Towards an optimal complexity SEM

Traditionally, there have been two perspectives on element methods:

① Domain decomposition

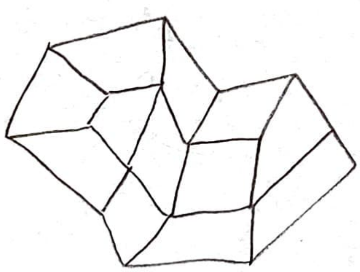
"Patching" / "multidomain"
Schur complement method
Schwarz method
Poincare - Steklov method

- Typically based on strong form of PDE.
- Elements treated as decoupled subdomains, continuity enforced directly.

② Variational formulation (popular)

Finite element method
"Spectral" element method
Discontinuous Galerkin method
"Static condensation"

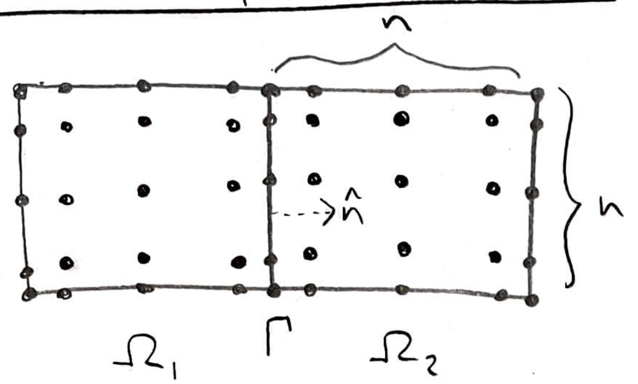
- Based on weak form of PDE
- Continuity enforced automatically by element basis



- ① is simpler if we have a fast spectrally-accurate solver for .
- ② potentially leads to structure-preserving discretizations. (symmetric, positive-definite).

Regardless of choice of ① or ②, the idea behind the Schur complement method can still be applied.

Schur complement method



$$\nabla^2 u = f \quad \text{in } \Omega = \Omega_1 \cup \Omega_2$$
$$u = 0 \quad \text{on } \partial\Omega$$

Think of "•" as coefficients or values.

From perspective ①: Decompose problem as

$$(*) \quad \begin{cases} \nabla^2 u_1 = f_1 & \text{in } \Omega_1 \\ \nabla^2 u_2 = f_2 & \text{in } \Omega_2 \\ u_1 = u_2 & \text{on } \Gamma \\ \dfrac{\partial u_1}{\partial \hat{n}} = -\dfrac{\partial u_2}{\partial \hat{n}} & \text{on } \Gamma \\ u_1 = u_2 = 0 & \text{on } \partial\Omega \end{cases}$$

We can decouple the subproblems in $(*)$ by introducing unknowns on Γ, u_Γ:

$$\begin{cases} \nabla^2 w_1 = f_1 & \text{in } \Omega_1 \\ w_1 = 0 & \text{on } \partial\Omega_1 \cap \partial\Omega \\ w_1 = u_\Gamma & \text{on } \Gamma \end{cases} \qquad \begin{cases} \nabla^2 w_2 = f_2 & \text{in } \Omega_2 \\ w_2 = 0 & \text{on } \partial\Omega_2 \cap \partial\Omega \\ w_2 = u_\Gamma & \text{on } \Gamma \end{cases}$$

Then $\begin{matrix} w_1 = u_1 \\ w_2 = u_2 \end{matrix} \iff \dfrac{\partial w_1}{\partial \hat{n}} = -\dfrac{\partial w_2}{\partial \hat{n}}$ on Γ

Since PDE is linear, we can write the w_i as a contribution from f_i and from u_Γ:

$$w_i = w_i^{source} + w_i^{harmonic}$$

where

$$\begin{cases} \nabla^2 w_i^{source} = f_i & \text{in } \Omega_i \\ w_i^{source} = 0 & \text{on } \partial\Omega_i \end{cases}$$

$$\begin{cases} \nabla^2 w_i^{harmonic} = 0 & \text{in } \Omega_i \\ w_i^{harmonic} = 0 & \text{on } \partial\Omega_i \cap \partial\Omega \\ w_i^{harmonic} = u_\Gamma & \text{on } \Gamma \end{cases}$$

$w_i^{harmonic}$ is called the "harmonic extension" of u_Γ into Ω_i, written $\mathcal{H}_i(u_\Gamma)$. We have yet to determine the u_Γ to satisfy (*).

For every function η that lives on Γ, define the operators

$$\Sigma_1 \eta = \frac{\partial}{\partial \hat{n}} \mathcal{H}_1(\eta)$$

$$\Sigma_2 \eta = \frac{\partial}{\partial \hat{n}} \mathcal{H}_2(\eta)$$

Σ_i is called the "Dirichlet-to-Neumann" map in Ω_i or the "local Poincare-Steklov operator." It takes in Dirichlet data (in η), computes the harmonic extension by solving a Laplace problem, then returns the normal derivative (Neumann data).

Define $\Sigma = \Sigma_1 + \Sigma_2$. Then

$$\Sigma \eta = \frac{\partial}{\partial \hat{n}} \mathcal{H}_1(\eta) + \frac{\partial}{\partial \hat{n}} \mathcal{H}_2(\eta).$$

Since $w_i = w_i^{\text{source}} + \mathcal{H}_i(u_\Gamma)$, then (*) holds iff

$$\Sigma u_\Gamma = z_\Gamma \qquad (+)$$

where

$$z_\Gamma = -\frac{\partial}{\partial \hat{n}} w_1^{\text{source}} - \frac{\partial}{\partial \hat{n}} w_2^{\text{source}}.$$

Σ is the "Poincare - Steklov operator."

(+) tells us the equation to solve for the glue so that we can solve for (*) separately on $\Omega_1 + \Omega_2$. In general, Σ is an elliptic operator that is often symmetric and positive definite.

From perspective ②: Variational form leads to discrete equations being satisfied at element interiors & boundaries. If unknowns are reordered so boundaries come last, a (linear algebraic) Schur complement can be taken to get a system for u_Γ, analagous to (+).

Discretization

$$Au = f \xrightarrow{\text{reorder}} \begin{bmatrix} A_{11} & & A_{1\Gamma} \\ & A_{22} & A_{2\Gamma} \\ A_{\Gamma 1} & A_{\Gamma 2} & A_{\Gamma\Gamma} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_\Gamma \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_\Gamma \end{bmatrix}$$

Taking Schur complement, can write A^{-1} as

$$A^{-1} = \begin{bmatrix} I & & -A_{11}^{-1} A_{1\Gamma} \\ & I & -A_{22}^{-1} A_{2\Gamma} \\ & & I \end{bmatrix} \begin{bmatrix} A_{11}^{-1} & & \\ & A_{22}^{-1} & \\ & & \Sigma^{-1} \end{bmatrix} \begin{bmatrix} I & & \\ & I & \\ -A_{\Gamma 1} A_{11}^{-1} & -A_{\Gamma 2} A_{22}^{-1} & I \end{bmatrix}$$

where discrete analog of operator Σ

$$\Sigma = A_{\Gamma\Gamma} - A_{\Gamma 1} A_{11}^{-1} A_{1\Gamma} - A_{\Gamma 2} A_{22}^{-1} A_{2\Gamma} \quad \text{is } n \times n.$$

So, can solve for $u = A^{-1} f$ via:

parallel { ① Solve subproblems: $A_{11} u_1^{\text{source}} = f_1$, $A_{22} u_2^{\text{source}} = f_2$ ← zero Dirichlet BCs

② Solve interface problem: $\Sigma u_\Gamma = f_\Gamma - A_{\Gamma 1} u_1^{\text{source}} - A_{\Gamma 2} u_2^{\text{source}}$

evaluate normal derivatives

Bottleneck

Dense

parallel { ③ Solve subproblems: $A_{11} u_1^{\text{harmonic}} = 0$, $A_{22} u_2^{\text{harmonic}} = 0$ ← u_Γ Dirichlet BCs

parallel { ④ Update solution: $u_1 = u_1^{\text{source}} + u_1^{\text{harmonic}}$

$u_2 = u_2^{\text{source}} + u_2^{\text{harmonic}}$

$$\Sigma u_\Gamma = \underbrace{f_\Gamma - A_{\Gamma 1} u_1^{\text{source}} - A_{\Gamma 2} u_2^{\text{source}}}_{z_\Gamma}$$

Note that we can apply Σ to a vector fast without explicitly constructing it since

$$\Sigma u_\Gamma = \Big(A_{\Gamma\Gamma} - \underbrace{A_{\Gamma 1} A_{11}^{-1} A_{1\Gamma}}_{\text{D2N map for } \Omega_1} - \underbrace{A_{\Gamma 2} A_{22}^{-1} A_{2\Gamma}}_{\text{D2N map for } \Omega_2}\Big) u_\Gamma$$

$$= A_{\Gamma\Gamma} u_\Gamma - \text{D2N}_1(u_\Gamma) - \text{D2N}_2(u_\Gamma)$$

where $\text{D2N}_i(u_\Gamma) =$ ① Solve $A_{ii} x_i = 0$ with u_Γ Dirichlet BC
② Evaluate the normal derivative of x_i on Γ

D2N_i takes $O(p^2 \log p)$ and $A_{\Gamma\Gamma} u_\Gamma$ takes $O(p^2)$.

We wish to solve $\Sigma u_\Gamma = z_\Gamma$ via an <u>iterative method</u>.