Semi-Algebraic Coarse Space for Parallel Sparse Hybrid Solvers

Louis Poirel    Emmanuel Agullo    Luc Giraud
Goal

- Solve \( Ax = b \), where \( A \) is a large sparse matrix, on a distributed platform

How?

- Use Domain Decomposition (DD)

Focus of the talk

- DD is relevant for linear algebra applications
  - Can a high performance algebraic solver compete with problem-dependent solvers?
- Coarse Space for Additive Schwarz on the Schur and MaPHyS
  - Only in the SPD case
  - Need access to local matrices
Outline

1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Outline

1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Outline

1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Step 1: Analysis

Global Matrix $\mathcal{A}$

- $\mathcal{A}$ is a general sparse matrix. We want to solve $\mathcal{A}x = b$. 
Step 1: Analysis

Global Matrix \( \mathcal{A} \)  

Adjacency graph \( G \)

- The adjacency graph of \( \mathcal{A} \) \( (n \times n) \) is used as an algebraic mesh:  
  \[
  G = (\{1, \ldots, n\}, \{(i,j), \quad a_{ij} \neq 0 \mid a_{ji} \neq 0\})
  \]
- On the first row of \( \mathcal{A} \), \( a_{1,1}, a_{1,2} \) and \( a_{1,11} \neq 0 \)  
  \[
  \Rightarrow (1,1), (1,2) \text{ and } (1,11) \in G
  \]
Step 1: Analysis

Global Matrix $A$

Adjacency graph $G$

- A graph partitioner is used to split the graph
Step 1: Analysis

Global Matrix $\mathbf{A}$

Adjacency graph $\mathbf{G}$

\[
\begin{pmatrix}
A_{II} & A_{I\Gamma} \\
A_{\Gamma I} & A_{\Gamma\Gamma}
\end{pmatrix}
\begin{pmatrix}
x_I \\
x_\Gamma
\end{pmatrix}
=
\begin{pmatrix}
b_I \\
b_\Gamma
\end{pmatrix}
\]
Step 1: Analysis

Global Matrix $\mathcal{A}$

Adjacency graph $G$

- $\mathcal{A}_{II}$ has a block diagonal structure suitable for parallel computation
Step 1: Analysis

- How do we distribute $A_{\Gamma \Gamma}$?
Step 1: Analysis

Local Matrix $A_i$

Adjacency graph $G$

- We assign each interface node to a neighboring subdomain
Step 1: Analysis

Local Matrix $A_i$

Adjacency graph $G$

- We assign each interface node to a neighboring subdomain
Step 1: Analysis

Local Matrix $A_i$

Adjacency graph $G$

- We assign each interface node to a neighboring subdomain
Step 1: Analysis

Local Matrix $A_i$

Adjacency graph $G$

- We assign each interface node to a neighboring subdomain

$$A_i = \begin{pmatrix} A_{I_iI_i} & A_{I_i\Gamma_i} \\ A_{\Gamma_iI_i} & A_{\Gamma_i\Gamma_i} \end{pmatrix}$$

$$A = \sum_{i=1}^{N} R_i^T A_i R_i$$
Step 2: Factorization

Local Matrix $\mathcal{A}_i$

Adjacency graph $G$

- We factorize $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}$ and compute $S_i = \mathcal{A}_{\mathcal{I}_i \Gamma_i} - \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1} \mathcal{A}_{\mathcal{I}_i \Gamma_i}$

\[
\mathcal{A}_i = \begin{pmatrix}
\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i \Gamma_i} \\
\mathcal{A}_{\Gamma_i \mathcal{I}_i} & \mathcal{A}_{\Gamma_i \Gamma_i}
\end{pmatrix}
\]
Step 2: Factorization

Local Matrix $A_i$

Adjacency graph $G$

- We factorize $A_{I_iI_i}$ and compute $S_i = A_{\Gamma_i\Gamma_i} - A_{\Gamma_iI_i}A_{I_iI_i}^{-1}A_{I_i\Gamma_i}$

$$A_i = \begin{pmatrix}
A_{I_iI_i} & A_{I_i\Gamma_i} \\
A_{\Gamma_iI_i} & A_{\Gamma_i\Gamma_i}
\end{pmatrix}$$
Step 2: Factorization

Local Schur $S_i$

- We factorize $A_{I_i I_i}$ and compute $S_i = A_{\Gamma_i \Gamma_i} - A_{\Gamma_i I_i} A_{I_i I_i}^{-1} A_{I_i \Gamma_i}$

- Now, on each subdomain, the whole local problem is condensed onto the interface (dense matrix)
Step 2: Factorization

Local Schur $S_i$

Adjacency graph $G$

We solve the interface problem $S \chi_{\Gamma} = f = b_{\Gamma} - A_{\Gamma I} A_{II}^{-1} b_{II}$ with a preconditioned Krylov method.
AS Preconditioner

Local Matrix $A_i$

Adjacency graph $G$

- No overlap in $A_i$: $A = \sum_{i=1}^{N} R_i^T A_i R_i$
AS Preconditioner

Assembled Loc. Mat. $\bar{A}_i$

Adjacency graph $G$

- No overlap in $A_i$:
  \[ A = \sum_{i=1}^{N} R_i^T A_i R_i \]
- Assemble $\bar{A}_i = R_i A R_i^T$ using neighbor-to-neighbor communications

- $M_{AS} = \sum_{i=1}^{N} R_i^T \bar{A}_i^{-1} R_i$
AS Preconditioner

Assembled Loc. Mat. $\bar{A}_i$

Adjacency graph $G$

- No overlap in $A_i$: $A = \sum_{i=1}^{N} R_i^T A_i R_i$
- Assemble $\bar{A}_i = R_i \bar{A}^T R_i^T$ using neighbor-to-neighbor communications

$\mathcal{M}_{AS/A} = \sum_{i=1}^{N} R_i^T \bar{A}_i^{-1} R_i$

Not what we do
Step 3: Preconditioner Setup ($AS/S$)

Local Schur $S_i$

Adjacency graph $G$

No overlap in $S_i = A_{\Gamma_i \Gamma_i} - A_{\Gamma_i I_i} A_{I_i I_i}^{-1} A_{I_i \Gamma_i}$:

$$S = \sum_{i=1}^{N} R_{\Gamma_i} S_i R_{\Gamma_i}$$
Step 3: Preconditioner Setup (AS/S)

Assembled Local Schur $\tilde{S}_i$

Adjacency graph $G$

- No overlap in $S_i = A_{\Gamma_i\Gamma_i} - A_{\Gamma_i\mathcal{I}_i}A_{\mathcal{I}_i\mathcal{I}_i}^{-1}A_{\mathcal{I}_i\Gamma_i}$: $S = \sum_{i=1}^{N} R_{\Gamma_i}^T S_i R_{\Gamma_i}$
- Assemble $\tilde{S}_i = R_{\Gamma_i} S R_{\Gamma_i}$
- $\mathcal{M}_{AS/S} = \sum_{i=1}^{N} R_{\Gamma_i}^T \tilde{S}_i^{-1} R_{\Gamma_i}$
Step 3: Preconditioner Setup (AS/S)

Assembled Local Schur $\tilde{S}_i$

Adjacency graph $G$

- Share not only the $A_{\Gamma_i, \Gamma_i}$ part, but also $A_{\Gamma_i, \Gamma_i} A_{I_i, I_i}^{-1} A_{I_i, \Gamma_i}$
- The neighbor’s interiors are condensed on the subdomain’s interface too.
Step 4: Solve

Local Schur $S_i$

Adjacency graph $G$

- on $\Gamma$: Krylov method
- $S \times_{\Gamma} f$ preconditioned with $M_{AS/S}$
Step 4: Solve

Local Matrix $\mathcal{A}_i$

Adjacency graph $G$

- on $\Gamma$: Krylov method
  - $S x_\Gamma = f$  preconditioned with $\mathcal{M}_{AS/S}$
- on $\mathcal{I}$: Direct method
  - $x_{\mathcal{I}_i} = \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1} (b_{\mathcal{I}_i} - \mathcal{A}_{\mathcal{I}_i\Gamma_i} x_{\Gamma_i})$
Step by step

Step 1: Analysis
- Graph partitioning and data distribution

Step 2: Factorization
- Computation of \( A_{\mathcal{I}_i \mathcal{I}_i}^{-1} \) and \( S_i = A_{\mathcal{I}_i \mathcal{I}_i} \) - \( A_{\mathcal{I}_i \mathcal{I}_i} A_{\mathcal{I}_i \mathcal{I}_i}^{-1} A_{\mathcal{I}_i \mathcal{I}_i} \)

Step 3: Preconditioner Setup
- Assembly and factorization of \( \bar{S}_i \)

Step 4: Solve
- on \( \Gamma \): Krylov method
  - \( S x_\Gamma = f \) pre conditioned with \( M_{AS/S} = \sum_{i=1}^{N} R_{\Gamma_i}^T \bar{S}_i^{-1} R_{\Gamma_i} \)
- on \( \mathcal{I} \): Direct method
  - \( x_{\mathcal{I}_i} = A_{\mathcal{I}_i \mathcal{I}_i}^{-1} (b_{\mathcal{I}_i} - A_{\mathcal{I}_i \Gamma_i} x_{\Gamma_i}) \)
1. **Additive Schwarz on the Schur (AS/S)**
   - AS/S step by step
   - Comparison with other DD preconditioners

2. **MaPHyS solver**
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. **Two-level preconditioner for AS/S**
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Related DD preconditioners

- **Neumann-Neumann (NN)**

  \[ \mathcal{M}_{NN} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T D_i S_i^\dagger D_i \mathcal{R}_{\Gamma_i} \]

  where \( D_i \) is a partition of unity and

  \[ S_i = A_{\Gamma_i\Gamma_i} - A_{\Gamma_i\mathcal{I}_i} A_{\mathcal{I}_i\mathcal{I}_i}^{-1} A_{\mathcal{I}_i\Gamma_i} \]

- **Schur of Additive Schwarz (S–AS)**

  \[ \mathcal{M}_{S–AS} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \hat{S}_i^{-1} \mathcal{R}_{\Gamma_i} \]

  where \( \hat{\mathcal{A}}_{\Gamma_i\Gamma_i} = \sum_{j=1}^{N} \mathcal{R}_{\Gamma_i} \mathcal{R}_{\Gamma_j}^T A_{\Gamma_j\Gamma_j} \mathcal{R}_{\Gamma_j} \mathcal{R}_{\Gamma_i}^T \)

  and \( \hat{S}_i = \hat{\mathcal{A}}_{\Gamma_i\Gamma_i} - A_{\Gamma_i\mathcal{I}_i} A_{\mathcal{I}_i\mathcal{I}_i}^{-1} A_{\mathcal{I}_i\Gamma_i} \) is the Schur of \( \hat{\mathcal{A}}_i \)

- **Additive Schwarz on the Schur (AS/S)**

  \[ \mathcal{M}_{AS/S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \tilde{S}_i^{-1} \mathcal{R}_{\Gamma_i} \]

  where \( \tilde{\mathcal{S}}_i = \mathcal{R}_{\Gamma_i} S \mathcal{R}_{\Gamma_i}^T \)

  \[ \tilde{S}_i = \sum_{j=1}^{N} \mathcal{R}_{\Gamma_i} \mathcal{R}_{\Gamma_j}^T \left( A_{\Gamma_j\Gamma_j} - A_{\Gamma_j\mathcal{I}_i} A_{\mathcal{I}_i\mathcal{I}_i}^{-1} A_{\mathcal{I}_i\Gamma_i} \right) \mathcal{R}_{\Gamma_j} \mathcal{R}_{\Gamma_i}^T \]
3D Test problem

Heterogeneous diffusion

- $\nabla (K \nabla u) = 1$
- Alternating conductivity layers of 3 elements
  \[(\text{ratio } K = K_{\text{max}}/K_{\text{min}} \text{ between layers})\]

Domain decomposition

- Constant subdomain size: $10 \times 10 \times 10$ elements
- $N$ subdomains
  - $N \times 1 \times 1$ (1D decomposition)
  - $N/2 \times 2 \times 1$ (2D decomposition)

Boundary conditions

- Dirichlet on the left
- Neumann elsewhere
Semi-Algebraic Coarse Space for Parallel Sparse Hybrid Solvers

1. ID decomposition

Number of iterations to achieve convergence

Number of subdomains

Heterogeneity K
- I
Numerical method
- NN
- S-AS
- AS/S
The graph shows the number of iterations to achieve convergence as a function of the number of subdomains for different heterogeneity K values and numerical methods. The x-axis represents the number of subdomains, while the y-axis represents the number of iterations. The graph compares different numerical methods (NN, S-AS, AS/S) and heterogeneity levels (1, 100, 10000). The convergence behavior varies with different parameters, indicating the impact of heterogeneity and method choice on parallel sparse hybrid solvers.
Convergence analysis of parallel sparse hybrid solvers for 1D and 2D decompositions with varying heterogeneity K.

- **1D decomposition**
  - Number of iterations to achieve convergence
  - Number of subdomains
  - Heterogeneity K
    - 1
    - 100
    - 10000
  - Numerical methods
    - Green: NN
    - Blue: S-AS
    - Red: AS/S

- **2D decomposition**
  - Number of iterations to achieve convergence
  - Number of subdomains
  - Heterogeneity K
    - 1
    - 100
    - 10000
  - Numerical methods
    - Green: NN
    - Blue: S-AS
    - Red: AS/S
Outline

1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Outline

1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Step by step

Step 1: Analysis
- Graph partitioning and data distribution

Step 2: Factorization
- Computation of $A^{-1}_{\mathcal{I}_i\mathcal{I}_i}$ and $S_i = A_{\Gamma_i\Gamma_i} - A_{\Gamma_i\mathcal{I}_i}A^{-1}_{\mathcal{I}_i\mathcal{I}_i}A_{\mathcal{I}_i\Gamma_i}$

Step 3: Preconditioner Setup
- Assembly and factorization of $\tilde{S}_i$

Step 4: Solve
- on $\Gamma$: Krylov method
  - $S x_\Gamma = f$ preconditioned with $M_{AS/S} = \sum_{i=1}^{N} R_{\Gamma_i}^T \tilde{S}_i^{-1} R_{\Gamma_i}$
- on $\mathcal{I}$: Direct method
  - $x_{\mathcal{I}_i} = A^{-1}_{\mathcal{I}_i\mathcal{I}_i} (b_{\mathcal{I}_i} - A_{\mathcal{I}_i\Gamma_i} x_{\Gamma_i})$
Software Framework

Graph Partitioner

- Scotch [F. Pellegrini et al.]
- Metis [G. Karypis and V. Kumar]

Sparse Direct Solver

- MUMPS [P.R. Amestoy et al.]
- PaStiX [P. Ramet et al.]

Dense Direct Solver

- MKL library (Intel)

Iterative Solver

- CG/GMRES/FGMRES [V.Fraysse and L.Giraud]
Installing MaPHyS

- MaPHyS and its dependencies can be installed through `spack` in \(\leq 15\) minutes + coffee break
  
  `morse.gforge.inria.fr/spack/spack.html`

- From a laptop to an heterogeneous supercomputer
  
  `morse.gforge.inria.fr/maphys/install-maphys-cluster.html`

Using MaPHyS

- Documented test cases
- Centralized/Distributed input
  
  `maphys.gforge.inria.fr/maphystp.html`

- CeCILL-C license
Outline

1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Interfaces for MaPHyS

Application ↓
Analysis ↓
Factorization ↓ ↓ ↓ ↓
Preconditioner Setup ↓ ↓ ↓ ↓ ↓
Solve ↓ ↓ ↓ ↓ ↓

Centralized Matrix Interface

- Application provides global matrix $A$ on one process
- MaPHyS performs algebraic domain decomposition and data distribution
Interfaces for MaPHyS

Distributed matrix interface

- Application provides global matrix $A$ in a distributed way
- MaPHyS performs parallel algebraic domain decomposition and data redistribution
Interfaces for MaPHyS

Application  ↓  ↓  ↓  ↓  ↓
Factorization  ↓  ↓  ↓  ↓  ↓
Preconditioner Setup  ↓  ↓  ↓  ↓  ↓
Solve  ↓  ↓  ↓  ↓  ↓

Distributed subdomain interface

- Application performs domain decomposition and provides subdomain connectivity and local matrices $A_i$ in a distributed way.
- Analysis is bypassed.

Naturally compliant with FEM, but also FV, DG, HDG...
Interfaces for MaPHyS

Application ↓ ↓ ↓ ↓ ↓ ↓
Factorization ↓ ↓ ↓ ↓ ↓ ↓
Preconditioner Setup ↓ ↓ ↓ ↓ ↓ ↓
Solve ↓ ↓ ↓ ↓ ↓ ↓

Distributed subdomain interface

- Application performs domain decomposition and provides subdomain connectivity and local matrices $A_i$ in a distributed way
- Analysis is bypassed

- A request from users
- Naturally compliant with FEM, but also FV, DG, HDG...
  - provides more relevant local information: $A_i$ is the true matrix of the local problem!
Outline

1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Two-level Parallelism [S. Nakov]

Nachos4M

\[ N \quad 4.1M \]

\[ \text{Nnz} \quad 256.4M \]
One subdomain per core leads to a huge number of subdomains on modern architectures
- Lack of robustness

1 thread per process (32 domains in total)
Two-level Parallelism [S. Nakov]

2 threads per process (16 domains in total)

- Multithreaded subdomains $\rightarrow$ fewer and bigger subdomains
  - Bigger local problem to solve 😞
  - Smaller and better-conditioned interface problem 😊
Two-level Parallelism [S. Nakov]

- 4 threads per process (8 domains in total)
- Multithreaded subdomains $\rightarrow$ fewer and bigger subdomains
  - Bigger local problem to solve 😐
  - Smaller and better-conditioned interface problem 😊
Two-level Parallelism [S. Nakov]

8 threads per process (4 domains in total)
- Multithreaded subdomains → fewer and bigger subdomains
  - Bigger local problem to solve
  - Smaller and better-conditioned interface problem
Results (2-level parallelism)

Hopper Platform (NERSC)

- Two twelve-core AMD 'MagnyCours' 2.1-GHz
- Memory: 32 GB GDDR3
- Double precision

Matrix

<table>
<thead>
<tr>
<th></th>
<th>Nachos4M</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>4.1M</td>
</tr>
<tr>
<td>Nnz</td>
<td>256.4M</td>
</tr>
</tbody>
</table>
Results (2-level parallelism)
Results (2-level parallelism)
Results (2-level parallelism)

![Graphs showing memory peak, time, and normalized elapsed time for different numbers of cores and nodes.]

- **Memory peak (MB):**
  - 3 t/p: 768, 1536, 3072, 6144, 12288, 24576
  - 6 t/p: 768, 1536, 3072, 6144, 12288, 24576
  - 12 t/p: 768, 1536, 3072, 6144, 12288, 24576
  - 24 t/p: 768, 1536, 3072, 6144, 12288, 24576

- **Time (s):**
  - 3 t/p: 100, 1000, 10000, 100000
  - 6 t/p: 100, 1000, 10000, 100000
  - 12 t/p: 100, 1000, 10000, 100000
  - 24 t/p: 100, 1000, 10000, 100000

- **Normalized elapsed time:**
  - Factorization step
  - Setup of the preconditioner
  - Solve step
1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Goal

- Stabilize the iterative solve time
- Improve the method’s scalability

How?

- Add some coarse correction in our preconditioner
  - No change to the API

My contribution

- Convergence proof
  - Only in the SPD case
  - Need $A_i$ to be Symmetric Positive Semi-Definite (SPSD)
    (e.g. through Distributed Subdomain Interface)
- Experimental results
  - Python/MPI prototype
Outline

1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
2D Test problem

Heterogeneous diffusion

- \( \nabla (K \nabla u) = q \)
- 7 alternating conductivity layers
- Subdomain: 20 \( \times \) 20 elements

Boundary conditions

- Dirichlet on the left
- Neumann elsewhere
- Source: \( q = 1 \)

Conductivity \( K \) (\( N = 8 \) subdomains)
2D Test problem

Heterogeneous diffusion

- $\nabla (K \nabla u) = q$
- 7 alternating conductivity layers
- Subdomain: $20 \times 20$ elements

Boundary conditions

- Dirichlet on the left
- Neumann elsewhere
- Source: $q = 1$

Conductivity $K$ ($N = 8$ subdomains)

Solution $x^*$ ($N = 8$ subdomains)
Convergence Behavior

$x_\Gamma, N = 128, n_{iter} = 0$
Convergence Behavior

$x_\Gamma, \ N = 128, \ n_{iter} = 10$
Convergence Behavior

\[ x_\Gamma, \ N = 128, \ n_{iter} = 20 \]
Convergence Behavior

$x_\Gamma, N = 128, n_{iter} = 30$
Convergence Behavior

\[ x_\Gamma, \ N = 128, \ n_{\text{iter}} = 40 \]
Convergence Behavior

$x_\Gamma, N = 128, n_{iter} = 50$
Convergence Behavior

$x_\Gamma, N = 128, \ n_{\text{iter}} = 60$
Convergence Behavior

$x_\Gamma$, $N = 128$, $n_{iter} = 70$
Convergence Behavior

\[ x_{\Gamma}, \ N = 128, \ \text{iter} = 70 \]

Problem

- No global exchange of information
- Algebraic bound on \( \lambda_{\text{max}}(M_{\text{AS/S}}) \), but problem with \( \lambda_{\text{min}} \)
Convergence Behavior

$x_{\Gamma}, N = 128, n_{\text{iter}} = 70$

Problem

- No global exchange of information
- Algebraic bound on $\lambda_{\text{max}}(M_{AS/SS})$, but problem with $\lambda_{\text{min}}$

Solution

- Use an exact direct solve on a coarse space $V_0$
Coarse Correction for AS

Coarse space $V_0$

- Should contain the problematic modes
- Often problem-dependent

Notations

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_0$</td>
<td>Basis of the coarse space</td>
</tr>
<tr>
<td>$R_0 = V_0^T$</td>
<td>Restriction to the coarse space</td>
</tr>
<tr>
<td>$\bar{S}_0 = R_0 S R_0^T$</td>
<td>Coarse matrix</td>
</tr>
<tr>
<td>$M_0 = R_0^T \bar{S}_0^{-1} R_0$</td>
<td>Coarse solve</td>
</tr>
<tr>
<td>$P_0 = M_0 S$</td>
<td>$S$-orthogonal projection on $V_0$</td>
</tr>
</tbody>
</table>
Coarse Correction for AS

2-level Additive Preconditioner

\[ M_{AS,2} = M_0 + M_{AS} \]

Deflated Preconditioner

\[ M_{AS,D} = M_0 + (I - P_0) M_{AS} (I - P_0)^T \]
Coarse Correction for AS

2-level Additive Preconditioner

\[ M_{AS,2} = M_0 + M_{AS} \]

Deflated Preconditioner

\[ M_{AS,D} = M_0 + (I - P_0) M_{AS} (I - P_0)^T \]
Outline

1 Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2 MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3 Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
Robust Solvers

- Bound on the condition number independent of the "difficulty" of the problem and the number of subdomains
- Coarse space for Additive Schwarz (AS), Neumann (NN) and Finite Element Tearing and Interconnecting (FETI)

Context

- A Symmetric Positive Definite (SPD)
- Element matrices $a_τ$

Method

- Solve a generalized eigenproblem in each subdomain
  - keep eigenvalues below a threshold $η$ in the coarse space
- Use a two-level preconditioner
Local Eigenproblem and Global Coarse Space

- Let \((p_j^k)^{m_j}_{k=1}\) be the eigenvectors of
  \[ a_{\Omega_j}(p, \nu) = \lambda \ a_{\Omega_j^\circ}(\Xi_j(p), \Xi_j(\nu)) \quad \forall \nu \in V_h(\Omega_j) \]
corresponding to the \(m_j\) smallest eigenvalues.

- \(V_0 = \text{span}\{R_j^T \Xi_j(p_j^k) : k = 1, \ldots, m_j; j = 1, \ldots, N\}\)

Convergence Theorems

\[
\kappa(M_{2A}) \leq (1 + k_0) \left[2 + k_0(2k_0 + 1) \max_{1 \leq j \leq N} \left(1 + \frac{1}{\lambda_{m_j+1}}\right)\right]
\]

\[
\kappa(M_{DA}) \leq k_0 \left[1 + k_0 \max_{1 \leq j \leq N} \left(1 + \frac{1}{\lambda_{m_j+1}}\right)\right]
\]
Local Eigenproblem and Global Coarse Space

- Let \((p_j^k)^{m_j}_{k=1}\) be the eigenvectors of
  \[ a_{\Omega_j}(p, v) = \lambda a_{\Omega_j^0}(\Xi_j(p), \Xi_j(v)) \quad \forall v \in V_h(\Omega_j) \]
corresponding to the \(m_j\) smallest eigenvalues.

- \(V_0 = \text{span}\{R_j^T\Xi_j(p_j^k) : k = 1, \ldots, m_j; j = 1, \ldots, N\}\)

Convergence Theorems

\[
\kappa(M_2A) \leq (1 + k_0) \left[ 2 + k_0(2k_0 + 1) \max_{1 \leq j \leq N} \left( 1 + \frac{1}{\lambda_{m_j+1}} \right) \right]
\]

\[
\kappa(M_D A) \leq k_0 \left[ 1 + k_0 \max_{1 \leq j \leq N} \left( 1 + \frac{1}{\lambda_{m_j+1}} \right) \right]
\]
Partition of Unity

Local Coarse Space

Global Coarse Space
My contribution (1/2)

Partition of Unity

\[ D_i = \mathcal{R}_{\Gamma_i} \left( \sum_{j=1}^{N} \mathcal{R}_{\Gamma_j}^T \mathcal{R}_{\Gamma_j} \right)^{-1} \mathcal{R}_{\Gamma_i}^T \]

Local Coarse Space

Global Coarse Space
My contribution (1/2)

Partition of Unity

\[ D_i = R_{\Gamma_i} \left( \sum_{j=1}^{N} R_{\Gamma_j}^T R_{\Gamma_j} \right)^{-1} R_{\Gamma_i}^T \]

Local Coarse Space

\[ V_0^i = \text{span}\{p_k^i, \quad S_i \ p_k^i = \chi_k^i \ D_i \tilde{S}_i D_i \ p_k^i \quad \text{with} \quad \chi_k^i \leq \eta \} \]

\( (S_i \text{ is SPSD}) \)

Global Coarse Space
My contribution (1/2)

Partition of Unity

\[ D_i = R_{\Gamma_i} \left( \sum_{j=1}^{N} R_{\Gamma_j}^T R_{\Gamma_j} \right)^{-1} R_{\Gamma_i}^T \]

Local Coarse Space

\[ V_i^0 = \text{span}\{ p_k^i, \quad S_i \quad p_k^i = \lambda_k^i D_i \tilde{S}_i D_i \quad p_k^i \quad \text{with} \quad \lambda_k^i \leq \eta \} \quad (S_i \text{ is SPSD}) \]

Global Coarse Space

\[ V_0 = \sum_{i=1}^{N} R_{\Gamma_i}^T D_i V_i^0 \]
Number of colors
Let $N_c$ be the minimal number of colors needed to assign a color $c_i$ to each subdomain $i$, such that:
\[ c_i = c_j \iff \mathcal{R}_{\Gamma_i}S\mathcal{R}_{\Gamma_j}^T = 0. \]

Convergence of the additive operator
\[
\kappa(\mathcal{M}_{\mathrm{AS}/S,2S}) \leq (1 + N_c) \left( N_c + 1 + \frac{N_c + 2}{\eta} \right)
\]

Convergence of the deflated operator
\[
\kappa(\mathcal{M}_{\mathrm{AS}/S,D\mathcal{S}}) \leq N_c \left( 1 + \frac{1}{\eta} \right)
\]
Outline of the proof: Fictitious Space Lemma

- Upper bound: coloring techniques
- Lower bound:
  - Existence of splittings \((u_i)_{1 \leq i \leq N}\) and \((v_i)_{1 \leq i \leq N}\) such that:

  \[
  u = R_0^T u_0 + \sum_{i=1}^{N} R_{T_i}^T u_i = R_0^T v_0 + (I - P_0) \sum_{i=1}^{N} R_{T_i}^T v_i.
  \]

  - Control the local norms of \((u_i)\) through the norm of \(u\):

    \[
    \sum_{i=0}^{N} \|u_i\|_{\bar{S}_i}^2 \leq \left( N_c + 1 + \frac{N_c + 2}{\eta} \right) \|u\|_{\bar{S}}^2,
    \]

    \[
    \sum_{i=0}^{N} \|v_i\|_{\bar{S}_i}^2 \leq \left( 1 + \frac{1}{\eta} \right) \|u\|_{\bar{S}}^2.
    \]

  - Use a Cauchy-Schwarz inequality to conclude.
Outline

1. Additive Schwarz on the Schur (AS/S)
   - AS/S step by step
   - Comparison with other DD preconditioners

2. MaPHyS solver
   - Software Framework
   - Distributed Subdomain Interface
   - Two-level Parallelism

3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results
3D Test problem

Heterogeneous diffusion

- $\nabla (K \nabla u) = 1$
- Alternating conductivity layers of 3 elements (ratio $K$ between layers)
- Dirichlet on the left, Neumann elsewhere

Domain decomposition

- $N \times 1 \times 1$ (1D decomposition)
- $N/2 \times 2 \times 1$ (2D decomposition)
- Constant subdomain size: $10 \times 10 \times 10$ elements

Implementation

- MPI+Python code (< 200 lines)
ev. threshold $\eta$: 0.01

Number of iterations to achieve convergence

Number of subdomains

Heterogeneity $K$
- $1$
- $100$
- $10000$

ID decomposition

Numerical method
- AS/S
The number of iterations is stabilized independently of $K$ and $N$. 
- More difficult problems require a bigger coarse space
\[
\kappa(M_{AS/S,2S}) \leq (1 + N_c) \left( N_c + 1 + \frac{N_c + 2}{\eta} \right) \kappa(M_{AS/S,DS}) \leq N_c \left( 1 + \frac{1}{\eta} \right)
\]
Perspectives

GenEO in MaPHyS

- Loosening the assumptions ($A_i$ SPSD and $A$ SPD)
- Implementation and test of the 2-level preconditioner on real applications

Other recent/ongoing efforts in MaPHyS

- Partitioning/balancing both interface and interior vertices (A. Casadei)
- Parallel analysis and dist. sub. API (M. Kuhn)
- $\mathcal{H}$-arithmetic for local solve ($\mathcal{H}$-PaStiX) and preconditioner (A. Falco, G. Pichon, Y. Harness)
- Numerical resilience policies (M. Zounon)
- Task-based implementation (S. Nakov)
Thanks for your attention!

Questions?

Funded by the Dedales ANR Project
Outline

- ANR
- 2-level parallelism
- Subdomain Interface
- Figures
Outline

- ANR
  - 2-level parallelism
  - Subdomain Interface
  - Figures
ANR DEDALES project

Goal:
- High performance software for the simulation of two phase flow in porous media

Challenges:
- Very large problems
- Highly heterogeneous medium, widely varying space and time scales

Solution:
- Improved Domain Decomposition algorithms
- Parallel hybrid linear solver

Partners:
Outline

- ANR
- 2-level parallelism
- Subdomain Interface
- Figures
MPI Parallelism in MaPHyS

Factorization  ↓ ↓ ↓ ↓ ↓ ↓
Preconditioner Setup  ↓ ↓ ↓ ↓ ↓ ↓
Solve  ↓ ↓ ↓ ↓ ↓ ↓
MPI + threads Parallelism in MaPHyS

Factorization
Preconditioner Setup
Solve
Outline

- ANR
- 2-level parallelism
- Subdomain Interface
- Figures
Global data

- myndof: number of degree of freedom
- mysizeintrf: number of interface nodes

Local data

- $A_i, b_i$
- myinterface(:): interface node list in global ordering
- mynbvi: number of neighbor processes
- myindexVi(:): list of neighbor processes (MPI ranks)
- myptrindexVi(:): pointer to common interface nodes of neighbors
- mynindexintrf(:): common interface node list of neighbors
Outline

- ANR
- 2-level parallelism
- Subdomain Interface
- Figures
\[ \lambda_{\text{max}}(M_{\text{AS}/S,2S}) \leq N_c + 1 \quad \lambda_{\text{max}}(M_{\text{AS}/S,Ds}) \leq N_c \]
\[ \lambda_{\min}(\mathcal{M}_{AS/S,2S}) \geq \frac{1}{N_c+1 + \frac{N_c+2}{\eta}} \quad \lambda_{\min}(\mathcal{M}_{AS/S,DS}) \geq \frac{1}{1+\frac{1}{\eta}} \]