Robust, algebraic, and scalable Schwarz preconditioners with extension-based coarse spaces

Alexander Heinlein

27th International Domain Decomposition Conference, Prague, Czech Republic, July 25-29, 2022

Delft University of Technology

Linear & Nonlinear Preconditioning

Let us consider the nonlinear problem arising from the discretization of a partial differential equation

$$\boldsymbol{F}(\boldsymbol{u}) = 0.$$

We solve the problem using a **Newton-Krylov approach**, i.e., we solve a sequence of linearized problems using a Krylov subspace method:

$$\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right) \Delta \boldsymbol{u}^{(k+1)} = \boldsymbol{F}\left(\boldsymbol{u}^{(k)}\right).$$

Linear preconditioning

In linear preconditioning, we **improve the convergence speed of the linear solver** by constructing a **linear operator** M^{-1} and solve linear systems

$$\boldsymbol{M}^{-1}\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right) \Delta \boldsymbol{u}^{(k+1)} = \boldsymbol{M}^{-1}\boldsymbol{F}(\boldsymbol{u}^{(k)}).$$

Goal:

- $\kappa\left(\boldsymbol{M}^{-1}\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right)\right) \approx 1.$

$\Rightarrow \boldsymbol{M}^{-1}\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right) \approx \boldsymbol{I}.$

Nonlinear preconditioning

In nonlinear preconditioning, we **improve the convergence speed of the nonlinear solver** by constructing a **nonlinear operator** G and solve the nonlinear system

$$(\boldsymbol{G} \circ \boldsymbol{F})(\boldsymbol{u}) = 0.$$

Goals:

- $\boldsymbol{G} \circ \boldsymbol{F}$ almost linear.
- Additionally: $\kappa(\boldsymbol{D}(\boldsymbol{G} \circ \boldsymbol{F})(\boldsymbol{u})) \approx 1.$

Linear & Nonlinear Preconditioning

Let us consider the nonlinear problem arising from the discretization of a partial differential equation

$$\boldsymbol{F}(\boldsymbol{u}) = 0.$$

We solve the problem using a **Newton-Krylov approach**, i.e., we solve a sequence of linearized problems using a Krylov subspace method:

$$\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right)\Delta\boldsymbol{u}^{(k+1)} = \boldsymbol{F}\left(\boldsymbol{u}^{(k)}\right).$$

Linear preconditioning

In linear preconditioning, we **improve the convergence speed of the linear solver** by constructing a **linear operator** M^{-1} and solve linear systems

$$\boldsymbol{M}^{-1}\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right)\Delta\boldsymbol{u}^{(k+1)} = \boldsymbol{M}^{-1}\boldsymbol{F}(\boldsymbol{u}^{(k)}).$$

Goal:

- $\kappa\left(\boldsymbol{M}^{-1}\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right)\right) \approx 1.$

$\Rightarrow$ $\boldsymbol{M}^{-1}\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right) \approx \boldsymbol{I}.$

Nonlinear preconditioning

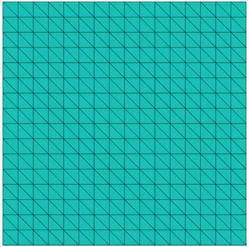
In nonlinear preconditioning, we **improve the convergence speed of the nonlinear solver** by constructing a **nonlinear operator** G and solve the nonlinear system

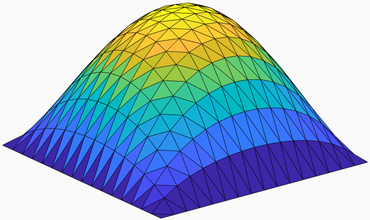
$$(\boldsymbol{G} \circ \boldsymbol{F})(\boldsymbol{u}) = 0.$$

Goals:

- $\boldsymbol{G} \circ \boldsymbol{F}$ almost linear.
- Additionally: $\kappa\left(\boldsymbol{D}(\boldsymbol{G} \circ \boldsymbol{F})(\boldsymbol{u})\right) \approx 1.$

Simple Model Problem





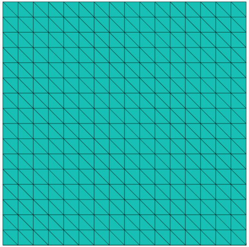
Consider a **homogeneous diffusion model problem**:

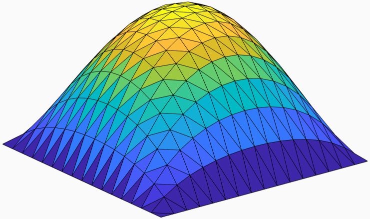
$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega = [0,1]^2, \\ u &= 0 \quad \text{on } \partial\Omega. \end{aligned}$$

Discretization using finite elements yields the linear equation system

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}.$$

Simple Model Problem





Consider a **homogeneous diffusion model problem**:

$$
\begin{aligned}
-\Delta u &= f \quad \text{in } \Omega = [0,1]^2, \\
u &= 0 \quad \text{on } \partial\Omega.
\end{aligned}
$$

Discretization using finite elements yields the linear equation system

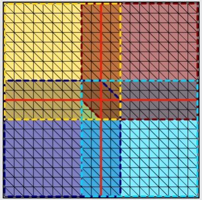
$$
\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}.
$$

$\Rightarrow$ Construct a preconditioner using **overlapping Schwarz domain decomposition methods**.

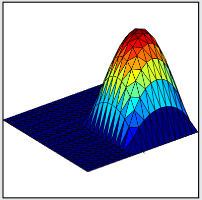
Two-Level Schwarz Preconditioners

One-level Schwarz preconditioner

Overlap $\delta = 1h$



Solution of local problem



Based on an **overlapping domain decomposition**, we define a **one-level Schwarz operator**

$$M_{\text{OS-1}}^{-1} K = \sum_{i=1}^{N} R_i^T K_i^{-1} R_i K,$$

where R_i and R_i^T are restriction and prolongation operators corresponding to Ω_i', and $K_i := R_i K R_i^T$.

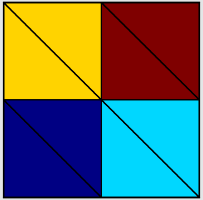
Condition number estimate:

$$\kappa\left(M_{\text{OS-1}}^{-1} K\right) \leq C \left(1 + \frac{1}{H\delta}\right)$$

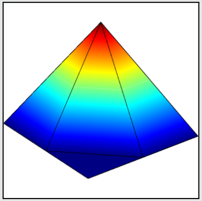
with subdomain size H and overlap width δ.

Adding a Lagrangian coarse space

Coarse triangulation



Coarse solution



The **two-level overlapping Schwarz operator** reads

$$M_{\text{OS-2}}^{-1} K = \underbrace{\Phi K_0^{-1} \Phi^T K}_{\text{coarse level – global}} + \underbrace{\sum_{i=1}^{N} R_i^T K_i^{-1} R_i K}_{\text{first level – local}},$$

where Φ contains the coarse basis functions and $K_0 := \Phi^T K \Phi$; cf., e.g., **Toselli, Widlund (2005)**.

The construction of a Lagrangian coarse basis requires a coarse triangulation.

Condition number estimate:

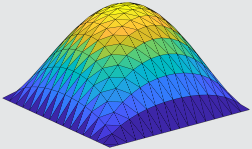
$$\kappa\left(M_{\text{OS-2}}^{-1} K\right) \leq C \left(1 + \frac{H}{\delta}\right)$$

Strengths and Weaknesses of Classical Two-Level Schwarz Preconditioners

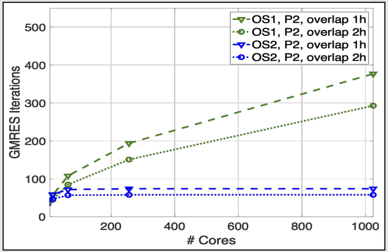
Numerical scalability

Diffusion with **heterogeneous coefficient**:

$$-\Delta u = f \quad \text{in } \Omega = [0,1]^2,$$
$$u = 0 \quad \text{on } \partial\Omega.$$

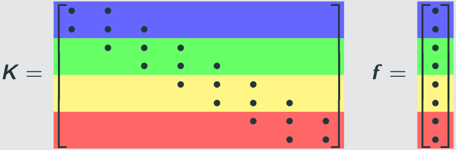


$\#$ subdomains $= \#$ cores, $H/h = 100$

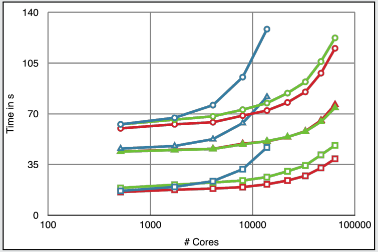


Algebraic construction

Requires coarse triangulation (geometric information). No construction based on:



Parallel scalability

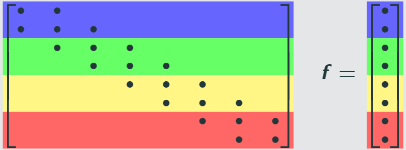


Strengths and Weaknesses of Classical Two-Level Schwarz Preconditioners

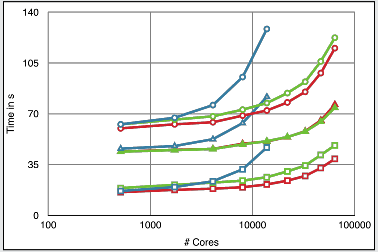
Algebraic construction

Requires coarse triangulation (geometric information). No construction based on:

$\boldsymbol{K} =$ $\boldsymbol{f} =$



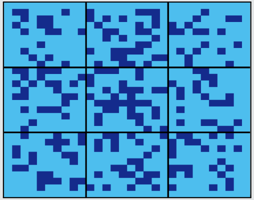
Parallel scalability



Robustness

Diffusion with **heterogeneous coefficient**:

$$-\nabla \cdot (\alpha(x)\nabla u(x)) = f(x) \quad \text{in } \Omega = [0,1]^2,$$
$$u = 0 \quad \text{on } \partial\Omega.$$



dark blue: $\alpha = 10^8$ **light blue:** $\alpha = 1$

10×10 subdomains with $H/h = 10$ and overlap $1h$

Prec.	its.	κ
–	**>2 000**	$4.51 \cdot 10^8$
$M_{OS\text{-}1}^{-1}$	**>2 000**	$4.51 \cdot 10^8$
$M_{OS\text{-}2}^{-1}$	**586**	$5.56 \cdot 10^5$

Outline

1. Extension-Based Coarse Spaces

2. Parallel Implementation in FROSch

   Based on joint work with Christian Hochmuth, Axel Klawonn (University of Cologne), Oliver Rheinbach, Friederike Röver (TU Bergakademie Freiberg), Mauro Perego, Siva Rajamanickam, Ichitaro Yamazaki (Sandia), Olof Widlund (New York University)

3. Adaptive Extension-Based Coarse Spaces

   Based on joint work with Axel Klawonn, Jascha Knepper (University of Cologne), Oliver Rheinbach (TU Bergakademie Freiberg), Olof Widlund (New York University), Kathrin Smetana (Stevens Institute of Technology)

4. Extension-Based Coarse Spaces in Nonlinear Schwarz Preconditioning

   Based on joint work with Axel Klawonn, Martin Lanser (University of Cologne)

Extension-Based Coarse Spaces

Energy-Minimizing Extensions

The **energy-minimizing extension** $v_i = E_{\partial\Omega_i \to \Omega_i}(v_{\partial\Omega_i})$ solves

$$v_i = \underset{v|_{\partial\Omega} = v_{\partial\Omega}}{\arg\min}\ a_\Omega(v, v) \iff \begin{array}{rcll} a_{\Omega_i}(v_i, w_i) & = & 0 & \forall w_i \in V^0_{\Omega_i}, \\ v_i & = & v_{\partial\Omega_i} & \text{on } \partial\Omega_i. \end{array}$$

$\rightarrow$ **Energy-minimizing extensions** and functions with **homogeneous Dirichlet boundary conditions** are *a*-**orthogonal**.

In **matrix form**, this corresponds to

$$\boldsymbol{v} = \begin{pmatrix} -\boldsymbol{K}_{II}^{-1} \boldsymbol{K}_{I\Gamma} \\ \boldsymbol{I}_\Gamma \end{pmatrix} \boldsymbol{v}_\Gamma,$$

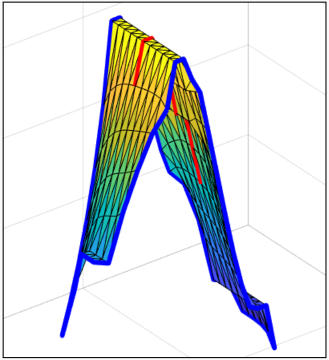
where we make use of the splitting of the rows and columns corresponding to interior (I) and interface (Γ) nodes

$$\boldsymbol{K} = \begin{pmatrix} \boldsymbol{K}_{II} & \boldsymbol{K}_{I\Gamma} \\ \boldsymbol{K}_{\Gamma I} & \boldsymbol{K}_{\Gamma\Gamma} \end{pmatrix}$$

See, e.g., Section 4.4 in the book **Toselli, Wildund (2005)**.

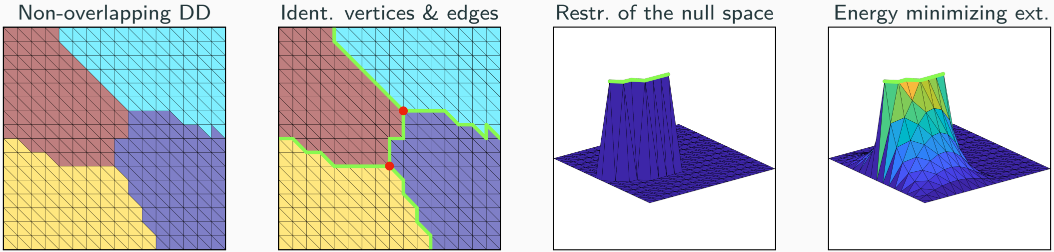
Diffusion model problem

$$a_\Omega(u, v) = \int_\Omega \alpha(x) \nabla u \cdot \nabla v \, dx$$



Two-Level Schwarz Preconditioners – GDSW Coarse Space

The following construction will lead to the **GDSW (Generalized–Dryja–Smith–Widlund) coarse space** introduced in Dohrmann, Klawonn, Widlund (2008).



The coarse interpolation is exact in the vertices, and the **energy of the edge functions** can be bounded as follows:

$$\|\theta_{\mathcal{E}}\|^2_{H^1(\Omega_i)} \leq C\left(1+\log\left(\frac{H}{h}\right)\right);$$

in **three dimensions**, **face basis functions** are added to the coarse space.

The **condition number of the GDSW two-level Schwarz operator** is bounded by

$$\kappa\left(\boldsymbol{M}_{\mathrm{GDSW}}^{-1}\boldsymbol{K}\right) \leq C\left(1+\frac{H}{\delta}\right)\left(1+\log\left(\frac{H}{h}\right)\right)^2;$$

cf. Dohrmann, Klawonn, Widlund (2008), Dohrmann, Widlund (2009, 2010, 2012).

Partition of Unity

The energy-minimizing extension $v_i = H_{\partial\Omega_i \to \Omega_i}(\mathbb{1})$ solves

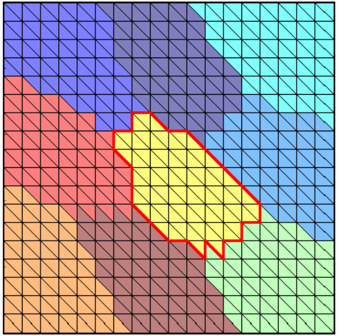
$$\begin{aligned} -\Delta v_i &= 0 \quad \text{in } \Omega_i, \\ v_i &= 1 \quad \text{on } \partial\Omega_i. \end{aligned}$$

Hence,

$$v_i = E_{\partial\Omega_i \to \Omega_i}\left(\mathbb{1}_{\partial\Omega_i}\right) = \mathbb{1}.$$

Therefore, for any partition of unity $\{\varphi_i\}_i$ on $\partial\Omega_i$, due to **linearity of the extension operator**, we have

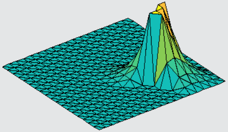
$$\sum_i \varphi_i = \mathbb{1}_{\partial\Omega_i} \Rightarrow \sum_i E_{\partial\Omega_i \to \Omega_i}(\varphi_i) = \mathbb{1}_{\Omega_i}$$



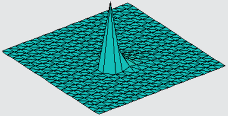
Null space property

Any extension-based coarse space built from a partition of unity on the domain decomposition interface satisfies the **null space property necessary for numerical scalability**:

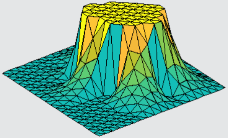
$\sum_{\text{edges} \subset \partial\Omega_i}$



$+ \sum_{\text{vertices} \subset \partial\Omega_i}$

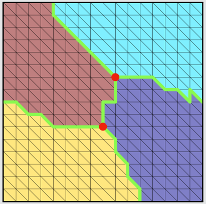


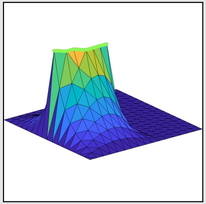
$=$



Examples of Extension-Based Coarse Spaces

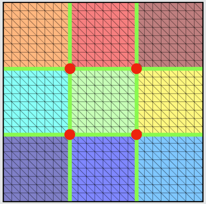
GDSW (Generalized Dryja–Smith–Widlund)

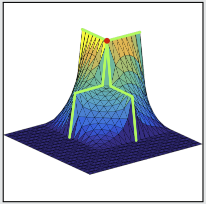




- Dohrmann, Klawonn, Widlund (2008)
- Dohrmann, Widlund (2009, 2010, 2012)

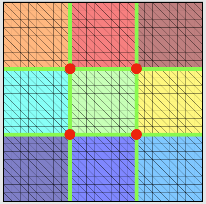
RGDSW (Reduced dimension GDSW)

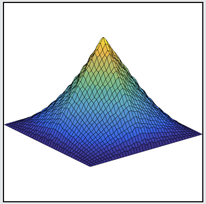




- Dohrmann, Widlund (2017)
- H., Klawonn, Knepper, Rheinbach, Widlund (2022)

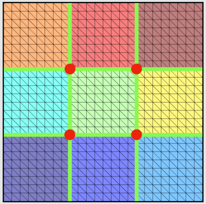
MsFEM (Multiscale Finite Element Method)

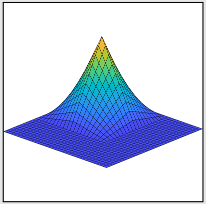




- Hou (1997), Efendiev and Hou (2009)
- Buck, Iliev, and Andrä (2013)
- H., Klawonn, Knepper, Rheinbach (2018)

Q1 Lagrangian / piecewise bilinear





Piecewise linear interface partition of unity functions and a **structured domain decomposition**.

Parallel Implementation in `FROSch`

FROSch (Fast and Robust Overlapping Schwarz) Framework in Trilinos

Software

- Object-oriented C++ domain decomposition solver framework with MPI-based distributed memory parallelization
- Part of `Trilinos` with support for both parallel linear algebra packages `Epetra` and `Tpetra`
- Node-level parallelization and performance portability on CPU and GPU architectures through `Kokkos`
- Accessible through unified `Trilinos` solver interface `Stratimikos`

Methodology

- Parallel scalable multi-level Schwarz domain decomposition preconditioners
- Algebraic construction based on the parallel distributed system matrix
- Extension-based coarse spaces

Team (active)

- Alexander Heinlein (TU Delft)
- Siva Rajamanickam (Sandia)
- Friederike Röver (TUBAF)
- Axel Klawonn (Uni Cologne)
- Oliver Rheinbach (TUBAF)
- Ichitaro Yamazaki (Sandia)

FROSch (Fast and Robust Overlapping Schwarz) Framework in Trilinos

Software

- Object-oriented C++ domain decomposition solver framework with MPI-based distributed memory parallelization
- Part of Trilinos with support for both parallel linear algebra packages Epetra and Tpetra
- Node-level parallelization and performance portability on CPU and GPU architectures through Kokkos
- Accessible through unified Trilinos solver interface Stratimikos

Methodology

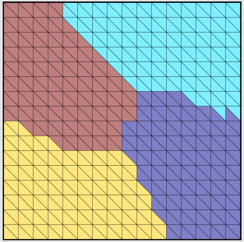
- **Parallel scalable** multi-level Schwarz domain decomposition preconditioners
- **Algebraic construction** based on the parallel distributed system matrix
- Extension-based coarse spaces

Team (active)

- Alexander Heinlein (TU Delft)
- Siva Rajamanickam (Sandia)
- Friederike Röver (TUBAF)
- Axel Klawonn (Uni Cologne)
- Oliver Rheinbach (TUBAF)
- Ichitaro Yamazaki (Sandia)

Overlapping domain decomposition

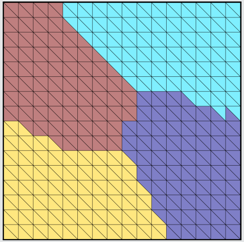
In FROSch, the overlapping subdomains $\Omega_1', \ldots, \Omega_N'$ are constructed by **recursively adding layers of elements** to the nonoverlapping subdomains; this can be performed based on the sparsity pattern of $\boldsymbol{K}$.



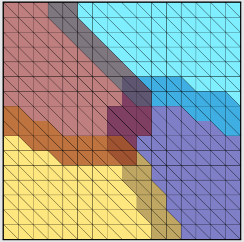
Nonoverlapping DD

Overlapping domain decomposition

In FROSch, the overlapping subdomains $\Omega_1', \ldots, \Omega_N'$ are constructed by **recursively adding layers of elements** to the nonoverlapping subdomains; this can be performed based on the sparsity pattern of $\boldsymbol{K}$.



Nonoverlapping DD

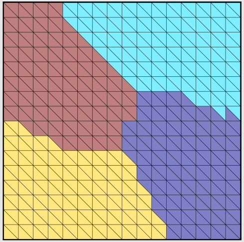


Overlap $\delta = 1h$

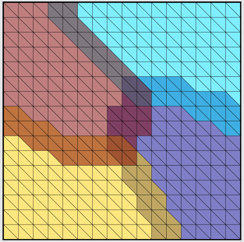
Algorithmic Framework for `FROSch` Overlapping Domain Decompositions

Overlapping domain decomposition

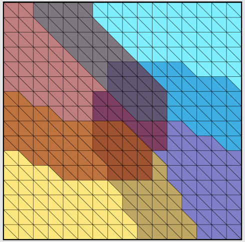
In `FROSch`, the overlapping subdomains $\Omega_1', \ldots, \Omega_N'$ are constructed by **recursively adding layers of elements** to the nonoverlapping subdomains; this can be performed based on the sparsity pattern of $\boldsymbol{K}$.



Nonoverlapping DD



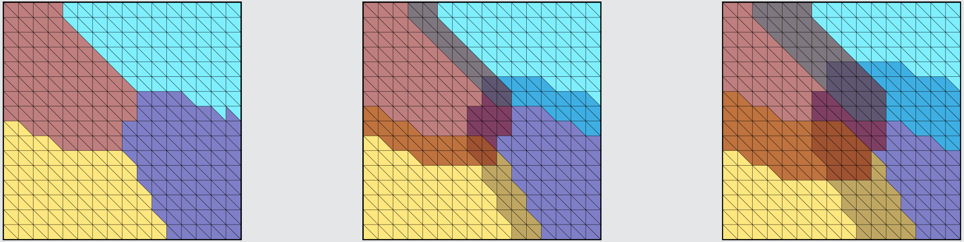
Overlap $\delta = 1h$



Overlap $\delta = 2h$

Overlapping domain decomposition

In FROSch, the overlapping subdomains $\Omega_1', \ldots, \Omega_N'$ are constructed by **recursively adding layers of elements** to the nonoverlapping subdomains; this can be performed based on the sparsity pattern of $\boldsymbol{K}$.



Nonoverlapping DD — Overlap $\delta = 1h$ — Overlap $\delta = 2h$

Computation of the overlapping matrices

The overlapping matrices

$$\boldsymbol{K}_i = \boldsymbol{R}_i \boldsymbol{K} \boldsymbol{R}_i^T$$

can easily be extracted from K since R_i is just a **global-to-local index mapping**.

Algorithmic Framework for FROSch Coarse Spaces

FROSch preconditioners use algebraic coarse spaces that are constructed in **four algorithmic steps**:

1. Identification of the **domain decomposition interface**
2. Construction of a **partition of unity (POU)** on the interface
3. Computation of a **coarse basis on the interface**
4. Harmonic extensions into the interior to obtain a **coarse basis** on the whole domain

Algorithmic Framework for FROSch Coarse Spaces

FROSch preconditioners use algebraic coarse spaces that are constructed in **four algorithmic steps**:

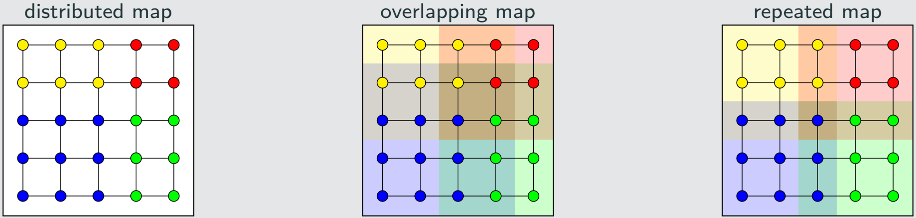
1. Identification of the **domain decomposition interface**
2. Construction of a **partition of unity (POU)** on the interface
3. Computation of a **coarse basis on the interface**
4. Harmonic extensions into the interior to obtain a **coarse basis** on the whole domain

Identification of the domain decomposition interface

If not provided by the user, FROSch will construct a **repeated map** where the interface (Γ) nodes are shared between processes from the parallel distribution of the matrix rows (**distributed map**).

Then, FROSch automatically identifies vertices, edges, and (in 3D) faces, by the multiplicities of the nodes.

$$
K = \begin{bmatrix} \bullet & \bullet & & & & & & \\ \bullet & \bullet & \bullet & & & & & \\ & \bullet & \bullet & \bullet & & & & \\ & & \bullet & \bullet & \bullet & & & \\ & & & \bullet & \bullet & \bullet & & \\ & & & & \bullet & \bullet & \bullet & \\ & & & & & \bullet & \bullet & \bullet \\ & & & & & & \bullet & \bullet \end{bmatrix} \qquad f = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \\ \bullet \end{bmatrix}
$$



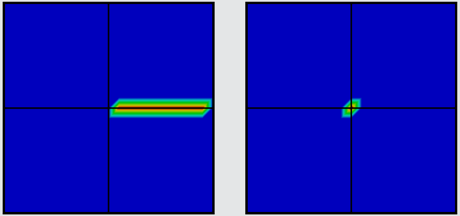
Algorithmic Framework for FROSch Coarse Spaces

FROSch preconditioners use algebraic coarse spaces that are constructed in **four algorithmic steps**:

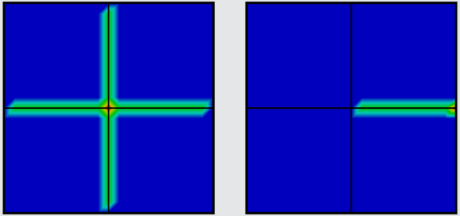
1. Identification of the **domain decomposition interface**
2. Construction of a **partition of unity (POU)** on the interface
3. Computation of a **coarse basis on the interface**
4. Harmonic extensions into the interior to obtain a **coarse basis** on the whole domain

Construction of a partition of unity on the interface

vertices, edges, and (in 3D) faces



overlapping vertex components

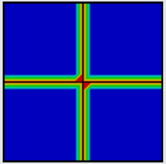


We construct a **partition of unity (POU)** $\{\pi_i\}_i$ with

$$\sum_i \pi_i = 1$$

on the interface Γ.

$\Rightarrow$



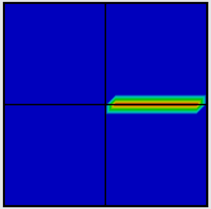
Algorithmic Framework for FROSch Coarse Spaces

FROSch preconditioners use **algebraic coarse spaces** that are constructed in **four algorithmic steps**:

1. Identification of the **domain decomposition interface**
2. Construction of a **partition of unity (POU)** on the interface
3. Computation of a **coarse basis on the interface**
4. Harmonic extensions into the interior to obtain a **coarse basis** on the whole domain

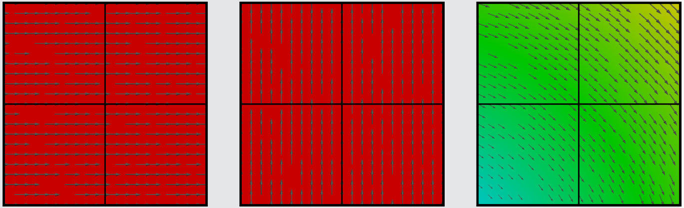
Computation of a coarse basis on the interface

interface POU function



$\times$

null space basis (linear elasticity: **translations**, **linearized rotation(s)**)



For each partition of unity function π_i, we compute a basis for the space

$$\operatorname{span}\{\pi_i \times z_j\}_j ,$$

where $\{z_j\}_j$ is a null space basis. In case of **linear dependencies**, we perform a **local QR factorization** to construct a basis.

This yields an **interface coarse basis** Φ_Γ.

The linearized rotation

$$\begin{bmatrix} y \\ -x \end{bmatrix}$$

depends on coordinates (geometric information).

Algorithmic Framework for FROSch Coarse Spaces

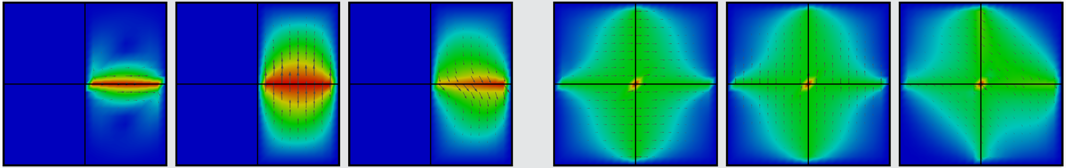
FROSch preconditioners use **algebraic coarse spaces** that are constructed in **four algorithmic steps**:

1. Identification of the **domain decomposition interface**
2. Construction of a **partition of unity (POU)** on the interface
3. Computation of a **coarse basis on the interface**
4. Harmonic extensions into the interior to obtain a **coarse basis** on the whole domain

Harmonic extensions into the interior

edge coarse basis functions

vertex component basis functions

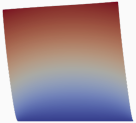


For each **interface coarse basis function**, we compute the interior values Φ_I by computing **harmonic / energy-minimizing extensions**:

$$\Phi = \begin{bmatrix} -K_{II}^{-1} K_{\Gamma I}^T \Phi_\Gamma \\ \Phi_\Gamma \end{bmatrix} = \begin{bmatrix} \Phi_I \\ \Phi_\Gamma \end{bmatrix}.$$

Algebraic FROSch Preconditioners for Elasticity

$$
\begin{aligned}
\operatorname{div} \boldsymbol{\sigma} &= (0, -100, 0)^T && \text{in } \Omega := [0,1]^3, \\
\boldsymbol{u} &= 0 && \text{on } \partial\Omega_D := \{0\} \times [0,1]^2, \\
\boldsymbol{\sigma} \cdot \boldsymbol{n} &= 0 && \text{on } \partial\Omega_N := \partial\Omega \setminus \partial\Omega_D
\end{aligned}
$$



St. Venant Kirchhoff material, P2 finite elements, $H/h = 9$; implementation in FEDDLib. (timings: setup + solve = **total**)

prec.	type	#cores	64	512	4 096
GDSW	rotations	#its.	**16.3**	**17.3**	**19.3**
		time	40.1 + 5.9 = **46.0**	55.0 + 8.5 = **63.5**	223.3 + 24.4 = **247.7**
	no rotations	#its.	**24.5**	**29.3**	**32.3**
		time	32.5 + 8.4 = **40.9**	38.4 + 11.8 = **46.7**	102.2 + 20.0 = **122.2**
	fully algebraic	#its.	**57.5**	**74.8**	**78.0**
		time	42.0 + 20.5 = **62.5**	46.0 + 29.9 = **75.9**	124.8 + 50.5 = **175.3**
RGDSW	rotations	#its.	**18.8**	**21.3**	**19.8**
		time	27.8 + 6.4 = **34.2**	31.1 + 8.0 = **39.1**	41.3 + 8.9 = **50.2**
	no rotations	#its.	**29.0**	**32.8**	**35.5**
		time	26.2 + 9.4 = **35.6**	27.3 + 11.8 = **39.1**	31.1 + 14.3 = **45.4**
	fully algebraic	#its.	**60.7**	**78.5**	**83.0**
		time	27.9 + 19.9 = **47.8**	28.7 + 27.9 = **56.6**	34.1 + 33.1 = **67.2**

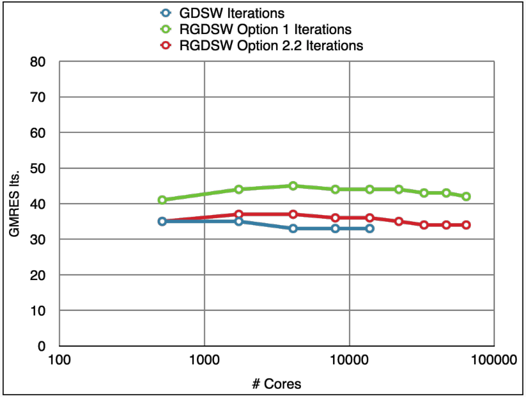
4 Newton iterations (with backtracking) were necessary for convergence (relative residual reduction of 10^{-8}) for all configurations.

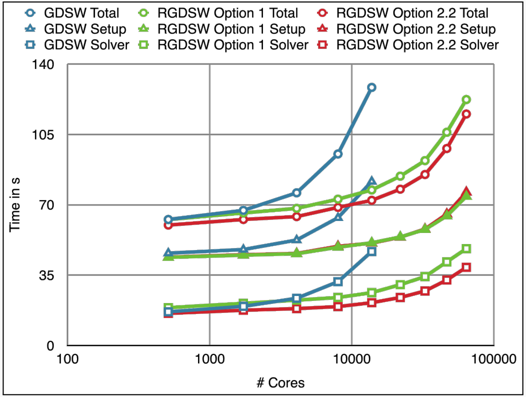
Computations on magnitUDE (University Duisburg-Essen). **Heinlein, Hochmuth, and Klawonn (2021)**

Weak Scalability up to $64\,\mathrm{k}$ MPI Ranks / $1.7\,\mathrm{b}$ Unknowns (3D Poisson; Juqueen)

Model problem: **Poisson equation in 3D** **Coarse solver: `MUMPS` (direct)**

Largest problem: **374 805 361 / 1 732 323 601 unknowns**



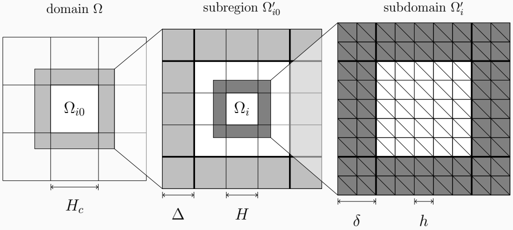


Cf. Heinlein, Klawonn, Rheinbach, Widlund (2017); computations performed on Juqueen, JSC, Germany.

⇒ Using the **reduced dimension coarse space**, we can **improve parallel scalability**.

To **extend the scalability even further**, we consider **multi-level Schwarz preconditioners**.

Three-Level GDSW Preconditioner



Heinlein, Klawonn, Rheinbach, Röver (2019, 2020), Heinlein, Rheinbach, Röver (accepted 2022)

Recursive approach

Instead of solving the coarse problem exactly, we apply another GDSW preconditioner on the coarse level $\Rightarrow$ **recursive application of the GDSW preconditioner**.

Therefore, we introduce **coarse subdomains on the coarse level**, denoted as **subregions**.

The **three-level GDSW preconditioner** is defined as

$$\boldsymbol{M}_{3GDSW}^{-1} = \underbrace{\Phi\left(\overbrace{\Phi_0 \boldsymbol{K}_{00}^{-1}\Phi_0^T}^{\text{third level}} + \overbrace{\sum_{i=1}^{N_0} \boldsymbol{R}_{i0}^T K_{i0}^{-1} \boldsymbol{R}_{i0}}^{\text{second level}}\right)\Phi^T}_{\text{coarse levels}} + \overbrace{\sum_{j=1}^{N} \boldsymbol{R}_j^T \boldsymbol{K}_j^{-1} \boldsymbol{R}_j}^{\text{first level}},$$

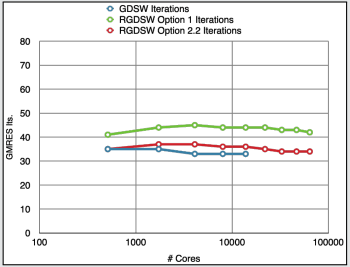
where $K_{00} = \Phi_0^T K_0 \Phi_0$ and $K_{i0} = R_{i0} K_0 R_{i0}^T \quad for\ i = 1, \cdots, N_0$.

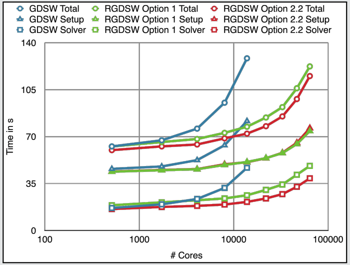
Here, let $R_{i0} : V^0 \to V_i^0 := V^0(\Omega'_{i0})$ for $i = 1, ..., N_0$ be **restriction operators on the subregion level** and Φ_0 contain to corresponding **coarse basis functions**. Our approach is related to other three-level DD methods; cf., e.g., three-level BDDC by Tu (2007).

Weak Scalability up to $64\,\mathrm{k}$ MPI Ranks / $1.7\,\mathrm{b}$ Unknowns (3D Poisson; Juqueen)

GDSW vs RGDSW (reduced dimension)

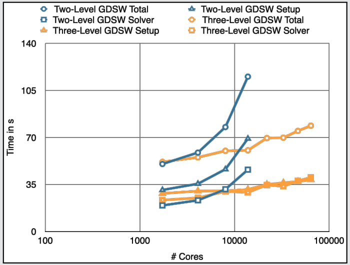
Heinlein, Klawonn, Rheinbach, Widlund (2019).





Two-level vs three-level GDSW

Heinlein, Klawonn, Rheinbach, Röver (2019, 2020).

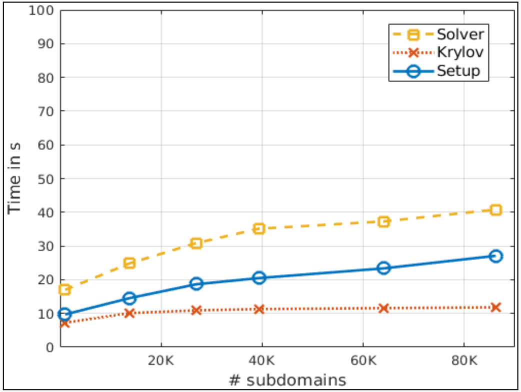


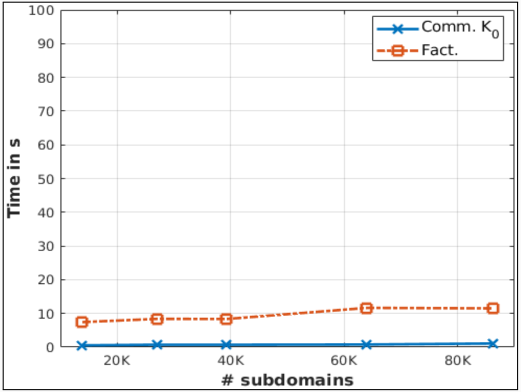
Weak Scalability of the Three-Level RGDSW Preconditioner for Linear Elasticity

In **Heinlein, Rheinbach, Röver (accepted 2022)**, it has been shown that the **null space can be transferred algebraically to higher levels**.

Model problem: **Linear elasticity in 3D**
Largest problem: **2 040 000 000 unknowns**

Coarse solver level 3: `MUMPS` (direct)





Cf. **Heinlein, Rheinbach, Röver (accepted 2022)**; computations performed on SuperMUC-NG, LRZ, Germany.

Monolithic GDSW preconditioner

Consider the discrete saddle point problem

$$\mathcal{A}x = \begin{bmatrix} \boldsymbol{K} & \boldsymbol{B}^T \\ \boldsymbol{B} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix} = \mathcal{b}.$$

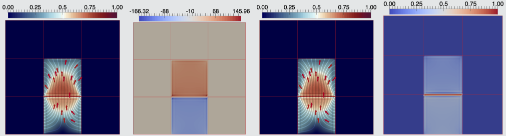
We construct a **monolithic GDSW Preconditioner**

$$\mathcal{M}_{\mathrm{GDSW}}^{-1} = \phi \mathcal{A}_0^{-1} \phi^T + \sum\nolimits_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i^{-1} \mathcal{R}_i,$$

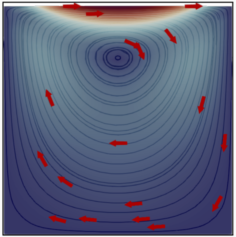
with block matrices $\mathcal{A}_0 = \phi^T \mathcal{A} \phi$, $\mathcal{A}_i = \mathcal{R}_i \mathcal{A} \mathcal{R}_i^T$, and

$$\mathcal{R}_i = \begin{bmatrix} \mathcal{R}_{u,i} & \boldsymbol{0} \\ \boldsymbol{0} & \mathcal{R}_{p,i} \end{bmatrix} \quad \text{and} \quad \phi = \begin{bmatrix} \Phi_{u,u_0} & \Phi_{u,p_0} \\ \Phi_{p,u_0} & \Phi_{p,p_0} \end{bmatrix}.$$

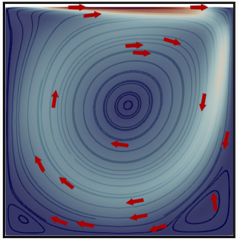
Using $\mathcal{A}$ to compute extensions: $\phi_I = -\mathcal{A}_{II}^{-1} \mathcal{A}_{I\Gamma} \phi_\Gamma$; cf. **Heinlein, Hochmuth, Klawonn (2019, 2020)**.



Φ_{u,u_0} Φ_{p,u_0} Φ_{u,p_0} Φ_{p,p_0}



Stokes flow



Navier–Stokes flow

Related work:

- Original work on monolithic Schwarz preconditioners: **Klawonn and Pavarino (1998, 2000)**
- Other publications on monolithic Schwarz preconditioners: e.g., **Hwang and Cai (2006)**, **Barker and Cai (2010)**, **Wu and Cai (2014)**, and the presentation **Dohrmann (2010)** at the *Workshop on Adaptive Finite Elements and Domain Decomposition Methods* in Milan.

Monolithic (R)GDSW Preconditioners for CFD Simulations

Monolithic GDSW preconditioner

Consider the discrete saddle point problem

$$\mathcal{A}x = \begin{bmatrix} \boldsymbol{K} & \boldsymbol{B}^T \\ \boldsymbol{B} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix} = \mathscr{b}.$$

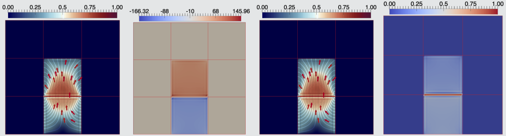
We construct a **monolithic GDSW Preconditioner**

$$\mathcal{M}_{\mathrm{GDSW}}^{-1} = \phi \mathcal{A}_0^{-1} \phi^T + \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i^{-1} \mathcal{R}_i,$$

with block matrices $\mathcal{A}_0 = \phi^T \mathcal{A} \phi$, $\mathcal{A}_i = \mathcal{R}_i \mathcal{A} \mathcal{R}_i^T$, and

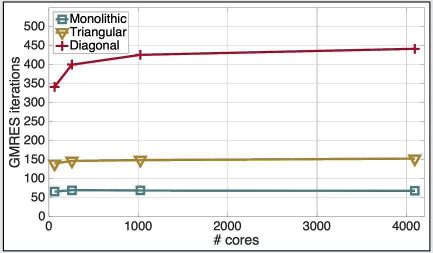
$$\mathcal{R}_i = \begin{bmatrix} \mathcal{R}_{u,i} & \boldsymbol{0} \\ \boldsymbol{0} & \mathcal{R}_{p,i} \end{bmatrix} \quad \text{and} \quad \phi = \begin{bmatrix} \Phi_{u,u_0} & \Phi_{u,p_0} \\ \Phi_{p,u_0} & \Phi_{p,p_0} \end{bmatrix}.$$

Using $\mathcal{A}$ to compute extensions: $\phi_I = -\mathcal{A}_{II}^{-1} \mathcal{A}_{I\Gamma} \phi_\Gamma$; cf. **Heinlein, Hochmuth, Klawonn (2019, 2020)**.



Φ_{u,u_0} Φ_{p,u_0} Φ_{u,p_0} Φ_{p,p_0}

Monolithic vs block preconditioners



prec.	MPI ranks	64	256	1 024	4 096
monolithic	time	154.7 s	170.0 s	175.8 s	188.7 s
	effic.	**100 %**	91 %	88 %	82 %
triangular	time	309.4 s	329.1 s	359.8 s	396.7 s
	effic.	50 %	47 %	43 %	39 %
diagonal	time	736.7 s	859.4 s	966.9 s	1 105.0 s
	effic.	21 %	18 %	16 %	14 %

Computations performed on magnitUDE, University Duisburg-Essen.

Monolithic GDSW preconditioner

Consider the discrete saddle point problem

$$\mathcal{A}x = \begin{bmatrix} \boldsymbol{K} & \boldsymbol{B}^T \\ \boldsymbol{B} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ 0 \end{bmatrix} = \mathcal{b}.$$

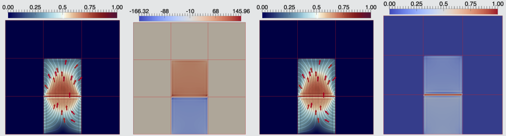
We construct a **monolithic GDSW Preconditioner**

$$\mathcal{M}_{\mathrm{GDSW}}^{-1} = \phi \mathcal{A}_0^{-1} \phi^T + \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i^{-1} \mathcal{R}_i,$$

with block matrices $\mathcal{A}_0 = \phi^T \mathcal{A} \phi$, $\mathcal{A}_i = \mathcal{R}_i \mathcal{A} \mathcal{R}_i^T$, and

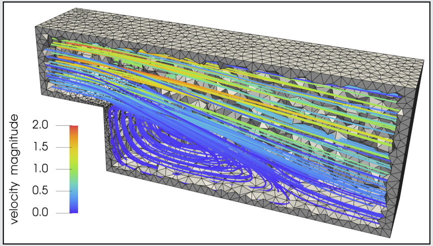
$$\mathcal{R}_i = \begin{bmatrix} \mathcal{R}_{u,i} & 0 \\ 0 & \mathcal{R}_{p,i} \end{bmatrix} \quad \text{and} \quad \phi = \begin{bmatrix} \Phi_{u,u_0} & \Phi_{u,p_0} \\ \Phi_{p,u_0} & \Phi_{p,p_0} \end{bmatrix}.$$

Using $\mathcal{A}$ to compute extensions: $\phi_I = -\mathcal{A}_{II}^{-1} \mathcal{A}_{I\Gamma} \phi_\Gamma$; cf. **Heinlein, Hochmuth, Klawonn (2019, 2020)**.



Φ_{u,u_0} Φ_{p,u_0} Φ_{u,p_0} Φ_{p,p_0}

Monolithic vs SIMPLE preconditioner



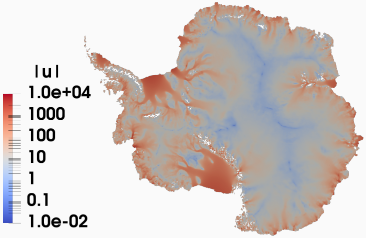
Steady-state Navier-Stokes equations

prec.	MPI ranks	243	1 125	15 562
Monolithic RGDSW (FROSch)	setup	39.6 s	57.9 s	95.5 s
	solve	57.6 s	69.2 s	74.9 s
	total	**97.2 s**	**127.7 s**	**170.4 s**
SIMPLE RGDSW (Teko & FROSch)	setup	39.2 s	38.2 s	68.6 s
	solve	86.2 s	106.6 s	127.4 s
	total	125.4 s	144.8 s	196.0 s

Computations on Piz Daint (CSCS). Implementation in the finite element software `FEDDLib`.

Albany

https://github.com/SNLComputation/Albany



The velocity of the ice sheet in Antarctica and Greenland is modeled by a **first-order-accurate Stokes approximation model**,

$$-\nabla \cdot (2\mu\dot{\epsilon}_1) + \rho g \frac{\partial s}{\partial x} = 0, \quad -\nabla \cdot (2\mu\dot{\epsilon}_2) + \rho g \frac{\partial s}{\partial y} = 0,$$

with a **nonlinear viscosity model** (Glen's law); cf., e.g., **Blatter (1995)** and **Pattyn (2003)**.

	Antarctica (**velocity**) 4 km resolution, 20 layers, 35 m dofs			Greenland (**multiphysics vel. & temperature**) 1-10 km resolution, 20 layers, 69 m dofs		
MPI ranks	avg. its	avg. setup	avg. solve	avg. its	avg. setup	avg. solve
512	41.9 (11)	25.10 s	12.29 s	41.3 (36)	18.78 s	4.99 s
1 024	43.3 (11)	9.18 s	5.85 s	53.0 (29)	8.68 s	4.22 s
2 048	41.4 (11)	4.15 s	2.63 s	62.2 (86)	4.47 s	4.23 s
4 096	41.2 (11)	1.66 s	1.49 s	68.9 (40)	2.52 s	2.86 s
8 192	40.2 (11)	1.26 s	1.06 s	-	-	-

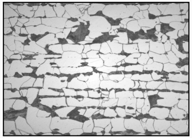
Computations on Cori (NERSC).

Heinlein, Perego, Rajamanickam (2022)

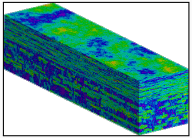
Adaptive Extension-Based Coarse Spaces

Highly Heterogeneous Multiscale Problems

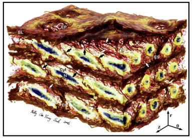
Highly heterogeneous multiscale problems appear in most areas of modern science and engineering, e.g., **composite materials**, **porous media**, and **turbulent transport in high Reynolds number flow**.



Micro section of a dual-phase steel. Courtesy of **J. Schröder**.



Groundwater flow (SPE10); cf. **Christie and Blunt (2001)**.



Composition of arterial walls; taken from **O'Connell et al. (2008)**.

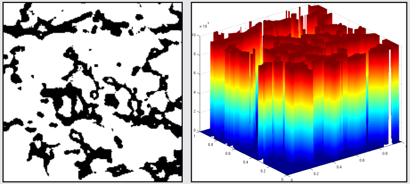
$\rightarrow$ The solution of such problems requires a **high spatial and temporal resolution** but also poses **challenges to the solvers**.

Heterogeneous model problem

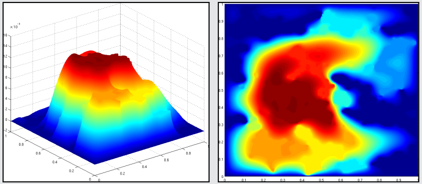
Consider the **heterogeneous diffusion boundary value problem**:

$$
\begin{aligned}
-\nabla \cdot (\alpha(x) \nabla u(x)) &= f(x) \quad && \text{in } \Omega, \\
u &= 0 \quad && \text{on } \partial\Omega.
\end{aligned}
$$

Binary coefficient function



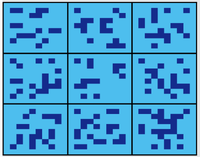
Solution of the BVP



Observations for Heterogeneous Problems

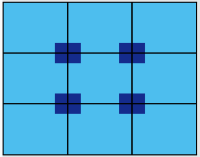
10×10 subdomains with $H/h = 10$ and overlap $1h$ **dark blue:** $\alpha = 10^8$ **light blue:** $\alpha = 1$

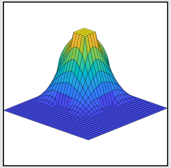
Heterogeneities inside subdomains



Prec.	its.	κ
–	**>2 000**	$7.99 \cdot 10^8$
$M_{\text{OS-1}}^{-1}$	**64**	133.16
$M_{\text{OS-2}}^{-1}$	**78**	139.15

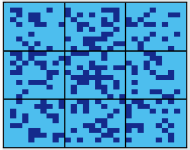
Vertex inclusions



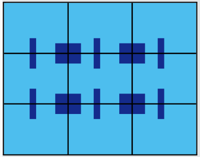


Prec.	its.	κ
–	**874**	$1.35 \cdot 10^9$
$M_{\text{OS-1}}^{-1}$	**163**	$4.06 \cdot 10^7$
$M_{\text{OS-2}}^{-1}$	**138**	$1.07 \cdot 10^6$
M_{MsFEM}^{-1}	**24**	8.05

General cases



Prec.	its.	κ
–	**>2 000**	$4.51 \cdot 10^8$
$M_{\text{OS-1}}^{-1}$	**>2 000**	$4.51 \cdot 10^8$
$M_{\text{OS-2}}^{-1}$	**586**	$5.56 \cdot 10^5$



Prec.	its.	κ
–	**1708**	$1.16 \cdot 10^9$
$M_{\text{OS-1}}^{-1}$	**447**	$4.17 \cdot 10^7$
$M_{\text{OS-2}}^{-1}$	**268**	$1.10 \cdot 10^6$
M_{MsFEM}^{-1}	**117**	$4.34 \cdot 10^5$

Idea of Adptive Coarse Spaces

Assumption 1: Stable Decomposition

There exists a constant C_0, s.t. for every $\boldsymbol{u} \in V$, there exists a decomposition $\boldsymbol{u} = \sum_{i=0}^{N} \boldsymbol{R}_i^T \boldsymbol{u}_i$, $\boldsymbol{u}_i \in V_i$, with

$$\sum_{i=0}^{N} a_i(\boldsymbol{u}_i, \boldsymbol{u}_i) \leq C_0^2 a(\boldsymbol{u}, \boldsymbol{u}).$$

Assumption 2: Strengthened Cauchy–Schwarz Inequality

There exist constants $0 \leq \epsilon_{ij} \leq 1$, $1 \leq i, j \leq N$, s.t.

$$\begin{aligned}\left|a(\boldsymbol{R}_i^T \boldsymbol{u}_i, \boldsymbol{R}_j^T u_j)\right| \quad &\leq \epsilon_{ij} \quad \left(a(\boldsymbol{R}_i^T \boldsymbol{u}_i, \boldsymbol{R}_i^T \boldsymbol{u}_i)\right)^{1/2} \\ &\qquad\quad \left(a(\boldsymbol{R}_j^T \boldsymbol{u}_j, \boldsymbol{R}_j^T \boldsymbol{u}_j)\right)^{1/2}\end{aligned}$$

for $\boldsymbol{u}_i \in V_i$ and $\boldsymbol{u}_j \in V_j$.
(Consider $\mathcal{E} = (\varepsilon_{ij})$ and $\rho(\mathcal{E})$ its spectral radius)

Assumption 3: Local Stability

There exists $\omega < 0$, such that, for $0 \leq \boldsymbol{u} \neq N$,

$$a(\boldsymbol{R}_i^T \boldsymbol{u}_i, \boldsymbol{R}_i^T \boldsymbol{u}_i) \leq \omega a_i(\boldsymbol{u}_i, \boldsymbol{u}_i), \quad \boldsymbol{u}_i \in \text{range}\left(\tilde{P}_i\right).$$

Idea of Adptive Coarse Spaces

Assumption 1: Stable Decomposition

There exists a constant C_0, s.t. for every $\boldsymbol{u} \in V$, there exists a decomposition $\boldsymbol{u} = \sum_{i=0}^{N} \boldsymbol{R}_i^T \boldsymbol{u}_i$, $\boldsymbol{u}_i \in V_i$, with

$$\sum_{i=0}^{N} a_i(\boldsymbol{u}_i, \boldsymbol{u}_i) \leq C_0^2 a(\boldsymbol{u}, \boldsymbol{u}).$$

Assumption 2: Strengthened Cauchy–Schwarz Inequality

There exist constants $0 \leq \epsilon_{ij} \leq 1$, $1 \leq i, j \leq N$, s.t.

$$\begin{aligned} \left| a(\boldsymbol{R}_i^T \boldsymbol{u}_i, \boldsymbol{R}_j^T u_j) \right| \quad &\leq \epsilon_{ij} \quad \left(a(\boldsymbol{R}_i^T \boldsymbol{u}_i, \boldsymbol{R}_i^T \boldsymbol{u}_i) \right)^{1/2} \\ &\qquad\quad \left(a(\boldsymbol{R}_j^T \boldsymbol{u}_j, \boldsymbol{R}_j^T \boldsymbol{u}_j) \right)^{1/2} \end{aligned}$$

for $\boldsymbol{u}_i \in V_i$ and $\boldsymbol{u}_j \in V_j$.
(Consider $\mathcal{E} = (\varepsilon_{ij})$ and $\rho(\mathcal{E})$ its spectral radius)

Assumption 3: Local Stability

There exists $\omega < 0$, such that, for $0 \leq \boldsymbol{u} \neq N$,

$$a(\boldsymbol{R}_i^T \boldsymbol{u}_i, \boldsymbol{R}_i^T \boldsymbol{u}_i) \leq \omega a_i(\boldsymbol{u}_i, \boldsymbol{u}_i), \quad \boldsymbol{u}_i \in \text{range}\left(\tilde{P}_i\right).$$

Idea of adaptive coarse spaces

Ensure

$$a(\boldsymbol{u}_0, \boldsymbol{u}_0) \leq C_0^2 a(u, u)$$

by introducing two bilinear forms $c(\cdot,\cdot)$ and $d(\cdot,\cdot)$

$$a(\boldsymbol{u}_0, \boldsymbol{u}_0) \leq C_1 d(\boldsymbol{u}_0, \boldsymbol{u}_0) \quad \textbf{(high energy)}$$

and

$$c(\boldsymbol{u}_0, \boldsymbol{u}_0) \leq C_2 a(\boldsymbol{u}, \boldsymbol{u}), \quad \textbf{(low energy)}$$

where **$C_1 C_2$ is independent of the contrast of the coefficient function** and $u_0 := I_0 u$ is a suitable coarse function.

We **enhance the coarse space by all eigenvectors with eigenvalues below a tolerance** *tol* of

$$d(\boldsymbol{v}, \boldsymbol{w}) = \lambda\, c(\boldsymbol{v}, \boldsymbol{w})$$

and directly obtain

$$\begin{aligned} a(u_0, u_0) \quad &\leq \quad C_1\, d(\boldsymbol{u}_0, \boldsymbol{u}_0) \leq C_1\, tol\, c(\boldsymbol{u}_0, \boldsymbol{u}_0) \\ &\leq \quad C_1\, C_2\, tol\, a(\boldsymbol{u}, \boldsymbol{u}) \end{aligned}$$

In practice, **eigenvalue problem is partitioned into many local eigenvalue problems** $\rightarrow$ parallelization!

Adaptive Coarse Spaces in Domain Decomposition Methods – Literature Overview

This list is **not exhaustive**:

- **FETI & Neumann–Neumann:** Bjørstad and Krzyzanowski (2002); Bjørstad, Koster, and Krzyzanowski (2001); Rixen and Spillane (2013); Spillane (2015, 2016)
- **BDDC & FETI-DP:** Mandel and Sousedík (2007); Sousedík (2010); Sístek, Mandel, and Sousedík (2012); Dohrmann and Pechstein (2013, 2016); Klawonn, Radtke, and Rheinbach (2014, 2015, 2016); Klawonn, Kühn, and Rheinbach (2015, 2016, 2017); Kim and Chung (2015); Kim, Chung, and Wang (2017); Beirão da Veiga, Pavarino, Scacchi, Widlund, and Zampini (2017); Calvo and Widlund (2016); Oh, Widlund, Zampini, and Dohrmann (2017); Klawonn, Lanser, and Wasiak (preprint 2021)
- **Overlapping Schwarz:** Galvis and Efendiev (2010, 2011); Nataf, Xiang, Dolean, and Spillane (2011); Spillane, Dolean, Hauret, Nataf, Pechstein, and Scheichl (2011); Gander, Loneland, and Rahman (preprint 2015); Eikeland, Marcinkowski, and Rahman (preprint 2016); Heinlein, Klawonn, Knepper, Rheinbach (2018); Marcinkowski and Rahman (2018); Al Daas, Grigori, Jolivet, Tournier (2021); Bastian, Scheichl, Seelinger, and Strehlow (2022); Spillane (preprint 2021, preprint 2021); Bootland, Dolean, Graham, Ma, Scheichl (preprint 2021); Al Daas and Jolivet (preprint 2021)
- Approaches for overlapping Schwarz methods in **this talk**:
  - **AGDSW:** Heinlein, Klawonn, Knepper, Rheinbach (2019, 2019), Heinlein, Klawonn, Knepper, Rheinbach, and Widlund (2022)
  - **Fully Algebraic Coarse Space:** Heinlein and Smetana (Preprint: arXiv:2207.05559)

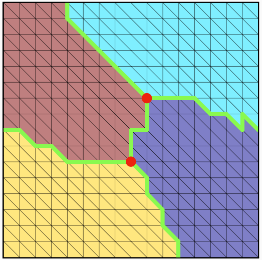
There is also related work on multigrid methods, such as **AMGe** by Brezina, Cleary, Falgout, Henson, Jones, Manteuffel, McCormick, Ruge (2000).

The **adaptive GDSW (AGDSW) coarse space** is a related approach, which also depends on a **partition of the domain decomposition interface** into edges and vertices. We use

- the **GDSW vertex basis functions** and
- edge functions computed from a **generalized edge eigenvalue problem**.

As a result, the AGDSW coarse space

- always **contains the classical GDSW coarse space**.

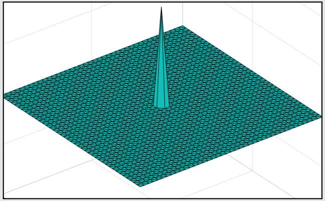


Cf. **Heinlein, Klawonn, Knepper, Rheinbach (2019, 2019)**.

AGDSW vertex basis function

The interior values are then obtained by extending 1 by zero onto the remainder of the interface followed by an energy minimizing extension into the interior:

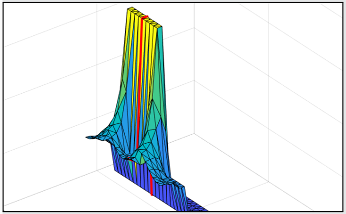
$$\varphi_v = E_{\Gamma \to \Omega}\left(R_{v \to \Gamma}\left(\mathbb{1}_v\right)\right)$$



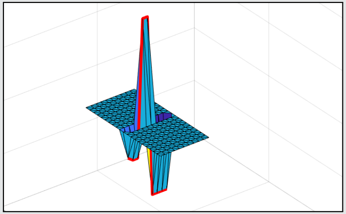
AGDSW – An Adaptive GDSW Coarse Space

AGDSW edge basis functions

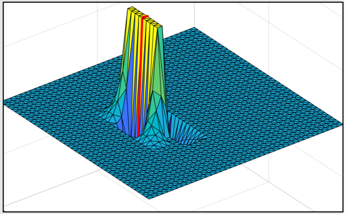
Low energy extension $\boldsymbol{E}_{e \rightarrow \Omega_e}(\cdot)$



High energy extension $\boldsymbol{R}_{e \rightarrow \Omega_e}(\cdot)$



Ext. into the interior



First, we solve the following eigenvalue problem (**in a-harmonic space**) for each edge $e \in \mathcal{E}$:

$$a_{\Omega_e}\left(E_{e\rightarrow\Omega_e}\left(\tau_{e,*}\right), E_{e\rightarrow\Omega_e}\left(\theta\right)\right) = \lambda_{e,*}\, a_{\Omega_e}\left(R_{e\rightarrow\Omega_e}\left(\tau_{e,*}\right), R_{e\rightarrow\Omega_e}\left(\theta\right)\right) \quad \forall \theta \in V_e$$

Then, we select eigenfunctions using the threshold TOL and extend the edge values to Ω:

$$\varphi_{e,*} = E_{\Gamma\rightarrow\Omega}\left(R_{e\rightarrow\Gamma}\left(\tau_{e,*}\right)\right)$$

Condition number bound

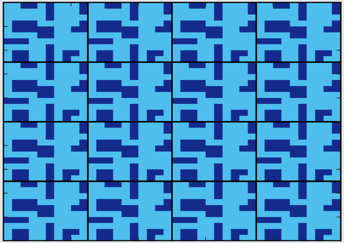
Using the coarse space $V_{\text{AGDSW}} = \{\varphi_v\} \cup \{\varphi_e\}$ in the two-level Schwarz preconditioner, we obtain

$$\kappa\left(\boldsymbol{M}_{\text{AGDSW}}^{-1}\boldsymbol{K}\right) \leq C\left(1/TOL\right),$$

where C is independent of H, h, and the contrast of the coefficient function α.

Numerical Results of Adaptive Coarse Spaces (2D)

Example 1

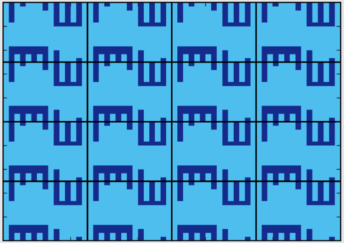


dark blue: $\alpha = 10^8$ **light blue:** $\alpha = 1$

4×4 subdomains, $H/h = 30$, $\delta = 2h$

V_0	*tol*	it.	κ	dim V_0
V_{MsFEM}	-	**199**	$7.8 \cdot 10^5$	**9**
$V_{\text{OS-ACMS}}$	10^{-2}	**23**	5.1	**69**
V_{SHEM}	10^{-3}	**20**	4.3	**69**
V_{AGDSW}	10^{-2}	**29**	7.2	**93**

Example 2



dark blue: $\alpha = 10^8$ **light blue:** $\alpha = 1$

4×4 subdomains, $H/h = 30$, $\delta = 2h$

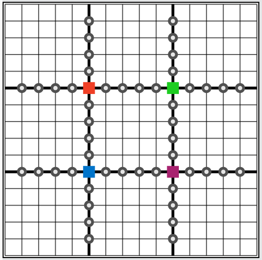
V_0	*tol*	it.	κ	dim V_0
V_{MsFEM}	-	**282**	$3.8 \cdot 10^7$	9
$V_{\text{OS-ACMS}}$	10^{-2}	**41**	13.2	**33**
V_{SHEM}	10^{-3}	**29**	6.4	**93**
V_{AGDSW}	10^{-2}	**42**	16.5	**45**

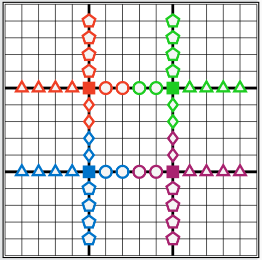
SHEM by Gander, Loneland, Rahman (TR 2015), **OS-ACMS** from H., Klawonn, Knepper, Rheinbach (2018), **AGDSW** from H., Klawonn, Knepper, Rheinbach (2019)

Extensions of the AGDSW Approach

Reducing the coarse space dimension

GDSW partition | RGDSW partition



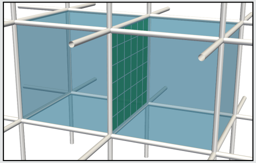


As in the reduced dimension GDSW (RGDSW) approach, we partition the interface into **interface components centered around the vertices**. On these interface components, we solve (slightly modified) eigenvalue problems.

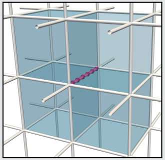
Cf. **Heinlein, Klawonn, Knepper, Rheinbach (2021)** and **Heinlein, Klawonn, Knepper, Rheinbach, Widlund (2022)**.

Extension to three dimensions

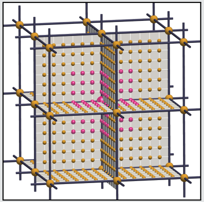
Face



Edge



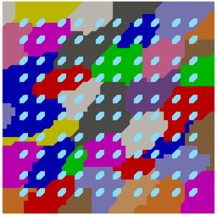
- In AGDSW, we have to solve **face and edge eigenvalue problems**
- In RAGDSW, only the definition of the **interface components changes**



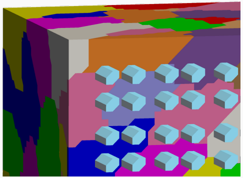
RGDSW interface component

Reduced Dimension (Adaptive) GDSW – 3D Numerical Example

cross section



detailed view of partially peeled beams



Heterogeneous linear elasticity problem

- Ω: cube; Dirichlet boundary condition on $\partial\Omega$.
- Structured tetrahedral mesh; 132 651 nodes (397 953 DOFs); unstructured domain decomposition (METIS); 125 subdomains.
- Poisson ration $\nu = 0.4$.
- Young modulus: elements with $E(T) = 10^6$ in light blue (beams); remainder set to $E(T) = 1$.
- Right hand side $f \equiv 1$.
- Overlap: two layers of finite elements.

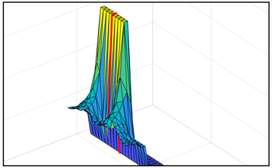
V_0	*tol*	iter	κ	dim V_0	$\frac{\dim V_0}{\dim V^h}$
GDSW	–	>2 000	$3.1 \cdot 10^5$	9 996	2.51%
RGDSW	–	>2 000	$3.9 \cdot 10^5$	3 358	0.84%
AGDSW	0.100	71	41.1	14 439	3.63%
AGDSW	0.050	90	59.5	13 945	3.50%
AGDSW	0.010	132	161.1	13 763	3.46%
RAGDSW	0.100	67	34.6	8 249	2.07%
RAGDSW	0.050	88	61.3	7 683	1.93%
RAGDSW	0.010	114	117.4	7 501	1.88%

- RAGDSW: 45% reduction of coarse space dimension compared to AGDSW (highlighted line).
- RAGDSW: smaller coarse space dimension compared to GDSW and still robust!

Neumann Matrices and Algebraicity

The **low energy property**

$$c(u_0, u_0) \leq C_2 a(u, u)$$



of the bilinear form in tge **left hand side of the eigenvalue problems** of AGDSW method is satisfied due to the use of **Neumann boundary conditions**:

$$a_{\Omega_e}\left(E_{e\to\Omega_e}\left(\tau_{e,*}\right), E_{e\to\Omega_e}(\theta)\right) = \lambda_{e,*} a_{\Omega_e}\left(R_{e\to\Omega_e}\left(\tau_{e,*}\right), R_{e\to\Omega_e}(\theta)\right) \quad \forall \theta \in V_e^0$$

The right hand side matrix just corresponds to the submatrix $\boldsymbol{K}_{ee}$ of $\boldsymbol{K}$ corresponding to the edge e, whereas the Neumann matrices on the left hand sides cannot be extracted from the fully assembled matrix $\boldsymbol{K}$. $\rightarrow$ **not algebraic**

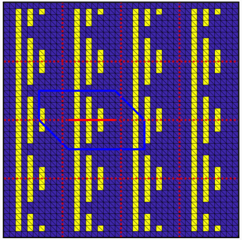
Fully Algebraic Adaptive Coarse Space

We can make use of the a-orthogonal decomposition

$$V_{\Omega_e} = V_{\Omega_e}^0 \oplus \underbrace{\{E_{\partial\Omega_e \to \Omega_e}(v) : v \in V_{\partial\Omega_e}\}}_{=: V_{\Omega_e,\text{harm}}}$$

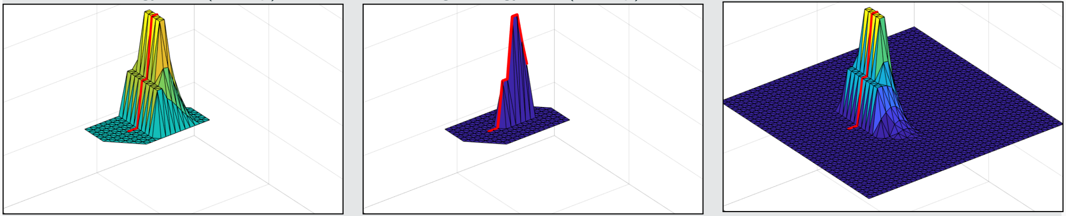
to ***"split the AGDSW eigenvalue problem"*** into two:

- **Dirichlet eigenvalue problem** on $V_{\Omega_e}^0$
- **Transfer eigenvalue problem** on $V_{\Omega_e,\text{harm}}$; cf. **Smetana, Patera (2016)**



Dirichlet eigenvalue problem

Low energy ext. (lhs evp) | High energy ext. (rhs evp) | Basis function

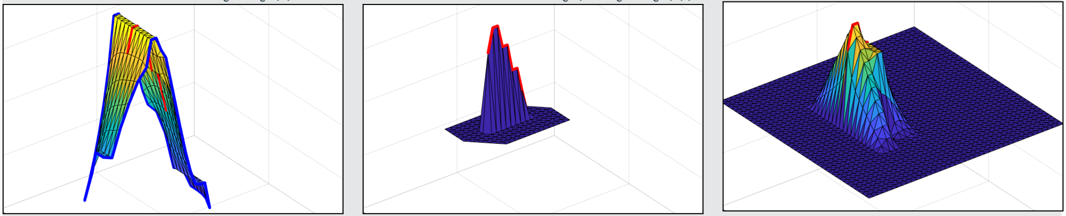


We solve the eigenvalue problem, choose $\lambda_{e,*} < TOL_1$, and extend the basis functions to Ω as before:

$$a_{\Omega_e}\left(E_{e\to\Omega_e}^{\partial\Omega_e}(\tau_{e,*}), E_{e\to\Omega_e}^{\partial\Omega_e}(\theta)\right) = \lambda_{e,*} a_{\Omega_e}\left(R_{e\to\Omega_e}(\tau_{e,*}), R_{e\to\Omega_e}(\theta)\right) \quad \forall \theta \in V_e^0$$

Transfer eigenvalue problem

Low energy ext. $E_{\partial\Omega_e\to\Omega_e}(\cdot)$ High energy ext. $R_{e\to\Omega_e}(E_{\partial\Omega_e\to\Omega_e}(\cdot))$ Basis function



The transfer eigenvalue problem is based on **Smetana, Patera (2016)**. Different from all the eigenvalue problems before, it is solved on the boundary of Ω_e:

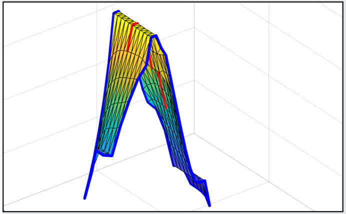
$$a_{\Omega_e}\left(E_{\partial\Omega_e\to\Omega_e}(\eta_{e,*}), E_{\partial\Omega_e\to\Omega_e}(\theta)\right) = \lambda_{e,*}\, a_{\Omega_e}\left(R_{e\to\Omega_e}(E_{\partial\Omega_e\to\Omega_e}(\tau_{e,*})), R_{e\to\Omega_e}(E_{\partial\Omega_e\to\Omega_e}(\theta))\right) \quad \forall \theta \in V^0_{\partial\Omega_e}$$

We select all eigenfunctions $\eta_{e,*}$ with $\lambda_{e,*}$ above a second **user-chosen threshold** TOL_2. Then, we first compute the edge values $\tau_{e,*} = E_{\partial\Omega_e\to\Omega_e}(\eta_{e,*})\,|_e$ and then extend them into the interior

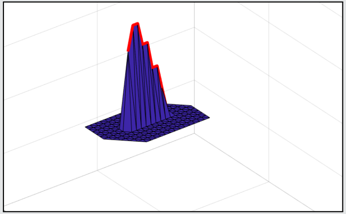
$$\varphi_{e,*} = E_{\Gamma\to\Omega}\left(R_{e\to\Gamma}(\tau_{e,*})\right)$$

Transfer eigenvalue problem

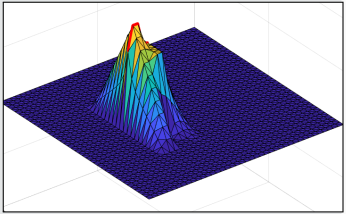
Low energy ext. $E_{\partial\Omega_e \to \Omega_e}(\cdot)$



High energy ext. $R_{e \to \Omega_e}\left(E_{\partial\Omega_e \to \Omega_e}(\cdot)\right)$



Basis function



The transfer eigenvalue problem is based on **Smetana, Patera (2016)**. Different from all the eigenvalue problems before, it is solved on the boundary of Ω_e:

$$a_{\Omega_e}\left(E_{\partial\Omega_e \to \Omega_e}(\eta_{e,*}), E_{\partial\Omega_e \to \Omega_e}(\theta)\right) = \lambda_{e,*} a_{\Omega_e}\left(R_{e \to \Omega_e}\left(E_{\partial\Omega_e \to \Omega_e}(\tau_{e,*})\right), R_{e \to \Omega_e}\left(E_{\partial\Omega_e \to \Omega_e}(\theta)\right)\right) \quad \forall \theta \in V^0_{\partial\Omega_e}$$

We select all eigenfunctions $\eta_{e,*}$ with $\lambda_{e,*}$ above a second **user-chosen threshold** TOL_2. Then, we first compute the edge values $\tau_{e,*} = E_{\partial\Omega_e \to \Omega_e}(\eta_{e,*})\,|_e$ and then extend them into the interior

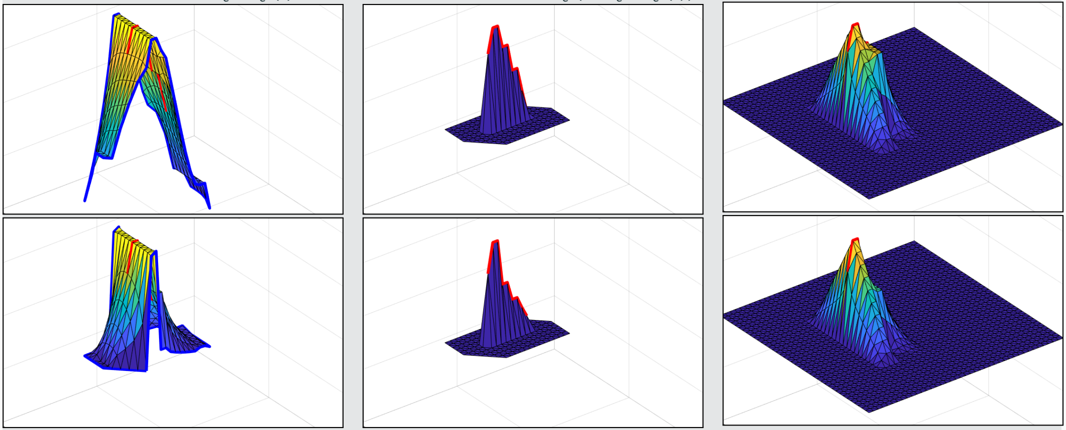
$$\varphi_{e,*} = E_{\Gamma \to \Omega}\left(R_{e \to \Gamma}(\tau_{e,*})\right)$$

$\to$ Even though **no Neumann matrices are needed to compute $E_{\partial\Omega_e \to \Omega_e}(\theta)$**, **Neumann matrices are needed to evaluate $a_{\Omega_e}(\cdot,\cdot)$** for functions with nonnegative trace on $\partial\Omega_e$

Fully Algebraic Adaptive Coarse Space – Transfer Eigenvalue Problem

Algebraic transfer eigenvalue problem

Low energy ext. $E_{\partial\Omega_e \to \Omega_e}(\cdot)$ | High energy ext. $R_{e \to \Omega_e}(E_{\partial\Omega_e \to \Omega_e}(\cdot))$ | Basis function for $a_{\Omega_e}(\cdot,\cdot)$



Low energy ext. $E_{\partial\Omega_e \to \Omega_e}(\cdot)$ | High energy ext. $R_{e \to \Omega_e}(E_{\partial\Omega_e \to \Omega_e}(\cdot))$ | Basis function for $(\cdot,\cdot)_{l_2(\partial\Omega_e)}$

In order to obtain an algebraic transfer eigenvalue problem, we replace $a_{\Omega_e}(\cdot,\cdot)$ by $(\cdot,\cdot)_{l_2(\partial\Omega_e)}$:

$$\left(E_{\partial\Omega_e \to \Omega_e}(\tau_{e,*}), E_{\partial\Omega_e \to \Omega_e}(\theta)\right)_{l_2(\partial\Omega_e)} = \lambda_{e,*} a_{\Omega_e}\left(R_{e \to \Omega_e}(E_{\partial\Omega_e \to \Omega_e}(\tau_{e,*})), R_{e \to \Omega_e}(E_{\partial\Omega_e \to \Omega_e}(\theta))\right) \quad \forall \theta \in V^0_{\partial\Omega_e}$$

Fully Algebraic Adaptive Coarse Space – Condition Number Bound

Condition number estimate (non-algebraic variant)

Using the non-algebraic eigenvalue problem (transfer eigenvalue problem with $a_{\Omega_e}(\cdot,\cdot)$), we obtain a condition number of the form:

$$\kappa\left(\boldsymbol{M}_{\mathrm{DIR\&TR}}^{-1}\boldsymbol{K}\right) \leq C \max\left(\frac{1}{TOL_1}, TOL_2\right),$$

where C is independent of H, h, and the contrast of the coefficient function α.

Condition number estimate (algebraic variant)

Using the algebraic eigenvalue problem (transfer eigenvalue problem with $(\cdot,\cdot)_{l_2(\partial\Omega_e)}$), we obtain a condition number of the form:

$$\kappa\left(\boldsymbol{M}_{\mathrm{DIR\&TR}}^{-1}\boldsymbol{K}\right) \leq C \max\left\{\frac{1}{TOL_1}, \frac{TOL_2}{\alpha_{\min}}\right\},$$

where C is independent of H, h, and the contrast of the coefficient function α.

Cf. **Heinlein and Smetana (Preprint: arXiv:2207.05559)**.

Fully Algebraic Adaptive Coarse Space – Condition Number Bound

Condition number estimate (non-algebraic variant)

Using the non-algebraic eigenvalue problem (transfer eigenvalue problem with $a_{\Omega_e}(\cdot,\cdot)$), we obtain a condition number of the form:

$$\kappa\left(\boldsymbol{M}_{\text{DIR\&TR}}^{-1}\boldsymbol{K}\right) \leq C \max\left(\frac{1}{TOL_1}, TOL_2\right),$$

where C is independent of H, h, and the contrast of the coefficient function α.

Condition number estimate (algebraic variant)

Using the algebraic eigenvalue problem (transfer eigenvalue problem with $(\cdot,\cdot)_{l_2(\partial\Omega_e)}$), we obtain a condition number of the form:

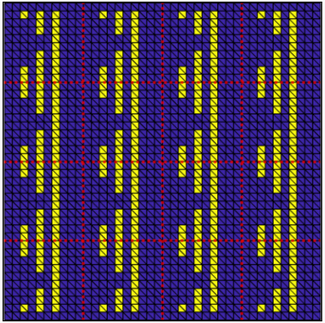
$$\kappa\left(\boldsymbol{M}_{\text{DIR\&TR}}^{-1}\boldsymbol{K}\right) \leq C \max\left\{\frac{1}{TOL_1}, \frac{TOL_2}{\alpha_{\min}}\right\},$$

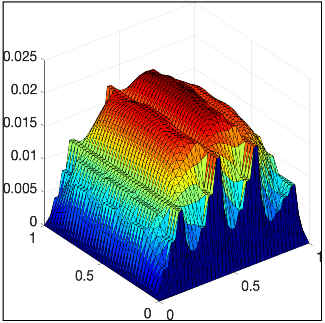
where C is independent of H, h, and the contrast of the coefficient function α.

$\rightarrow$ The $\alpha_{\min}$ arises from the fact that

$$\frac{h}{N_{\partial\Omega_e}}\alpha_{min}\|\theta\|_{l_2(\partial\Omega_e)}^2 \equiv \left|E_{\partial\Omega_e\rightarrow\Omega_e}(\theta)\right|_{a,\Omega_e}^2 \quad \forall\theta\in V_{\partial\Omega_e}.$$

Cf. **Heinlein and Smetana (Preprint: arXiv:2207.05559)**.



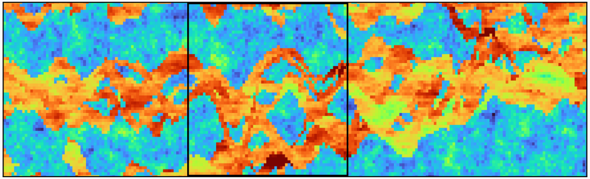


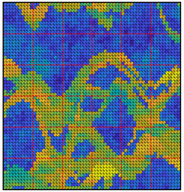
yellow: $\alpha = 10^6$ **blue:** $\alpha = 1$

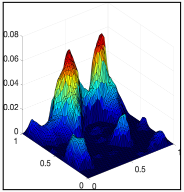
V_0	variant	TOL_{DIR}	TOL_{TR}	TOL_{POD}	dim V_0	κ	# its.
V_{GDSW}	-	-	-	-	33	$2.7 \cdot 10^5$	118
V_{AGDSW}	-	$1.0 \cdot 10^{-2}$			57	7.4	24
$V_{\mathrm{DIR\&TR}}$	$a_{\Omega_e}(\cdot,\cdot)$	$1.0 \cdot 10^{-3}$	$1.0 \cdot 10^{1}$	$1.0 \cdot 10^{-5}$	57	7.2	24
$V_{\mathrm{DIR\&TR}}$	$(\cdot,\cdot)_{l_2(\partial\Omega_e)}$	$1.0 \cdot 10^{-3}$	$1.0 \cdot 10^{1}$	$1.0 \cdot 10^{-5}$	57	7.2	24

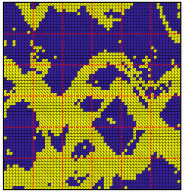
$\rightarrow$ In order to get rid of potential **linear dependencies** between the V_{DIR} and V_{TR} spaces, apply a **proper orthogonal decomposition (POD)** with threshold TOL_{POD} for each edge.

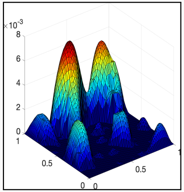
Layer 70 from model 2 of the SPE10 benchmark; cf. Christie and Blunt (2001)











V_0	variant	TOL_{DIR}	TOL_{TR}	TOL_{POD}	dim V_0	κ	# its.
V_{GDSW}	-	-	-	-	85	$2.0 \cdot 10^5$	57
V_{AGDSW}	-		$1.0 \cdot 10^{-2}$		93	19.3	38
$V_{\text{DIR\&TR}}$	$a_{\Omega_e}(\cdot,\cdot)$	$1.0 \cdot 10^{-3}$	$1.0 \cdot 10^{5}$	$1.0 \cdot 10^{-5}$	90	19.4	39
$V_{\text{DIR\&TR}}$	$(\cdot,\cdot)_{l_2(\partial\Omega_e)}$	$1.0 \cdot 10^{-3}$	$1.0 \cdot 10^{5}$	$1.0 \cdot 10^{-5}$	147	9.6	31
Original coefficient $\alpha_{\max} \approx 10^4, \alpha_{\min} \approx 10^{-2}$ (without thresholding)							
V_{GDSW}	-	-	-	-	85	20.6	42

Extension-Based Coarse Spaces in Nonlinear Schwarz Preconditioning

Linear & Nonlinear Preconditioning

Let us consider the nonlinear problem arising from the discretization of a partial differential equation

$$\boldsymbol{F}(\boldsymbol{u}) = 0.$$

We solve the problem using a **Newton-Krylov approach**, i.e., we solve a sequence of linearized problems using a Krylov subspace method:

$$\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right)\Delta\boldsymbol{u}^{(k+1)} = \boldsymbol{F}\left(\boldsymbol{u}^{(k)}\right).$$

Linear preconditioning

In linear preconditioning, we **improve the convergence speed of the linear solver** by constructing a **linear operator** M^{-1} and solve linear systems

$$\boldsymbol{M}^{-1}\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right)\Delta\boldsymbol{u}^{(k+1)} = \boldsymbol{M}^{-1}\boldsymbol{F}(\boldsymbol{u}^{(k)}).$$

Goal:

- $\kappa\left(\boldsymbol{M}^{-1}\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right)\right) \approx 1.$

$\Rightarrow \boldsymbol{M}^{-1}\boldsymbol{DF}\left(\boldsymbol{u}^{(k)}\right) \approx \boldsymbol{I}.$

Nonlinear preconditioning

In nonlinear preconditioning, we **improve the convergence speed of the nonlinear solver** by constructing a **nonlinear operator** G and solve the nonlinear system

$$(\boldsymbol{G} \circ \boldsymbol{F})(\boldsymbol{u}) = 0.$$

Goals:

- $\boldsymbol{G} \circ \boldsymbol{F}$ almost linear.
- Additionally: $\kappa(\boldsymbol{D}(\boldsymbol{G} \circ \boldsymbol{F})(\boldsymbol{u})) \approx 1.$

Nonlinear Domain Decomposition Methods

Additive nonlinear left preconditioners (based on Schwarz methods)

ASPIN/ASPEN: Cai, Keyes 2002; Cai, Keyes, Marcinkowski (2002); Hwang, Cai (2005, 2007); Groß, Krause (2010, 2013)

RASPEN: Dolean, Gander, Kherijii, Kwok, Masson (2016)

MSPIN: Keyes, Liu, (2015, 2016, 2021); Liu, Wei, Keyes (2017)

Two-Level nonlinear Schwarz: Heinlein, Lanser (2020); Heinlein, Lanser, Klawonn (accepted 2022)

Nonlinear right preconditioners (based on either FETI or BDDC)

Nonlinear FETI-DP/BDDC: Klawonn, Lanser, Rheinbach (2012, 2013, 2014, 2015, 2016, 2018); Klawonn, Lanser, Rheinbach, Uran (2017, 2018)
Nonlinear Elimination: Hwang, Lin, Cai (2010); Cai, Li (2011); Wang, Su, Cai (2015); Hwang, Su, Cai (2016); Gong, Cai (2018); Luo, Shiu, Chen, Cai (2019); Gong, Cai (2019)
Nonlinear Neumann-Neumann: Bordeu, Boucard, Gosselet (2009)
Nonlinear FETI-1: Pebrel, Rey, Gosselet (2008); Negrello, Gosselet, Rey (2021)

Other DD work reversing linearization and decomposition: Ganis, Juntunen, Pencheva, Wheeler, Yotov (2014); Ganis, Kumar, Pencheva, Wheeler, Yotov (2014)
Early nonlinear DD work: Cai, Dryja (1994); Dryja, Hackbusch (1997)

ASPEN & ASPIN

Our approach is based on the nonlinear one-level Schwarz methods **ASPEN (Additive Schwarz Preconditioned Exact Newton)** and **ASPIN (Additive Schwarz Preconditioned Inexact Newton)** introduced in Cai and Keyes (2002). The nonlinear finite element problem

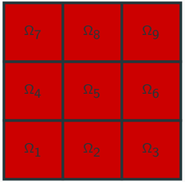
$$\boldsymbol{F}(\boldsymbol{u}) = 0 \quad \text{with } \boldsymbol{F} : V \to V$$

is reformulated to

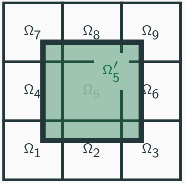
$$\mathcal{F}(\boldsymbol{u}) = \boldsymbol{G}(\boldsymbol{F}(\boldsymbol{u})) = 0.$$

The **nonlinear left-preconditioner $\boldsymbol{G}$ is only given implicitly** by **solving the nonlinear problem locally** on each of the (overlapping) subdomains. Roughly,

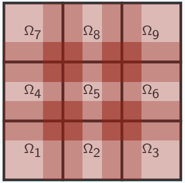
$$\boldsymbol{F}_i(\boldsymbol{u} - \underbrace{\boldsymbol{C}_i(\boldsymbol{u})}_{\text{local correction}}), \; i = 1, \ldots, N.$$



$\boldsymbol{F}(\boldsymbol{u}) = 0$



$\boldsymbol{F}_i(\boldsymbol{u} - \boldsymbol{C}_i(\boldsymbol{u})) = 0$



$\mathcal{F}(\boldsymbol{u}) = 0$

Nonlinear One-Level Schwarz Preconditioners

RASPEN (Dolean et al. (2016))

Local corrections $\boldsymbol{T}_i(\boldsymbol{u})$:

$$\boldsymbol{R}_i \boldsymbol{F}(\boldsymbol{u} - \boldsymbol{P}_i \boldsymbol{T}_i(\boldsymbol{u})) = 0, \; i = 1, \ldots, N, \text{ with}$$

$$\text{restrictions} \quad \boldsymbol{R}_i \quad : V \to V_i,$$

$$\text{prolongations} \quad \boldsymbol{P}_i, \widetilde{\boldsymbol{P}}_i : V_i \to V.$$

Nonlinear RASPEN problem:

$$\mathcal{F}_{RA}(\boldsymbol{u}) := \sum_{i=1}^N \widetilde{\boldsymbol{P}}_i \boldsymbol{T}_i(\boldsymbol{u}) = 0$$

We solve $\mathcal{F}_{RA}(\boldsymbol{u}) = 0$ using Newton's method with $\boldsymbol{u}_i = \boldsymbol{u} - \boldsymbol{P}_i \boldsymbol{T}_i(\boldsymbol{u})$. The Jacobian writes

$$\boldsymbol{D}\mathcal{F}_{RA}(\boldsymbol{u}) = \sum_{i=1}^N \underbrace{\widetilde{\boldsymbol{P}}_i \left(\boldsymbol{R}_i \boldsymbol{DF}(\boldsymbol{u}_i) P_i\right)^{-1} \boldsymbol{R}_i \; \boldsymbol{DF}(\boldsymbol{u}_i)}_{\substack{\text{local Schwarz operators} \\ \text{(preconditioned operators)}}}$$

- $\sum_{i=1}^N \widetilde{\boldsymbol{P}}_i \boldsymbol{R}_i = \boldsymbol{I}$
- **Reduced communication** & (often) **better conv.**

Results

p-Laplacian model problem

$$\begin{aligned} -\alpha \Delta_p u &= 1 && \text{in } \Omega, \\ u &= 0 && \text{on } \partial\Omega. \end{aligned}$$

with $\alpha \Delta_p u := \mathrm{div}(\alpha |\nabla u|^{p-2} \nabla u)$.

$p = 4$; $H/h = 16$; overlap $\delta = 1$				
		nonlin.		**lin.**
N	**solver**	**outer it.**	**inner it. (avg.)**	**GMRES it. (sum)**
9	NK-RAS	18	-	272
	RASPEN	5	25.2	89
25	NK-RAS	19	-	488
	RASPEN	6	28.3	172
49	NK-RAS	20	-	691
	RASPEN	6	27.3	232

$\Rightarrow$ **Improved nonlinear convergence**, but **no scalability** in the linear iterations.

Nonlinear Two-Level Schwarz Preconditioners

Two-level (R)ASPEN (Heinlein & Lanser (2020))

Local/Coarse corrections $T_i(\boldsymbol{u})$:

$$R_i F(u - P_i T_i(\boldsymbol{u})) = 0, \; i = 0, 1, \ldots, N, \text{ with}$$

$$\text{restrictions} \quad \boldsymbol{R}_i : V \to V_i,$$

$$\text{prolongations} \quad \boldsymbol{P}_i : V_i \to V.$$

Nonlinear two-level ASPEN problem:

$$\mathcal{F}_A(\boldsymbol{u}) := \boldsymbol{P}_0 \boldsymbol{T}_0(\boldsymbol{u}) + \sum_{i=1}^{N} \boldsymbol{P}_i \boldsymbol{T}_i(\boldsymbol{u}) = 0$$

We solve $\mathcal{F}_A(\boldsymbol{u}) = 0$ using Newton's method with $\boldsymbol{u}_i = \boldsymbol{u} - \boldsymbol{P}_i \boldsymbol{T}_i(\boldsymbol{u})$. The Jacobian writes

$$D\mathcal{F}_{RA}(\boldsymbol{u}) = \overbrace{P_0 \left(\boldsymbol{R}_0 \boldsymbol{DF}(\boldsymbol{u}_0) \boldsymbol{P}_0\right)^{-1} R_0 \; DF(u_0)}^{\text{coarse Schwarz operator}} + \sum_{i=1}^{N} \underbrace{P_i \left(\boldsymbol{R}_i \boldsymbol{DF}(\boldsymbol{u}_i) \boldsymbol{P}_i\right)^{-1} \boldsymbol{R}_i \; \boldsymbol{DF}(\boldsymbol{u}_i)}_{\text{local Schwarz operators}}$$

Results for p-Laplace

1-lvl	One-level RASPEN
2-lvl A	Two-level RASPEN with **additively coupled coarse level**
2-lvl M	Two-level RASPEN with **multiplicatively coupled coarse level**

$p = 4$; $H/h = 16$; overlap $\delta = 1$

N	RASPEN solver	nonlin. outer it.	nonlin. inner it. (avg.)	nonlin. coarse it.	lin. GMRES it. (sum)
9	1-lvl	5	25.2	-	89
	2-lvl A	6	33.4	27	93
	2-lvl M	4	17.1	29	52
49	1-lvl	6	27.3	-	**232**
	2-lvl A	6	29.2	28	**137**
	2-lvl M	4	12.6	29	**80**

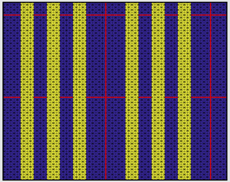
$\Rightarrow$ **Improved nonlinear convergence and scalability**.

Problem configuration (Heinlein, Klawonn, Lanser (accepted 2022))

p-Laplacian problem with $p = 4$ and a **binary coefficient** α: find u such that

$$\begin{aligned} -\alpha \Delta_p u &= 1 && \text{in } \Omega, \\ u &= 0 && \text{on } \partial\Omega. \end{aligned}$$

Domain decomposition into 6×6 subdomains with $H/h = 32$ and overlap $1h$.



yellow: $\alpha = 10^3$ **blue:** $\alpha = 1$

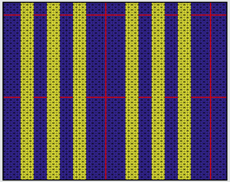
size cp	method	coarse space	outer it.	local it. (avg.)	coarse it.	GMRES it. (sum)
no globalization						
145	H1-RASPEN	AGDSW	5	27.0	35	77
25	H1-RASPEN	MsFEM-D	>20	-	-	-
25	H1-RASPEN	MsFEM-E	>20	-	-	-
145	NK-RAS	AGDSW	>20	-	-	-
inexact Newton backtracking (INB); cf. Eisenstat and Walker (1994)						
145	H1-RASPEN	AGDSW	5	24.8	21	77
25	H1-RASPEN	MsFEM-D	**15**	**75.8**	**62**	**645**
25	H1-RASPEN	MsFEM-E	**18**	**83.9**	**75**	**852**
145	NK-RAS	AGDSW	**13**	-	-	207

Problem configuration (Heinlein, Klawonn, Lanser (accepted 2022))

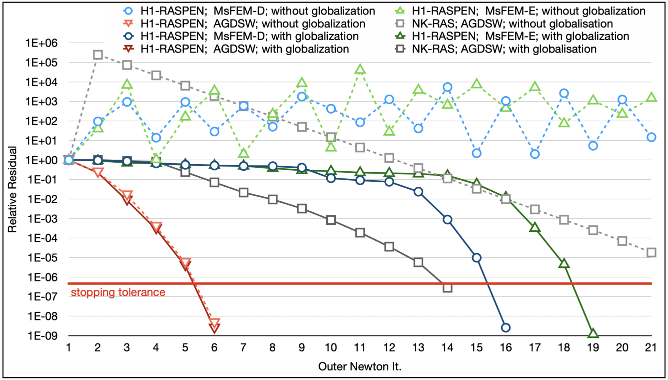
p-Laplacian problem with $p = 4$ and a **binary coefficient** α: find u such that

$$\begin{aligned} -\alpha\Delta_p u &= 1 && \text{in } \Omega, \\ u &= 0 && \text{on } \partial\Omega. \end{aligned}$$

Domain decomposition into 6×6 subdomains with $H/h = 32$ and overlap $1h$.



yellow: $\alpha = 10^3$ **blue:** $\alpha = 1$



Thank you for your attention!

Summary

- Extension-based coarse spaces are a powerful **framework for robust and scalable**
  - **algebraic**,
  - **multilevel**,
  - **adaptive**, and
  - **nonlinear**

  Schwarz domain decomposition methods.

Acknowledgements

- **Financial support:** DFG (KL2094/3-1, RH122/4-1)
- **Computing ressources:** Cori (NERSC), JUQUEEN (JSC), magnitUDE (UDE), Piz Daint (CSCS)

Related Talks

- Alexander Heinlein
  Robust Coarse Spaces for Nonlinear Schwarz Methods
  MS16, Wednesday, July 27, 10.45–11.15, HALL 4 (C219)
- Jascha Knepper
  Low-dimensional adaptive coarse spaces for Schwarz methods and multiscale elliptic problems
  MS4, Thursday, July 28, 16.30–17.00, HALL 5 (C221)
- Kathrin Smetana
  A fully algebraic and robust two-level overlapping Schwarz method based on optimal local approximation spaces
  MS04, Thursday, July 28, 11.30–12.00, HALL 5 (C221)
- Olof Widlund
  Adaptive overlapping Schwarz algorithms for linear elasticity
  MS11, Tuesday, July 26, 10.30–11.00, HALL 6 (C223)