HPCDD 04/07/2016

Semi-Algebraic Coarse Space for Parallel Sparse Hybrid Solvers

Louis Poirel   Emmanuel Agullo   Luc Giraud

HiePACS Project Team
Inria Bordeaux Sud-Ouest

Introduction

Goal

- Solve $\mathcal{A}x = b$, where $\mathcal{A}$ 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

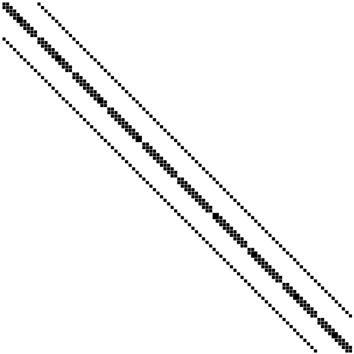
3. Two-level preconditioner for AS/S
   - Need for Coarse Correction
   - Coarse Space for AS/S
   - Experimental results

Outline

Outline

Step 1: Analysis

Global Matrix $\mathcal{A}$

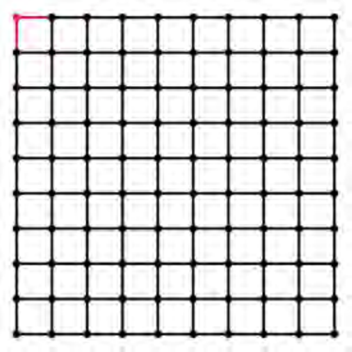


- $\mathcal{A}$ is a general sparse matrix. We want to solve $\mathcal{A}x = b$.

Step 1: Analysis

Global Matrix $\mathcal{A}$

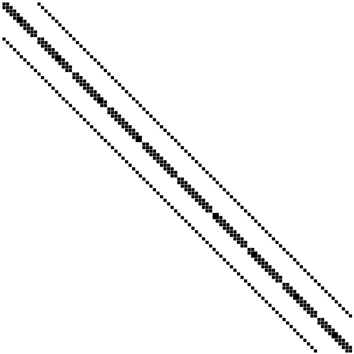
Adjacency graph G



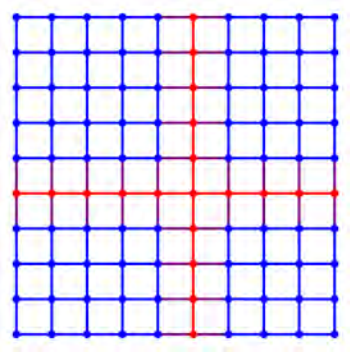
- The adjacency graph of $\mathcal{A}$ $(n \times n)$ is used as an algebraic mesh: $G = (\{1, \ldots, n\}, \{(i,j), \; a_{ij} \neq 0 \mid a_{ji} \neq 0\})$
  - On the first row of $\mathcal{A}$, $a_{1,1}$, $a_{1,2}$ and $a_{1,11} \neq 0$ $\Rightarrow (1,1)$, $(1,2)$ and $(1,11) \in G$

Step 1: Analysis

Global Matrix $\mathcal{A}$



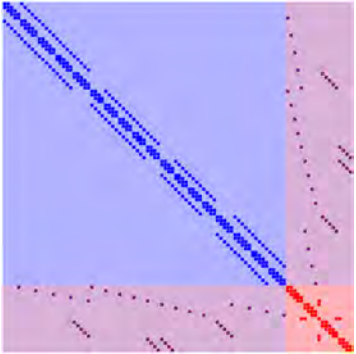
Adjacency graph G



- A graph partitioner is used to split the graph

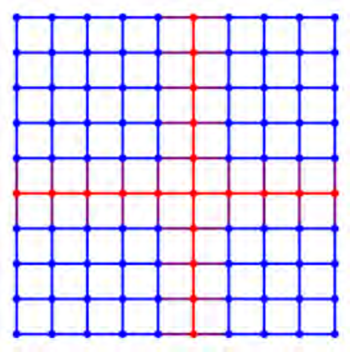
Step 1: Analysis

Global Matrix $\mathcal{A}$



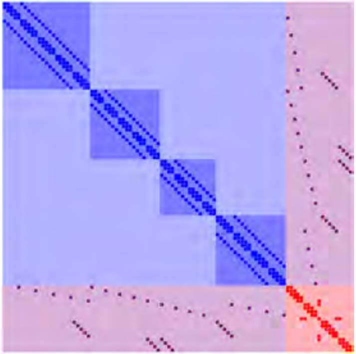
$$\begin{pmatrix} \mathcal{A}_{\mathcal{I}\mathcal{I}} & \mathcal{A}_{\mathcal{I}\Gamma} \\ \mathcal{A}_{\Gamma\mathcal{I}} & \mathcal{A}_{\Gamma\Gamma} \end{pmatrix} \begin{pmatrix} x_{\mathcal{I}} \\ x_{\Gamma} \end{pmatrix} = \begin{pmatrix} b_{\mathcal{I}} \\ b_{\Gamma} \end{pmatrix}$$

Adjacency graph G

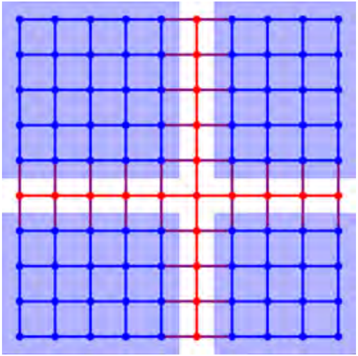


Step 1: Analysis

Global Matrix $\mathcal{A}$



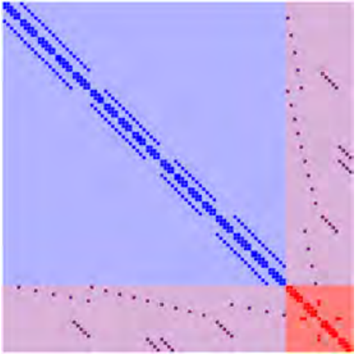
Adjacency graph G



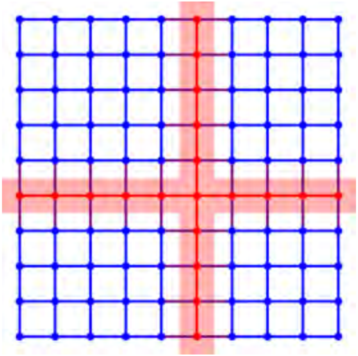
- $\mathcal{A}_{\mathcal{II}}$ has a block diagonal structure suitable for parallel computation

Step 1: Analysis

Global Matrix $\mathcal{A}$



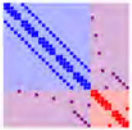
Adjacency graph G



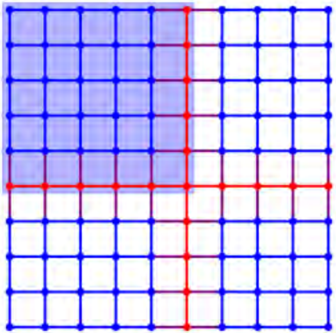
- How do we distribute $\mathcal{A}_{\Gamma\Gamma}$?

Step 1: Analysis

Local Matrix $\mathcal{A}_i$



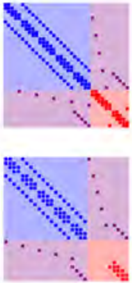
Adjacency graph G



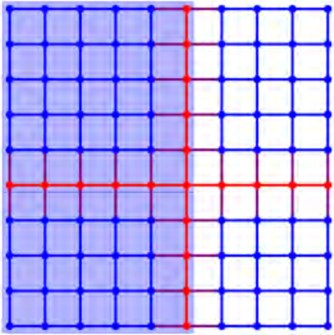
- We assign each interface node to a neighboring subdomain

Step 1: Analysis

Local Matrix $\mathcal{A}_i$



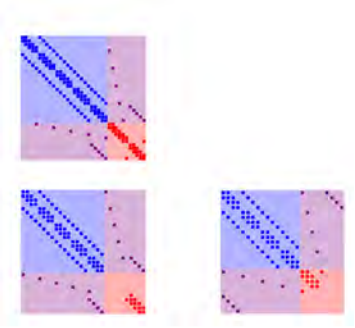
Adjacency graph G



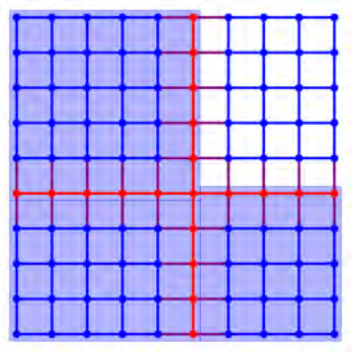
- We assign each interface node to a neighboring subdomain

Step 1: Analysis

Local Matrix $\mathcal{A}_i$



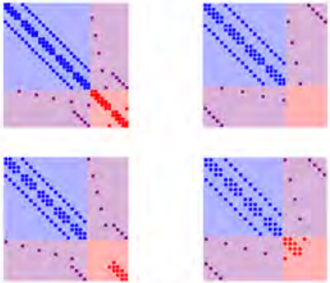
Adjacency graph G



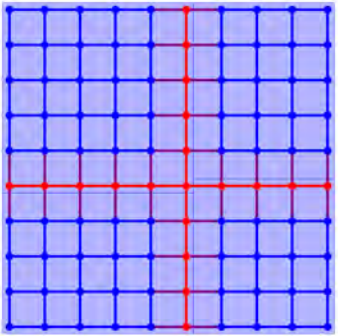
- We assign each interface node to a neighboring subdomain

Step 1: Analysis

Local Matrix $\mathcal{A}_i$



Adjacency graph G

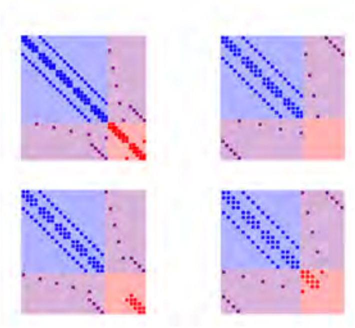


- We assign each interface node to a neighboring subdomain

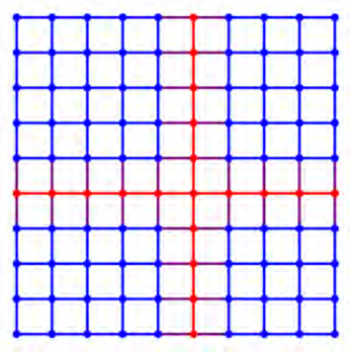
$$\mathcal{A}_i = \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\Gamma_i\mathcal{I}_i} & \mathcal{A}_{\Gamma_i\Gamma_i} \end{pmatrix} \qquad \mathcal{A} = \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i \mathcal{R}_i$$

Step 2: Factorization

Local Matrix $\mathcal{A}_i$



Adjacency graph G

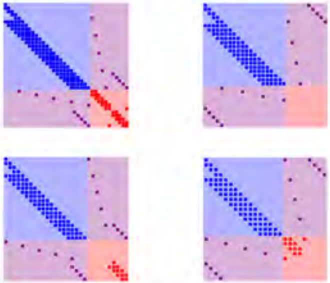


- We factorize $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}$ and compute $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

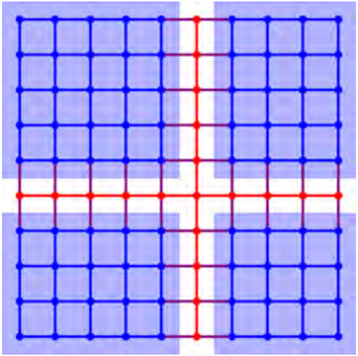
$$\mathcal{A}_i = \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\Gamma_i\mathcal{I}_i} & \mathcal{A}_{\Gamma_i\Gamma_i} \end{pmatrix}$$

Step 2: Factorization

Local Matrix $\mathcal{A}_i$



Adjacency graph G

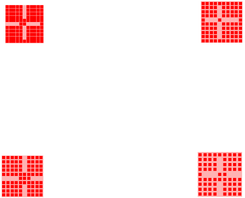


- We factorize $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}$ and compute $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

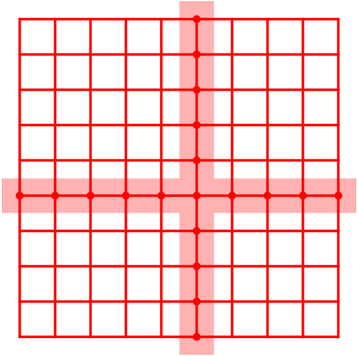
$$\mathcal{A}_i = \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\Gamma_i\mathcal{I}_i} & \mathcal{A}_{\Gamma_i\Gamma_i} \end{pmatrix}$$

Step 2: Factorization

Local Schur $\mathcal{S}_i$



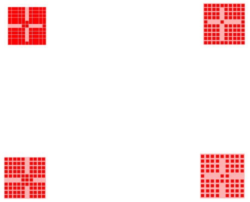
Adjacency graph G



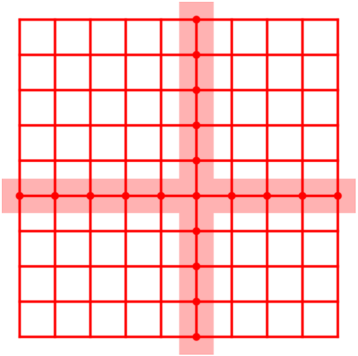
- We factorize $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}$ and compute $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$
- Now, on each subdomain, the whole local problem is condensed onto the interface (dense matrix)

Step 2: Factorization

Local Schur $\mathcal{S}_i$



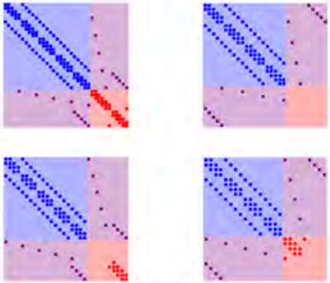
Adjacency graph G



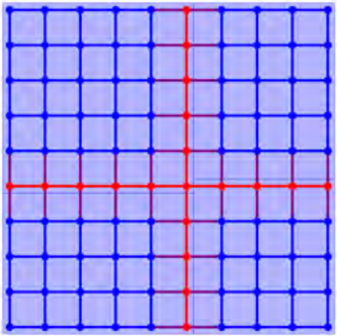
- We solve the interface problem $\mathcal{S}x_\Gamma = f = b_\Gamma - \mathcal{A}_{\Gamma\mathcal{I}}\mathcal{A}_{\mathcal{I}\mathcal{I}}^{-1}b_\mathcal{I}$ with a preconditioned Krylov method

AS Preconditioner

Local Matrix $\mathcal{A}_i$



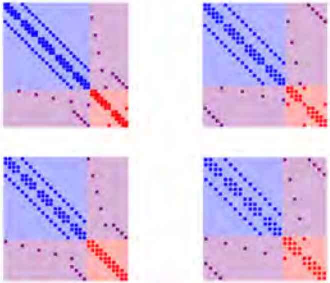
Adjacency graph G



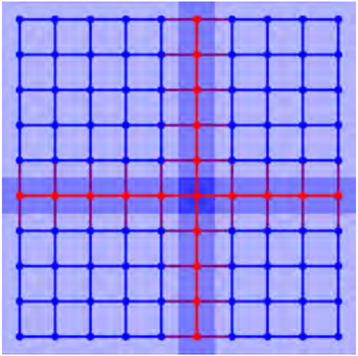
- No overlap in $\mathcal{A}_i$: $\mathcal{A} = \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i \mathcal{R}_i$

AS Preconditioner

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



Adjacency graph G

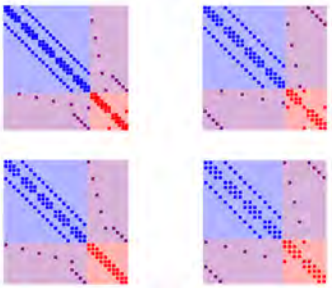


- No overlap in $\mathcal{A}_i$: $\mathcal{A} = \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i \mathcal{R}_i$
- Assemble $\bar{\mathcal{A}}_i = \mathcal{R}_i \mathcal{A} \mathcal{R}_i^T$ using neighbor-to-neighbor communications

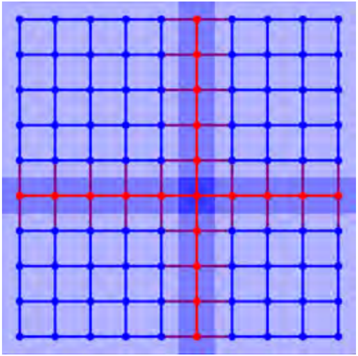
- $\mathcal{M}_{AS} = \sum_{i=1}^{N} \mathcal{R}_i^T \bar{\mathcal{A}}_i^{-1} \mathcal{R}_i$

AS Preconditioner

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



Adjacency graph G



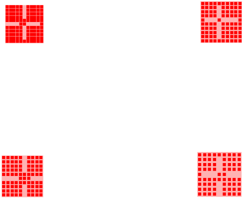
- No overlap in $\mathcal{A}_i$: $\mathcal{A} = \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i \mathcal{R}_i$
- Assemble $\bar{\mathcal{A}}_i = \mathcal{R}_i \mathcal{A} \mathcal{R}_i^T$ using neighbor-to-neighbor communications

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

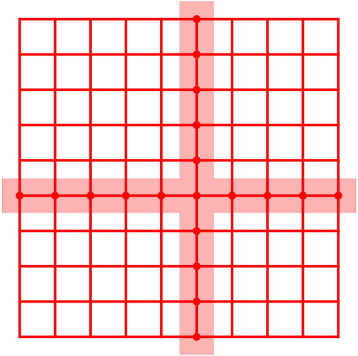
Not what we do

Step 3: Preconditioner Setup (AS/S)

Local Schur $\mathcal{S}_i$



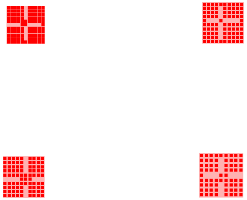
Adjacency graph G



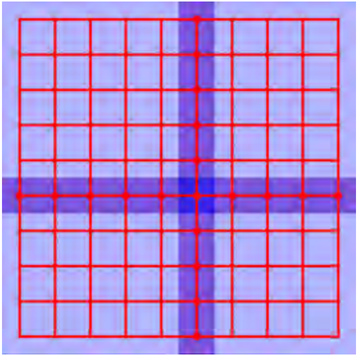
- No overlap in $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$: $\quad \mathcal{S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \mathcal{S}_i \mathcal{R}_{\Gamma_i}$

Step 3: Preconditioner Setup (AS/S)

Assembled Local Schur $\bar{\mathcal{S}}_i$



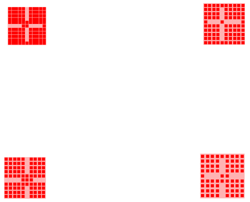
Adjacency graph G



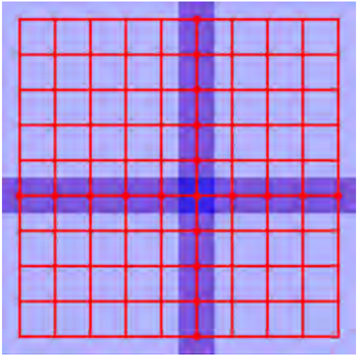
- No overlap in $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$: $\quad \mathcal{S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \mathcal{S}_i \mathcal{R}_{\Gamma_i}$
- Assemble $\bar{\mathcal{S}}_i = \mathcal{R}_{\Gamma_i}\mathcal{S}\mathcal{R}_{\Gamma_i}$
  - $\mathcal{M}_{AS/S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \bar{\mathcal{S}}_i^{-1} \mathcal{R}_{\Gamma_i}$

Step 3: Preconditioner Setup (AS/S)

Assembled Local Schur $\bar{S}_i$



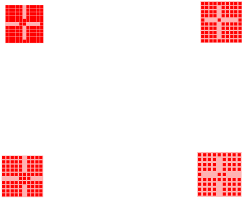
Adjacency graph G



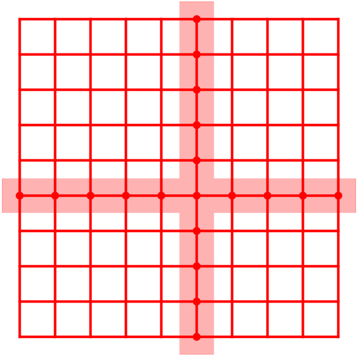
- Share not only the $\mathcal{A}_{\Gamma_i\Gamma_i}$ part, but also $\mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$
  - The neighbor's interiors are condensed on the subdomain's interface too.

Step 4: Solve

Local Schur $\mathcal{S}_i$



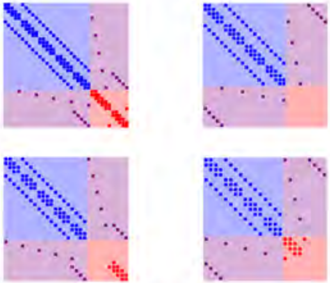
Adjacency graph G



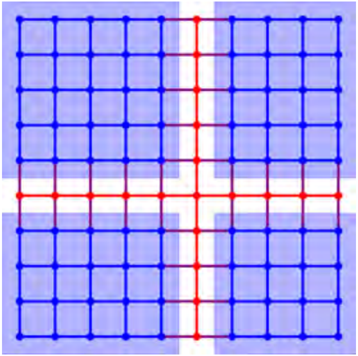
- on Γ: *Krylov method*
  - $\mathcal{S}\, x_\Gamma = f$ preconditioned with $\mathcal{M}_{AS/S}$

Step 4: Solve

Local Matrix $\mathcal{A}_i$



Adjacency graph G



- on Γ: *Krylov method*
  - $\mathcal{S}\, x_\Gamma = f$ preconditioned with $\mathcal{M}_{AS/S}$
- on $\mathcal{I}$: *Direct method*
  - $x_{\mathcal{I}_i} = \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1} \left(b_{\mathcal{I}_i} - \mathcal{A}_{\mathcal{I}_i\Gamma_i} x_{\Gamma_i}\right)$

Step 1: Analysis

- Graph partitioning and data distribution

Step 2: Factorization

- Computation of $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}$ and $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

Step 3: Preconditioner Setup

- Assembly and factorization of $\bar{\mathcal{S}}_i$

Step 4: Solve

- on Γ: *Krylov method*
  - $\mathcal{S}\, x_\Gamma = f$ preconditioned with $\mathcal{M}_{AS/S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \bar{\mathcal{S}}_i^{-1} \mathcal{R}_{\Gamma_i}$
- on $\mathcal{I}$: *Direct method*
  - $x_{\mathcal{I}_i} = \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\left(b_{\mathcal{I}_i} - \mathcal{A}_{\mathcal{I}_i\Gamma_i} x_{\Gamma_i}\right)$

Outline

Related DD preconditioners

- Neumann-Neumann (NN)
  - $\mathcal{M}_{NN} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \; D_i \mathcal{S}_i^{\dagger} D_i \; \mathcal{R}_{\Gamma_i}$ where D_i is a partition of unity and $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

- Schur of Additive Schwarz (S−AS)
  - $\mathcal{M}_{S-AS} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \; \hat{\mathcal{S}}_i^{-1} \; \mathcal{R}_{\Gamma_i}$ where $\bar{\mathcal{A}}_{\Gamma_i\Gamma_i} = \sum_{j=1}^{N} \mathcal{R}_{\Gamma_i}\mathcal{R}_{\Gamma_j}^T \; \mathcal{A}_{\Gamma_j\Gamma_j} \; \mathcal{R}_{\Gamma_j}\mathcal{R}_{\Gamma_i}^T$ and $\hat{\mathcal{S}}_i = \bar{\mathcal{A}}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$ is the Schur of $\bar{\mathcal{A}}_i$

- Additive Schwarz on the Schur (AS/S)
  - $\mathcal{M}_{AS/S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \; \bar{\mathcal{S}}_i^{-1} \; \mathcal{R}_{\Gamma_i}$ where $\bar{\mathcal{S}}_i = \mathcal{R}_{\Gamma_i} \; \mathcal{S} \; \mathcal{R}_{\Gamma_i}^T$

$$\bar{\mathcal{S}}_i = \sum_{j=1}^{N} \mathcal{R}_{\Gamma_i}\mathcal{R}_{\Gamma_j}^T \; \left(\mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}\right) \; \mathcal{R}_{\Gamma_j}\mathcal{R}_{\Gamma_i}^T$$

3D Test problem

Heterogeneous diffusion

- $\nabla(K\nabla u) = 1$
- Alternating conductivity layers of 3 elements

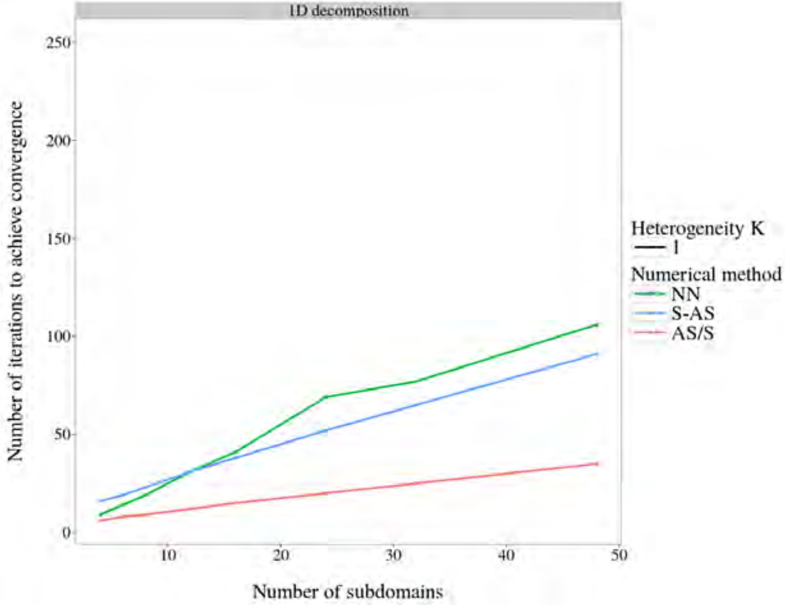
(ratio K $= K_{\text{max}}/K_{\text{min}}$ between layers)

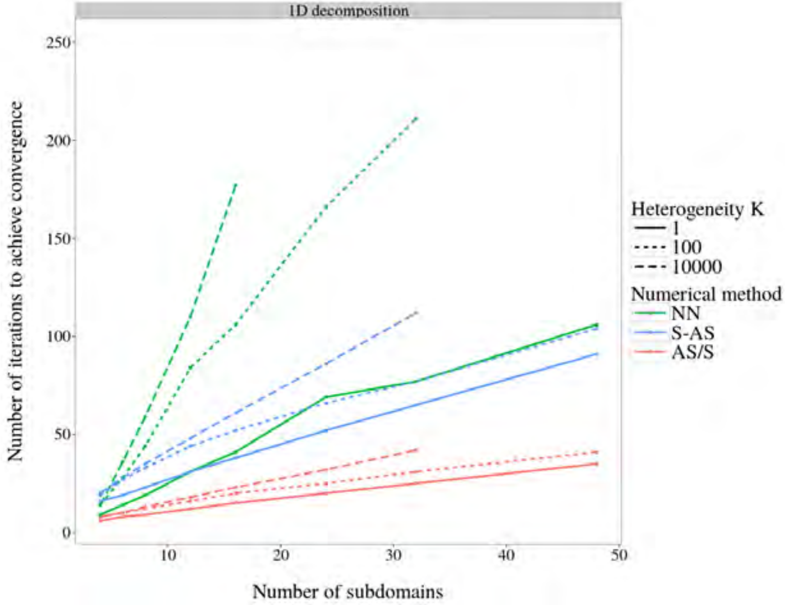
Domain decomposition

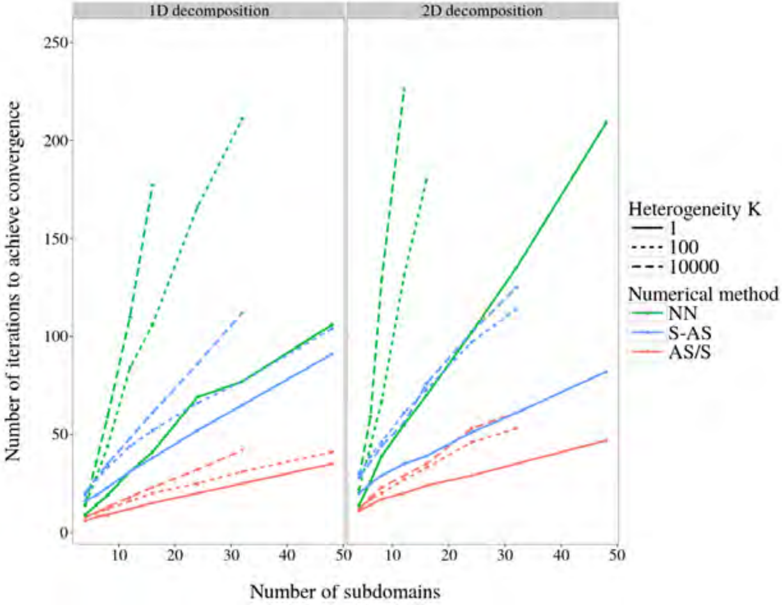
- Constant subdomain size: $10 \times 10 \times 10$ elements
- N subdomains
  - $N \times 1 \times 1$ (1D decomposition)
  - $N/2 \times 2 \times 1$ (2D decomposition)

Boundary conditions

- Dirichlet on the left
- Neumann elsewhere


1D decomposition
250
200
150
100
50
0
Number of iterations to achieve convergence
10
20
30
40
50
Number of subdomains
Heterogeneity K
1
Numerical method
NN
S-AS
AS/S


1D decomposition
Number of iterations to achieve convergence
0
50
100
150
200
250
10
20
30
40
50
Number of subdomains
Heterogeneity K
1
100
10000
Numerical method
NN
S-AS
AS/S



1D decomposition
2D decomposition
Number of iterations to achieve convergence
0
50
100
150
200
250
10
20
30
40
50
10
20
30
40
50
Number of subdomains
Heterogeneity K
1
100
10000
Numerical method
NN
S-AS
AS/S

Outline

Outline

Step by step

Step 1: Analysis

- Graph partitioning and data distribution

Step 2: Factorization

- Computation of $\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}$ and $\mathcal{S}_i = \mathcal{A}_{\Gamma_i\Gamma_i} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$

Step 3: Preconditioner Setup

- Assembly and factorization of $\bar{\mathcal{S}}_i$

Step 4: Solve

- on Γ: *Krylov method*
  - $\mathcal{S}\, x_\Gamma = f$ preconditioned with $\mathcal{M}_{AS/S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \bar{\mathcal{S}}_i^{-1} \mathcal{R}_{\Gamma_i}$
- on $\mathcal{I}$: *Direct method*
  - $x_{\mathcal{I}_i} = \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}^{-1}(b_{\mathcal{I}_i} - \mathcal{A}_{\mathcal{I}_i\Gamma_i} x_{\Gamma_i})$

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]

Use it!

project.inria.fr/maphys/

Installing MaPHyS

- MaPHyS and its dependencies can be installed through `spack` in ≤ 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

Interfaces for MaPHyS

Application			$\downarrow$		
Analysis			$\downarrow$		
Factorization	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Preconditioner Setup	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Solve	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$

Centralized Matrix Interface

- Application provides global matrix $\mathcal{A}$ on one process
- MaPHyS performs algebraic domain decomposition and data distribution

Interfaces for MaPHyS

Application	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Analysis	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Factorization	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Preconditioner Setup	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Solve	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$

Distributed matrix interface

- Application provides global matrix $\mathcal{A}$ in a distributed way
- MaPHyS performs parallel algebraic domain decomposition and data redistribution

Interfaces for MaPHyS

Application	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Factorization	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Preconditioner Setup	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Solve	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$

Distributed subdomain interface

- Application performs domain decomposition and provides subdomain connectivity and local matrices $\mathcal{A}_i$ in a distributed way
- Analysis is bypassed

Interfaces for MaPHyS

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

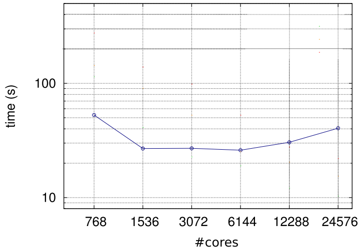
Distributed subdomain interface

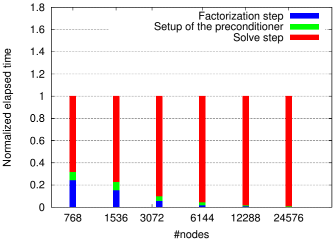
- Application performs domain decomposition and provides subdomain connectivity and local matrices $\mathcal{A}_i$ in a distributed way
- Analysis is bypassed

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

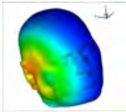
Outline

Two-level Parallelism [S. Nakov]

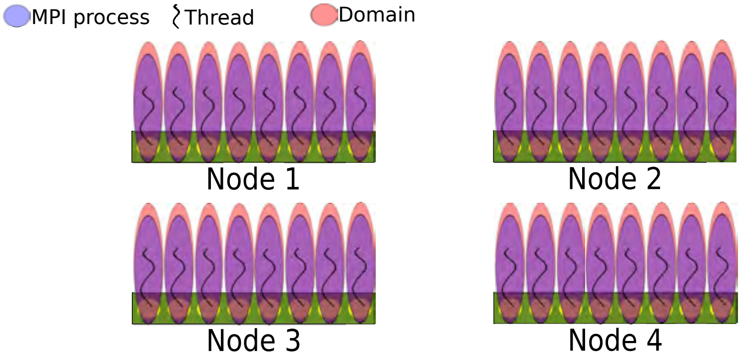




	Nachos4M
N	$4.1M$
Nnz	$256.4M$



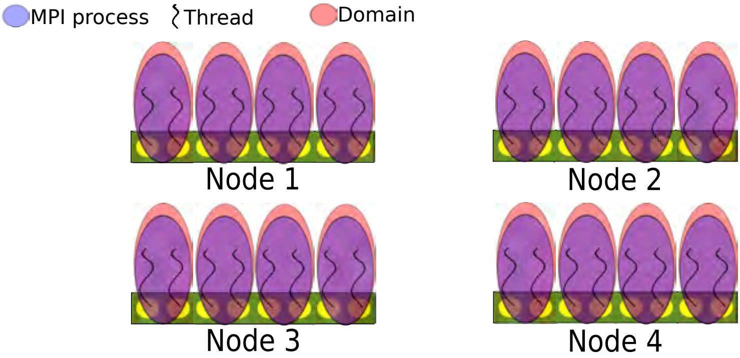
Two-level Parallelism [S. Nakov]



1 thread per process (32 domains in total)

- One subdomain per core leads to a huge number of subdomains on modern architectures
    - Lack of robustness

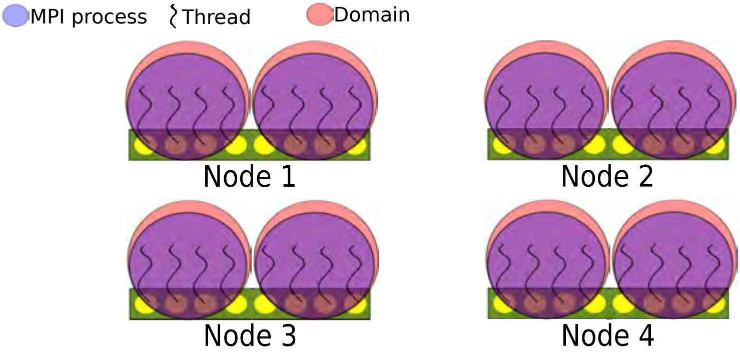
Two-level Parallelism [S. Nakov]



2 threads per process (16 domains in total)

- Multithreaded subdomains → fewer and bigger subdomains
  - Bigger local problem to solve ☹
  - Smaller and better-conditioned interface problem ☺

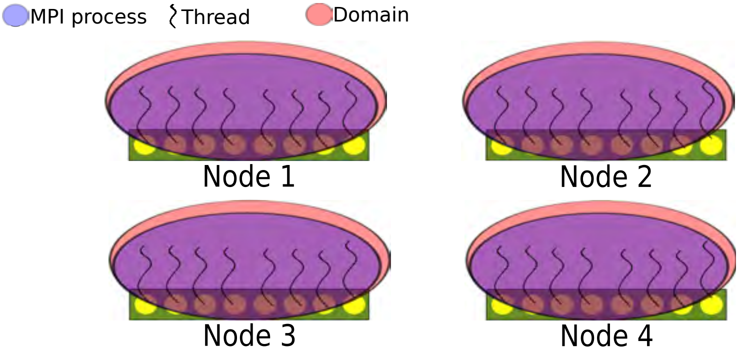
Two-level Parallelism [S. Nakov]



4 threads per process (8 domains in total)

- Multithreaded subdomains → 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 ☺

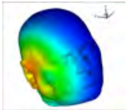
Results (2-level parallelism)

Hopper Platform (NERSC)

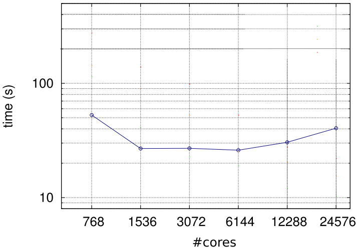
- Two twelve-core AMD 'MagnyCours' 2.1-GHz
- Memory: 32 GB GDDR3
- Double precision

Matrix

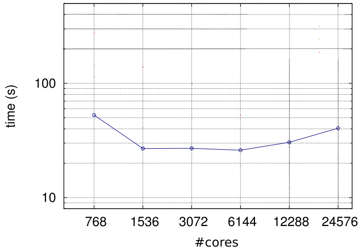
	Nachos4M
N	$4.1M$
Nnz	$256.4M$

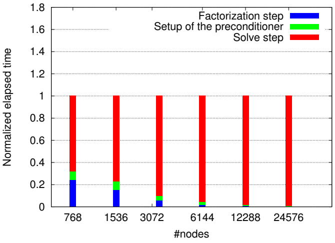


Results (2-level parallelism)

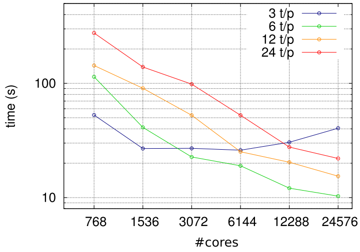


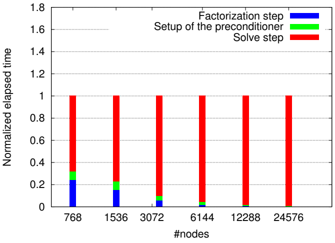
Results (2-level parallelism)

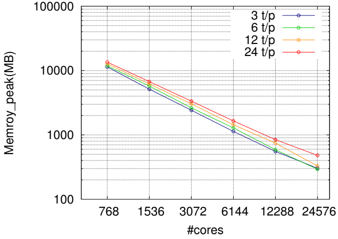




Results (2-level parallelism)







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

Context

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 $\mathcal{A}_i$ to be Symmetric Positive Semi-Definite (SPSD)

    (e.g. through Distributed Subdomain Interface)
- Experimental results
  - Python/MPI prototype

Outline

2D Test problem

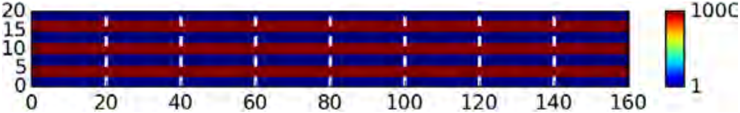
Heterogeneous diffusion

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

Boundary conditions

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

Conductivity K ($N = 8$ subdomains)



2D Test problem

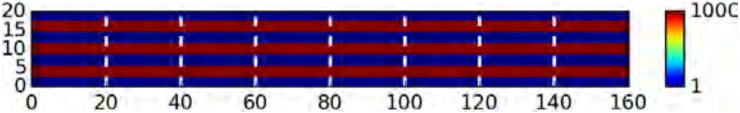
Heterogeneous diffusion

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

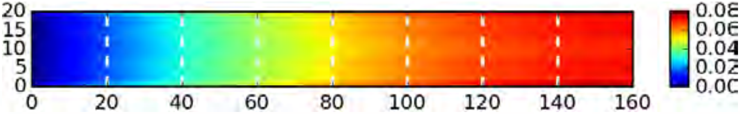
Boundary conditions

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

Conductivity K ($N = 8$ subdomains)

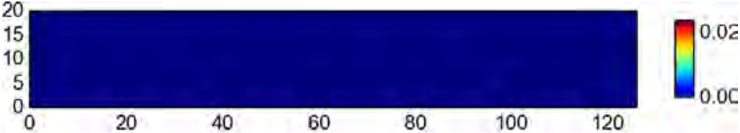


Solution x^* ($N = 8$ subdomains)



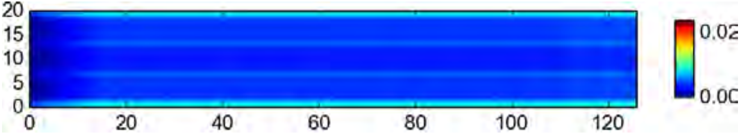
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 0$



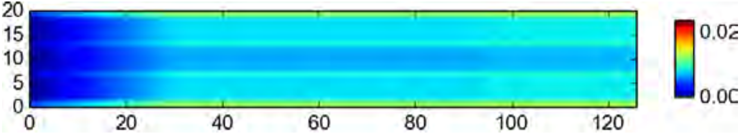
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 10$



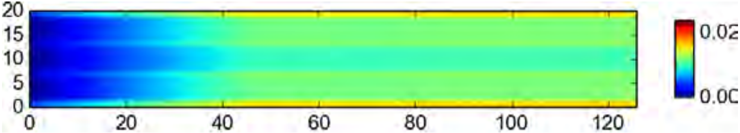
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 20$



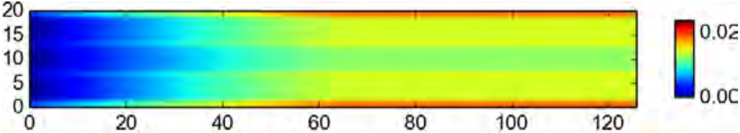
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 30$



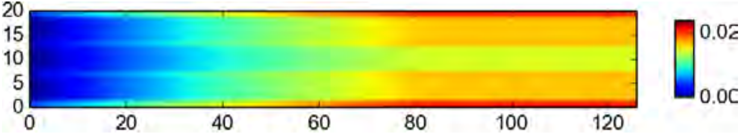
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 40$



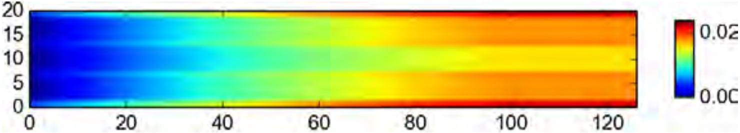
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 50$



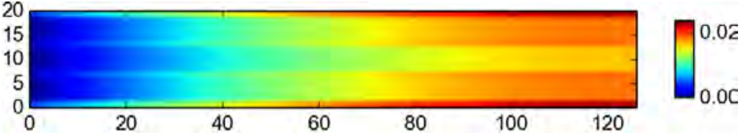
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 60$



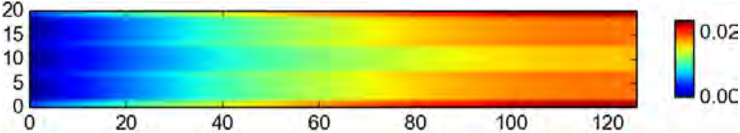
Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 70$



Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 70$

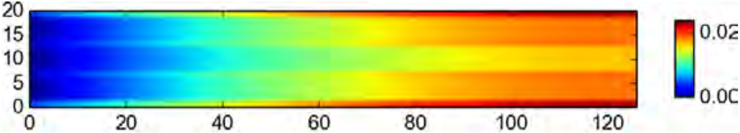


Problem

- No global exchange of information
- Algebraic bound on $\lambda_{\max}(\mathcal{M}_{AS/S}S)$, but problem with $\lambda_{\min}$

Convergence Behavior

x_Γ, $N = 128$, $n_{iter} = 70$



Problem

- No global exchange of information
- Algebraic bound on $\lambda_{\max}(\mathcal{M}_{AS/S}S)$, but problem with $\lambda_{\min}$

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

V_0		Basis of the coarse space
$\mathcal{R}_0$	$= V_0^T$	Restriction to the coarse space
$\bar{\mathcal{S}}_0$	$= \mathcal{R}_0 \mathcal{S} \mathcal{R}_0^T$	Coarse matrix
$\mathcal{M}_0$	$= \mathcal{R}_0^T \bar{\mathcal{S}}_0^{-1} \mathcal{R}_0$	Coarse solve
$\mathcal{P}_0$	$= \mathcal{M}_0 \mathcal{S}$	$\mathcal{S}$-orthogonal projection on V_0

Coarse Correction for AS

2-level Additive Preconditioner

$$\mathcal{M}_{AS,2} = \mathcal{M}_0 + \mathcal{M}_{AS}$$

Deflated Preconditioner

$$\mathcal{M}_{AS,D} = \mathcal{M}_0 + (\mathcal{I} - \mathcal{P}_0)\,\mathcal{M}_{AS}\,(\mathcal{I} - \mathcal{P}_0)^T$$

Coarse Correction for AS

2-level Additive Preconditioner

$$\mathcal{M}_{AS,2} = \mathcal{M}_0 + \mathcal{M}_{AS}$$

Deflated Preconditioner

$$\mathcal{M}_{AS,D} = \mathcal{M}_0 + (\mathcal{I} - \mathcal{P}_0)\,\mathcal{M}_{AS}\,(\mathcal{I} - \mathcal{P}_0)^T$$

Outline

GenEO coarse space [N. Spillane 2014]

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

- $\mathcal{A}$ 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

GenEO coarse space [N. Spillane 2014]

Local Eigenproblem and Global Coarse Space

- Let $(p_j^k)_{k=1}^{m_j}$ be the eigenvectors of

$$a_{\Omega_j}(p, v) = \lambda \ a_{\Omega_j^\circ}(\Xi_j(p), \Xi_j(v)) \qquad \forall v \in V_h(\Omega_j)$$

corresponding to the m_j smallest eigenvalues.

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

Convergence Theorems

$$\kappa(\mathcal{M}_2\mathcal{A}) \leq (1 + k_0) \left[2 + k_0(2k_0 + 1) \max_{1 \leq j \leq N} \left(1 + \frac{1}{\lambda_{m_j+1}}\right)\right]$$

$$\kappa(\mathcal{M}_D\mathcal{A}) \leq k_0 \left[1 + k_0 \max_{1 \leq j \leq N} \left(1 + \frac{1}{\lambda_{m_j+1}}\right)\right]$$

GenEO coarse space [N. Spillane 2014]

Local Eigenproblem and Global Coarse Space

- Let $(p_j^k)_{k=1}^{m_j}$ be the eigenvectors of

$$a_{\Omega_j}(p, v) = \lambda \; a_{\Omega_j^\circ}(\Xi_j(p), \Xi_j(v)) \qquad \forall v \in V_h(\Omega_j)$$

corresponding to the m_j smallest eigenvalues.

- $V_0 = \text{span}\{\mathcal{R}_j^T \Xi_j(p_j^k) \; : \; k = 1, \dots, m_j; \; j = 1, \dots, N\}$

Convergence Theorems

$$\kappa(\mathcal{M}_2\mathcal{A}) \leq (1 + k_0) \left[2 + k_0(2k_0 + 1) \max_{1 \leq j \leq N} \left(1 + \frac{1}{\lambda_{m_j+1}} \right) \right]$$

$$\kappa(\mathcal{M}_D\mathcal{A}) \leq k_0 \left[1 + k_0 \max_{1 \leq j \leq N} \left(1 + \frac{1}{\lambda_{m_j+1}} \right) \right]$$

My contribution (1/2)

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

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

Local Coarse Space

- $V_0^i = \text{span}\{p_k^i, \quad \mathcal{S}_i \; p_k^i = \lambda_k^i \; D_i \bar{\mathcal{S}}_i D_i \; p_k^i \quad \text{with} \quad \lambda_k^i \leq \eta\}$
  ($\mathcal{S}_i$ is SPSD)

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

- $V_0^i = \text{span}\{p_k^i, \quad \mathcal{S}_i \; p_k^i = \lambda_k^i \; D_i \bar{\mathcal{S}}_i D_i \; p_k^i \quad \text{with} \quad \lambda_k^i \leq \eta\}$
  ($\mathcal{S}_i$ is SPSD)

Global Coarse Space

- $V_0 = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T D_i V_0^i$

My contribution (2/2)

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:

$$c_i = c_j \iff \mathcal{R}_{\Gamma_i} \mathcal{S} \mathcal{R}_{\Gamma_j}^T = 0.$$

Convergence of the additive operator

$$\kappa(\mathcal{M}_{AS/S,2}\mathcal{S}) \leq (1 + N_c)\left(N_c + 1 + \frac{N_c + 2}{\eta}\right)$$

Convergence of the deflated operator

$$\kappa(\mathcal{M}_{AS/S,D}\mathcal{S}) \leq N_c\left(1 + \frac{1}{\eta}\right)$$

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:

$$u = \mathcal{R}_0^T u_0 + \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T u_i = \mathcal{R}_0^T v_0 + (\mathcal{I} - \mathcal{P}_0) \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T v_i.$$

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

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

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

  - Use a Cauchy-Schwarz inequality to conclude.

Outline

3D Test problem

Heterogeneous diffusion

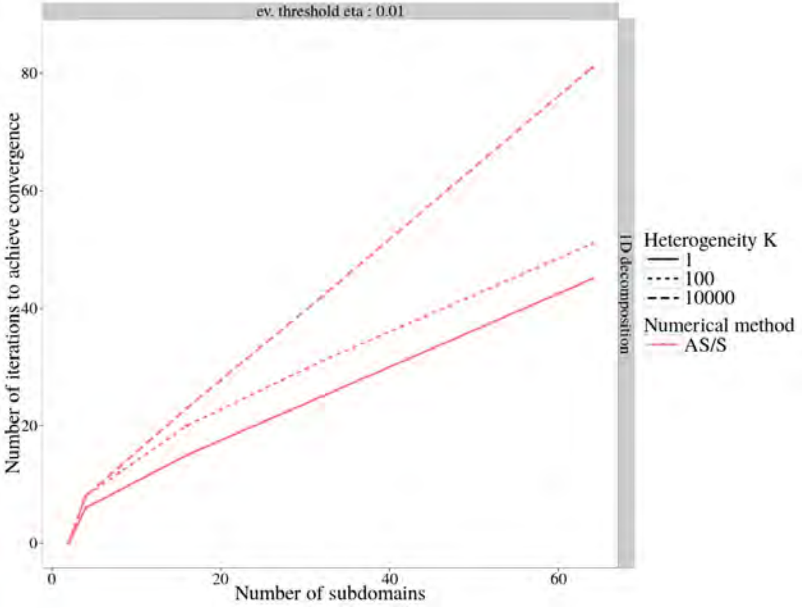
- $\nabla(K\nabla u) = 1$
- Alternating conductivity layers of 3 elements (ratio K between layers)
- Dirichlet on the left, Neumann elsewhere

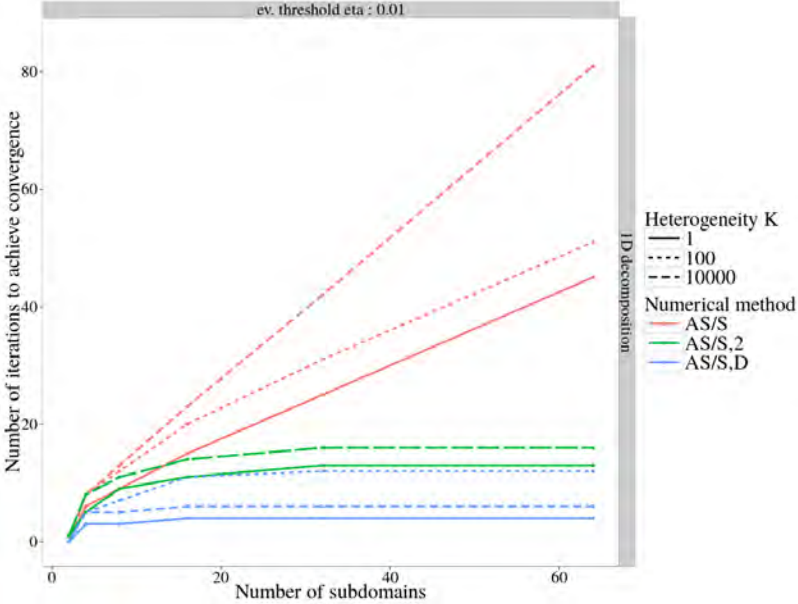
Domain decomposition

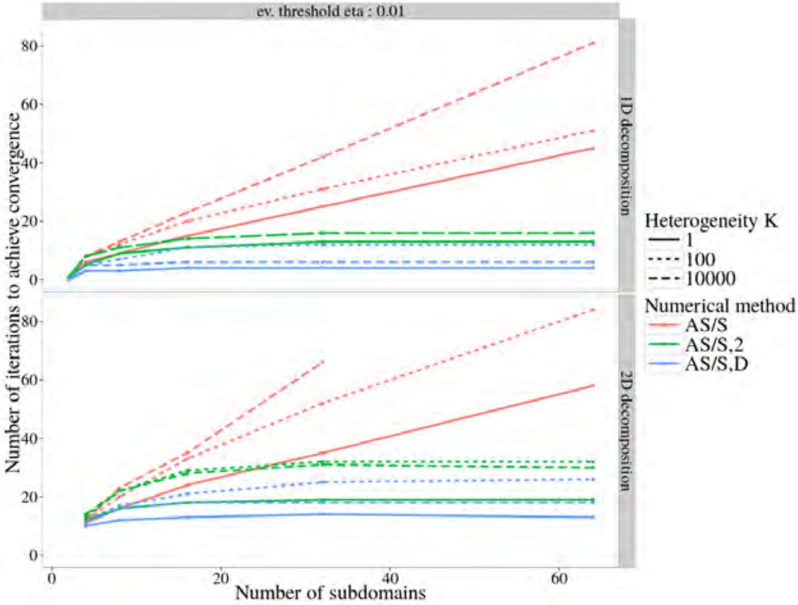
- $N \times 1 \times 1$ (1D decomposition)
- $N/2 \times 2 \times 1$ (2D decomposition)
- Constant subdomain size: $10 \times 10 \times 10$ elements

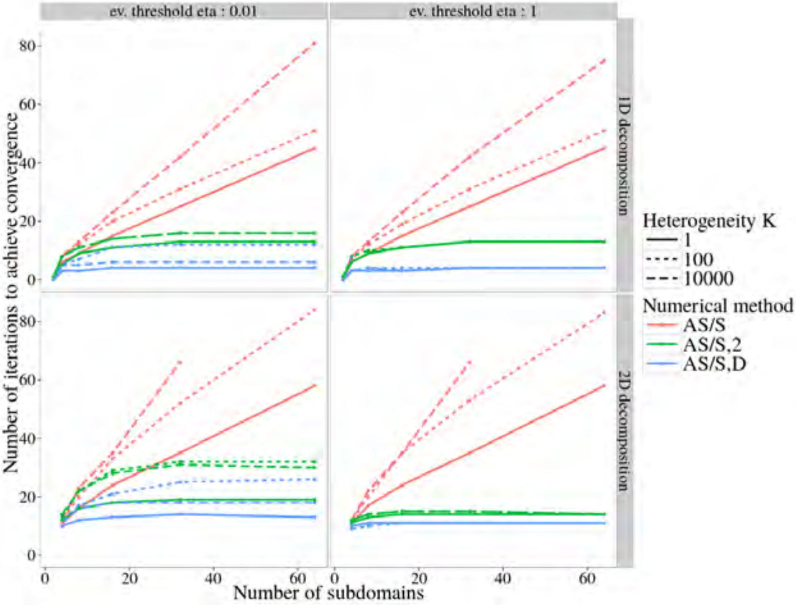
Implementation

- MPI+Python code ($<$ 200 lines)

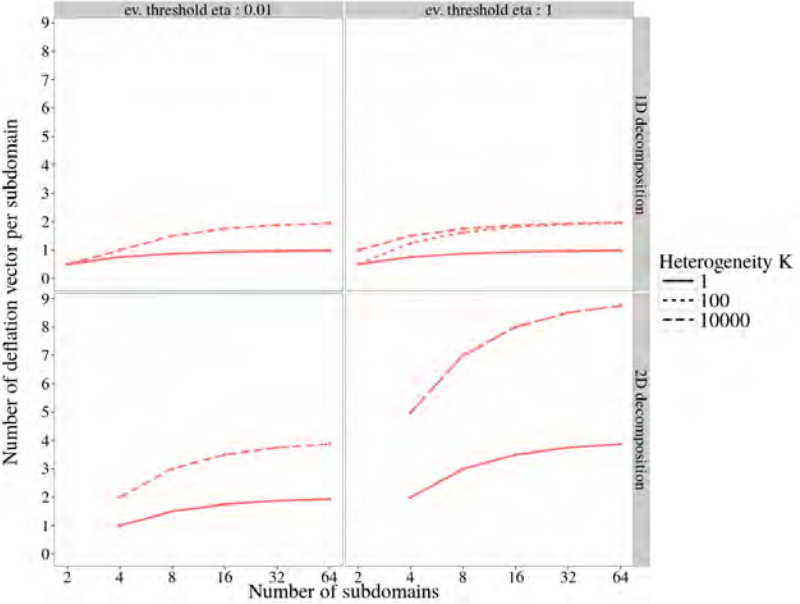

ev. threshold eta : 0.01
Number of iterations to achieve convergence
0
20
40
60
80
0
20
40
60
Number of subdomains
1D decomposition
Heterogeneity K
1
100
10000
Numerical method
AS/S


ev. threshold eta : 0.01
Number of iterations to achieve convergence
0
20
40
60
80
Number of subdomains
0
20
40
60
1D decomposition
Heterogeneity K
1
100
10000
Numerical method
AS/S
AS/S,2
AS/S,D

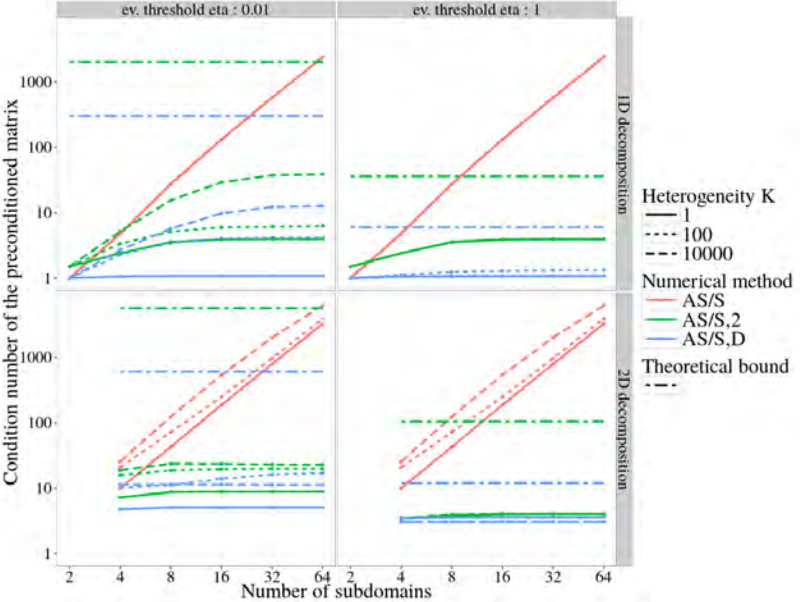

ev. threshold eta : 0.01
Number of iterations to achieve convergence
Number of subdomains
1D decomposition
2D decomposition
0
20
40
60
80
0
20
40
60
80
0
20
40
60
Heterogeneity K
1
100
10000
Numerical method
AS/S
AS/S,2
AS/S,D



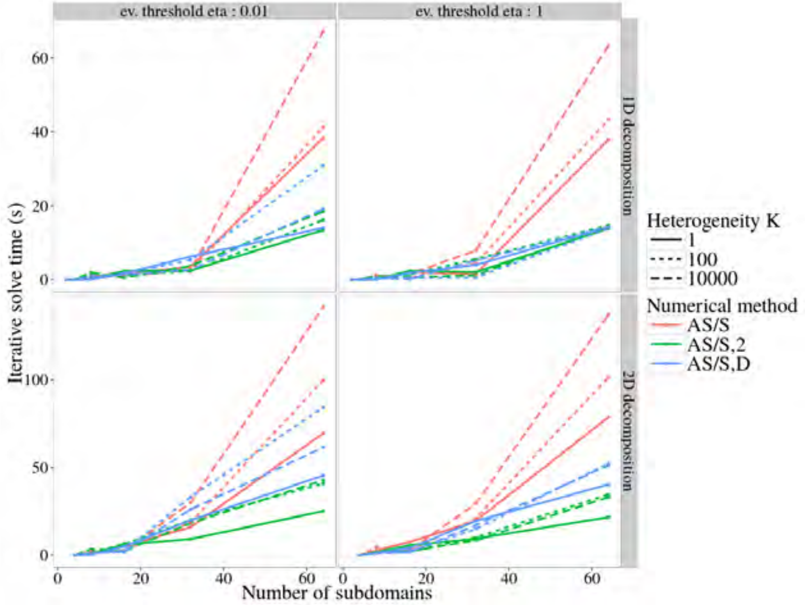
- The number of iterations is stabilized independently of K and N



- More difficult problems require a bigger coarse space



$$\kappa(\mathcal{M}_{AS/S,2}\mathcal{S}) \leq (1+N_c)\left(N_c+1+\frac{N_c+2}{\eta}\right) \quad \kappa(\mathcal{M}_{AS/S,D}\mathcal{S}) \leq N_c\left(1+\frac{1}{\eta}\right)$$


ev. threshold eta : 0.01
ev. threshold eta : 1
1D decomposition
2D decomposition
Iterative solve time (s)
Number of subdomains
0
20
40
60
0
50
100
0
20
40
60
Heterogeneity K
1
100
10000
Numerical method
AS/S
AS/S,2
AS/S,D

Perspectives

GenEO in MaPHyS

- Loosening the assumptions ($\mathcal{A}_i$ SPSD and $\mathcal{A}$ SPD)
- Implementation and test of the 2-level preconditioner on real applications

Other recent/ongoing efforts in MaPHyS

- Partioning/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

MPI Parallelism in MaPHyS

Factorization	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Preconditioner Setup	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Solve	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$

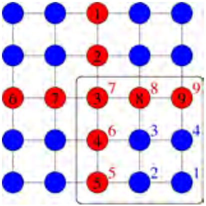
MPI + threads Parallelism in MaPHyS

Factorization	$\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$
Preconditioner Setup	$\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$
Solve	$\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$ $\downarrow\downarrow\downarrow$

Outline

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

Distributed Subdomain Interface [M. Kuhn]



Global data

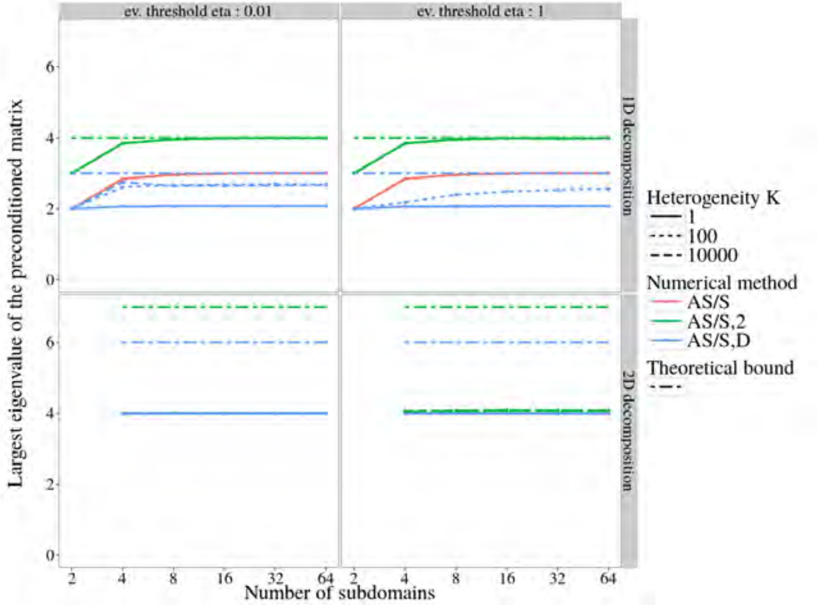
- `myndof`: number of degree of freedom
- `mysizeintrf`: number of interface nodes

Local data

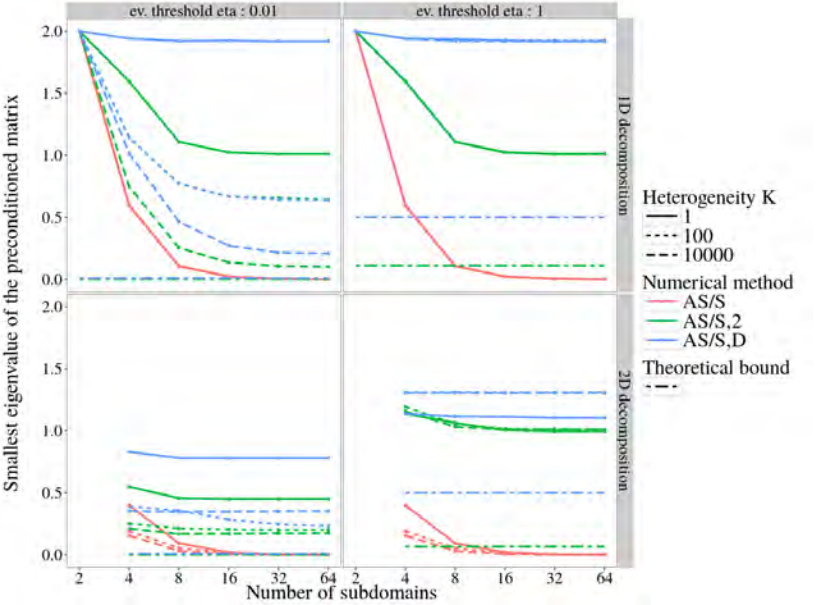
- $\mathcal{A}_i$, b_i
- `myinterface(:)`: interface node list in global ordering
- `mynbvi`: number of neighbor processes
- `myindexVi(:)`: list of neighbor processes (MPI ranks)
- `myptrindexVi(:)`: pointer to common interface nodes of neighbors
- `myindexintrf(:)`: common interface node list of neighbors

Outline

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



$\lambda_{\max}(\mathcal{M}_{AS/S,2}\mathcal{S}) \leq N_c + 1$ $\lambda_{\max}(\mathcal{M}_{AS/S,D}\mathcal{S}) \leq N_c$



$$\lambda_{\min}(\mathcal{M}_{AS/S,2}\mathcal{S}) \geq \frac{1}{N_c+1+\frac{N_c+2}{\eta}} \qquad \lambda_{\min}(\mathcal{M}_{AS/S,D}\mathcal{S}) \geq \frac{1}{1+\frac{1}{\eta}}$$