Introduction to Additive Schwarz Methods and Krylov Solvers

Victorita Dolean

Contents

1 Introduction and Motivation: The Need for Domain Decomposition

In modern computational science and engineering, solving partial differential equations (PDEs) efficiently and accurately lies at the heart of many applications. These range from medical imaging to electronics, from seismic modeling to aeronautics. The numerical resolution of PDEs often leads to very large linear systems. This section introduces the fundamental motivations for developing parallel algorithms, particularly domain decomposition methods (DDM), by illustrating the computational challenges and opportunities.

1.1 Electromagnetic Wave Propagation in Heterogeneous Media

A motivating example comes from electromagnetics, where the inverse problem of reconstructing the electric permittivity ε from measurements of the electric field $\mathbf{E}$ involves solving Maxwell's equations:

$$\nabla \times (\mu^{-1} \nabla \times \mathbf{E}) - \omega^2 \varepsilon \mathbf{E} = \mathbf{J} \tag{1}$$

where:

- $\mathbf{E}$ is the electric field,
- $\mu > 0$ is the magnetic permeability,
- $\varepsilon > 0$ is the electric permittivity,
- ω is the angular frequency,
- and $\mathbf{J}$ is a current source.

Key Challenge

The inverse problem — reconstructing ε from $\mathbf{E}$ — is ill-posed and numerically challenging, especially in complex geometries and high-frequency regimes.

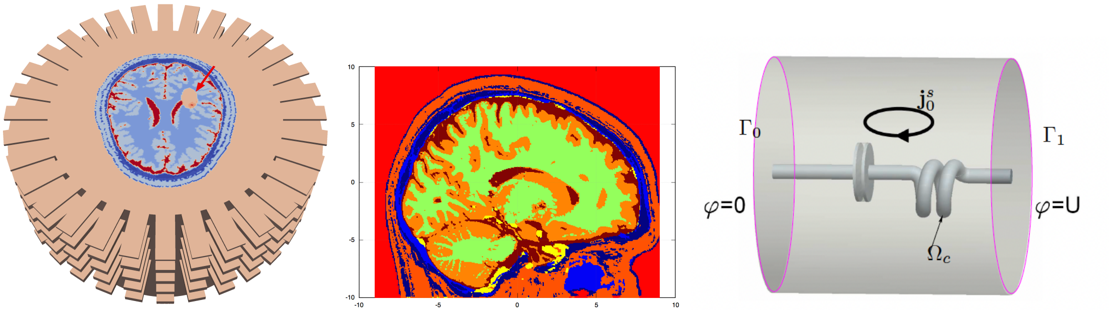


Figure 1: Examples of wave propagation in complex domains: brain imaging for stroke localization (left, center), and current injection in an electric component (right).

1.2 Why Efficient Linear Solvers Matter

Most wave propagation problems require solving large-scale linear systems of the form:

$$A\mathbf{u} = \mathbf{b}$$

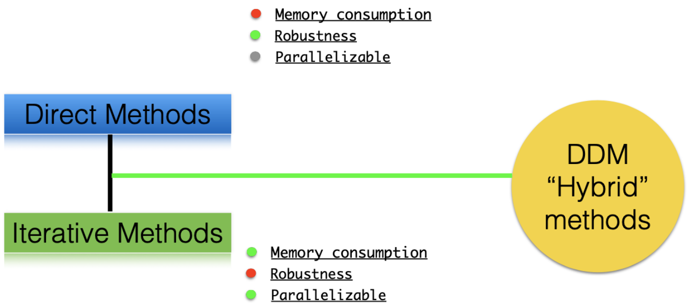


Figure 2: Solver landscape: balancing accuracy, robustness, and scalability.

We broadly distinguish:

- **Direct methods:** robust but memory-intensive and hard to parallelize efficiently.
- **Iterative methods:** scalable, but convergence may be poor for ill-conditioned systems.
- **Hybrid strategies:** such as domain decomposition or multigrid, aim to combine robustness and scalability.

Hybrid Solvers

Domain decomposition (DDM) and multigrid methods provide flexible frameworks for parallelism, ideally suited for large-scale simulations on modern hardware.

1.3 Why Not Just Use Direct Solvers?

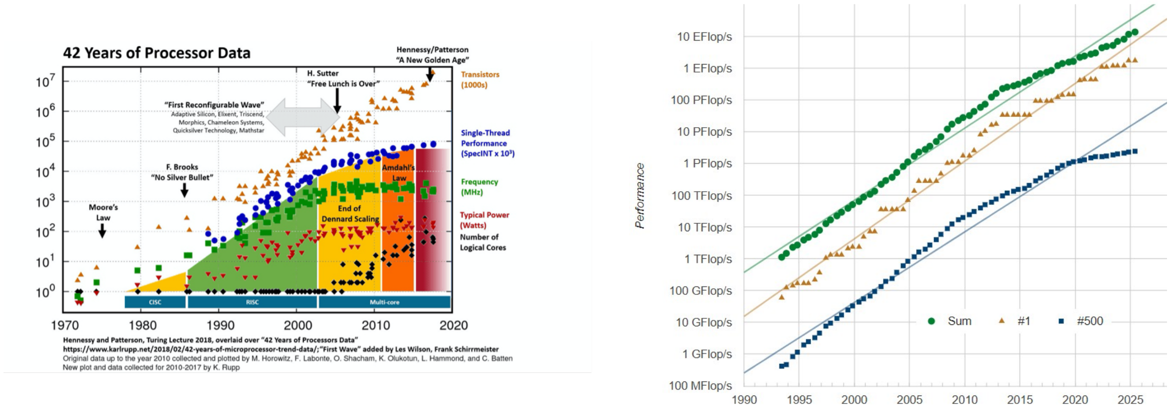
Let's consider the asymptotic complexity of sparse Gaussian elimination for structured PDE matrices:

Method	**1D** ($d = 1$)	**2D** ($d = 2$)	**3D** ($d = 3$)
Dense matrix	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$
Band structure exploited	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{7/3})$
Sparse (e.g. nested dissection)	$\mathcal{O}(n)$	$\mathcal{O}(n^{3/2})$	$\mathcal{O}(n^2)$

In practice:

- 2D problems can be tackled up to 10^7 unknowns.
- 3D problems are more demanding — fill-in leads to memory exhaustion around 10^5 unknowns.
- Direct solvers remain essential in small or medium-size subproblems (e.g., inside DDM).

1.4 Hardware Trends and Parallelism Opportunities



Hennessy and Patterson, Turing Lecture 2018, overlaid over "42 Years of Processors Data"
https://www.karlrupp.net/2018/02/42-years-of-microprocessor-trend-data/;"First Wave" added by Les Wilson, Frank Schirrmeister
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

Figure 3: Left: End of Dennard scaling — frequency gains stall. Right: Top500 supercomputers performance trends.

Why Parallelism is Essential

Modern processors can no longer deliver faster performance by clock speed alone. All hardware performance gains come from increased **parallelism**: more cores, vectorization, GPUs, and distributed systems.

- Workstations and laptops now host 8–64 cores.
- National-scale supercomputers (e.g., Fugaku, LUMI) feature $> 10^5$ cores.
- Heterogeneous computing (CPU+GPU) is the new norm.
- Domain decomposition algorithms map naturally onto such architectures.

Takeaway

Why Domain Decomposition?

- Flexible, modular framework for parallelization.
- Supports hybrid solvers combining local direct solves with global convergence acceleration.
- Well-suited for modern hardware trends.

2 Domain Decomposition Foundations

2.1 The Schwarz Method: Origin of Domain Decomposition

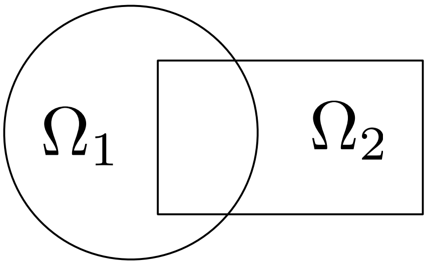
The Original Schwarz Method (1870)

A classical overlapping method solving $-\Delta u = f$ in Ω with homogeneous Dirichlet boundary conditions, by iteratively updating solutions over overlapping subdomains.

Schwarz Alternating Iteration

Given u_1^n, u_2^n, compute:

$$\begin{aligned} -\Delta u_1^{n+1} &= f \quad \text{in } \Omega_1, & u_1^{n+1} &= 0,\ \partial\Omega_1 \cap \partial\Omega \\ & & u_1^{n+1} &= u_2^n,\ \partial\Omega_1 \cap \overline{\Omega}_2 \\ -\Delta u_2^{n+1} &= f \quad \text{in } \Omega_2, & u_2^{n+1} &= 0,\ \partial\Omega_2 \cap \partial\Omega \\ & & u_2^{n+1} &= u_1^{n+1},\ \partial\Omega_2 \cap \overline{\Omega}_1 \end{aligned}$$



Two overlapping subdomains Ω_1 and Ω_2

Key Observations

- The method is naturally parallel, but has slow convergence.
- Overlap is essential for communication between subdomains.
- The parallel version is known as the Jacobi–Schwarz Method (JSM).

2.2 Abstract ASM and RAS: Local-to-Global Strategy

Definition 1 (Extension Operators)**.** *Each E_i extends a function $w_i : \Omega_i \to \mathbb{R}$ to a global function $E_i(w_i) : \Omega \to \mathbb{R}$ by zero outside Ω_i.*

Definition 2 (Partition of Unity)**.** *Let $\chi_i : \Omega_i \to \mathbb{R}$ such that:*

$$\chi_i(x) \geq 0, \quad \chi_i(x) = 0 \ \textit{on } \partial\Omega_i, \quad \sum_i \chi_i(x) = 1 \ \textit{on } \Omega$$

We can reconstruct global functions from local ones:

$$w(x) = \sum_i E_i(\chi_i \cdot w|_{\Omega_i})$$

Iteration Scheme Given a global approximation u^n, solve local problems on Ω_i and glue the updated local solutions via extension operators:

- Additive Schwarz (ASM): $u^{n+1} = \sum_i E_i(u_i^{n+1})$
- Restricted Additive Schwarz (RAS): $u^{n+1} = \sum_i E_i(\chi_i u_i^{n+1})$

2.3 Connection to Block-Jacobi Iteration

Jacobi for Linear Systems

Given $A\mathbf{U} = \mathbf{F}$, the Jacobi method updates:

$$\mathbf{U}^{n+1} = \mathbf{U}^n + D^{-1}(\mathbf{F} - A\mathbf{U}^n) = \mathbf{U}^n + D^{-1}\mathbf{r}^n$$

Block Partitioning Divide $\mathcal{N}$ into $\mathcal{N}_1$, $\mathcal{N}_2$ and write A in block form:

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \quad \mathbf{U} = \begin{pmatrix} \mathbf{U}_1 \\ \mathbf{U}_2 \end{pmatrix}$$

Block Jacobi Update

$$\begin{aligned} A_{11}\mathbf{U}_1^{n+1} &= \mathbf{F}_1 - A_{12}\mathbf{U}_2^n \\ A_{22}\mathbf{U}_2^{n+1} &= \mathbf{F}_2 - A_{21}\mathbf{U}_1^n \end{aligned}$$

Compact Residual Form Let R_i be Boolean restriction matrices, $A_i = R_i A R_i^T$. Then:

$$\boxed{\mathbf{U}^{n+1} = \mathbf{U}^n + \left(R_1^T A_1^{-1} R_1 + R_2^T A_2^{-1} R_2\right) \mathbf{r}^n}$$

This abstractly mirrors the RAS and ASM formulations in global PDE solvers.

Key Insight

Block Jacobi is the algebraic equivalent of domain decomposition: local solves, restricted updates, parallel structure.

1D Finite Difference Setup Consider the 1D Poisson problem with Dirichlet boundary conditions:

$$-\Delta u = f \quad \text{in } \Omega = (0,1), \quad u(0) = u(1) = 0$$

Discretized on a uniform mesh with m internal nodes using centered finite differences leads to:

$$A\mathbf{U} = \mathbf{F}, \quad A \in \mathbb{R}^{m\times m}, \quad A = \frac{1}{h^2}\text{tridiag}(-1, 2, -1)$$

where $h = \frac{1}{m+1}$. We split the domain into overlapping subdomains:

$$\Omega_1 = (0, (m_s + 1)h), \quad \Omega_2 = (m_s h, 1)$$

The overlap is one mesh cell (h).

Jacobi–Schwarz Local Update on Ω_1

$$\begin{cases} -\dfrac{u_{1,j-1}^{n+1} - 2u_{1,j}^{n+1} + u_{1,j+1}^{n+1}}{h^2} = f_j, & 1 \le j \le m_s \\ u_{1,0}^{n+1} = 0, & \text{(Dirichlet BC)} \\ u_{1,m_s+1}^{n+1} = u_{2,m_s+1}^{n} & \text{(interface update)} \end{cases}$$

An analogous update is performed in Ω_2 using the latest values from Ω_1.

Matrix Formulation and Block Decomposition Splitting the system across index sets $\mathcal{N}_1$, $\mathcal{N}_2$ yields:

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \quad \mathbf{U} = \begin{pmatrix} \mathbf{U}_1 \\ \mathbf{U}_2 \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \end{pmatrix}$$

Block-Jacobi Update (Matrix View)

$$\begin{pmatrix} A_{11} & 0 \\ 0 & A_{22} \end{pmatrix} \begin{pmatrix} \mathbf{U}_1^{n+1} \\ \mathbf{U}_2^{n+1} \end{pmatrix} = \begin{pmatrix} \mathbf{F}_1 - A_{12}\mathbf{U}_2^n \\ \mathbf{F}_2 - A_{21}\mathbf{U}_1^n \end{pmatrix}$$

Each subsystem can be solved independently — the essence of parallelism.

Minimal Overlap and Operator View Visually this gives

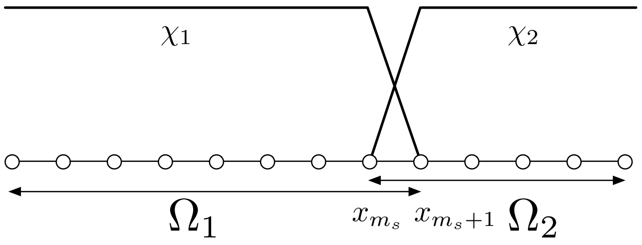


Figure 4: Simple 1D decomposition with minimal overlap (one shared point)

Extension and Gluing

Define the extension operators:

$$E_1(U_1) = \begin{pmatrix} U_1 \\ 0 \end{pmatrix}, \quad E_2(U_2) = \begin{pmatrix} 0 \\ U_2 \end{pmatrix}$$

Then for a partition of unity χ_1, χ_2, we have:

$$E_1(U_1) + E_2(U_2) = E_1(\chi_1 U_1) + E_2(\chi_2 U_2) = \mathbf{U}$$

Key Insight

When the overlap is minimal, the following methods coincide:

- Additive Schwarz (AS)
- Restricted Additive Schwarz (RAS)
- Jacobi–Schwarz (JS)

They all reduce to the same block-Jacobi scheme in algebraic form.

2.4 Discrete Setting and Matrix Formulation

Domain decomposition methods rely on local computations in subdomains that must be recombined to recover global accuracy and convergence. This section translates continuous domain decomposition concepts into discrete linear algebra tools essential for implementation and theory.

Continuous vs. Discrete: Key Correspondence

Continuous Level

- Domain: $\Omega = \bigcup_{i=1}^{N} \Omega_i$ (overlapping)
- Function: $u : \Omega \to \mathbb{R}$
- Restriction: $u_i = u|_{\Omega_i}$
- Extension: $E_i(u_i)$ zero outside Ω_i
- Partition of unity: χ_i s.t.

$$u = \sum_{i=1}^{N} E_i(\chi_i u_i)$$

Discrete Level

- Degrees of freedom: $\mathcal{N} = \bigcup_{i=1}^{N} \mathcal{N}_i$
- Global vector: $\mathbf{U} \in \mathbb{R}^{\#\mathcal{N}}$
- Restriction: $R_i \in \{0,1\}^{\#\mathcal{N}_i \times \#\mathcal{N}}$
- Extension: R_i^T
- Partition of unity: diag. D_i such that

$$\sum_{i=1}^{N} R_i^T D_i R_i = I$$

Restriction Operators and Examples We illustrate how domain decomposition operates on mesh-based discretizations, both for finite differences (FD) and finite elements (FE).

Mesh-Based Decomposition

- $\mathcal{T}_h$: global mesh on Ω
- $\mathcal{T}_{h,i}$: mesh of Ω_i
- V_h: global FE space
- $V_{h,i}$: local FE space
- $r_i(u_h) = u_h|_{\Omega_i} \in V_{h,i}$

Restriction is realized algebraically by Boolean matrices R_i.

Algebraic Ingredients in DDM

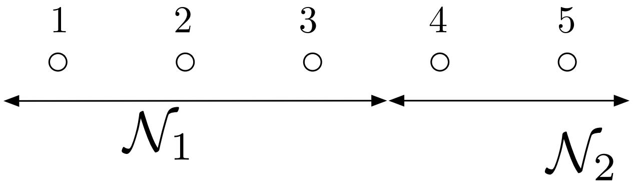
- Restriction: $R_i \in \mathbb{R}^{\#\mathcal{N}_i \times \#\mathcal{N}}$
- Prolongation: R_i^T
- Local matrix: $A_i = R_i A R_i^T$
- Partition matrix: D_i diagonal with

$$\sum_{i=1}^{N} R_i^T D_i R_i = I$$

These define how local residuals and updates map to the global context.

Finite Difference Example — No Overlap

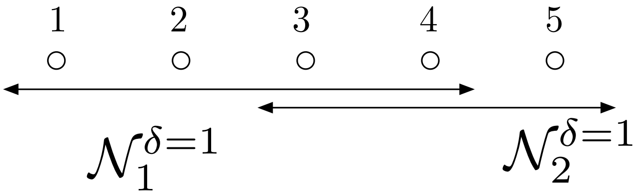
- $\mathcal{N} = \{1,2,3,4,5\}$, $\mathcal{N}_1 = \{1,2,3\}$, $\mathcal{N}_2 = \{4,5\}$
- R_1, R_2 select DOFs, $D_1, D_2 = I$



Finite Difference Example — Overlap

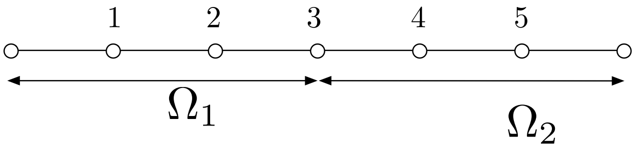
- $\mathcal{N}_1^{\delta=1} = \{1,2,3,4\}$, $\mathcal{N}_2^{\delta=1} = \{3,4,5\}$
- Overlap handled by partition weights:

$$D_1 = \operatorname{diag}(1,1,\tfrac{1}{2},\tfrac{1}{2}), \quad D_2 = \operatorname{diag}(\tfrac{1}{2},\tfrac{1}{2},1)$$



Finite Element Example — Overlap

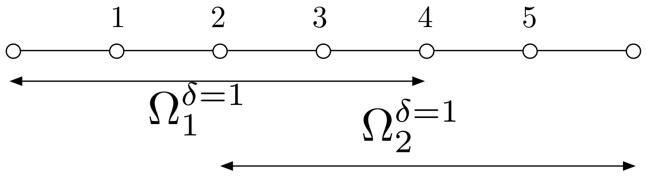
- $\mathcal{N}_1 = \{1,2,3\}$, $\mathcal{N}_2 = \{3,4,5\}$
- R_i selects basis functions, $D_1 = \operatorname{diag}(1,1,\tfrac{1}{2})$, $D_2 = \operatorname{diag}(\tfrac{1}{2},1,1)$



Finite Element Example — Extended Overlap

- $\mathcal{N}_1^{\delta=1} = \{1,2,3,4\}$, $\mathcal{N}_2^{\delta=1} = \{2,3,4,5\}$
- Balanced partitioning:

$$D_1 = \operatorname{diag}(1, \tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}), \quad D_2 = \operatorname{diag}(\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}, 1)$$



2.5 Algebraic Partitioning and Overlap

Graph-Based Partitioning

From a sparse matrix A, we define a connectivity graph G:

- **Nodes** correspond to degrees of freedom (DoFs).
- **Edges** exist between nodes i and j if $A_{ij} \neq 0$.
- The graph is symmetrized if A is not symmetric.
- Standard tools (`METIS`, `SCOTCH`) are used to partition G into N balanced subgraphs.

Partitioning Goals

- Load balancing: Each subdomain contains roughly the same number of unknowns.
- Minimized communication: The number of interface edges between subdomains is reduced.

Definition 3 (Overlapping Subdomains). *Given disjoint subsets $\{\mathcal{N}_i\}$ of nodes (DoFs), the overlap extension is defined by:*

$$\mathcal{N}_i^{\delta=1} = \mathcal{N}_i \cup \{j : A_{ij} \neq 0\}.$$

This collects all neighbors of $\mathcal{N}_i$ in the graph.

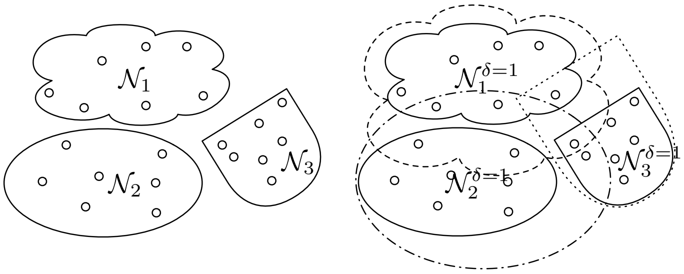

$\mathcal{N}_1$
$\mathcal{N}_2$
$\mathcal{N}_3$
$\mathcal{N}_1^{\delta=1}$
$\mathcal{N}_2^{\delta=1}$
$\mathcal{N}_3^{\delta=1}$

Figure 5: Extension to overlapping neighborhoods

Definition 4 (Algebraic Partition of Unity). *Let R_i be the restriction operator from $\mathcal{N}$ to $\mathcal{N}_i^{\delta=1}$. Define a diagonal matrix D_i of size $\#\mathcal{N}_i^{\delta=1}$ by:*

$$(D_i)_{jj} = \frac{1}{\#\mathcal{M}_j}, \quad \mathcal{M}_j = \{i \mid j \in \mathcal{N}_i^{\delta=1}\}.$$

Then,

$$\sum_{i=1}^{N} R_i^T D_i R_i = I$$

forms a partition of unity.

Finite Element Decomposition in Multiple Subdomains

Definition 5 (FE Subdomain Partitioning). *Given a finite element basis $\{\phi_k\}_{k\in\mathcal{N}}$:*

$$\mathcal{N}_i = \{k \mid \operatorname{supp}(\phi_k) \cap \Omega_i \neq \emptyset\}$$

$$\mu_k = \#\{j \mid \operatorname{supp}(\phi_k) \cap \Omega_j \neq \emptyset\}$$

Then,

$$(D_i)_{kk} = \frac{1}{\mu_k}, \quad \textit{for } k \in \mathcal{N}_i$$

provides an algebraic partition of unity.

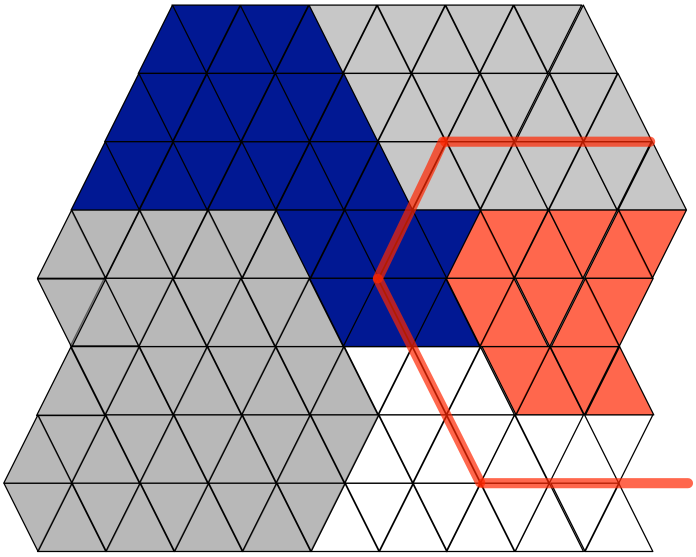


Figure 6: Overlapping finite element subdomains $\{\Omega_i\}$ covering Ω

Additive Schwarz Iteration

Let $A\mathbf{U} = \mathbf{F}$ and subdomains indexed by $\mathcal{N}_1, \mathcal{N}_2$. Then at each iteration:

$$\mathbf{U}_j^{m+1} = \mathbf{U}_j^m + A_j^{-1} R_j (\mathbf{F} - A\mathbf{U}^m), \quad j = 1, 2$$

where $\mathbf{U}_j^m = R_j \mathbf{U}^m$, $A_j = R_j A R_j^T$.

3 Convergence Analysis of Schwarz Methods

3.1 1D Analysis

Let $\Omega = (0, L)$ be decomposed into two overlapping subdomains:

- $\Omega_1 = (0, L_1)$,
- $\Omega_2 = (l_2, L)$ with $l_2 \leq L_1$.

The iteration error in subdomain Ω_i is defined as:

$$e_i^n := u_i^n - u|_{\Omega_i}$$

In Ω_1:

$$\begin{aligned} -\frac{d^2 e_1^{n+1}}{dx^2} &= 0, \\ e_1^{n+1}(0) &= 0, \\ e_1^{n+1}(L_1) &= e_2^n(L_1) \end{aligned}$$

In Ω_2:

$$\begin{aligned} -\frac{d^2 e_2^{n+1}}{dx^2} &= 0, \\ e_2^{n+1}(l_2) &= e_1^{n+1}(l_2), \\ e_2^{n+1}(L) &= 0 \end{aligned}$$

Since the Laplace equation yields affine solutions, we have:

$$\begin{aligned} e_1^{n+1}(x) &= e_2^n(L_1) \cdot \frac{x}{L_1}, \\ e_2^{n+1}(x) &= e_1^{n+1}(l_2) \cdot \frac{L - x}{L - l_2} \end{aligned}$$

Combining these relations:

$$e_2^{n+1}(L_1) = e_2^n(L_1) \cdot \frac{l_2}{L_1} \cdot \frac{L - L_1}{L - l_2}$$

Defining overlap $\delta = L_1 - l_2$, we get the iteration relation:

$$e_2^{n+1}(L_1) = \frac{1 - \delta/(L - l_2)}{1 + \delta/l_2} \cdot e_2^n(L_1)$$

Convergence Condition

Necessary and sufficient: $\delta > 0$ for convergence.
Without overlap ($\delta = 0$), the method extcolorredstagnates.

3.2 2D Fourier Mode Analysis

Decompose $\mathbb{R}^2$ into overlapping half-planes:

- $\Omega_1 = (-\infty, \delta) \times \mathbb{R}$,
- $\Omega_2 = (0, \infty) \times \mathbb{R}$.

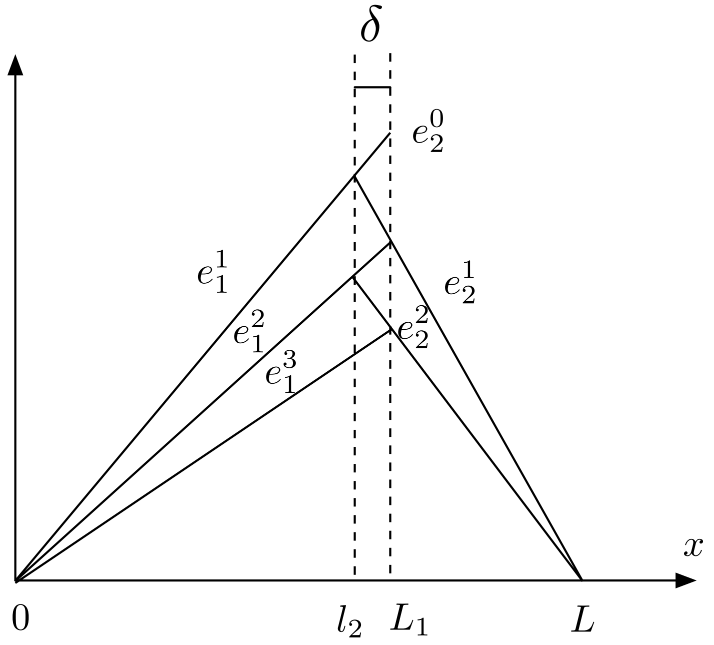


Figure 7: Decay of interface error as a function of overlap δ

Problem:

$$(\eta - \Delta)u = f, \quad u \text{ bounded at infinity}$$

Applying partial Fourier transform in y yields for each wavenumber k:

$$\left(\eta - \frac{\partial^2}{\partial x^2} + k^2\right) \hat{e}_j^{n+1}(x,k) = 0$$

General solution:

$$\hat{e}_j^{n+1}(x,k) = \gamma_+^{n+1}(k)e^{\lambda(k)x} + \gamma_-^{n+1}(k)e^{-\lambda(k)x}, \quad \lambda(k) = \sqrt{\eta + k^2}$$

Boundedness restricts the exponentials to:

$$\hat{e}_1^{n+1}(x,k) = \gamma_+^{n+1}(k)e^{\lambda(k)x},$$
$$\hat{e}_2^{n+1}(x,k) = \gamma_-^{n+1}(k)e^{-\lambda(k)x}$$

Matching across interfaces:

$$\gamma_+^{n+1}(k) = \gamma_-^{n}(k)e^{-\lambda(k)\delta},$$
$$\gamma_-^{n+1}(k) = \gamma_+^{n}(k)e^{-\lambda(k)\delta} \Rightarrow \gamma_\pm^{n+1}(k) = e^{-2\lambda(k)\delta} \cdot \gamma_\pm^{n-1}(k)$$

2D Convergence Summary

- **Decay per mode:** $\rho(k) = e^{-\lambda(k)\delta} < 1$
- **High frequencies:** larger $k \Rightarrow$ faster convergence
- **No overlap ($\delta = 0$):** $\rho = 1 \Rightarrow$ no decay

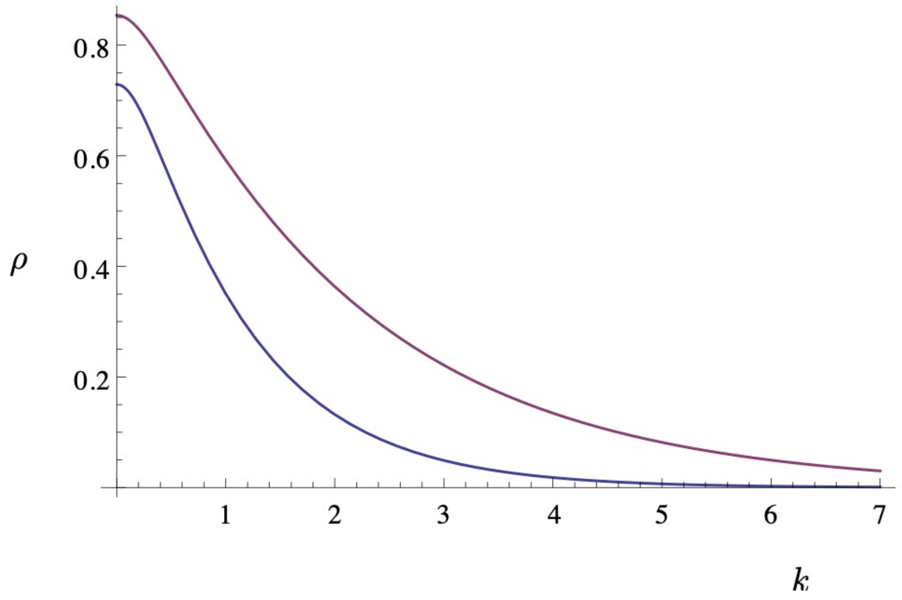


Figure 8: Convergence factor $\rho(k)$ as a function of wavenumber k and overlap δ

Conclusion: Convergence Insights

- **Overlap is essential** for convergence in both 1D and 2D.
- Interface error contracts by a factor depending on δ.
- High-frequency modes damp faster—relevant for preconditioning.

4 Preconditioners and Krylov methods

4.1 Global Iteration and Partitioned Updates

The RAS method can be interpreted as a global fixed-point iteration constructed from local solvers and algebraic glueing. It fits naturally into the Krylov framework.

RAS Global Iteration

Given residual $\mathbf{r}^n = \mathbf{F} - A\mathbf{U}^n$, the RAS update reads:

$$\mathbf{U}^{n+1} = \mathbf{U}^n + M_{\text{RAS}}^{-1}\mathbf{r}^n$$

Local to Global Decomposition

The global iterate is assembled from local solutions:

$$\mathbf{U}^n = \sum_{i=1}^{N} R_i^T D_i \mathbf{U}_i^n$$

where $\mathbf{U}_i^n$ is the local solution over the extended subdomain $\mathcal{N}_i^\delta$.

Remark 1. *Each $\mathbf{U}_i^n$ solves a restricted local system $A_i\mathbf{U}_i = R_i\mathbf{r}^n$, with $A_i = R_iAR_i^T$.*

The preconditioner M_{RAS}^{-1} is widely used in conjunction with non-symmetric Krylov solvers such as GMRES or BiCGStab. It improves robustness and convergence especially for ill-conditioned or high-frequency problems.

4.2 Variants: RAS, ASM and SORAS

Different Schwarz variants yield different preconditioners, balancing symmetry, parallelism and theoretical properties.

Preconditioner Definitions

- **RAS (Restricted Additive Schwarz):**

$$M_{\text{RAS}}^{-1} = \sum_{i=1}^{N} R_i^T D_i A_i^{-1} R_i$$

- **ASM (Additive Schwarz Method):**

$$M_{\text{ASM}}^{-1} = \sum_{i=1}^{N} R_i^T A_i^{-1} R_i$$

- **SORAS (Symmetrized Overlapping RAS):**

$$M_{\text{SORAS}}^{-1} = \sum_{i=1}^{N} R_i^T D_i B_i^{-1} D_i R_i$$

Remark 2. *B_i in SORAS may represent a symmetric approximation of A_i to ensure global symmetry.*

Here are some trade-offs: Theory vs. Performance

- **ASM** is symmetric and better suited for convergence analysis.
- **RAS** is faster in practice due to reduced overlap and communication.
- **SORAS** bridges symmetry and performance.

In this section, we review fixed-point and Krylov iterative strategies for solving linear systems, and how domain decomposition methods such as RAS and ASM act as preconditioners.

4.3 Fixed-Point Iterations and Krylov methods

Definition 6 (Fixed-Point Iteration). *Given a linear system $A\mathbf{x} = \mathbf{b}$, a fixed-point iterative method updates the solution by:*

$$\mathbf{x}^{n+1} = \mathbf{x}^n + B^{-1}(\mathbf{b} - A\mathbf{x}^n)$$

This is equivalent to computing a fixed point of the operator:

$$\mathbf{x} \mapsto \mathbf{x} + B^{-1}(\mathbf{b} - A\mathbf{x})$$

Letting $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}^0$ *and* $C = B^{-1}A$*, we have:*

$$\mathbf{x}^n = \sum_{i=0}^{n} (I - C)^i B^{-1}\mathbf{r}_0 + \mathbf{x}^0$$

Convergence condition: *Spectral radius* $\rho(I - C) < 1$*.*

Definition 7 (Krylov Subspace Method)**.** *We consider solving the preconditioned system:*

$$C\mathbf{x} = B^{-1}\mathbf{b}, \quad C = B^{-1}A$$

Define the initial residual $\mathbf{r}^0 = B^{-1}\mathbf{b} - C\mathbf{x}^0$*, and let* $\mathbf{y} := \mathbf{x} - \mathbf{x}^0$*. Then:*

$$C\mathbf{y} = \mathbf{r}^0$$

The iterates are constructed from the Krylov subspace:

$$\mathcal{K}^n(C, \mathbf{r}^0) := \mathit{span}\{\mathbf{r}^0, C\mathbf{r}^0, \ldots, C^{n-1}\mathbf{r}^0\}$$

Remark 3 (Polynomial Inversion Insight)**.** *There exists a polynomial* $\mathcal{P}$ *of degree less than* N *such that:*

$$C^{-1} = \mathcal{P}(C)$$

This justifies the use of polynomial Krylov approximations.

Conjugate Gradient (CG) Method: For symmetric positive definite (SPD) systems:

$$\mathbf{y}^n = \arg \min_{\mathbf{w} \in \mathcal{K}^n(A, \mathbf{r}^0)} \|A\mathbf{w} - \mathbf{r}^0\|_{A^{-1}}$$

$$\mathbf{x}^n = \mathbf{x}^0 + \mathbf{y}^n$$

Algorithm 1 Conjugate Gradient (CG) Iteration

```
$\rho_{i-1} = (\mathbf{r}_{i-1}, \mathbf{r}_{i-1})$
if $i = 1$ then
    $\mathbf{p}_1 = \mathbf{r}_0$
else
    $\beta_{i-1} = \rho_{i-1}/\rho_{i-2}$
    $\mathbf{p}_i = \mathbf{r}_{i-1} + \beta_{i-1}\mathbf{p}_{i-1}$
end if
$\mathbf{q}_i = A\mathbf{p}_i$
$\alpha_i = \rho_{i-1}/(\mathbf{p}_i, \mathbf{q}_i)$
$\mathbf{x}_i = \mathbf{x}_{i-1} + \alpha_i\mathbf{p}_i$
$\mathbf{r}_i = \mathbf{r}_{i-1} - \alpha_i\mathbf{q}_i$
```

GMRES with Preconditioning: For general non-symmetric systems:

$$\mathbf{y}^n = \arg \min_{\mathbf{w} \in \mathcal{K}^n(C, \mathbf{r}^0)} \|C\mathbf{w} - \mathbf{r}^0\|_2$$

4.4 Schwarz Preconditioners

Definition 8 (Algebraic Schwarz Preconditioners)**.** *The most used algebraic Schwarz preconditioners are*

- ***Restricted Additive Schwarz (RAS)****:*

$$B^{-1} = M_{RAS}^{-1} = \sum_{i=1}^{N} R_i^T D_i (R_i A R_i^T)^{-1} R_i$$

- ***Additive Schwarz Method (ASM)****:*

$$B^{-1} = M_{ASM}^{-1} = \sum_{i=1}^{N} R_i^T (R_i A R_i^T)^{-1} R_i$$

Proposition 1 (Preconditioned CG Convergence with ASM)**.** *If M_{ASM}^{-1} is used in CG, then:*

$$\|\bar{\mathbf{x}} - \mathbf{x}_m\|_{M_{ASM}^{-1}A} \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^m \|\bar{\mathbf{x}} - \mathbf{x}_0\|_{M_{ASM}^{-1}A},$$

where $\kappa = cond(M_{ASM}^{-1}A)$.

Algorithm 2 Preconditioned CG with ASM

```
$\rho_{i-1} = (\mathbf{r}_{i-1}, M_{\mathrm{ASM}}^{-1}\mathbf{r}_{i-1})$
**if** $i = 1$ **then**
    $\mathbf{p}_1 = M_{\mathrm{ASM}}^{-1}\mathbf{r}_0$
**else**
    $\beta_{i-1} = \rho_{i-1}/\rho_{i-2}$
    $\mathbf{p}_i = M_{\mathrm{ASM}}^{-1}\mathbf{r}_{i-1} + \beta_{i-1}\mathbf{p}_{i-1}$
**end if**
$\mathbf{q}_i = A\mathbf{p}_i$
$\alpha_i = \rho_{i-1}/(\mathbf{p}_i, \mathbf{q}_i)$
$\mathbf{x}_i = \mathbf{x}_{i-1} + \alpha_i\mathbf{p}_i$
$\mathbf{r}_i = \mathbf{r}_{i-1} - \alpha_i\mathbf{q}_i$
```

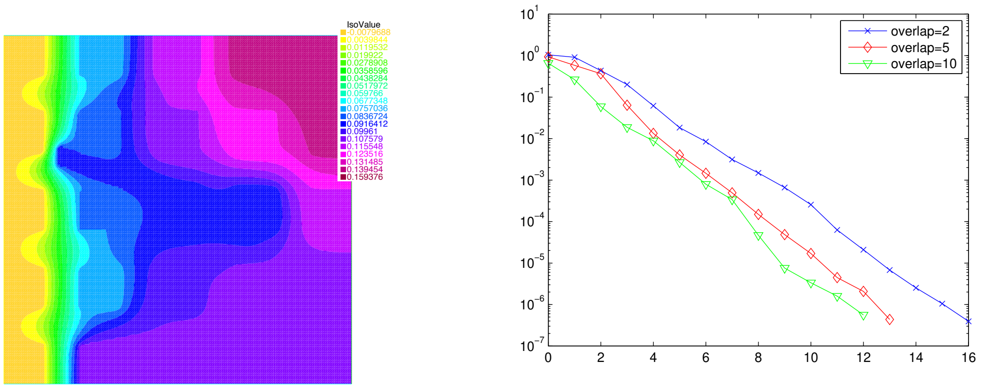


Figure 9: Example: Solution profile and CG convergence with ASM preconditioning.