

Preconditioning I: Schwarz Methods

- Recall our principal result for the convergence of PCG:

$$\text{CG: } \# \text{ iterations} \sim \# \text{ digits} \times \sqrt{\kappa} \quad (1)$$

$$\kappa = \frac{\lambda_{\max}(M^{-1}A)}{\lambda_{\min}(M^{-1}A)} \quad (2)$$

- Clearly, if we can reduce the condition number of $(M^{-1}A)$ we have the potential to dramatically reduce the number of iterations and improve performance.
- Here, we consider **Overlapping Schwarz Methods**, which can be viewed as *generalized Block Jacobi* or *Block Gauss-Seidel*.
- We will alternate between a continuous (PDE) representation,

$$\mathcal{L}u = f \text{ in } \Omega, \quad (3)$$

and its discrete counterpart, $A\underline{u} = \underline{b}$.

- Think of $\mathcal{L}u := -\nabla^2 u$ + Dirichlet conditions on $\partial\Omega$, but other PDEs are possible.

- Consider the *Alternating Schwarz* iterative method, where we partition the domain into two overlapping domains, Ω_1 and Ω_2 , as illustrated in the two examples below.

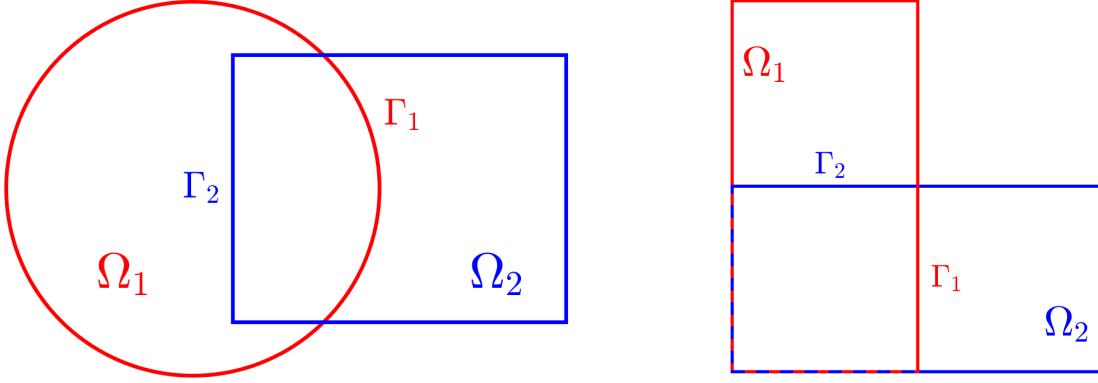


Figure 1: Domain decomposition examples following Schwarz [1871] and Lyons [1988].

- The alternating Schwarz algorithm works as follows,

for $k = 0 : k_{\max}$,

- (i) Solve $\mathcal{L}u_{k+\frac{1}{2}} = f$ in Ω_1 with $u_{k+\frac{1}{2}}|_{\partial\Omega_1} = u_k|_{\partial\Omega_1}$.
- (ii) Solve $\mathcal{L}u_{k+1} = f$ in Ω_2 with $u_k|_{\partial\Omega_2} = u_{k+\frac{1}{2}}|_{\partial\Omega_2}$.

end

- Let's consider the specific case, $\mathcal{L}\tilde{u} = -\nabla^2\tilde{u} = f$, $\tilde{u} = 0$ on $\partial\Omega$.
- Starting with $u_0(\mathbf{x}) = 0$, $\mathbf{x} \in \Omega$,

for $k = 0 : k_{\max}$,

$$(i) \quad -\nabla^2 u_{k+\frac{1}{2}} = f \text{ in } \Omega_1 \text{ with } u_{k+\frac{1}{2}} = u_k \text{ in } \Omega \setminus \Omega_1.$$

$$(ii) \quad -\nabla^2 u_{k+1} = f \text{ in } \Omega_2 \text{ with } u_{k+1} = u_{k+\frac{1}{2}} \text{ in } \Omega \setminus \Omega_2.$$

end

- At each subiteration, we have a BVP on Ω_s , $s = 1, \dots, S$, where $S = 2$ is the number of subdomains in this case.
- Notice that taking $u_{k+\frac{1}{2}} = u_k$ in $\Omega \setminus \Omega_1$ implies that the subdomain boundary condition, $u_{k+\frac{1}{2}} = u_k$ on $\partial\Omega_1$, is satisfied.
- This is our *algorithm*.
- To understand how it converges, we analyze the error behavior.

- Since $\mathcal{L}$ is *linear*, with exact solution $\tilde{u}$, we can write a formula for the error, $e_k := \tilde{u} - u_k$,

$$\begin{array}{rcl}
+ \mathcal{L}\tilde{u} & = & f \text{ in } \Omega_1, \quad \tilde{u} = \tilde{u} \text{ in } \Omega \setminus \Omega_1 \\
- \mathcal{L}u_{k+\frac{1}{2}} & = & f \text{ in } \Omega_1, \quad u_{k+\frac{1}{2}} = u_k \text{ in } \Omega \setminus \Omega_1 \\
\hline
\mathcal{L}e_{k+\frac{1}{2}} & = & 0 \text{ in } \Omega_1, \quad e_{k+\frac{1}{2}} = e_k \text{ in } \Omega \setminus \Omega_1
\end{array} \tag{4}$$

- Starting with $e_0 = \tilde{u}$ and repeating the same steps for e_{k+1} leads to the *error propagator*,

for $k = 0 : k_{\max}$,

$$(i) \quad -\nabla^2 e_{k+\frac{1}{2}} = 0 \text{ in } \Omega_1 \text{ with } e_{k+\frac{1}{2}} = e_k \text{ in } \Omega \setminus \Omega_1.$$

$$(ii) \quad -\nabla^2 e_{k+1} = 0 \text{ in } \Omega_2 \text{ with } e_{k+1} = e_{k+\frac{1}{2}} \text{ in } \Omega \setminus \Omega_2.$$

end

- Each subiteration is equivalent to solving Laplace's equation for the *error*.

- Laplace's equation satisfies a *maximum principle*, which means that the maximum value of the solution is found on $\partial\Omega_s$.
- Since $\Gamma_2 \subset \Omega_1$ and $\Gamma_1 \subset \Omega_2$, the error evolves as

$$e_{k+\frac{1}{2}}|_{\Gamma_2} < e_k|_{\Gamma_1} \quad (5)$$

$$e_{k+1}|_{\Gamma_1} < e_{k+\frac{1}{2}}|_{\Gamma_2} \quad (6)$$

- We can state further that, for $k \geq 0$,

$$\|e_{k+\frac{1}{2}}(\Omega_1)\|_{\infty} < \|e_k(\Omega_1)\|_{\infty} \quad (7)$$

$$\|e_{k+1}(\Omega_2)\|_{\infty} < \|e_{k+\frac{1}{2}}(\Omega_2)\|_{\infty}, \quad (8)$$

and, for $k \geq 1$,

$$\|e_{k+\frac{1}{2}}(\Omega)\|_{\infty} < \|e_k(\Omega)\|_{\infty} \quad (9)$$

$$\|e_{k+1}(\Omega)\|_{\infty} < \|e_{k+\frac{1}{2}}(\Omega)\|_{\infty}. \quad (10)$$

(The error might not be minimized on the first $\frac{1}{2}$ iteration. Why?)

- Because of the decay of solutions to Laplace's equations we might expect that the error for the Schwarz alternating method would decay faster if we have more overlap.
- In fact, this is precisely the type of behavior one observes in practice.

- The following 1D domain decomposition illustrates this idea.

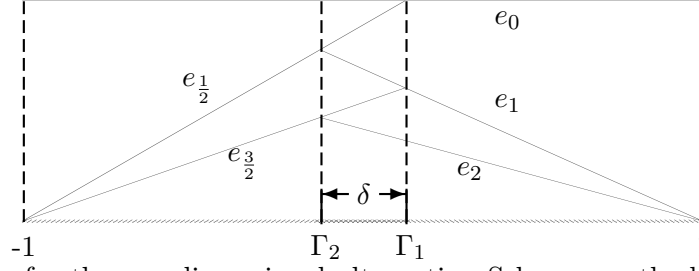


Figure 2: Error behavior for the one-dimensional alternating Schwarz method on $\Omega=(-1,1)$ with overlap $\delta=0.25$ with respect to $x = 0$.

- In this 1D example, the error satisfies, $e_0 = \tilde{u}$.

for $k = 0 : k_{\max}$,

$$(i) \quad -\frac{d^2 e_{k+\frac{1}{2}}}{dx^2} = 0 \text{ in } \Omega_1 \text{ with } e_{k+\frac{1}{2}} = e_k \text{ in } \Omega \setminus \Omega_1.$$

$$(ii) \quad -\frac{d^2 e_{k+1}}{dx^2} = 0 \text{ in } \Omega_2 \text{ with } e_{k+1} = e_{k+\frac{1}{2}} \text{ in } \Omega \setminus \Omega_2.$$

end

- In 1D, the Green's functions (solutions to the Laplace equation) are straight lines.
- **At each $\frac{1}{2}$ -iteration, the error is set by the subdomain boundary value on Γ_s .** (The error on $\partial\Omega = \pm 1$ is 0.)

- The geometry of the solution reveals that the overall error for $k > 1$ is

$$\|e_k\|_\infty = \left(\frac{1 - \delta/2}{1 + \delta/2} \right)^{2(k-1)} \|e_1\|_\infty, \quad (11)$$

from which the advantage of increased overlap, δ , is evident.

- Notice that this convergence result depends only on the decay rate of the Green's functions for the PDE and the amount of domain overlap, δ , but not on the discretization parameters, n or h .
- We can therefore expect that Schwarz methods will have iteration counts that are independent of n .
- The amount of work per iteration will of course depend on n .

- Demo: $n=200$; $istd=1$; $plotu=0$; `alt_schwarz_1d`

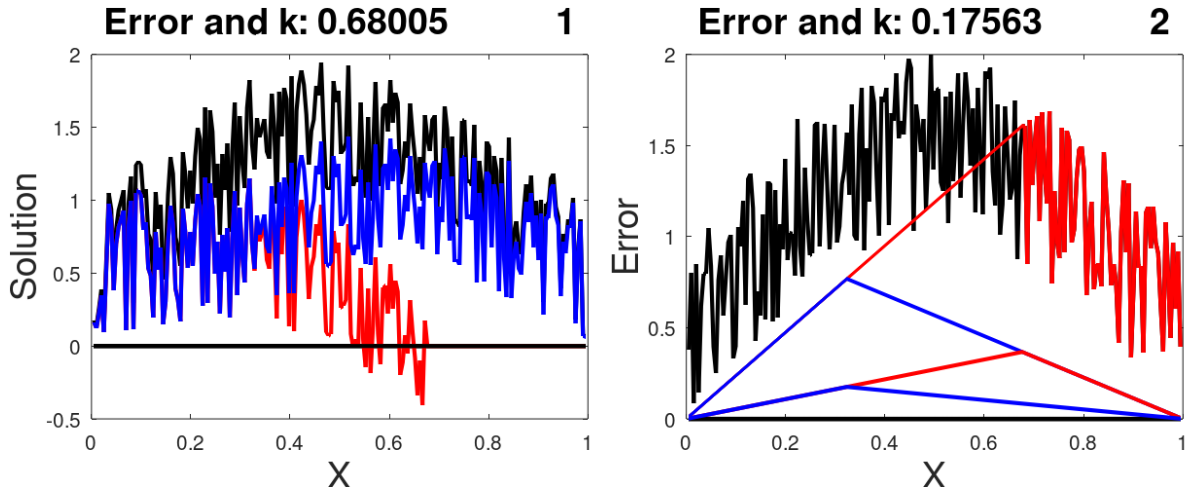


Figure 3: 1D alternating Schwarz example: (left) solution history and (right) error history.

Discrete Implementation of Alternating Schwarz

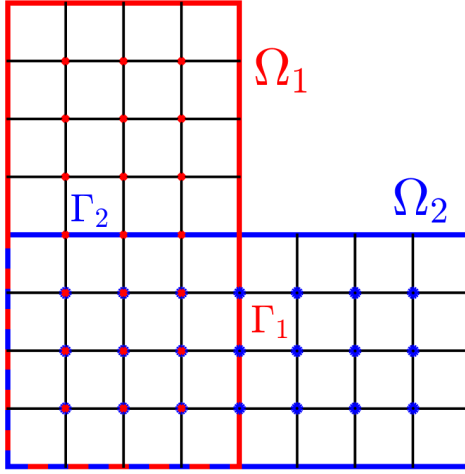


Figure 4: Two-domain decomposition example using finite differences.

- We now want to derive the continuous to discrete mapping,

$$\mathcal{L}u = f \text{ in } \Omega \longrightarrow A\underline{u} = \underline{b}. \quad (12)$$

- Let $\bar{\Omega}_s$ denote the closure of (overlapping) subdomain Ω_s , $s=1$ or 2 , and $\Omega_{12} = \Omega_1 \cap \Omega_2$ be the *interior* of the overlap region.

In the picture above, Ω_{12} has 9 grid points on a finite-difference grid.

- Assume that we enumerate degrees-of-freedom (dofs) in the following order,

- (i) interior to $\Omega_1 \setminus \Omega_{12}$ first,
- (ii) interior to Ω_{12} next, and
- (iii) interior to $\Omega_2 \setminus \Omega_{12}$ last.

- We define the *prolongation matrices*, R_s^T , $s = 1, 2$,

$$R_1^T := \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \\ \hline & & & & & 0 \end{bmatrix}, \quad R_2^T := \begin{bmatrix} & & & & 0 \\ \hline & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \\ & & & & & 1 \end{bmatrix}, \quad (13)$$

which takes a vector of dofs on Ω_s and extends it, *by zero*, to $\mathbb{R}^n$. (Here, we've truncated the matrices to simplify the illustration.)

- Let $\underline{u}_s$ denote the set of dofs on Ω_s (with n_s entries).
- We define the *prolongation* of $\underline{u}_s$ as

$$\bar{\underline{u}}_s = R_s^T \underline{u}_s. \quad (14)$$

It is a vector that accounts for all dofs in Ω .

- From the form of R_s^T we can see that $\bar{\underline{u}}_s$ is zero for all grid points in $\Omega \setminus \Omega_s$.
- By contrast, starting with a vector $\underline{u} \in \mathbb{R}^n$, the *restriction* of $\underline{u}$ to Ω_s is

$$\underline{u}_s = R_s \underline{u} \quad (15)$$

- R_s picks out those elements of $\underline{u}$ that belong to Ω_s , and discards the remainder.
- **Summary:**
 - R_s^T *extends by zero* vectors from Ω_s to Ω .
 - R_s *extracts* vectors from $\mathbb{R}^n$ to Ω_s .
- We will use these sparse restriction matrices to implement the Schwarz method.

- Recall that we have the following equivalence,

$$\mathcal{L}u = f \text{ in } \Omega \longrightarrow Au = \underline{b}, \quad (16)$$

where the discrete system could be derived by finite-differences or finite-elements (say).

- Define $\delta u_{k+\frac{1}{2}} = u_{k+\frac{1}{2}} - u_k$, which is defined on all of Ω , and which is identically zero in $\Omega \setminus \Omega_1$.
- Specifically, $\delta u_{k+\frac{1}{2}} = 0$ on Γ_1 .
- For alternating Schwarz, starting with $u_0 = 0$, we have

$$\mathcal{L}\delta u_{k+\frac{1}{2}} = f - \mathcal{L}u_k \text{ in } \Omega_1 \quad (17)$$

$$\delta u_{k+\frac{1}{2}} = 0 \text{ in } \Omega \setminus \Omega_1. \quad (18)$$

- The boundary condition on $\partial\Omega$ and Γ_1 is

$$\delta u_{\frac{1}{2}} = 0, \quad (19)$$

i.e., *homogeneous Dirichlet*, which is very convenient.

- For the case of Poisson (or any other PDE), the system that governs (17) and ultimately $u_{k+\frac{1}{2}}$, is

$$A_1 \delta \underline{u}_{k+\frac{1}{2}} = R_1 (\underline{f} - A \bar{\underline{u}}_k) \quad (20)$$

$$\delta \bar{\underline{u}}_{k+\frac{1}{2}} = R_1^T \delta \underline{u}_{k+\frac{1}{2}} \quad (21)$$

$$\bar{\underline{u}}_{k+\frac{1}{2}} = \bar{\underline{u}}_k + \delta \bar{\underline{u}}_{k+\frac{1}{2}}. \quad (22)$$

- Here, $A_s := R_s A R_s^T$ is the system matrix *restricted to* Ω_s .
- A_1 is the leading principal $n_1 \times n_1$ submatrix of A .

$$\left[\begin{bmatrix} A_1 \end{bmatrix} \right] \quad \left[\begin{bmatrix} A_2 \end{bmatrix} \right]$$

- A_2 is the trailing principal $n_2 \times n_2$ submatrix of A .
- If A is SPD, then so is A_s .

- With these tools, we can write down the *Alternating Schwarz Method*,

$$\underline{\bar{u}}_0 = 0$$

for $k = 0 : k_{\max}$

$$\delta \underline{u}_{k+\frac{1}{2}} = A_1^{-1} R_1 (\underline{f} - A \underline{\bar{u}}_k)$$

$$\underline{\bar{u}}_{k+\frac{1}{2}} = \underline{\bar{u}}_k + R_1^T \delta \underline{u}_{k+\frac{1}{2}}$$

$$\delta \underline{u}_{k+1} = A_2^{-1} R_2 \left(\underline{f} - A \underline{\bar{u}}_{k+\frac{1}{2}} \right)$$

$$\underline{\bar{u}}_{k+1} = \underline{\bar{u}}_{k+\frac{1}{2}} + R_2^T \delta \underline{u}_{k+1}$$

end

- This system is convergent by virtue of the *maximum principle*.
- For each iteration, k , we have *two* systems to solve.
- Each system, however, is smaller ($n_s < n$) and if the costs of solving a system are superlinear in n , then there is potential for significant savings if the number of iterations is relatively small.

Alternating Schwarz Viewed as Projectors

- Let's begin by looking at a full step of the ASM.
- We start with the last line of the preceding algorithm and make substitutions:

$$\bar{u}_{k+1} = \underbrace{\bar{u}_k + R_1^T A_1^{-1} R_1 \overbrace{\left(\underline{f} - A \bar{u}_k \right)}^{r_k}}_{\bar{u}_{k+\frac{1}{2}}} + R_2^T A_2^{-1} R_2 \underbrace{\left(\underline{f} - A \bar{u}_{k+\frac{1}{2}} \right)}_{r_{k+\frac{1}{2}}} \quad (23)$$

- We now make the substitution for $\bar{u}_{k+\frac{1}{2}}$,

$$\begin{aligned} \bar{u}_{k+1} = \bar{u}_k + R_1^T A_1^{-1} R_1 \left(\underline{f} - A \bar{u}_k \right) \\ + R_2^T A_2^{-1} R_2 \left[\underline{f} - A \bar{u}_k - A R_1^T A_1^{-1} R_1 \left(\underline{f} - A \bar{u}_k \right) \right] \end{aligned} \quad (24)$$

- As usual, we have $\underline{f} - A \bar{u}_k \equiv A \underline{e}_k$.
- Subtracting each side of (24) from the solution, $\underline{u}$, yields the error propagation relationship,

$$\underline{e}_{k+1} = \underline{e}_k - R_1^T A_1^{-1} R_1 A \underline{e}_k - R_2^T A_2^{-1} R_2 \left[A \underline{e}_k - A R_1^T A_1^{-1} R_1 A \underline{e}_k \right] \quad (25)$$

$$= \underline{e}_k - R_1^T A_1^{-1} R_1 A \underline{e}_k - R_2^T A_2^{-1} R_2 A \underline{e}_k + R_2^T A_2^{-1} R_2 A R_1^T A_1^{-1} R_1 A \underline{e}_k \quad (26)$$

$$= (I - R_2^T A_2^{-1} R_2 A) (I - R_1^T A_1^{-1} R_1 A) \underline{e}_k \quad (27)$$

$$= (I - P_2) (I - P_1) \underline{e}_k. \quad (28)$$

- Here, we have introduced the *A-orthogonal projector* onto $\mathcal{R}(R_s^T)$,

$$P_s = R_s^T A_s^{-1} R_s A, \quad (29)$$

and the projector onto $\mathcal{R}^\perp(R_s^T)$, the orthogonal complement of $\mathcal{R}(R_s^T)$,

$$I - P_s = I - R_s^T A_s^{-1} R_s A. \quad (30)$$

- From (28), we see that the alternating Schwarz error is successively projected onto the complement of the range of R_s^T , $s = 1, 2$.

- Let's look more closely at the error behavior.
- The first step computes

$$\bar{u}_{\frac{1}{2}} = P_1 \underline{u} = R_1^T A_1^{-1} R_1 A \underline{u}. \quad (31)$$

- $\bar{u}_{\frac{1}{2}}$ is thus the *closest* element of $\mathcal{R}(R_1^T)$ to $\underline{u}$ (in the A -norm).
- It differs from $\underline{u}$ in that it has the wrong values on the interior boundary, Γ_1 .
 - $\bar{u}_{\frac{1}{2}}$ is 0 on Γ_1 , rather than being equal to $\underline{u}$ there.
 - In fact, that boundary difference is the sole source of error at the end of iteration $\frac{1}{2}$.
 - Because the *error* satisfies (a discrete) Laplace's equation, it decays smoothly from Γ_1 to the interior of Ω_1 .
 - On the second half of the alternating Schwarz iteration, $\bar{u}_1$ suffers from having the wrong value on Γ_2 .
- This procedure continues iteratively, with decaying error at each step.

- For reasons that are relatively clear, the alternating Schwarz method is viewed as a *multiplicative* method: the error is first multiplied by $(I - P_1)$, then by $(I - P_2)$.
- It is also referred to as *Gauss-Seidel-like*.
- Naturally, there is a complementary Jacobi-like method, also known as *additive Schwarz*.
- We derive that by first noting that we can define a multiplicative-Schwarz preconditioner through the relationship,

$$I - M_{MS}^{-1}A := (I - P_2)(I - P_1) = I - P_1 - P_2 + P_2P_1. \quad (32)$$

- This suggests the idea of an *additive Schwarz* preconditioner, due originally to Dryja & Widlund [87],

$$I - M_{AS}^{-1}A := I - P_1 - P_2. \quad (33)$$

- The algorithm for *Additive Schwarz* is,

```

 $\bar{u}_0 = 0$ 
for  $k = 0 : k_{\max}$ 
   $\delta \underline{u}_{k+\frac{1}{2}} = A_1^{-1} R_1 (\underline{f} - A \bar{u}_k)$ 
   $\bar{u}_{k+\frac{1}{2}} = \bar{u}_k + R_1^T \delta \underline{u}_{k+\frac{1}{2}}$ 
   $\delta \underline{u}_{k+1} = A_2^{-1} R_2 (\underline{f} - A \bar{u}_k) \quad \leftarrow \text{vs. } \left( \underline{f} - A \bar{u}_{k+\frac{1}{2}} \right)$ 
   $\bar{u}_{k+1} = \bar{u}_{k+\frac{1}{2}} + R_2^T \delta \underline{u}_{k+1}$ 
end

```

- It differs from alternating (multiplicative) Schwarz in that it uses the prior iterate, $\bar{u}_k$, in both subdomain updates, rather than using the update on Ω_1 when computing the update on Ω_2 .
- In this sense, additive Schwarz is Jacobi-like and, crucially, it is fully parallel because all subdomains can update simultaneously.

- Another important feature of additive Schwarz is that it is SPD if A is, which is not true of multiplicative Schwarz.
- Thus, additive Schwarz (AS) can be used directly with PCG.
- It is possible, however, to symmetrize multiplicative Schwarz just as can be done with GS.

- Finally, it's important to note that the error propagator for additive Schwarz,

$$I - M_{AS}^{-1}A := I - P_1 - P_2, \quad (34)$$

is **not** a contractor.

- You must embed AS in a KSP or other accelerator to guarantee convergence.
- Multiplicative Schwarz, however, *is* a contractor. It can converge as a stand-alone method, but of course benefits from being used with a KSP.

Schwarz Preconditioning

- In the context of PCG, we have the preconditioner step,

$$\text{Solve } M\underline{z} = \underline{r} \quad (35)$$

- Assuming that we have S overlapping subdomains, Ω_s , that cover all of Ω , the additive Schwarz preconditioner effects the action of M_{AS}^{-1} on the residual vector, $\underline{r}$.
- The mathematical formula for AS is

$$\underline{z} = M^{-1}\underline{r} = \sum_{s=1}^S R_s^T A_s^{-1} R_s \underline{r}. \quad (36)$$

- Using the relationship $\underline{r} = A\underline{e}$, we see that the preconditioner is the sum of S projections,

$$\underline{z} = \sum_{s=1}^S R_s^T A_s^{-1} R_s A \underline{e} = \sum_{s=1}^S P_s \underline{e}. \quad (37)$$

- The eigenvalues of a projector are 1 and 0 (because $P^2 = P \implies \lambda^2 = \lambda$, which implies $\lambda = 1$ or 0).
- Consider the matrix $Q = P_1 + P_2$.
- If there is a vector $\underline{z} \in \mathcal{R}(P_1)$ and $\underline{z} \in \mathcal{R}(P_2)$, then

$$Q\underline{z} = 1 \cdot \underline{z} + 1 \cdot \underline{z} = 2 \cdot \underline{z}, \quad (38)$$

so the maximum eigenvalue of the sum of two projectors is bounded above by 2.

- Consider now the matrix $Q = \sum_{s=1}^S P_s = M^{-1}A$.
- By the same reasoning, its maximum possible eigenvalue is S .
- However, it may be possible to color the domains into c *disjoint sets*, $\mathcal{I}_1, \dots, \mathcal{I}_C$ such that, for any $i, j \in \mathcal{I}_c$, $\Omega_i \cap \Omega_j = \emptyset$, for $c = 1, \dots, C$.
- In this case, the maximum eigenvalue of $P_i + P_j$ is 1 because there is no vector $\underline{z} \in \mathcal{R}(P_1)$ and $\underline{z} \in \mathcal{R}(P_2)$.
- The maximum eigenvalue of Q is thus $\leq C$ —the minimum number of colors required to color the subdomains into disjoint sets.
(In 1D, $C = 2$.)
- Note that the user does not need to find this coloring. The algorithm will automatically inherit C as a property that is intrinsic to the given decomposition, $\Omega = \bigcup \Omega_s$.

- An example of a (bad) decomposition where $C = S$ is a pizza-pie decomposition of a disk, where S wedge-shaped subdomains touch at the disk center. In this case, with any amount of overlap $\delta > 0$, any extended subdomain will touch the other $S - 1$ subdomains and we would need $C = S$ colors to identify disjoint sets (i.e., there are in fact, no disjoint sets). In this case, the maximum eigenvalue is $\leq C = S$. (The value C is an upper bound, but not necessarily a sharp upper bound.)
- Note that one can construct Q as follows.

```

for  $j = 1 : n$ 
     $\underline{u} = 0, \quad u_j = 1$ 
     $\underline{w} = A\underline{u}$ 
     $\underline{q}_j = \text{AS}(\underline{w})$ 
end

```

- It's thus possible to construct (for testing purposes) $M^{-1}A$ just through function calls.
- You could also simply set $\underline{w} = \underline{a}_j$ in the above loop.

QUIZ / HW:

For all questions, discuss your findings/observations. Don't just turn in a plot without putting some thought into what you are submitting.

Answer 1., 2., and 6. as quiz questions for the next class.

Answer the remaining questions for HW.

1. What are two advantages of AS?
2. What are some advantages of MS?
3. Take a coloring and find the upper bound on $\lambda(Q)$. (This is a computational exercise—form Q and then ask matlab to provide the upper bound on $\lambda(Q)$.)
4. Solve (computationally) the 1D alternating Schwarz problem for Poisson for which the exact solution is 1 in Ω . Try a variety of overlaps. Plot the error distribution on successive iterations as in the example in the notes for a few different overlaps. On a separate semilogy graph, plot the convergence of the computed error vs. iteration k for these cases using solid lines of different colors. On the same graph, plot the theoretical estimate given in the notes.
5. Repeat the above for 1D alternating Schwarz exercise for advection-diffusion equation

$$-\nu \frac{d^2 u}{dx^2} + cu = f, \quad u(0) = u(1) = 0. \quad (39)$$

Choose $\underline{f} = L\underline{u}$ with $\underline{u} = 1$, so that your initial error is 1. Answer the same questions (save for the theory part, which is tractable, but tedious).

6. Suppose the minimum number of colors to color a given domain decomposition is C . What is your best estimate for the spectral radius of the additive Schwarz error propagator?
7. Implement additive Schwarz preconditioning in PCG for the 2D Poisson problem. Take your subdomains to be squares that are subsets of your originating $m \times m$ grid. Note that rather than multiplying A by R_s^T and R_s , you can simply extract the rows and columns from A according to the given subdomain. This might be faster than using sparse matrix-matrix multiplies.
 - Choose a 4×4 array of overlapping subdomains with, say, overlap containing 2 neighboring points in each direction.
 - Starting with a random solution, u_0 , make a mesh plot of e_0 , e_1 , and e_{10} .
 - For the case $m=50$, plot the PCG residual history with AS preconditioning and without AS preconditioning.
 - For the case $m=100$, plot the PCG residual history with AS preconditioning and without AS preconditioning. (Use a semilogy plot.)
8. Repeat the preceding question with a coarse-grid term added, $\underline{z}_0 = R_0^T A_0^{-1} R_0 \underline{x}$, where R_0^T is an *interpolator* from a coarse grid of size H (the subdomain size) to the fine grid, h .

Coarse-Grid Spaces

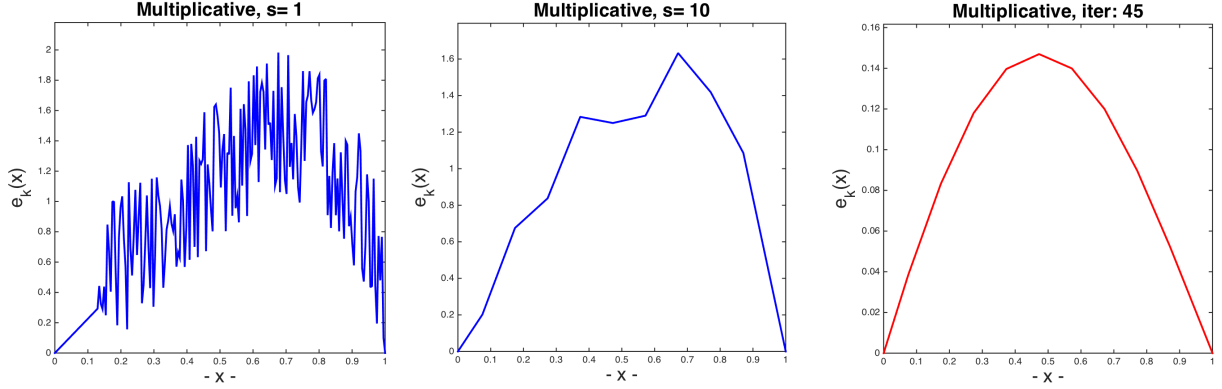


Figure 5: Error behavior for 1D Multiplicative Schwarz without coarse-grid correction.

Basic Concept

- As illustrated in the plots above, the multiplicative Schwarz iteration eliminates the fine-scale $O(h)$ -error after a single pass.
- Without KSP (e.g., GMRES) acceleration, the error decay is governed by the most slowly decaying eigenmode, as shown in the rightmost figure.
- Note that it is the *error* that is smooth in the plots above. This particular solution is in fact very rough.
- The basic (but not efficient!) algorithm shown here is

```

u = 0; r = b
for k = 1 : kmax
    {
        Multiplicative overlapping Schwarz: Mz = r
        z = 0; tilde r = r
        for s = 1 : S
            z = z + RsT As-1 Rs tilde r
            tilde r = r - Az
        end
        u = u + z
        r = r - Az
    end
end

```

- The idea of a coarse-space correction is to approximate the solution (or, more precisely, the *error*), to the underlying system with a *coarse function* that is represented by a few degrees-of-freedom (and therefore easily solved).
- The coarse-space should represent functions on a scale, H , that is commensurate with the subdomain size in the overlapping Schwarz space
- Recall, $H > h$ and $H > \delta$ (the subdomain-overlap parameter).
- To construct the coarse space, we seek a set of interpolating functions that will map from a coarse representation to the originating set of grid points, which we often call the *fine grid*.
- We start with the space of *piecewise linear* interpolants for the 1D example of the figure above.
- Let $\underline{u}_c = [u_0^c \ u_1^c \ \dots \ u_{n_c}^c]^T$ be a set of coarse-grid degrees-of-freedom associated with coarse nodes,

$$x_j^c = j H, \quad j = 0, \dots, n_c, \quad H := L/n_c, \quad (40)$$

where we assume that the 1D domain length is L .

- The interpolant to the *fine mesh*, $\underline{\bar{x}} = [x_0 \ x_1 \ \dots \ x_{n+1}]^T$, can of course be defined as

$$u_i = u_j^c + (u_{j+1}^c - u_j^c) \frac{x_i - x_j^c}{H}, \quad \text{for } x_j^c \leq x_i \leq x_{j+1}^c. \quad (41)$$

- However, this approach does not readily fit into our linear-algebra formalism.
- So, we instead seek to define the interpolation problem as a matrix-vector product, $\underline{u}_f = J \underline{u}_c$, where J is an *interpolation matrix* that maps $\underline{u}_c \longrightarrow \underline{u}_f$.

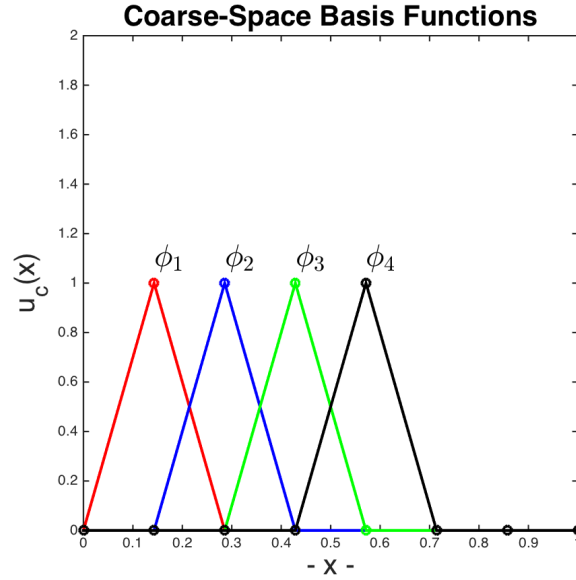


Figure 6: Coarse-space basis functions.

- Begin by introducing *coarse basis functions*, $\phi_j(x)$, such as those illustrated in the figure above.
- Here, we have:

$$\phi_j(x) = \begin{cases} \frac{x-x_{j-1}^c}{x_j^c-x_{j-1}^c}, & x_{j-1}^c \leq x < x_j^c \\ \frac{x_{j+1}^c-x}{x_{j+1}^c-x_j^c}, & x_j^c \leq x < x_{j+1}^c \\ 0 & \text{otherwise} \end{cases} . \quad (42)$$

- Notice that each of the ϕ_j s is defined on the *entire domain*, Ω .
- We can consequently sum them to define a *coarse function*,

$$u_c(x) = \sum_{j=0}^{n_c+1} u_j^c \phi_j(x). \quad (43)$$

- In particular, at the originating (fine) mesh points, x_i , we have:

$$u_c(x_i) = \sum_{j=0}^{n_c+1} u_j^c \phi_j(x_i) = \sum_{j=0}^{n_c+1} J_{ij} u_j^c, \quad (44)$$

where

$$J_{ij} := \phi_j(x_i). \quad (45)$$

is the interpolation matrix.

- We often refer to J as a *prolongation matrix* or *coarse-grid prolongation operator*.
- It allows us to generate the coarse-to-fine map, $\underline{u}_c \longrightarrow \underline{u}_f$: $\underline{u}_f = J\underline{u}_c$.
- Note that vectors $\underline{u}_c \in \mathcal{R}(J)$ are equivalent to functions $u_c \in \{\phi_0 \dots \phi_{n_c+1}\}$.
- We define the (small, of size n_c) coarse-grid matrix, $A_c = J^T A J$.
- With these tools, we have the *coarse-grid projection* of the error, $\underline{e}$, onto $\mathcal{R}(J)$,

$$\underline{e}_c = J A_c^{-1} J^T A \underline{e} = J A_c^{-1} J^T \underline{r}, \quad (46)$$

which is *computable*!

- As is standard with projection operators, $\underline{e}_c$ is the closest element of $\mathcal{R}(J)$ to the unknown error $\underline{e}$, in the A -norm.

- We can incorporate a coarse-grid correction into the multiplicative Schwarz algorithm in either an additive or multiplicative fashion.
- Here, we consider the multiplicative approach, which means we are acting on the *updated residual*, after the overlapping correction has been removed.

```

u = 0; r = b
for  $k = 1 : k_{\max}$ 
    {
        Multiplicative overlapping Schwarz:  $M\underline{z} = \underline{r}$ 
         $\underline{z} = 0$ ;  $\tilde{\underline{r}} = \underline{r}$ 
        for  $s = 1 : S$ 
             $\underline{z} = \underline{z} + R_s^T A_s^{-1} R_s \tilde{\underline{r}}$ 
             $\tilde{\underline{r}} = \underline{r} - A\underline{z}$ 
        end
         $\underline{z} = \underline{z} + JA_c^{-1} J^T \tilde{\underline{r}}$  (coarse-grid correction)
         $\underline{u} = \underline{u} + \underline{z}$ 
         $\underline{r} = \underline{r} - A\underline{z}$ 
    end
end

```

- Note that one would not update $\tilde{\underline{r}}$ as $\tilde{\underline{r}} = \underline{r} - A\underline{z}$ in practice because most of $\underline{z}$ has not been changed by the $R_s^T A_s^{-1} R_s$ update.
- One would instead just update the degrees-of-freedom and residual values in the neighborhood of Ω_s .
- Also, it is most common to embed the Schwarz iteration into a KSP for robustness and faster convergence.

- We next illustrate the additive method with a coarse-grid correction.

```

u = 0; r = b
for  $k = 1 : k_{\max}$ 
    { Additive overlapping Schwarz:  $M\underline{z} = \underline{r}$ 
       $\underline{z} = 0$ 
      for  $s = 1 : S$ 
           $\underline{z} = \underline{z} + R_s^T A_s^{-1} R_s \underline{r}$ 
      end
       $\underline{z} = \underline{z} + J A_c^{-1} J^T \underline{r}$  (coarse-grid correction)
      u = u + z
      r = r - Az
    end

```

- Note that we don't recompute the intermediate residuals in this case, which makes the additive method faster per iteration.
- However, this scheme *will not converge* without being embedded in a KSP.
- Because the additive method is SPD if A is, PCG is the ideal KSP because it has a lower storage and work complexity than GMRES.

- If we define $R_0^T := J$ and $A_0 := R_0 A R_0^T$, additive Schwarz with a coarse grid has a compact representation,

$$\underline{z} = M_{AS}^{-1} \underline{r} = \sum_{j=0}^S R_s^T A_s^{-1} R_s \underline{r}. \quad (47)$$

- In practice, one would precompute the LU (or other, perhaps approximate) factorization of each A_s and amortize this setup cost over each iteration.
- Also, *exact* solves are not required—one can use approximate (cheap) solves both for the local and global (coarse-grid) problems.

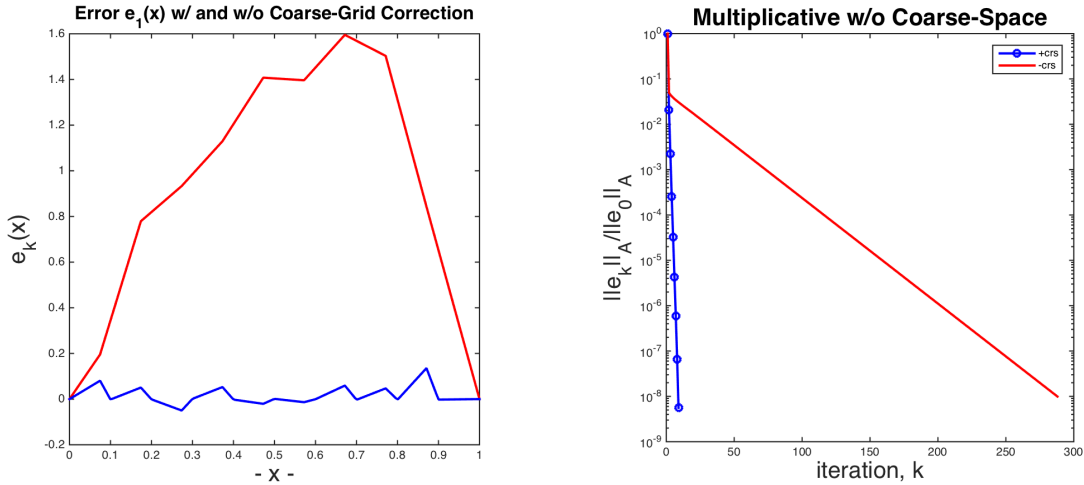


Figure 7: Error behavior for 1D Multiplicative Schwarz with and without coarse-grid correction.

- In the accompanying figures we see how using the coarse-grid correction helps the 1D multiplicative convergence of the preceding example.
- The left figure shows the error after the first iteration (i.e., first application of M^{-1}) for multiplicative Schwarz without (red) and with (blue) the coarse-grid correction.
- The improvement is impressive.
- In this 1D case, that improvement leads to a dramatic reduction in iteration counts, as is evident in the figure on the right.

Questions you could explore.

- How does additive compare with multiplicative? (What do you need to make that work?)
- How much faster would this be with a KSP?
- What about extensions to higher space dimensions and complex domains?
- How does one generate a coarse-grid space (i.e., operator) for unstructured problems?
- etc.

Main Schwarz Theoretical Results

- Recall that for AS we have local projectors, $P_s := R_s^T A_s^{-1} R_s A$ and

$$P_{AS} = M^{-1}A = P_1 + P_2 + \dots + P_S. \quad (48)$$

- If the domain decomposition can be colored in N_C colors then $\rho(M^{-1}A) \leq N_C$.
- If we add a coarse-space, then that space covers *all* of the subdomains and we must add 1 to the upper bound on $\rho(M^{-1}A)$.
- That is, with the coarse space, we have

$$P_{AS} = M^{-1}A = \sum_{s=0}^S P_s \quad (49)$$

and $\rho(M^{-1}A) \leq N_C + 1$.

- The lower bound is more difficult to derive.
- Then main point depends on identifying (in the FEM context) coercivity constants that are rarely provided in the literature.
- Here, we simply summarize the principal (and most important) result.
- *Assuming that the subdomains are shape-regular, then preconditioning the Poisson problem with additive Schwarz having subdomain diameter $\sim H$, overlap $\sim \delta$ and an additive coarse space with grid scale $\sim H$ leads to a condition number*

$$\kappa(M^{-1}A) = C \left(1 + \frac{H}{\delta} \right), \quad (50)$$

where C depends on N_C but is otherwise independent of H , δ , and h .

[Toselli and Widlund, Thm 3.13].

<https://link.springer.com/content/pdf/10.1007/b137868.pdf>

- Thus, we have the significant result that the number of PCG iterations with two-level AS should be bounded, independent of problem size, n .

Restricted Additive Schwarz [M. Sarkis and X.C. Cai, 97]

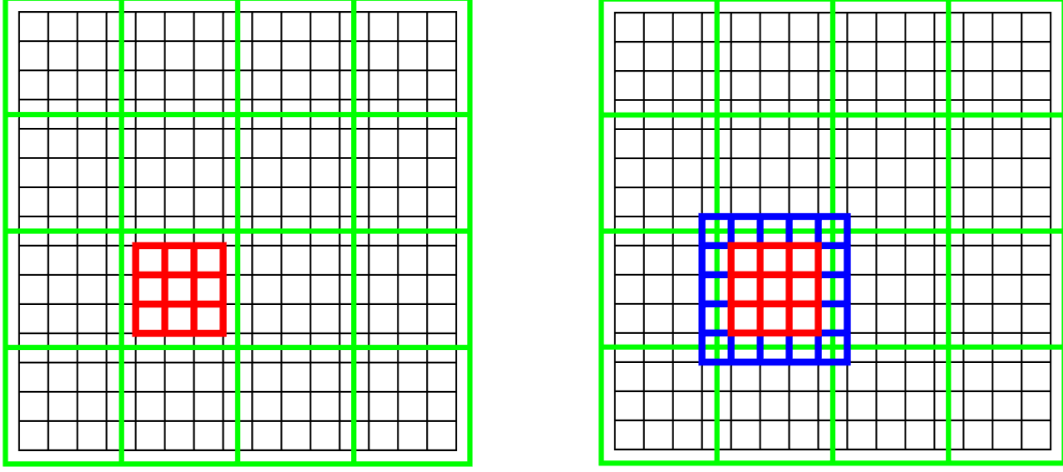


Figure 8: Domain-decomposition example highlighting (left) single subdomain in red and (right) its overlapping extension in blue.

- Consider the following implementation of the local part of overlapping additive Schwarz on just one extended subdomain, $\bar{\Omega}_s$,

$$\underline{z}_s = R_s^T A_s^{-1} R_s \underline{r} \quad (51)$$

- Suppose that we have a decomposition of a domain in which we assume that all degrees-of-freedom assigned to a given processor, p , are assigned to a single subdomain, Ω_s .
- To effect the restriction, $\underline{r}_s := R_s \underline{r}$, we need to gather residual values from neighboring processors into the extended domain, as illustrate in the figure on the right.
- Once $\underline{r}_s$ is formed, we can solve the local problems, $\bar{\underline{z}}_s := A_s^{-1} \underline{r}_s$, in parallel, on all processors, p .
- After that step is complete, we send the values of $\bar{\underline{z}}_s$ that are exterior to Ω_s *back* to their originating processors, where the solution is added to form the local components of $\underline{z}$.
- By reciprocity, processor p will also be receiving values from its neighbors and adding them together in the overlap region.
- Additive Schwarz thus requires two communication phases to implement $\underline{z} = M^{-1} \underline{r}$.

- Marcus Sarkis and Xiao-Chuan Cai (97) developed a nonsymmetric, *restricted additive Schwarz* (RAS) method in which one *does not* perform the exchange-and-sum step after the solve is complete.
- Instead, each processor (or subdomain) simply retains its own local result and declares that to be the local output of the preconditioner.
- Mathematically, we write RAS as

$$\underline{z}_s = \tilde{R}_s^T A_s^{-1} R_s \underline{r}, \quad (52)$$

where $\tilde{R}_s^T \underline{\bar{z}}_s$ simply retains the interior values of $\underline{\bar{z}}_s$ and sets to 0 any grid point values outside of Ω_s .

- From the maximum principle, we can understand why this method is attractive: *We do not retain any values that are near the extended domain boundary, $\overline{\partial\Omega}$, which is where the local value of $\underline{z} - \underline{e}$ is maximal.*
- Another advantage is that RAS requires less communication.
- The principal downside of RAS is that one can no longer use CG.
- We will see in the examples why this might be attractive, but one generally has to test on a case-by-case basis to determine if RAS is advantageous compared to the classical symmetric AS.

Why Additive Schwarz Works

Here, we consider the influence of the coarse-grid correction in an *additive Schwarz* context.

Consider, say, a finite-difference Poisson operator, A , on $\Omega = [0, L]^d$ in d space dimensions with homogeneous Diriclet conditions. (The finite-element case would be analogous.) Let the mesh size be $h = L/N$, such that A is $n \times n$ with $n = (N - 1)^d$. Suppose we also have a coarse-grid space of size H in each direction with J an interpolation matrix from $x_j^c = j \cdot H$ to $x_i = i \cdot h$. The discrete fine-space representations of coarse-space functions of functions on the coarse points are given by $\underline{u} = J\underline{u}^c$.

Consider the additive two-level preconditioner defined by

$$M^{-1}A := (\gamma J A_c^{-1} J^T + D^{-1}) A, \quad (53)$$

with $A_c := J^T A J$ and $D = \text{diag}(a_{ii})$ being the diagonal of A . Evaluation of $M^{-1}\underline{r}$ amounts to Jacobi- (diagonal-) scaling of the residual vector $\underline{r}$ followed by the addition of the coarse-grid correction $\underline{u}^c := J A_c^{-1} J^T \underline{r}$, scaled by γ .

To understand the importance of the coarse-grid solve, we look at the spectrum of $M^{-1}A$. To make things concise, we'll consider the one-dimensional case ($d=1$), for which $A = \text{tridiag}(-1, 2, -1)/h^2$. If $\gamma = 0$, we have simple Jacobi scaling. We know that the eigenvalues of $D^{-1}A$ are on $[0, 2]$ by Gershgorin's disc theorem. However, we need a bit more precision to determine the condition number,

$$\kappa := \frac{\lambda_{\max}(D^{-1}A)}{\lambda_{\min}(D^{-1}A)}, \quad (54)$$

since the lowest eigenvalue might be arbitrarily close to zero.

In the 1D case, the closed-form expression for the eigenvalues, $\lambda_k(D^{-1}A)$, $k = 1, \dots, n$, is

$$\lambda_k(D^{-1}A) = 1 - \cos \theta_k, \quad (55)$$

with $\theta_k := \pi k / (n + 1) = \pi k h / L$. We plot these values in Fig. ?? . It is clear that the ill-conditioning arises here because, with the diagonal scaling, the smallest eigenvalues can be arbitrarily close to zero.

The asymptotic result for $n \gg 1$ is

$$\kappa(D^{-1}A) \sim 4n^2. \quad (56)$$

Multi-Dimensional Interpolation

Basic Concept

- One way to construct a *coarse space* is to identify an interpolation matrix that will map from a reduced and nearly arbitrarily-spaced set of grid point values onto the fine-scale grid point values. Note that the fine scale points are typically unstructured—they are often positioned to optimally resolve some complex geometry and multiscale solution. That requirement, however, is relaxed when it comes to developing a *coarse-grid* operator. According to the Schwarz theory, we require only that the coarse grid be at a scale that is $\approx H$, which is a characteristic size associated with the diameter of the Schwarz overlapping subdomains. (H might be variable.) Aside from that constraint, and the fact that one should respect an appropriate decomposition of $\partial\Omega_c$ (on this new, coarse, mesh) into homogeneous Neumann/Dirichlet subsets, one is at liberty to pick a set of interpolating functions for which the basis coefficients are relatively easy to solve. (In fact, it is likely that none of the nodal values on the coarse mesh need to be restricted to zero at Dirichlet points because that restriction will be imposed on the *fine-mesh* solution.)
- Here, we consider *bilinear* interpolation in 2D (trilinear in 3D).
- For the moment, we'll assume that our chosen coarse-space is a tensor-product of 1D piecewise linear interpolants,

$$p_i(x) = \begin{cases} \frac{x-x_{i-1}^c}{x_i^c-x_{i-1}^c}, & x_{i-1}^c \leq x < x_i^c \\ \frac{x_{i+1}^c-x}{x_{i+1}^c-x_i^c}, & x_i^c \leq x < x_{i+1}^c \\ 0 & \text{otherwise} \end{cases} \quad (57)$$

$$q_j(y) = \begin{cases} \frac{y-y_{j-1}^c}{y_j^c-y_{j-1}^c}, & y_{j-1}^c \leq y < y_j^c \\ \frac{y_{j+1}^c-y}{y_{j+1}^c-y_j^c}, & y_j^c \leq y < y_{j+1}^c \\ 0 & \text{otherwise} \end{cases} \quad (58)$$

such that, at any point (x, y) ,

$$u_c(x, y) = \sum_{j=1}^{n_{c,y}} \sum_{i=1}^{n_{c,x}} q_j(y) p_i(x) u_{c,ij}. \quad (59)$$

- For notational convenience, we also define

$$\psi_k(\mathbf{x}) = q_j(y)p_i(x), \quad (60)$$

with $k = i + n_{c,x}(j - 1)$ representing a lexicographical ordering of the coarse-space nodes, (x_i^c, y_j^c) .

- Let $\mathbf{x}_l$, $l = 1, \dots, n$ represent the nodal points for the fine-scale solution and define the *interpolation matrix*,

$$J_{lk} = \psi_k(\mathbf{x}_l). \quad (61)$$

- We can then define a coarse-grid correction as

$$\underline{z}_c = J_c A_c^{-1} J_c^T \underline{r}, \quad (62)$$

where $A_c = J_c^T A J_c$.

- If A is SPD, then the solution $\underline{z}_c$ will be the *closest element of $\mathcal{R}(J_c)$* to the current error, $\underline{e}$, in the A -norm.
- Notice that A_c will not be invertible if there are coarse-grid points whose interpolative support does not intersect Ω (i.e., $\psi_k(\mathbf{x}_l) = 0$ for every $\mathbf{x}_l$).
- In this case, one can simply set $(A_c)_{kk} = 1$ and the result will be void since the right-hand side will be zero. In matlab, even this step is not necessary since the matlab solve returns the action of the pseudo-inverse in cases where the system is singular. In that case, the answer returns 0 for the k th entry as a property of the pseudoinverse is that the solution is determined by minimizing the L^2 norm of the output.

- The coarse-grid strategy proposed above has several interesting features.
- First, the fine-scale detail of $z_c(\mathbf{x})$ will match the geometry and (homogenous) boundary conditions of the originating problem.
- Note that points $\mathbf{x}_k \in \partial\Omega_D$ are *not* included in the interpolation process, so nodal values on $\partial\Omega_D$ will not be set to anything other than zero.
- On the other hand, values on $\partial\Omega_N$ will be set to nontrivial values.
- The coarse-grid solution is a valid solution in the fine-scale solution search space (e.g., in $X_0^N(\Omega)$ as defined in the next section on the finite element method).
- The main role of the coarse-grid problem is to propagate long-range information quickly (i.e., in a single coarse-grid solve, rather than through many iterations on a fine mesh).
- Within the FEM context, the elements of A_c continue to represent integrals over subsets of Ω —they do not represent integrals over the coarse grid covering, aside from the fact that one could view those integrals as being partitioned into subsets in Ω and those outside Ω , where the integrand is identically zero.