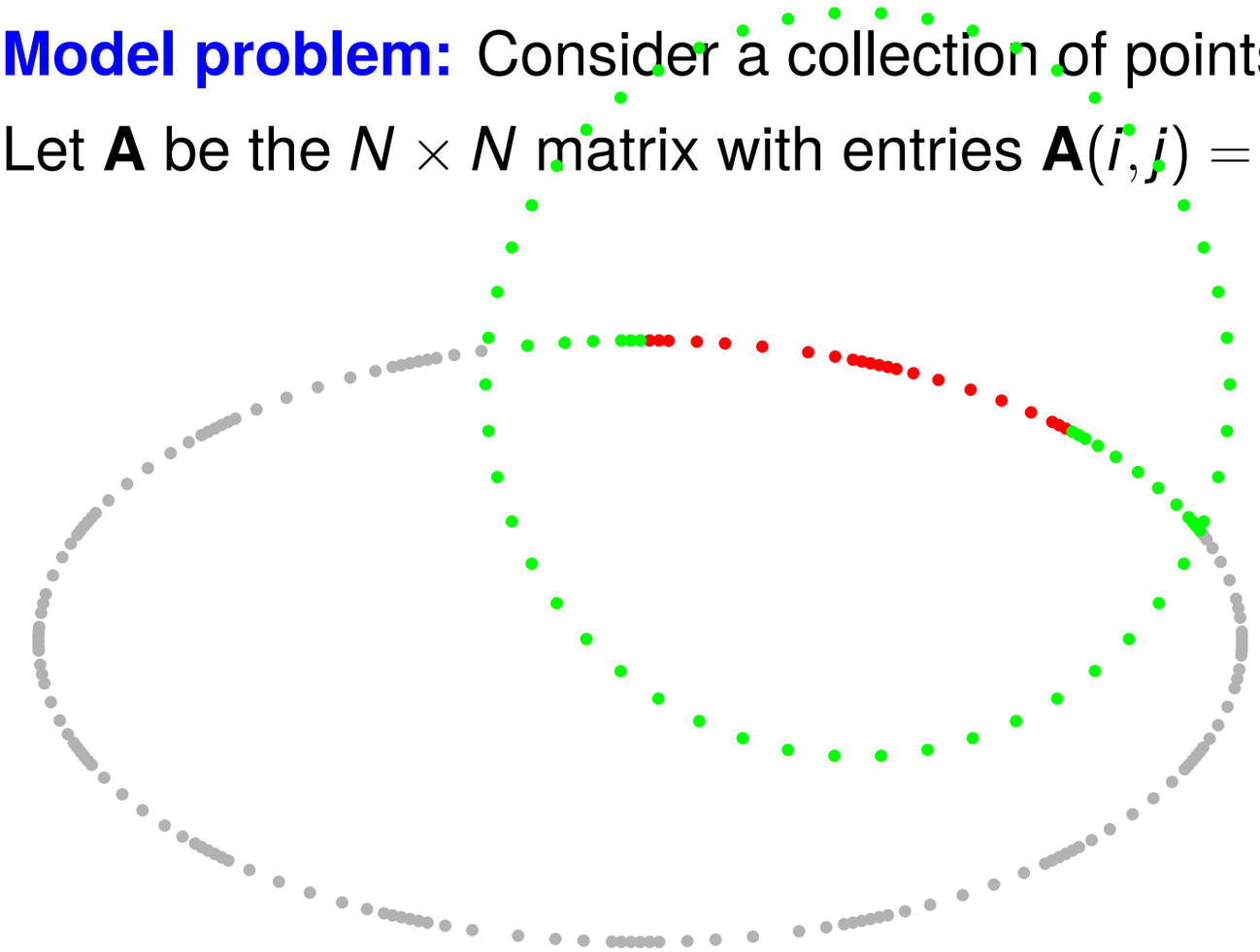
The contour

The matrix

Now consider compression of a parent node.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.

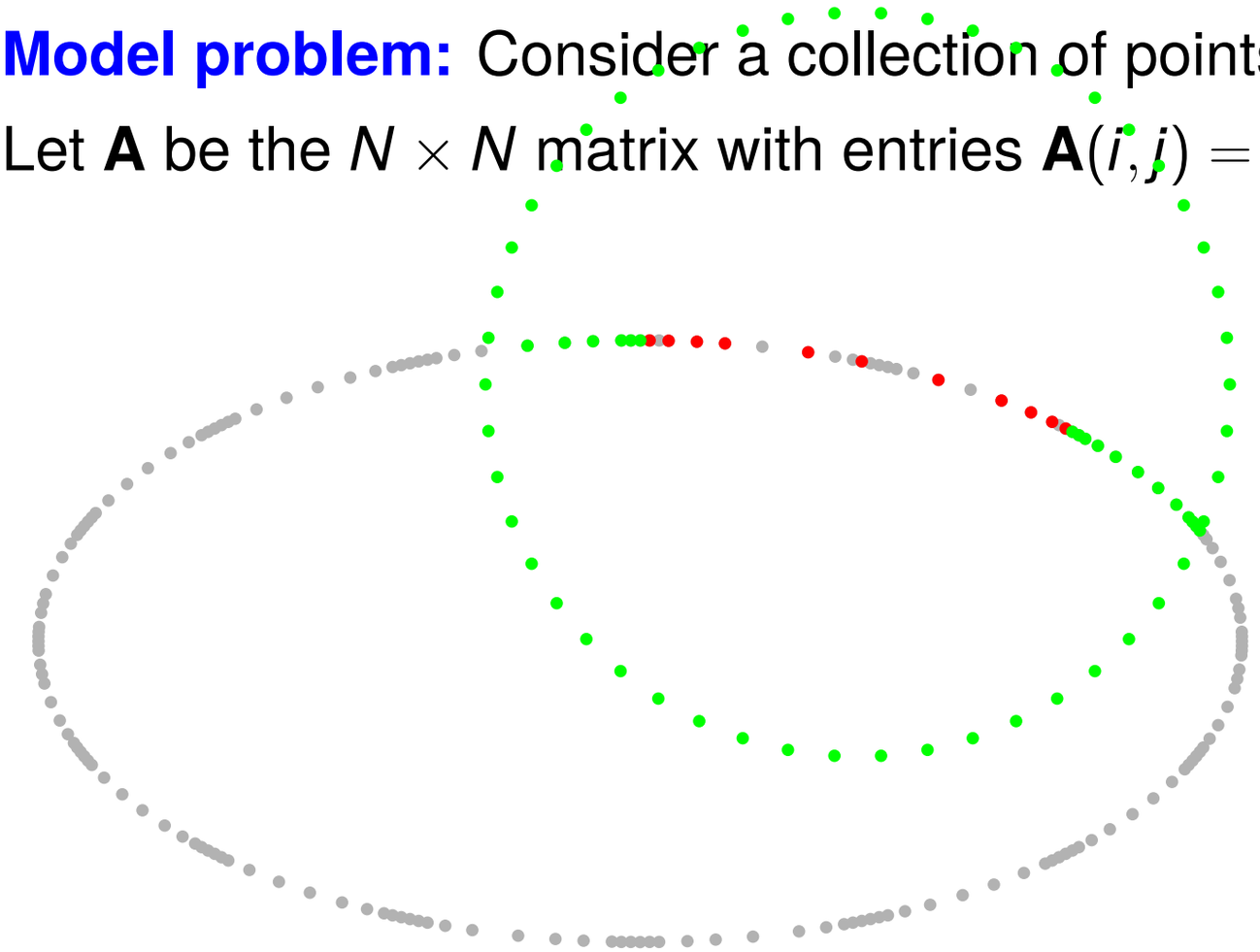


The contour

Replace far-field nodes by a small set of proxy charges.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.

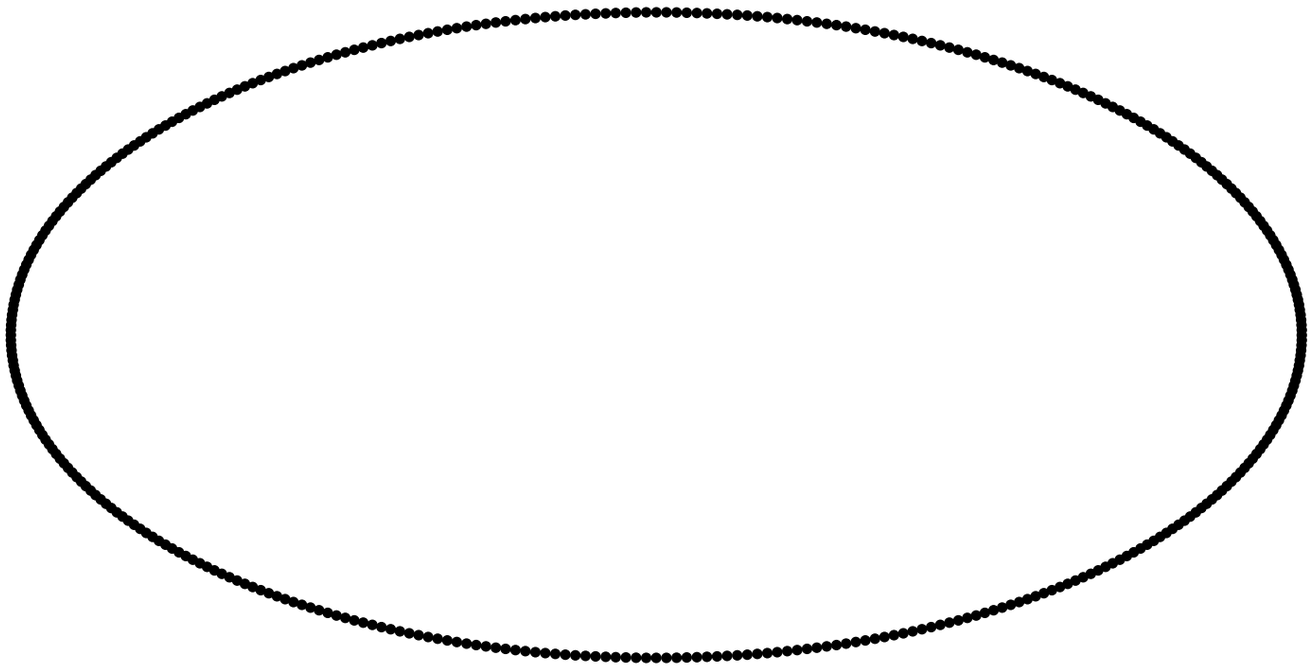


The contour

Points remaining after compression.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

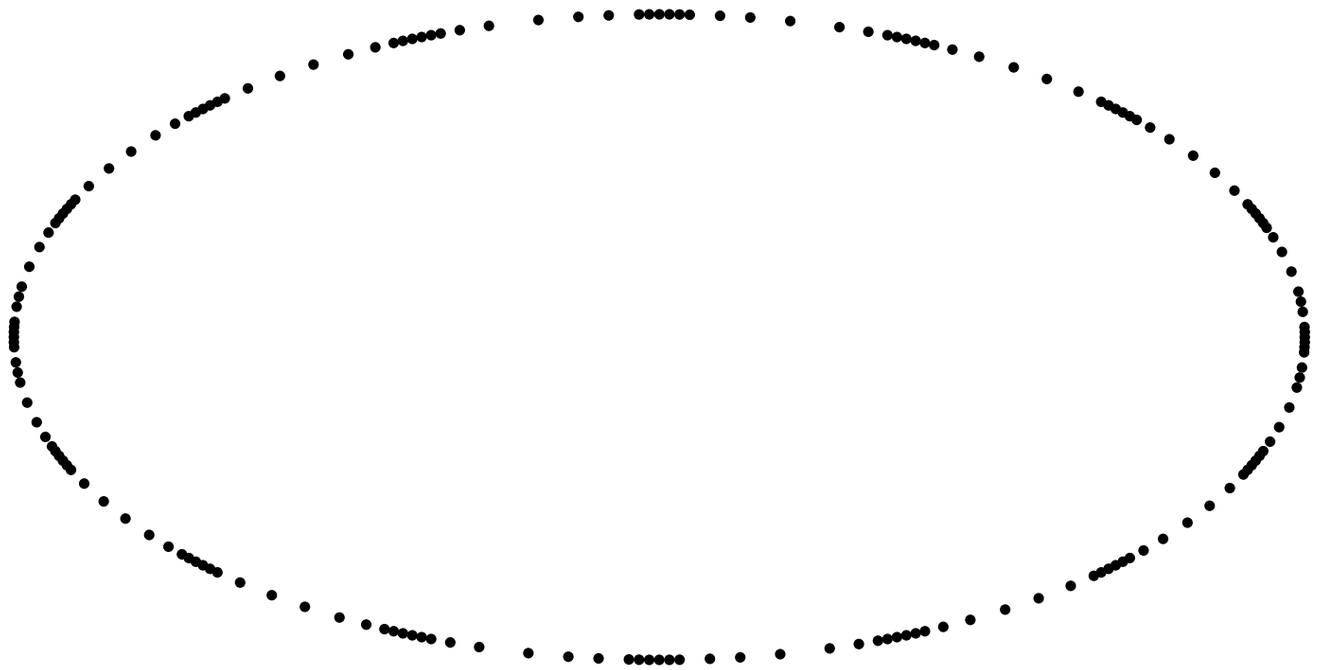
Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



All points

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.



After level 4 compression.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.

After level 3 compression.

Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.

After level 2 compression.

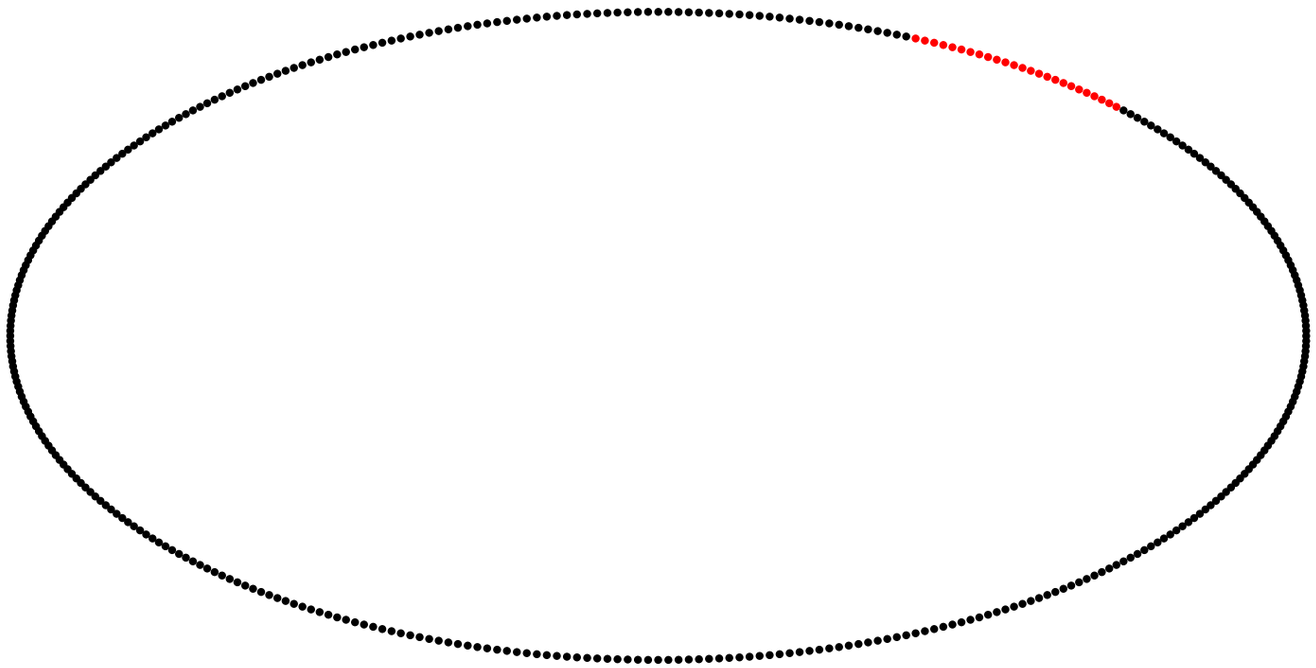
Model problem: Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ matrix with entries $\mathbf{A}(i,j) = \log|\boldsymbol{x}_i - \boldsymbol{x}_j|$ for $i \neq j$.

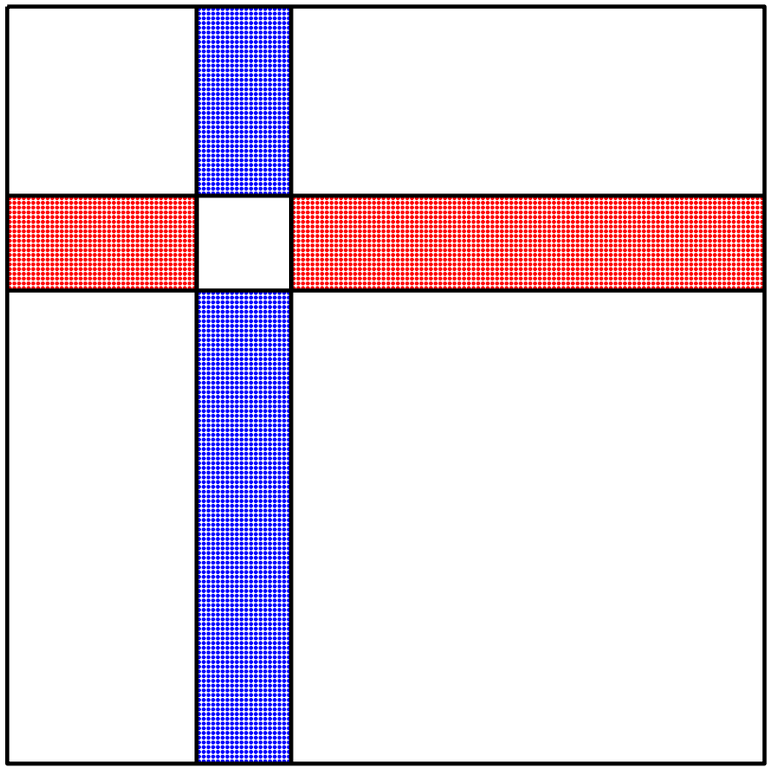
After level 1 compression.

Model problem (non-sym): Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ *non-symmetric* matrix with entries $\mathbf{A}(i,j) = \dfrac{(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot \boldsymbol{n}_j}{|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}$ for $i \neq j$.



The contour



$\mathbf{A}(I_\tau, I_\tau^c)$ *in red.*
$\mathbf{A}(I_\tau^c, I_\tau)$ *in blue.*

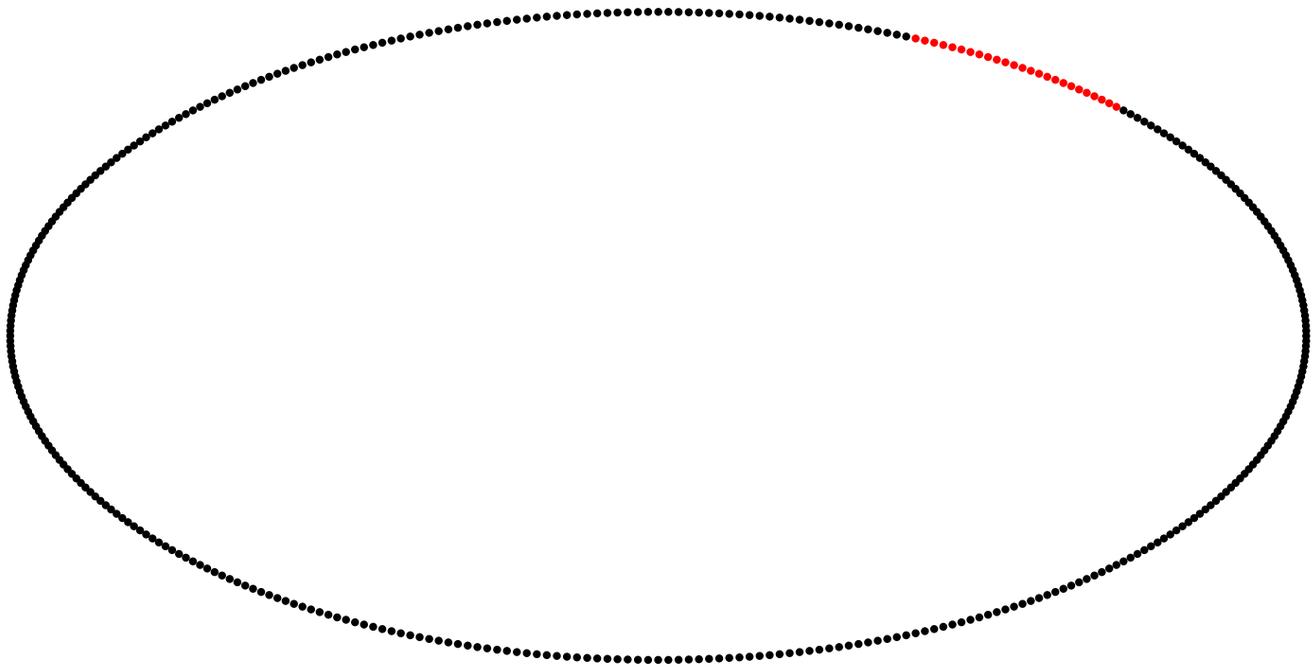
Let Γ_τ be a panel associated with an index vector I_τ.

Our task is to determine basis matrices $\mathbf{U}_\tau$, $\mathbf{V}_\tau$ and index vectors $\tilde{I}_\tau$, $\hat{I}_\tau$, s.t.

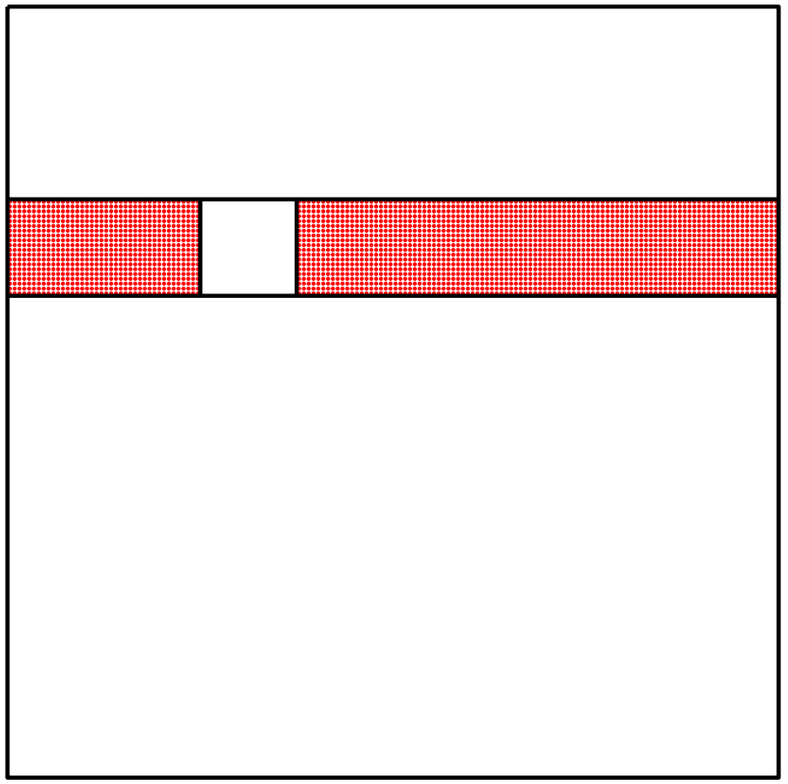
$$\underset{n \times (N-n)}{\mathbf{A}(I_\tau, I_\tau^c)} = \underset{n \times k}{\mathbf{U}_\tau} \; \underset{k \times (N-n)}{\mathbf{A}(\tilde{I}_\tau, I_\tau^c)} \quad \text{and} \quad \underset{(N-n) \times n}{\mathbf{A}(I_\tau^c, I_\tau)} = \underset{(N-n) \times k}{\mathbf{A}(I_\tau^c, \hat{I}_\tau)} \; \underset{k \times n}{\mathbf{V}_\tau^*}$$

Model problem (non-sym): Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ *non-symmetric* matrix with entries $\mathbf{A}(i,j) = \dfrac{(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot \boldsymbol{n}_j}{|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}$ for $i \neq j$.



The contour



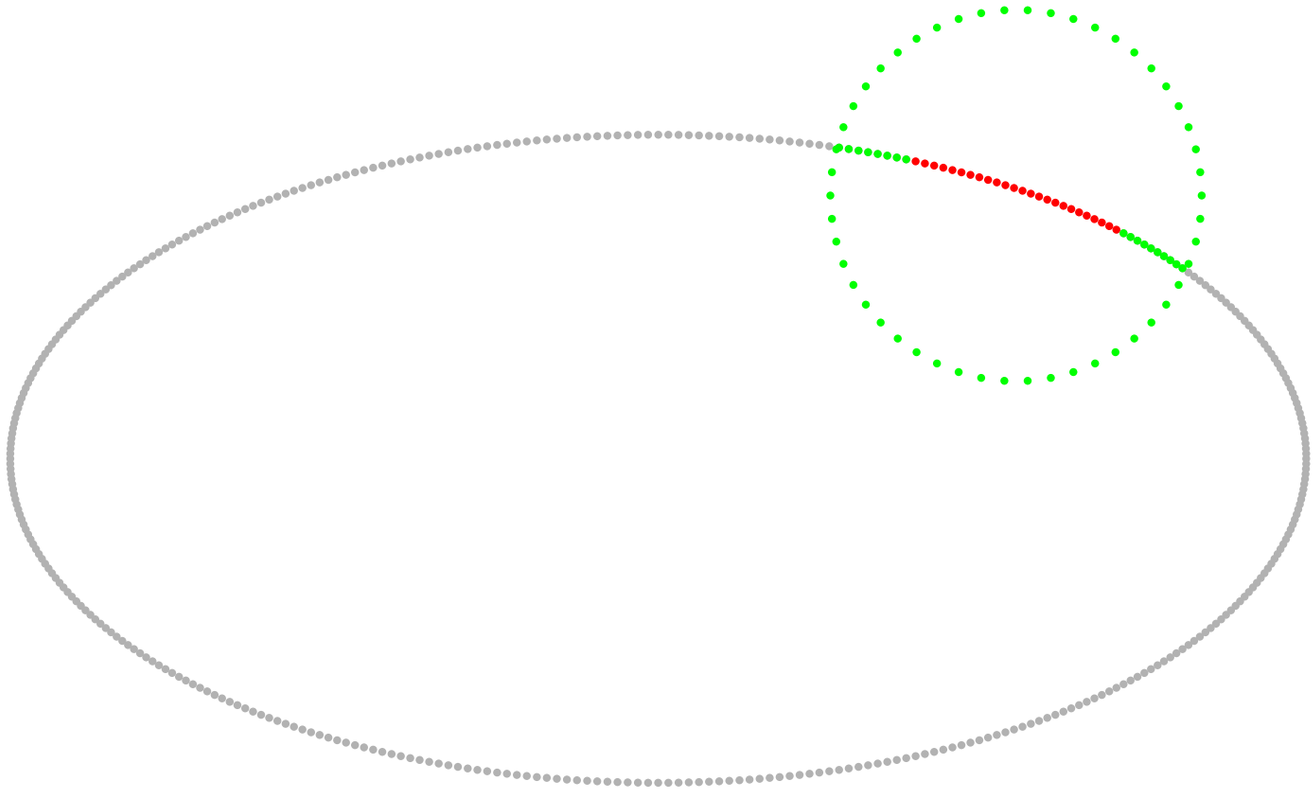
$\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$ *in red.*

Let us first consider the task of finding $\mathbf{U}_\tau$. We need to factor

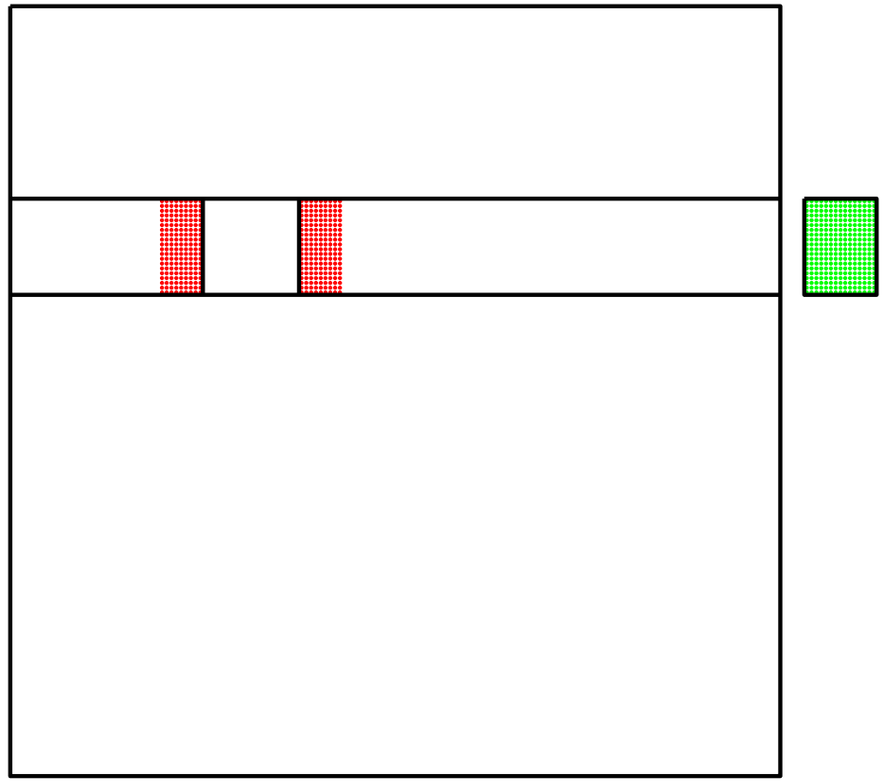
$$\begin{array}{ccccc} \mathbf{A}(I_\tau, I_\tau^{\mathrm{c}}) & = & \mathbf{U}_\tau & & \mathbf{A}(\tilde{I}_\tau, I_\tau^{\mathrm{c}}) \\ n \times (N-n) & & n \times k & & k \times (N-n) \end{array}$$

Model problem (non-sym): Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ *non-symmetric* matrix with entries $\mathbf{A}(i,j) = \dfrac{(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot \boldsymbol{n}_j}{|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}$ for $i \neq j$.



The contour



The block $\mathbf{A}(I_\tau, I_\tau^{(\text{near})})$ *shown in red.*
The block $\mathbf{G}$ *shown in green.*

Everything works the same!

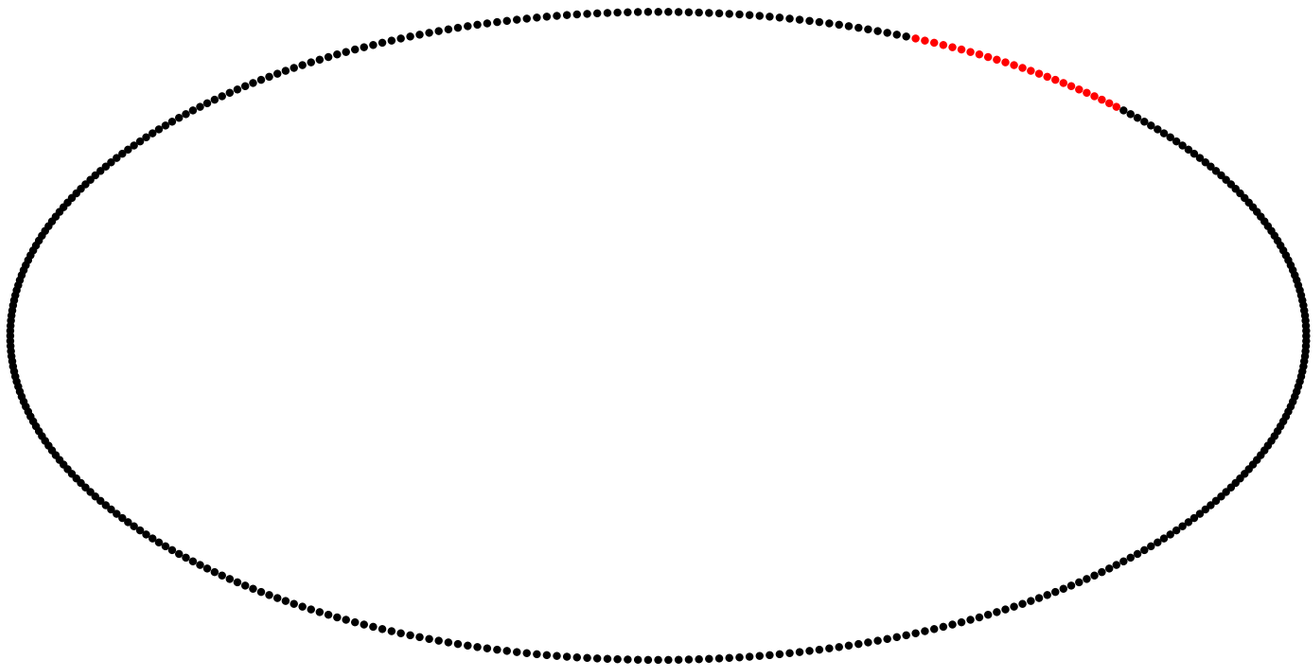
Replace charges in the far-field by "proxy" charges, let $I_\tau^{(\text{near})}$ denote the near-field points and let $\mathbf{G}$ denote a matrix of size $n \times n_{\text{proxy}}$ that maps *monopole* charges on the proxy locations to potentials on Γ_τ. Then factor the smaller matrix $\mathbf{B} = [\mathbf{A}(I_\tau, I_\tau^{(\text{near})}), \mathbf{G}]$:

$$\underset{n \times (n_{\text{near}} + n_{\text{proxy}})}{\mathbf{B}} = \underset{n \times k}{\mathbf{U}_\tau} \; \underset{k \times (n_{\text{near}} + n_{\text{proxy}})}{\mathbf{B}(J,:)}$$

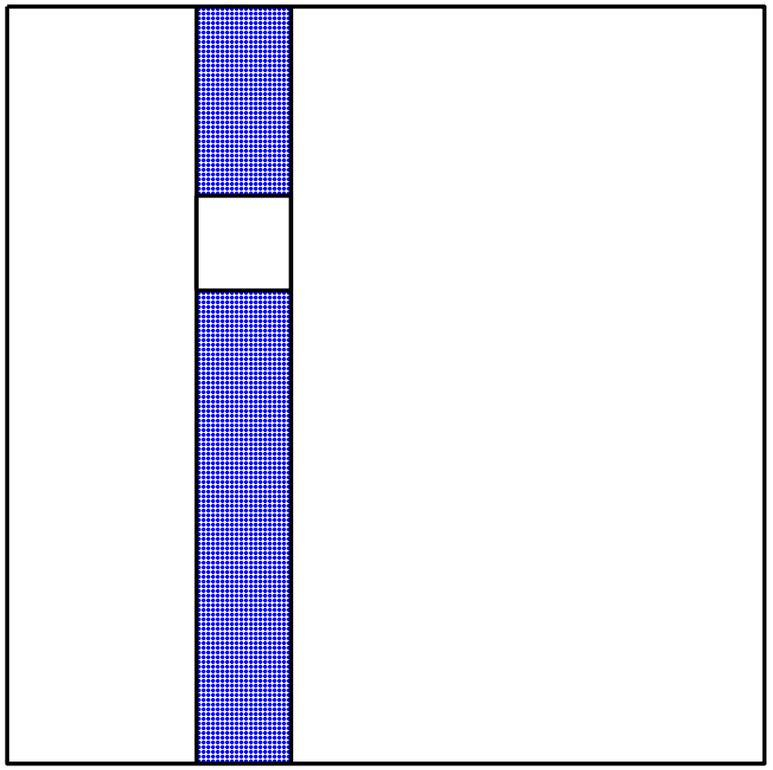
and set $\tilde{I}_\tau = I_\tau(J)$.

Model problem (non-sym): Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ *non-symmetric* matrix with entries $\mathbf{A}(i,j) = \dfrac{(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot \boldsymbol{n}_j}{|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}$ for $i \neq j$.



The contour



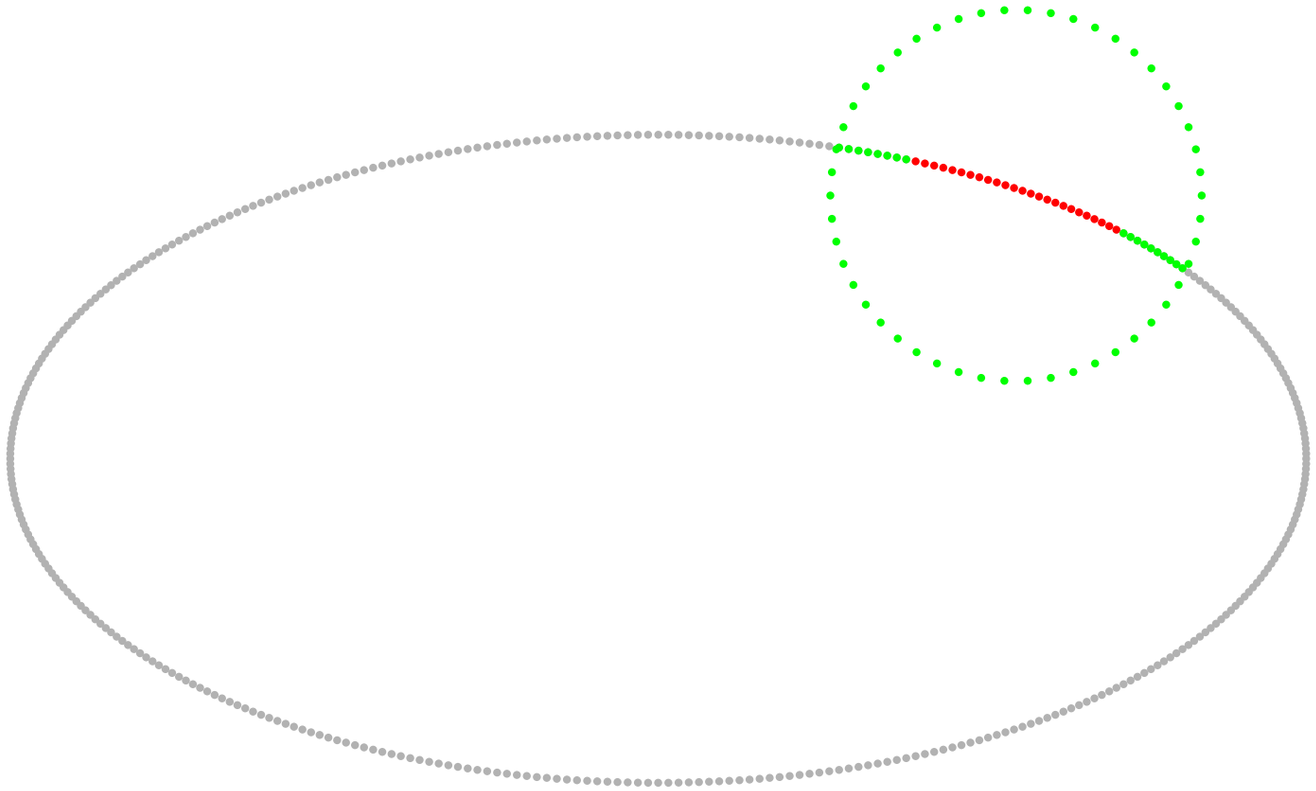
$\mathbf{A}(I_\tau^{\mathrm{c}}, I_\tau)$ *in blue.*

Next we consider the task of finding $\mathbf{V}_\tau$. We need to factor

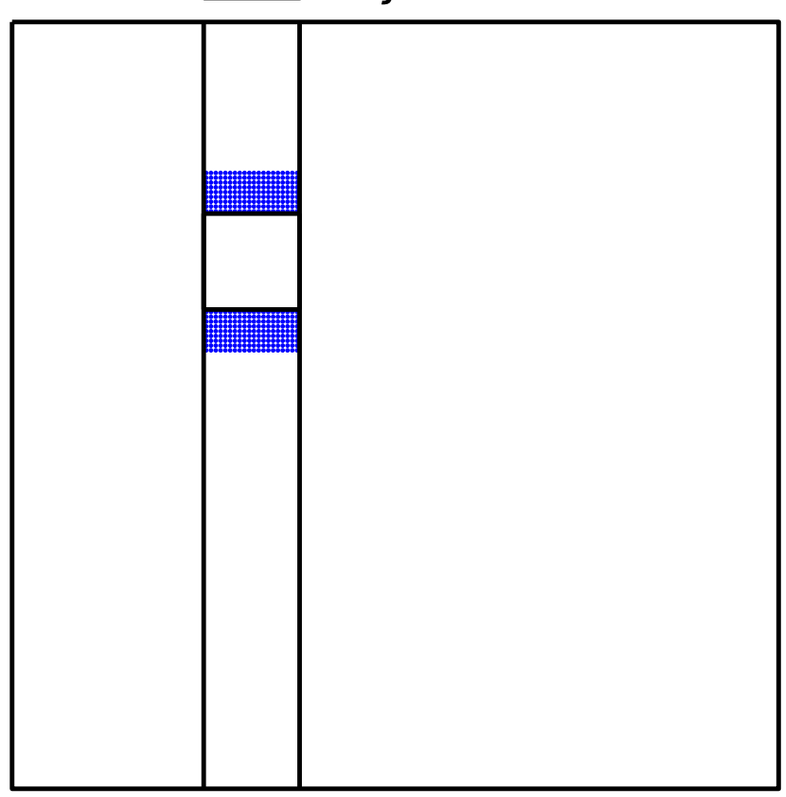
$$\begin{array}{ccccc} \mathbf{A}(I_\tau^{\mathrm{c}}, I_\tau) & = & \mathbf{A}(I_\tau^{\mathrm{c}}, \hat{I}_\tau) & \mathbf{V}_\tau^* \\ (N-n) \times n & & (N-n) \times k & k \times n \end{array}$$

Model problem (non-sym): Consider a collection of points $\{\boldsymbol{x}_i\}_{i=1}^N$ along a contour Γ.

Let $\mathbf{A}$ be the $N \times N$ *non-symmetric* matrix with entries $\mathbf{A}(i,j) = \dfrac{(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot \boldsymbol{n}_j}{|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}$ for $i \neq j$.



The contour



$\mathbf{A}(I_\tau^{\mathrm{c}}, I_\tau)$ *in blue.*
$\mathbf{G}$ *in magenta.*

Things work *almost* the same . . .

Replace charges in the far-field by "proxy" charges, let $I_\tau^{(\mathrm{near})}$ denote the near-field points and let $\mathbf{G}$ denote a matrix of size $n \times n_{\mathrm{proxy}}$ that maps *dipole* charges on Γ_τ to potentials on the proxy points. Then factor the smaller problem:

$$\begin{bmatrix} \mathbf{A}(I_\tau^{(\mathrm{near})}, I_\tau) \\ \mathbf{G} \end{bmatrix} = \begin{bmatrix} \mathbf{A}(I_\tau^{(\mathrm{near})}, \hat{I}_\tau) \\ \mathbf{G}(:, J) \end{bmatrix} \mathbf{V}_\tau^*.$$

Notes:

- There are in fact two potentially different sets of skeleton points:
  1. The *incoming skeleton points* resulting from an ID of the *rows* of $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.
  2. The *outgoing skeleton points* resulting from an ID of the *columns* of $\mathbf{A}(I_\tau^{\mathrm{c}}, I_\tau)$.

  It is possible, and often practical, to enforce that these skeletons be the same. This can be done by constructing an ID for the rows of $[\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}}),\ \mathbf{A}(I_\tau^{\mathrm{c}}, I_\tau)^*]$.

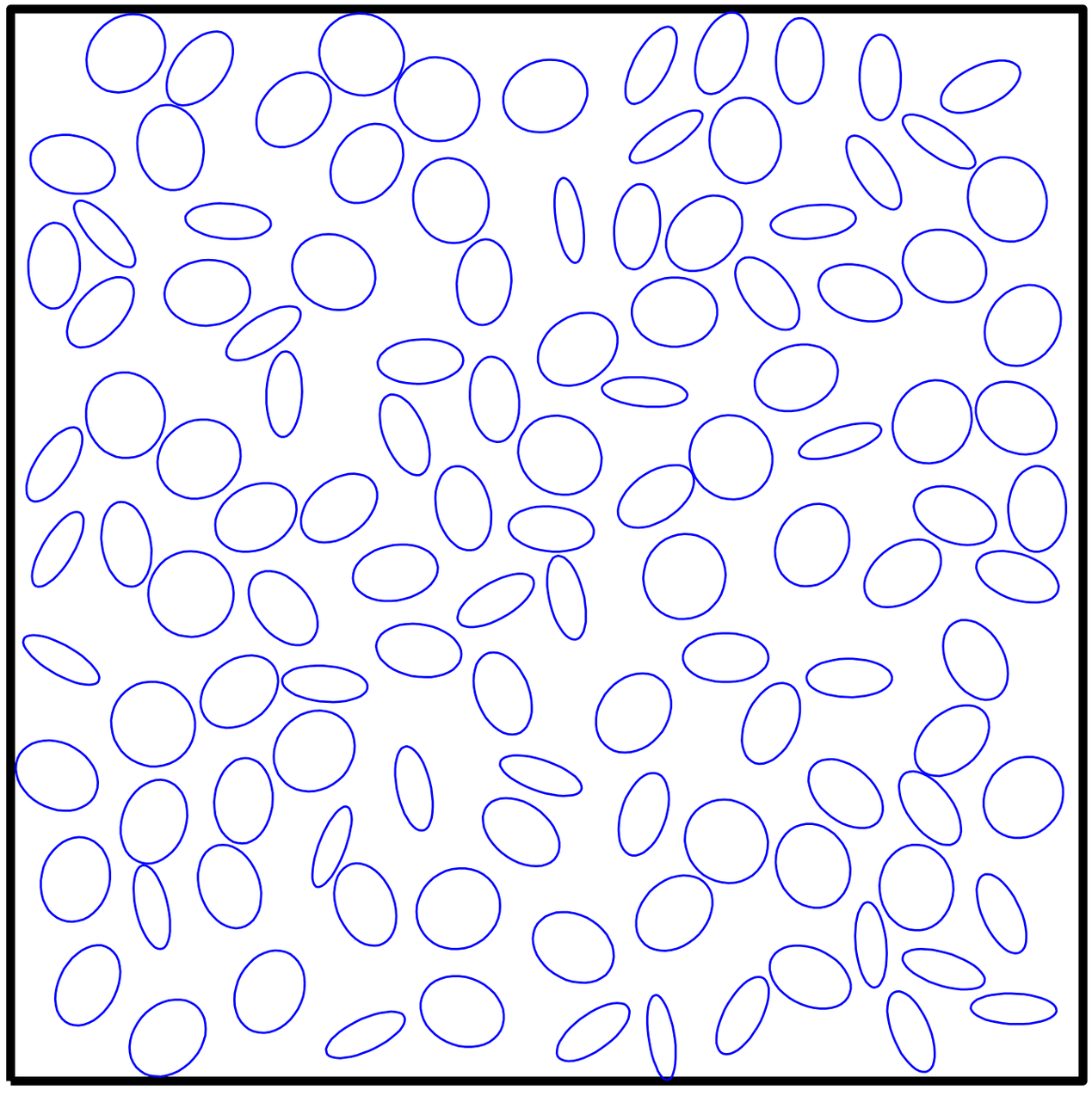
- In real life, the presence of quadrature corrections for "near-diagonal" elements slightly complicates matters. However, these complications can all be handled.

- For *Helmholtz*, the compression technique based on a proxy domain (e.g. circle) to account for the far-field has to be modifed to avoid the possibility of resonances (avoid using resonant radii, or, use *two* concentric sets of proxy circles separated by a distance $\lambda/4$, or, use both monopoles and dipoles on the proxy surface, etc).

- For other elliptic PDEs (Stokes, elasticity, time-harmonic Maxwell, etc), analogous representations can be worked out. Each case has its own subtleties, but the basic ideas carry over. (At least, it currently appears that they do!)

- Some care is necessary in determining how finely to sample the proxy surface, in particular for Helmholtz.

A "volume filling" domain: Now consider a contour like this:

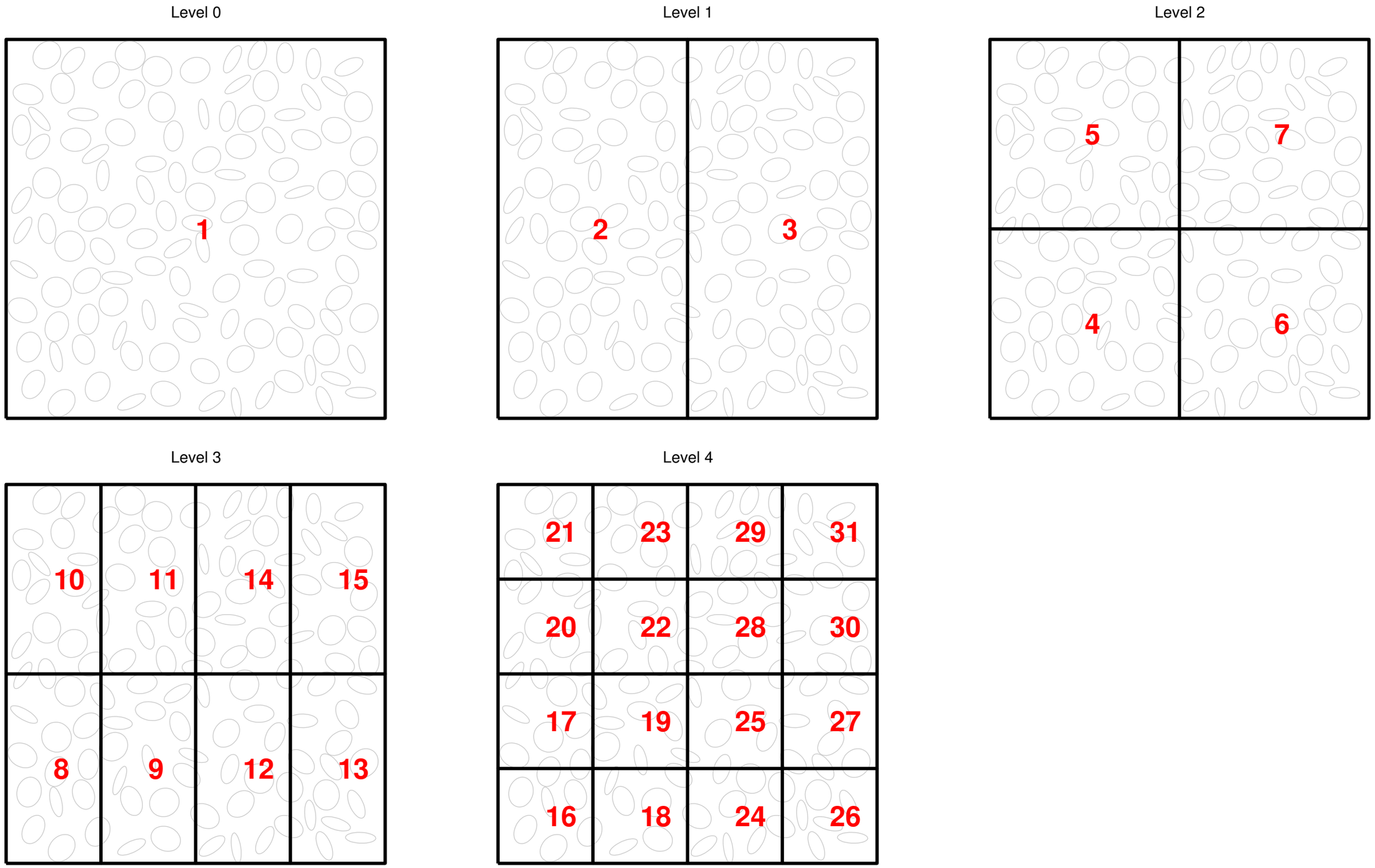


Let **A** denote an $N \times N$ matrix arising upon discretizing a boundary integral operator

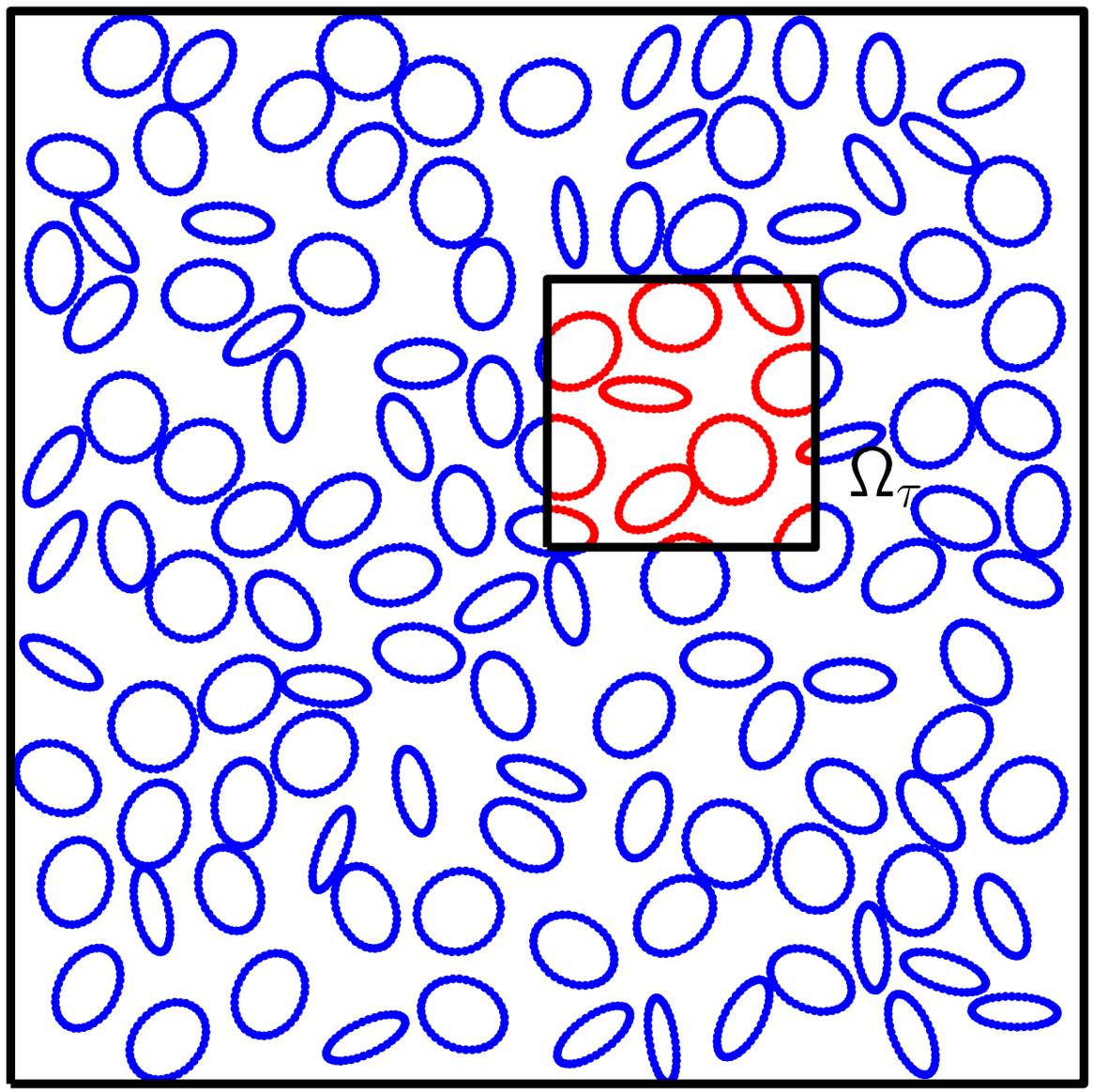
$$[Aq](\boldsymbol{x}) = q(\boldsymbol{x}) + \int_\Gamma \log|\boldsymbol{x} - \boldsymbol{y}|\, q(\boldsymbol{y})\, dA(\boldsymbol{y}), \qquad \boldsymbol{x} \in \Gamma,$$

where Γ is the collection of ellipses shown.

We must now use a binary tree based on *splitting in physical space* (as opposed to parameter space).

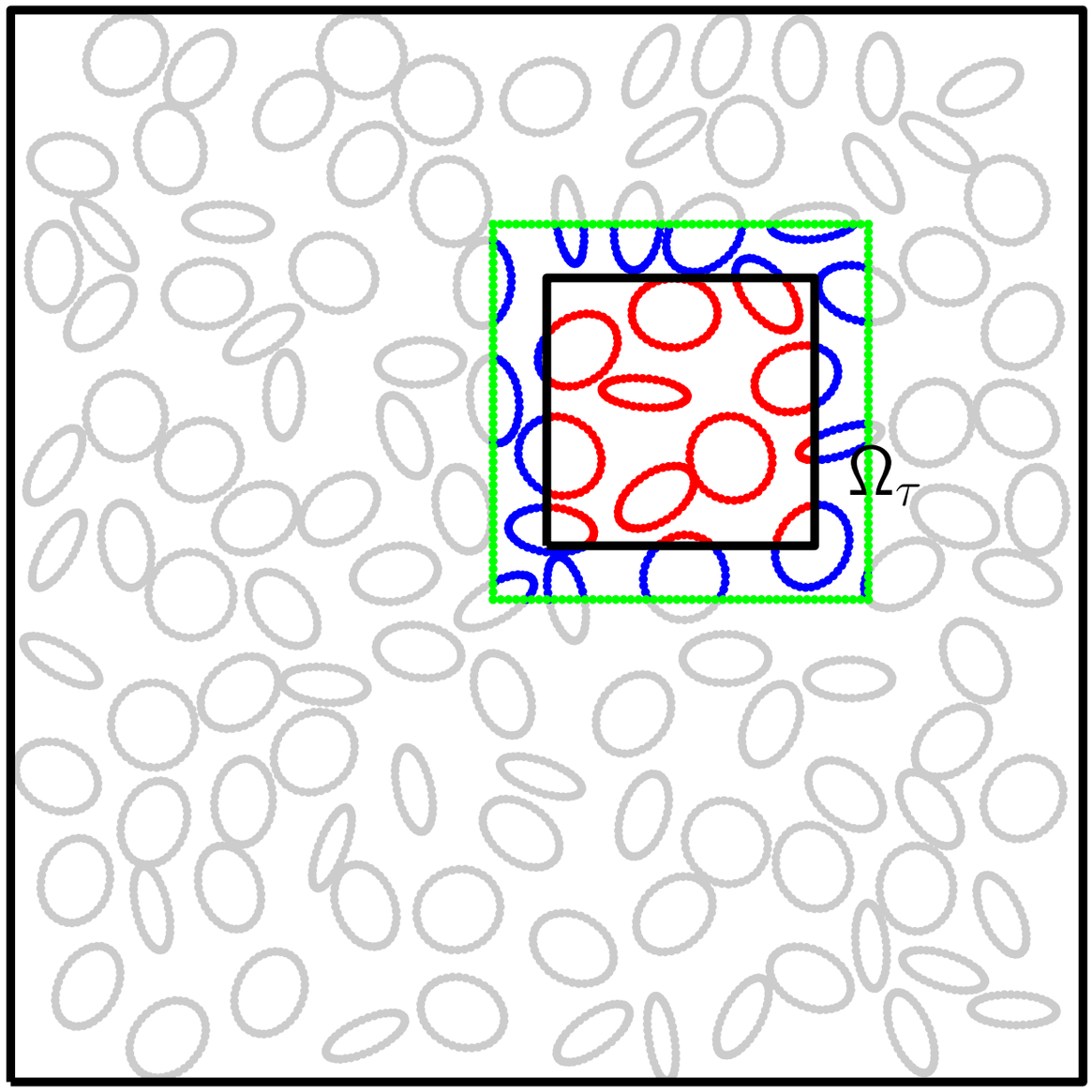


Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle the same as before, but the proxy surfaces are chosen a bit differently.



At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.

Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle the same as before, but the proxy surfaces are chosen a bit differently.

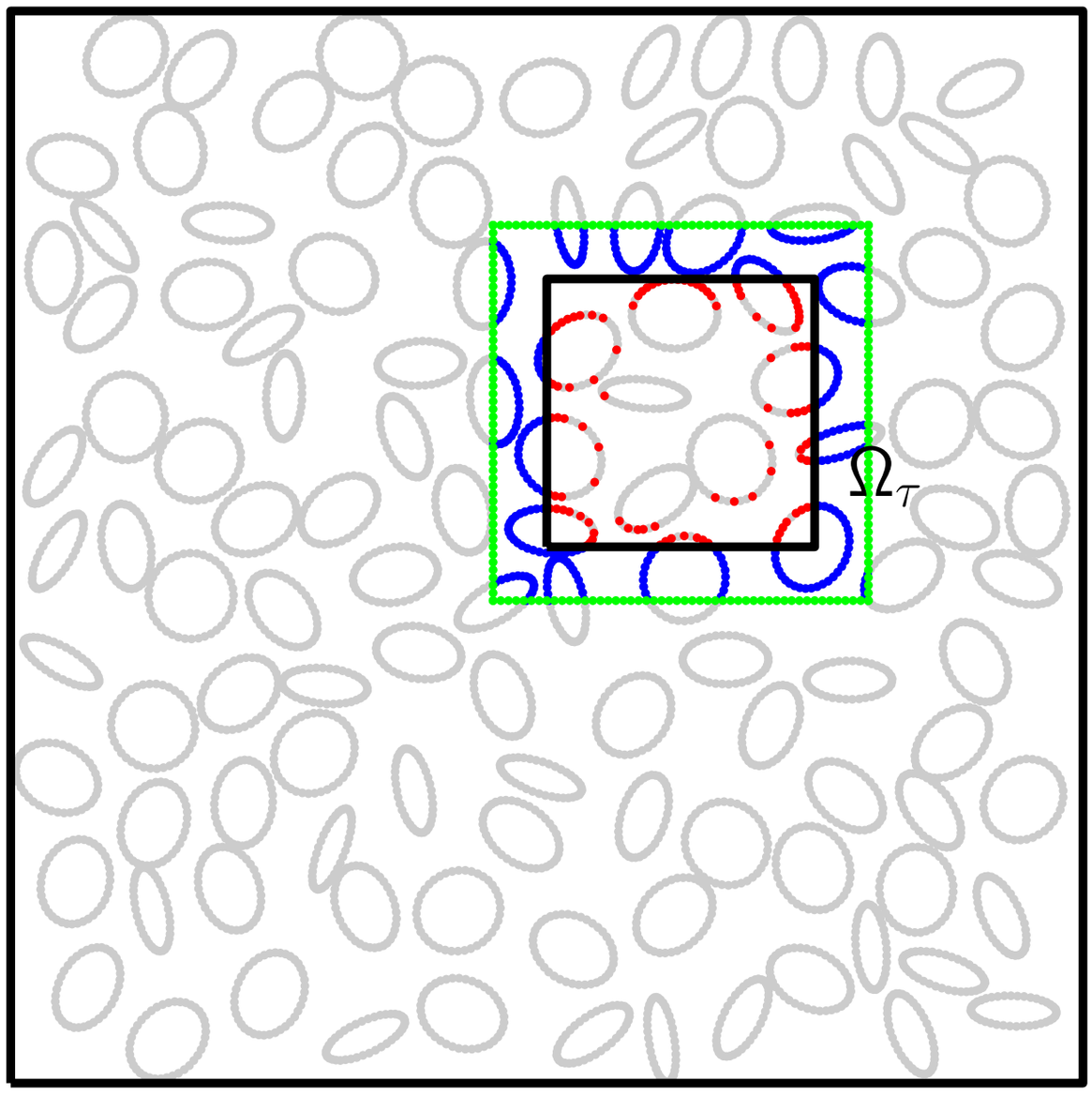


- Points in I_τ.
- Points in $I_\tau^{(\mathrm{near})}$.
- Points in Γ_{proxy}.

(gray points are inactive)

At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.
But, using the *Green localization trick,* we only need to ID the matrix $[\mathbf{A}(I_\tau, I_\tau^{(\mathrm{near})})\ \mathbf{G}]$, where $\mathbf{G}$ is the matrix of interaction with the proxy surface (green).

Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle the same as before, but the proxy surfaces are chosen a bit differently.

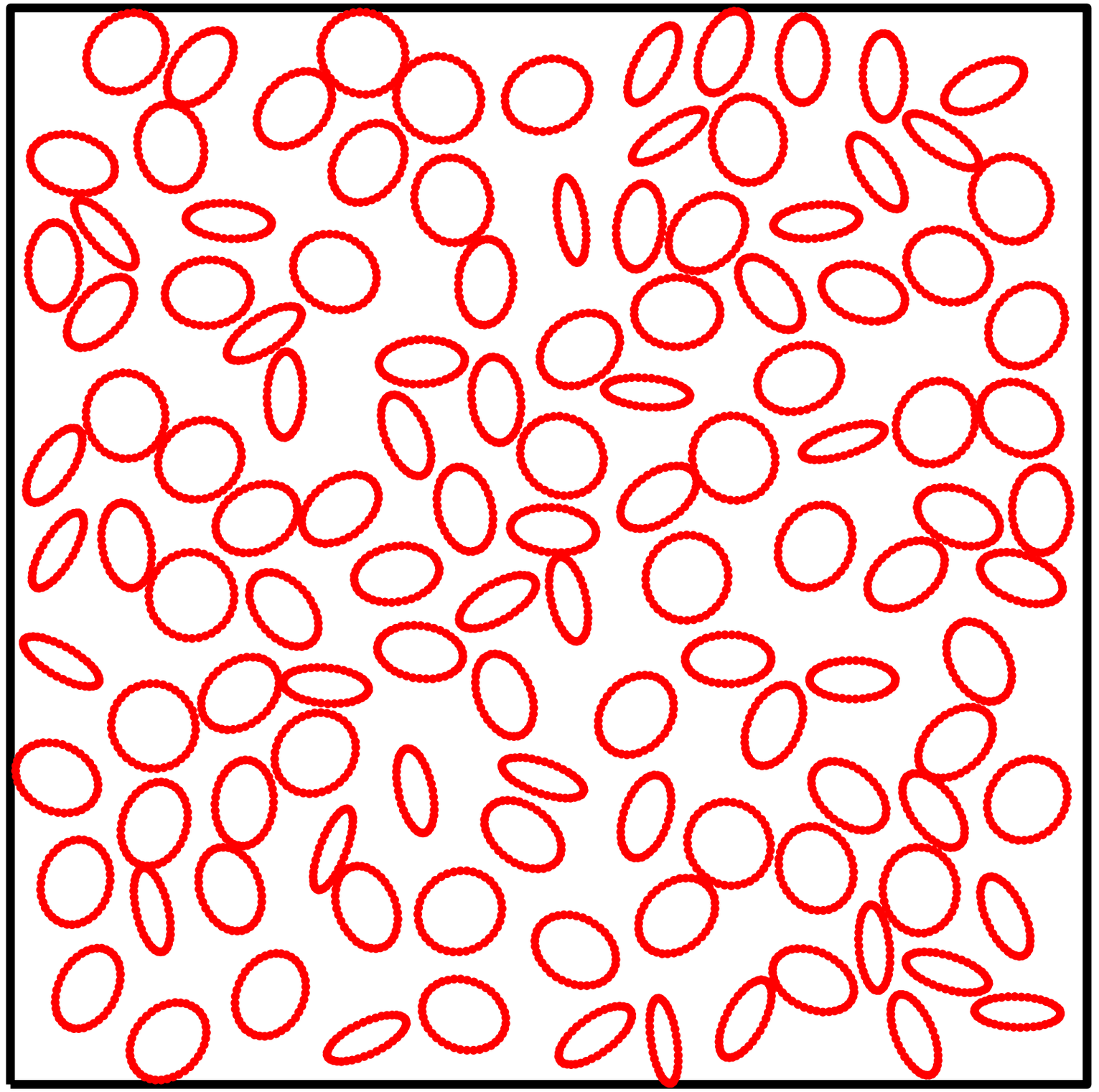


- Points in $\tilde{I}_\tau$.
- Points in $I_\tau^{(\mathrm{near})}$.
- Points in Γ_{proxy}.

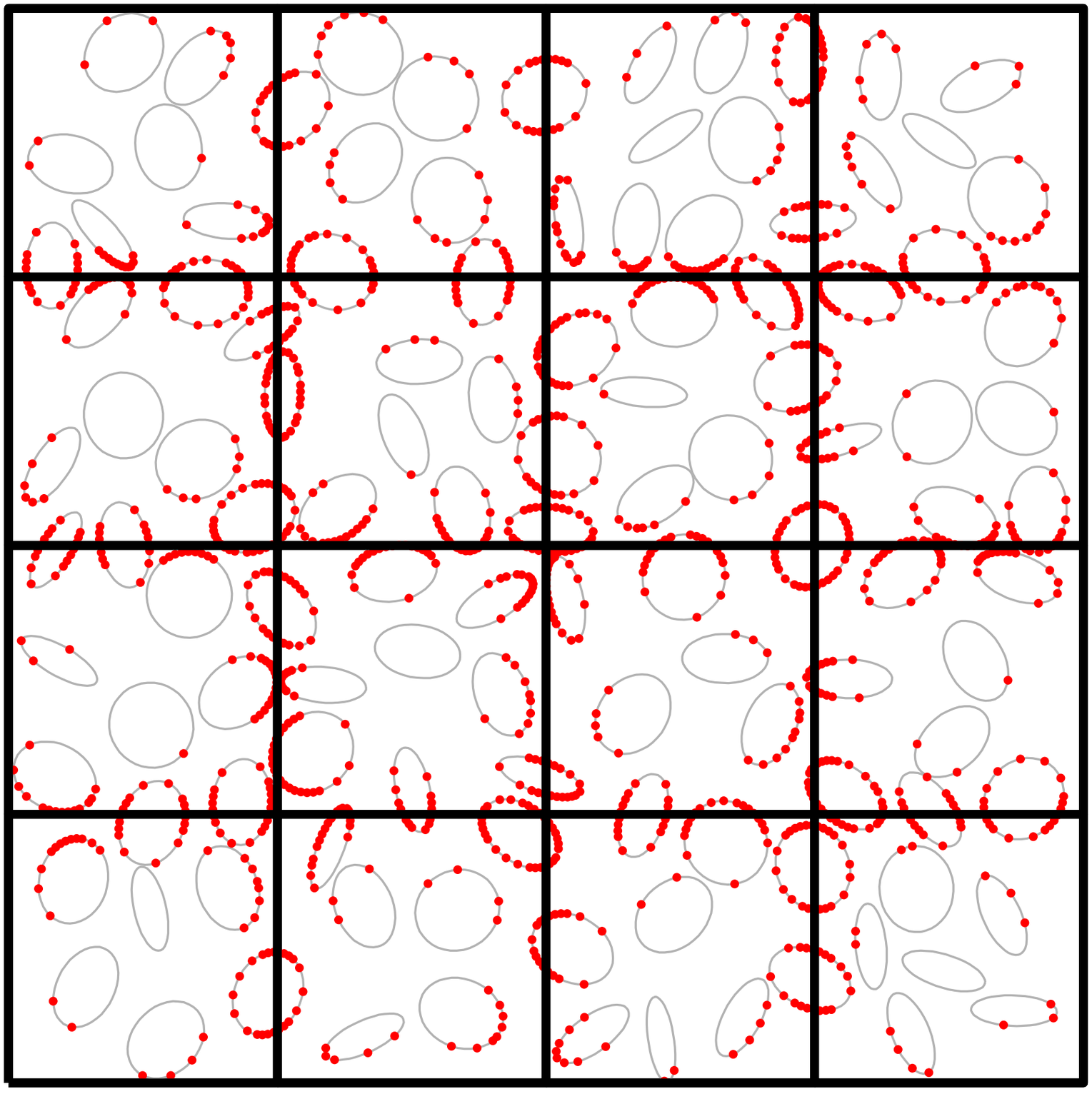
(gray points are inactive)

At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.
But, using the *Green localization trick,* we only need to ID the matrix $[\mathbf{A}(I_\tau, I_\tau^{(\mathrm{near})})\ \mathbf{G}]$,
where **G** is the matrix of interaction with the proxy surface (green).

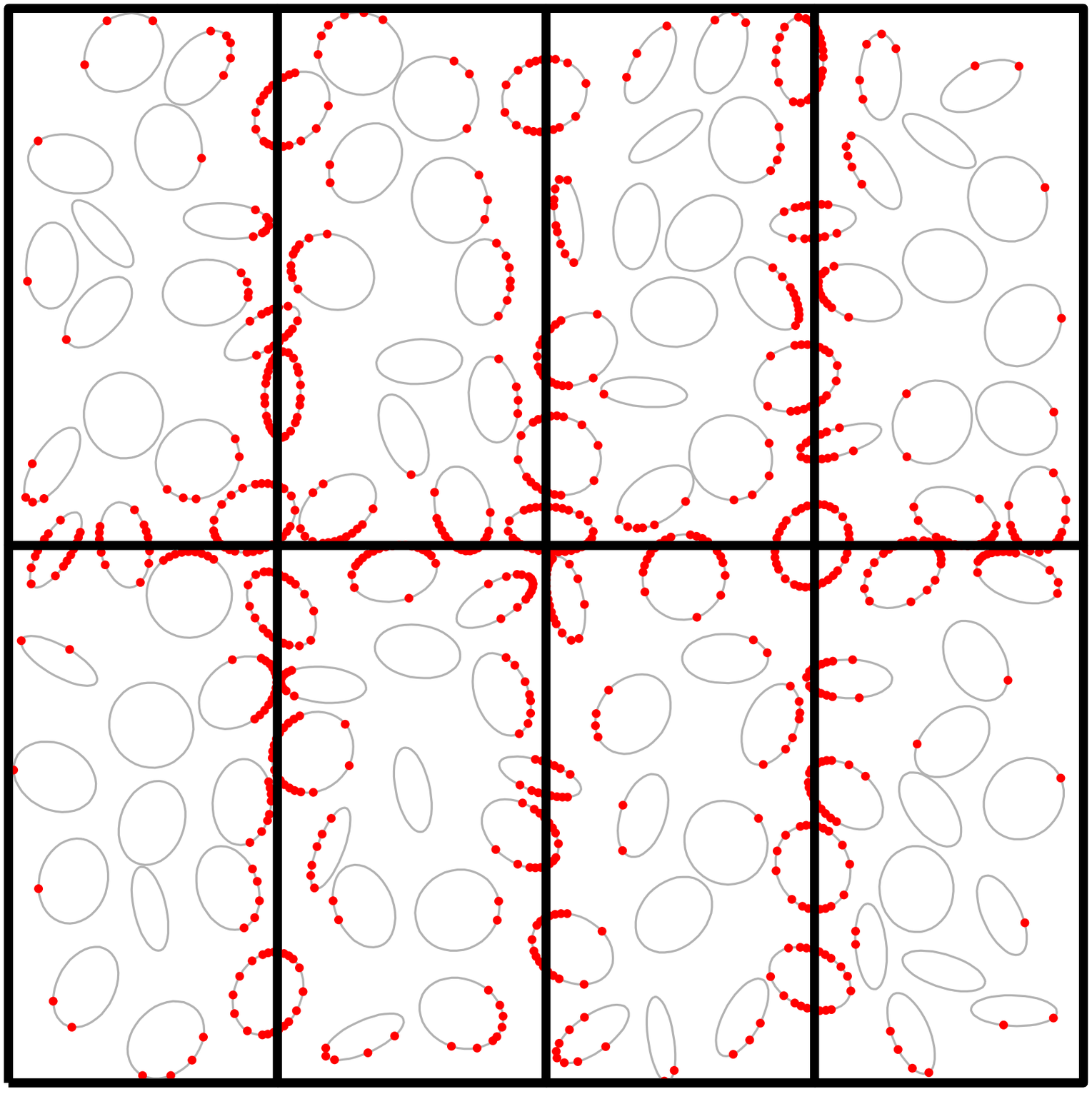
Original set of points



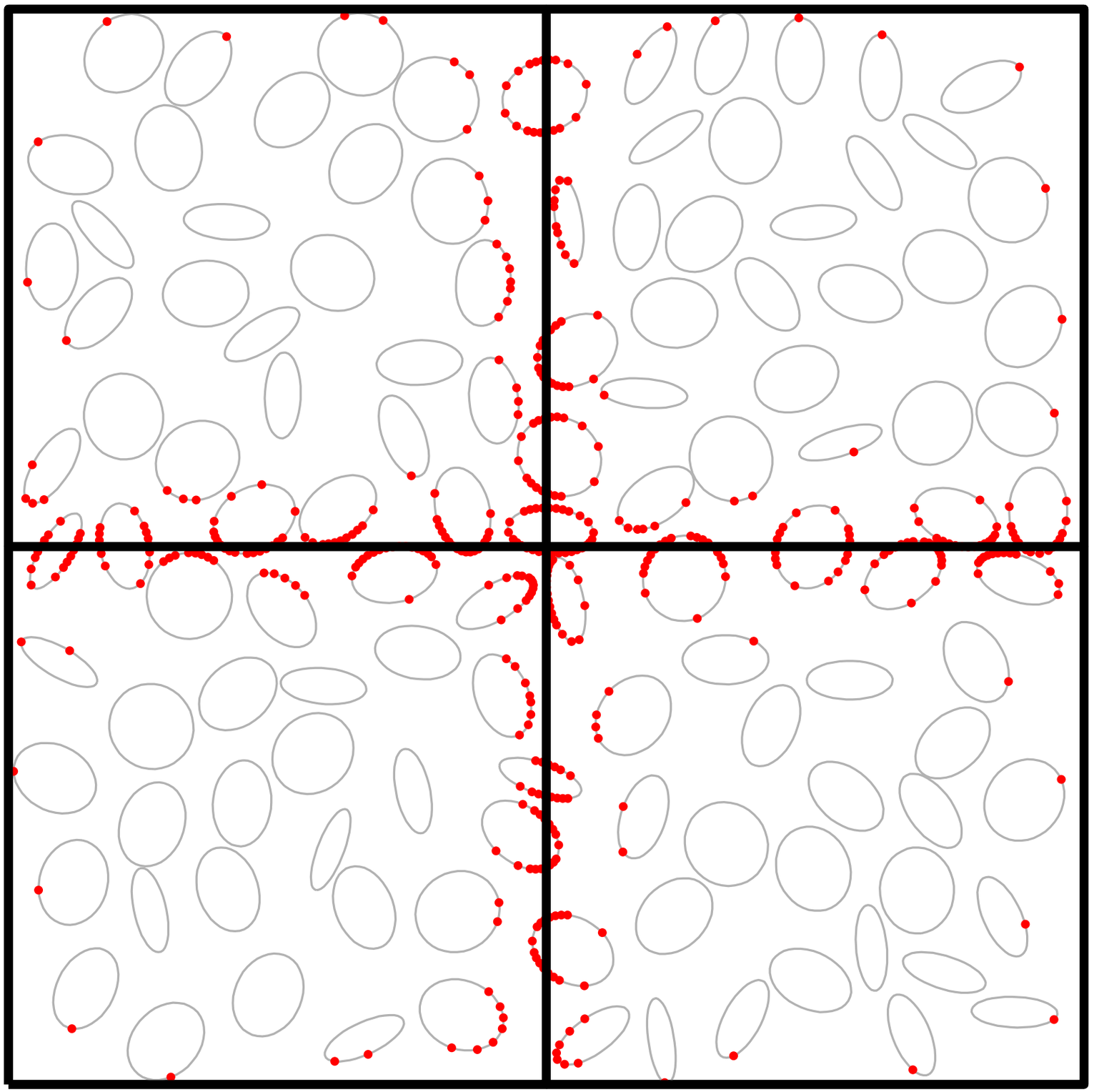
Skeleton points on level 4, acc = 1.000e−09



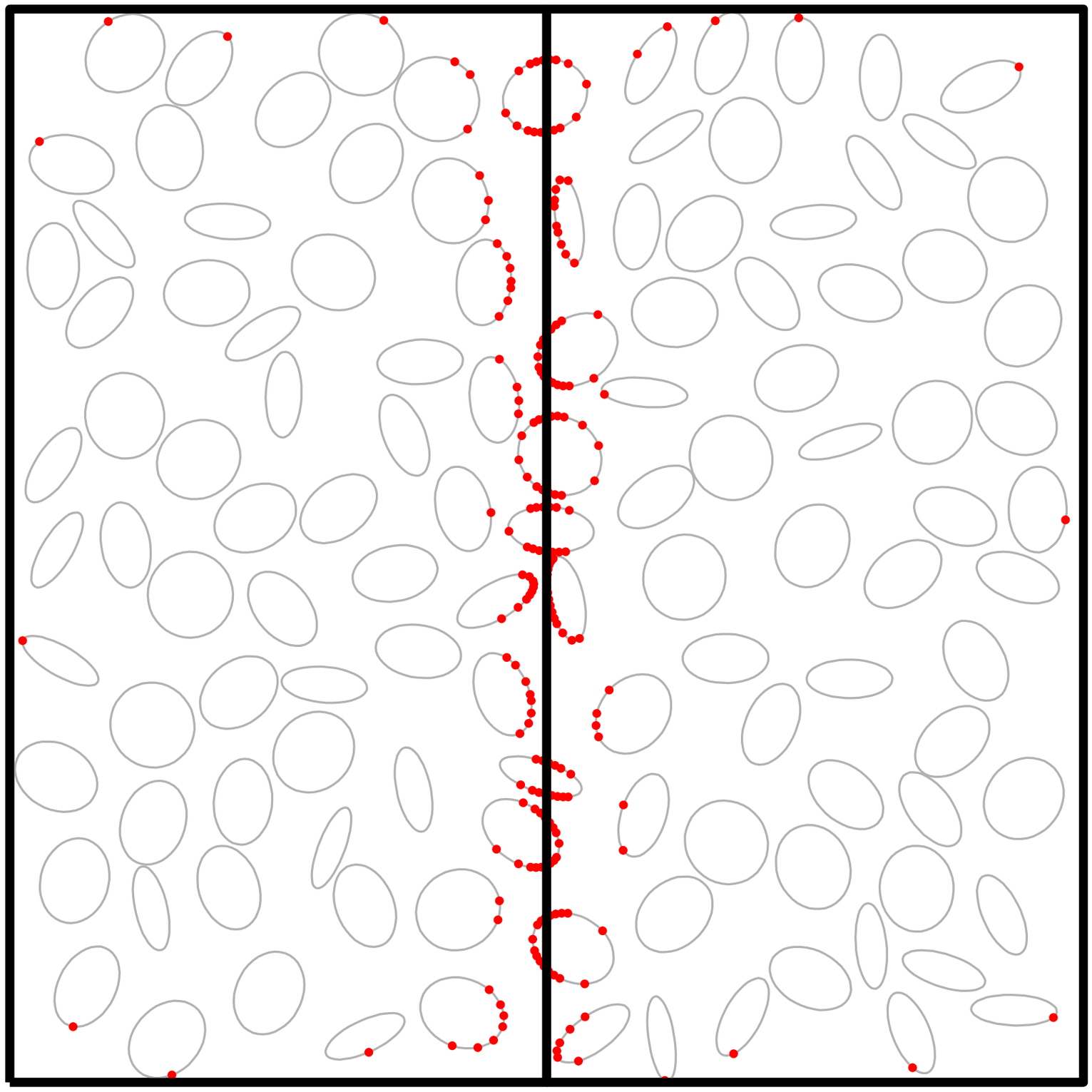
Skeleton points on level 3, acc = 1.000e−09



Skeleton points on level 2, acc = 1.000e−09



Skeleton points on level 1, acc = 1.000e−09



Good news: The direct solver based on HBS matrix algebra works with only minor modifications.

Good news: The direct solver based on HBS matrix algebra works with only minor modifications.

Bad news: The simple direct solver no longer has $O(N)$ complexity.

Good news: The direct solver based on HBS matrix algebra works with only minor modifications.

Bad news: The simple direct solver no longer has $O(N)$ complexity.

Complexity analysis: For a box τ, define quantities:

N_τ	Number of discretization points in τ.
n	Number of points in the skeletons for the children of τ.
g	Number of points in the proxy contour.
k	Rank of interaction between τ and the outside world.

Then

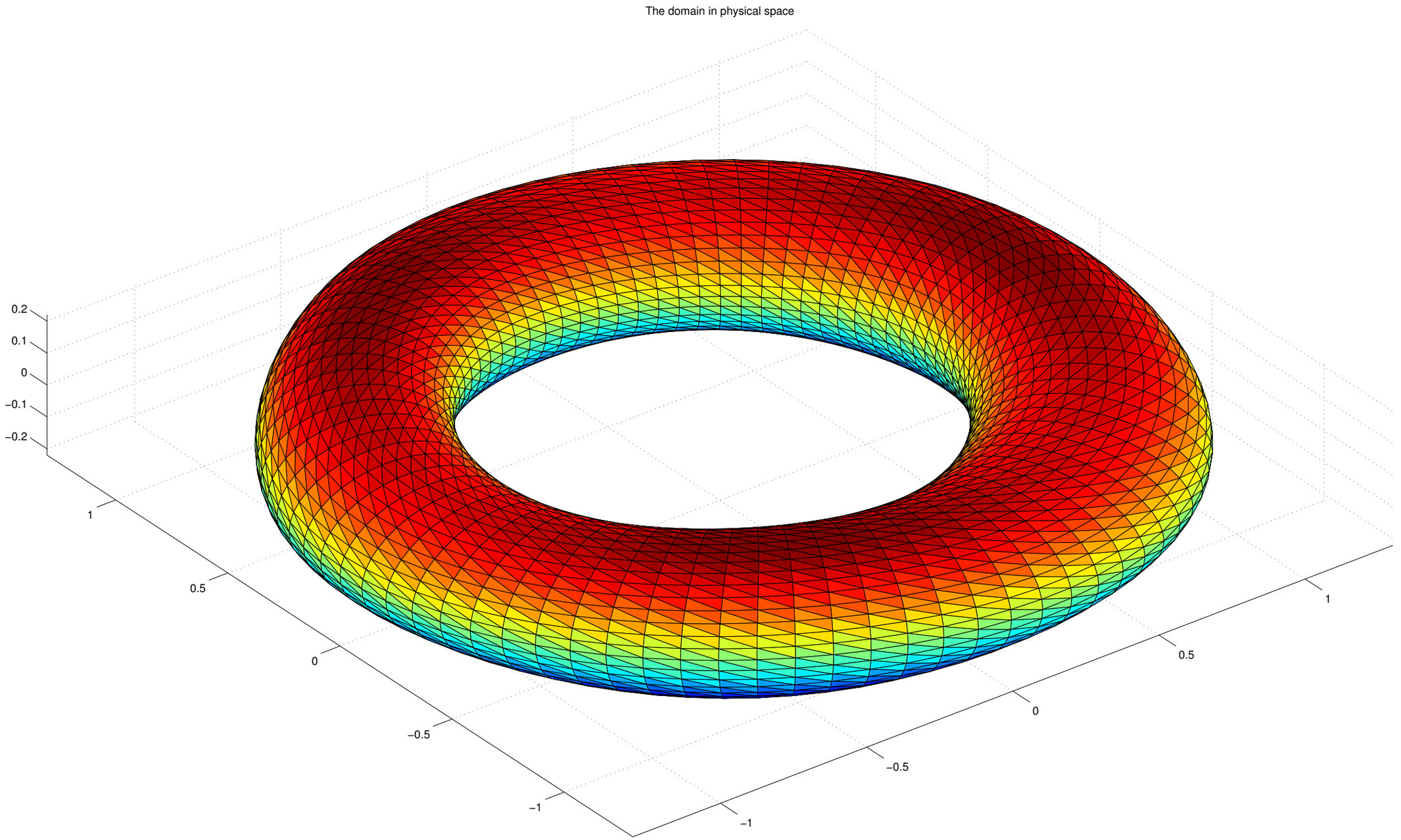
$$\begin{aligned} \text{Cost of compressing } \tau &\sim n\,g\,k \\ \text{Cost of building local operators } \tau &\sim n^3 \end{aligned}$$

Unfortunately, for a "volume filling" set of points, we have

$$n \sim \sqrt{N_\tau}, \qquad g \sim \sqrt{N_\tau}, \qquad k \sim \sqrt{N_\tau},$$

so the overall cost of the direct solver is $O(N^{3/2})$.

A surface in 3D: Now consider a surface in $\mathbb{R}^3$:

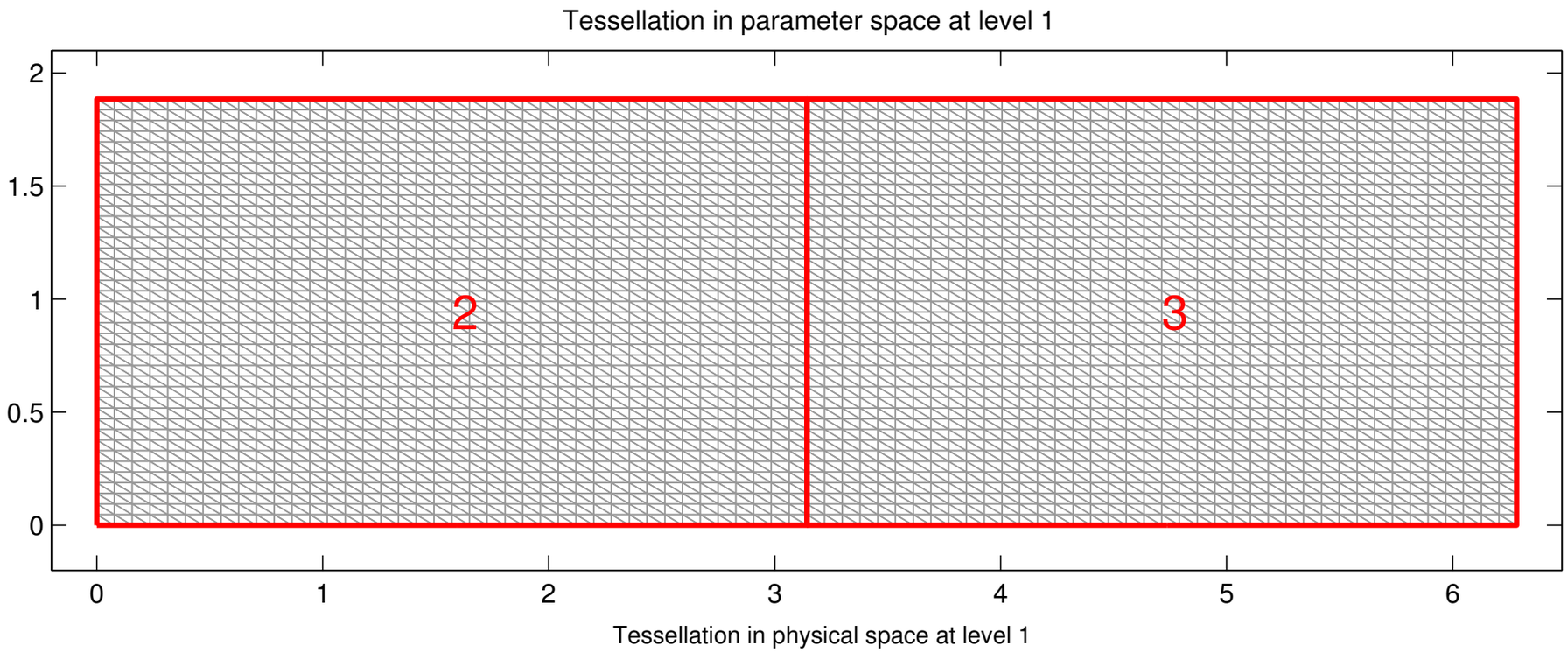


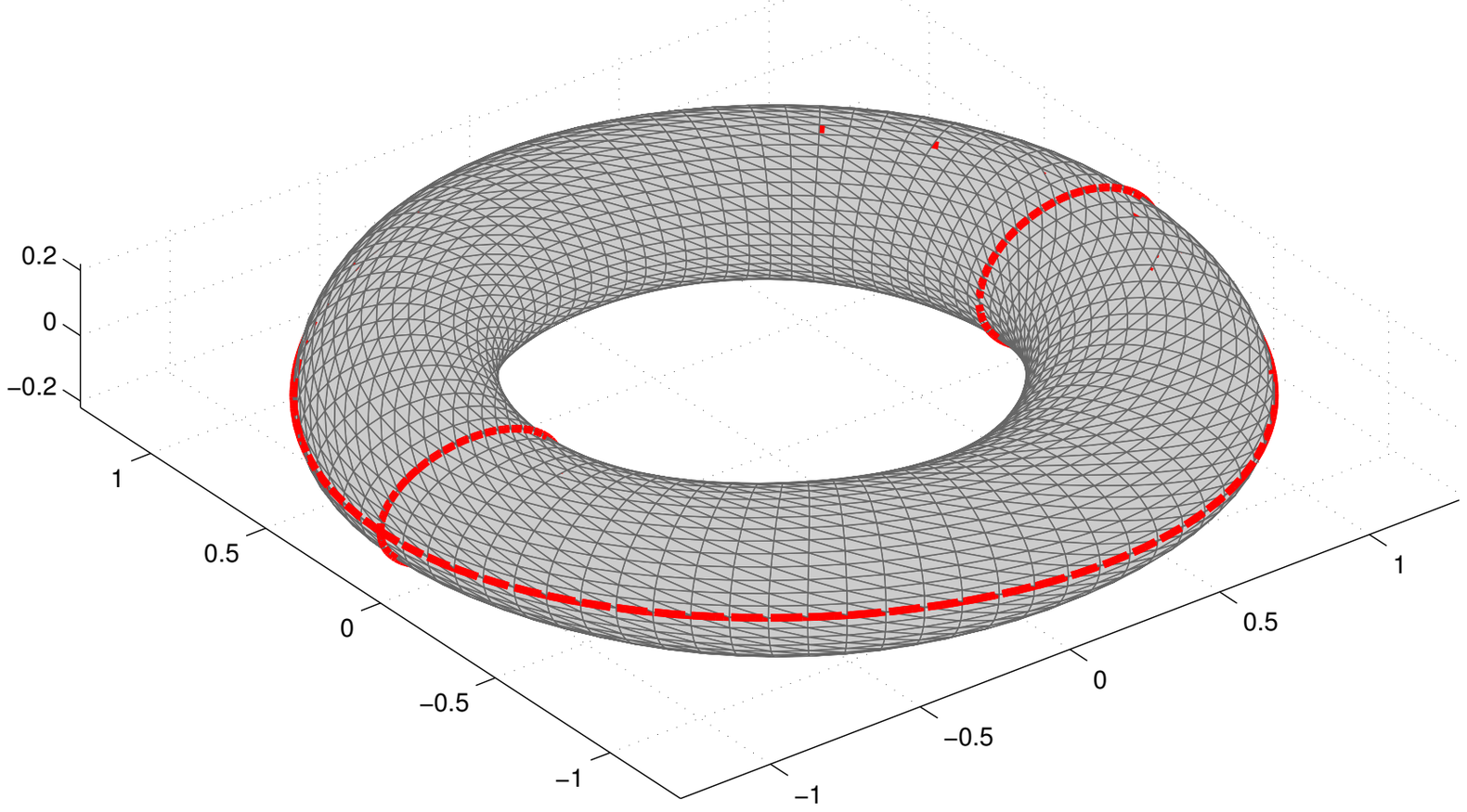
Let **A** denote an $N \times N$ matrix arising upon discretizing a boundary integral operator

$$[Aq](\boldsymbol{x}) = q(\boldsymbol{x}) + \int_\Gamma \frac{1}{|\boldsymbol{x}-\boldsymbol{y}|}\, q(\boldsymbol{y})\, dA(\boldsymbol{y}), \qquad \boldsymbol{x} \in \Gamma,$$

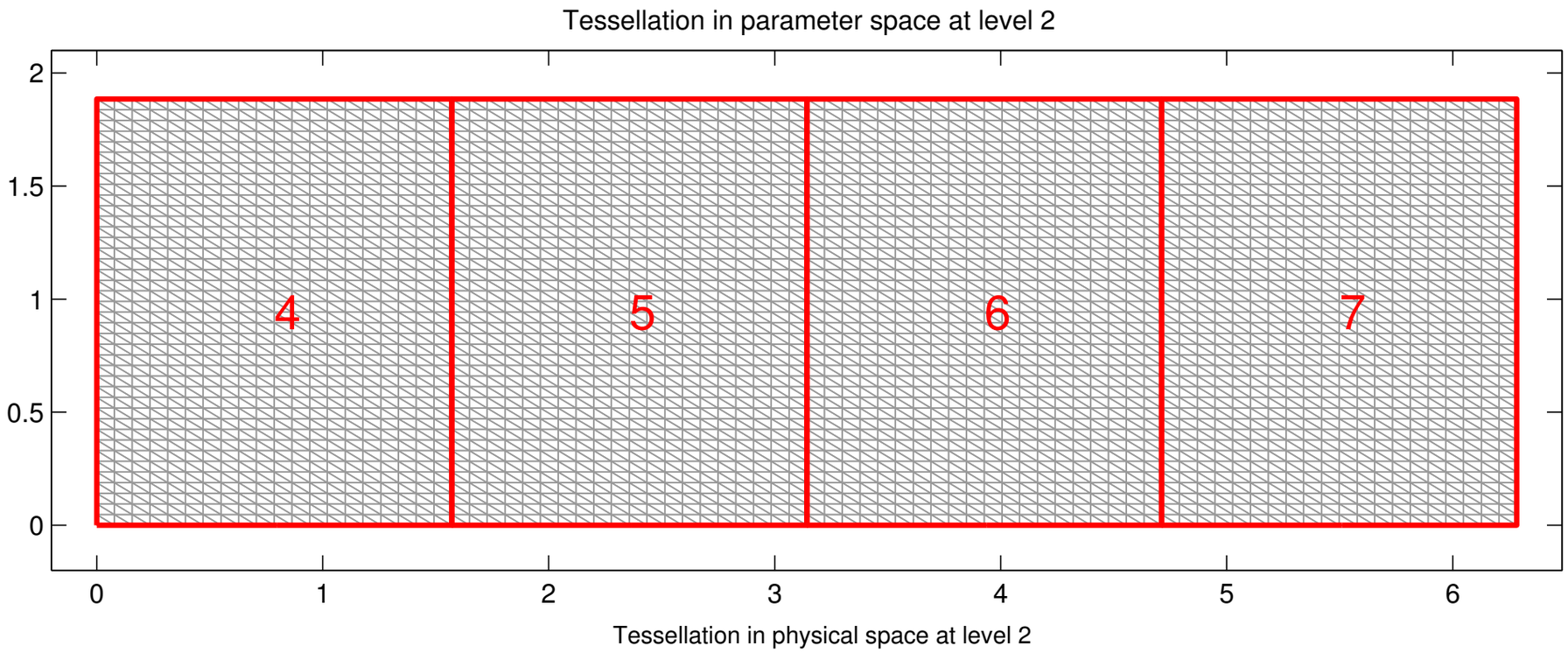
where Γ is the "torus-like" domain shown (it is deformed to avoid rotational symmetry).

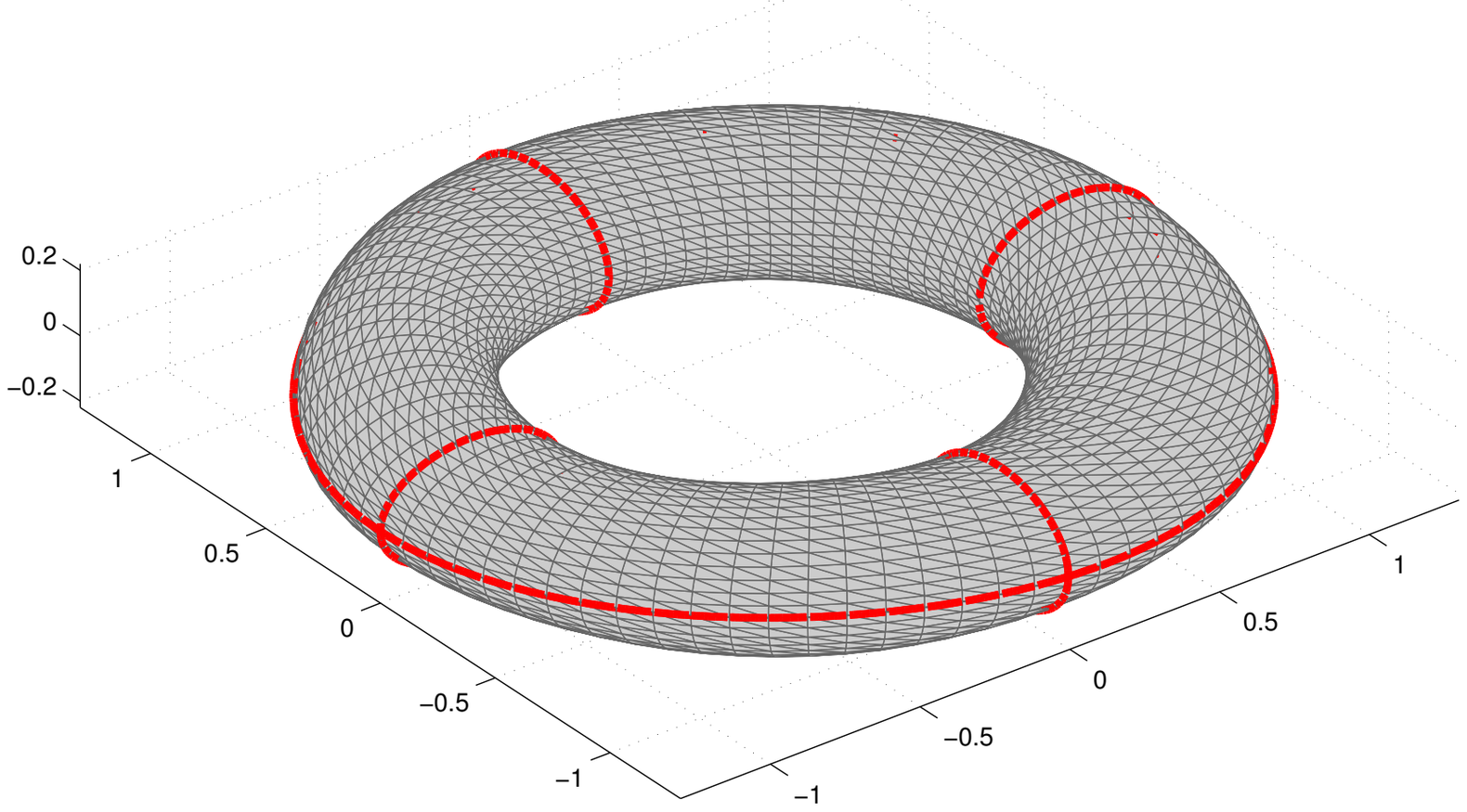
We construct a tree by bisecting *in parameter space* — level 1.



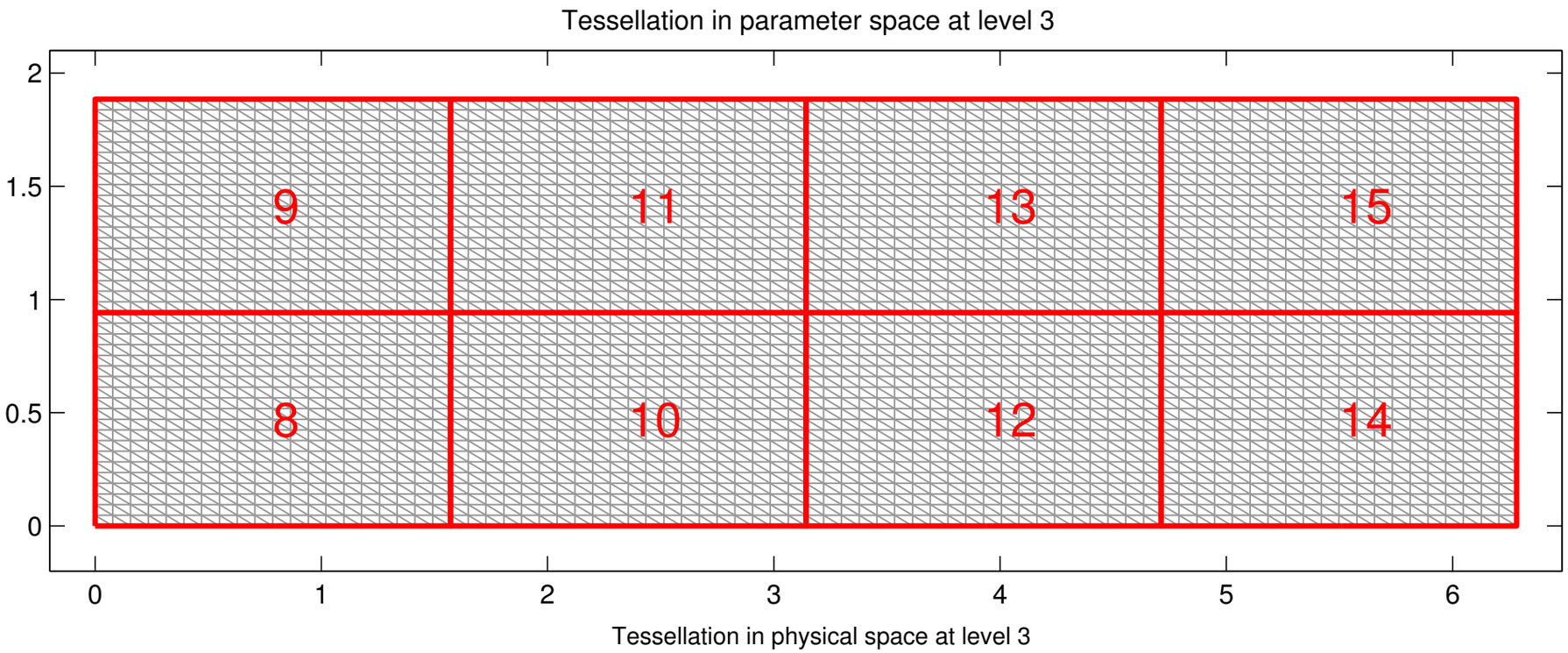


We construct a tree by bisecting *in parameter space* — level 2.

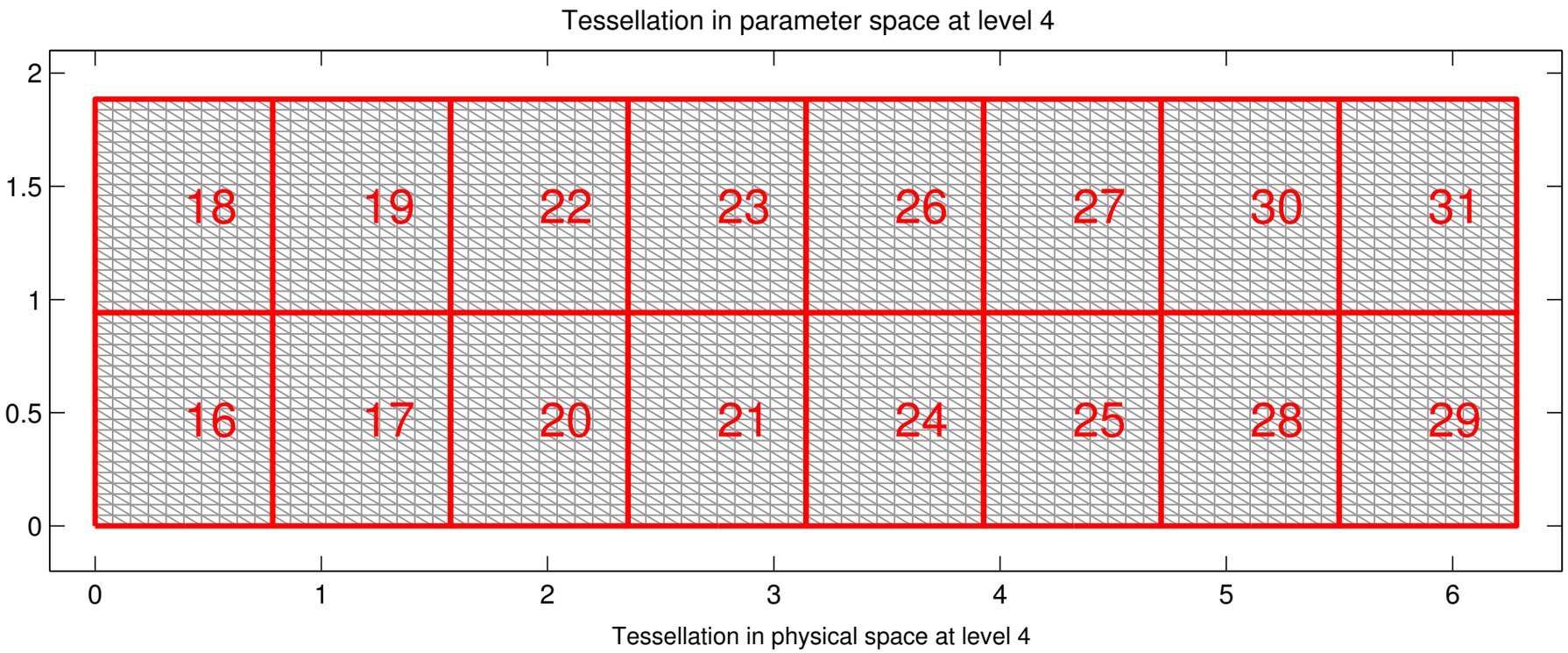




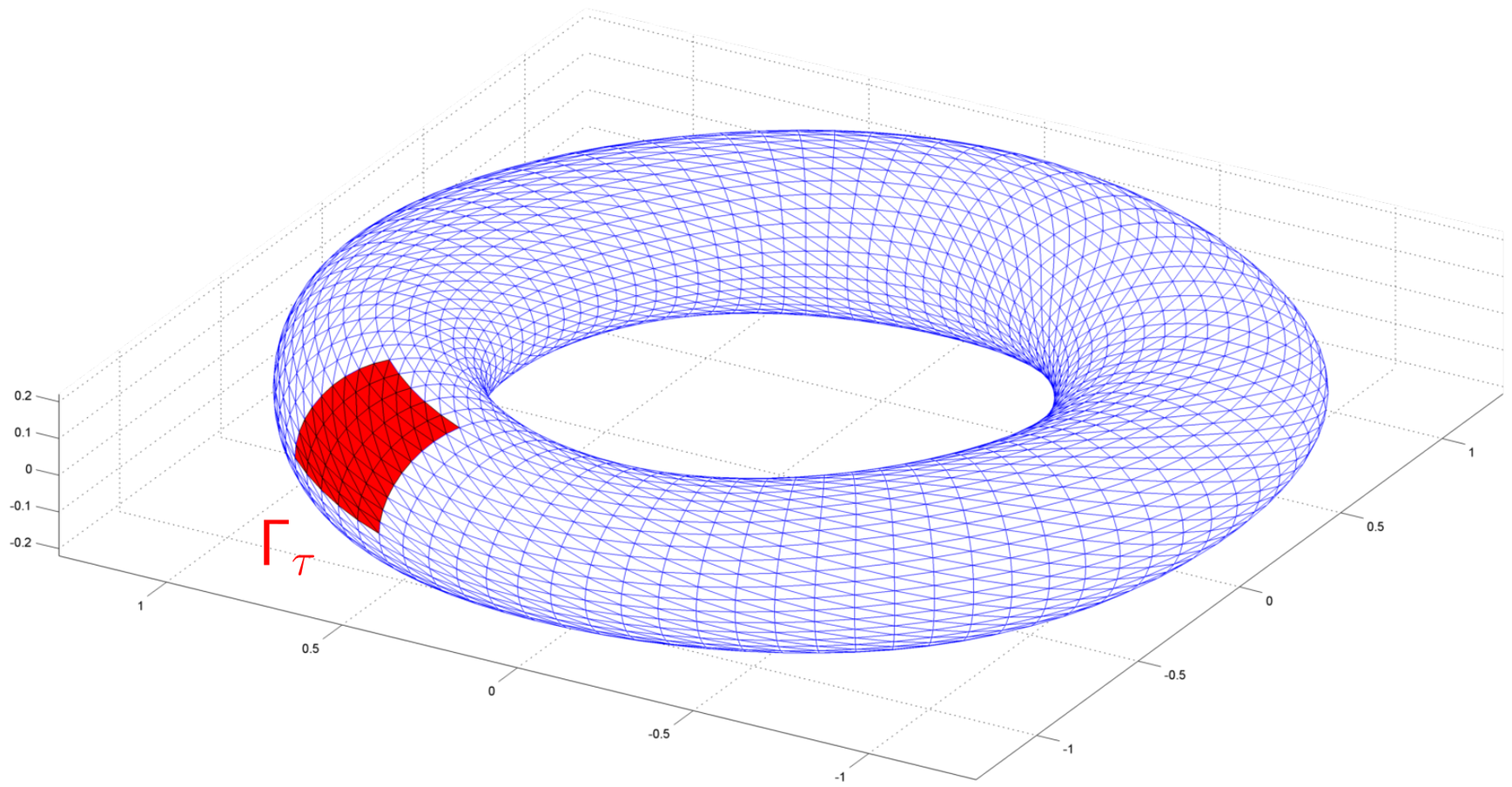
We construct a tree by bisecting *in parameter space* — level 3.



We construct a tree by bisecting *in parameter space* — level 4.



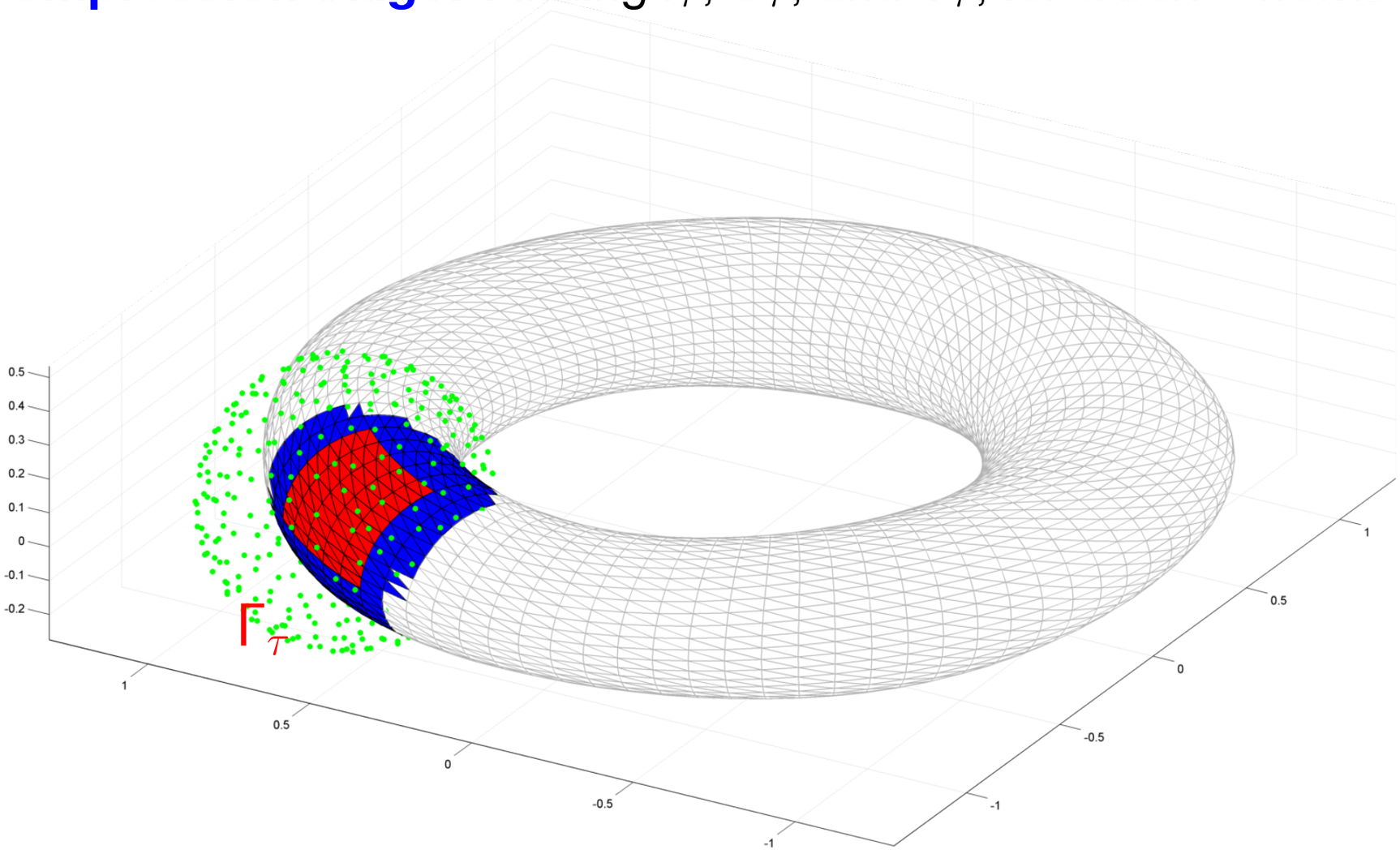
Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle as before.



- Points in I_τ.
- Points in I_τ^{c}.

At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.

Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle as before.

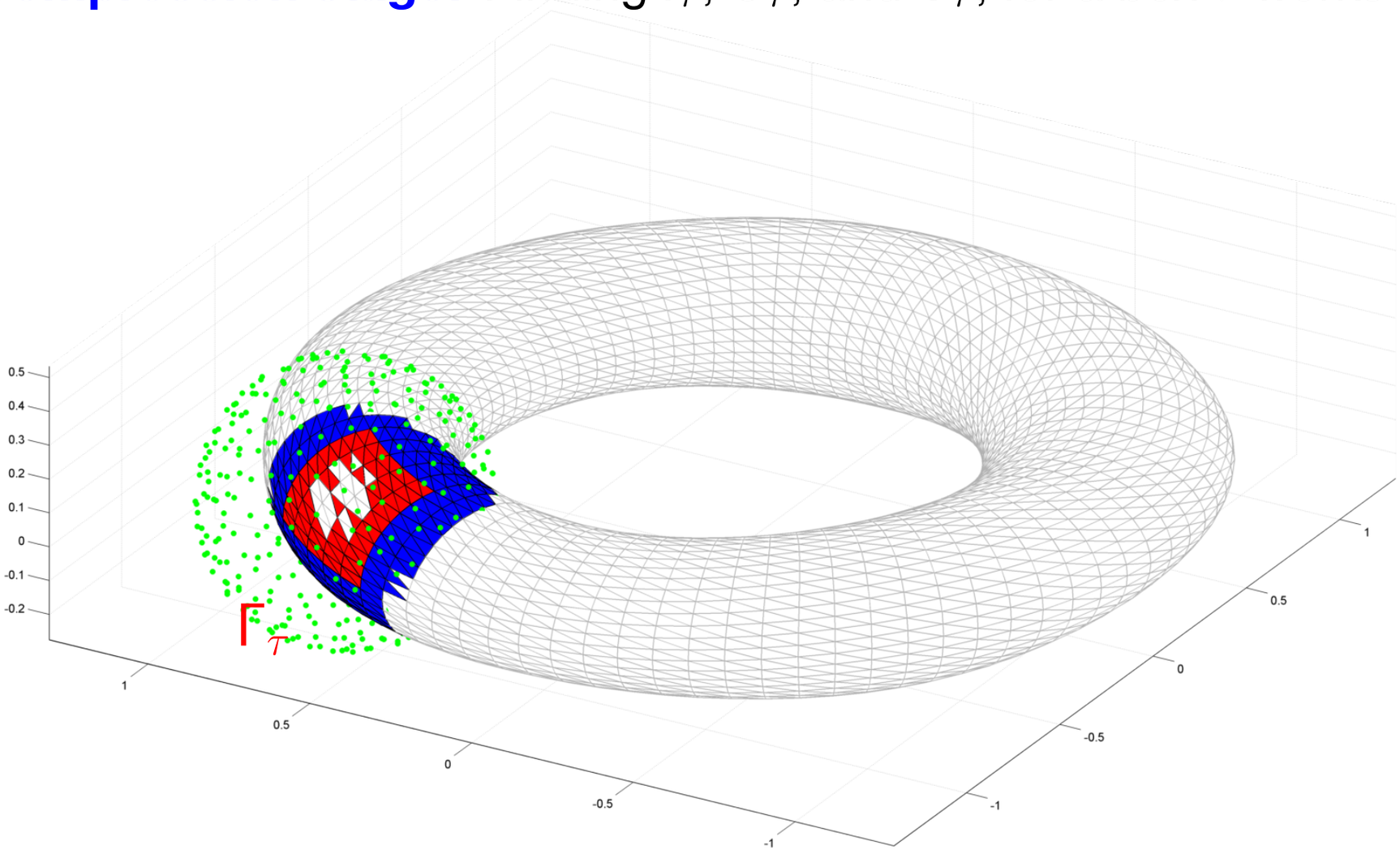


- Points in I_τ.
- Points in $I_\tau^{(\mathrm{near})}$.
- Points in Γ_{proxy}.

At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.

But, using the *Green localization trick,* we only need to ID the matrix $[\mathbf{A}(I_\tau, I_\tau^{(\mathrm{near})})\ \mathbf{G}]$, where **G** is the matrix of interaction with the proxy surface (green).

Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle as before.

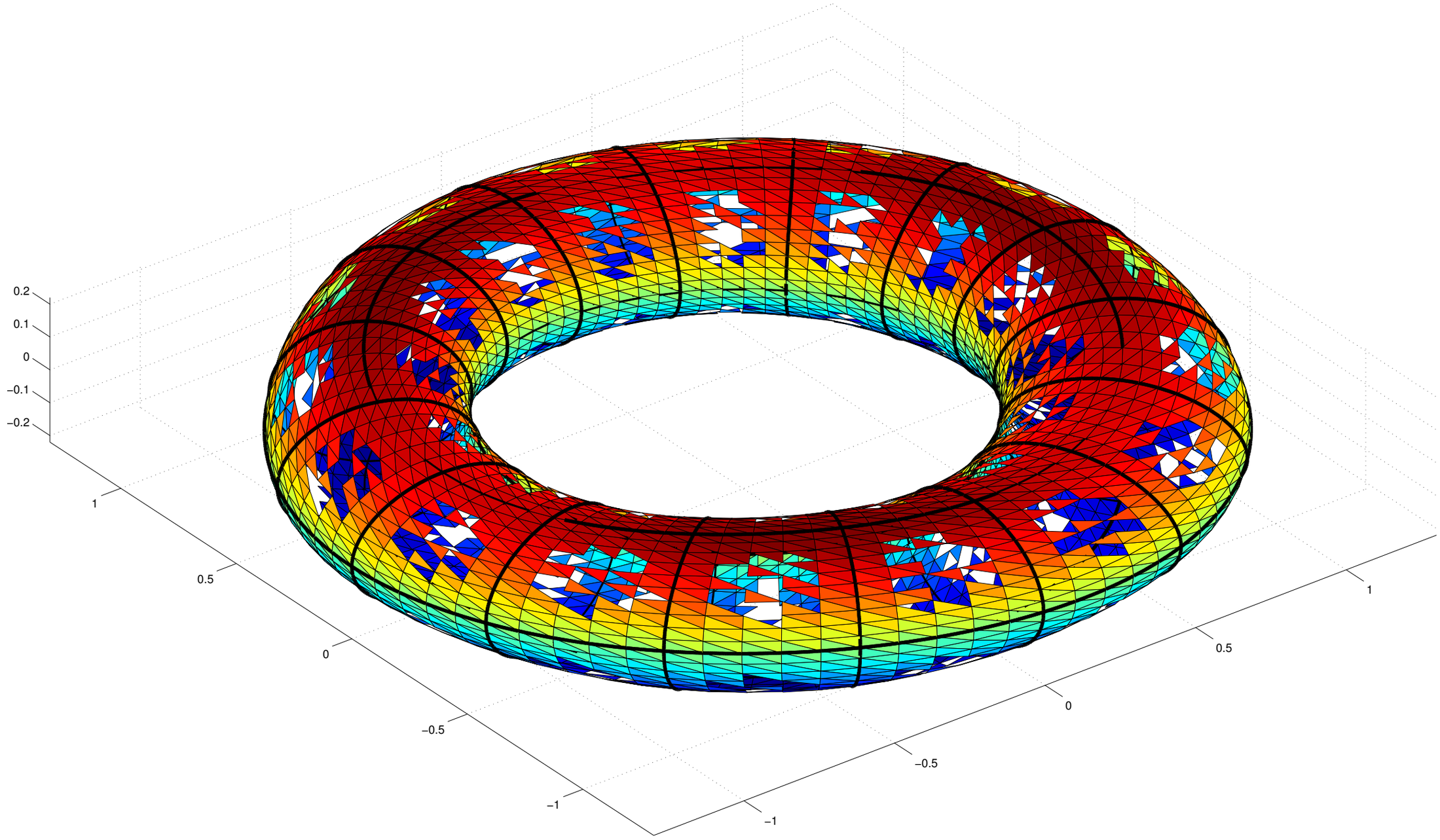


- Points in $\tilde{I}_\tau$.
- Points in $I_\tau^{(\mathrm{near})}$.
- Points in Γ_{proxy}.

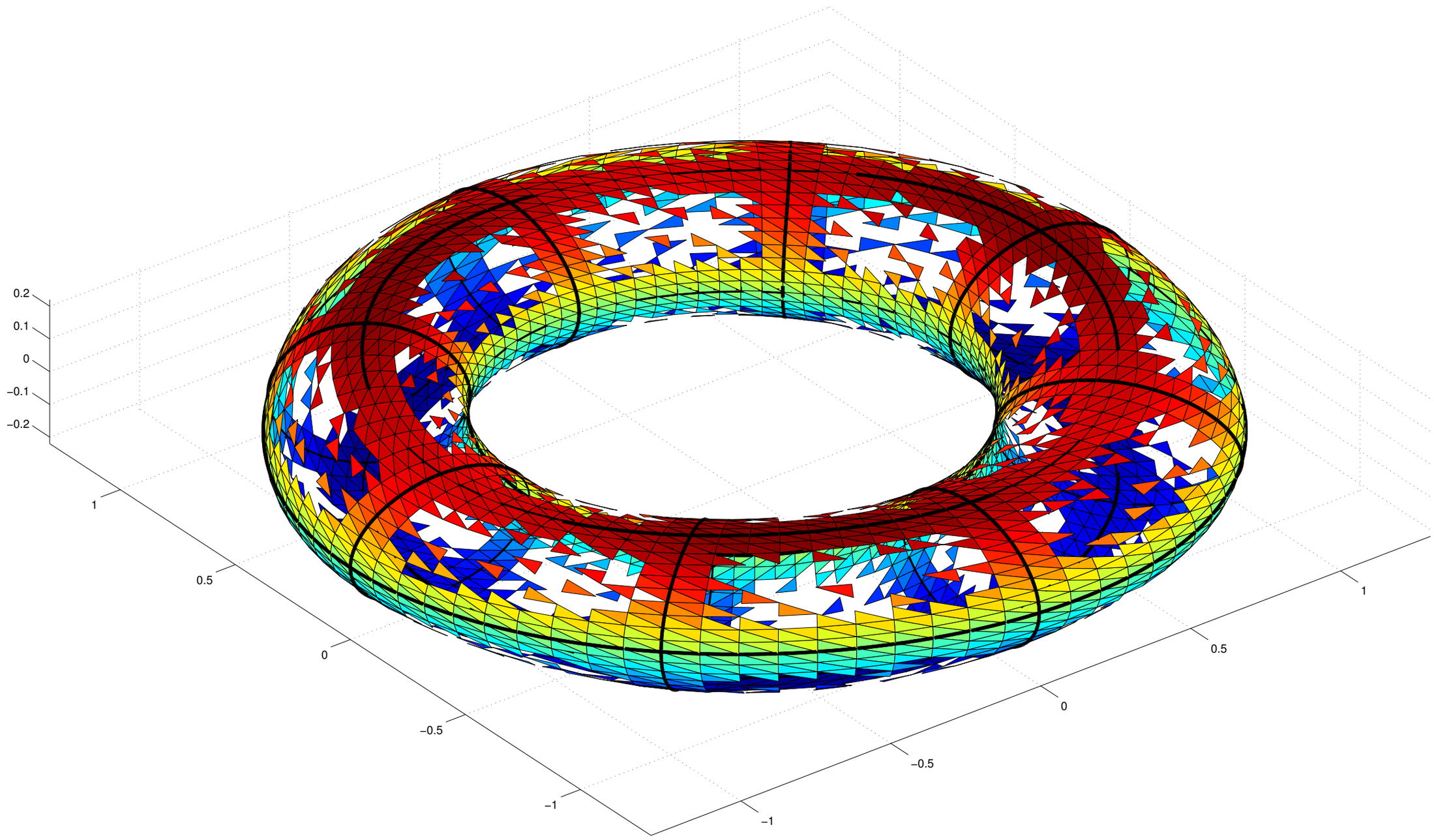
At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.

But, using the *Green localization trick,* we only need to ID the matrix $[\mathbf{A}(I_\tau, I_\tau^{(\mathrm{near})})\ \mathbf{G}]$, where **G** is the matrix of interaction with the proxy surface (green).

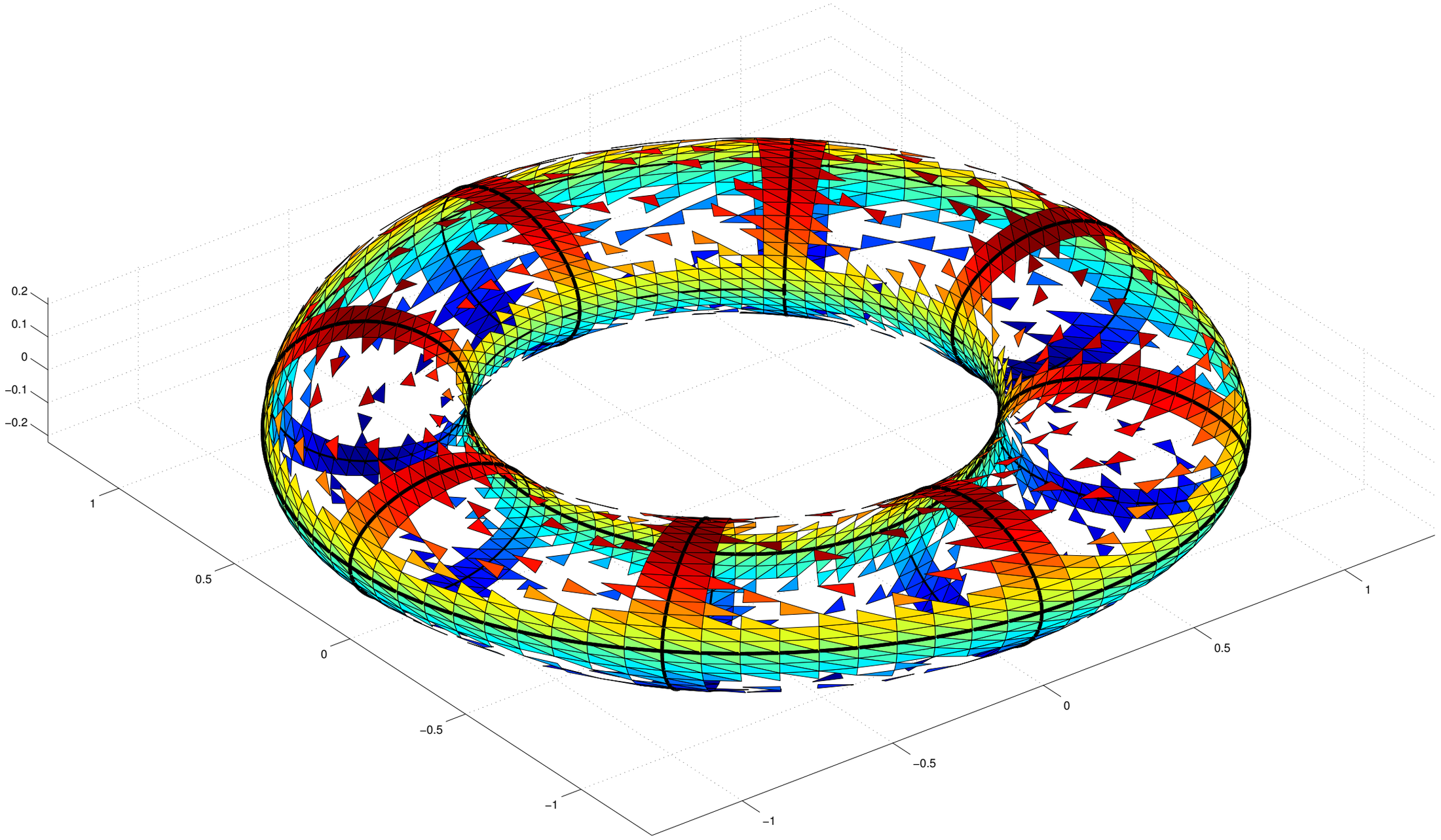
The domain in physical space



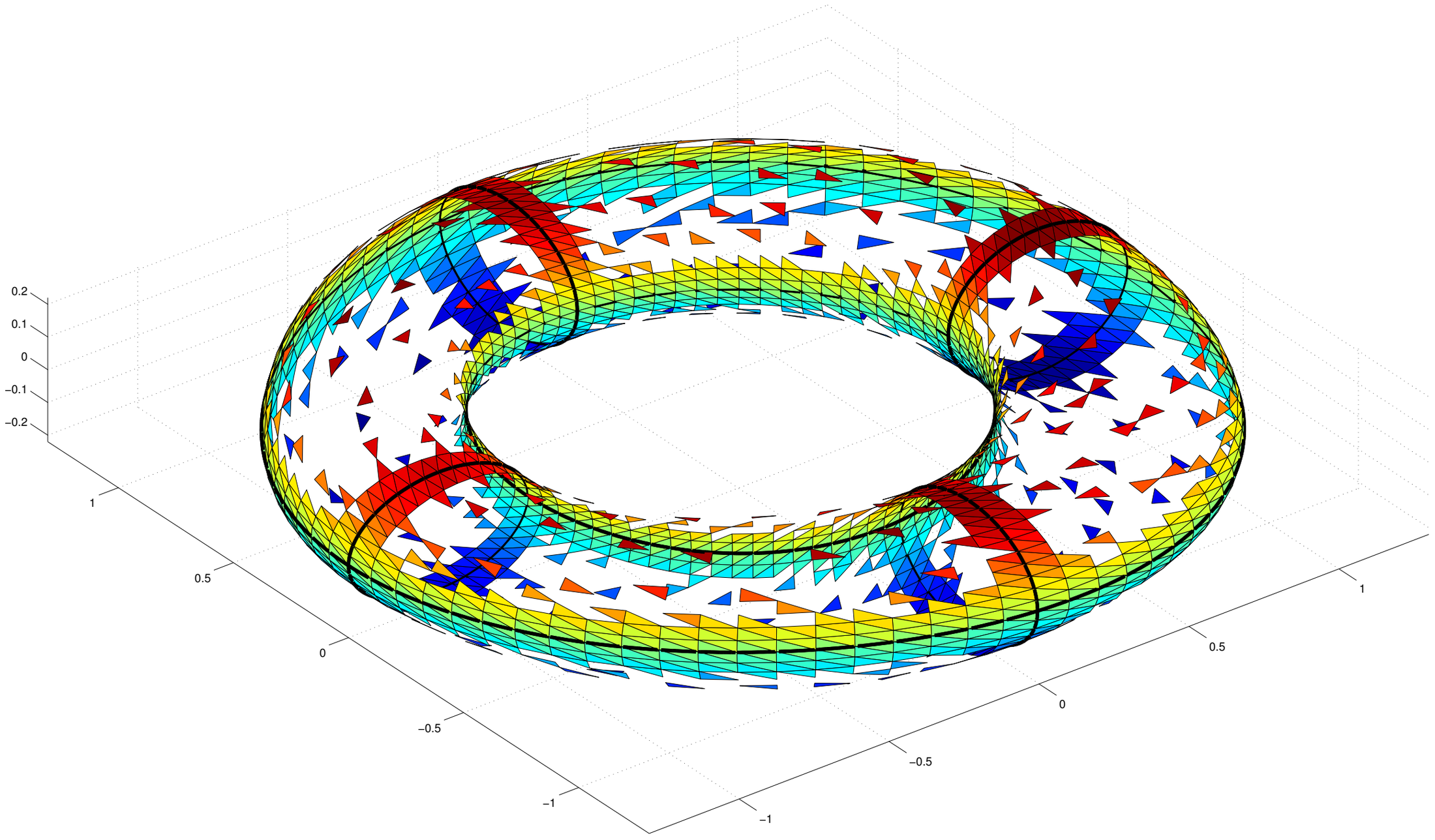
The domain in physical space



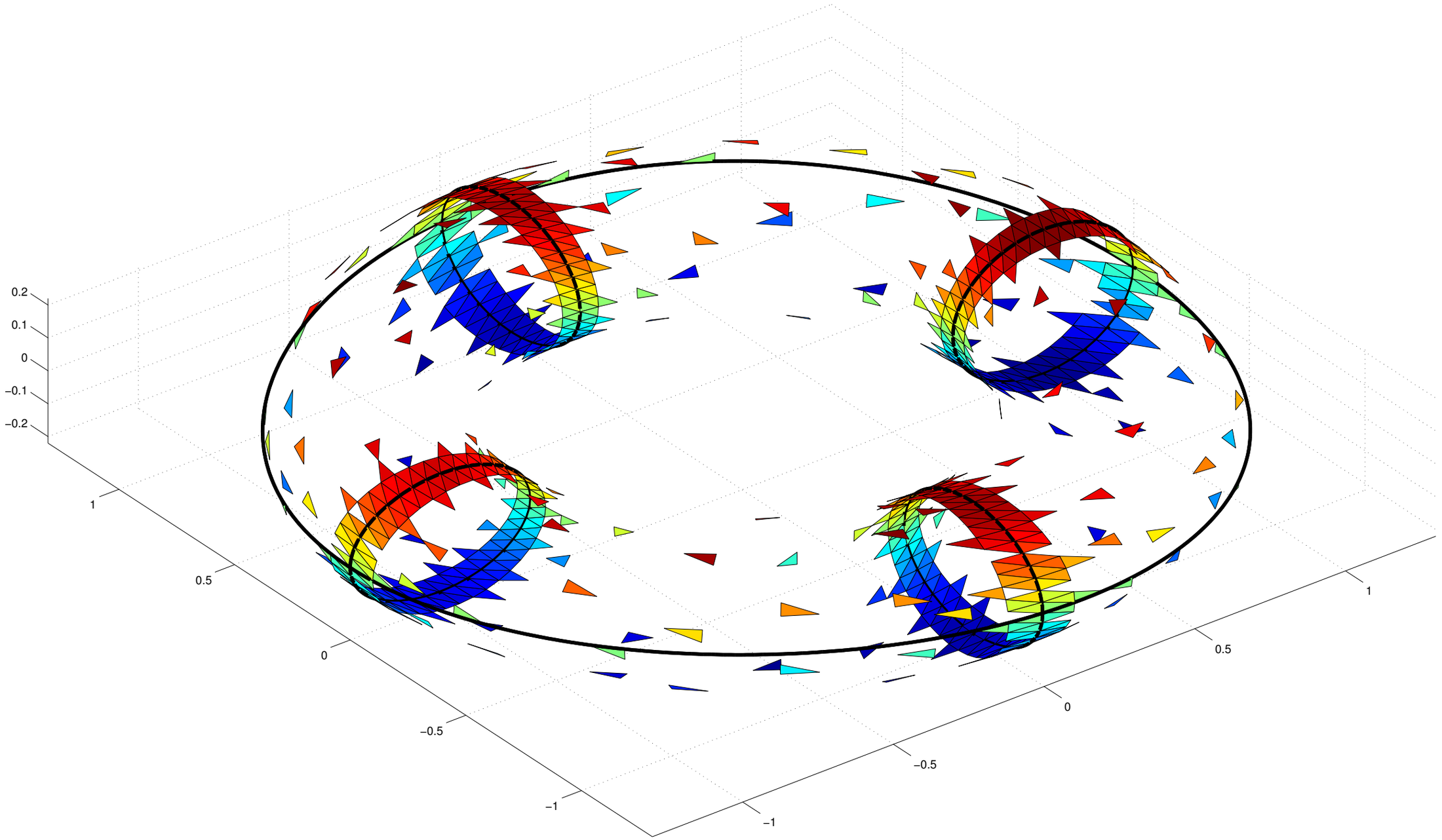
The domain in physical space



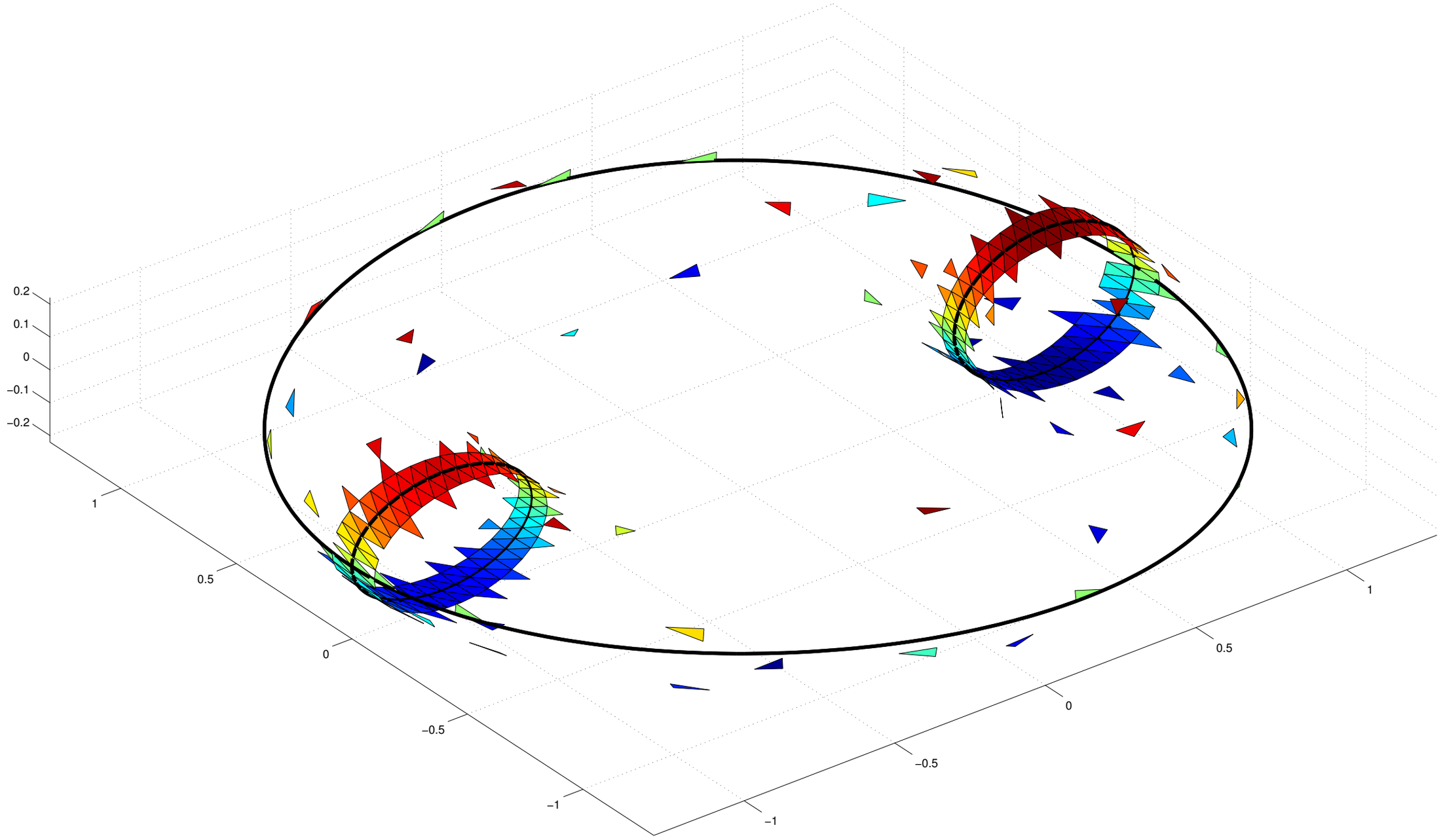
The domain in physical space

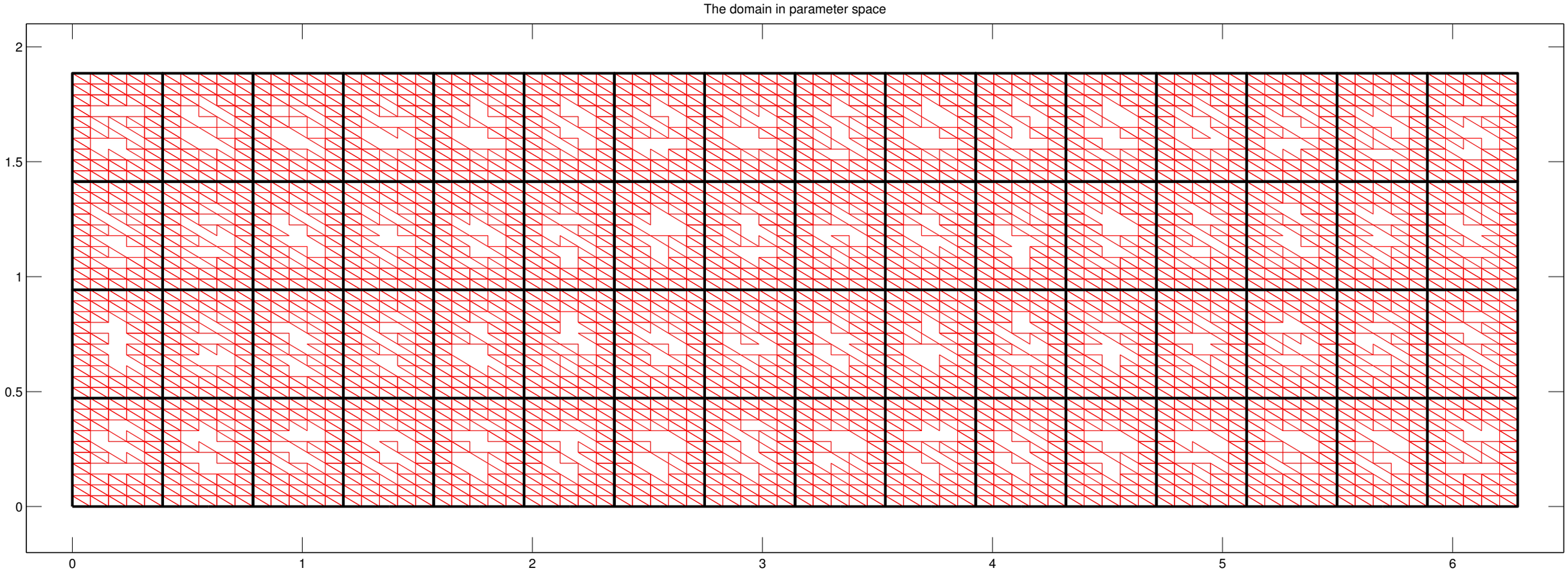


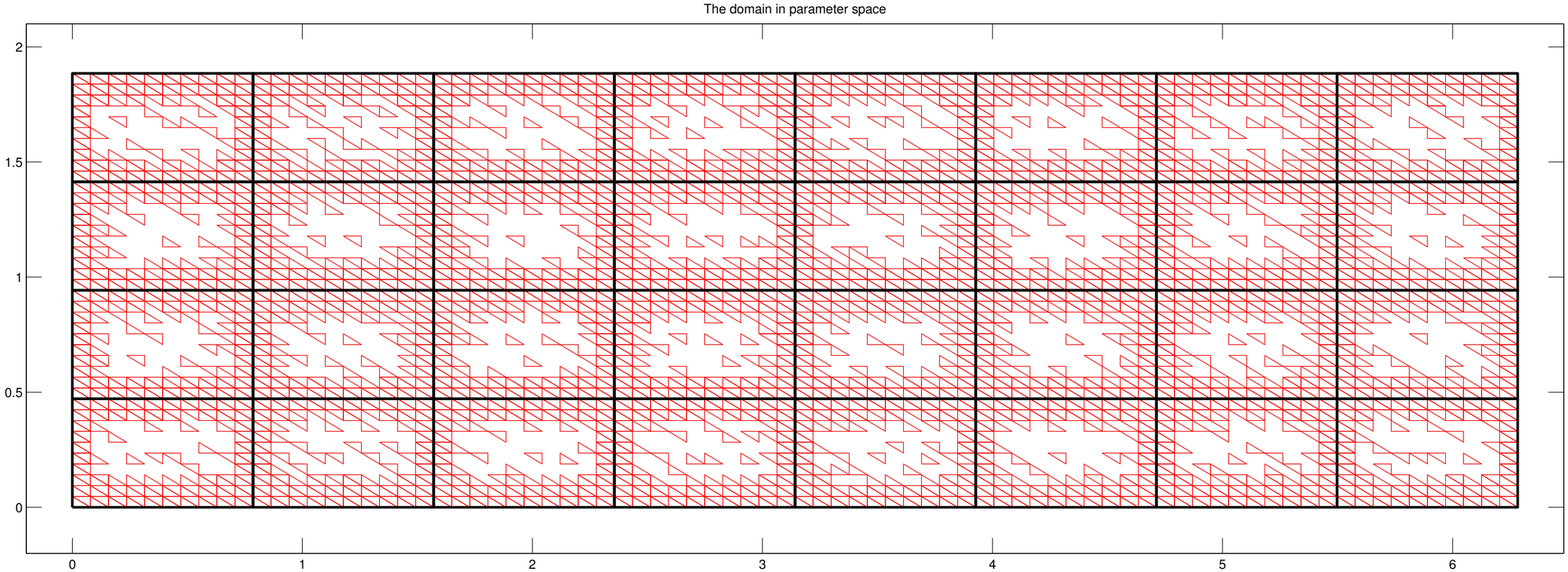
The domain in physical space



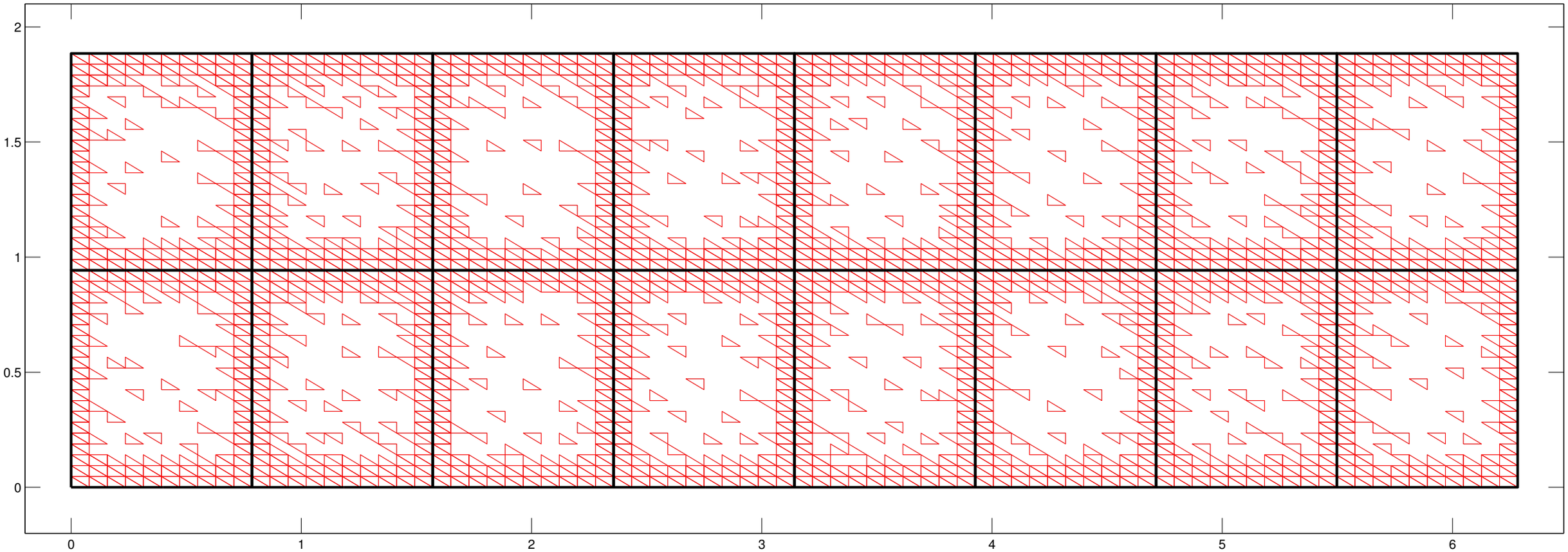
The domain in physical space



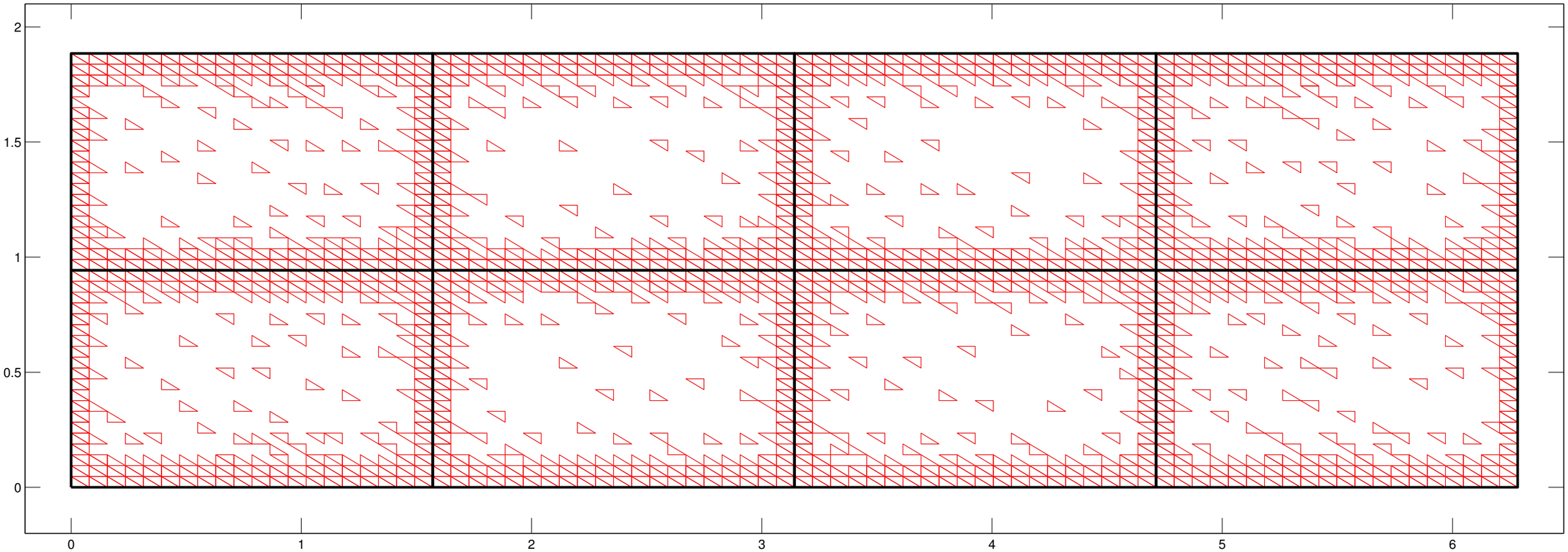

The domain in parameter space
2
1.5
1
0.5
0
0
1
2
3
4
5
6

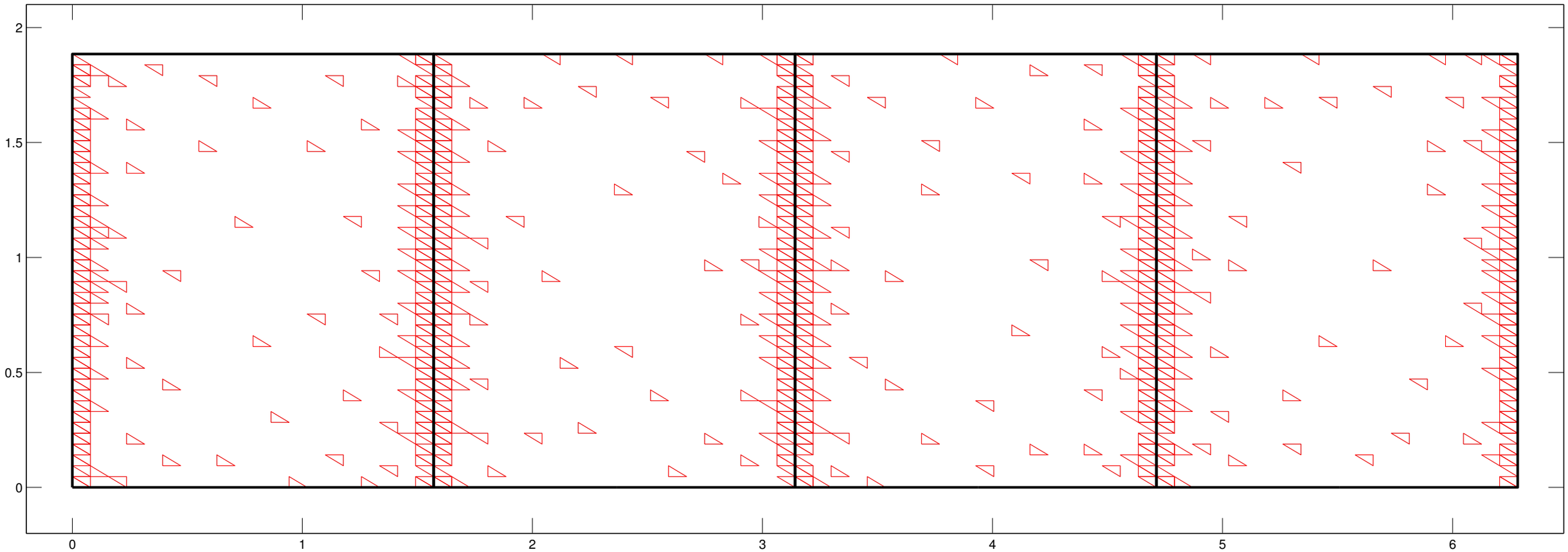

The domain in parameter space
2
1.5
1
0.5
0
0
1
2
3
4
5
6

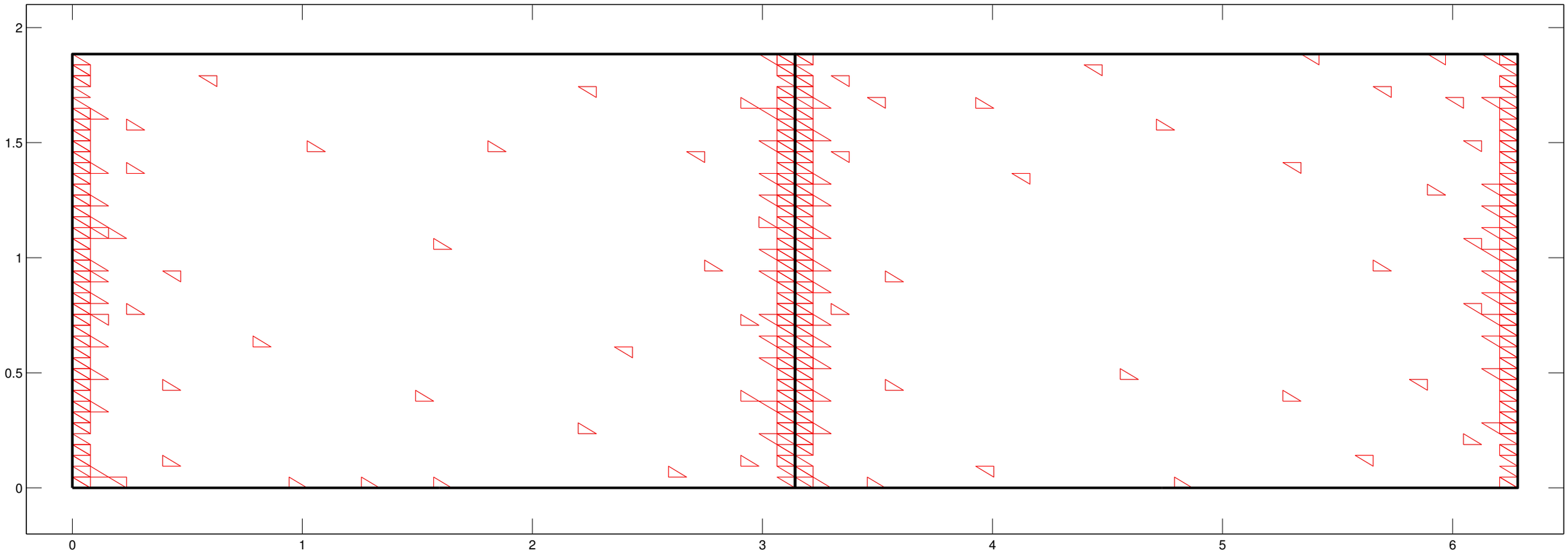
The domain in parameter space



The domain in parameter space




The domain in parameter space
2
1.5
1
0.5
0
0
1
2
3
4
5
6


The domain in parameter space
2
1.5
1
0.5
0
0
1
2
3
4
5
6

Example: Consider free space scattering from a domain with variable wave speed.

Given an "incoming wave" v, we seek to determine an "outgoing wave" u that solves

$$-\Delta u(\boldsymbol{x}) - k^2\,(1 - b(\boldsymbol{x}))\,u(\boldsymbol{x}) = -\,k^2\,b(\boldsymbol{x})\,v(\boldsymbol{x}) \qquad \boldsymbol{x} \in \mathbb{R}^2 \tag{2}$$

$$\lim_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\left(\partial_{|\boldsymbol{x}|}u(\boldsymbol{x}) - ik\,u(\boldsymbol{x})\right) = 0 \tag{3}$$

We suppose that b is a smooth "scattering potential" whose support is contained to some rectangle Ω, $\text{support}(b) \subset \Omega$.

The scattering potential specifies the deviation of the local wave speed $v = v(\boldsymbol{x})$ from the free space wave speed v_{free}: $b(\boldsymbol{x}) = 1 - \left(\dfrac{v_{\text{free}}}{v(\boldsymbol{x})}\right)^2$.

We look for a solution of the form

$$u(\boldsymbol{x}) = [\phi_\kappa * q](\boldsymbol{x}) = \int_{\mathbb{R}^2} \phi_\kappa(\boldsymbol{x} - \boldsymbol{y})\,q(\boldsymbol{y})\,dA(\boldsymbol{y}). \tag{4}$$

where $\phi_\kappa(\boldsymbol{x}) = H_0^{(1)}(\kappa|\boldsymbol{x}|)$ is the free space fundamental solution. u satisfies (3) automatically, and (2) is satisfied if q satisfies the *Lippman-Schwinger integral equation*:

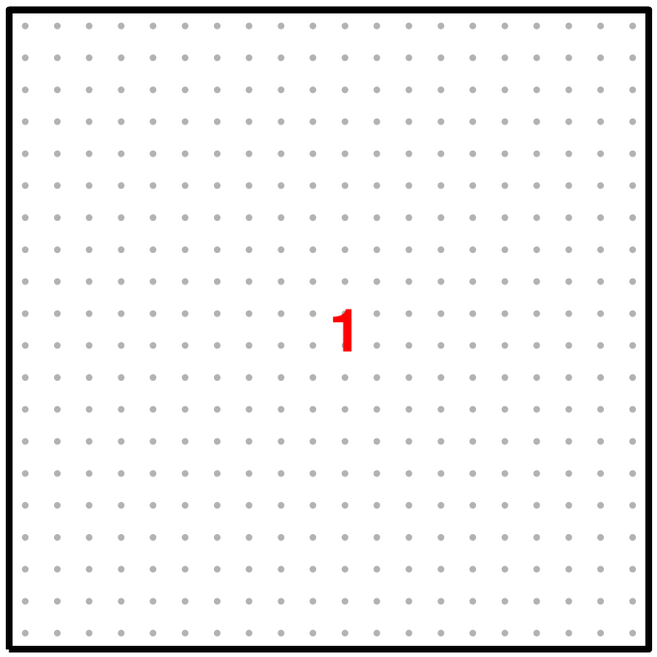
$$q(\boldsymbol{x}) + \kappa^2\,b(\boldsymbol{x}) \int_\Omega H_0^{(1)}(\kappa|\boldsymbol{x} - \boldsymbol{y}|)\,q(\boldsymbol{y})\,dA(\boldsymbol{y}) = -\kappa^2\,b(\boldsymbol{x})\,v(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Omega. \tag{5}$$

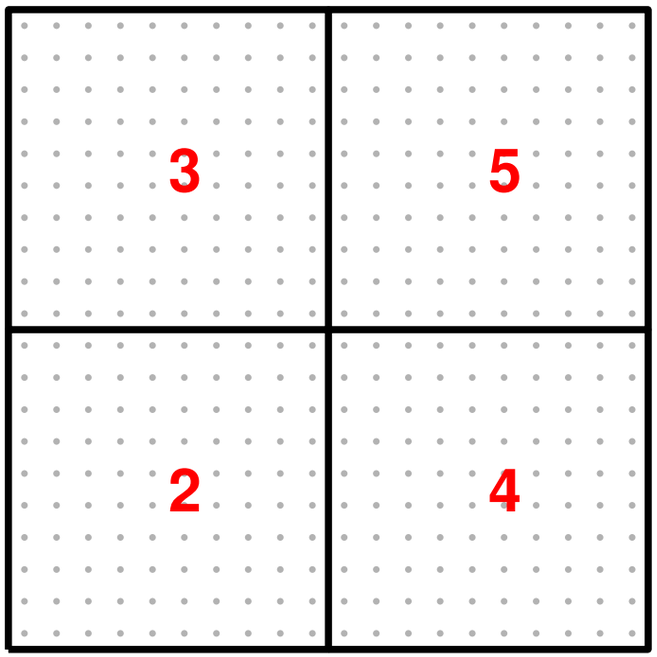
Observe that (5) is a *local equation* defined on the bounded set Ω.

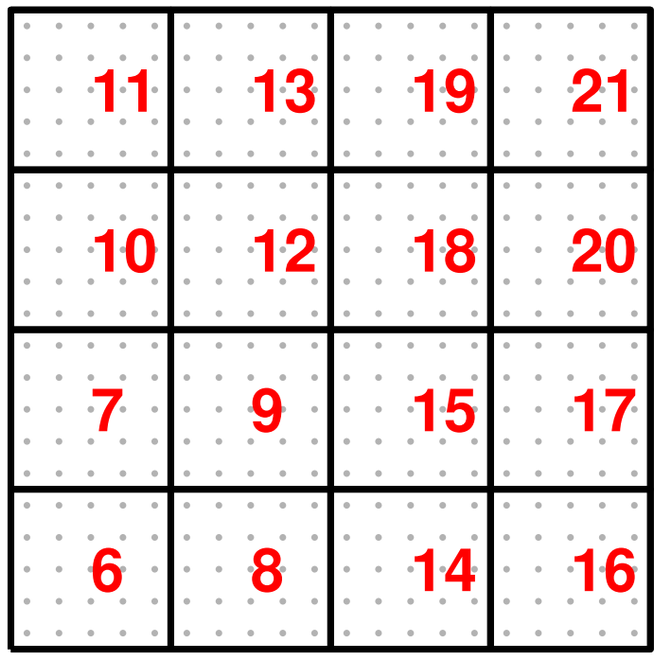
(It is also a second kind Fredholm equation, which is very nice.)

Recall: We seek to solve $q(\boldsymbol{x}) + \kappa^2 b(\boldsymbol{x}) \int_\Omega H_0^{(1)}(\kappa|\boldsymbol{x}-\boldsymbol{y}|)\, q(\boldsymbol{y})\, dA(\boldsymbol{y}) = -\kappa^2 b(\boldsymbol{x})\, v(\boldsymbol{x}),\ \boldsymbol{x} \in \Omega.$

We discretize Ω using a uniform grid, and then split the points into a quad-tree:







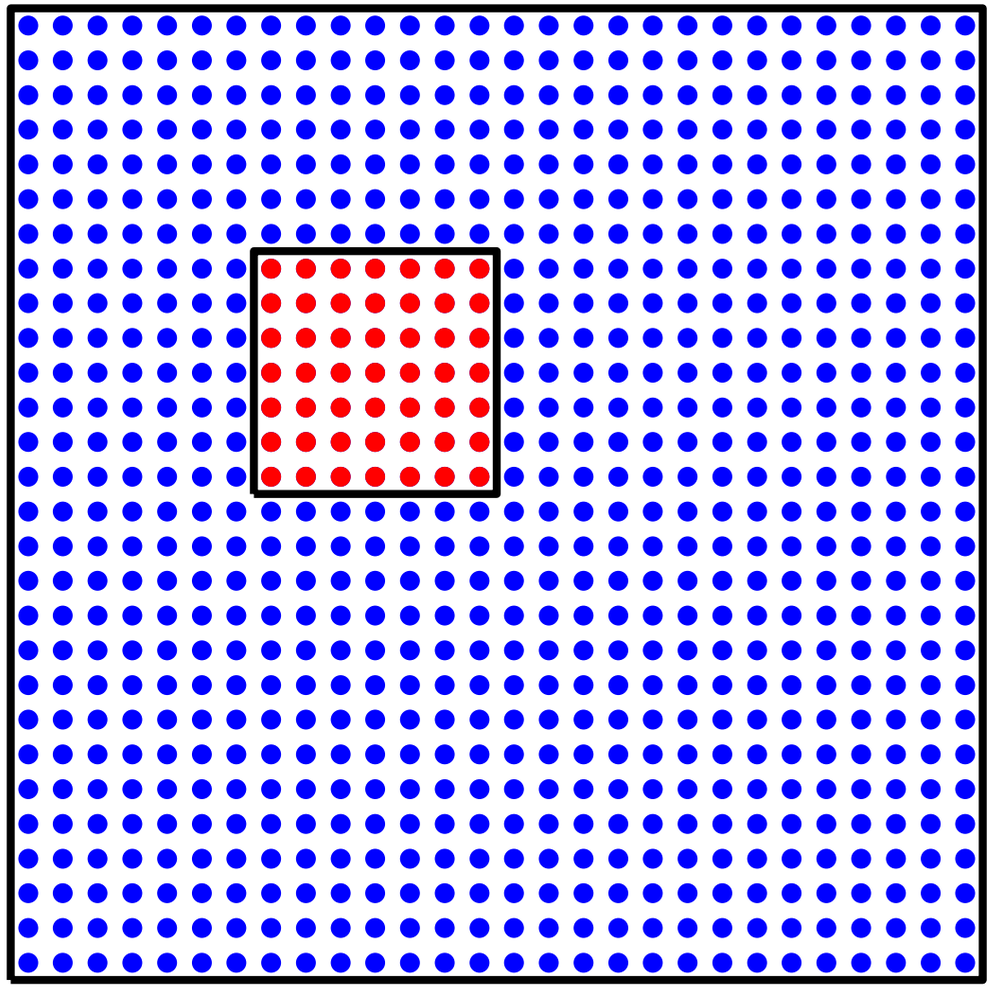
Now discretize the integral equation using Nyström with the trapezoidal rule.

A small number of elements "close to the diagonal" (in physical space) are modified since the kernel in the integral is singular, but most matrix elements are given by

$$\mathbf{A}(i,j) = \kappa^2 b(\boldsymbol{x}_i)\, H_0^{(1)}(\kappa|\boldsymbol{x}_i - \boldsymbol{x}_j|)\, \sqrt{w_i\, w_j}.$$

We will build a direct solver for $\mathbf{Aq} = \mathbf{f}$, where $\mathbf{f}(i) = -\kappa^2 b(\boldsymbol{x}_i)\, v(\boldsymbol{x}_i)\, \sqrt{w_i}$.

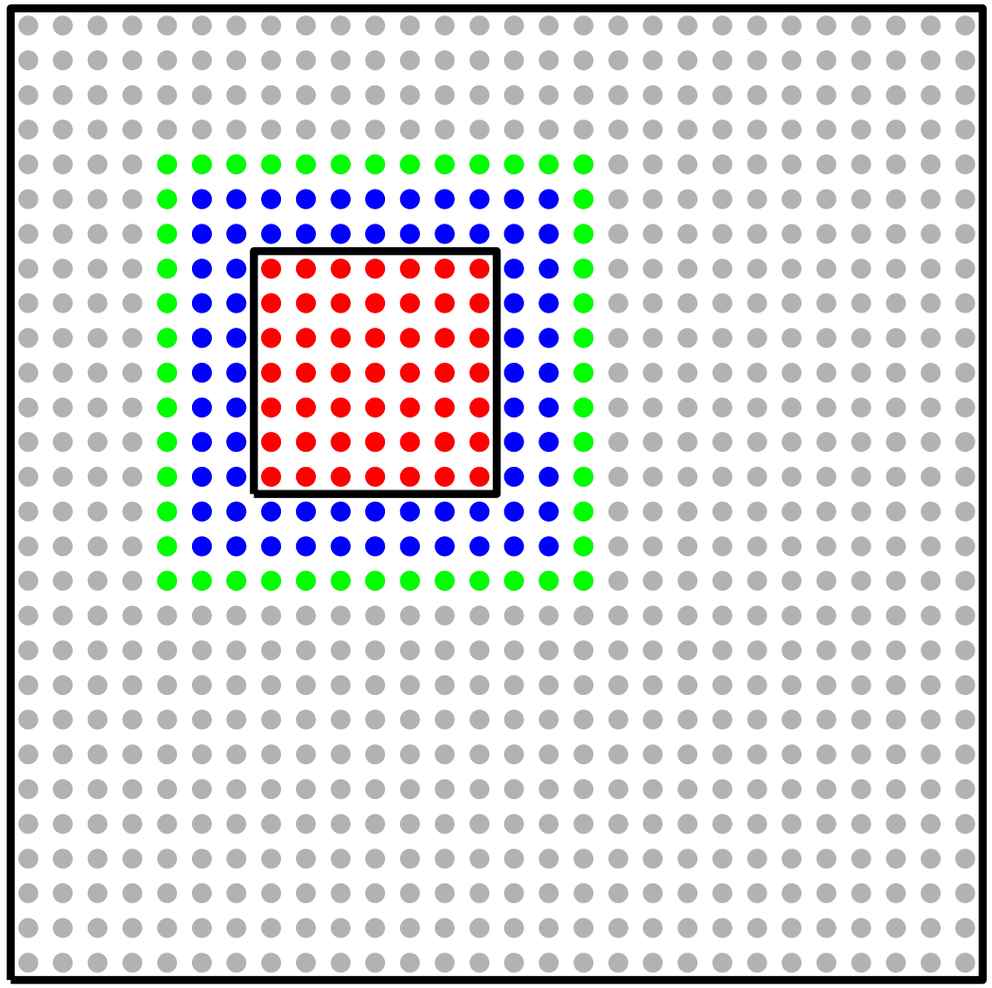
Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle as before.



- Points in I_τ.
- Points in I_τ^{c}.

At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.

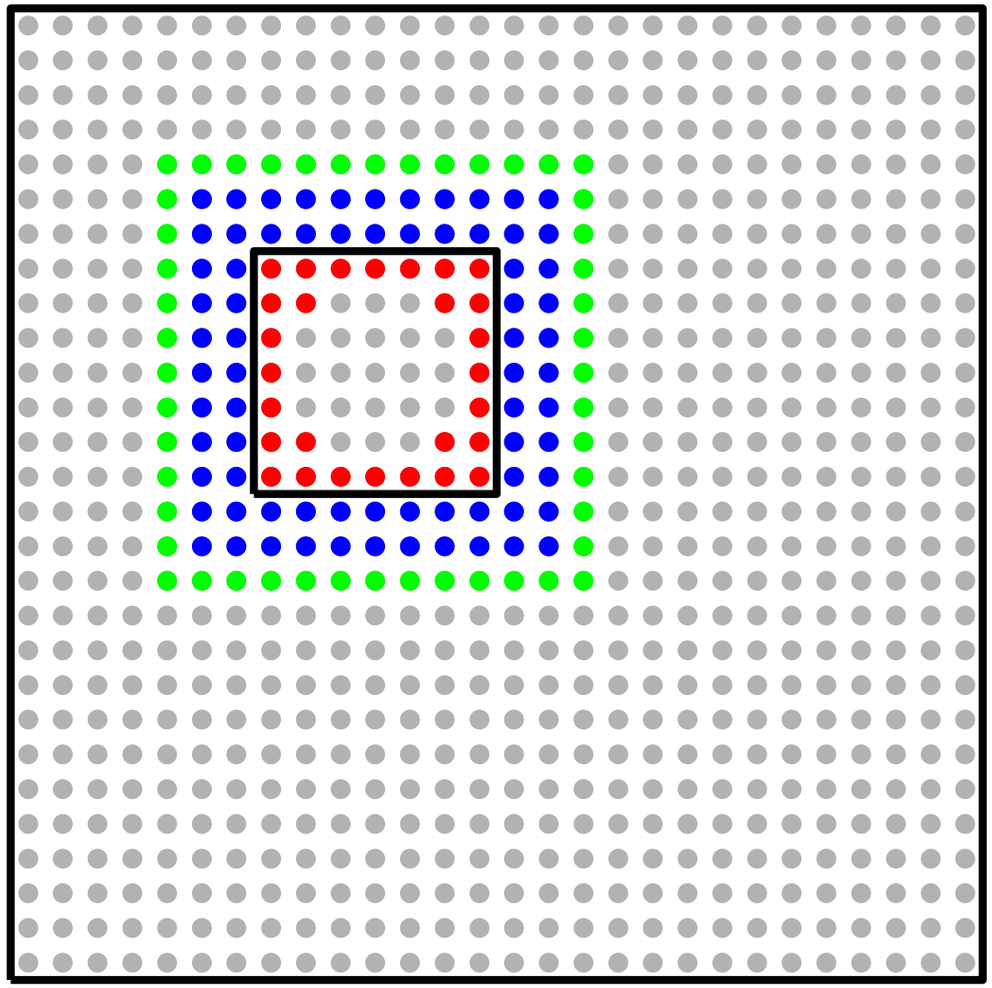
Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle as before.



- Points in I_τ.
- Points in $I_\tau^{(\mathrm{near})}$.
- Points in Γ_{proxy}.

At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.
But, using the *Green localization trick,* we only need to ID the matrix $[\mathbf{A}(I_\tau, I_\tau^{(\mathrm{near})})\ \mathbf{G}]$,
where **G** is the matrix of interaction with the proxy surface (green).

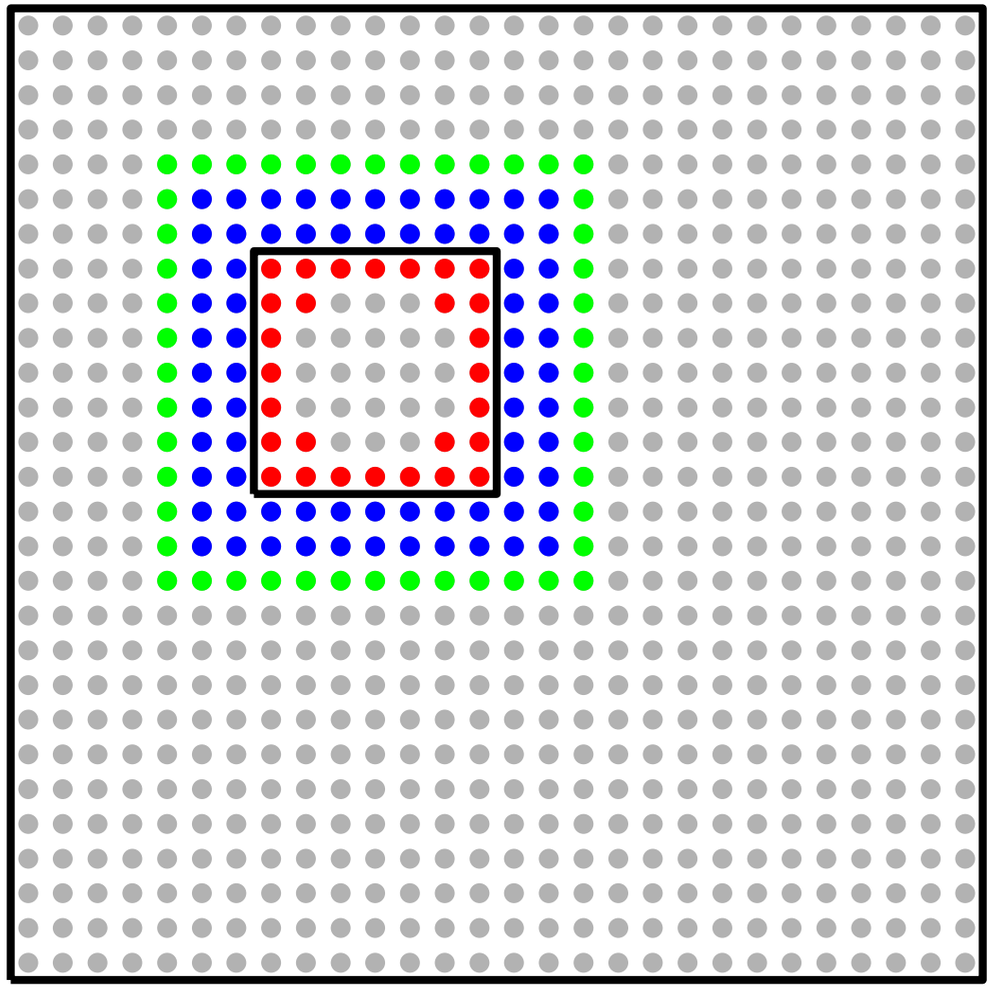
Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle as before.



- Points in $\tilde{I}_\tau$.
- Points in $I_\tau^{(\mathrm{near})}$.
- Points in Γ_{proxy}.

At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.
But, using the *Green localization trick,* we only need to ID the matrix $[\mathbf{A}(I_\tau, I_\tau^{(\mathrm{near})})\ \mathbf{G}]$,
where **G** is the matrix of interaction with the proxy surface (green).

Compression stage: Finding $\tilde{I}_\tau$, $\mathbf{U}_\tau$, and $\mathbf{V}_\tau$, for a box τ works in principle as before.



- Points in $\tilde{I}_\tau$.
- Points in $I_\tau^{(\mathrm{near})}$.
- Points in Γ_{proxy}.

At first, it seems like we need to perform an ID of the large matrix $\mathbf{A}(I_\tau, I_\tau^{\mathrm{c}})$.
But, using the *Green localization trick,* we only need to ID the matrix $[\mathbf{A}(I_\tau, I_\tau^{(\mathrm{near})})\ \mathbf{G}]$, where $\mathbf{G}$ is the matrix of interaction with the proxy surface (green).
Peculiarity of Lippman-Schwinger I: There is no need for a proxy surface in this case ...
Peculiarity of Lippman-Schwinger II: $\mathbf{A} = \mathbf{I} + \mathbf{BG}$ where $\mathbf{B}$ is diagonal, and $\mathbf{G}$ is translation invariant. This means we only need to compress one box per level.

Alternative approaches:

- *$\mathcal{H}$-matrix methods:* Rely on buffering. "Adaptive Cross Approximation (ACA)" for compression (instead of "Green trick"). Work by Bebendorf, Börm, Grasedyck, Hackbusch, Khoromskij, Sauter, etc.

- *Inversion of FMM structure:* You can "roll out" the FMM and formulate it as a block-sparse matrix. Then invoke sparse direct solvers to invert. Work by Pals & Chandrasekaran.

- *Piggy-back on sparse direct solvers:* You can "roll out" the HBS representation and formulate it as a block-sparse matrix. Then invoke sparse direct solvers (e.g. UMFPACK, MUMPS) to invert. Work by Ho & Greengard.

- *"Shifted skeletonization":* It appears that $O(N)$ complexity can be attained by skeletonizing "intermediate levels" that are shifted by "half a grid spacing." Recent work by Ho and Ying.

- *Higher-frequency scattering:* Very recent work in Eric Michielssen's group seems to show that so called "butterfly algorithms" (see, e.g., M. O'Neil, F. Woolfe, and V. Rokhlin, *ACHA* 2010) can be used to build linear complexity algorithms for high frequency scattering problems.

Bibliography: *See page 2 for background, and website for details.*

- Skeletonization: H. Cheng, Z. Gimbutas, P.G. Martinsson, V. Rokhlin, "On the compression of low rank matrices". *SIAM Journal of Scientific Computing*, **26**(4), pp. 1389-1404, 2005.
- Original paper (2D BIE): P.G. Martinsson and V. Rokhlin, "A fast direct solver for boundary integral equations in two dimensions". *Journal of Computational Physics*, **205**(1), pp. 1 – 23, 2005. (Inspired by earlier work by Greengard, Rokhlin, Starr . . .)
- $O(N)$ direct solver for high-frequency scattering (elongated domains): P.G. Martinsson and V. Rokhlin, "A fast direct solver for scattering problems involving elongated structures". *Journal of Computational Physics*, **221**, pp. 288–302, 2007.
- Extension to 3D: L. Greengard, D. Gueyffier, P.G. Martinsson, V. Rokhlin, "Fast direct solvers for integral equations in complex three-dimensional domains". *Acta Numerica*, **18**, pp. 243–275, 2009.
- Survey on HBS based methods for integral equations: A. Gillman, P. Young, and P.G. Martinsson, "A direct solver with $O(N)$ complexity for integral equations on one-dimensional domains". *Frontiers of Math. in China*, **7**(2), pp. 217–247, 2012.
- $O(N)$ volume IE in 2D: E. Corona, P.G. Martinsson, D. Zorin "An $O(N)$ Direct Solver for Integral Equations in the Plane". To appear in *ACHA.* (arXiv.org report #1303.5466).
- 3D BIE: J. Bremer, A. Gillman, P.G. Martinsson, "A high-order accurate accelerated direct solver for acoustic scattering from surfaces." To appear in *BIT Numerical Mathematics.* (arXiv.org #1308.6643)
- Corner compression: A. Gillman, S. Hao, and P.G. Martinsson, "A simplified technique for the efficient and highly accurate discretization of boundary integral equations in 2D on domains with corners." *Journal of Computational Physics*, **256**(1), pp. 214–219, 2014. (Drawing on Bremer, Helsing, etc.)