Projected Schur Complement Method for Solving Non-Symmetric Saddle-Point Systems (Arising from Fictitious Domain Approaches)

Jaroslav Haslinger, Charles University, Prague

Tomáš Kozubek, VŠB–TU Ostrava

Radek Kučera, VŠB–TU Ostrava

SNA'07, Januar 22, Ostrava

OUTLINE

Preliminaries on saddle-point systems

Projected Schur complement method

Projected BiCGSTAB algorithm

Hierarchical multigrid

Poisson-like solver for singular matrices

Numerical experiments

OUTLINE

Preliminaries on saddle-point systems

Projected Schur complement method

Projected BiCGSTAB algorithm

Hierarchical multigrid

Poisson-like solver for singular matrices

Numerical experiments

Formulation: Non-symmetric sadle-point system

$$\begin{pmatrix} \mathbf{A} & \mathbf{B}_1^\top \\ \mathbf{B}_2 & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix}$$

Assumptions

$\mathbf{A}$ . . . non-symmetric $(n \times n)$–matrix
. . . singular with $p = \dim \mathit{Ker}\, \mathbf{A}$ as $V_h \approx H^1_{per}(\Omega)$
. . . actions of $A^\dagger$ can be computed by Poisson-like solver based on FFT

$\mathbf{B}_1,\ \mathbf{B}_2$ . . . full rank $(m \times n)$–matrices, $m \ll n$
. . . sparse, their actions are cheap
. . . $\mathbf{B}_1 \neq \mathbf{B}_2$ as $\Gamma \neq \gamma$ or as mixed boudary condions are prescribed

Algorithms based on Schur complement reductions

Case 1: symmetric non-singular
Case 2: non-symmetric non-singular
Case 3: symmetric singular
Case 4: non-symmetric singular

Case 1: symmetric non-singular (positive definite $\mathbf{A}$)

$$\begin{pmatrix} \mathbf{A} & \mathbf{B}^\top \\ \mathbf{B} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix} \quad \begin{array}{ll} \Longrightarrow & \mathbf{u} = \mathbf{A}^{-1}(\mathbf{f} - \mathbf{B}^\top \boldsymbol{\lambda}) \\ \Longrightarrow & \mathbf{B}\mathbf{A}^{-1}\mathbf{B}^\top \boldsymbol{\lambda} = \mathbf{B}\mathbf{A}^{-1}\mathbf{f} - \mathbf{g} \end{array}$$

Algorithm

Step 1: Assemble $\mathbf{d} := \mathbf{B}\mathbf{A}^{-1}\mathbf{f} - \mathbf{g}$.

Step 2: Solve $-\mathbf{S}\boldsymbol{\lambda} = \mathbf{d}$ by the CGM with $\mathbf{S} := -\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^\top$.

Step 3: Assemble $\mathbf{u} = \mathbf{A}^{-1}(\mathbf{f} - \mathbf{B}^\top \boldsymbol{\lambda})$.

Matrix-vector products $-\mathbf{S}\boldsymbol{\mu}$ are computed by:

$$-\mathbf{S}\boldsymbol{\mu} := \left(\mathbf{B}\left(\mathbf{A}^{-1}\left(\mathbf{B}^\top \boldsymbol{\mu}\right)\right)\right)$$

Algorithm requires $\mathcal{O}((m+2)n_{A^\dagger})$ flops in the worst case.

Case 2: non-symmetric non-singular

$$\begin{pmatrix} \mathbf{A} & \mathbf{B}_1^\top \\ \mathbf{B}_2 & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix} \quad \begin{array}{ll} \Longrightarrow & \mathbf{u} = \mathbf{A}^{-1}(\mathbf{f} - \mathbf{B}_1^\top \boldsymbol{\lambda}) \\ \Longrightarrow & \mathbf{B}_2\mathbf{A}^{-1}\mathbf{B}_1^\top \boldsymbol{\lambda} = \mathbf{B}_2\mathbf{A}^{-1}\mathbf{f} - \mathbf{g} \end{array}$$

Algorithm is analogous.

- the Schur complement $\mathbf{S} := -\mathbf{B}_2\mathbf{A}^{-1}\mathbf{B}_1^\top$ is non-symmetric
- a Krylov method for non-symmetric matrices is required (GMRES, BiCG, ...)

$$\mathcal{A} := \begin{pmatrix} \mathbf{A} & \mathbf{B}_1^\top \\ \mathbf{B}_2 & \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{B}_2\mathbf{A}^{-1} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{A} & \mathbf{B}_1^\top \\ \mathbf{0} & \mathbf{S} \end{pmatrix}$$

Theorem. Let $\mathbf{A}$ be non-singular. Then $\mathcal{A}$ is invertible iff $\mathbf{S}$ is invertible.

Case 3: symmetric singular (positive semidefinite $\mathbf{A}$, FETI)

- a generalized inverse $\mathbf{A}^\dagger$ satisfying $\mathbf{A} = \mathbf{A}\mathbf{A}^\dagger\mathbf{A}$
- an $(n \times p)$–matrix $\mathbf{N}$ whose columns span *Ker* $\mathbf{A}$

$$\mathbf{A}\mathbf{u} + \mathbf{B}^\top\boldsymbol{\lambda} = \mathbf{f} \quad \Longleftrightarrow \quad \mathbf{f} - \mathbf{B}^\top\boldsymbol{\lambda} \in \mathit{Im}\,\mathbf{A} \perp \mathit{Ker}\,\mathbf{A}$$

$$\Updownarrow \qquad\qquad \Updownarrow$$

$$\mathbf{u} = \mathbf{A}^\dagger(\mathbf{f} - \mathbf{B}^\top\boldsymbol{\lambda}) + \mathbf{N}\boldsymbol{\alpha} \qquad\qquad \mathbf{N}^\top(\mathbf{f} - \mathbf{B}^\top\boldsymbol{\lambda}) = \mathbf{0}$$

$$\&$$

$$\mathbf{B}\mathbf{u} = \mathbf{g}$$

$$\Downarrow$$

$$\mathbf{B}\mathbf{A}^\dagger\mathbf{B}^\top\boldsymbol{\lambda} - \mathbf{B}\mathbf{N}\boldsymbol{\alpha} = \mathbf{B}\mathbf{A}^\dagger\mathbf{f} - \mathbf{g}$$

Reduced system:

$$\begin{pmatrix} \mathbf{B}\mathbf{A}^\dagger\mathbf{B}^\top & -\mathbf{B}\mathbf{N} \\ -\mathbf{N}^\top\mathbf{B}^\top & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} \mathbf{B}\mathbf{A}^\dagger\mathbf{f} - \mathbf{g} \\ -\mathbf{N}^\top\mathbf{f} \end{pmatrix}$$

an orthogonal projector + the projected CG

Case 4: non-symmetric and singular

- a generalized inverse $\mathbf{A}^\dagger$ satisfying $\mathbf{A} = \mathbf{A}\mathbf{A}^\dagger\mathbf{A}$
- an $(n \times p)$–matrices $\mathbf{N}$, $\mathbf{M}$ whose columns span *Ker* $\mathbf{A}$, *Ker* $\mathbf{A}^\top$, respectively

$$\mathbf{A}\mathbf{u} + \mathbf{B}_1^\top \boldsymbol{\lambda} = \mathbf{f} \qquad \Longleftrightarrow \qquad \mathbf{f} - \mathbf{B}_1^\top \boldsymbol{\lambda} \in \mathit{Im}\,\mathbf{A} \perp \mathit{Ker}\,\mathbf{A}^\top$$

$$\Updownarrow \qquad\qquad\qquad \Updownarrow$$

$$\mathbf{u} = \mathbf{A}^\dagger(\mathbf{f} - \mathbf{B}_1^\top \boldsymbol{\lambda}) + \mathbf{N}\boldsymbol{\alpha} \qquad\qquad \mathbf{M}^\top(\mathbf{f} - \mathbf{B}_1^\top \boldsymbol{\lambda}) = \mathbf{0}$$

$$\&$$

$$\mathbf{B}_2\mathbf{u} = \mathbf{g}$$

$$\Downarrow$$

$$\mathbf{B}_2\mathbf{A}^\dagger\mathbf{B}_1^\top\boldsymbol{\lambda} - \mathbf{B}_2\mathbf{N}\boldsymbol{\alpha} = \mathbf{B}_2\mathbf{A}^\dagger\mathbf{f} - \mathbf{g}$$

Reduced system:

$$\begin{pmatrix} \mathbf{B}_2\mathbf{A}^\dagger\mathbf{B}_1^\top & -\mathbf{B}_2\mathbf{N} \\ -\mathbf{M}^\top\mathbf{B}_1^\top & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} \mathbf{B}_2\mathbf{A}^\dagger\mathbf{f} - \mathbf{g} \\ -\mathbf{M}^\top\mathbf{f} \end{pmatrix}$$

two orthogonal projectors + the projected BiCG

Theorem 1. The saddle-point matrix $\mathcal{A} := \begin{pmatrix} \mathbf{A} & \mathbf{B}_1^\top \\ \mathbf{B}_2 & \mathbf{0} \end{pmatrix}$ is invertible iff

$$\left.\begin{array}{l} \mathbf{B}_1, \mathbf{B}_2 \text{ have full ranks} \\[1em] \mathit{Ker}\,\mathbf{A} \cap \mathit{Ker}\,\mathbf{B}_2 = \{\mathbf{0}\} \\[1em] \mathbf{A}\,\mathit{Ker}\,\mathbf{B}_2 \cap \mathit{Im}\,\mathbf{B}_1^\top = \{\mathbf{0}\} \end{array}\right\} \quad \text{(NSC)}$$

The generalized Schur complement:

$$\mathcal{S} := \begin{pmatrix} -\mathbf{B}_2\mathbf{A}^\dagger\mathbf{B}_1^\top & \mathbf{B}_2\mathbf{N} \\ \mathbf{M}^\top\mathbf{B}_1^\top & \mathbf{0} \end{pmatrix}$$

Theorem 2. The following three statements are equivalent:

- The necessary and sufficient condition (NSC) holds.
- $\mathcal{A}$ is invertible.
- $\mathcal{S}$ is invertible.

OUTLINE

Preliminaries on saddle-point systems

Projected Schur complement method

Projected BiCGSTAB algorithm

Hierarchical multigrid

Poisson-like solver for singular matrices

Numerical experiments

New notation in the reduced system:

$\mathbf{F} = \mathbf{B}_2\mathbf{A}^\dagger\mathbf{B}_1^\top$, $\mathbf{G}_1 = -\mathbf{N}^\top\mathbf{B}_2^\top$, $\mathbf{G}_2 = -\mathbf{M}^\top\mathbf{B}_1^\top$, $\mathbf{d} = \mathbf{B}_2\mathbf{A}^\dagger\mathbf{f} - \mathbf{g}$, $\mathbf{e} = -\mathbf{M}^\top\mathbf{f}$

$$\begin{pmatrix} \mathbf{F} & \mathbf{G}_1^\top \\ \mathbf{G}_2 & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} \mathbf{d} \\ \mathbf{e} \end{pmatrix}$$

Two orthogonal projectors: $\mathbf{P}_1 : \mathbb{R}^m \mapsto \mathit{Ker}\,\mathbf{G}_1, \quad \mathbf{P}_1 = \mathbf{I} - \mathbf{G}_1^\top(\mathbf{G}_1\mathbf{G}_1^\top)^{-1}\mathbf{G}_1$

$\mathbf{P}_2 : \mathbb{R}^m \mapsto \mathit{Ker}\,\mathbf{G}_2, \quad \mathbf{P}_2 = \mathbf{I} - \mathbf{G}_2^\top(\mathbf{G}_2\mathbf{G}_2^\top)^{-1}\mathbf{G}_2$

- $\mathbf{P}_1$ splits the saddle-point structure of the system:

$$\mathbf{P}_1\mathbf{F}\boldsymbol{\lambda} = \mathbf{P}_1\mathbf{d}, \quad \mathbf{G}_2\boldsymbol{\lambda} = \mathbf{e}$$

- $\mathbf{P}_2$ decomposes $\boldsymbol{\lambda} = \boldsymbol{\lambda}_{Im} + \boldsymbol{\lambda}_{Ker}, \quad \boldsymbol{\lambda}_{Im} \in \mathit{Im}\,\mathbf{G}_2^\top, \quad \boldsymbol{\lambda}_{Ker} \in \mathit{Ker}\,\mathbf{G}_2$.

  At first: $\boldsymbol{\lambda}_{Im} = \mathbf{G}_2^\top(\mathbf{G}_2\mathbf{G}_2^\top)^{-1}\mathbf{e}$

  At second: $\mathbf{P}_1\mathbf{F}\boldsymbol{\lambda}_{Ker} = \mathbf{P}_1(\mathbf{d} - \mathbf{F}\boldsymbol{\lambda}_{Im}) \quad \text{on} \quad \mathit{Ker}\,\mathbf{G}_2$

Theorem 3. The linear operator $\mathbf{P}_1\mathbf{F} : Ker\,\mathbf{G}_2 \mapsto Ker\,\mathbf{G_1}$ is invertible.

Proof.

As both null-spaces $Ker\,\mathbf{G}_1$ and $Ker\,\mathbf{G}_2$ have the same dimension, it is enough to prove that $\mathbf{P}_1\mathbf{F}$ is injective.
Let $\boldsymbol{\mu} \in Ker\,\mathbf{G}_2$ be such that $\mathbf{P}_1\mathbf{F}\boldsymbol{\mu} = \mathbf{0}$. Then $\mathbf{F}\boldsymbol{\mu} \in Ker\,\mathbf{P}_1 = Im\,\mathbf{G}_1^\top$ and, therefore, there is $\boldsymbol{\beta} \in R^l$ so that

$$\mathbf{F}\boldsymbol{\mu} = \mathbf{G}_1^\top\boldsymbol{\beta} \quad \text{and} \quad \mathbf{G}_2\boldsymbol{\mu} = \mathbf{0}.$$

Letting $\mathbf{y} = (\boldsymbol{\mu}^\top, -\boldsymbol{\beta}^\top)^\top$, we get $-\mathcal{S}\mathbf{y} = \mathbf{0}$ implying $\boldsymbol{\mu} = \mathbf{0}$ as $\mathcal{S}$ is invertible.

Algorithm PSCM

Step 1.a: Assemble $\mathbf{G}_1 := -\mathbf{N}^\top \mathbf{B}_2^\top$, $\mathbf{G}_2 := -\mathbf{M}^\top \mathbf{B}_1^\top$.

Step 1.b: Assemble $\mathbf{d} := \mathbf{B}_2 \mathbf{A}^\dagger \mathbf{f} - \mathbf{g}$, $\mathbf{e} := -\mathbf{M}^\top \mathbf{f}$.

Step 1.c: Assemble $\mathbf{H}_1 := (\mathbf{G}_1 \mathbf{G}_1^\top)^{-1}$, $\mathbf{H}_2 := (\mathbf{G}_2 \mathbf{G}_2^\top)^{-1}$.

Step 1.d: Assemble $\boldsymbol{\lambda}_{Im} := \mathbf{G}_2^\top \mathbf{H}_2 \mathbf{e}$, $\tilde{\mathbf{d}} := \mathbf{P}_1(\mathbf{d} - \mathbf{F}\boldsymbol{\lambda}_{Im})$.

Step 1.e: Solve $\mathbf{P}_1 \mathbf{F} \boldsymbol{\lambda}_{Ker} = \tilde{\mathbf{d}}$ on $Ker\, \mathbf{G}_2$.

Step 1.f: Assemble $\boldsymbol{\lambda} := \boldsymbol{\lambda}_{Im} + \boldsymbol{\lambda}_{Ker}$.

Step 2: Assemble $\boldsymbol{\alpha} := \mathbf{H}_1(\mathbf{d} - \mathbf{F}\boldsymbol{\lambda})$.

Step 3: Assemble $\mathbf{u} = \mathbf{A}^\dagger(\mathbf{f} - \mathbf{B}_1^\top \boldsymbol{\lambda}) + \mathbf{N}\boldsymbol{\alpha}$.

An iterative projected Krylov subspace method for non-symmetric operators can be used in Step 1.e.

Matrix-vector products $\mathbf{F}\boldsymbol{\mu}$ and $\mathbf{P}_k \boldsymbol{\mu}$, $k = 1, 2$ are computed by:

$$\mathbf{F}\boldsymbol{\mu} := \left(\mathbf{B}_2 \left(\mathbf{A}^\dagger \left(\mathbf{B}_1^\top \boldsymbol{\mu}\right)\right)\right), \quad \mathbf{P}_k \boldsymbol{\mu} := \boldsymbol{\mu} - \left(\mathbf{G}_k^\top \left(\mathbf{H}_k \left(\mathbf{G}_k \boldsymbol{\mu}\right)\right)\right).$$

OUTLINE

Preliminaries on saddle-point systems

Projected Schur complement method

Projected BiCGSTAB algorithm

Hierarchical multigrid

Poisson-like solver for singular matrices

Numerical experiments

$$\mathbf{F}\boldsymbol{\lambda} = \mathbf{d} \quad \text{on} \quad R^m$$

<u>*Algorithm BiCGSTAB*</u> $[\epsilon, \boldsymbol{\lambda}^0, \mathbf{F}, \mathbf{d}] \rightarrow \boldsymbol{\lambda}$

Initialize: $\mathbf{r}^0 := \mathbf{d} - \mathbf{F}\boldsymbol{\lambda}^0$, $\mathbf{p}^0 := \mathbf{r}^0$, $\tilde{\mathbf{r}}^0$ arbitrary, $k := 0$

While $\|\mathbf{r}^k\| > \epsilon$

1° $\quad \tilde{\mathbf{p}}^k := \mathbf{F}\mathbf{p}^k$

2° $\quad \alpha_k := (\mathbf{r}^k)^\top \tilde{\mathbf{r}}^0 / (\tilde{\mathbf{p}}^k)^\top \tilde{\mathbf{r}}^0$

3° $\quad \mathbf{s}^k := \mathbf{r}^k - \alpha_k \tilde{\mathbf{p}}^k$

4° $\quad \tilde{\mathbf{s}}^k := \mathbf{F}\mathbf{s}^k$

5° $\quad \omega_k := (\tilde{\mathbf{s}}^k)^\top \mathbf{s}^k / (\tilde{\mathbf{s}}^k)^\top \tilde{\mathbf{s}}^k$

6° $\quad \boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \alpha_k \mathbf{p}^k + \omega_k \mathbf{s}^k$

7° $\quad \mathbf{r}^{k+1} := \mathbf{s}^k - \omega_k \tilde{\mathbf{s}}^k$

8° $\quad \beta_{k+1} := (\alpha_k / \omega_k)(\mathbf{r}^{k+1})^\top \tilde{\mathbf{r}}^0 / (\mathbf{r}^k)^\top \tilde{\mathbf{r}}^0$

9° $\quad \mathbf{p}^{k+1} := \mathbf{r}^{k+1} + \beta_{k+1}(\mathbf{p}^k - \omega_k \tilde{\mathbf{p}}^k)$

10° $\quad k := k + 1$

end

$\mathbf{P}_1\mathbf{F}\boldsymbol{\lambda} = \tilde{\mathbf{d}}$ on $Ker\,\mathbf{G}_2$

<u>*Algorithm ProjBiCGSTAB*</u> $[\epsilon, \boldsymbol{\lambda}^0, \mathbf{F}, \mathbf{P}_1, \mathbf{P}_2, \tilde{\mathbf{d}}] \rightarrow \boldsymbol{\lambda}$

Initialize: $\boldsymbol{\lambda}^0 \in Ker\,\mathbf{G}_2$, $\mathbf{r}^0 := \tilde{\mathbf{d}} - \mathbf{P}_1\mathbf{F}\boldsymbol{\lambda}^0$, $\mathbf{p}^0 := \mathbf{r}^0$, $\tilde{\mathbf{r}}^0$ arbitrary, $k := 0$

While $\|\mathbf{r}^k\| > \epsilon$

1° $\tilde{\mathbf{p}}^k := \mathbf{P}_1\mathbf{F}\mathbf{p}^k$

2° $\alpha_k := (\mathbf{r}^k)^\top\tilde{\mathbf{r}}^0/(\tilde{\mathbf{p}}^k)^\top\tilde{\mathbf{r}}^0$

3° $\mathbf{s}^k := \mathbf{r}^k - \alpha_k\tilde{\mathbf{p}}^k$

4° $\tilde{\mathbf{s}}^k := \mathbf{P}_1\mathbf{F}\mathbf{s}^k$

5° $\omega_k := (\tilde{\mathbf{s}}^k)^\top\mathbf{s}^k/(\tilde{\mathbf{s}}^k)^\top\tilde{\mathbf{s}}^k$

6° $\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \alpha_k\mathbf{P}_2\mathbf{p}^k + \omega_k\mathbf{P}_2\mathbf{s}^k$

7° $\mathbf{r}^{k+1} := \mathbf{s}^k - \omega_k\tilde{\mathbf{s}}^k$

8° $\beta_{k+1} := (\alpha_k/\omega_k)(\mathbf{r}^{k+1})^\top\tilde{\mathbf{r}}^0/(\mathbf{r}^k)^\top\tilde{\mathbf{r}}^0$

9° $\mathbf{p}^{k+1} := \mathbf{r}^{k+1} + \beta_{k+1}(\mathbf{p}^k - \omega_k\tilde{\mathbf{p}}^k)$

10° $k := k + 1$

end

$$\mathbf{P}_1\mathbf{F}\boldsymbol{\lambda} = \tilde{\mathbf{d}} \quad \text{on} \quad Ker\,\mathbf{G}_2$$

<u>*Algorithm ProjBiCGSTAB*</u> $[\epsilon, \boldsymbol{\lambda}^0, \mathbf{F}, \mathbf{P}_1, \mathbf{P}_2, \tilde{\mathbf{d}}\,] \rightarrow \boldsymbol{\lambda}$

Initialize: $\boldsymbol{\lambda}^0 \in Ker\,\mathbf{G}_2$, $\mathbf{r}^0 := \mathbf{P}_2(\tilde{\mathbf{d}} - \mathbf{P}_1\mathbf{F}\boldsymbol{\lambda}^0)$, $\mathbf{p}^0 := \mathbf{r}^0$, $\tilde{\mathbf{r}}^0$, $k := 0$

While $\|\mathbf{r}^k\| > \epsilon$

1° $\tilde{\mathbf{p}}^k := \mathbf{P}_2\mathbf{P}_1\mathbf{F}\mathbf{p}^k$

2° $\alpha_k := (\mathbf{r}^k)^\top\tilde{\mathbf{r}}^0/(\tilde{\mathbf{p}}^k)^\top\tilde{\mathbf{r}}^0$

3° $\mathbf{s}^k := \mathbf{r}^k - \alpha_k\tilde{\mathbf{p}}^k$

4° $\tilde{\mathbf{s}}^k := \mathbf{P}_2\mathbf{P}_1\mathbf{F}\mathbf{s}^k$

5° $\omega_k := (\tilde{\mathbf{s}}^k)^\top\mathbf{s}^k/(\tilde{\mathbf{s}}^k)^\top\tilde{\mathbf{s}}^k$

6° $\boldsymbol{\lambda}^{k+1} := \boldsymbol{\lambda}^k + \alpha_k\mathbf{p}^k + \omega_k\mathbf{s}^k$

7° $\mathbf{r}^{k+1} := \mathbf{s}^k - \omega_k\tilde{\mathbf{s}}^k$

8° $\beta_{k+1} := (\alpha_k/\omega_k)(\mathbf{r}^{k+1})^\top\tilde{\mathbf{r}}^0/(\mathbf{r}^k)^\top\tilde{\mathbf{r}}^0$

9° $\mathbf{p}^{k+1} := \mathbf{r}^{k+1} + \beta_{k+1}(\mathbf{p}^k - \omega_k\tilde{\mathbf{p}}^k)$

10° $k := k + 1$

end

OUTLINE

Preliminaries on saddle-point systems

Projected Schur complement method

Projected BiCGSTAB algorithm

Hierarchical multigrid

Poisson-like solver for singular matrices

Numerical experiments

Consider a family of nested partitions of the fictitious domain Ω with stepsizes:

$$h_j, \quad 0 \le j \le J$$

- the first iterate is determined by the result from the nearest lower level
- the terminating tolerance ϵ on each level is $\epsilon := \nu h_j^p$

Algorithm: Hierarchical Multigrid Scheme

Initialize: Let $\boldsymbol{\lambda}_{Ker}^{0,(0)} \in Ker\,(\mathbf{G}_2^{(0)})$ be given.

ProjBiCGSTAB$[\nu h_0^p, \boldsymbol{\lambda}_{Ker}^{0,(0)}, \mathbf{F}^{(0)}, \mathbf{P}_1^{(0)}, \mathbf{P}_2^{(0)}, \tilde{\mathbf{d}}^{(0)}] \longrightarrow \boldsymbol{\lambda}_{Ker}(0)$.

For $j = 1, \ldots, J,$

1° prolongate $\boldsymbol{\lambda}_{Ker}^{(j-1)} \longrightarrow \tilde{\boldsymbol{\lambda}}_{Ker}^{0,(j)}$

2° project $\tilde{\boldsymbol{\lambda}}_{Ker}^{0,(j)} \longrightarrow \boldsymbol{\lambda}_{Ker}^{0,(j)} := \mathbf{P}_2^{(j)} \tilde{\boldsymbol{\lambda}}_{Ker}^{0,(j)}$

3° ProjBiCGSTAB$[\nu h_j^p, \boldsymbol{\lambda}_{Ker}^{0,(j)}, \mathbf{F}^{(j)}, \mathbf{P}_1^{(j)}, \mathbf{P}_2^{(j)}, \tilde{\mathbf{d}}^{(j)}] \longrightarrow \boldsymbol{\lambda}_{Ker}^{(j)}$

end

Return: $\boldsymbol{\lambda}_{Ker} := \boldsymbol{\lambda}_{Ker}^{(J)}$.

OUTLINE

Preliminaries on saddle-point systems

Projected Schur complement method

Projected BiCGSTAB algorithm

Hierarchical multigrid

Poisson-like solver for singular matrices

Numerical experiments

Circulant matrices and Fourier transform

$$\mathbf{A} = \begin{pmatrix} a_1 & a_n & \ldots & a_2 \\ a_2 & a_1 & \ldots & a_3 \\ a_3 & a_2 & \ldots & a_4 \\ \vdots & \vdots & \ddots & \vdots \\ a_n & a_{n-1} & \ldots & a_1 \end{pmatrix} = \left(\mathbf{a}, \mathbf{Ta}, \mathbf{T}^2\mathbf{a}, \cdots, \mathbf{T}^{n-1}\mathbf{a}\right)$$

$$\widehat{\mathcal{T}_k f}(\omega) = \int_R f(x-k)e^{-ix\omega}\,dx = e^{-ik\omega}\widehat{f}(\omega)$$

$$\mathbf{XA} = \left(\mathbf{Dx}_0, \mathbf{Dx}_1, \mathbf{Dx}_2, \cdots, \mathbf{Dx}_{n-1}\right) = \mathbf{DX}$$

Let $\mathbf{A}$ be circulant. Then

$$\mathbf{A} = \mathbf{X}^{-1}\mathbf{DX},$$

where $\mathbf{X}$ is the DFT matrix and $\mathbf{D} = diag(\widehat{\mathbf{a}})$, $\widehat{\mathbf{a}} = \mathbf{Xa}$, $\mathbf{a} = \mathbf{A}(:, 1)$.

Multiplying procedure: $\mathbf{A}^{\dagger}\mathbf{v} = \mathbf{X}^{-1}\left(\mathbf{D}^{\dagger}\left(\mathbf{X}\mathbf{v}\right)\right)$ ... Moore-Penrose

$$\left.\begin{array}{ll} 0^\circ & \mathbf{d} := \texttt{fft}(\mathbf{a}) \\ 1^\circ & \mathbf{v} := \texttt{fft}(\mathbf{v}) \\ 2^\circ & \mathbf{v} := \mathbf{v}.*\mathbf{d}^{-1} \\ 3^\circ & \mathbf{A}^{\dagger}\mathbf{v} := \texttt{ifft}(\mathbf{v}) \end{array}\right\} \mathcal{O}(2n\log_2 n)$$

Multiplying procedures: $\mathbf{N}\boldsymbol{\alpha}$, $\mathbf{N}^{\top}\mathbf{v}$ and $\mathbf{M}\boldsymbol{\alpha}$, $\mathbf{M}^{\top}\mathbf{v}$

$$\mathbf{I} - \mathbf{D}\mathbf{D}^{\dagger} = \operatorname{diag}(1,1,1,0,\ldots,0) \implies \mathbf{X}^{-1} = (\mathbf{N},\mathbf{Y})$$

Therefore we can define the operation: $\texttt{ind}(\boldsymbol{\alpha}) = \begin{pmatrix} \boldsymbol{\alpha} \\ \mathbf{0} \end{pmatrix} \in R^n$

$$\begin{array}{ll} 1^\circ & \mathbf{v}_{\boldsymbol{\alpha}} := \texttt{ind}(\boldsymbol{\alpha}) \\ 2^\circ & \mathbf{N}\boldsymbol{\alpha} := \texttt{ifft}(\mathbf{v}_{\boldsymbol{\alpha}}) \end{array} \qquad \left.\begin{array}{ll} 1^\circ & \mathbf{v} := \texttt{ifft}(\mathbf{v}) \\ 2^\circ & \mathbf{N}^{\top}\mathbf{v} := \texttt{ind}^{-1}(\mathbf{v}) \end{array}\right\} \mathcal{O}(n\log_2 n)$$

Kronecker product of matrices: $\mathbf{A}_x \in R^{n_x \times n_x}$, $\mathbf{A}_y \in R^{n_y \times n_y}$

$$\mathbf{A}_x \otimes \mathbf{A}_y = \begin{pmatrix} a^y_{11}\mathbf{A}_x & \dots & a^y_{1n_y}\mathbf{A}_x \\ \vdots & \ddots & \vdots \\ a^y_{n_y 1}\mathbf{A}_x & \dots & a^y_{n_y n_y}\mathbf{A}_x \end{pmatrix}$$

Lemma 1:

$$\begin{aligned} (\mathbf{A}_x \otimes \mathbf{A}_y)(\mathbf{B}_x \otimes \mathbf{B}_y) &= \mathbf{A}_x\mathbf{B}_x \otimes \mathbf{A}_y\mathbf{B}_y \\ (\mathbf{A}_x \otimes \mathbf{A}_y)^\dagger &= \mathbf{A}_x^\dagger \otimes \mathbf{A}_y^\dagger \\ \mathbf{N} &= \mathbf{N}_x \otimes \mathbf{N}_y \end{aligned}$$

Lemma 2: $(\mathbf{A}_x \otimes \mathbf{A}_y)\mathbf{v} = \mathtt{vec}(\mathbf{A}_x \mathbf{V} \mathbf{A}_y^\top)$, where $\mathbf{V} = \mathtt{vec}^{-1}(\mathbf{v})$.

$$\mathbf{V} = (\mathbf{v}_1, \dots, \mathbf{v}_{n_y}) \in R^{n_x \times n_y} \quad \Longleftrightarrow \quad \mathtt{vec}(\mathbf{V}) = \begin{pmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_{n_y} \end{pmatrix} \in R^{n_x n_y}$$

Kronecker product and circulant matrices: Let $\mathbf{A}_x, \mathbf{A}_y$ be circulant, then:

$$\begin{aligned}
\mathbf{A} &= \mathbf{A}_x \otimes \mathbf{I}_y + \mathbf{I}_x \otimes \mathbf{A}_y \\
&= \mathbf{X}_x^{-1}\mathbf{D}_x\mathbf{X}_x \otimes \mathbf{X}_y^{-1}\mathbf{X}_y + \mathbf{X}_x^{-1}\mathbf{X}_x \otimes \mathbf{X}_y^{-1}\mathbf{D}_y\mathbf{X}_y \\
&= (\mathbf{X}_x^{-1} \otimes \mathbf{X}_y^{-1})(\mathbf{D}_x \otimes \mathbf{I}_y + \mathbf{I}_x \otimes \mathbf{D}_y)(\mathbf{X}_x \otimes \mathbf{X}_y) \\
&= \mathbf{X}^{-1}\mathbf{D}\mathbf{X}
\end{aligned}$$

with

$$\begin{aligned}
\mathbf{X} &= \mathbf{X}_x \otimes \mathbf{X}_y && \text{(DFT matrix in 2D)} \\
\mathbf{D} &= \mathbf{D}_x \otimes \mathbf{I}_y + \mathbf{I}_x \otimes \mathbf{D}_y && \text{(diagonal matrix)}
\end{aligned}$$

where $\mathbf{X}_x$, $\mathbf{X}_y$ are the DFT matrices, $\mathbf{D}_x = \mathrm{diag}(\mathbf{X}_x\mathbf{a}_x)$, $\mathbf{D}_y = \mathrm{diag}(\mathbf{X}_y\mathbf{a}_y)$ and $\mathbf{a}_x = \mathbf{A}_x(:,1)$, $\mathbf{a}_y = \mathbf{A}_y(:,1)$, respectively.

Multiplying procedure: $\mathbf{A}^\dagger \mathbf{v} = \mathbf{X}^{-1}\left(\mathbf{D}^\dagger\left(\mathbf{X}\mathbf{v}\right)\right)$

$$
\begin{array}{ll}
0^\circ & \mathbf{d}_x := \mathtt{fft}(\mathbf{a}_x),\ \mathbf{d}_y := \mathtt{fft}(\mathbf{a}_y) \\
 & \mathbf{V} := \mathtt{vec}^{-1}(\mathbf{v}) \\
1^\circ & \mathbf{V} := \mathtt{fft}(\mathbf{V}) \\
2^\circ & \mathbf{V} := \mathtt{fft}(\mathbf{V}^\top)^\top \\
3^\circ & \mathbf{V} := \mathtt{vec}^{-1}(\mathbf{D}^\dagger \mathtt{vec}(\mathbf{V})) \\
4^\circ & \mathbf{V} := \mathtt{ifft}(\mathbf{V}) \\
5^\circ & \mathbf{V} := \mathtt{ifft}(\mathbf{V}^\top)^\top \\
 & \mathbf{A}^\dagger \mathbf{v} := \mathtt{vec}(\mathbf{V})
\end{array}
$$

Number of arithmetic operations :

$$
\mathcal{O}(2n(\log_2 n_x + \log_2 n_y) + n) \approx \mathcal{O}(n \log_2 n), \qquad n = n_x n_y
$$

Multiplying procedures: $\mathbf{N}\boldsymbol{\alpha}$, $\mathbf{N}^\top\mathbf{v}$, $\mathbf{M}\boldsymbol{\alpha}$, $\mathbf{M}^\top\mathbf{v}$ ... analogous

OUTLINE

Preliminaries on saddle-point systems

Projected Schur complement method

Projected BiCGSTAB algorithm

Hierarchical multigrid

Poisson-like solver for singular matrices

Numerical experiments

$$\Omega = (0,1) \times (0,1)$$

$$\omega = \{(x,y) \in \mathbb{R}^2 : (x-0.5)^2 + (y-0.5)^2 = 0.3^2\}$$

$$\begin{cases} -\Delta u = f & \text{in } \omega \\ u = g & \text{on } \gamma \end{cases}$$

where the right hand-sides f, g are chosen appropriately to the exact solution

$$\hat{u}_{ex}(x,y) = 100((x-0.5)^3 - (y-0.5)^3) - x^2.$$

The auxiliary boundary Γ is obtained by shifting γ in the normal direction with

$$\delta = 8h.$$

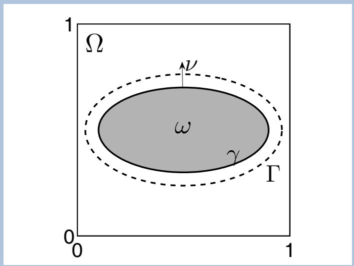

Figure 1: Geometry of ω.

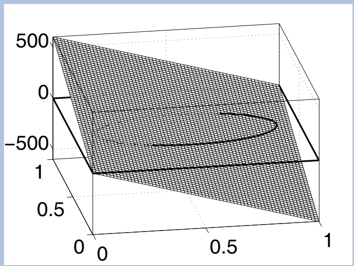

Figure 2: Right hand side f.

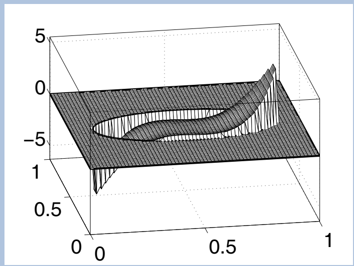

Figure 3: Ex. solution $\hat{u}_{ex|_\omega}$.

Classical FDM

Step h	n/m	Iters.	C.time[s]	$\mathrm{Err}_{L^2(\omega)}$	$\mathrm{Err}_{H^1(\omega)}$	$\mathrm{Err}_{L^2(\partial\omega)}$
1/128	16641/35	**8**	0.14	2.0860e-2	1.9647e+0	6.6516e-2
1/256	66049/62	**9**	0.56	1.1092e-2	1.2884e+0	3.2175e-2
1/512	263169/110	**12**	5.19	5.3989e-3	8.6517e-1	1.5019e-2
1/1024	1050625/198	**20**	33.05	2.7453e-3	6.0511e-1	7.3265e-3
1/2048	4198401/360	**26**	167.00	1.3349e-3	4.4015e-1	3.6245e-3
Convergence rates:				0.995	0.541	1.053

New FDM, PSCM

Step h	n/m	Iters.	C.time[s]	$\mathrm{Err}_{L^2(\omega)}$	$\mathrm{Err}_{H^1(\omega)}$	$\mathrm{Err}_{L^2(\partial\omega)}$
1/128	16641/35	**13**	0.17	2.2550e-4	1.6884e-2	1.1689e-3
1/256	66049/62	**25**	1.33	5.4869e-5	7.7891e-3	2.9342e-4
1/512	263169/110	**40**	14.97	1.4177e-5	4.0160e-3	1.1504e-4
1/1024	1050625/198	**55**	83.56	3.4507e-6	1.9028e-3	2.4769e-5
1/2048	4198401/360	**94**	571.50	9.0638e-7	9.9895e-4	1.2495e-5
Convergence rates:				**1.991**	**1.019**	**1.666**

New FDM, PSCM+Multigrid

Step h	n/m	Iters.	C.time[s]	$\mathrm{Err}_{L^2(\omega)}$	$\mathrm{Err}_{H^1(\omega)}$	$\mathrm{Err}_{L^2(\partial\omega)}$
1/128	16641/34	**11**	0.22	2.4444e-4	1.8988e-2	1.4694e-3
1/256	66049/62	**13**	0.88	5.5030e-5	7.6303e-3	2.5171e-4
1/512	263169/110	**19**	8.41	1.3952e-5	3.8638e-3	8.3976e-5
1/1024	1050625/198	**22**	41.91	3.3209e-6	1.8681e-3	2.5253e-5
1/2048	4198401/360	**31**	243.50	8.5762e-7	9.6771e-4	1.1555e-5
Convergence rates:				**2.036**	**1.062**	**1.730**

New FDM: BiCGM - iterations, discretization error

	Classical FDM		New FDM		New FDM+Multigrid	
Step h	Iters.	$\text{Err}_{H^1(\omega)}$	Iters.	$\text{Err}_{H^1(\omega)}$	Iters.	$\text{Err}_{H^1(\omega)}$
1/128	**8**	1.9647e+0	**13**	1.6884e-2	**11**	1.8988e-2
1/256	**9**	1.2884e+0	**25**	7.7891e-3	**13**	7.6303e-3
1/512	**12**	8.6517e-1	**40**	4.0160e-3	**19**	3.8638e-3
1/1024	**18**	6.0511e-1	**55**	1.9028e-3	**22**	1.8681e-3
1/2048	**25**	4.4015e-1	**94**	9.9895e-4	**31**	9.6771e-4
Conv. rates:		**0.54**		**1.02**		**1.06**

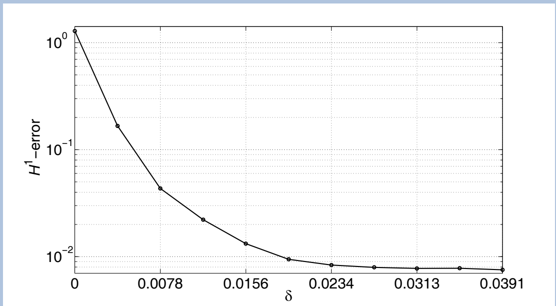

Figure 1: $H^1(\omega)$-error sensitivity on δ.

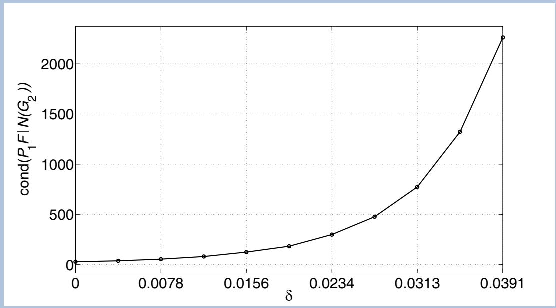

Figure 2: $cond(P_1F|\mathbb{N}(G_2))$ sensitivity on δ.

Conclusions

- The efficient implementation of the FDM with an auxiliary boundary.
- The saddle-point system is solved by the PSCM (non-symmetric analogy of FETI).
- The fast Poisson-like solver for singular matrices are used.
- The fast implementation is ”matrix free”.

References

R. Glowinski, T. Pan, J. Periaux (1994): *A fictitious domain method for Dirichlet problem and applications*. Comput. Meth. Appl. Mech. Engrg. 111, 283–303.

Farhat, C., Mandel, J., Roux, F., X. (1994): *Optimal convergence properties of the FETI domain decomposition method*, Comput. Methods Appl. Mech. Engrg., 115, 365–385.

R. K. (2005): *Complexity of an algorithm for solving saddle-point systems with singular blocks arising in wavelet-Galerkin discretizations*, Appl. Math. 50, 291–308.

J. Haslinger, T. Kozubek, R.K., G. Peichel (2007): *Projected Schur complement method for solving non-symmetric systems arising from a smooth fictitious domain approach*, in preparing.