Domain decomposition

Karl Meerbergen

KU Leuven

October 16, 2013

Literature

- James Demmel, Applied numerical Linear Algebra, SIAM, 1997.

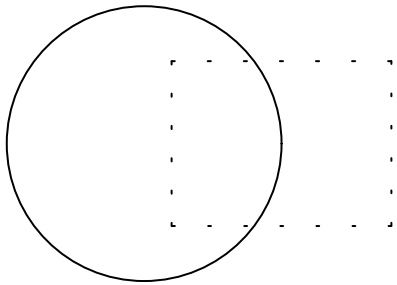
  Gives a good overview of preconditioning techniques.

- Anne Greenbaum, Iterative methods for solving linear systems, SIAM, 1997.

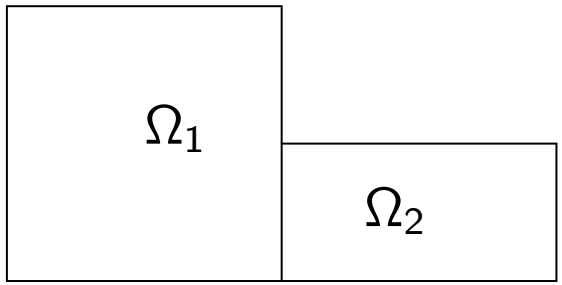
  Gives an excellent overview of Krylov methods. The book also briefly discusses preconditioning techniques.

Literature

- In the early days (19th century), analytical solutions were available for regular domains: disk, rectangle, ...
- Then, domain decomposition was used to decompose an irregular domain in a set of overlapping regular domains.
- Domain decomposition is therefore suitable for parallel computing.
- Two classes of methods:
  - Overlapping domains



  - Non-overlapping domains

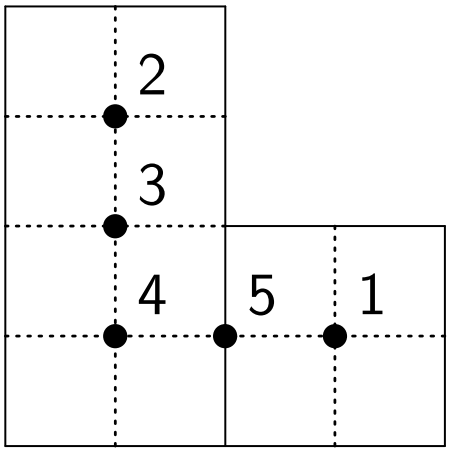


Non-overlapping methods

- Model problem:

$$
\begin{aligned}
-\frac{\partial^2 u(x,y)}{\partial x^2} - \frac{\partial^2 u(x,y)}{\partial y^2} &= f(x,y), \quad 0 < x, y < 1 \\
u(0,y) = u(1,y) = u(x,0) = u(x,1) &= 0, \quad 0 < x, y < 1 \, .
\end{aligned}
$$

- Mesh:



Domain $\Omega_1 = \{1\}$
Domain $\Omega_2 = \{2, 3, 4\}$
Interface $\Gamma = \{5\}$

Non-overlapping methods

- Linear system:

$$\left[\begin{array}{c|cccc} 4 & 0 & 0 & 0 & -1 \\ \hline 0 & 4 & -1 & & \\ 0 & -1 & 4 & -1 & \\ 0 & & -1 & 4 & -1 \\ \hline -1 & & & -1 & 4 \end{array}\right] \left(\begin{array}{c} x_1 \\ \hline x_2 \\ x_3 \\ x_4 \\ \hline x_5 \end{array}\right) = \left(\begin{array}{c} f_1 \\ \hline f_2 \\ f_3 \\ f_4 \\ \hline f_5 \end{array}\right)$$

- General structure for two domains:

$$\begin{bmatrix} A_{11} & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix}$$

Non-overlapping methods

- block LU factorization

$$\begin{bmatrix} A_{11} & & A_{13} \\ & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} = \begin{bmatrix} I & & \\ & I & \\ A_{31}A_{11}^{-1} & A_{32}A_{22}^{-1} & I \end{bmatrix} \cdot \begin{bmatrix} A_{11} & & A_{13} \\ & A_{22} & A_{23} \\ & & S \end{bmatrix}$$

  where

$$S = A_{33} - A_{31}A_{11}^{-1}A_{13} - A_{32}A_{22}^{-1}A_{23} \ .$$

  is the Schur complement.

- Solution of linear system:
  - Forward solve:
    1. $y_s = f_s,\ s = 1, 2,\ y_3 = f_3 - A_{31}A_{11}^{-1}y_1 - A_{32}A_{22}^{-1}y_2$
    2. Solve with A_{11} and A_{22}
  - Backward solve:
    1. $x_3 = S^{-1}y_3,\ x_s = A_{ss}^{-1}(y_s - A_{s3}x_3),\ s = 1, 2.$
    2. Solve with A_{11} and A_{22}
    3. Solve with S: this is the hard part.

Parallelism in domain decomposition

- Forward solve:

$$y_s = f_s, \quad s = 1, 2, \quad y_3 = f_3 - A_{31}A_{11}^{-1}y_1 - A_{32}A_{22}^{-1}y_2$$

Algorithm (Forward solve)

1: $y_s = f_s$, $s = 1, 2$ *in parallel*
2: *Compute* $y_{3s} = A_{3s}A_{ss}^{-1}y_s$ *in parallel*
3: *Compute* $y_3 = f_3 - y_{31} - y_{32}$

- Backward solve:

$$x_3 = S^{-1}y_3, \quad x_s = A_{ss}^{-1}(y_s - A_{s3}x_3)$$

Algorithm (Forward solve)

1: *First solve* $Sx_3 = y_3$ *(see next slide)*
2: *Solve* $A_{ss}x_s = y_s - A_{s3}x_3$, $s = 1, 2$ *in parallel*

Solving the Schur complement system

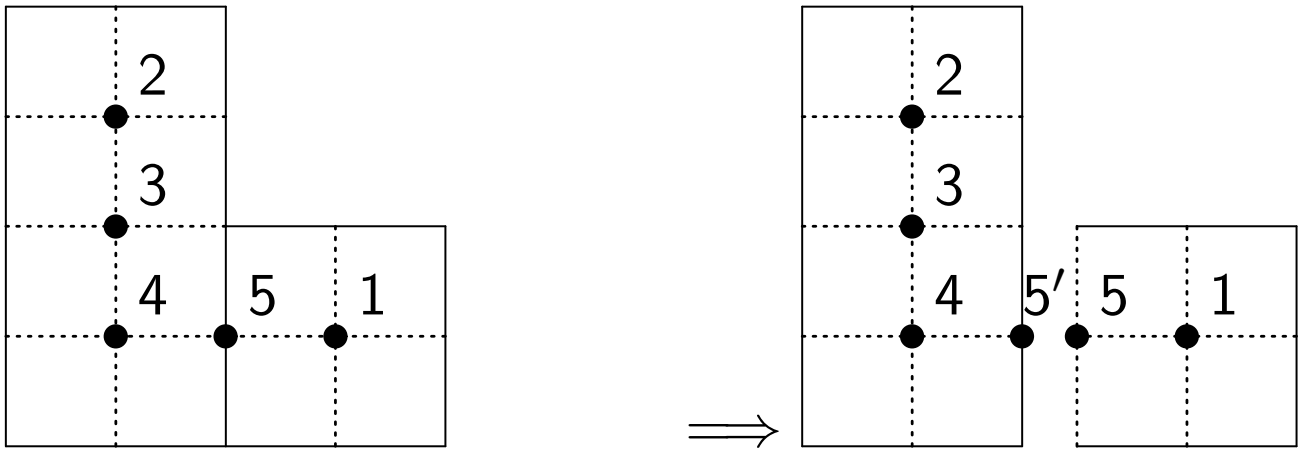
- S is usually a dense (=full) matrix.
- For our model problem:
  - S is symmetric positive definite.
  - $\text{cond}(A) = O(h^{-2})$.
  - $\text{cond}(S) = O(h^{-1})$.
  - We can use CG: no need to explicitly compute S.
  - Matrix vector product with $S = A_{33} - A_{31}A_{11}^{-1}A_{13} - A_{32}A_{22}^{-1}A_{23}$:

Algorithm (Compute $y = S \cdot x$)

1: *Compute independently:* $y_s = A_{3s}A_{ss}^{-1}A_{s3}x$
2: *Add contributions from each domain together:* $y = A_{33}x - y_1 - y_2$

Finite Element Tearing and Interconnecting (FETI)

- Tear the domains apart from each other by adding interface unknowns and then link them together by additional conditions.



- On domain s:

$$
\begin{aligned}
-\nabla^2 u_s &= f_s \quad \text{on} \quad \Omega_s \\
u_1 &= u_2 \quad \text{on} \quad \Gamma \\
-\frac{\partial u_1}{\partial n} &= \frac{\partial u_2}{\partial n} = \lambda \quad \text{on} \quad \Gamma
\end{aligned}
$$

λ is called the Lagrange multipliers.

Finite Element Tearing and Interconnecting (FETI)

- After discretization, the system for domain s becomes:

$$\begin{bmatrix} A_{ss} & A_{s3} \\ A_{3s} & A_{33}^{(s)} \end{bmatrix} \begin{pmatrix} x_s \\ x_3^{(s)} \end{pmatrix} = \begin{pmatrix} f_s \\ (-1)^s \lambda \end{pmatrix}$$

$$A_s \mathbf{x}_s = \mathbf{f}_s + B_s^T \lambda$$

  where $B_s = [0\ I]$.

- The interface system is

$$x_3^{(1)} - x_3^{(2)} = 0$$

  or

$$B_1 \mathbf{x}_1 - B_2 \mathbf{x}_2 = 0$$

  By eliminating $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$(B_1 A_1^{-1} B_1^T + B_2 A_2^{-1} B_2^T)\lambda = B_1 A_1^{-1} \mathbf{f}_1 - B_2 A_2^{-1} \mathbf{f}_2 \ .$$

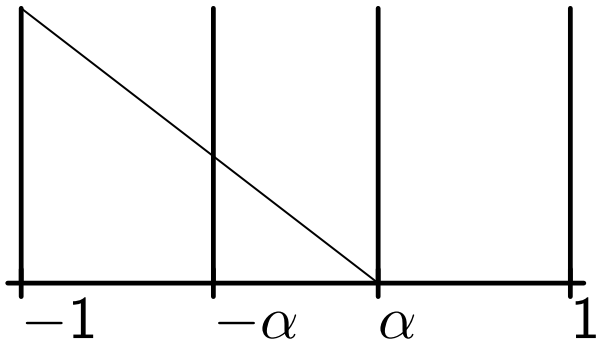
  This is a linear system in λ, that can be solved iteratively.

Overlapping domains: Schwartz methods

- Model problem:

$$\frac{d^2u}{dx^2} = 0 \quad , \quad u(-1) = u(1) = 1$$

- Exact solution $u \equiv 1$
- Subdivide in two overlapping domains: $(-1, \alpha)$ and $(-\alpha, 1)$ with $\alpha > 0$.
- Iterative process:



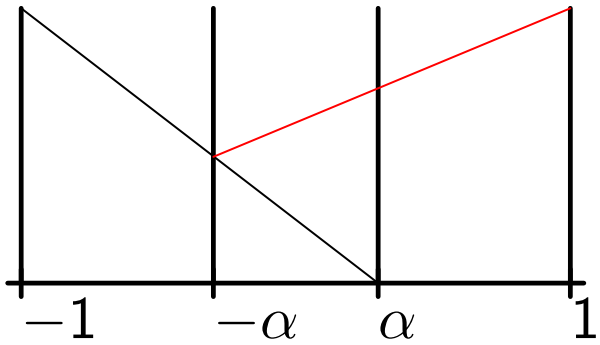
Initial solution Ω_1

Overlapping domains: Schwartz methods

- Model problem:

$$\frac{d^2u}{dx^2} = 0 \quad , \quad u(-1) = u(1) = 1$$

- Exact solution $u \equiv 1$
- Subdivide in two overlapping domains: $(-1, \alpha)$ and $(-\alpha, 1)$ with $\alpha > 0$.
- Iterative process:



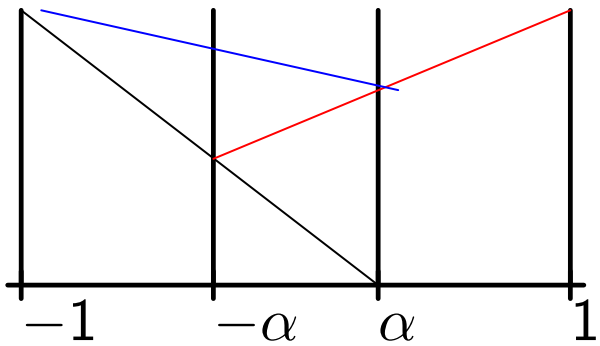
Compute u_2 with
BC $u_2(-\alpha) = u_1(-\alpha)$

Overlapping domains: Schwartz methods

- Model problem:

$$\frac{d^2u}{dx^2} = 0 \quad , \quad u(-1) = u(1) = 1$$

- Exact solution $u \equiv 1$
- Subdivide in two overlapping domains: $(-1, \alpha)$ and $(-\alpha, 1)$ with $\alpha > 0$.
- Iterative process:

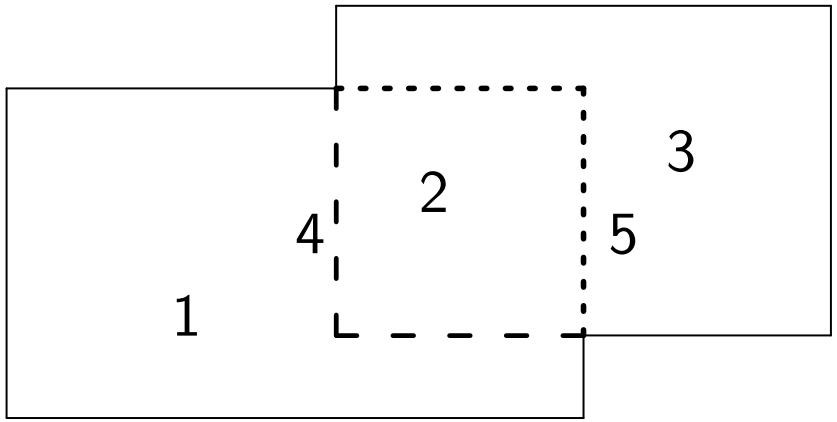


Compute u_1 with BC $u_1(\alpha) = u_2(\alpha)$

- It does not work without overlap.
- The larger the overlap, the faster is the convergence

Multiplicative Schwartz methods

- Overlapping domains: ordering of variables:



- Linear system:

$$\begin{bmatrix} A_{11} & & & A_{14} & \\ & A_{22} & & A_{24} & A_{25} \\ & & A_{33} & & A_{35} \\ A_{41} & A_{42} & & A_{44} & \\ & A_{52} & A_{53} & & A_{55} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix},$$

Multiplicative Schwartz methods

1: Given starting vector $x_1, \ldots, x_5$.
2: **while** no convergence **do**
3: Solve

$$\begin{bmatrix} A_{11} & & A_{14} \\ & A_{22} & A_{24} \\ A_{41} & A_{42} & A_{44} \end{bmatrix} \begin{pmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \tilde{x}_4 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 - A_{25}x_5 \\ f_4 \end{pmatrix}$$

4: Solve

$$\begin{bmatrix} A_{22} & & A_{25} \\ & A_{33} & A_{35} \\ A_{52} & A_{53} & A_{55} \end{bmatrix} \begin{pmatrix} \hat{x}_2 \\ \hat{x}_3 \\ \hat{x}_5 \end{pmatrix} = \begin{pmatrix} f_2 - A_{24}\tilde{x}_4 \\ f_3 \\ f_5 \end{pmatrix}$$

5: Set $x = \hat{x}$
6: **end while**

Multiplicative Schwartz methods

- Define E_s as a matrix that takes a vector from Ω_s and expands it to a vector of the entire domain.
- Then the algorithm can be written as:

$$\begin{aligned} \tilde{x} &= x + (E_1^T A E_1)^{-1} E_1^T (f - Ax) \\ \hat{x} &= \tilde{x} + (E_2^T A E_2)^{-1} E_2^T (f - A\tilde{x}) \end{aligned}$$

- With $B_s = (E_s^T A E_s)^{-1} E_s^T$, elimination of $\tilde{x}$ leads to

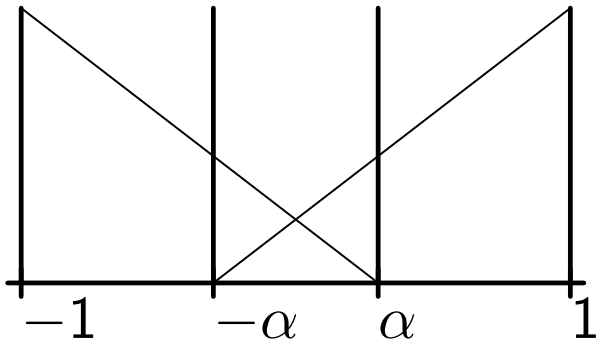
$$\hat{x} = x + \{B_1 + B_2 - B_2 A B_1\}(f - Ax)$$

This is a simple iteration method with preconditioner

$$B_1 + B_2 - B_2 A B_1 \ .$$

Additive Schwartz methods

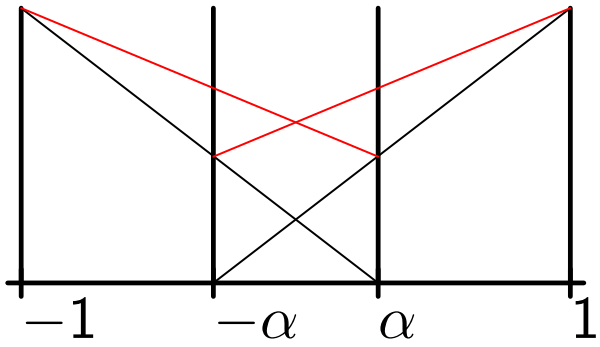
- Multiplicative Schwartz uses the last information from the 'other' subdomains. The method is a kind of block Gauss-Seidel method.
- Additive Schwartz only uses the information of the last iteration. As a result, the unknowns can be updated for each domain independently, as for Jacobi iteration.
- Iterative method:



Initial solution Ω_1 and Ω_2

Additive Schwartz methods

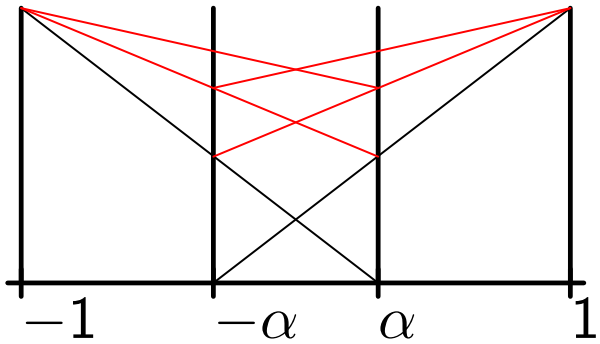
- Multiplicative Schwartz uses the last information from the 'other' subdomains. The method is a kind of block Gauss-Seidel method.
- Additive Schwartz only uses the information of the last iteration. As a result, the unknowns can be updated for each domain independently, as for Jacobi iteration.
- Iterative method:



Compute u_2 with
BC $u_2(-\alpha) = u_1(-\alpha)$

Additive Schwartz methods

- Multiplicative Schwartz uses the last information from the 'other' subdomains. The method is a kind of block Gauss-Seidel method.
- Additive Schwartz only uses the information of the last iteration. As a result, the unknowns can be updated for each domain independently, as for Jacobi iteration.
- Iterative method:



Compute u_1 with BC $u_1(\alpha) = u_2(\alpha)$

- It does not work without overlap.
- The larger the overlap, the faster is the convergence

Additive Schwarz methods

- Multiplicative Schwarz:

$$\begin{aligned}\tilde{x} &= x + (E_1^T A E_1)^{-1} E_1^T (f - Ax) \\ \hat{x} &= \tilde{x} + (E_2^T A E_2)^{-1} E_2^T (f - A\tilde{x})\end{aligned}$$

or

$$\hat{x} = x + \{B_1 + B_2 - B_2 A B_1\}(f - Ax)$$

- Additive Schwartz:

$$\begin{aligned}\tilde{x} &= x + (E_1^T A E_1)^{-1} E_1^T (f - Ax) \\ \hat{x} &= x + (E_2^T A E_2)^{-1} E_2^T (f - Ax)\end{aligned}$$

or

$$\hat{x} = x + \{B_1 + B_2\}(f - Ax)$$

- Multiplicative Schwarz converges faster than additive Schwarz but is harder to parallelize (similar to Gauss-Seidel and Jacobi)