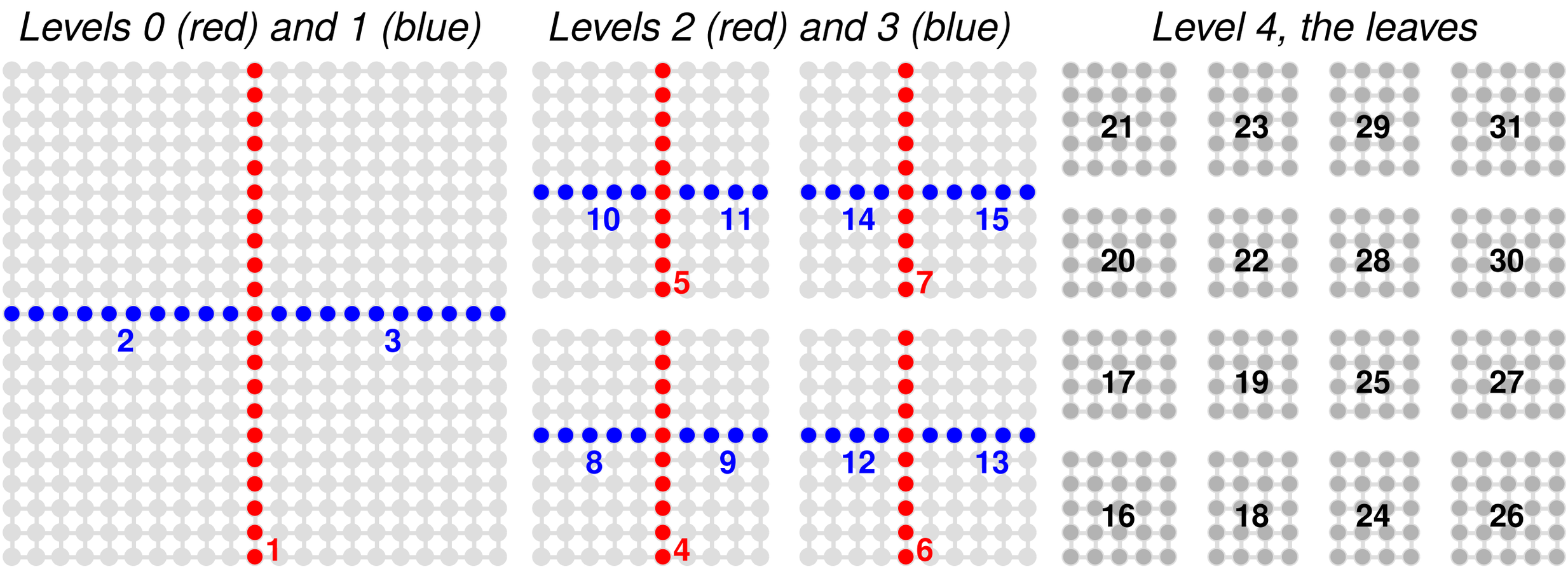
MATH 393C: Fast Methods in Scientific Computing

Lecture on April 23, 2019

P.G. Martinsson

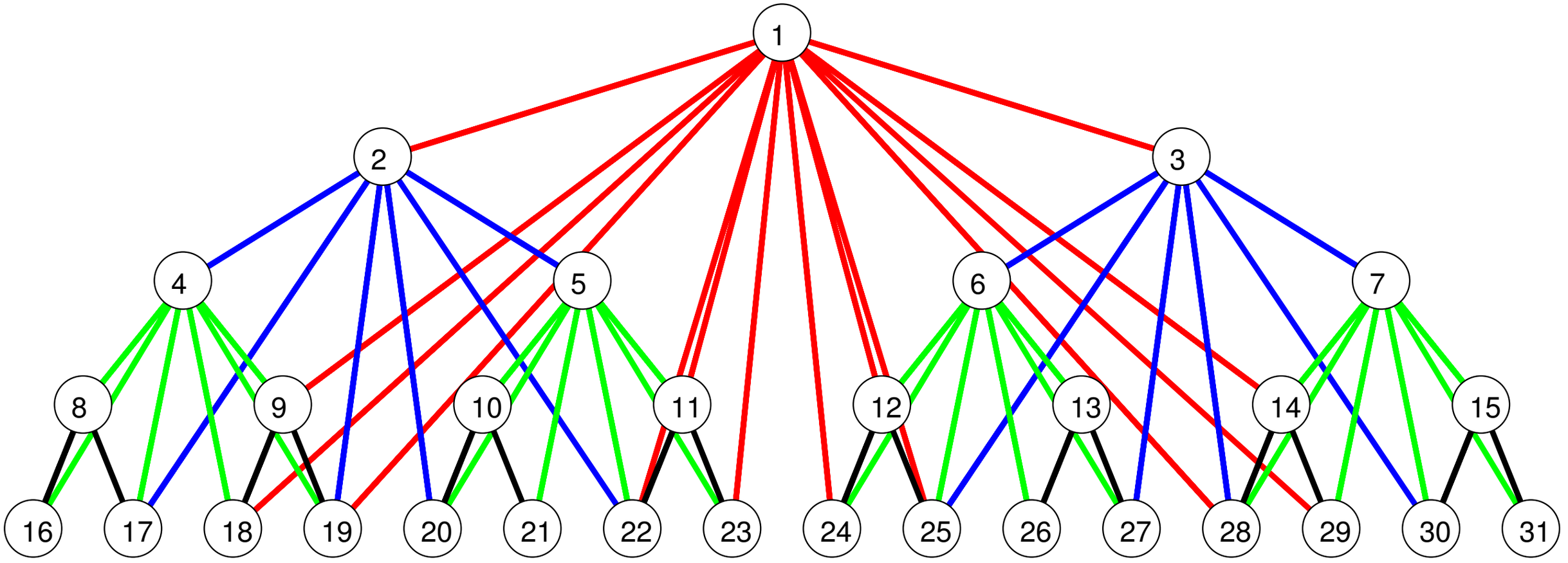
The University of Texas at Austin



(a) (b) (c)

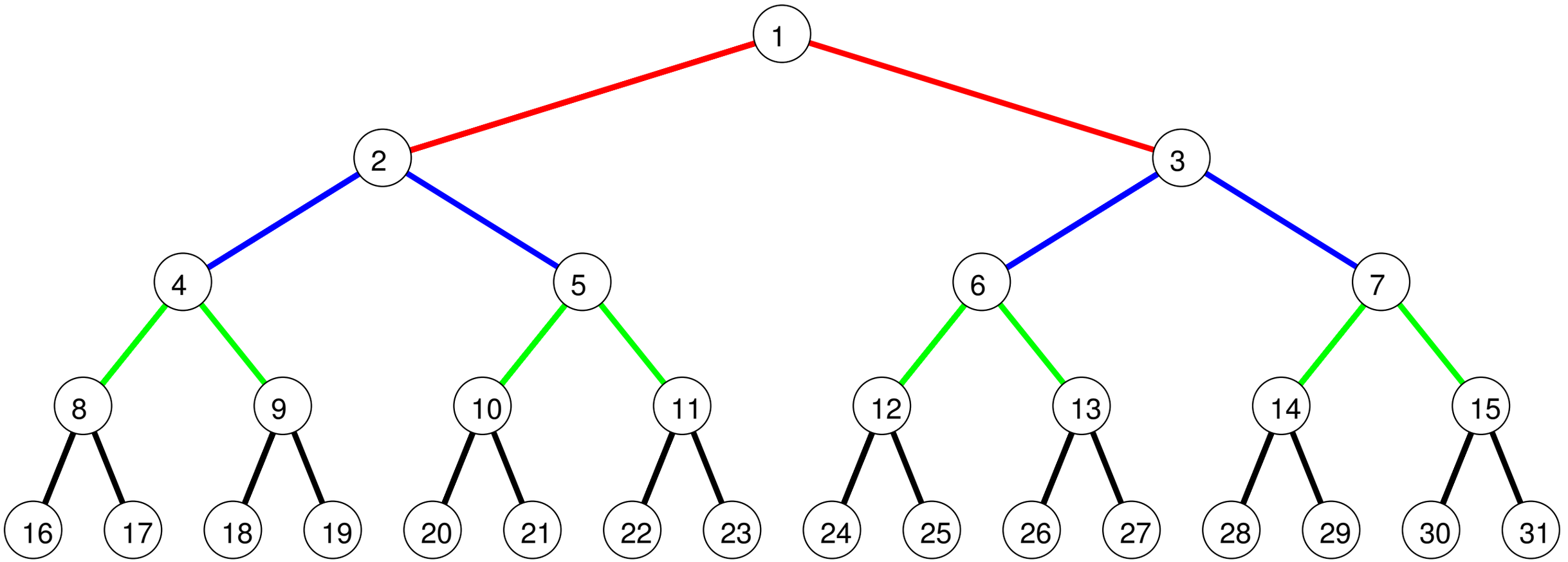
Nested dissection ordering of a uniform grid on a square. The hierarchical tree has 5 levels in this example.

Problem: There is a lot of "communication" between nodes at different levels.

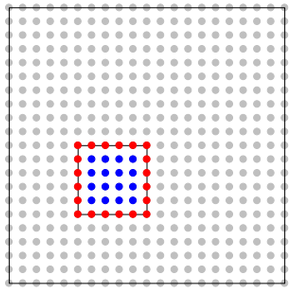


The figure illustrates the dependencies between clusters in executing an LU factorization based on the nested dissection ordering shown in the previous slide. A line in the diagram between two clusters indicates that the nodes in the two clusters are connected. If σ and τ are connected, with $\sigma > \tau$ (so that τ is closer to the root of the tree), then what this means in practice is that after the cluster σ has been processed, then the blocks $\mathbf{L}(I_\tau, I_\sigma)$ *and* $\mathbf{U}(I_\sigma, I_\tau)$ *must be updated. Note that some clusters (e.g. cluster 19) communicate with every single one of their ancestors in the tree.*

It is possible to recast the method slightly to obtain a tree like this:



Any node talks only to its two children, and to its parent.



Let τ denote a leaf. Partition

$$I = I_{\mathrm{i}} \cup I_{\mathrm{b}} \cup I_{\mathrm{e}}$$

standing for interior, boundary, and exterior. In the figure, these index vectors are shown as blue for I_{i}, red for I_{b}, and gray for I_{e}.) The equilibrium equations for box τ read

$$\begin{bmatrix} \mathbf{A}_{\mathrm{ii}} & \mathbf{A}_{\mathrm{ib}} \\ \mathbf{A}_{\mathrm{bi}} & \mathbf{A}_{\mathrm{bb}} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{\mathrm{i}} \\ \mathbf{u}_{\mathrm{e}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{A}_{\mathrm{be}} \end{bmatrix} \mathbf{u}_{\mathrm{e}} = \mathbf{0}. \tag{1}$$

Solve for $\mathbf{u}_{\mathrm{i}}$ to get

$$\mathbf{u}_{\mathrm{i}} = -\mathbf{A}_{\mathrm{ii}}^{-1}\mathbf{A}_{\mathrm{ib}}\mathbf{u}_{\mathrm{b}}.$$

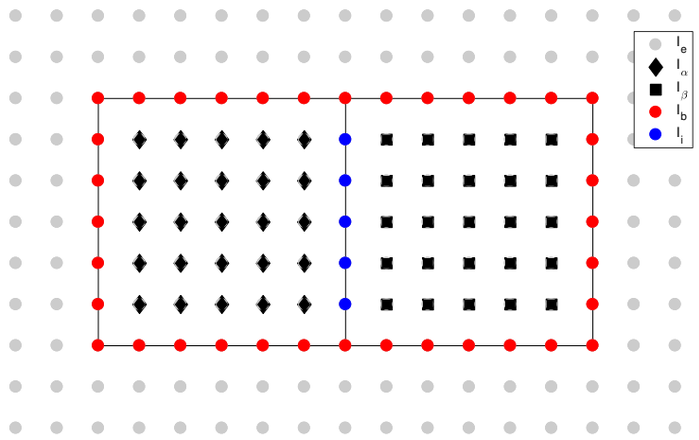
Upon elimination of $\mathbf{u}_{\mathrm{i}}$ we get the new equilibrium equation

$$\left(\mathbf{A}_{\mathrm{bb}} - \mathbf{A}_{\mathrm{bi}}\mathbf{A}_{\mathrm{ii}}^{-1}\mathbf{A}_{\mathrm{ib}}\right)\mathbf{u}_{\mathrm{b}} + \mathbf{A}_{\mathrm{be}}\mathbf{u}_{\mathrm{e}} = \mathbf{0}.$$

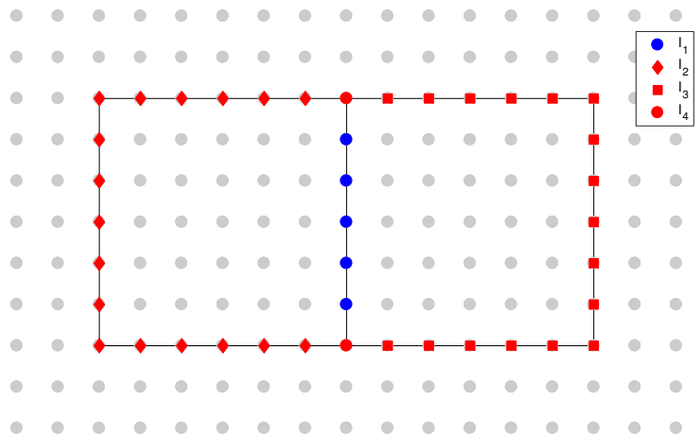
Define

$$\mathbf{S}^{\tau} = -\mathbf{A}_{\mathrm{ii}}^{-1}\mathbf{A}_{\mathrm{ib}}, \tag{2}$$

$$\mathbf{T}^{\tau} = -\mathbf{A}_{\mathrm{bi}}\mathbf{A}_{\mathrm{ii}}^{-1}\mathbf{A}_{\mathrm{ib}}. \tag{3}$$



(a)



(b)

We can use the DtN maps for two leaves to build the DtN map for the parent.

$$\left[\begin{array}{c|c|c|c} \mathbf{A}_{11}+\mathbf{T}_{11}^{\alpha}+\mathbf{T}_{11}^{\beta} & \mathbf{T}_{12}^{\alpha} & \mathbf{T}_{13}^{\beta} & \mathbf{A}_{14} \\ \hline \mathbf{T}_{21}^{\alpha} & \mathbf{A}_{2,2}+\mathbf{T}_{22}^{\alpha} & \mathbf{0} & \mathbf{A}_{24} \\ \hline \mathbf{T}_{31}^{\beta} & \mathbf{0} & \mathbf{A}_{3,3}+\mathbf{T}_{33}^{\beta} & \mathbf{A}_{34} \\ \hline \mathbf{A}_{41} & \mathbf{A}_{42} & \mathbf{A}_{43} & \mathbf{A}_{44} \end{array}\right] \left[\begin{array}{c} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \\ \mathbf{u}_4 \end{array}\right] + \left[\begin{array}{c} \mathbf{0} \\ \mathbf{A}_{2,\mathrm{e}} \\ \mathbf{A}_{3,\mathrm{e}} \\ \mathbf{A}_{4,\mathrm{e}} \end{array}\right] \mathbf{u}_{\mathrm{e}} = \mathbf{0}.$$

Eliminating $\mathbf{u}_1$ we find the reduced equilibrium equation

$$\left(\underbrace{\left[\begin{array}{ccc} \mathbf{A}_{2,2} & \mathbf{0} & \mathbf{A}_{24} \\ \mathbf{0} & \mathbf{A}_{3,3} & \mathbf{A}_{34} \\ \mathbf{A}_{42} & \mathbf{A}_{43} & \mathbf{A}_{44} \end{array}\right]}_{=\mathbf{A}_{\mathrm{bb}}} + \underbrace{\left[\begin{array}{ccc} \mathbf{T}_{22}^{\alpha} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_{33}^{\beta} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] - \left[\begin{array}{c} \mathbf{T}_{21}^{\alpha} \\ \mathbf{T}_{31}^{\beta} \\ \mathbf{A}_{41} \end{array}\right] \left(\mathbf{A}_{1,1}+\mathbf{T}_{11}^{\alpha}+\mathbf{T}_{11}^{\beta}\right)^{-1} \left[\mathbf{T}_{12}^{\alpha} \;\; \mathbf{T}_{13}^{\beta} \;\; \mathbf{A}_{14}\right]}_{=\mathbf{T}_{\mathrm{bb}}} \right) \left[\begin{array}{c} \mathbf{u}_2 \\ \mathbf{u}_3 \\ \mathbf{u}_4 \end{array}\right] + \left[\begin{array}{c} \mathbf{A}_{2\mathrm{e}} \\ \mathbf{A}_{3\mathrm{e}} \\ \mathbf{A}_{4\mathrm{e}} \end{array}\right] \mathbf{u}_{\mathrm{e}} = \mathbf{0}.$$

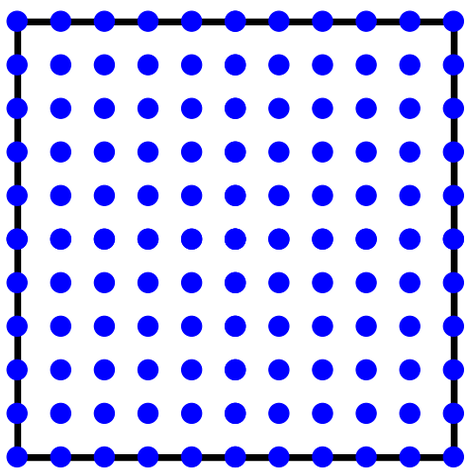
The local solution operator is

$$\mathbf{S}^{\tau} = -\left(\mathbf{A}_{1,1}+\mathbf{T}_{11}^{\alpha}+\mathbf{T}_{11}^{\beta}\right)^{-1}\left[\mathbf{T}_{12}^{\alpha} \;\; \mathbf{T}_{13}^{\beta} \;\; \mathbf{A}_{14}\right].$$

Outline of direct solver

All direct solvers to be described are based on hierarchical domain decomposition.

Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.



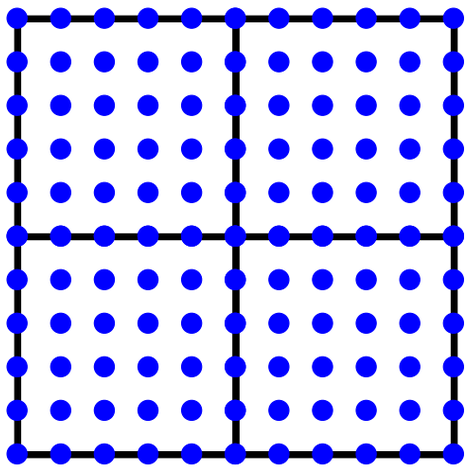
The original grid.

Outline of direct solver

All direct solvers to be described are based on hierarchical domain decomposition.

Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.

Split the domain into “small” patches we call “leaves” (they will be organized in a tree).



The original grid.

Outline of direct solver

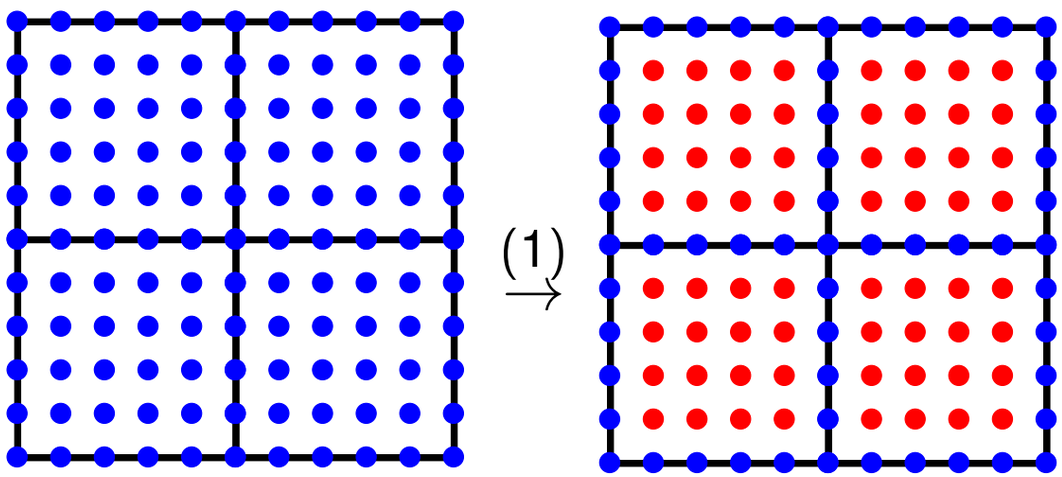
All direct solvers to be described are based on hierarchical domain decomposition.

Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.

Split the domain into "small" patches we call "leaves" (they will be organized in a tree).

On each leaf, compute by "brute force" a local solution operator (e.g. a DtN operator).

This eliminates "internal" grid points from the computation. ("Static condensation.")



The original grid. *Leaves reduced.*

Outline of direct solver

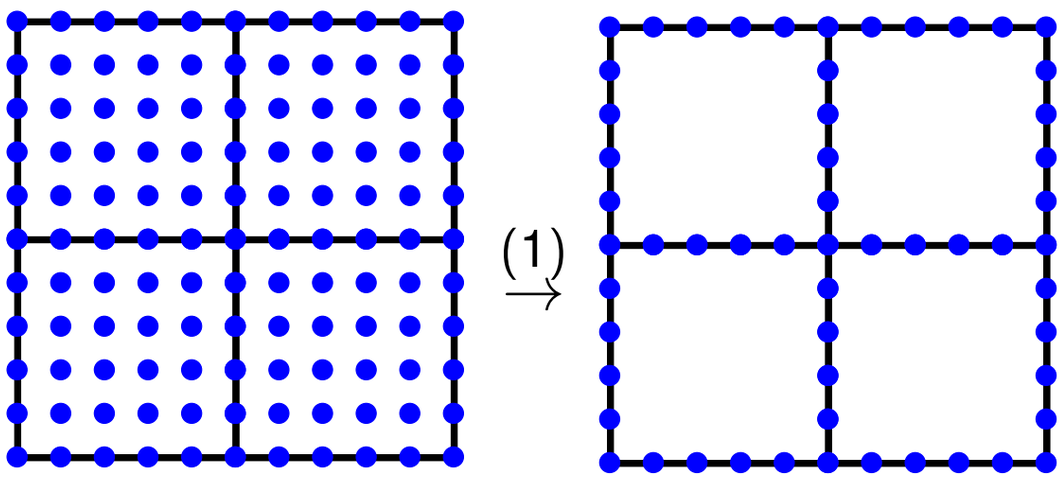
All direct solvers to be described are based on hierarchical domain decomposition.

Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.

Split the domain into "small" patches we call "leaves" (they will be organized in a tree).

On each leaf, compute by "brute force" a local solution operator (e.g. a DtN operator).

This eliminates "internal" grid points from the computation. ("Static condensation.")



The original grid. *Leaves reduced.*

Outline of direct solver

All direct solvers to be described are based on hierarchical domain decomposition.

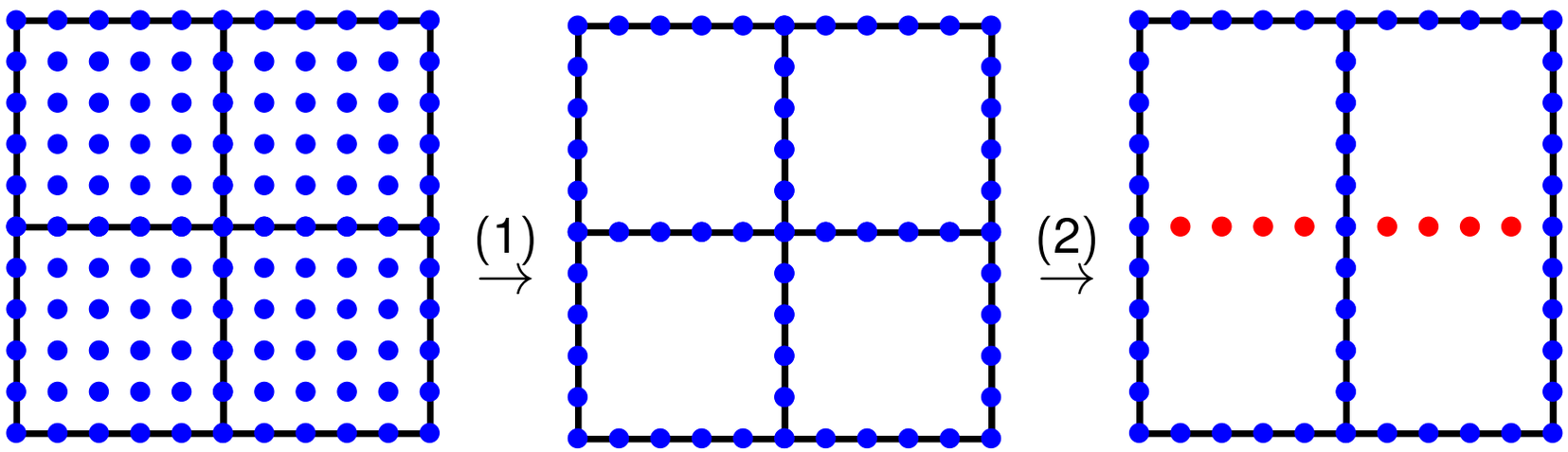
Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.

Split the domain into "small" patches we call "leaves" (they will be organized in a tree).

On each leaf, compute by "brute force" a local solution operator (e.g. a DtN operator).

This eliminates "internal" grid points from the computation. ("Static condensation.")

Merge the leaves in pairs of two.



The original grid. *Leaves reduced.* *After merge.*

Outline of direct solver

All direct solvers to be described are based on hierarchical domain decomposition.

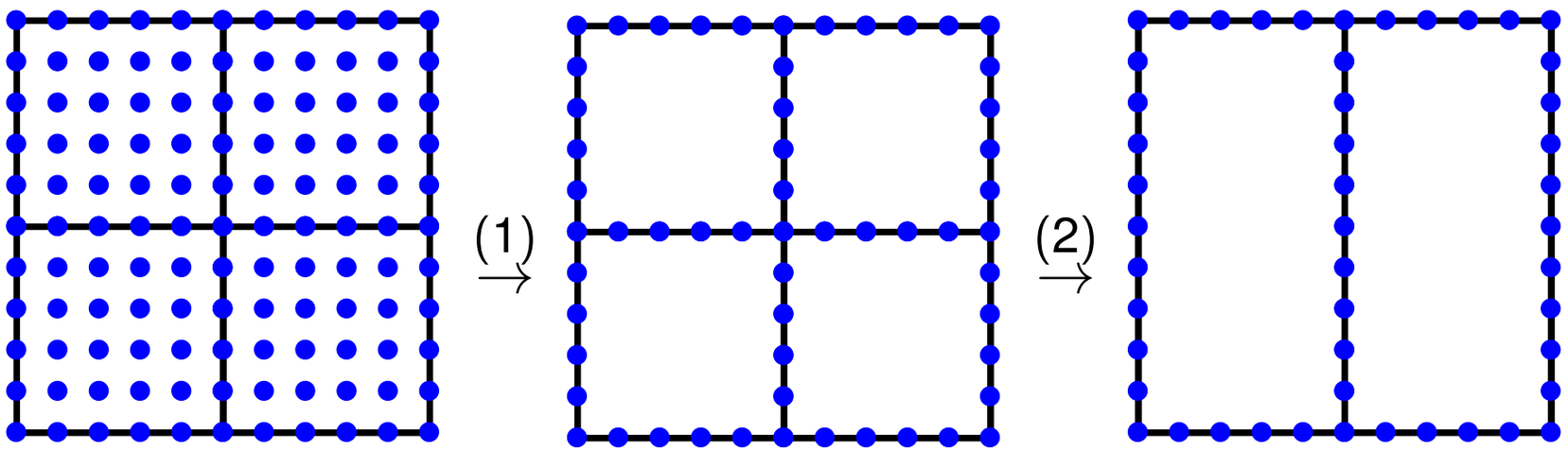
Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.

Split the domain into "small" patches we call "leaves" (they will be organized in a tree).

On each leaf, compute by "brute force" a local solution operator (e.g. a DtN operator).

This eliminates "internal" grid points from the computation. ("Static condensation.")

Merge the leaves in pairs of two. For each pair, compute a local solution operator by combining the solution operators of the two leaves.



The original grid. *Leaves reduced.* *After merge.*

Outline of direct solver

All direct solvers to be described are based on hierarchical domain decomposition.

Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.

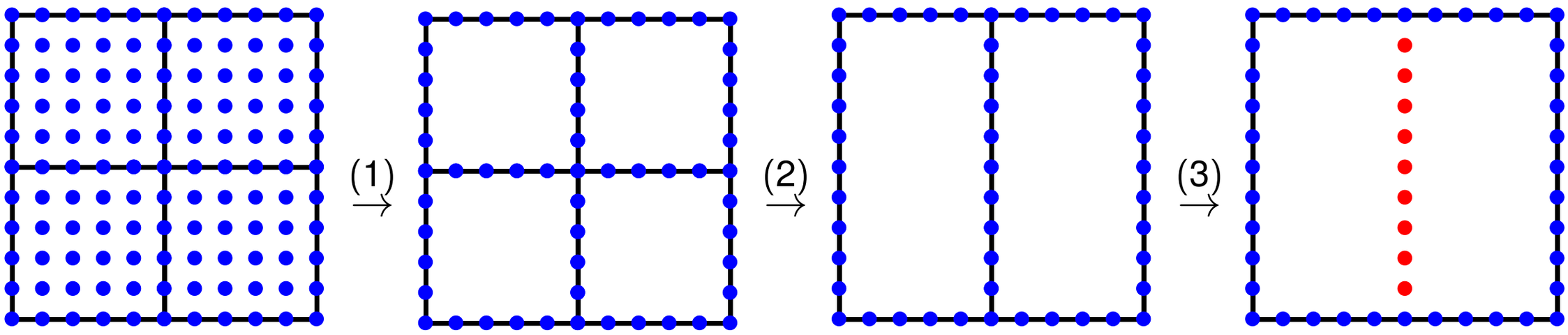
Split the domain into "small" patches we call "leaves" (they will be organized in a tree).

On each leaf, compute by "brute force" a local solution operator (e.g. a DtN operator).

This eliminates "internal" grid points from the computation. ("Static condensation.")

Merge the leaves in pairs of two. For each pair, compute a local solution operator by combining the solution operators of the two leaves.

Continue merging by pairs, organizing the domain in a tree of patches.



The original grid. *Leaves reduced.* *After merge.* *After merge.*

Outline of direct solver

All direct solvers to be described are based on hierarchical domain decomposition.

Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.

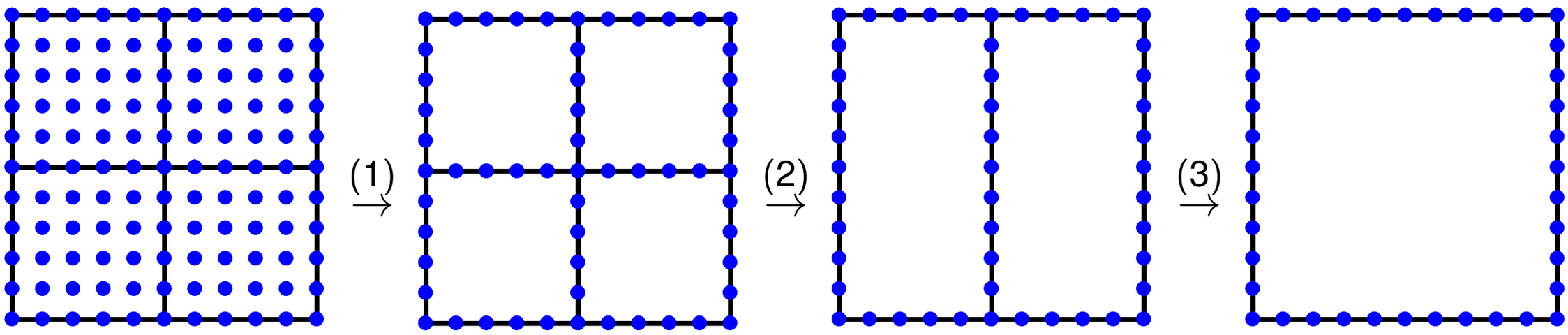
Split the domain into "small" patches we call "leaves" (they will be organized in a tree).

On each leaf, compute by "brute force" a local solution operator (e.g. a DtN operator).

This eliminates "internal" grid points from the computation. ("Static condensation.")

Merge the leaves in pairs of two. For each pair, compute a local solution operator by combining the solution operators of the two leaves.

Continue merging by pairs, organizing the domain in a tree of patches.



The original grid. *Leaves reduced.* *After merge.* *After merge.*

Outline of direct solver

All direct solvers to be described are based on hierarchical domain decomposition.

Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.

Split the domain into "small" patches we call "leaves" (they will be organized in a tree).

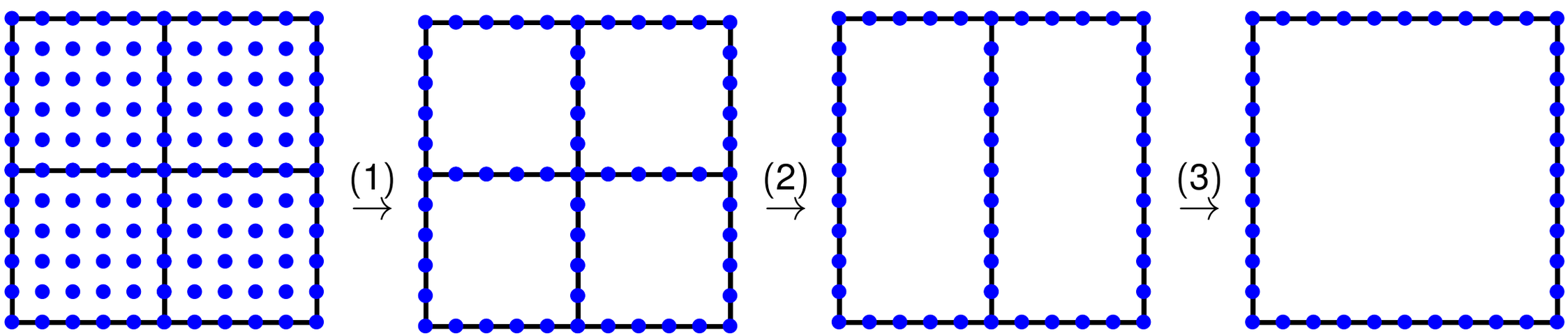
On each leaf, compute by "brute force" a local solution operator (e.g. a DtN operator).

This eliminates "internal" grid points from the computation. ("Static condensation.")

Merge the leaves in pairs of two. For each pair, compute a local solution operator by combining the solution operators of the two leaves.

Continue merging by pairs, organizing the domain in a tree of patches.

When you reach the top level, perform a solve on the reduced problem by brute force.



The original grid. *Leaves reduced.* *After merge.* *After merge.*

Outline of direct solver

All direct solvers to be described are based on hierarchical domain decomposition.

Consider a PDE $Au = f$ defined on a square $\Omega = [0, 1]$. Put a grid on the square.

Split the domain into "small" patches we call "leaves" (they will be organized in a tree).

On each leaf, compute by "brute force" a local solution operator (e.g. a DtN operator).

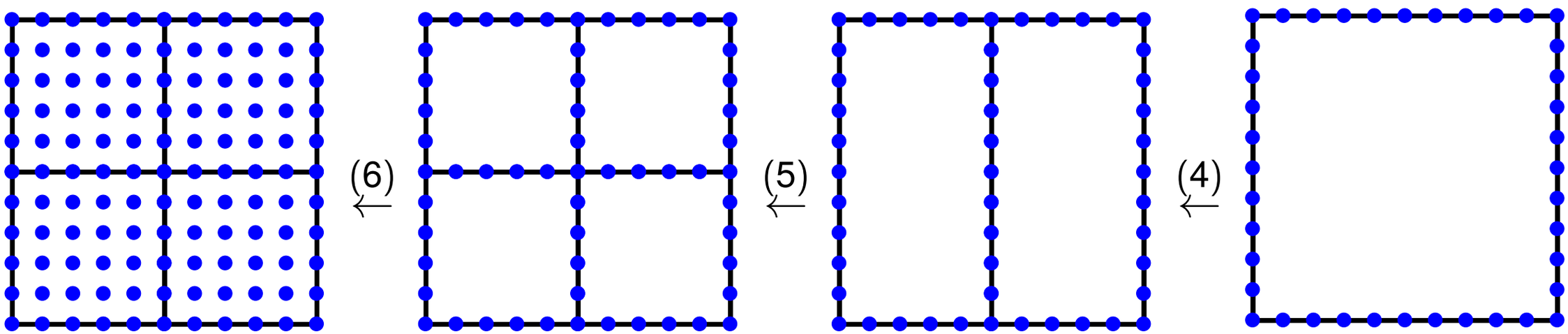
This eliminates "internal" grid points from the computation. ("Static condensation.")

Merge the leaves in pairs of two. For each pair, compute a local solution operator by combining the solution operators of the two leaves.

Continue merging by pairs, organizing the domain in a tree of patches.

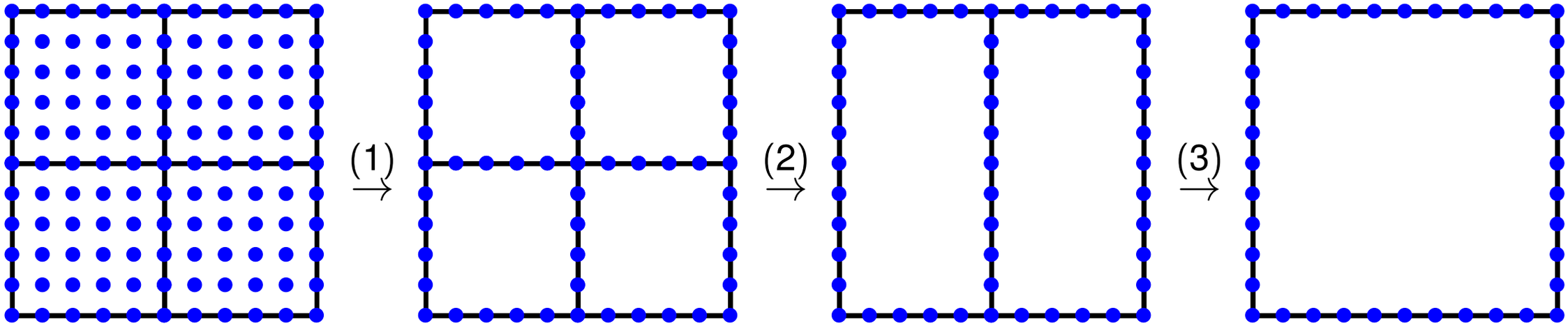
When you reach the top level, perform a solve on the reduced problem by brute force.

Then reconstruct the solution at all internal points via a downwards pass.



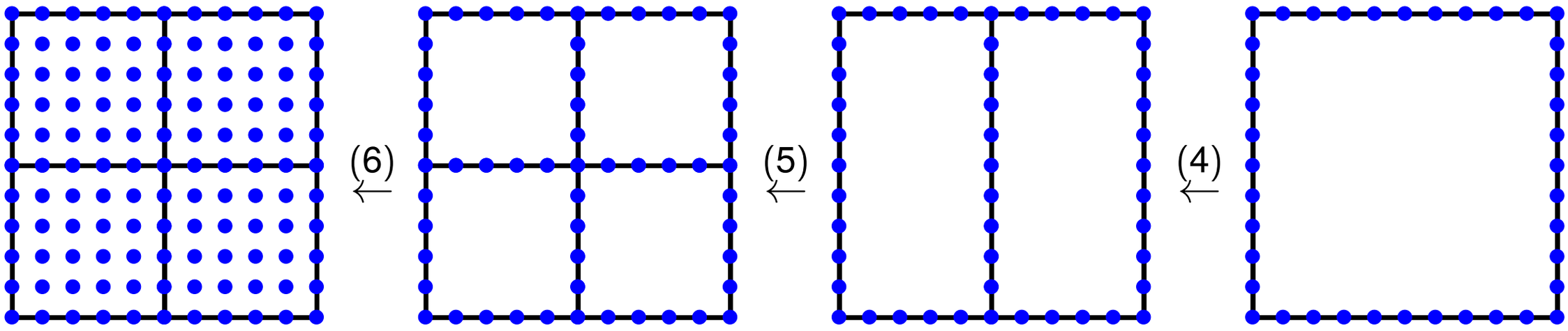
Full solution. *Solve.* *Solve.* *Top level solve.*

Upwards pass — build all solution operators:



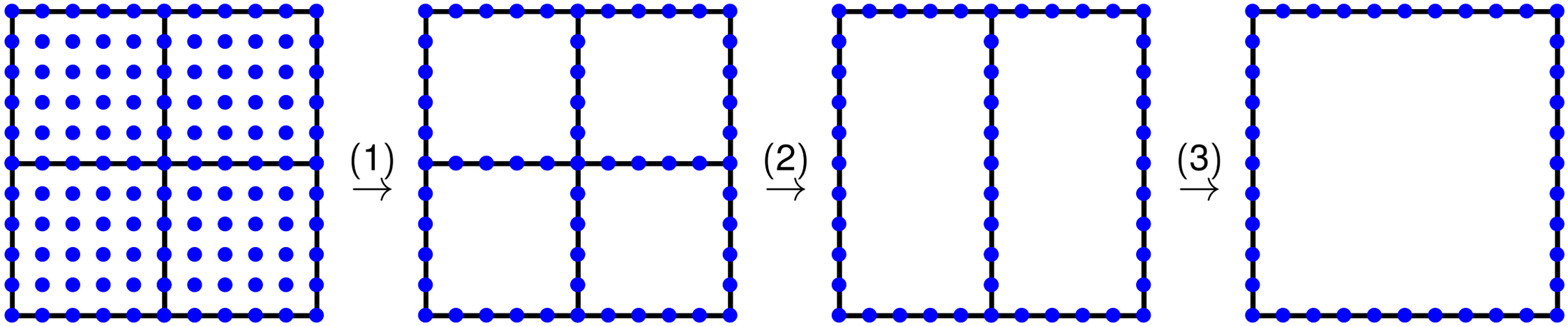
The original grid. *Leaves reduced.* *After merge.* *After merge.*

Downwards pass — solve for a particular data function (*very* fast!):



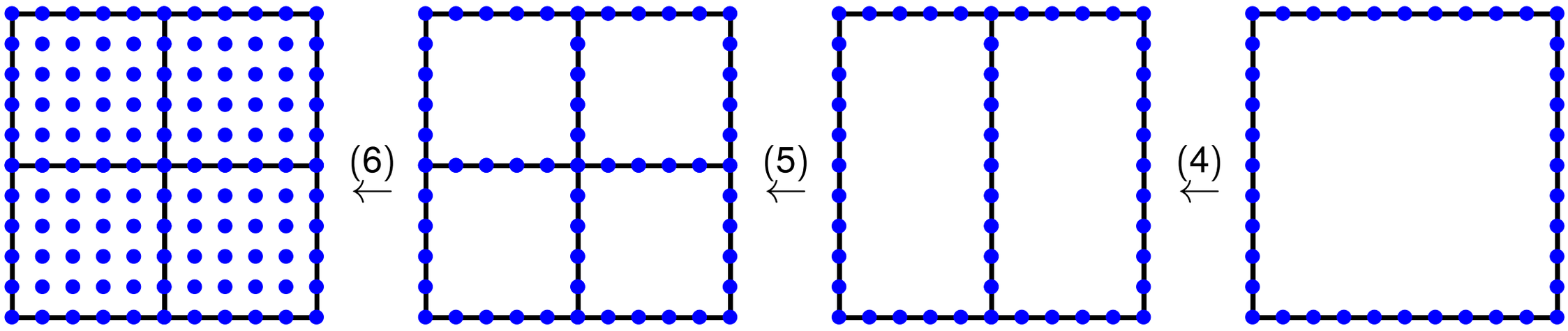
Full solution. *Solve.* *Solve.* *Top level solve.*

Upwards pass — build all solution operators:



The original grid. *Leaves reduced.* *After merge.* *After merge.*

Downwards pass — solve for a particular data function (*very* fast!):



Full solution. *Solve.* *Solve.* *Top level solve.*

Well-established idea: Classical multifrontal / nested dissection method (1973).

Upwards pass — build all solution operators:

D

Alan George

Iain Duff

Tim Davis

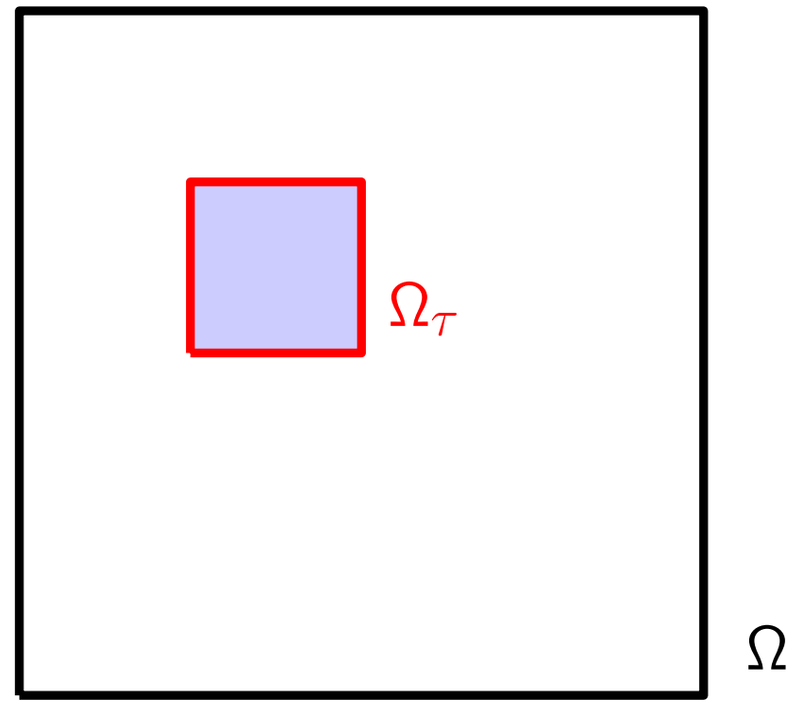
Well-established idea: Classical multifrontal / nested dissection method (1973).

We will next build a direct solver that *explicitly* builds an approximate DtN operator.

Consider a domain Ω on which we are given a PDE

$$(4) \qquad \begin{cases} -\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega \\ u(\boldsymbol{x}) = f(\boldsymbol{x}) & \boldsymbol{x} \in \partial\Omega, \end{cases}$$

where $b = b(\boldsymbol{x})$ is a given function (smooth, non-negative).



Let $\Omega_\tau \subset \Omega$, and suppose that we somehow know the value of u on $\partial\Omega_\tau$

$$f_\tau = u|_{\partial\Omega_\tau}.$$

Now, the restriction of (4) to Ω_τ,

$$(5) \qquad \begin{cases} -\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega_\tau \\ u(\boldsymbol{x}) = f_\tau(\boldsymbol{x}) & \boldsymbol{x} \in \partial\Omega_\tau, \end{cases}$$

has a unique solution, so given f_τ, we can determine $u|_{\Omega_\tau}$, and then also $u_n|_{\partial\Omega_\tau}$.

We define the *Dirichlet-to-Neumann map* T_τ as the unique map

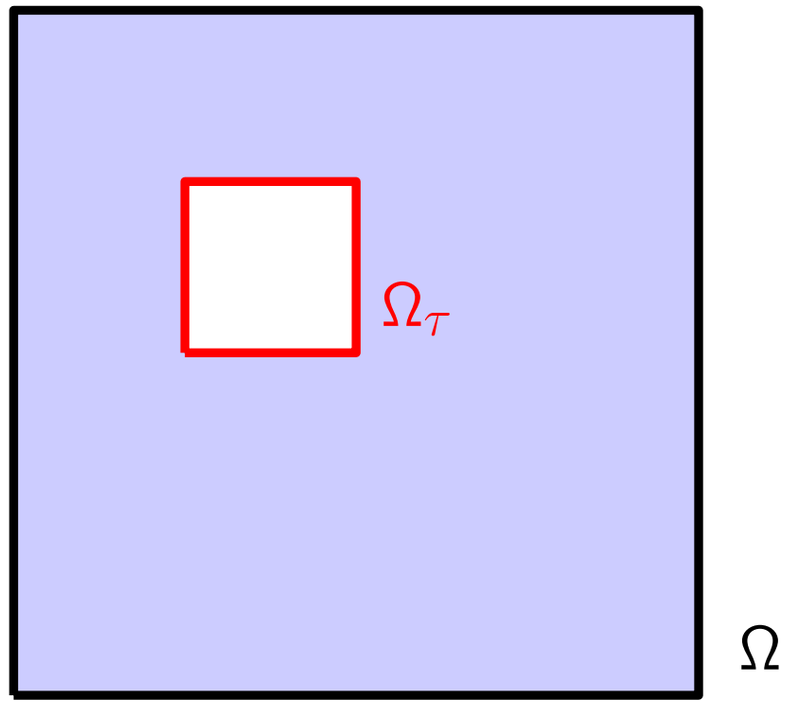
$$T_\tau : \; u|_{\partial\Omega_\tau} \mapsto u_n|_{\partial\Omega_\tau}, \qquad \text{subject to } \; -\Delta u + b\, u = 0 \text{ in } \Omega_\tau.$$

The map T_τ *encodes everything you need to know about* Ω_τ *to solve the PDE on* $\Omega \backslash \Omega_\tau$.

Consider a domain Ω on which we are given a PDE

$$(6) \qquad \begin{cases} -\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega \\ u(\boldsymbol{x}) = f(\boldsymbol{x}) & \boldsymbol{x} \in \partial\Omega, \end{cases}$$

where $b = b(\boldsymbol{x})$ is a given function (smooth, non-negative).



Blue area is $\Omega \backslash \Omega_\tau$.

Suppose we are given T_τ for the sub-domain Ω_τ.

Then in $\Omega \backslash \Omega_\tau$, the solution to the boundary value problem

$$(7) \qquad \begin{cases} -\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega \backslash \Omega_\tau \\ u(\boldsymbol{x}) = f(\boldsymbol{x}) & \boldsymbol{x} \in \partial\Omega, \\ u_n(\boldsymbol{x}) = [T_\tau u](\boldsymbol{x}) & \boldsymbol{x} \in \partial\Omega_\tau, \end{cases}$$

is exactly the same as the solution to (6).

We can split the problem of solving (6) into two parts:

1. Solve the PDE on Ω_τ.
2. Solve the PDE on Ω_τ^{c}.

The DtN map allows us to “glue” the two solutions together.

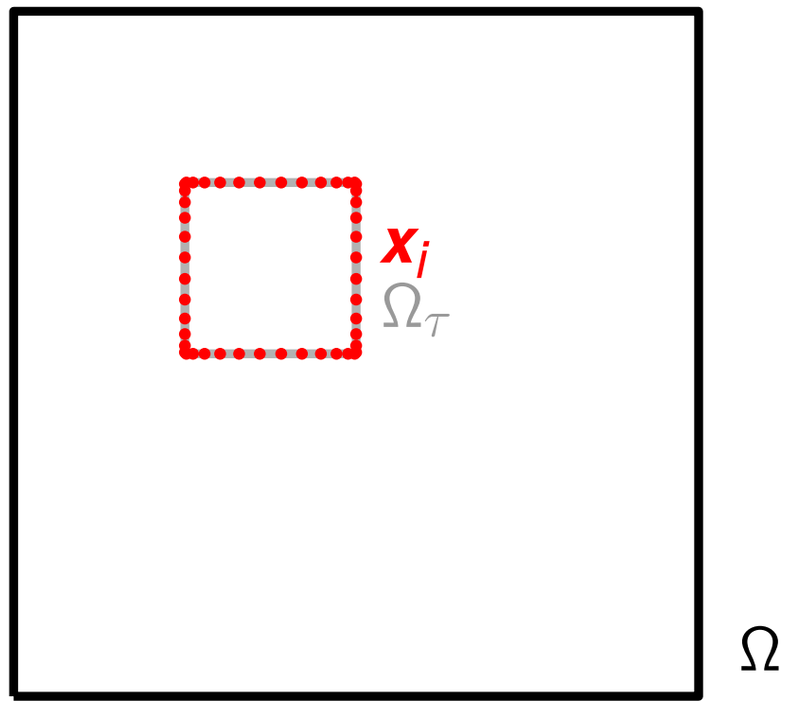
Consider a domain Ω on which we are given a PDE

$$(8) \qquad \begin{cases} -\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega \\ u(\boldsymbol{x}) = f(\boldsymbol{x}) & \boldsymbol{x} \in \partial\Omega, \end{cases}$$

where $b = b(\boldsymbol{x})$ is a given function (smooth, non-negative).

Let T_τ be the DtN map for a subdomain $\Omega_\tau \subset \Omega$,

$$T_\tau : \; u|_{\partial\Omega_\tau} \mapsto u_n|_{\partial\Omega_\tau}, \qquad \text{subject to } \; -\Delta u + b\, u = 0 \text{ in } \Omega_\tau.$$



Representing functions on $\partial\Omega_\tau$ numerically:

- Place p *Legendre nodes* on each side of Ω_τ, to get points $\{\boldsymbol{x}_i\}_{i=1}^{4p}$.
- Represent a function w on $\partial\Omega_\tau$ via the vector $\mathbf{w} \in \mathbb{R}^{4p}$ of *tabulated values* $\mathbf{w}(i) = w(\boldsymbol{x}_i)$.

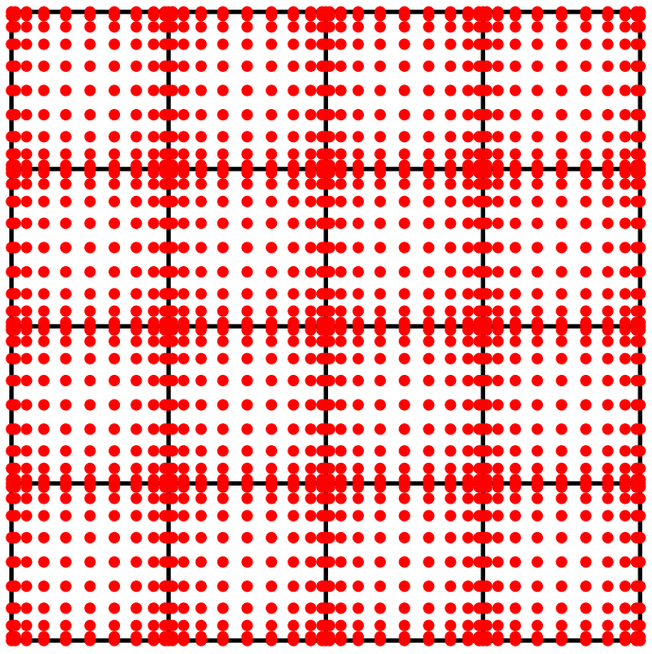
Representing the DtN map T_τ numerically:

- Let $\mathbf{T}_\tau$ denote the $4p \times 4p$ matrix that takes a vector $\mathbf{u}_\tau$ of tabulated Dirichlet data on $\partial\Omega_\tau$ and maps it to the corresponding vector $\mathbf{v}_\tau$ of tabulated Neumann data.

To illustrate the process for building DtN operators supported on Chebyshev nodes, let us consider an elliptic Boundary Value Problem (BVP) of the form

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

where $\Omega = [0,1]^2$ and $\Gamma = \partial\Omega$. The solver relies on a composite spectral grid:

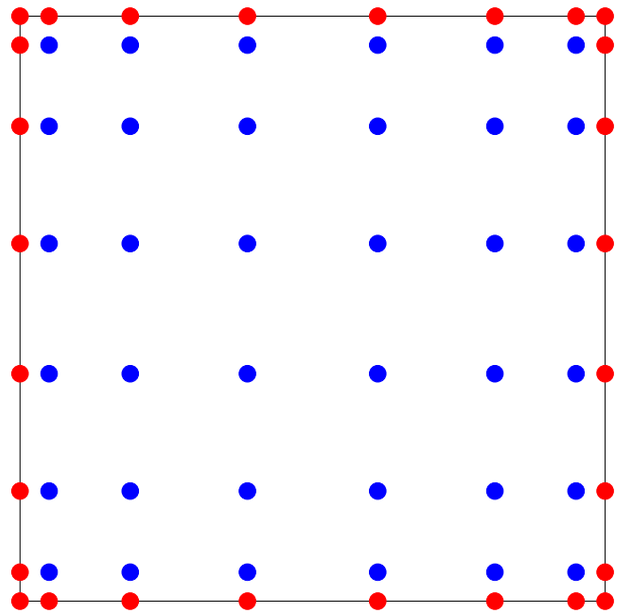


The unknown function u will be represented as a vector holding approximations to its pointwise values at the grid points (collocation). We will perform a local brute-force solve on each small square to build a local "solution operator" in the form of a "Dirichlet-to-Neumann" operator. The global "solution operator" will be built via a hierarchical merge process.

Classical spectral collocation — notation

Recall: Our model problem is $-\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0$ with Dirichlet BC on $\Omega = [0,1]^2$.

Pick an integer p and place a Cartesian mesh of $p \times p$ Chebyshev nodes on Ω.



Let $\{\boldsymbol{x}_j\}_{j=1}^{p^2}$ denote an enumeration of the nodes in the grid.

Partition the index vector $I = \{1, 2, 3, \ldots, p^2\} = I_{\mathrm{i}} \cup I_{\mathrm{e}}$ as follows:

- I_{i} holds the $(p-2)^2$ *interior* nodes (blue dots).
- I_{e} holds the $4p-4$ *exterior* nodes (red dots).

Let $\mathbf{D}^{(1)}$, $\mathbf{D}^{(2)}$, $\mathbf{L}$, denote the $p^2 \times p^2$ matrices approximating $\partial/\partial x_1$, $\partial/\partial x_2$, $-\Delta$, in the spectral sense (i.e. they are exact for tensor products of polynomials of degree $\leq p-1$.)

Classical spectral collocation — solving a Dirichlet problem

Recall: Our model problem is $-\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0$ with Dirichlet BC on a square $\Omega = [0,1]^2$. The domain is discretized using a $p \times p$ tensor product grid of Chebyshev nodes split into I_{e} *exterior nodes* and I_{i} *interior nodes*.

Let $\mathbf{B}$ denote the diagonal matrix with entries $[b(\boldsymbol{x}j)]_{j=1}^{p^2}$ and let $\mathbf{L}$ be the spectral Laplacian. Then $\mathbf{A} = \mathbf{L} + \mathbf{B}$ is our spectral approximation of the differential operator.

Let $\mathbf{u} \in \mathbb{R}^{p^2}$ denote a vector of approximate values of the solution u, $\mathbf{u}(j) \approx u(\mathbf{x}_j)$.

For the ***exterior nodes***, simply set $\mathbf{u}$ equal to the Dirichlet data: $\mathbf{u}(j) = f(\mathbf{x}_j)$ for $j \in I_{\mathrm{e}}$.

For ***interior nodes***, enforce the PDE via collocation: $\mathbf{A}(j,:)\mathbf{u} = 0$ for $j \in I_{\mathrm{i}}$.

Set $\mathbf{u}_{\mathrm{i}} = \mathbf{u}(I_{\mathrm{i}})$, $\mathbf{u}_{\mathrm{e}} = \mathbf{u}(I_{\mathrm{e}})$, $\mathbf{A}_{\mathrm{i,i}} = \mathbf{A}(I_{\mathrm{i}}, I_{\mathrm{i}})$, and $\mathbf{A}_{\mathrm{i,e}} = \mathbf{A}(I_{\mathrm{i}}, I_{\mathrm{e}})$.
Then the collocation condition can be written

$$\mathbf{A}_{\mathrm{i,i}}\, \mathbf{u}_{\mathrm{i}} + \mathbf{A}_{\mathrm{i,e}}\, \mathbf{u}_{\mathrm{e}} = \mathbf{0}.$$

Solving for $\mathbf{u}_{\mathrm{i}}$ we find the solution process:

$$\mathbf{u}_{\mathrm{e}} = \mathbf{f}_{\mathrm{e}} = [f(\boldsymbol{x}_j)]_{j \in I_{\mathrm{e}}}$$
$$\mathbf{u}_{\mathrm{i}} = -\mathbf{A}_{\mathrm{i,i}}^{-1}\, \mathbf{A}_{\mathrm{i,e}}\, \mathbf{f}_{\mathrm{e}}.$$

Classical spectral collocation — build the Dirichlet-to-Neumman (DtN) map

Recall: Our model problem is $-\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0$ with Dirichlet BC on a square $\Omega = [0,1]^2$. The domain is discretized using a $p \times p$ tensor product grid of Chebyshev nodes split into I_{e} *exterior nodes* and I_{i} *interior nodes*.

At this point, we have constructed a linear map from Dirichlet data $\mathbf{f}_{\mathrm{e}}$ to the full solution vector $\mathbf{u}$ via:

1. For exterior nodes, set the potential to equal the given Dirichlet data

$$\mathbf{u}_{\mathrm{e}} = [f(\mathbf{x}_j)]_{j \in I_{\mathrm{e}}} = \mathbf{f}_{\mathrm{e}}.$$

2. For the interior nodes, enforce the PDE via spectral collocation,

$$\mathbf{A}_{\mathrm{i,i}}\mathbf{u}_{\mathrm{i}} + \mathbf{A}_{\mathrm{i,e}}\mathbf{u}_{\mathrm{e}} = \mathbf{0}.$$

   Solving for $\mathbf{u}_{\mathrm{i}}$, we find $\mathbf{u}_{\mathrm{i}} = -\mathbf{A}_{\mathrm{i,i}}^{-1}\,\mathbf{A}_{\mathrm{i,e}}\,\mathbf{f}_{\mathrm{e}}$.

New objective: We seek to build the *Dirichlet-to-Neumann (DtN) map* that maps given Dirichlet data to the corresponding boundary fluxes. This map acts as a local solution operator that encodes all information about the box that we need to solve the global problem.

Classical spectral collocation — build the Dirichlet-to-Neumman (DtN) map

Recall: Our model problem is $-\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0$ with Dirichlet BC on a square $\Omega = [0,1]^2$. The domain is discretized using a $p \times p$ tensor product grid of Chebyshev nodes split into I_{e} *exterior nodes* and I_{i} *interior nodes*.

At this point, we have constructed a linear map from Dirichlet data $\mathbf{f}_{\mathrm{e}}$ to the full solution vector $\mathbf{u}$ via:

1. For exterior nodes, set the potential to equal the given Dirichlet data

$$\mathbf{u}_{\mathrm{e}} = [f(\mathbf{x}_j)]_{j \in I_{\mathrm{e}}} = \mathbf{f}_{\mathrm{e}}.$$

2. For the interior nodes, enforce the PDE via spectral collocation,

$$\mathbf{A}_{\mathrm{i,i}}\mathbf{u}_{\mathrm{i}} + \mathbf{A}_{\mathrm{i,e}}\mathbf{u}_{\mathrm{e}} = \mathbf{0}.$$

   Solving for $\mathbf{u}_{\mathrm{i}}$, we find $\mathbf{u}_{\mathrm{i}} = -\mathbf{A}_{\mathrm{i,i}}^{-1}\,\mathbf{A}_{\mathrm{i,e}}\,\mathbf{f}_{\mathrm{e}}$.

New objective: We seek to build the *Dirichlet-to-Neumann (DtN) map* that maps given Dirichlet data to the corresponding boundary fluxes. This map acts as a local solution operator that encodes all information about the box that we need to solve the global problem.

Solution: Simply apply spectral differentiation to the constructed solution $\mathbf{u} = [\mathbf{u}_{\mathrm{i}},\ \mathbf{u}_{\mathrm{e}}]$.

Classical spectral collocation — build the Dirichlet-to-Neumman (DtN) map

Recall: Our model problem is $-\Delta u(\boldsymbol{x}) + b(\boldsymbol{x})\, u(\boldsymbol{x}) = 0$ with Dirichlet BC on a square $\Omega = [0,1]^2$. The domain is discretized using a $p \times p$ tensor product grid of Chebyshev nodes split into I_{e} *exterior nodes* and I_{i} *interior nodes*.

At this point, we have constructed a linear map from Dirichlet data $\mathbf{f}_{\mathrm{e}}$ to the full solution vector $\mathbf{u}$ via:

1. For exterior nodes, set the potential to equal the given Dirichlet data

$$\mathbf{u}_{\mathrm{e}} = [f(\mathbf{x}_j)]_{j \in I_{\mathrm{e}}} = \mathbf{f}_{\mathrm{e}}.$$

2. For the interior nodes, enforce the PDE via spectral collocation,

$$\mathbf{A}_{\mathrm{i,i}}\mathbf{u}_{\mathrm{i}} + \mathbf{A}_{\mathrm{i,e}}\mathbf{u}_{\mathrm{e}} = \mathbf{0}.$$

   Solving for $\mathbf{u}_{\mathrm{i}}$, we find $\mathbf{u}_{\mathrm{i}} = -\mathbf{A}_{\mathrm{i,i}}^{-1}\,\mathbf{A}_{\mathrm{i,e}}\,\mathbf{f}_{\mathrm{e}}$.

3. Now that $\mathbf{u}$ is known at *all* nodes, apply the spectral differentiation matrices $\mathbf{D}^{(1)}$ or $\mathbf{D}^{(2)}$ to compute the boundary fluxes. (Corners get special treatment.)

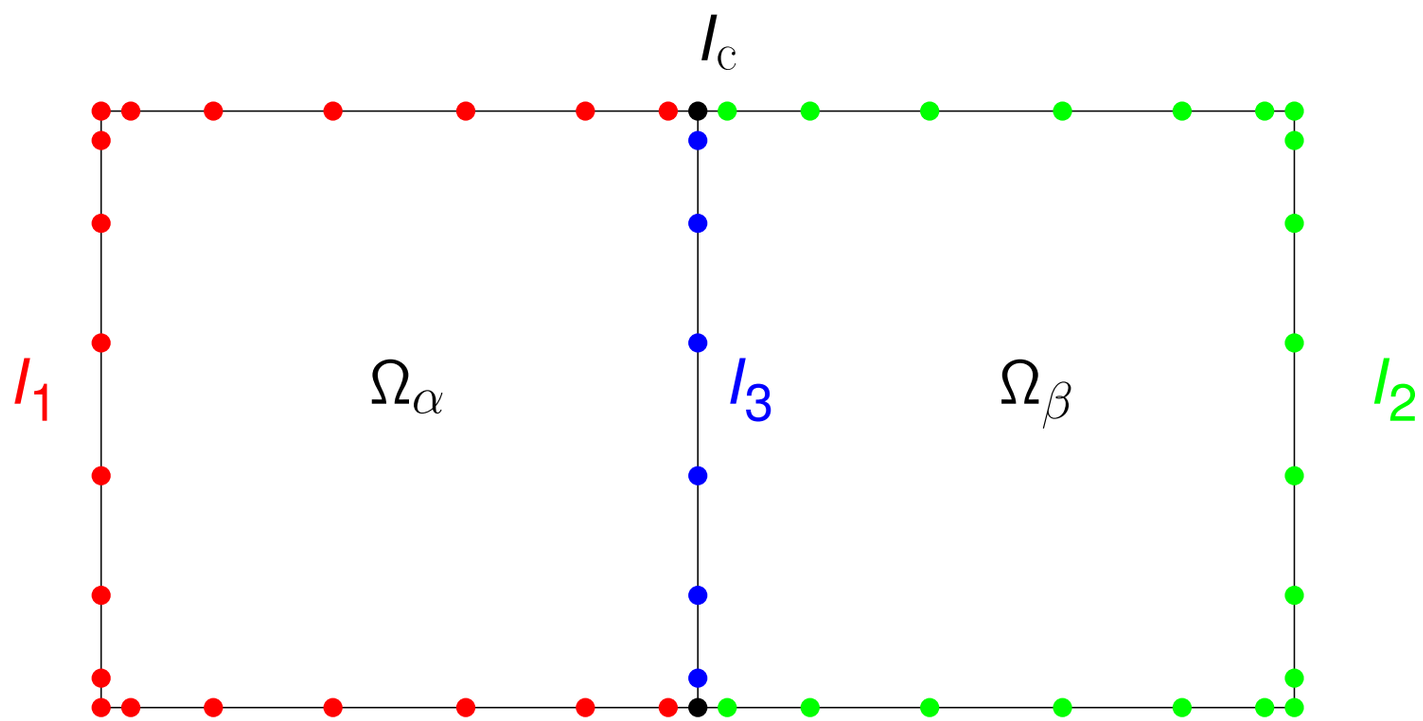
Merging two DtN operators

Question: Given DtN matrices of two boxes, how form the DtN matrix of the union box?

Merging two DtN operators

Question: Given DtN matrices of two boxes, how form the DtN matrix of the union box?

Answer: Let the rectangular domain Ω be formed by two squares Ω_α and Ω_β. The sets I_1, I_2, and I_3 form the exterior nodes, while I_4 consists of the interior nodes.



Let $\mathbf{v}_j$ denote the boundary fluxes on side j, and let $\mathbf{u}_j$ denote the potential. Then from the left and the right DtN maps we get the equilibrium equations

$$\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{T}^{\alpha}_{1,1} & \mathbf{T}^{\alpha}_{1,3} \\ \mathbf{T}^{\alpha}_{3,1} & \mathbf{T}^{\alpha}_{3,3} \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_3 \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} \mathbf{v}_2 \\ \mathbf{v}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{T}^{\beta}_{2,2} & \mathbf{T}^{\beta}_{2,3} \\ \mathbf{T}^{\beta}_{3,2} & \mathbf{T}^{\beta}_{3,3} \end{bmatrix} \begin{bmatrix} \mathbf{u}_2 \\ \mathbf{u}_3 \end{bmatrix}.$$

Collating the two equilibrium equations (by eliminating $\mathbf{v}_3$) we get

$$\left[\begin{array}{c|c|c} \mathbf{T}^{\alpha}_{1,1} & \mathbf{0} & \mathbf{T}^{\alpha}_{1,3} \\ \hline \mathbf{0} & \mathbf{T}^{\beta}_{2,2} & \mathbf{T}^{\beta}_{2,3} \\ \hline \mathbf{T}^{\alpha}_{3,1} & -\mathbf{T}^{\beta}_{3,2} & \mathbf{T}^{\alpha}_{3,3} - \mathbf{T}^{\beta}_{3,3} \end{array}\right] \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{0} \end{bmatrix}.$$

Eliminate $\mathbf{u}_3$ to find the desired map

$$\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \mathbf{T}^{\tau} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix}$$

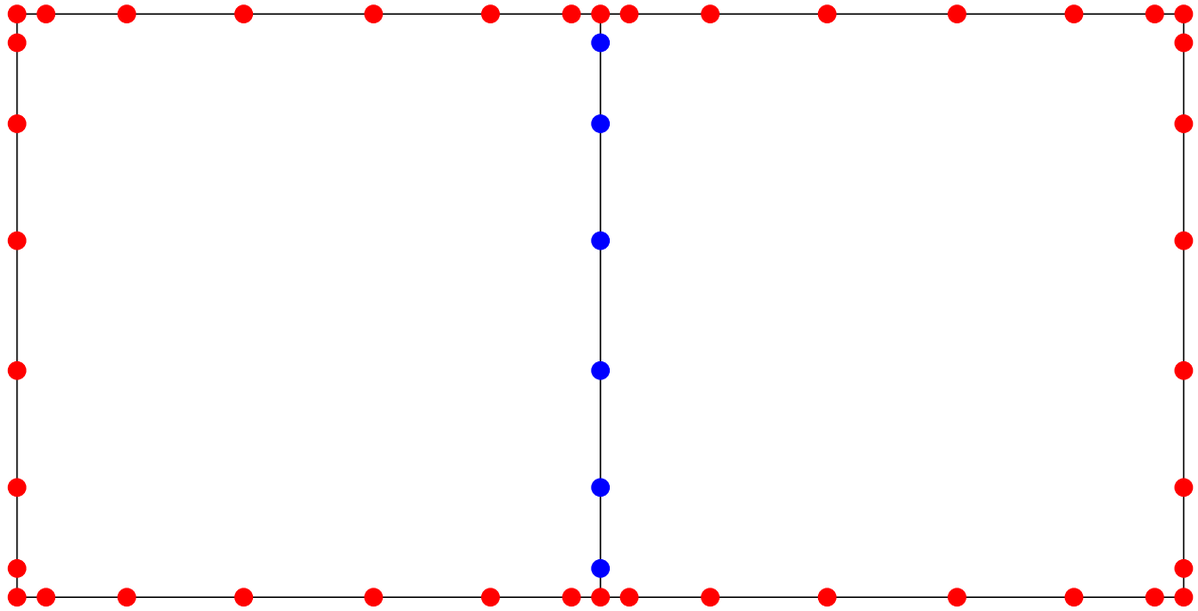
where

$$\mathbf{T}^{\tau} = \left[\begin{array}{c|c} \mathbf{T}^{\alpha}_{1,1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}^{\beta}_{2,2} \end{array}\right] - \begin{bmatrix} \mathbf{T}^{\alpha}_{1,3} \\ \mathbf{T}^{\beta}_{2,3} \end{bmatrix} \left(\mathbf{T}^{\alpha}_{3,3} - \mathbf{T}^{\beta}_{3,3}\right)^{-1} \left[\begin{array}{c|c} \frac{1}{2}\mathbf{T}^{\alpha}_{3,1} & -\frac{1}{2}\mathbf{T}^{\beta}_{3,2} \end{array}\right].$$

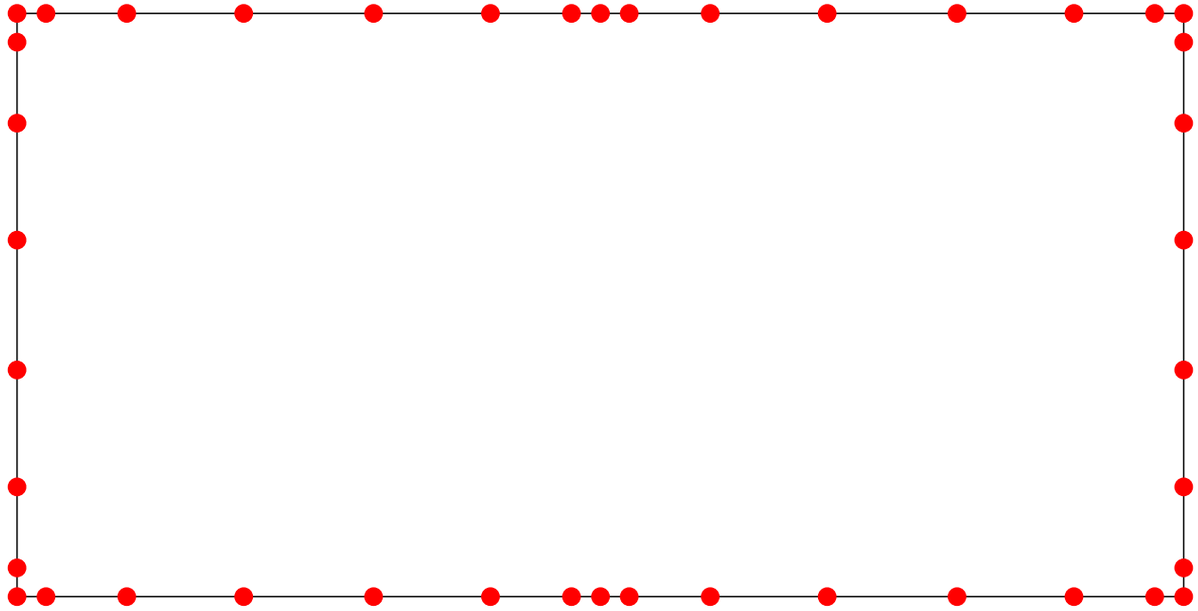
(We skipped a step — the corner nodes are eliminated by re-interpolating to *Legendre* nodes on the boundaries.)

Illustration of the merge operation

Before elimination of interior (blue) nodes:



After elimination of interior nodes:

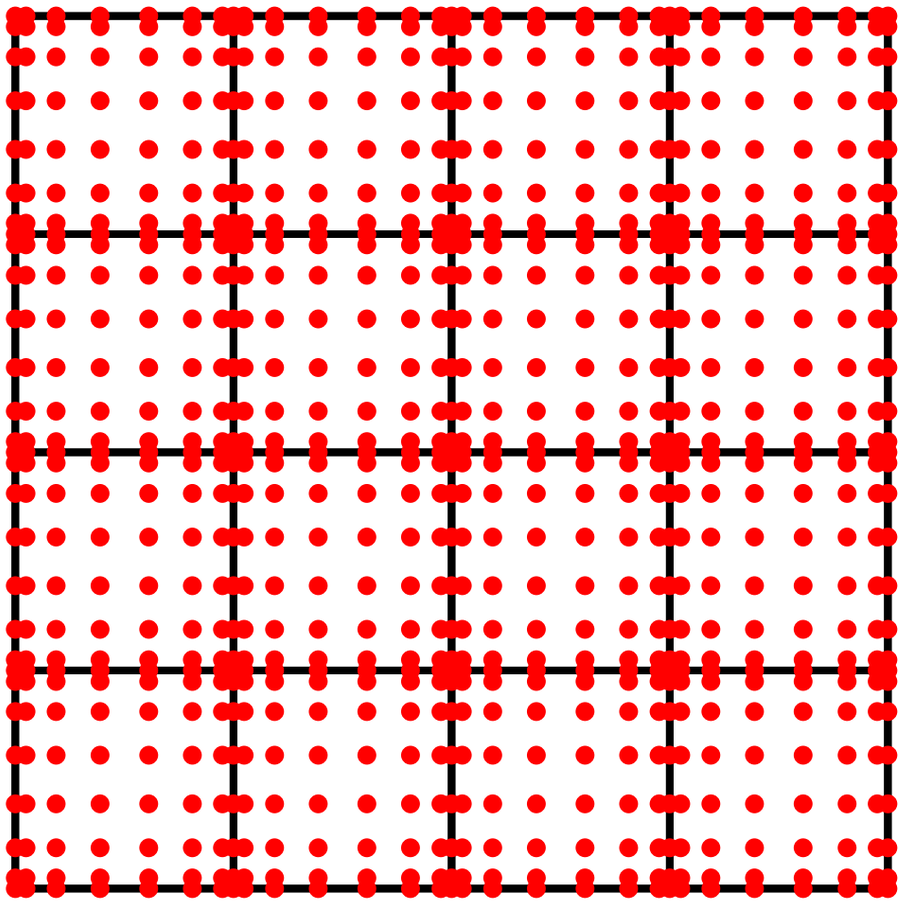


Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Pre-process: Put down a spectral composite grid on Ω (Chebyshev nodes):



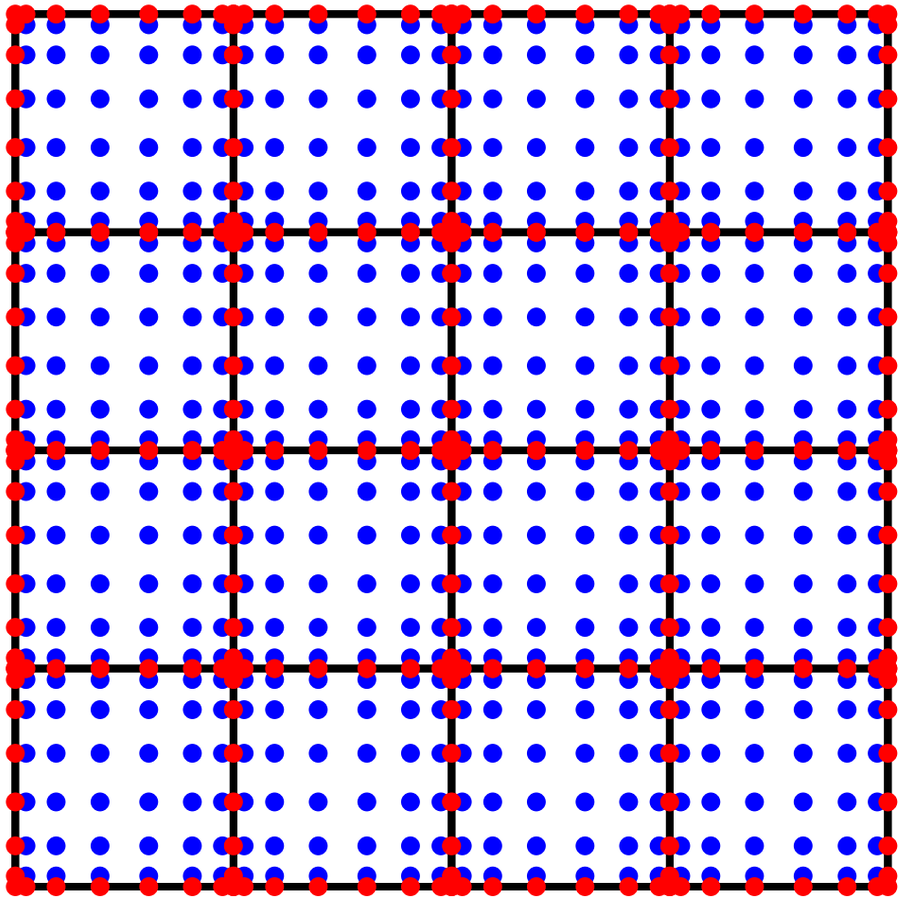
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Process leaves: Eliminate the interior (blue) nodes.

Technically, we compute the Dirichlet-to-Neumann operator via a local spectral computation.



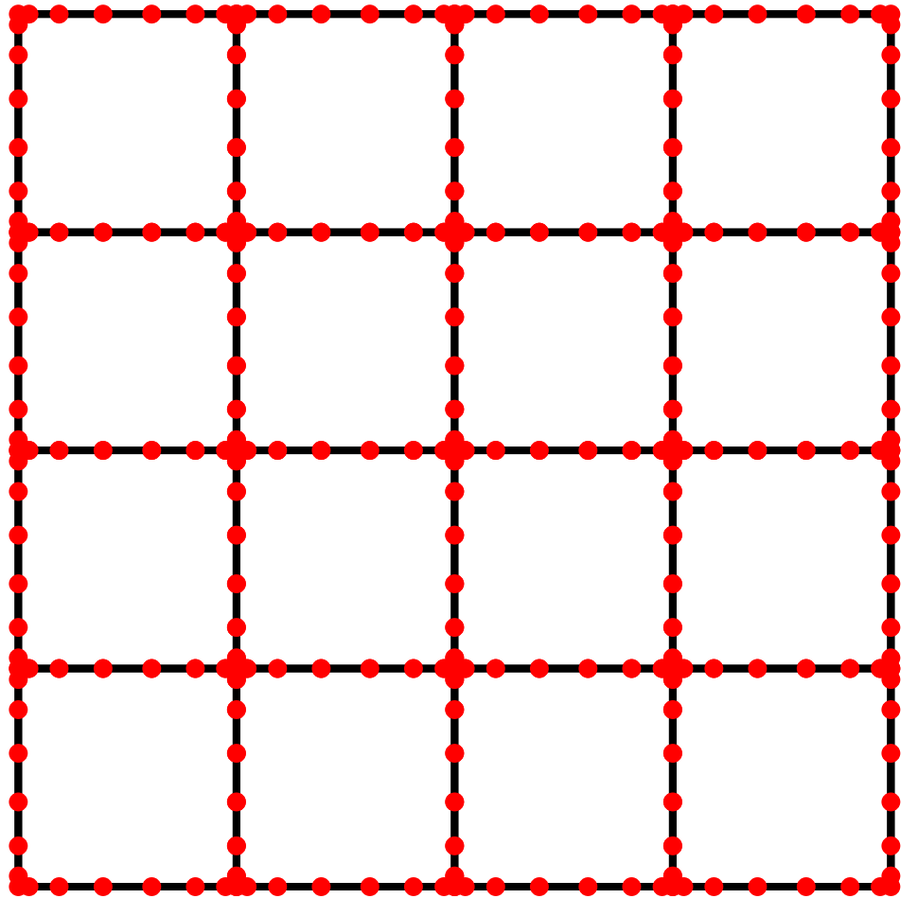
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Process leaves: Eliminate the interior (blue) nodes.

Technically, we compute the Dirichlet-to-Neumann operator via a local spectral computation.

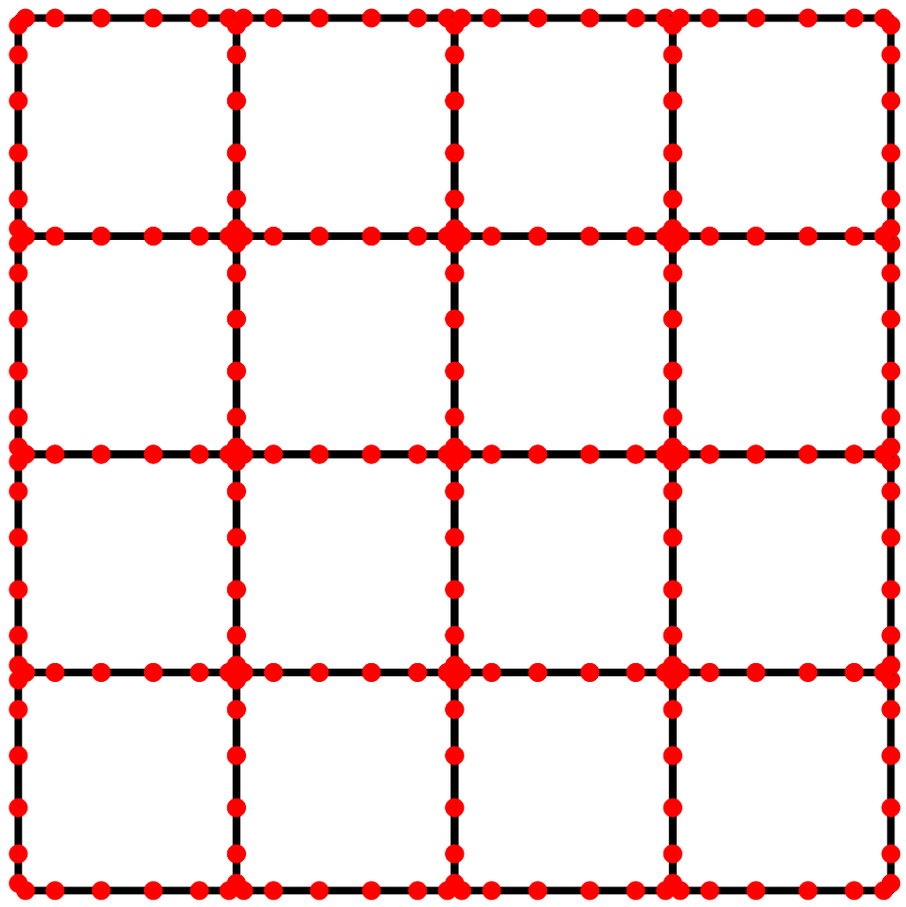


Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Process leaves: Retabulate from Chebyshev to *Legendre nodes* on boundaries.



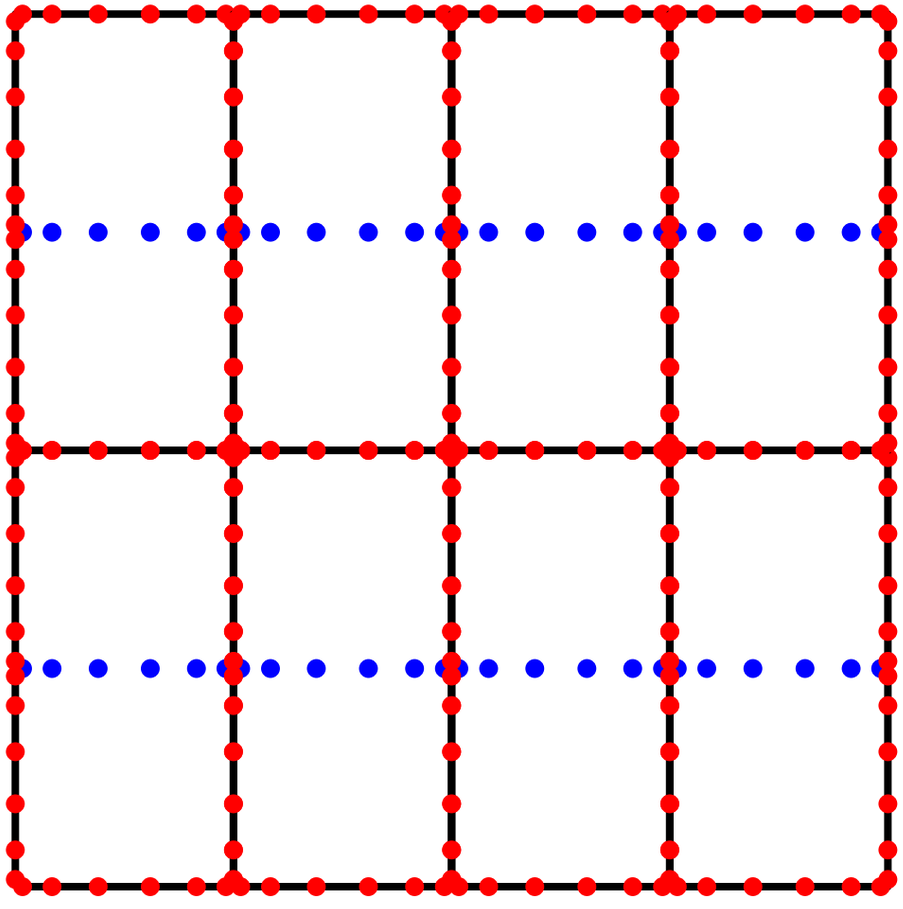
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Upwards sweep: Merge boxes by pairs and eliminate the interior (blue) nodes.

To do this, use the computed DtN operators to enforce continuity of u and du/dn across interior boundaries. Compute the DtN operator for the larger box.



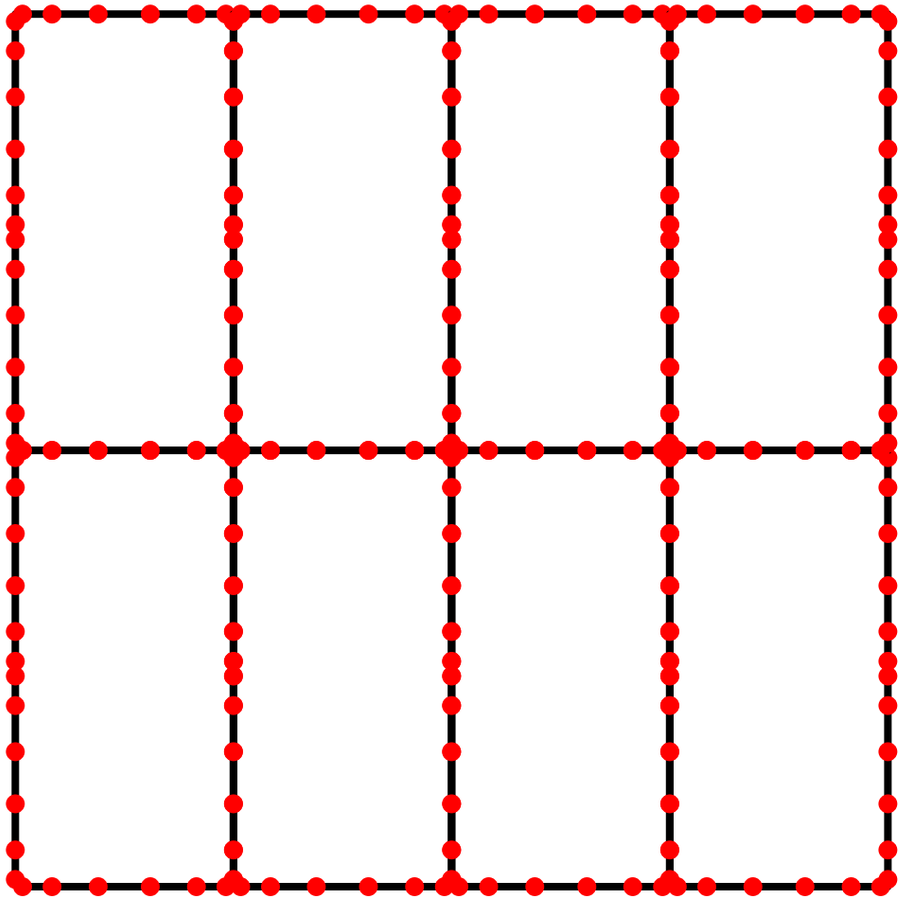
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Upwards sweep: Merge boxes by pairs and eliminate the interior (blue) nodes.

To do this, use the computed DtN operators to enforce continuity of u and du/dn across interior boundaries. Compute the DtN operator for the larger box.



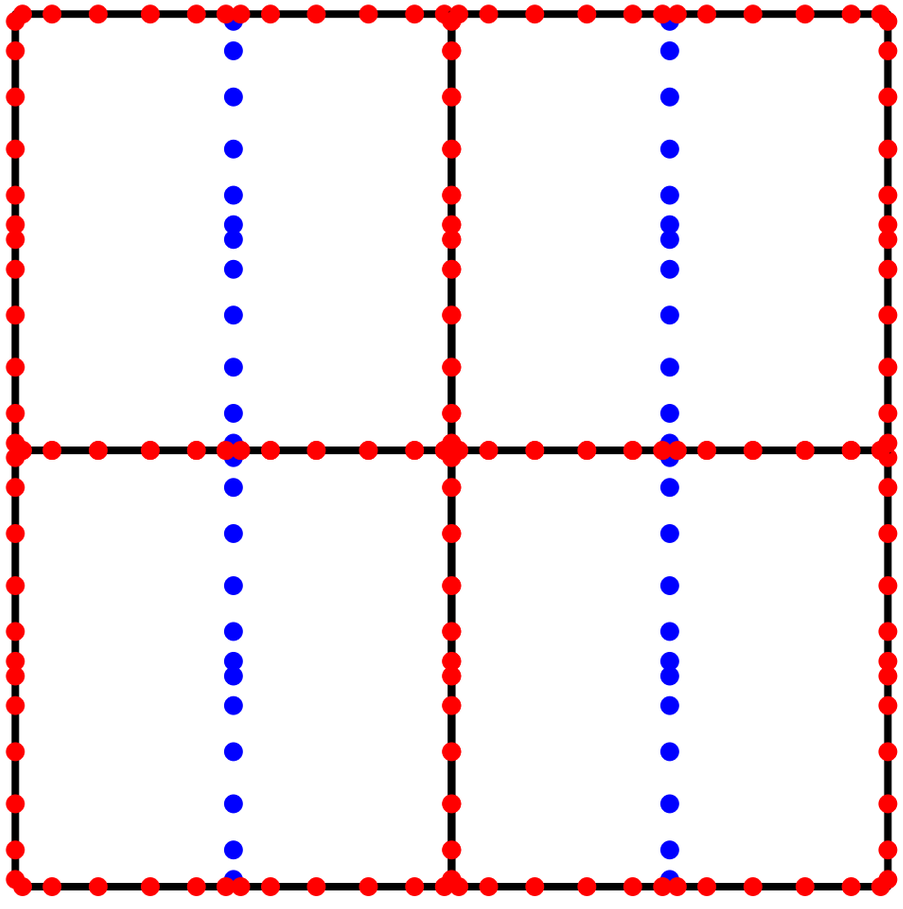
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Upwards sweep: Merge boxes by pairs and eliminate the interior (blue) nodes.

To do this, use the computed DtN operators to enforce continuity of u and du/dn across interior boundaries. Compute the DtN operator for the larger box.



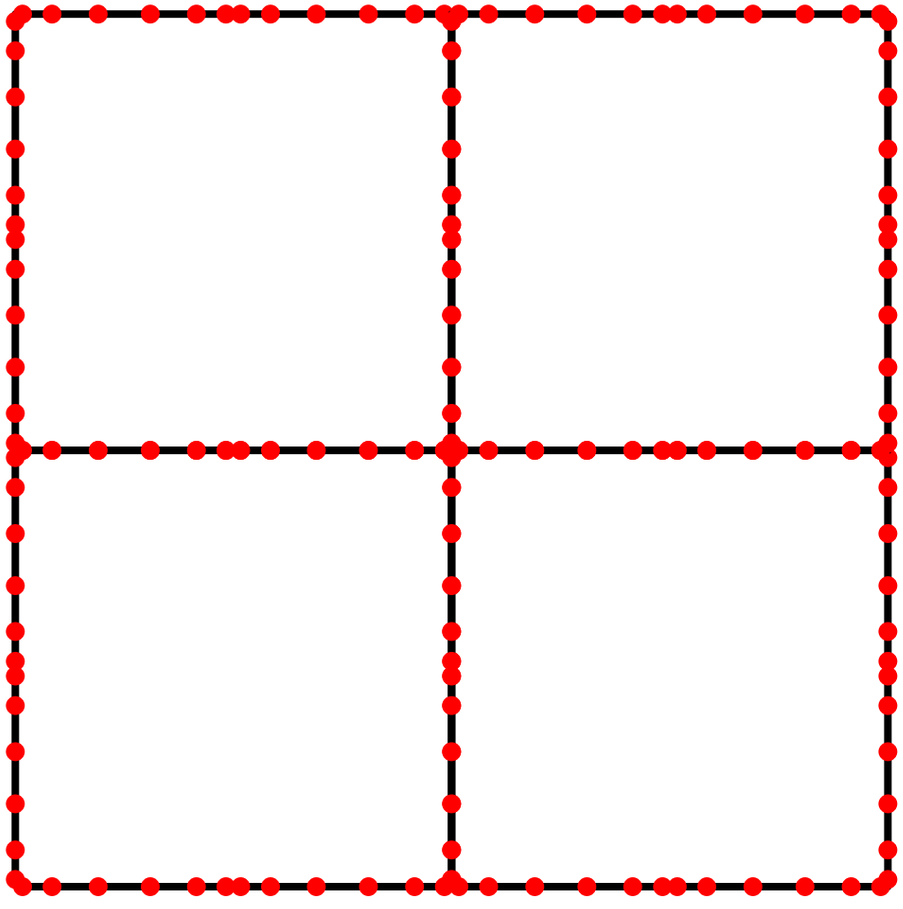
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Upwards sweep: Merge boxes by pairs and eliminate the interior (blue) nodes.

To do this, use the computed DtN operators to enforce continuity of u and du/dn across interior boundaries. Compute the DtN operator for the larger box.



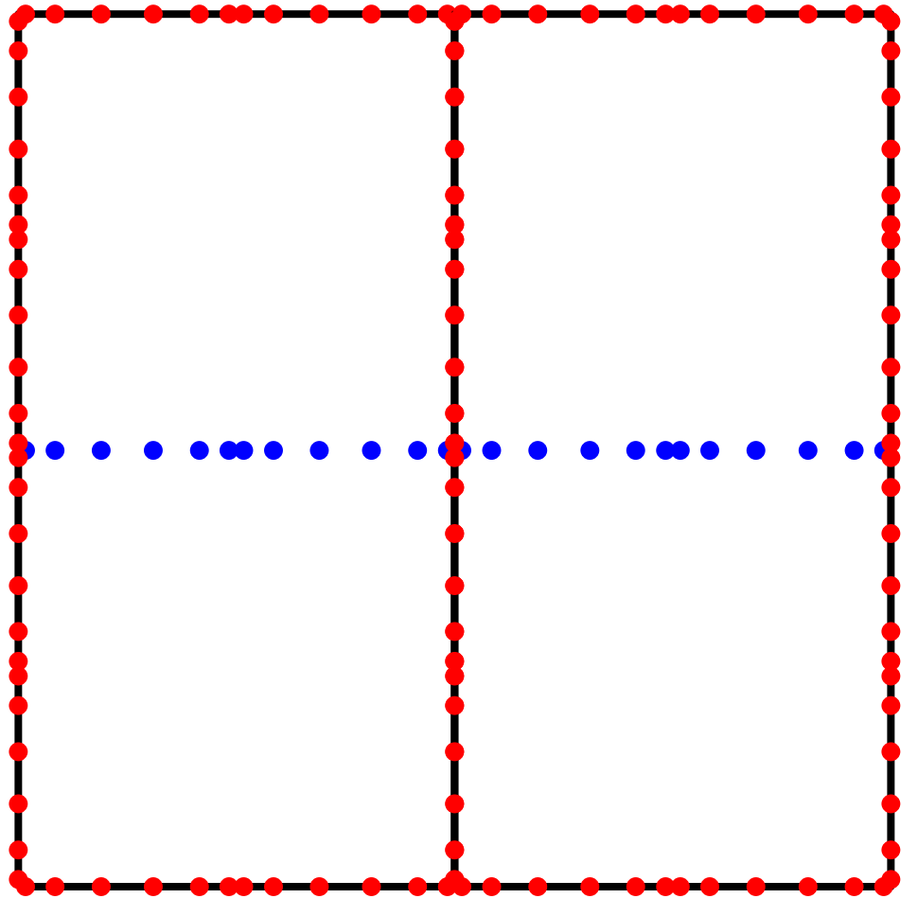
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Upwards sweep: Merge boxes by pairs and eliminate the interior (blue) nodes.

To do this, use the computed DtN operators to enforce continuity of u and du/dn across interior boundaries. Compute the DtN operator for the larger box.



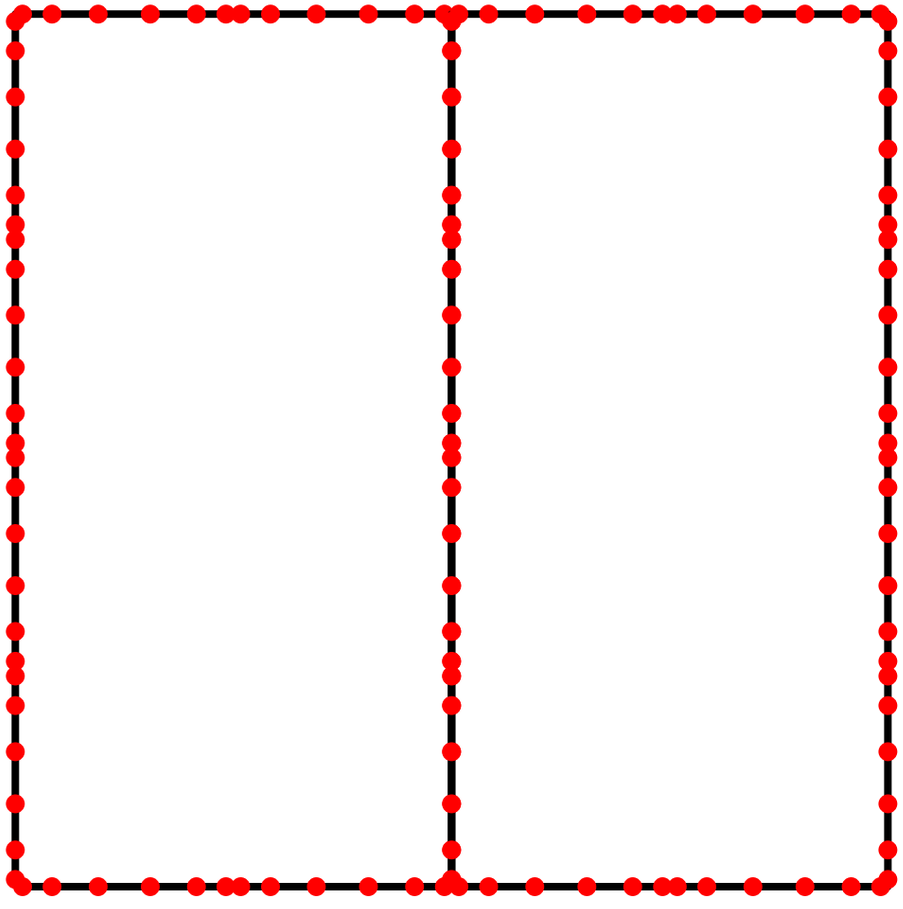
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Upwards sweep: Merge boxes by pairs and eliminate the interior (blue) nodes.

To do this, use the computed DtN operators to enforce continuity of u and du/dn across interior boundaries. Compute the DtN operator for the larger box.



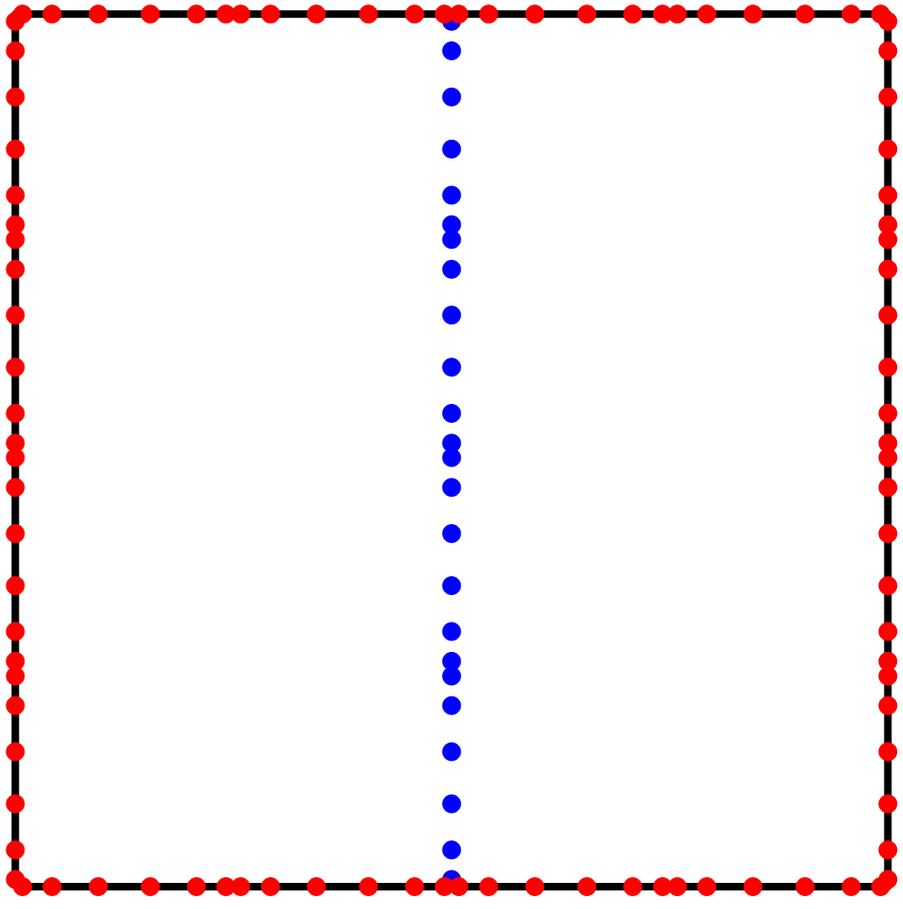
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Upwards sweep: Merge boxes by pairs and eliminate the interior (blue) nodes.

To do this, use the computed DtN operators to enforce continuity of u and du/dn across interior boundaries. Compute the DtN operator for the larger box.



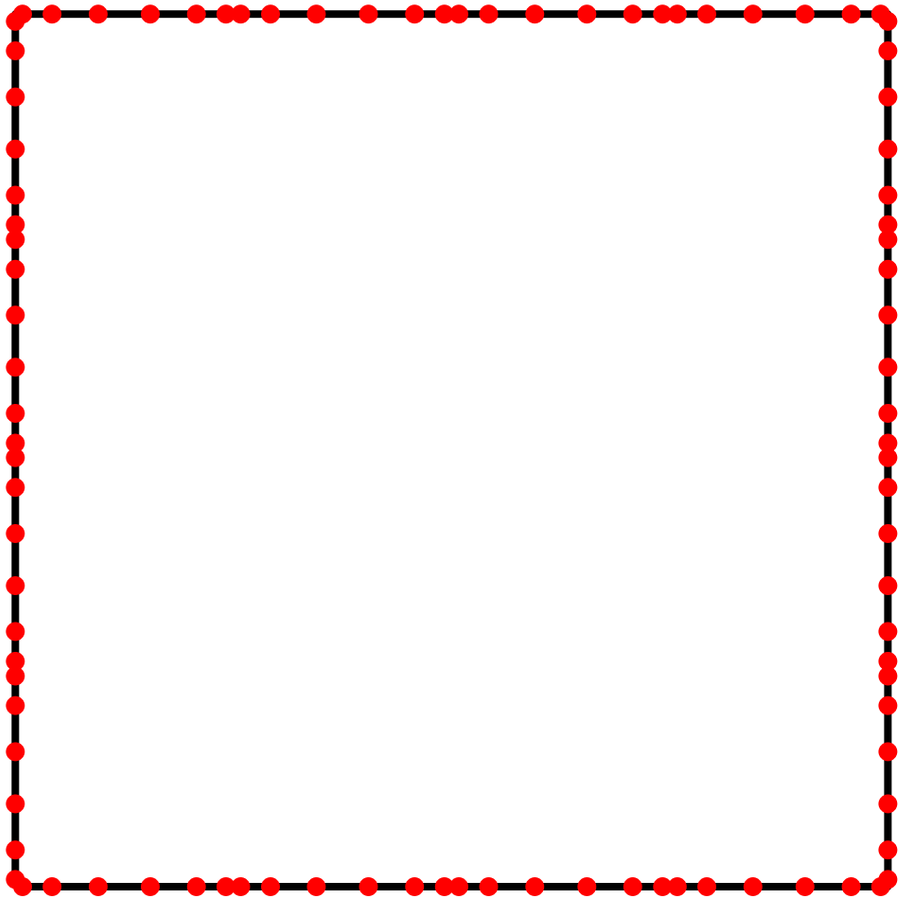
Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Upwards sweep: Merge boxes by pairs and eliminate the interior (blue) nodes.

To do this, use the computed DtN operators to enforce continuity of u and du/dn across interior boundaries. Compute the DtN operator for the larger box.

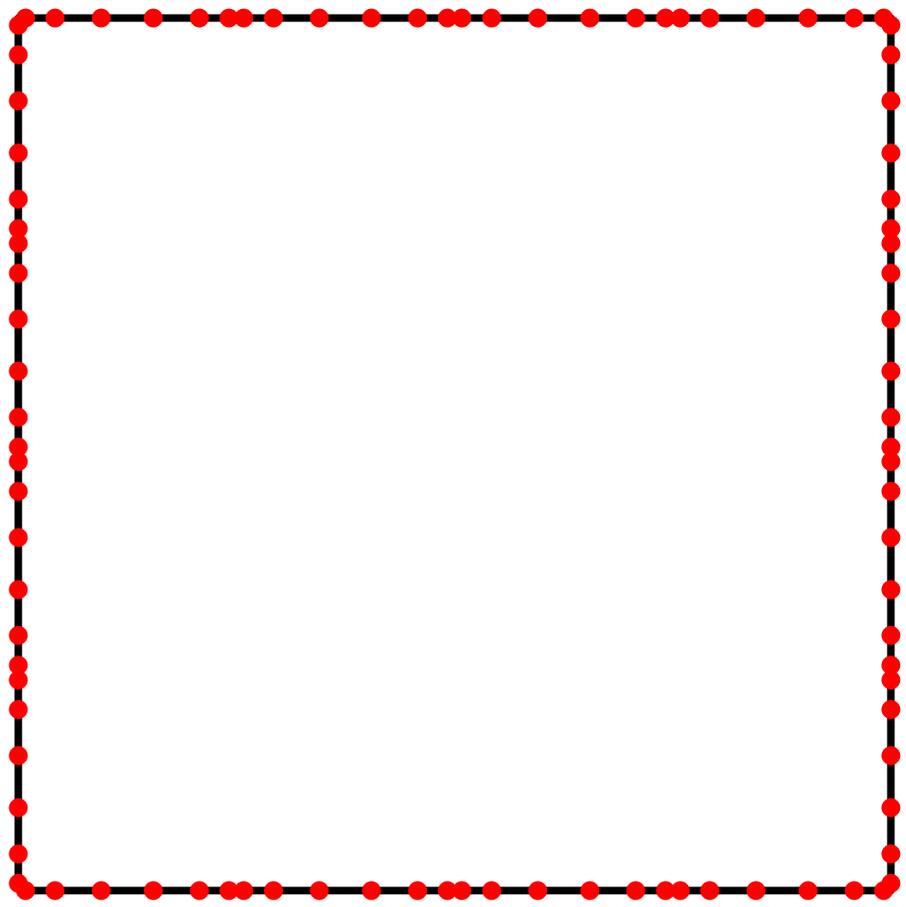


Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Top level solve: Invert the DtN operator for the top level box.

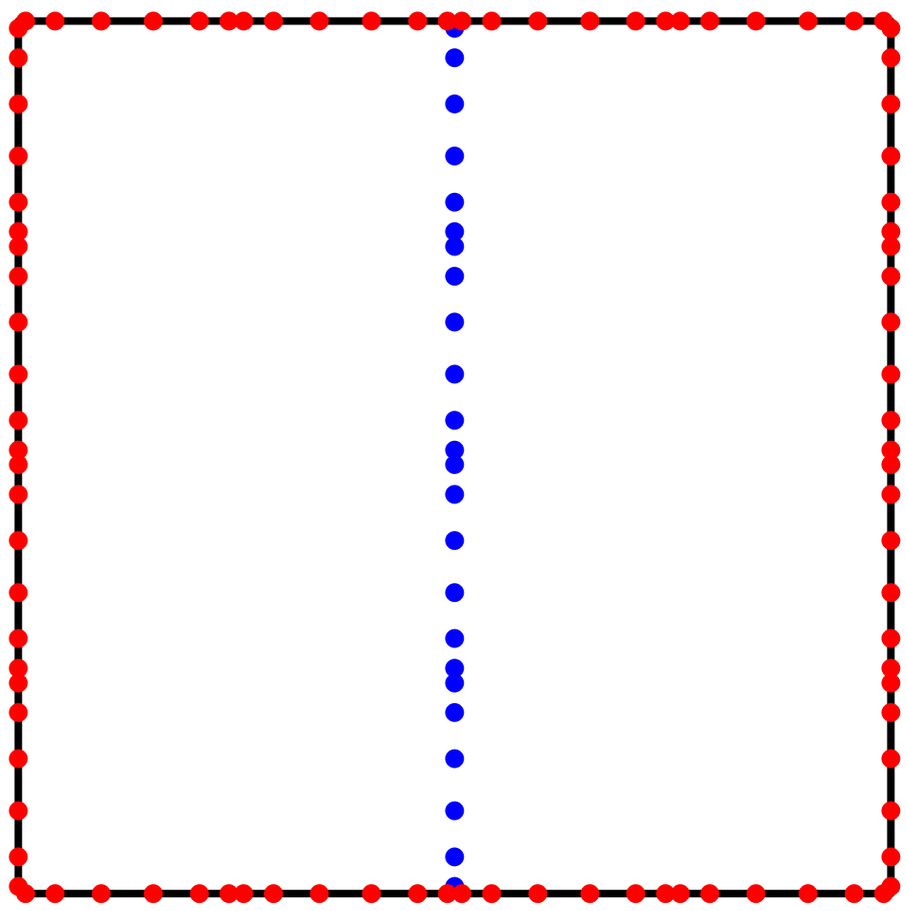


Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Downwards sweep: We know u on the red nodes. We can use the computed DtN operators to reconstruct u on the blue nodes.

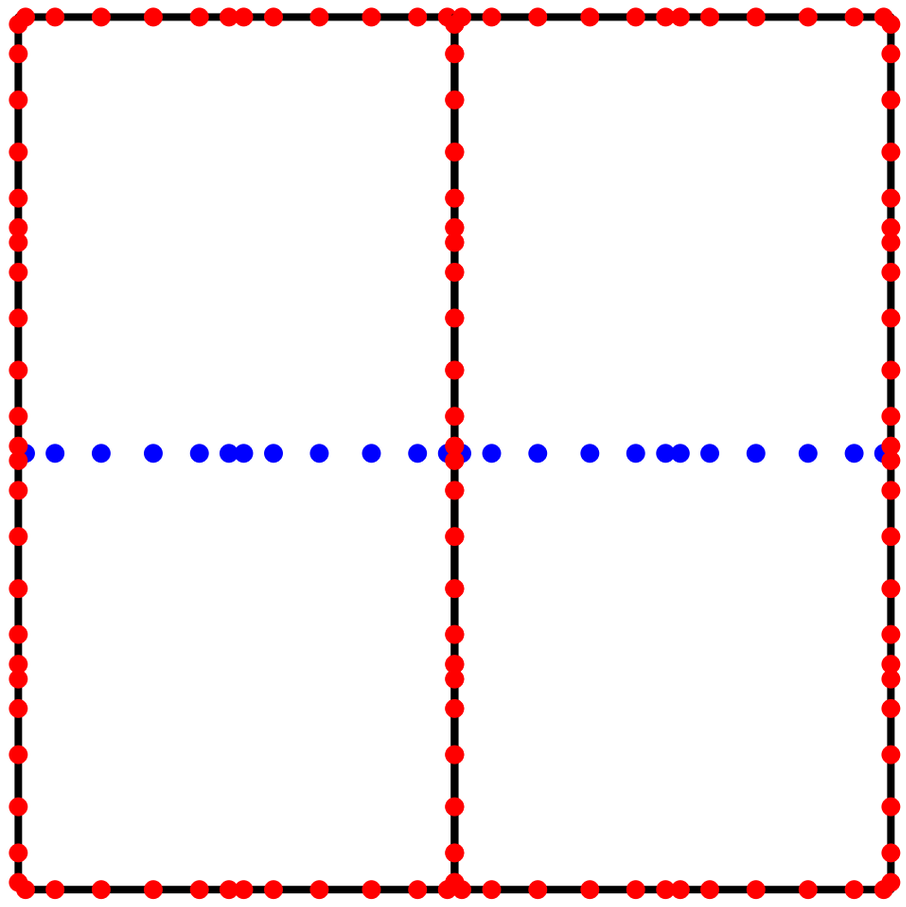


Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Downwards sweep: We know u on the red nodes. We can use the computed DtN operators to reconstruct u on the blue nodes.

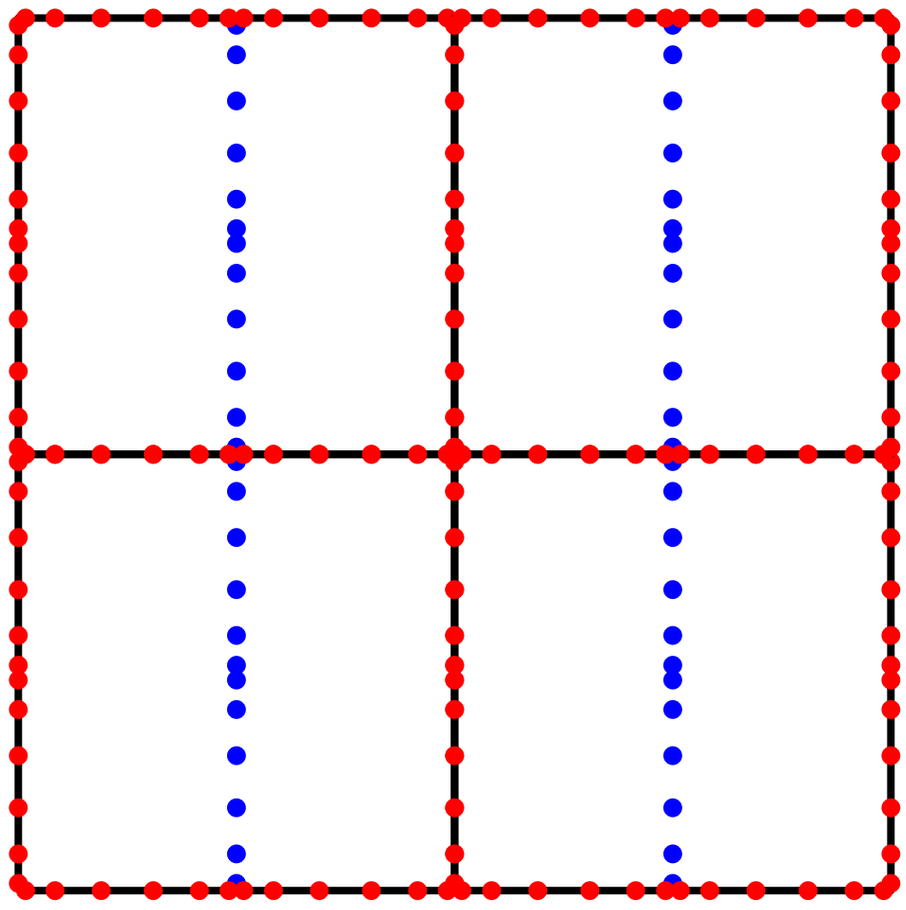


Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Downwards sweep: We know u on the red nodes. We can use the computed DtN operators to reconstruct u on the blue nodes.

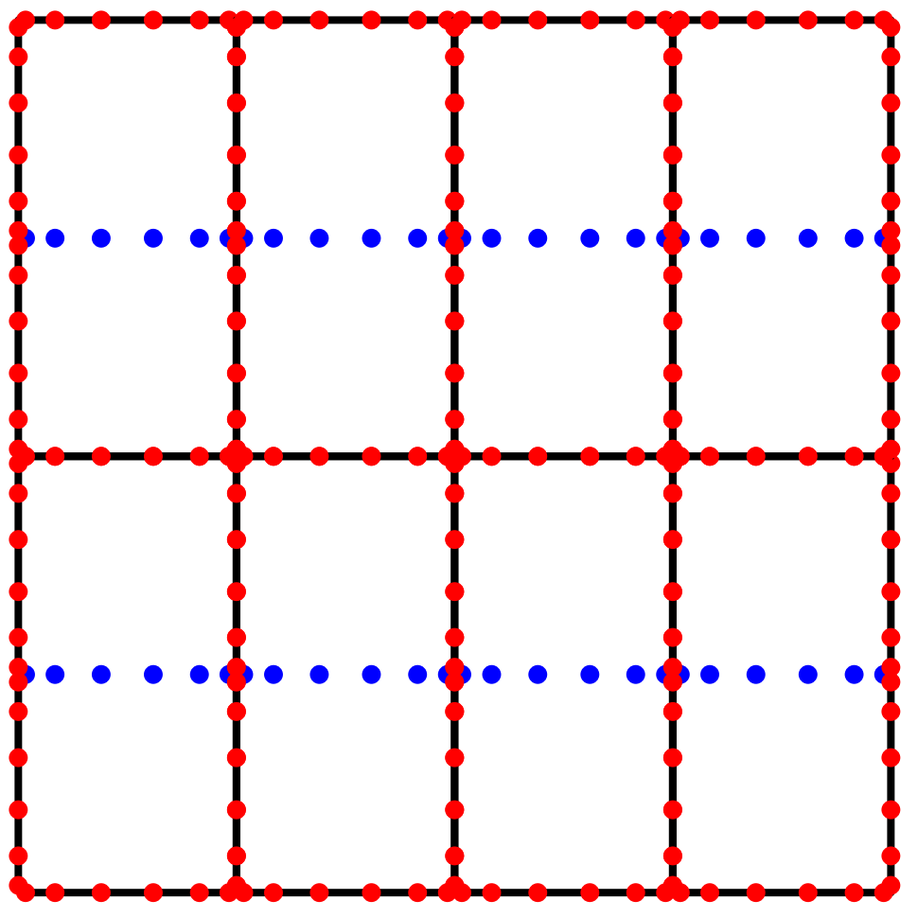


Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Downwards sweep: We know u on the red nodes. We can use the computed DtN operators to reconstruct u on the blue nodes.

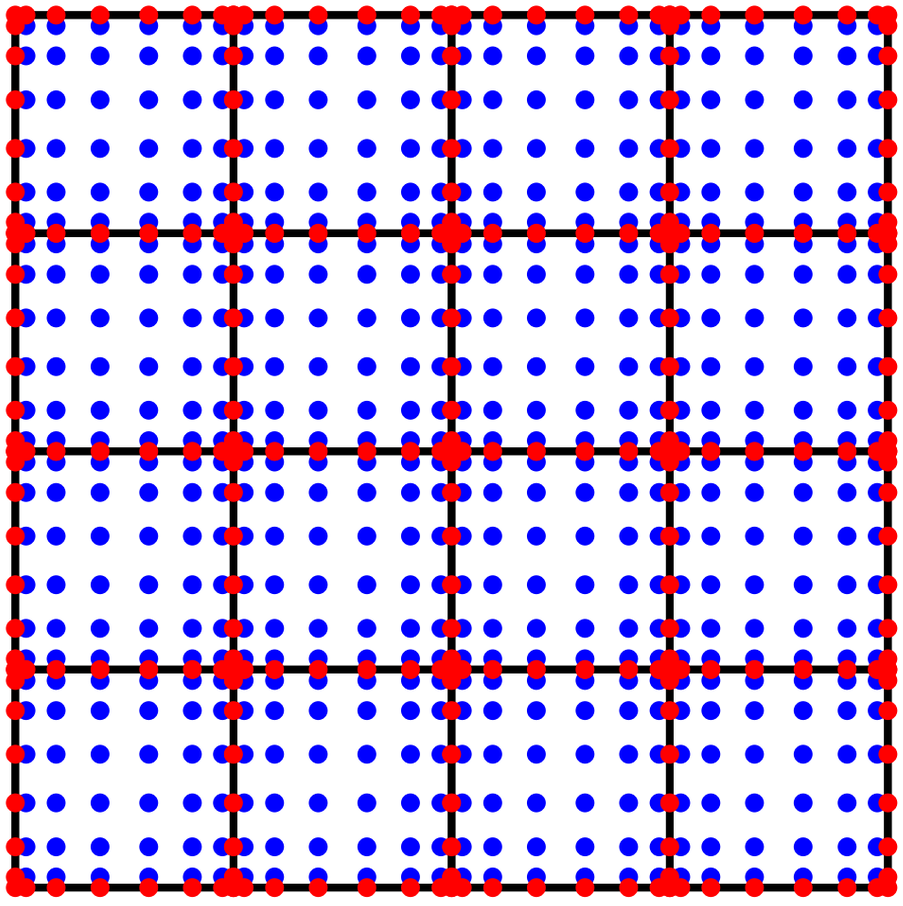


Model problem: Given f and b, find u such that

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

where $\Omega = [0,1]^2$ is the unit square and $\Gamma = \partial\Omega$. We assume u is smooth.

Downwards sweep: We know u on the red nodes. We can use the computed DtN operators to reconstruct u on the blue nodes.



Summary of the hierarchical scheme:

1. *Construct a quad-tree:* Partition the grid into a hierarchy of boxes.
2. *Process the leaves:* For each leaf box in the tree, construct its DtN (Dirichlet-to-Neumann) operator.
3. *Hierarchical merge:* Loop over all levels of the tree, from finer to smaller. For each box on a level, compute its DtN operator by merging the (already computed) DtN operators of its children.
4. *Process the root of the tree:* After completing Step 3, the DtN operator for the entire domain is available. Invert (or factor) it to construct the solution operator.

The first solve costs $O(N^{1.5})$ operations.

Subsequent solves cost $O(N \log N)$ operations.

Remark: For simplicity, the algorithm is described in a level-by-level manner (process all leaves first, then proceed one level at a time in going upwards). In fact, there is flexibility to travel through the tree in any order that ensures that no node is processed before its children. Since all Schur complements can be discarded once their information has been passed on to a parent, smarter orderings can greatly reduce the memory requirements.

Note: Very high order schemes can be used while retaining "thin" borders between boxes. Recall that a major drawback of classical nested dissection schemes is that their performance *plummets* when high-order discretization schemes are used.

Note: There are no corner singularities!

The exact mathematical object we are approximating is the Dirichlet-to-Neumann operator for a rectangle. This object exhibits complicated (singular) behavior near the corners.

However, *we are only concerned with the "projection" of the exact operator onto a space of smooth functions*. In particular, we need high accuracy only for functions that are restrictions of smooth global solutions.

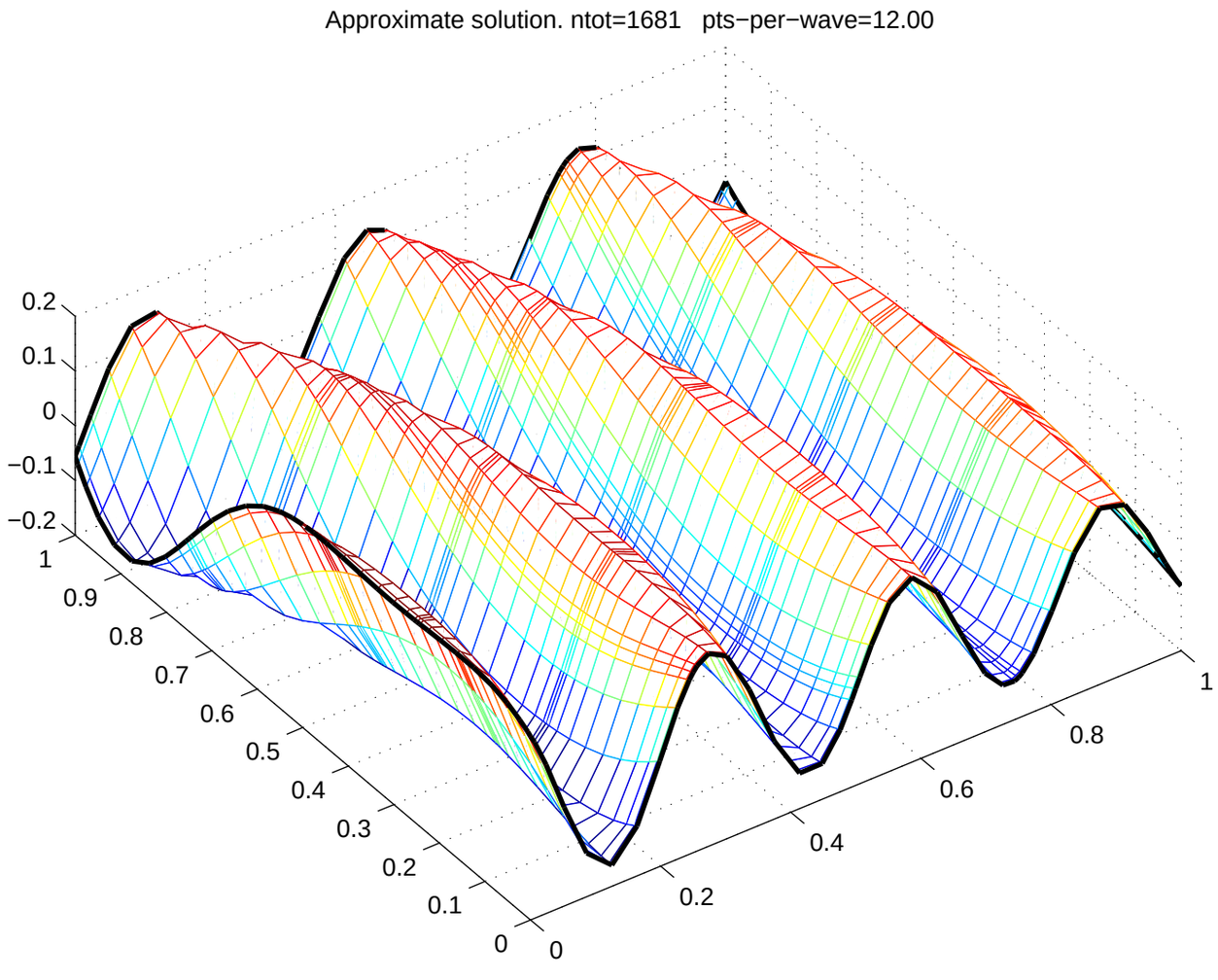
Hierarchical Poincaré-Steklov Method: numerical results

Set $\Omega = [0,1]^2$ and $\Gamma = \partial\Omega$. Consider the problem

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega, \\ u(\boldsymbol{x}) = f(\boldsymbol{x}), & \boldsymbol{x} \in \Gamma. \end{cases}$$

We pick f as the restriction of a wave from a point source, $\boldsymbol{x} \mapsto Y_0(\kappa|\boldsymbol{x} - \hat{\boldsymbol{x}}|)$.

We then know the exact solution, $u_{\text{exact}}(\boldsymbol{x}) = Y_0(\kappa|\boldsymbol{x} - \hat{\boldsymbol{x}}|)$.



Hierarchical Poincaré-Steklov Method: numerical results

Set $\Omega = [0,1]^2$ and $\Gamma = \partial\Omega$. Consider the problem

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega, \\ u(\boldsymbol{x}) = f(\boldsymbol{x}), & \boldsymbol{x} \in \Gamma. \end{cases}$$

We pick f as the restriction of a wave from a point source, $\boldsymbol{x} \mapsto Y_0(\kappa|\boldsymbol{x} - \hat{\boldsymbol{x}}|)$.

We then know the exact solution, $u_{\text{exact}}(\boldsymbol{x}) = Y_0(\kappa|\boldsymbol{x} - \hat{\boldsymbol{x}}|)$.

The spectral computation on a leaf involves 21×21 points.

κ is chosen so that there are 12 points per wave-length.

p	N	N_{wave}	t_{build} (sec)	t_{solve} (sec)	E_{pot}	E_{grad}	M (MB)	M/N (reals/DOF)
21	6561	6.7	0.23	0.0011	2.56528e-10	1.01490e-08	4.4	87.1
21	25921	13.3	0.92	0.0044	5.24706e-10	4.44184e-08	18.8	95.2
21	103041	26.7	4.68	0.0173	9.49460e-10	1.56699e-07	80.8	102.7
21	410881	53.3	22.29	0.0727	1.21769e-09	3.99051e-07	344.9	110.0
21	1640961	106.7	99.20	0.2965	1.90502e-09	1.24859e-06	1467.2	117.2
21	6558721	213.3	551.32	20.9551	2.84554e-09	3.74616e-06	6218.7	124.3

Error is measured in sup-norm: $e = \max_{\boldsymbol{x} \in \Omega} |u(\boldsymbol{x}) - u_{\text{exact}}(\boldsymbol{x})|$.

Note 1: Translation invariance is *not* exploited.

Note 2: The times refer to a simple Matlab implementation executed on a $1k laptop.

Note 3: Keeping a fixed number of points per wave-length works well for this scheme!

Hierarchical Poincaré-Steklov Method: numerical results

Set $\Omega = [0,1]^2$ and $\Gamma = \partial\Omega$. Consider the problem

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 u(\boldsymbol{x}) = 0, & \boldsymbol{x} \in \Omega, \\ u(\boldsymbol{x}) = f(\boldsymbol{x}), & \boldsymbol{x} \in \Gamma. \end{cases}$$

We pick f as the restriction of a wave from a point source, $\boldsymbol{x} \mapsto Y_0(\kappa|\boldsymbol{x} - \hat{\boldsymbol{x}}|)$.

We then know the exact solution, $u_{\text{exact}}(\boldsymbol{x}) = Y_0(\kappa|\boldsymbol{x} - \hat{\boldsymbol{x}}|)$.

The spectral computation on a leaf involves 41×41 points.

κ is chosen so that there are 12 points per wave-length.

p	N	N_{wave}	t_{build} (sec)	t_{solve} (sec)	E_{pot}	E_{grad}	M (MB)	M/N (reals/DOF)
41	6561	6.7	1.50	0.0025	9.88931e-14	3.46762e-12	7.9	157.5
41	25921	13.3	4.81	0.0041	1.58873e-13	1.12883e-11	32.9	166.4
41	103041	26.7	18.34	0.0162	3.95531e-13	5.51141e-11	137.1	174.4
41	410881	53.3	75.78	0.0672	3.89079e-13	1.03546e-10	570.2	181.9
41	1640961	106.7	332.12	0.2796	1.27317e-12	7.08201e-10	2368.3	189.2

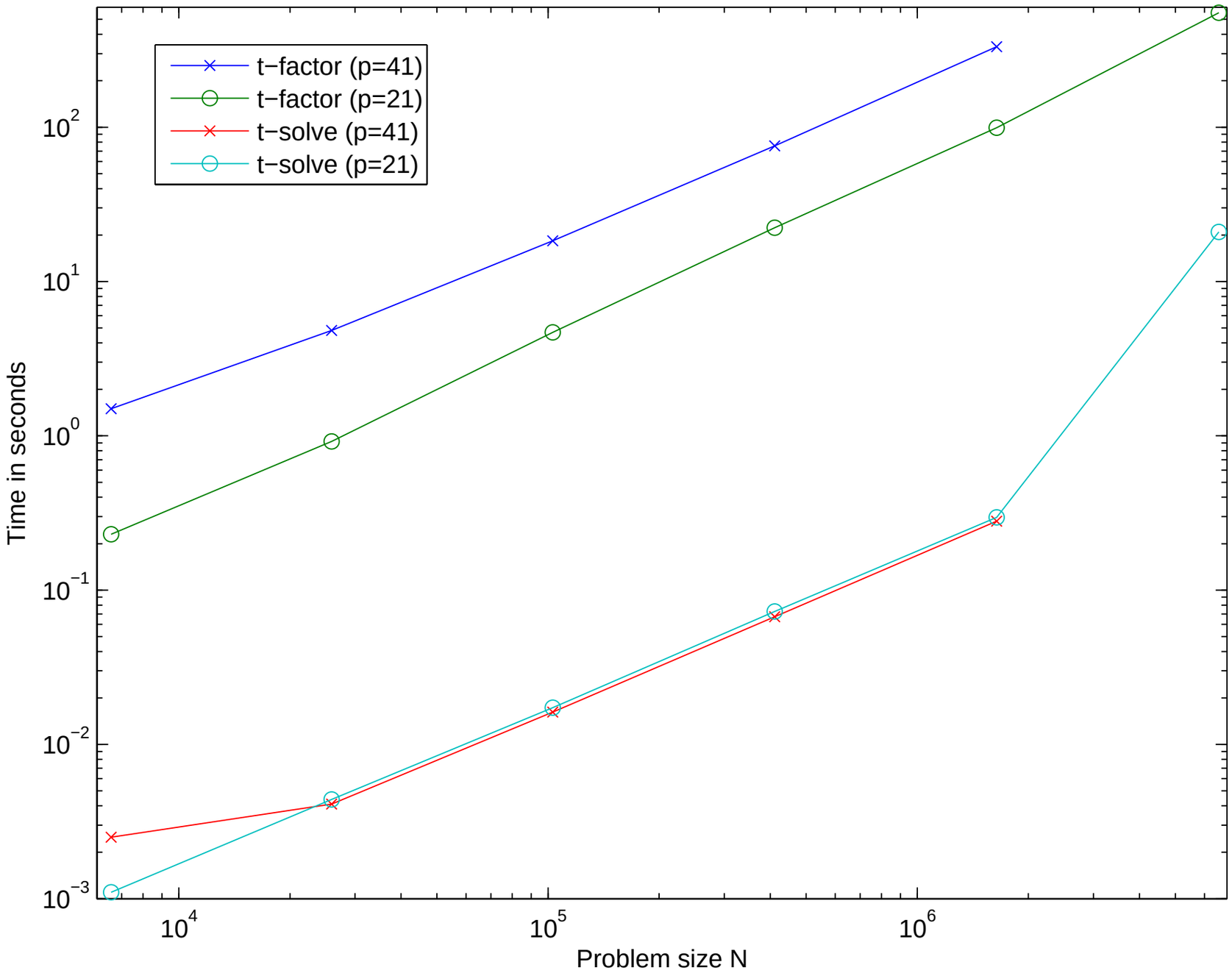
Error is measured in sup-norm: $e = \max_{\boldsymbol{x} \in \Omega} |u(\boldsymbol{x}) - u_{\text{exact}}(\boldsymbol{x})|$.

Note 1: Translation invariance is *not* exploited.

Note 2: The times refer to a simple Matlab implementation executed on a $1k laptop.

Note 3: Keeping a fixed number of points per wave-length works well for this scheme!

Spectral composite method: numerical results



The line t_{solve} scales perfectly linearly (until memory problems kick in), as expected.

Interesting: The line t_{build} also scales almost linearly. (Unexpectedly?) It turns out that t_{build} is dominated by the leaf computation; we have not yet hit the $O(N^{1.5})$ asymptotic.

Hierarchical Poincaré-Steklov Method: numerical results — variable coefficients

Now consider the variable coefficient problem

$$\begin{aligned} -\Delta u(\boldsymbol{x}) - \kappa^2 \left(1 - b(\boldsymbol{x})\right) u(\boldsymbol{x}) &= 0 && \boldsymbol{x} \in \Omega, \\ u(\boldsymbol{x}) &= f(\boldsymbol{x}) && \boldsymbol{x} \in \Gamma, \end{aligned}$$

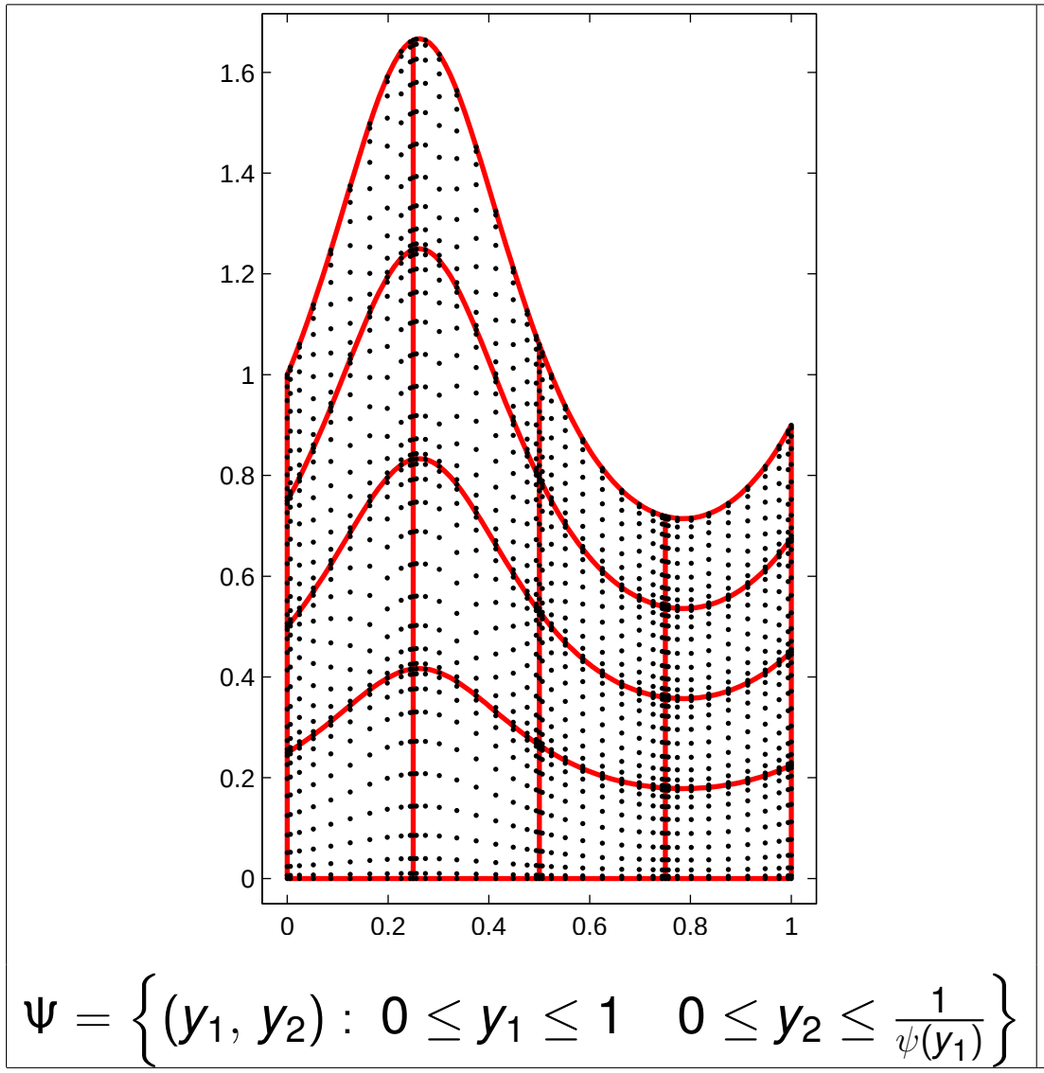
where $\Omega = [0,1]^2$, where $\Gamma = \partial\Omega$, and where $b(\boldsymbol{x}) = (\sin(4\pi x_1)\,\sin(4\pi x_2))^2$.

The Helmholtz parameter was kept fixed at $\kappa = 80$, corresponding to a domain size of 12.7×12.7 wave lengths. The boundary data was given by $f(\boldsymbol{x}) = \cos(8x_1)\left(1 - 2x_2\right)$.

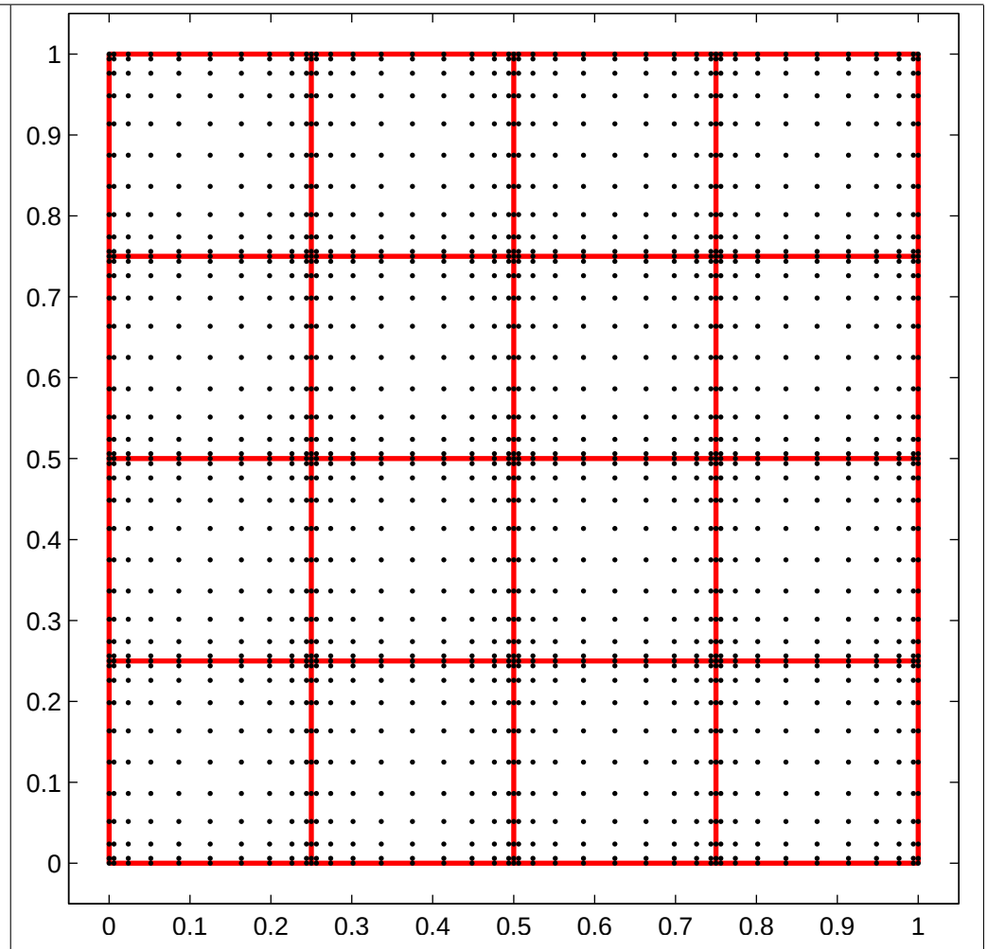
The error estimator $E_N^{\text{int}} = u_N(\hat{\boldsymbol{x}}) - u_{4N}(\hat{\boldsymbol{x}})$ where $\hat{\boldsymbol{x}} = (0.75,\, 0.25)$ is reported below:

p	N	pts per wave	$u_N(\hat{\boldsymbol{x}})$	E_N^{int}	$w_N(\hat{\boldsymbol{y}})$	E_N^{bnd}
21	6561	6.28	-2.448236804078803	-1.464e-03	-32991.4583727724	2.402e+02
21	25921	12.57	-2.446772430608166	7.976e-08	-33231.6118304666	5.984e-03
21	103041	25.13	-2.446772510369452	5.893e-11	-33231.6178142514	-5.463e-06
21	410881	50.27	-2.446772510428384	2.957e-10	-33231.6178087887	-2.792e-05
21	1640961	100.53	-2.446772510724068		-33231.6177808723	
41	6561	6.28	-2.446803898373796	-3.139e-05	-33233.0037457220	-1.386e+00
41	25921	12.57	-2.446772510320572	1.234e-10	-33231.6179029824	-8.940e-05
41	103041	25.13	-2.446772510443995	2.888e-11	-33231.6178135860	-1.273e-05
41	410881	50.27	-2.446772510472872	7.731e-11	-33231.6178008533	-4.668e-05
41	1640961	100.53	-2.446772510550181		-33231.6177541722	

A curved domain



$$\Psi = \left\{ (y_1, y_2) : \ 0 \le y_1 \le 1 \quad 0 \le y_2 \le \frac{1}{\psi(y_1)} \right\}$$



$$\Omega = [0, 1]^2$$

Consider a *curved domain* Ψ as shown above and the equation

$$\begin{cases} -\Delta u(\boldsymbol{y}) - \kappa^2 u(\boldsymbol{y}) = 0 & \boldsymbol{y} \in \Psi, \\ u(\boldsymbol{y}) = f(\boldsymbol{y}) & \boldsymbol{y} \in \partial\Psi. \end{cases} \tag{10}$$

The reparameterization is $y_1 = x_1$ and $y_2 = \psi(y_1)\, y_2$, and so the Helmholtz equation (10) takes the form

$$\frac{\partial^2 u}{\partial x_1^2} + \frac{2\psi'(x_1)\, x_2}{\psi(x_1)} \frac{\partial^2 u}{\partial x_1 \partial x_2} + \left(\frac{x_2^2 \psi'(x_1)^2}{\psi(x_1)^2} + \psi(x_1)^2 \right) \frac{\partial^2 u}{\partial x_2^2} + \frac{x_2 \psi''(x_1)}{\psi(x_1)} \frac{\partial u}{\partial x_2} + k^2 u = 0, \qquad \boldsymbol{x} \in \Omega.$$

Numerical results for curved domain

The equation is (constant coefficient) Helmholtz on a domain of size 35×50 wave lengths.

	Exact solution known		*Dirichlet data* $f = 1$		
N	E_{pot}	E_{grad}	$E_N^{(1)}$	$E_N^{(2)}$	$E_N^{(3)}$
25921	2.12685e+02	3.55772e+04	2.24618e-01	4.99854e-01	6.69023e-01
103041	3.29130e-01	5.89976e+01	1.10143e-02	5.28238e-03	6.14890e-02
410881	1.40813e-05	1.98907e-03	4.57900e-06	2.18438e-06	1.13415e-05
1640961	7.22959e-10	1.17852e-07	5.12914e-07	1.67971e-06	4.97764e-06
3690241	1.63144e-09	2.26204e-07	—	—	—

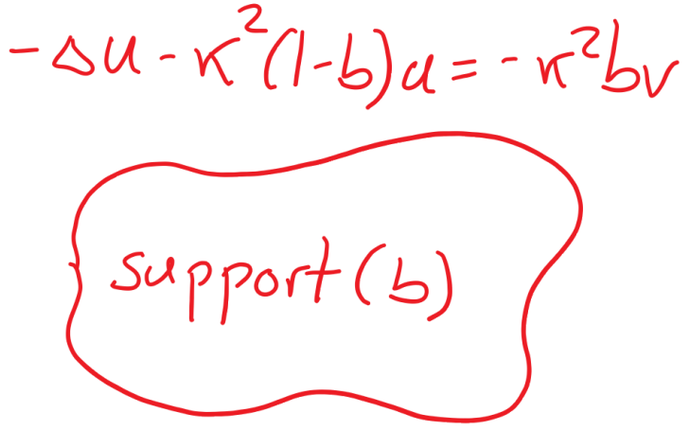
Hierarchical Poincaré-Steklov Method: "FEM-BEM coupling"

Consider the free space acoustic scattering problem

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u(\boldsymbol{x}) = -\kappa^2 b(\boldsymbol{x})\, v(\boldsymbol{x}), & \boldsymbol{x} \in \mathbb{R}^2 \\ \lim_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\big(\partial_{|\boldsymbol{x}|} u(\boldsymbol{x}) - i\kappa\, u(\boldsymbol{x})\big) = 0, \end{cases}$$

where

- b is a smooth scattering potential with compact support, where
- v is a given "incoming potential" and where
- u is the sought "outgoing potential."



Joint work with A. Barnett and A. Gillman.

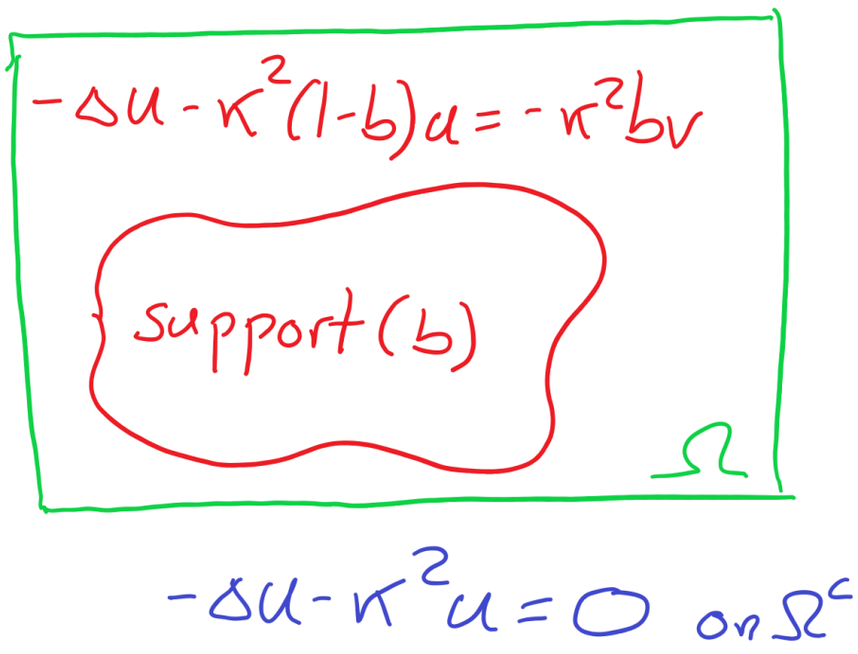
Hierarchical Poincaré-Steklov Method: "FEM-BEM coupling"

Consider the free space acoustic scattering problem

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u(\boldsymbol{x}) = -\kappa^2 b(\boldsymbol{x})\, v(\boldsymbol{x}), & \boldsymbol{x} \in \mathbb{R}^2 \\ \lim_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\left(\partial_{|\boldsymbol{x}|} u(\boldsymbol{x}) - i\kappa\, u(\boldsymbol{x})\right) = 0, \end{cases}$$

where

- b is a smooth scattering potential with compact support, where
- v is a given "incoming potential" and where
- u is the sought "outgoing potential."



Introduce an artificial box Ω such that support(b) $\subseteq \Omega$.

Joint work with A. Barnett and A. Gillman.

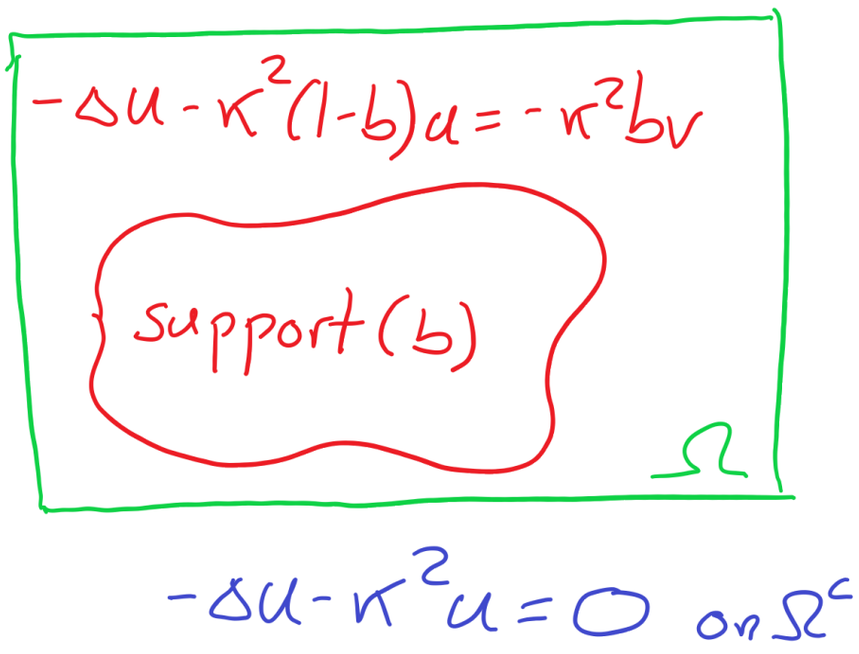
Hierarchical Poincaré-Steklov Method: "FEM-BEM coupling"

Consider the free space acoustic scattering problem

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u(\boldsymbol{x}) = -\kappa^2 b(\boldsymbol{x})\, v(\boldsymbol{x}), & \boldsymbol{x} \in \mathbb{R}^2 \\ \lim_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\left(\partial_{|\boldsymbol{x}|} u(\boldsymbol{x}) - i\kappa\, u(\boldsymbol{x})\right) = 0, \end{cases}$$

where

- b is a smooth scattering potential with compact support, where
- v is a given "incoming potential" and where
- u is the sought "outgoing potential."



Introduce an artificial box Ω such that support(b) $\subseteq \Omega$.

On Ω:

- Variable coefficient PDE.

On Ω^c:

- Constant coefficient PDE.

Joint work with A. Barnett and A. Gillman.

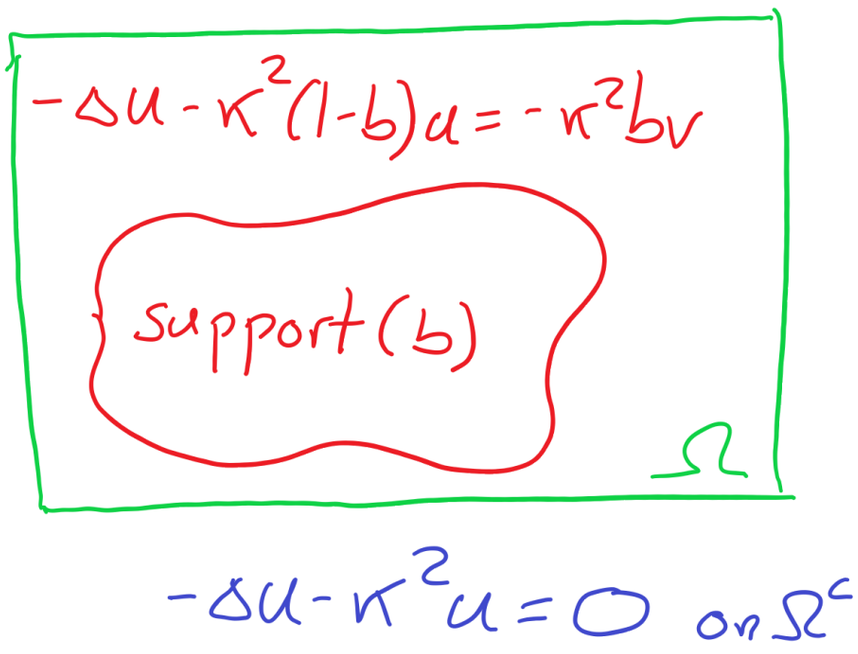
Hierarchical Poincaré-Steklov Method: "FEM-BEM coupling"

Consider the free space acoustic scattering problem

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u(\boldsymbol{x}) = -\kappa^2 b(\boldsymbol{x})\, v(\boldsymbol{x}), & \boldsymbol{x} \in \mathbb{R}^2 \\ \lim_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\left(\partial_{|\boldsymbol{x}|} u(\boldsymbol{x}) - i\kappa\, u(\boldsymbol{x})\right) = 0, \end{cases}$$

where

- b is a smooth scattering potential with compact support, where
- v is a given "incoming potential" and where
- u is the sought "outgoing potential."



Introduce an artificial box Ω such that support$(b) \subseteq \Omega$.

On Ω:

- Variable coefficient PDE.
- Use HPS.

On Ω^c:

- Constant coefficient PDE.
- Use BIE.

Joint work with A. Barnett and A. Gillman.

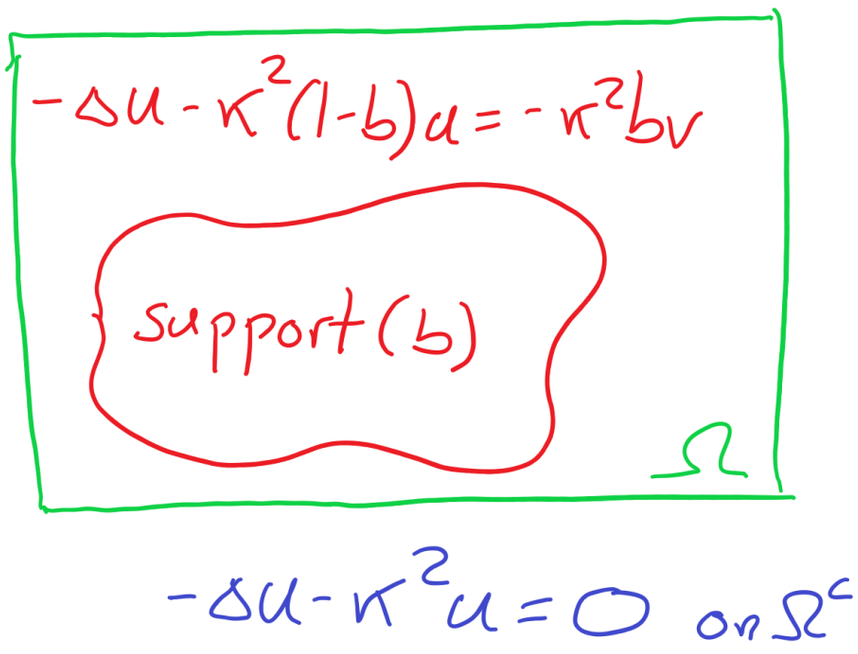
Hierarchical Poincaré-Steklov Method: "FEM-BEM coupling"

Consider the free space acoustic scattering problem

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u(\boldsymbol{x}) = -\kappa^2 b(\boldsymbol{x})\, v(\boldsymbol{x}), & \boldsymbol{x} \in \mathbb{R}^2 \\ \lim_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\left(\partial_{|\boldsymbol{x}|} u(\boldsymbol{x}) - i\kappa\, u(\boldsymbol{x})\right) = 0, \end{cases}$$

where

- b is a smooth scattering potential with compact support, where
- v is a given "incoming potential" and where
- u is the sought "outgoing potential."



Introduce an artificial box Ω such that support$(b) \subseteq \Omega$.

On Ω:

- Variable coefficient PDE.
- Use HPS.
- Build DtN for $\partial\Omega$.

On Ω^c:

- Constant coefficient PDE.
- Use BIE.
- Build DtN for $\partial\Omega^c$.

Joint work with A. Barnett and A. Gillman.

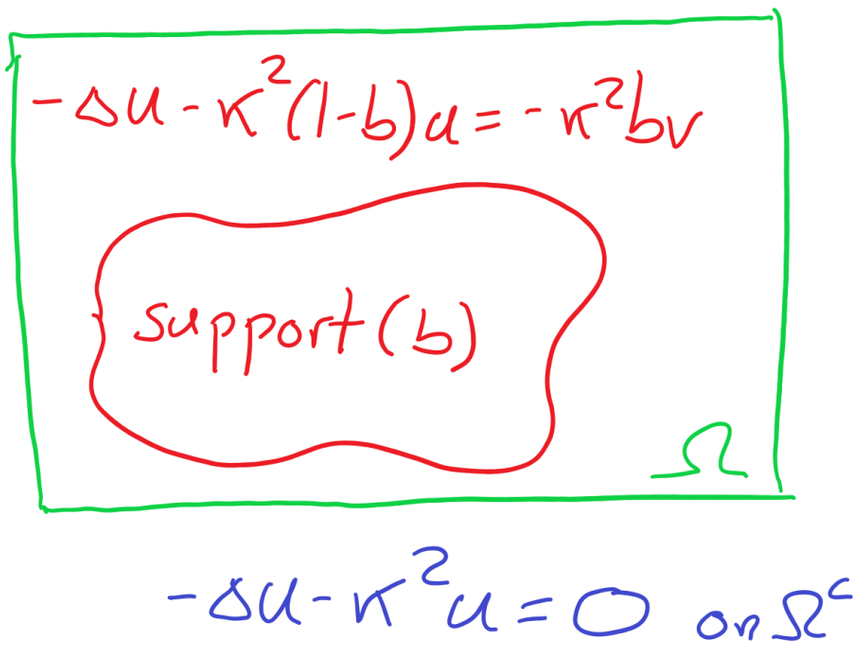
Hierarchical Poincaré-Steklov Method: "FEM-BEM coupling"

Consider the free space acoustic scattering problem

$$\begin{cases} -\Delta u(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u(\boldsymbol{x}) = -\kappa^2 b(\boldsymbol{x})\, v(\boldsymbol{x}), & \boldsymbol{x} \in \mathbb{R}^2 \\ \lim_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\left(\partial_{|\boldsymbol{x}|} u(\boldsymbol{x}) - i\kappa\, u(\boldsymbol{x})\right) = 0, \end{cases}$$

where

- b is a smooth scattering potential with compact support, where
- v is a given "incoming potential" and where
- u is the sought "outgoing potential."



Introduce an artificial box Ω such that support$(b) \subseteq \Omega$.

On Ω:

- Variable coefficient PDE.
- Use HPS.
- Build DtN for $\partial\Omega$.

On Ω^c:

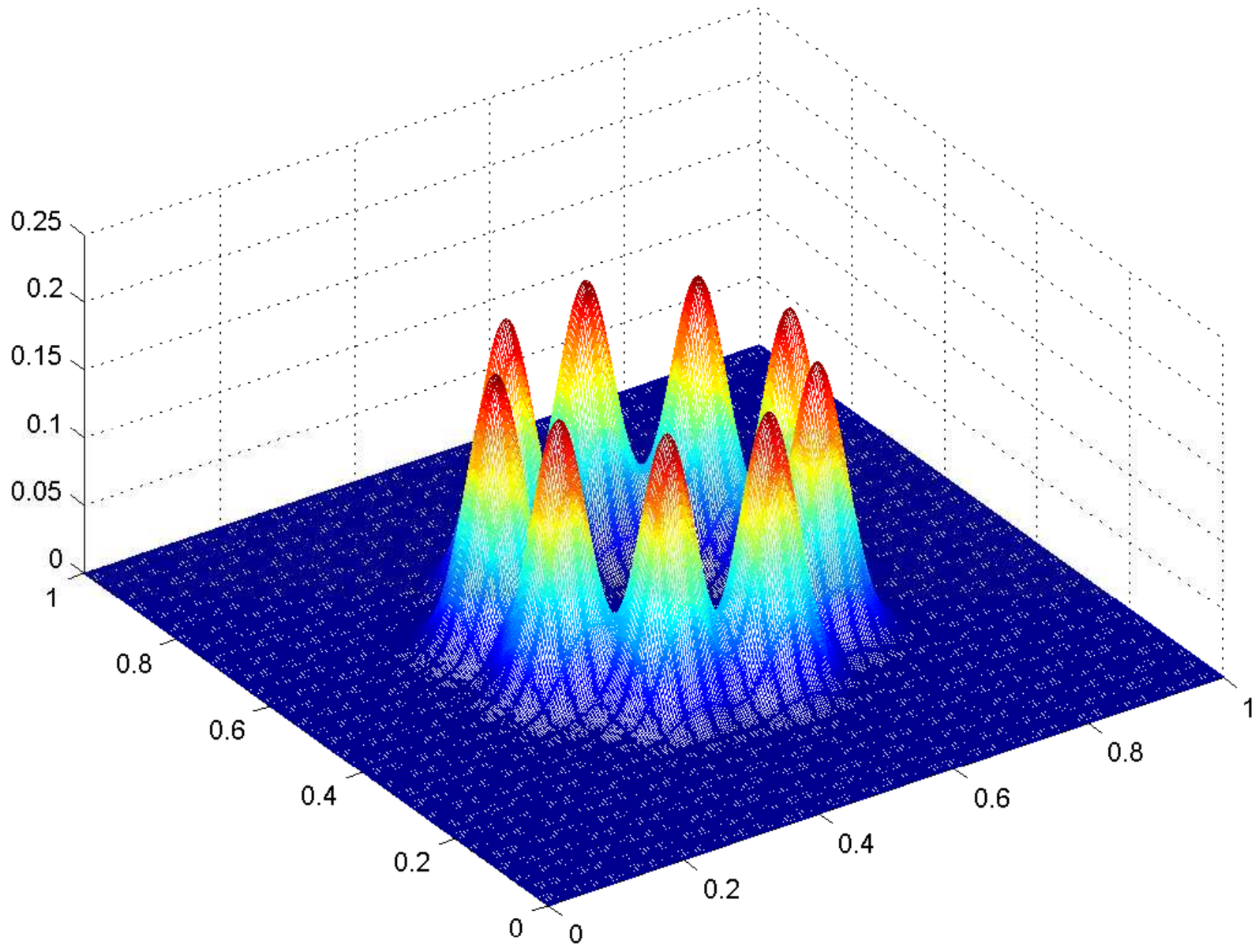
- Constant coefficient PDE.
- Use BIE.
- Build DtN for $\partial\Omega^c$.

- *Merge using fast operator algebra!*

Joint work with A. Barnett and A. Gillman.

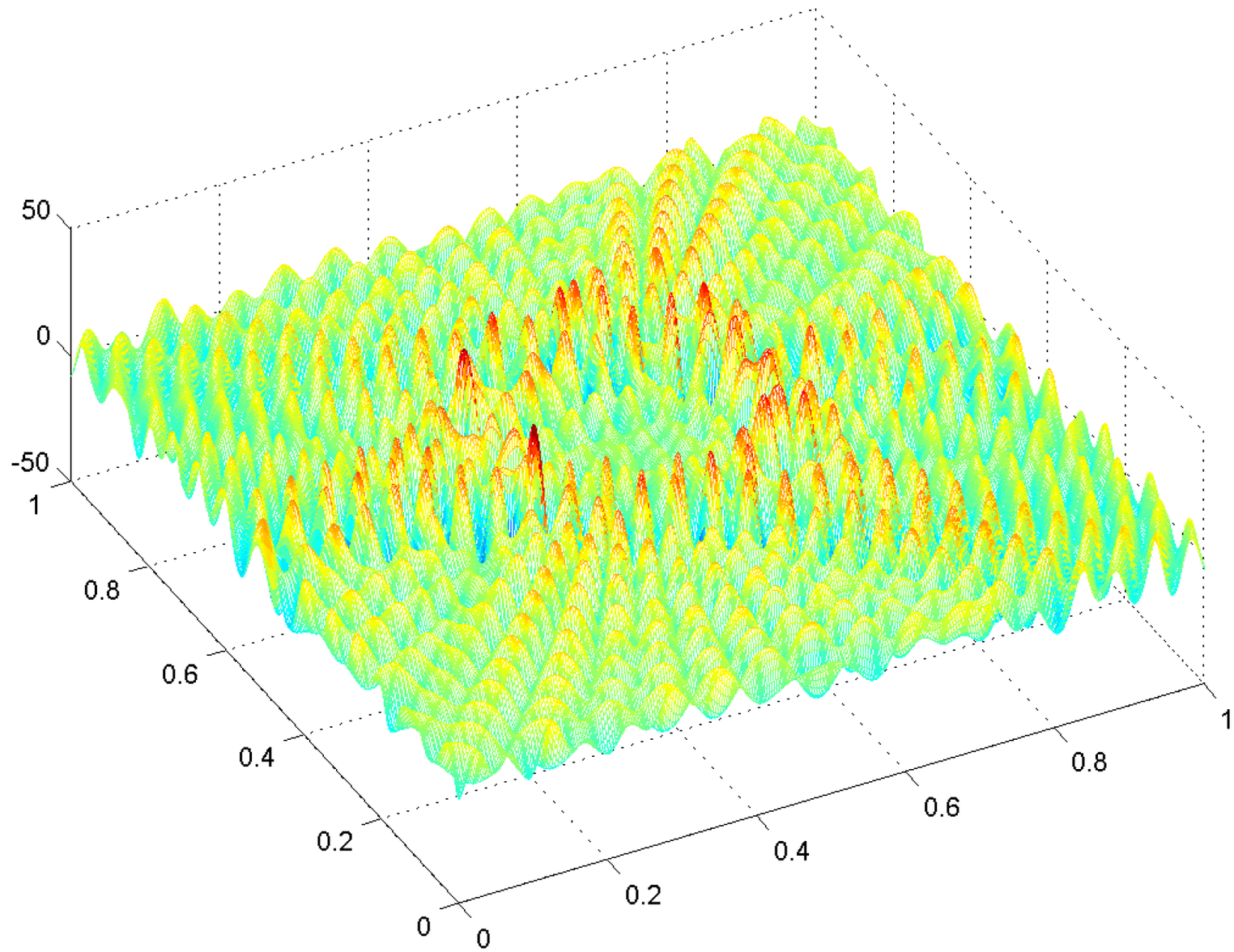
Example: Free space scattering $\begin{cases} -\Delta u_{\text{out}}(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u_{\text{out}}(\boldsymbol{x}) = -\kappa^2\, b(\boldsymbol{x})\, u_{\text{in}}(\boldsymbol{x}) \\ \lim\limits_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\left(\partial_{|\boldsymbol{x}|} u_{\text{out}}(\boldsymbol{x}) - i\kappa\, u_{\text{out}}(\boldsymbol{x})\right) = 0 \end{cases}$

The scattering potential b



Example: Free space scattering $\begin{cases} -\Delta u_{\text{out}}(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u_{\text{out}}(\boldsymbol{x}) = -\kappa^2\, b(\boldsymbol{x})\, u_{\text{in}}(\boldsymbol{x}) \\ \lim\limits_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\left(\partial_{|\boldsymbol{x}|} u_{\text{out}}(\boldsymbol{x}) - i\kappa\, u_{\text{out}}(\boldsymbol{x})\right) = 0 \end{cases}$

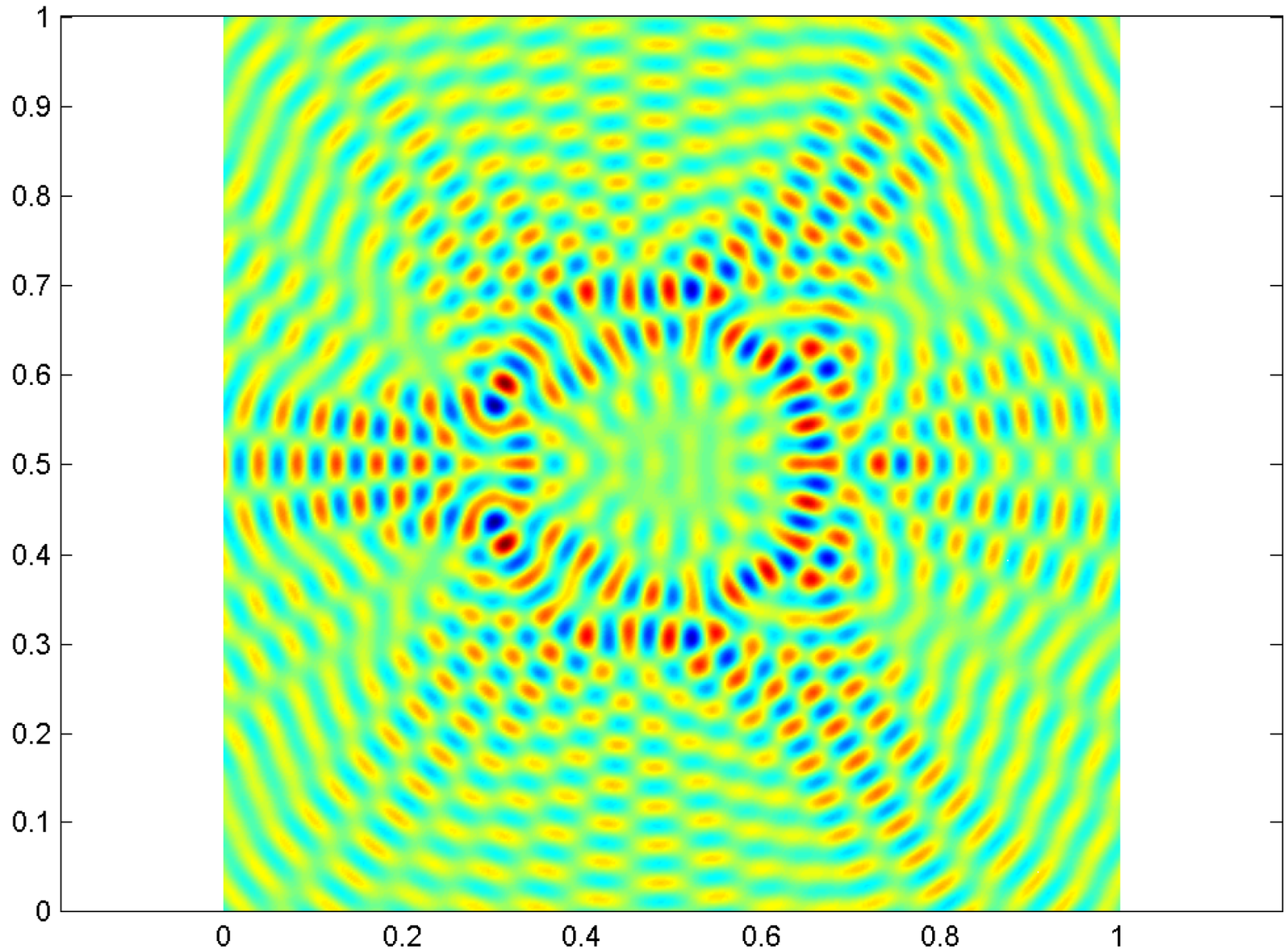
The outgoing field u_{out} (resulting from an incoming plane wave $u_{\text{in}}(x) = \cos(\kappa x_1)$)



$N = 231\,361$ $T_{\text{build}} = 7.2$ sec $T_{\text{solve}} = 0.06$ sec $E \approx 10^{-7}$ (estimated)

Example: Free space scattering $\begin{cases} -\Delta u_{\text{out}}(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u_{\text{out}}(\boldsymbol{x}) = -\kappa^2\, b(\boldsymbol{x})\, u_{\text{in}}(\boldsymbol{x}) \\ \lim\limits_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\left(\partial_{|\boldsymbol{x}|} u_{\text{out}}(\boldsymbol{x}) - i\kappa\, u_{\text{out}}(\boldsymbol{x})\right) = 0 \end{cases}$

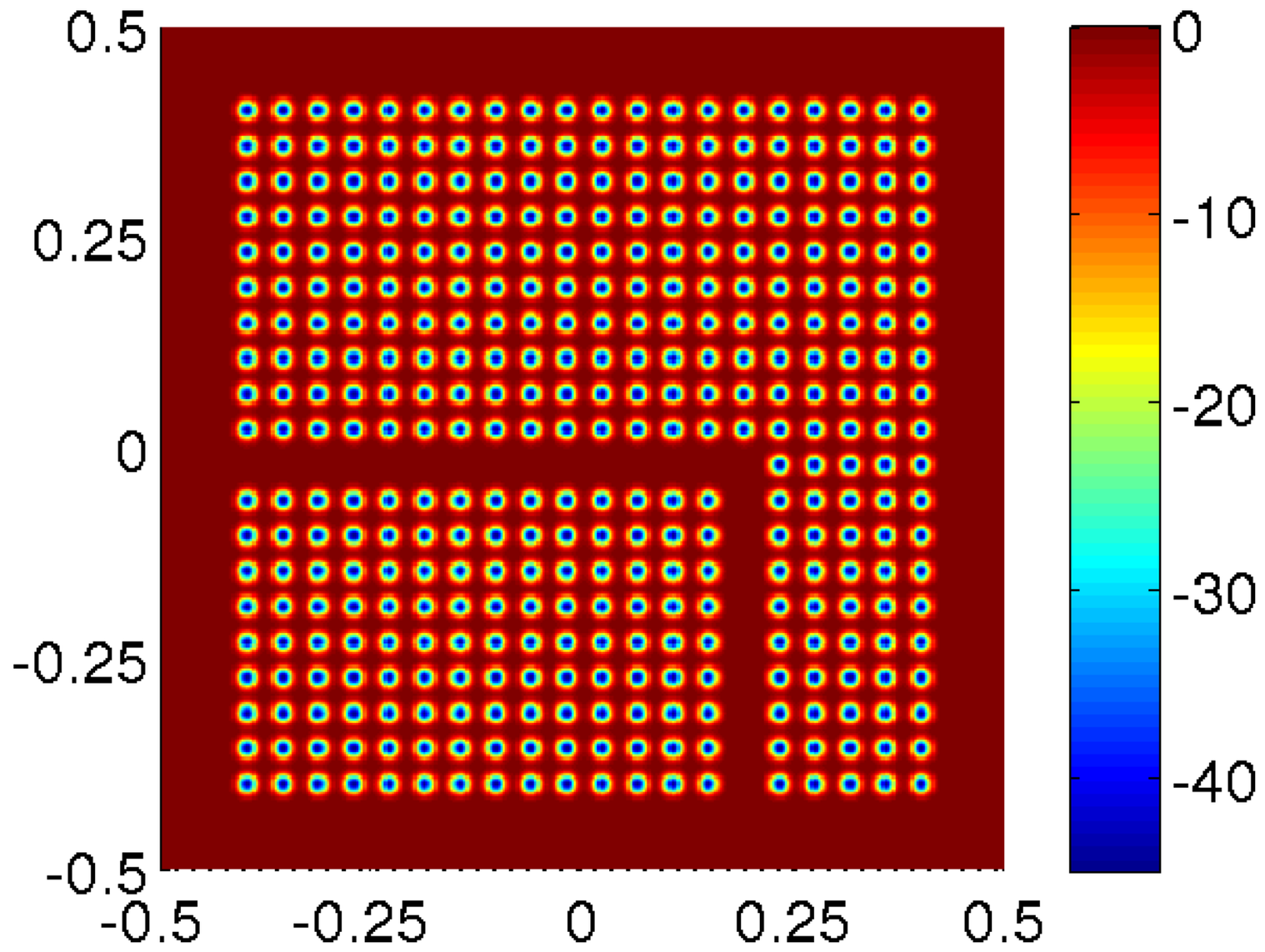
The outgoing field u_{out} (resulting from an incoming plane wave $u_{\text{in}}(x) = \cos(\kappa x_1)$)



$N = 231\,361$ $T_{\text{build}} = 7.2$ sec $T_{\text{solve}} = 0.06$ sec $E \approx 10^{-7}$ (estimated)

Example: Free space scattering $\begin{cases} -\Delta u_{\text{out}}(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u_{\text{out}}(\boldsymbol{x}) = -\kappa^2\, b(\boldsymbol{x})\, u_{\text{in}}(\boldsymbol{x}) \\ \lim\limits_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\big(\partial_{|\boldsymbol{x}|} u_{\text{out}}(\boldsymbol{x}) - i\kappa\, u_{\text{out}}(\boldsymbol{x})\big) = 0 \end{cases}$

The scattering potential b — now a photonic crystal with a wave guide.



$N = 231\,361$ $T_{\text{build}} = 7.2$ sec $T_{\text{solve}} = 0.06$ sec $E \approx 10^{-6}$ (estimated)

Example: Free space scattering $\begin{cases} -\Delta u_{\text{out}}(\boldsymbol{x}) - \kappa^2 (1 - b(\boldsymbol{x}))\, u_{\text{out}}(\boldsymbol{x}) = -\kappa^2\, b(\boldsymbol{x})\, u_{\text{in}}(\boldsymbol{x}) \\ \lim\limits_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\big(\partial_{|\boldsymbol{x}|} u_{\text{out}}(\boldsymbol{x}) - i\kappa\, u_{\text{out}}(\boldsymbol{x})\big) = 0 \end{cases}$

The total field $u = u_{\text{in}} + u_{\text{out}}$ (resulting from an incoming plane wave $u_{\text{in}}(x) = \cos(\kappa\, x_1)$).