ANR Project DEDALES
Algebraic and Geometric Domain Decomposition for Subsurface Flow

Michel Kern

Inria Paris–Rocquencourt — Maison de la Simulation

C2S@Exa Days, Inria Paris Centre, Novembre 2016

Outline

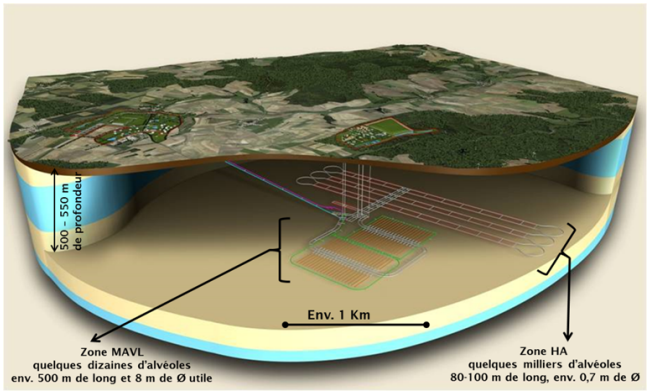
Outline

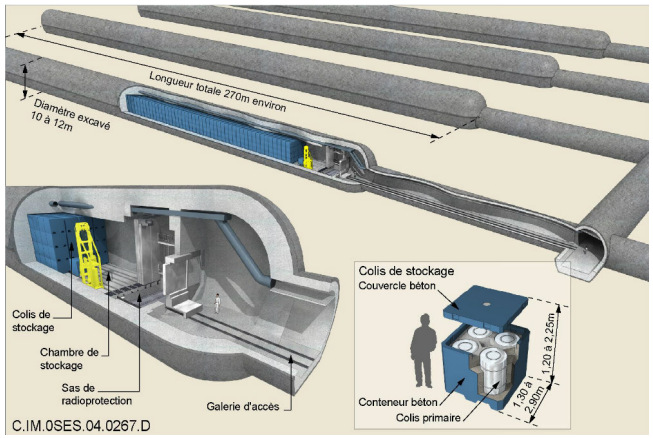
Application challenge

Develop HPC simulation software for the simulation of two phase flow in the subsurface

Target application

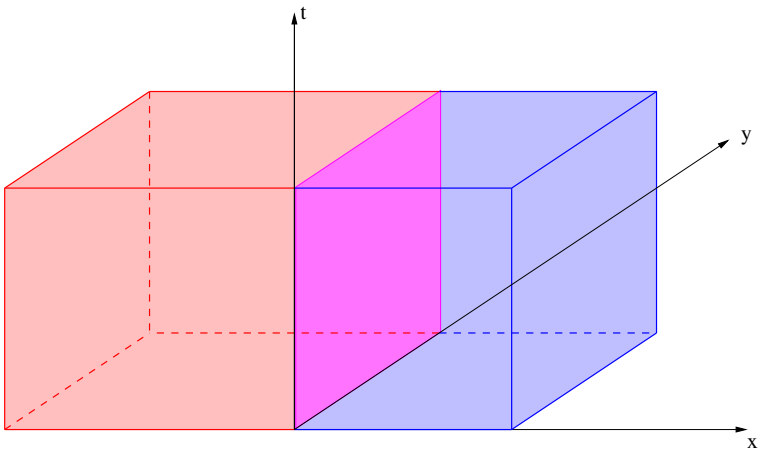
Water gas flow for nuclear waste storage

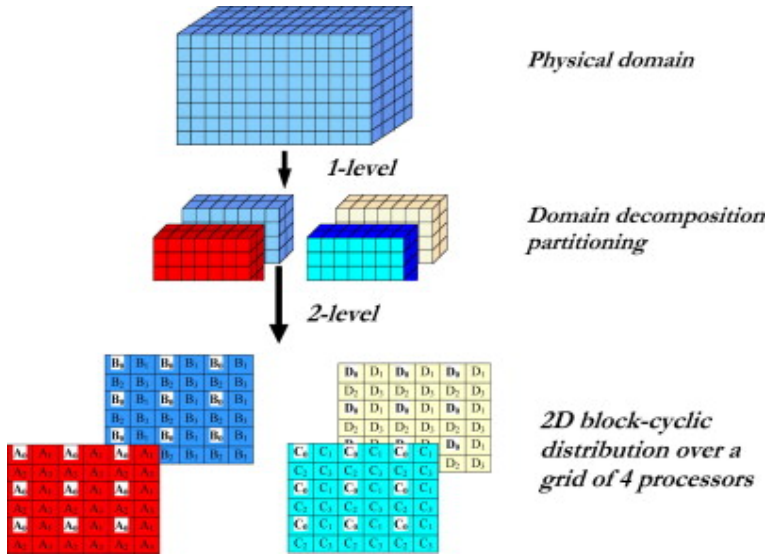




Scientific challenges

- Formulate space–time DD methods for two-phase flow (non-linearity, discontinuous capillary pressure) ;
- Develop a hybrid (MPI + threads) linear solvers, over a runtime system ;
- Show how Domain Decomposition methods can take advantage of upcoming hierarchical and heterogeneous architectures.





Consortium

- Serena, Inria Paris
- LAGA, Université Paris Nord, CNRS
- HiePACS, Inria Bordeaux Sud–Ouest
- Andra
- Maison de la Simulation (CEA, CNRS, Inria, UVSQ, UPS)

Skills in analyzing DD methods, simulation of flow in porous media, parallel linear algebra, high performance computing, waste storage simulation

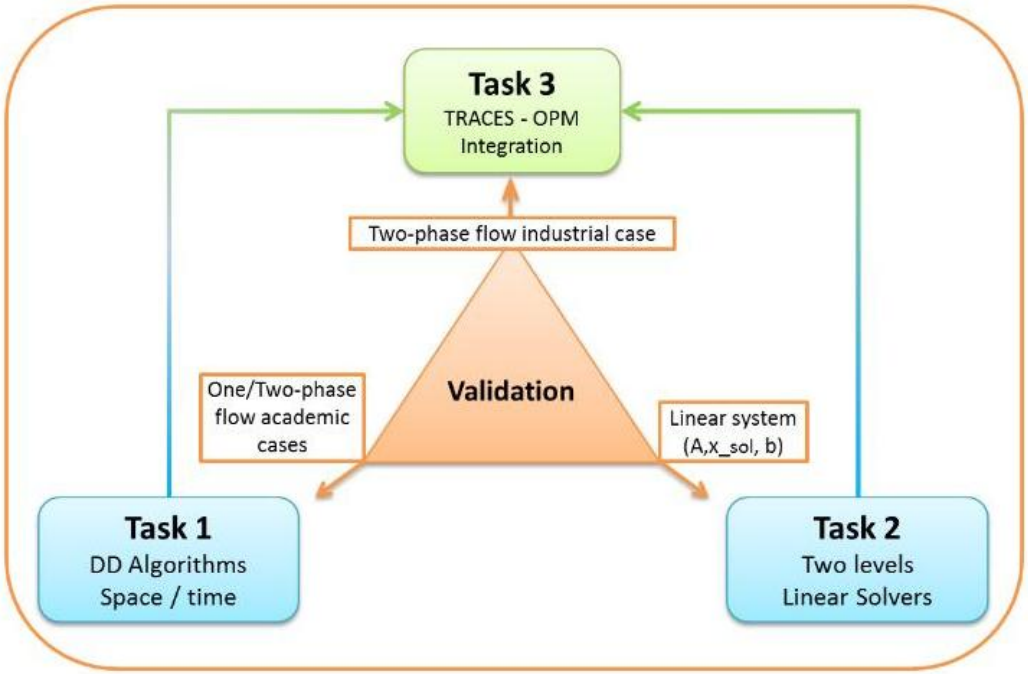
Project organization

Three development tasks, validation

Domain decomposition algorithms

Hybrid parallel linear solver

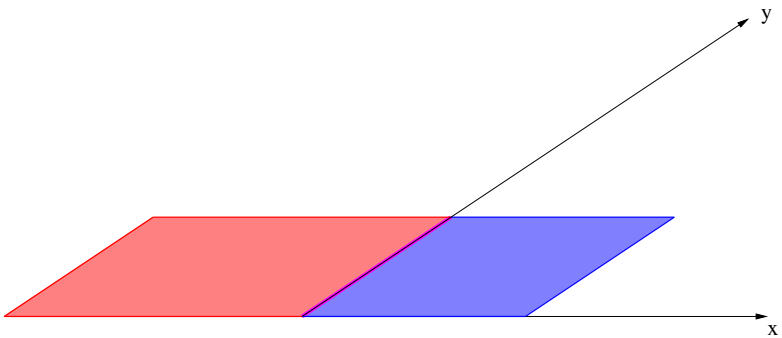
Integration on hierarchical architecture



Outline

Space–time domain decomposition

Domain decomposition in space



- Discretize in time and apply DD algorithm at each time step :
  - ► Solve stationary problems in the subdomains
  - ► Exchange information through the interface
- Use the same time step on the whole domain.

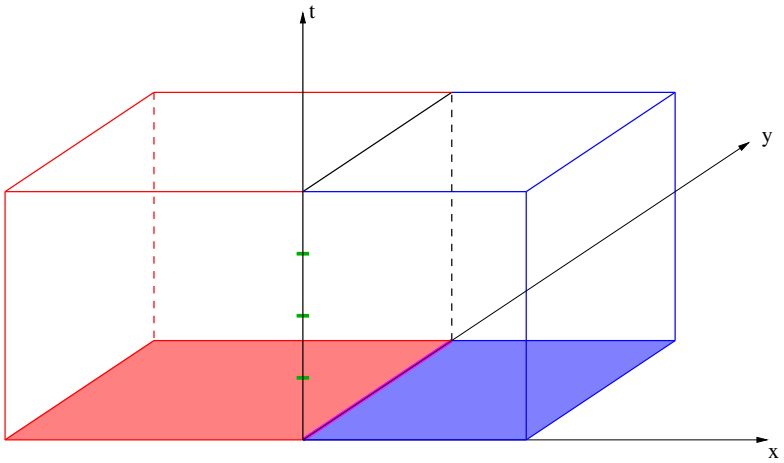
Space-time domain decomposition

- Solve time-dependent problems in the subdomains
- Exchange information through the space-time interface
- Enable local discretizations both in space and in time

  $\longrightarrow$ local time stepping

Space–time domain decomposition

Domain decomposition in space



- Discretize in time and apply DD algorithm at each time step :
  - ► Solve stationary problems in the subdomains
  - ► Exchange information through the interface
- Use the same time step on the whole domain.

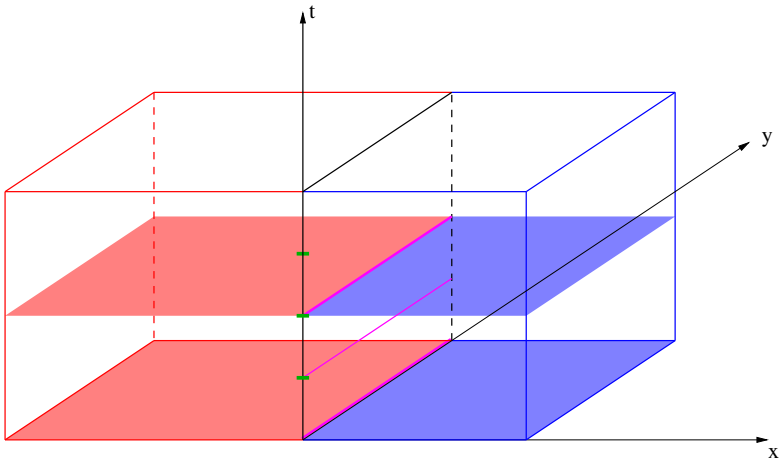
Space-time domain decomposition

- Solve time-dependent problems in the subdomains
- Exchange information through the space-time interface
- Enable local discretizations both in space and in time

  $\longrightarrow$ local time stepping

Space–time domain decomposition

Domain decomposition in space



- Discretize in time and apply DD algorithm at each time step :
  - ► Solve stationary problems in the subdomains
  - ► Exchange information through the interface
- Use the same time step on the whole domain.

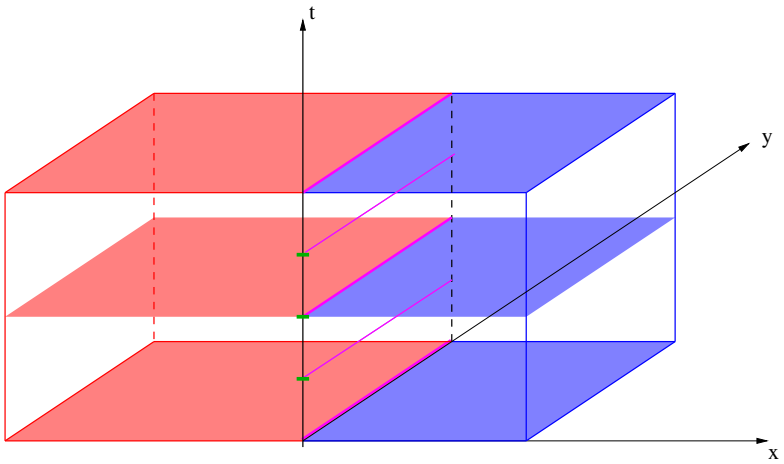
Space-time domain decomposition

- Solve time-dependent problems in the subdomains
- Exchange information through the space-time interface
- Enable local discretizations both in space and in time

  $\longrightarrow$ local time stepping

Space–time domain decomposition

Domain decomposition in space



- Discretize in time and apply DD algorithm at each time step :

  ► Solve stationary problems in the subdomains

  ► Exchange information through the interface

- Use the same time step on the whole domain.

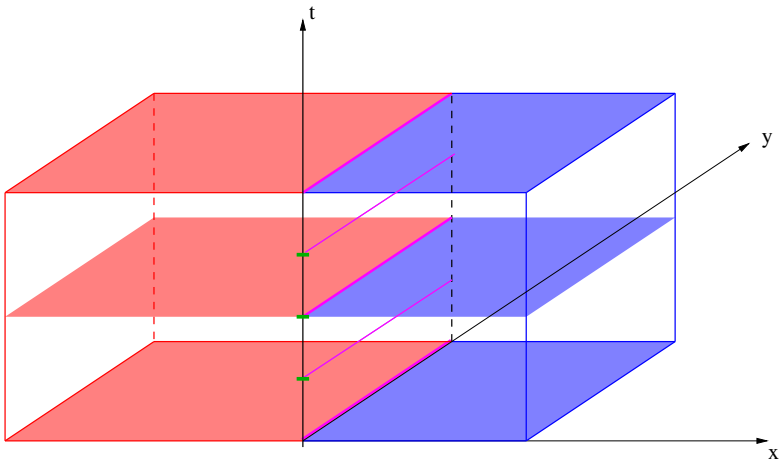
Space-time domain decomposition

- Solve time-dependent problems in the subdomains
- Exchange information through the space-time interface
- Enable local discretizations both in space and in time

  $\longrightarrow$ local time stepping

Space–time domain decomposition

Domain decomposition in space



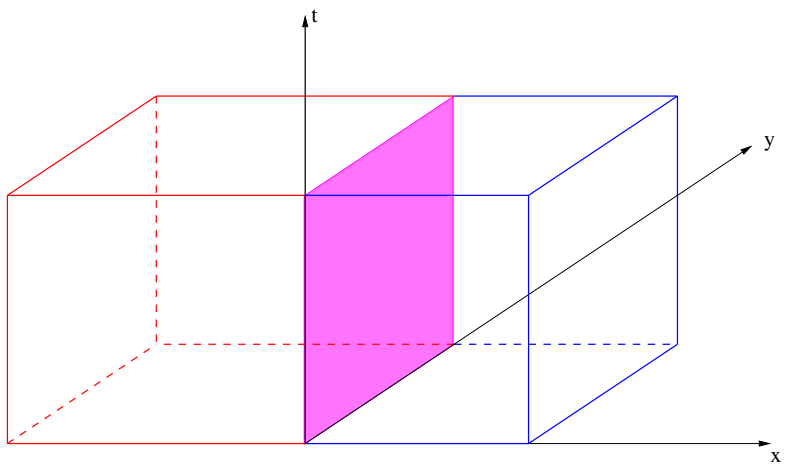
- Discretize in time and apply DD algorithm at each time step :

  ► Solve stationary problems in the subdomains

  ► Exchange information through the interface

- Use the same time step on the whole domain.

Space-time domain decomposition

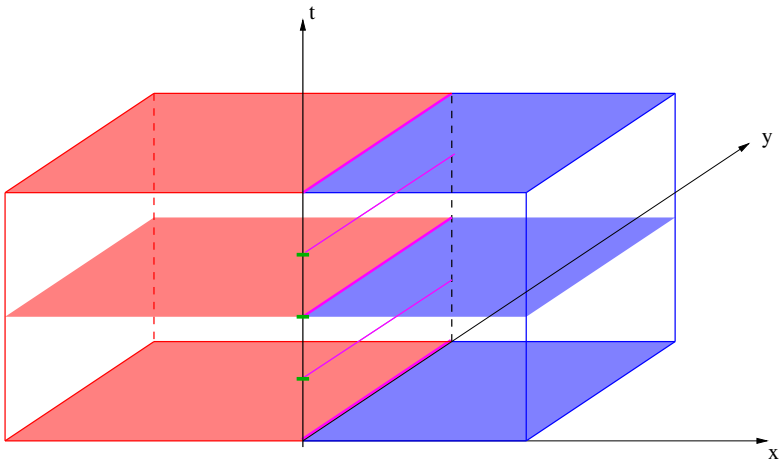


- Solve time-dependent problems in the subdomains
- Exchange information through the space-time interface
- Enable local discretizations both in space and in time

  $\longrightarrow$ local time stepping

Space–time domain decomposition

Domain decomposition in space



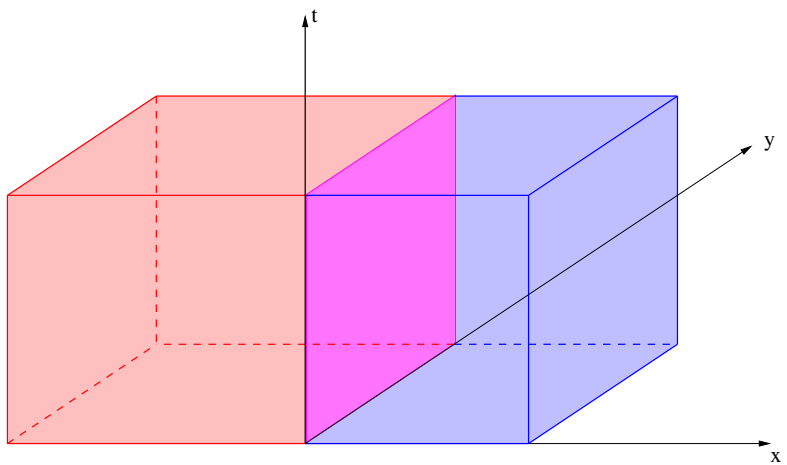
- Discretize in time and apply DD algorithm at each time step :

  ► Solve stationary problems in the subdomains

  ► Exchange information through the interface

- Use the same time step on the whole domain.

Space-time domain decomposition



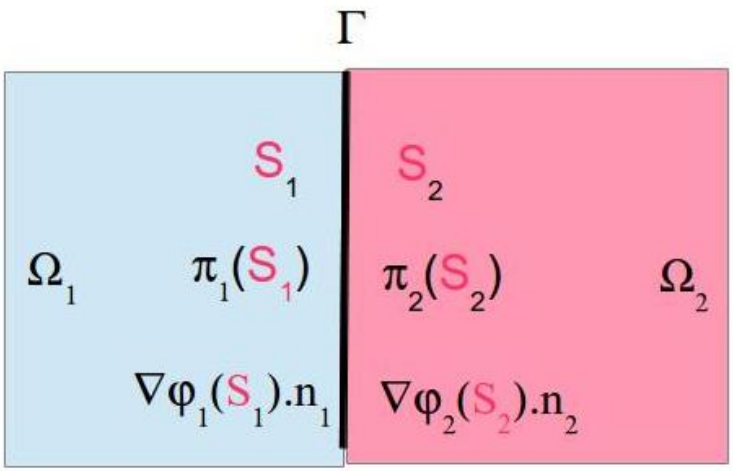
- Solve time-dependent problems in the subdomains
- Exchange information through the space-time interface
- Enable local discretizations both in space and in time

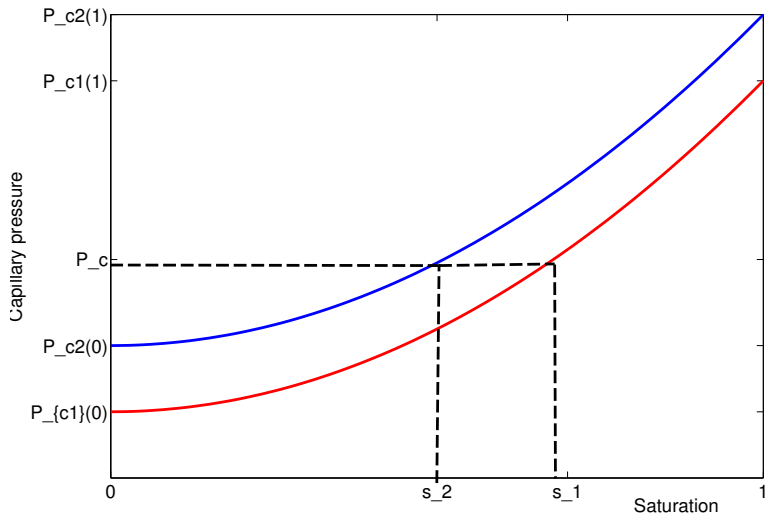
  $\longrightarrow$ local time stepping

Simplified two–phase flow with discontinuous capillary pressure

Simplified non-linear degenerate diffusion model

$$\omega \partial_t S - \Delta \phi(S) = 0 \quad \text{in } \Omega \times [0, T]$$





Transmission conditions on the interface

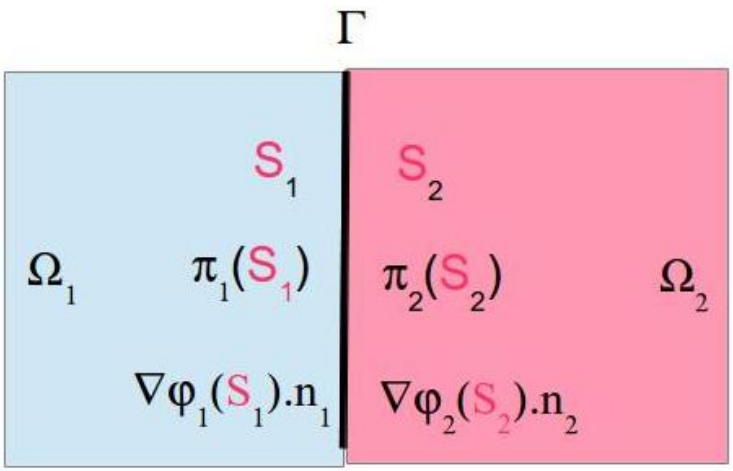
Continuity of capillary pressure $\pi_1(S_1) = \pi_2(S_2)$ on Γ

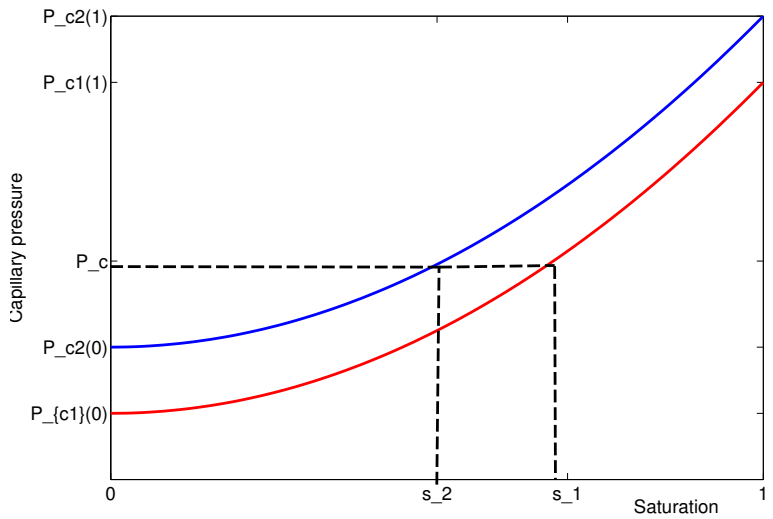
Continuity of the flux $\nabla \phi_1(S_1).n_1 = -\nabla \phi_2(S_2).n_2$ on Γ

Simplified two–phase flow with discontinuous capillary pressure

Simplified non-linear degenerate diffusion model

$$\omega \partial_t S - \Delta \phi(S) = 0 \quad \text{in } \Omega \times [0, T]$$





Equivalent transmission conditions

- $\nabla \phi_1(S_1).n_1 + \beta_1 \pi_1(S_1) = -\nabla \phi_2(S_2).n_2 + \beta_1 \pi_2(S_2)$
- $\nabla \phi_2(S_2).n_2 + \beta_2 \pi_2(S_2) = -\nabla \phi_1(S_1).n_1 + \beta_2 \pi_1(S_1)$

Non-linear Schwarz algorithm

Schwarz algorithm

Given S_i^0, iterate for $k = 0, \ldots$
Solve for S_i^{k+1}, $i = 1, 2, j = 3 - i$

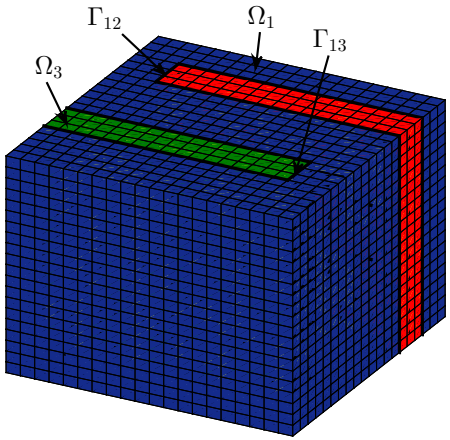
$$\omega \partial_t S_i^{k+1} - \Delta \phi_i(S_i^{k+1}) = 0 \qquad \text{in } \Omega_i \times [0, T]$$
$$\nabla \phi_i(S_i^{k+1}).n_i + \beta_i \pi_i(S_i^{k+1}) = -\nabla \phi_j(S_j^k).n_j + \beta_i \pi_j(S_j^k) \qquad \text{on } \Gamma \times [0, T],$$

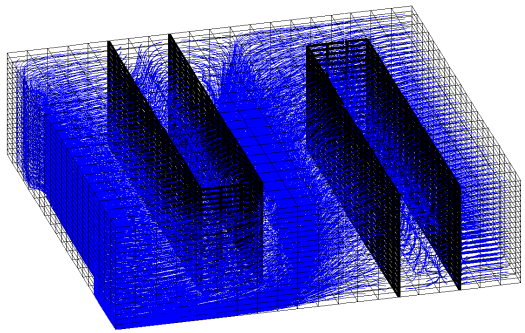
(β_1, β_2) are free parameters chosen to accelerate convergence

- Basic ingredient : subdomain solver with Robin bc.
- Discretization : extension to Robin bc of cell centered FV scheme by Enchéry, Eymard, Michel (2006).
- Implemented with Matlab Reservoir Simulation Toolbox (K. A. Lie et al. (14))

E. Ahmed, C. Japhet, M. Kern (in preparation)

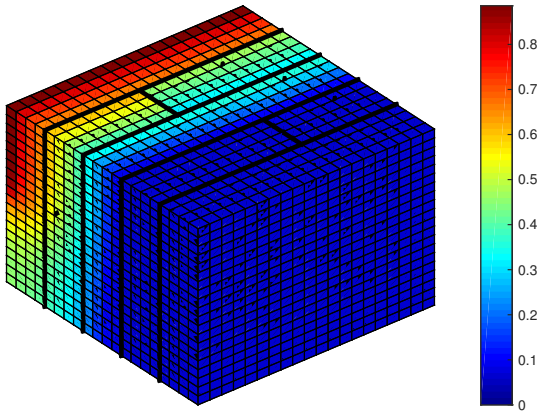
Example : a model with three rock type

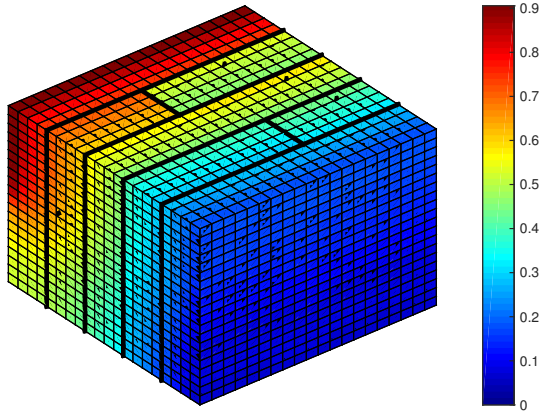




Geometry

Streamlines





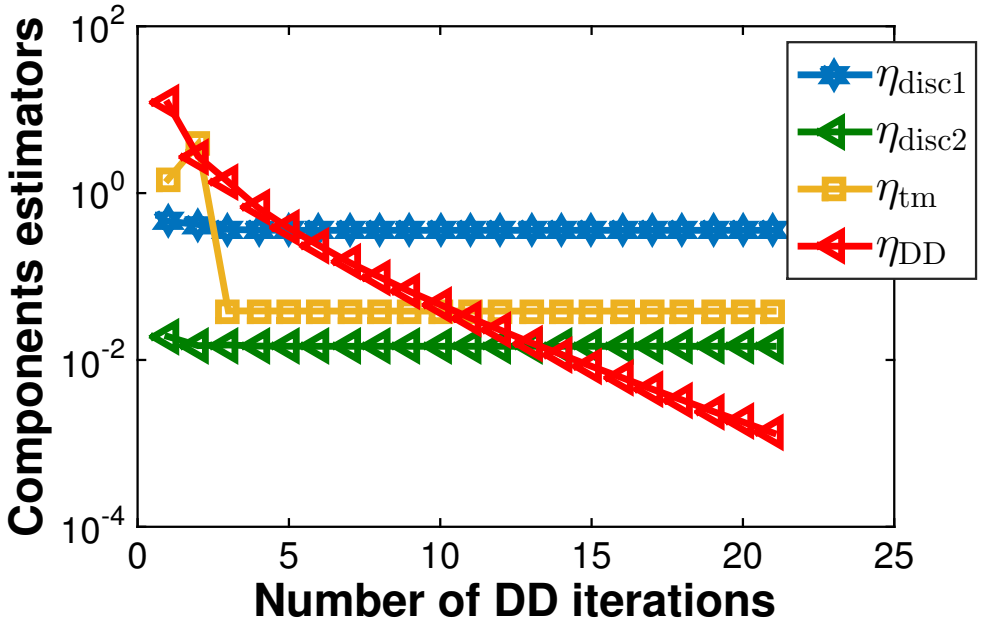
Saturation t=5000, t=2000

Stopping criteria through a posteriroi error estimates (Cemracs 2016)

Goal

Stop DD iterations as soon as discretization error is reached

Develop fully computable error estimator with guaranteed bound (no implicit constant), based on potential and flux reconstruction.
Allows separation of space, time, and iteration errors (S. Ali Hassan's PhD, M. Vohralìk, C. Japhet)



Outline

Motivation : Coarse Correction for MaPHyS

Need for Coarse Correction

- Good scalability of the direct part ☺
- The size and condition number of the iterative problem increases with the number of subdomains ☹

A proved robust coarse space for a larger class of methods

- Generalized Abstract Schwarz (GAS) methods
  - Neumann-Neumann, Additive Schwarz, Additive Schwarz on the Schur (*MaPHyS*), ...
- Only works in the SPD case, with distributed input

Two implementations

- A python prototype, providing a framework for distributed GAS methods
- (partially) integrated in MaPHyS0.9.4

Two level precondtioner : aS Step by step

Step 1 : Domain Decomposition

- $\mathscr{A} = \sum_{i=1}^{N} \mathbf{R}_i^T \mathscr{A}_i \mathbf{R}_i$

Step 2 : Factorization

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

Step 3 : Preconditioner Setup

- $\mathscr{M}_{aS} = \sum_{i=1}^{N} \mathbf{R}_{\Gamma_i}^T \left(\mathbf{R}_{\Gamma_i} \mathscr{S} \mathbf{R}_{\Gamma_i}^T\right)^{-1} \mathbf{R}_{\Gamma_i}$

Step 4 : Solve

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

Two level precondtioner : aS, 2 Step by step

Step 1 : Domain Decomposition (Application level)

- $\mathscr{A} = \sum_{i=1}^{N} \mathbf{R}_i^T \mathscr{A}_i \mathbf{R}_i$

Step 2 : Factorization

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

Step 3 : Preconditioner Setup

- $\mathscr{M}_{aS,2} = \mathscr{M}_0 + \sum_{i=1}^{N} \mathbf{R}_{\Gamma_i}^T \left(\mathbf{R}_{\Gamma_i} \mathscr{S} \mathbf{R}_{\Gamma_i}^T\right)^{-1} \mathbf{R}_{\Gamma_i}$

Step 4 : Solve

- on Γ : *Krylov method* $\quad \mathscr{S} x_\Gamma = f \quad$ preconditioned with $\mathscr{M}_{aS,2}$
- on $\mathscr{I}$: *Direct method* $\quad x_{\mathscr{I}_i} = \mathscr{A}_{\mathscr{I}_i\mathscr{I}_i}^{-1}\left(b_{\mathscr{I}_i} - \mathscr{A}_{\mathscr{I}_i\Gamma_i} x_{\Gamma_i}\right)$

Coarse space for GAS

2-level method needed to keep the number of iterations independant of # cores.

Two-level abstract Schwarz

Coarse space	V_0
Coarse solve	$\mathscr{M}_0 = V_0 (V_0^T \mathscr{S} V_0)^\dagger V_0^T$
Proj. onto coarse space	$\mathscr{P}_0 = \mathscr{M}_0 \mathscr{S}$

Two-level AS : $\mathscr{M}_D = \mathscr{M}_0 + (I - \mathscr{P}_0)\mathscr{M}_1(I - \mathscr{P}_0)$, $\mathscr{M}_1$ 1 level preconditioner
Generalized AS : replace $\mathscr{S}$ by approximation $\hat{\mathscr{S}}$.

Extend GENEO (Spillane at al.) to GAS

1. Solve locally generalized eigenvalue problems for λ and η above threshold α and β

$$\hat{\mathscr{S}}_i p = \lambda \tilde{\mathscr{S}}_i^{\mathrm{NN}} p \quad \text{and} \quad \tilde{\mathscr{S}}_i^{\mathrm{AS}} p = \eta \hat{\mathscr{S}}_i p$$

2. Assemble resulting coarse space : $V_0 = \sum_{i=1}^N \mathscr{R}_\Gamma^T V_i^0$

Condition number bounded independently of N and coefficients

3D Test problem

Heterogeneous diffusion

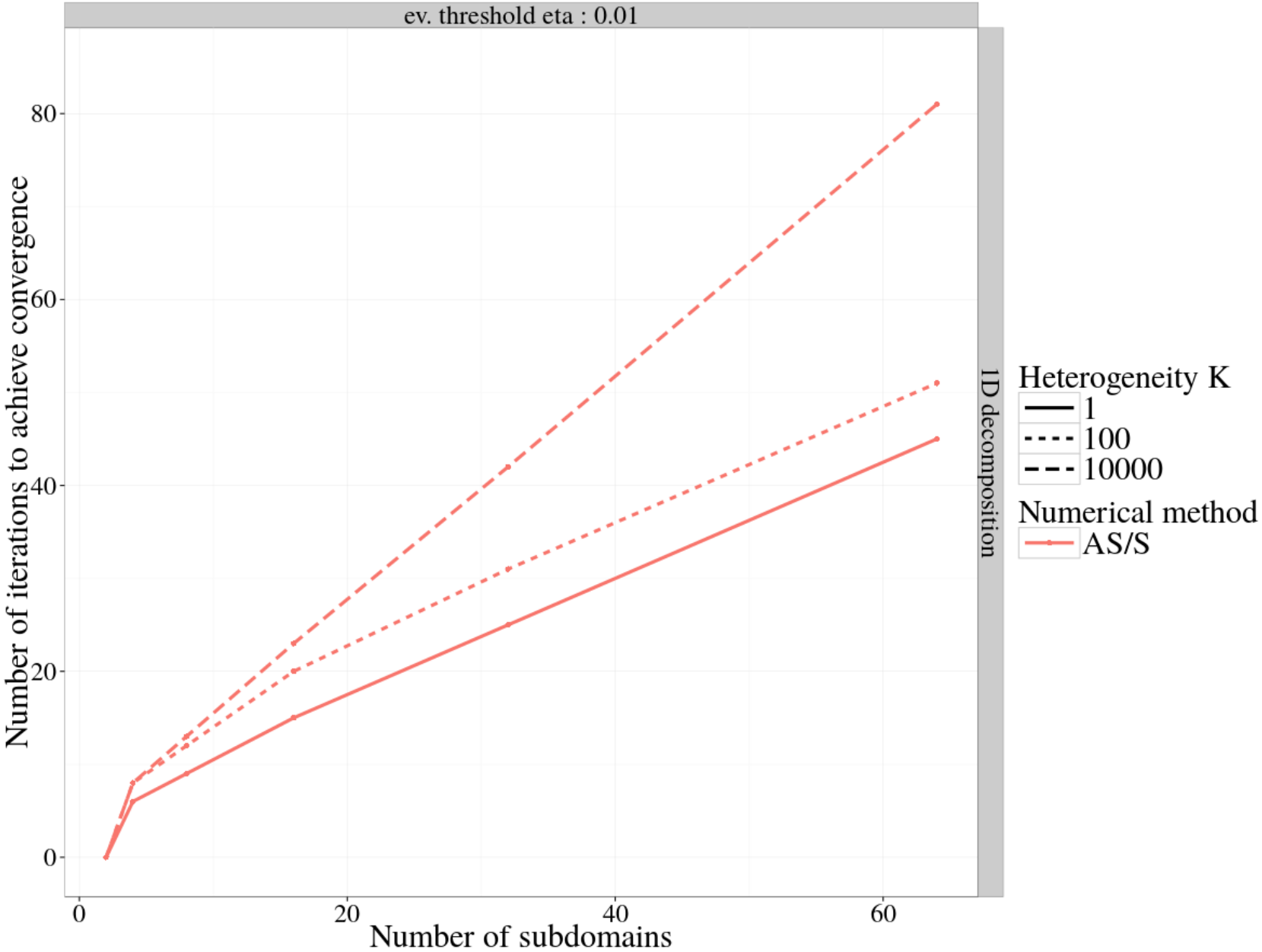
- $\nabla \cdot (K \nabla u) = 1$
- Alternating conductivity layers of 3 elements (ratio K between layers)
- Dirichlet on the left, Neumann elsewhere

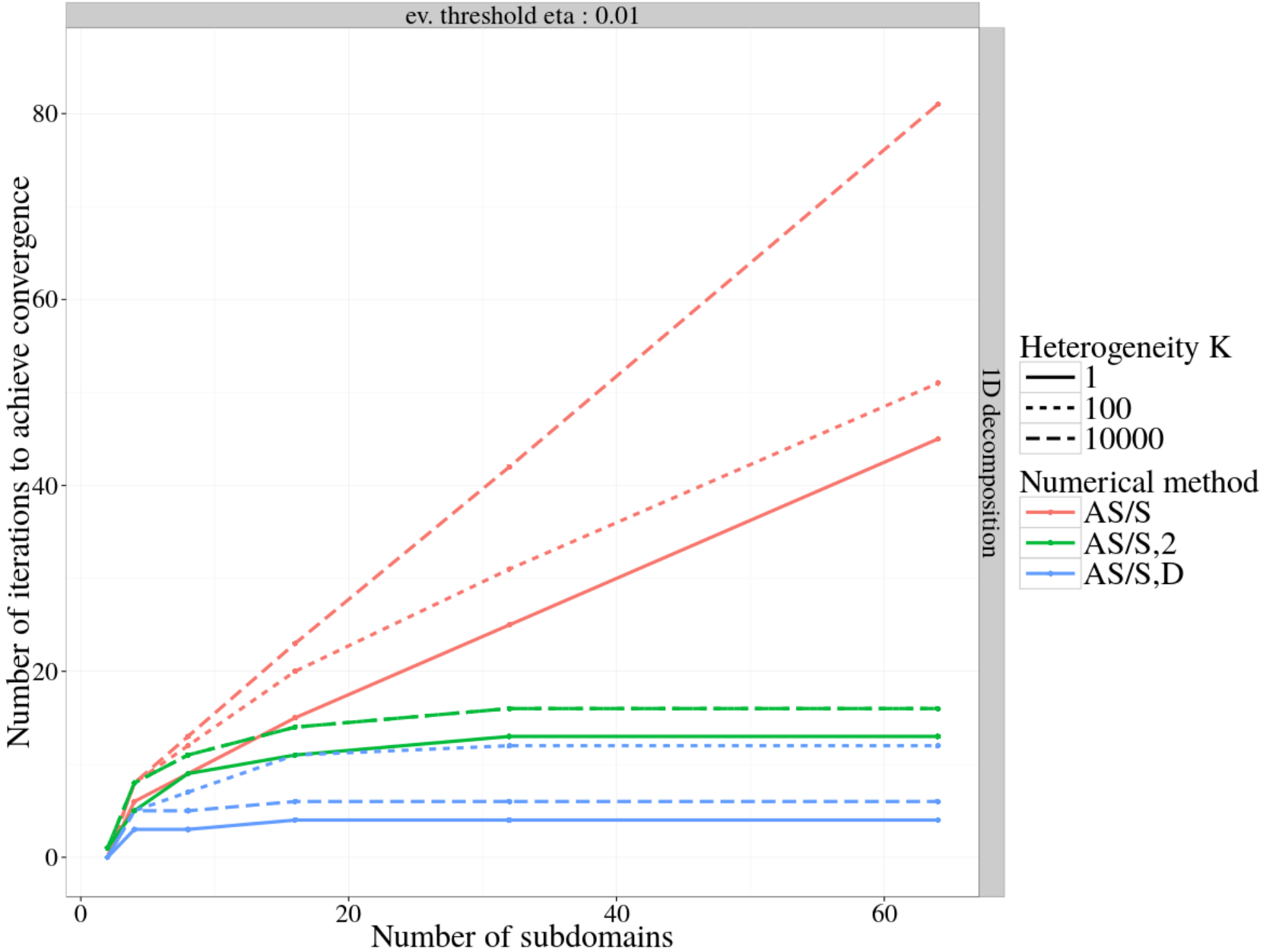
Domain decomposition

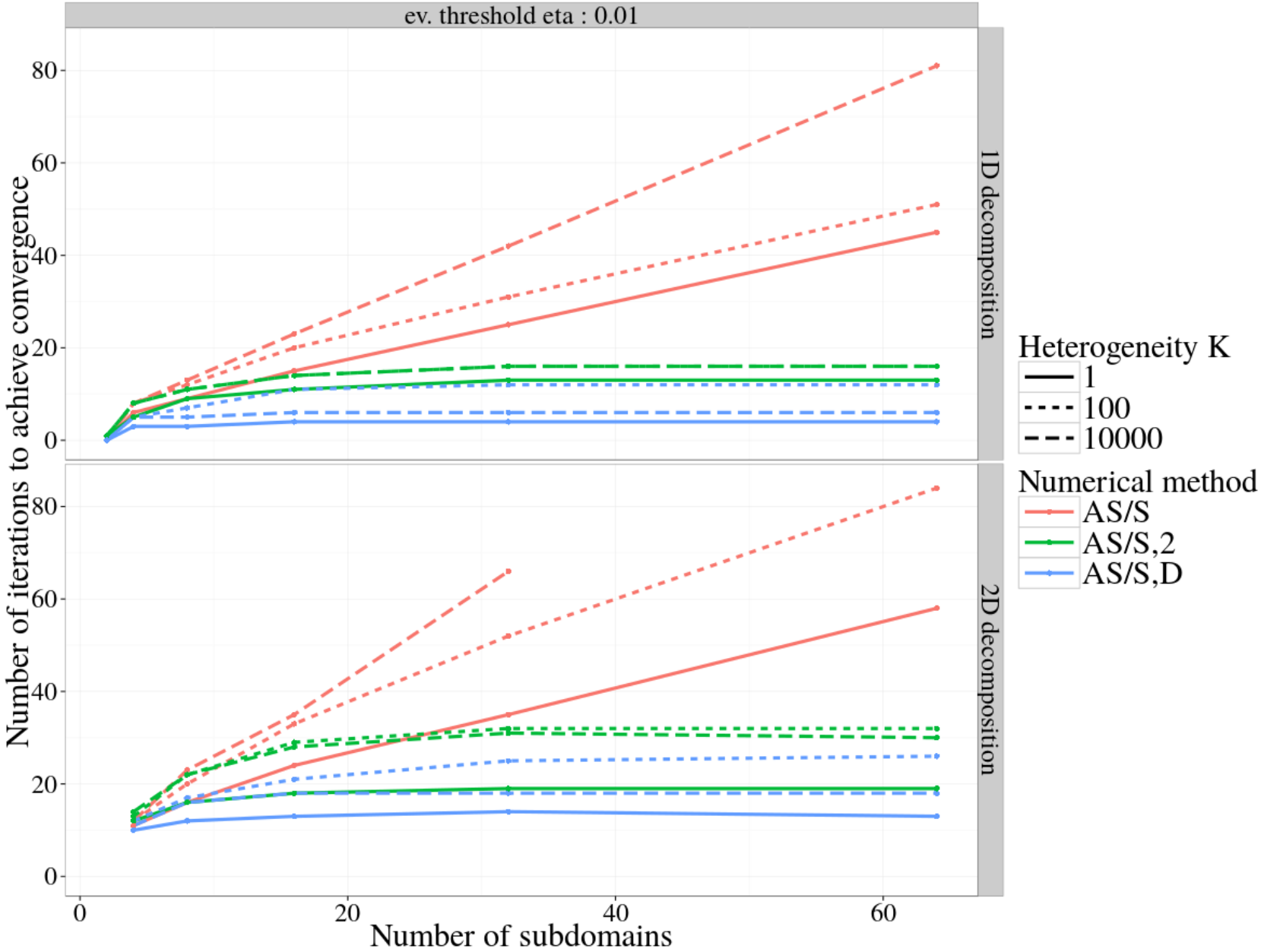
- $N \times 1 \times 1$ (1D decomposition)
- $N/2 \times 2 \times 1$ (2D decomposition)
- Constant subdomain size : $10 \times 10 \times 10$ elements

Implementation

- python/MPI


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


ev. threshold eta : 0.01
1D decomposition
Number of iterations to achieve convergence
0
20
40
60
80
Number of subdomains
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

Outline

Andra's role in the project

Stake

Improving the physical and geometrical representation of numerical simulations on water–gas flow on the various storage scales

Andra's role :

- Integrate methods (MaPHyS) into TRACES software (simulation of saturated flow and transport)
- Validation through test cases

Three test cases

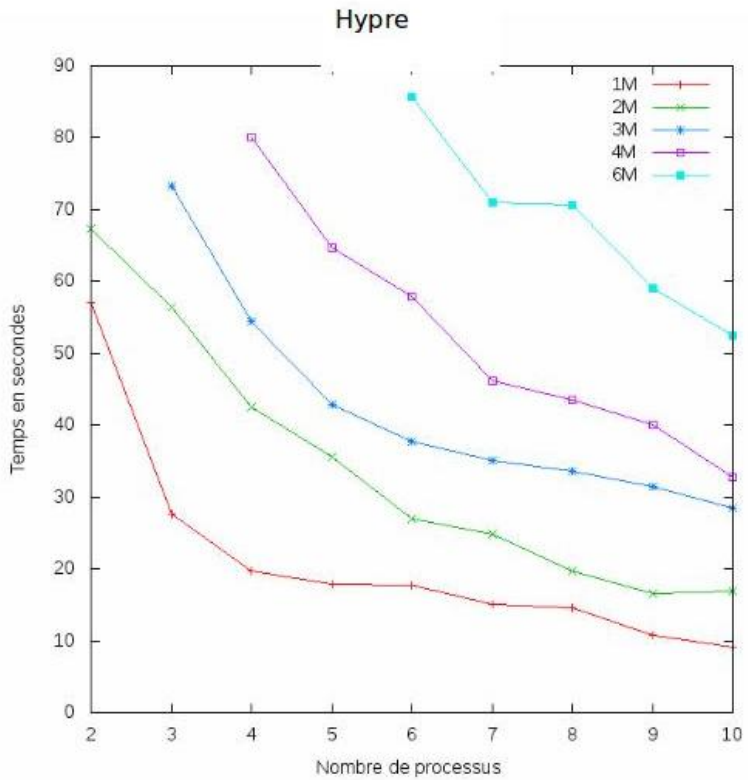
- Academic two-phase, non compositional, but heterogeneous (for DD algs)
- Matrices extracted from Tough-MP for testing linear solvers
- Full desaturation and resaturation case for final software

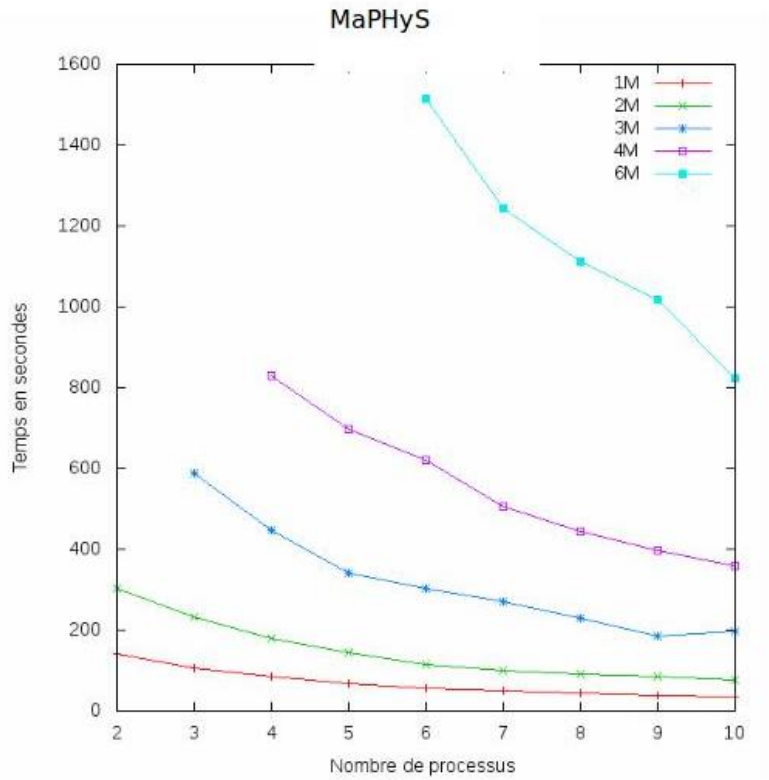
See B. Vialay's (Tuesday) and M. Leconte's talks (Wednesday) for results

Improvements to Traces due to MaPHyS

First comparison between TRACES/Hypre and TRACES/MaPHyS

- Test case from 1 to 6 million elements (up to 18M DOFs)
- Flow test case : GMRES/AMG solver, 1 subdomain per node



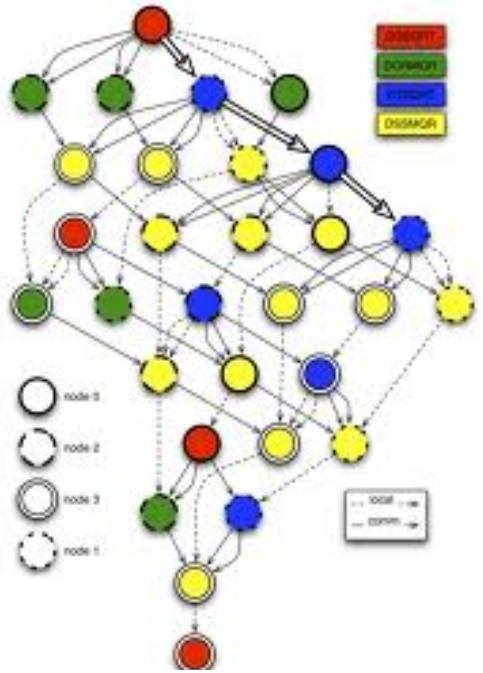


Work in progress

Algebraic DD

Hybrid linear solver on top of runtime system

- Iterative (DD) method on interfaces, direct method in subdomains
- Heterogeneous nodes : runtime system (StarPU)



Geometric DD

Two level parallel solver : space-time DD methods on “large” sudomains, parallel (MPI) solver in subdomains