Student Krylov Day 2015
SIAM Student Chapter at TU Delft
Netherlands
2nd February 2015

Preconditioning of Large-Scale Saddle Point Systems for Coupled Flow Problems

Eberhard Bänsch[1] Peter Benner[2] Jens Saak[2] Martin Stoll[2]
Heiko K. Weichelt[2]

[1]Department of Applied Mathematics III
Friedrich-Alexander-Universität Erlangen-Nürnberg

[2]Research group Computational Methods in Systems and Control Theory
Max Planck Institute for Dynamics of Complex Technical Systems Magdeburg

Overview

Motivation

Model Problems

Flow Models

Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \nu \Delta \vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

Navier-Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \frac{1}{\mathrm{Re}} \Delta \vec{v} + (\vec{v} \cdot \nabla)\vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

- defined on $(0, \infty) \times \Omega$, $\Omega \subset \mathbb{R}^2$ bounded and "smooth enough" $\Gamma = \partial\Omega$
- $+$ boundary and initial conditions

- models describe incompressible, instationary flow
- viscosity $\nu \in \mathbb{R}^+$, (NSE: Reynolds number $\mathrm{Re} = \frac{v_{ch} \cdot d_{ch}}{\nu} \in \mathbb{R}^+$)
- initial boundary value problem with additional algebraic constraints

Motivation

Model Problems

Flow Models

Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \nu \Delta \vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

Navier-Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \frac{1}{\mathrm{Re}} \Delta \vec{v} + (\vec{v} \cdot \nabla)\vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

Motivation

Model Problems

Flow Models

Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \nu \Delta \vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

Navier-Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \frac{1}{\mathrm{Re}} \Delta \vec{v} + (\vec{v} \cdot \nabla)\vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

Diffusion-Convection Models

Concentration Equation

$$\frac{\partial c}{\partial t} - \frac{1}{\mathrm{Re}\,\mathrm{Sc}} \Delta c + (\vec{v} \cdot \nabla) c = 0$$

Temperature Equation

$$\frac{\partial \vartheta}{\partial t} - \frac{1}{\mathrm{Re}\,\mathrm{Pr}} \Delta \vartheta + (\vec{v} \cdot \nabla) \vartheta = 0$$

- defined on $(0, \infty) \times \Omega$, $\Omega \subset \mathbb{R}^2$ bounded and "smooth enough" $\Gamma = \partial\Omega$
- $+$ boundary and initial conditions

- models describe diffusion and convection process
- Schmidt number $\mathrm{Sc} \in \mathbb{R}^+$, Prandtl number $\mathrm{Pr} \in \mathbb{R}^+$

Motivation

Model Problems

Flow Models

Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \nu \Delta \vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

Navier-Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \frac{1}{\mathrm{Re}} \Delta \vec{v} + (\vec{v} \cdot \nabla)\vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

Diffusion-Convection Models

Concentration Equation

$$\frac{\partial c}{\partial t} - \frac{1}{\mathrm{Re}\,\mathrm{Sc}} \Delta c + (\vec{v} \cdot \nabla)c = 0$$

Temperature Equation

$$\frac{\partial \vartheta}{\partial t} - \frac{1}{\mathrm{Re}\,\mathrm{Pr}} \Delta \vartheta + (\vec{v} \cdot \nabla)\vartheta = 0$$

- **Scenario 1:** Feedback stabilization of flow field around stationary trajectory in "von Kármán Vortex Street".

Motivation

Model Problems

Flow Models

Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \nu \Delta \vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

Navier-Stokes Equations

$$\frac{\partial \vec{v}}{\partial t} - \frac{1}{\mathrm{Re}} \Delta \vec{v} + (\vec{v} \cdot \nabla)\vec{v} + \nabla p = \vec{f}$$
$$\operatorname{div} \vec{v} = 0$$

Diffusion-Convection Models

Concentration Equation

$$\frac{\partial c}{\partial t} - \frac{1}{\mathrm{Re}\,\mathrm{Sc}} \Delta c + (\vec{v} \cdot \nabla) c = 0$$

Temperature Equation

$$\frac{\partial \vartheta}{\partial t} - \frac{1}{\mathrm{Re}\,\mathrm{Pr}} \Delta \vartheta + (\vec{v} \cdot \nabla) \vartheta = 0$$

- **Scenario 2:** Feedback stabilization of coupled flow and diffusion-convection field in a reactor model.

Motivation

Basic Ideas of Feedback Stabilization

- **Motivation:**
  - $\hookrightarrow$ Stabilize flow profiles.
  - $\hookrightarrow$ Attenuate external perturbations.
  - $\hookrightarrow$ Influence flow via boundary conditions.

Motivation

Basic Ideas of Feedback Stabilization

- **Motivation:**
  - $\hookrightarrow$ Stabilize flow profiles.
  - $\hookrightarrow$ Attenuate external perturbations.
  - $\hookrightarrow$ Influence flow via boundary conditions.
- Riccati-based feedback stabilization with boundary control input.
  - $\hookrightarrow$ Use *linear quadratic regulator* **(LQR)** approach.
  - $\hookrightarrow$ Influence the model via **boundary control**.
  - $\hookrightarrow$ **Stabilize** the flow around a desired flow profile (stationary trajectory) that is used as **linearization point**.

Motivation

Basic Ideas of Feedback Stabilization

- **Motivation:**
  - ↪ Stabilize flow profiles.
  - ↪ Attenuate external perturbations.
  - ↪ Influence flow via boundary conditions.
- Riccati-based feedback stabilization with boundary control input.
  - ↪ Use *linear quadratic regulator* **(LQR)** approach.
  - ↪ Influence the model via **boundary control**.
  - ↪ **Stabilize** the flow around a desired flow profile (stationary trajectory) that is used as **linearization point**.
- Analytical approach by [Raymond since 2005].
  - ↪ Uses *Leray projector* to project onto the correct subspace.
  - ↪ Extended to finite dimensional controllers [Raymond/Thevenet '10].

Motivation

Basic Ideas of Feedback Stabilization

- **Motivation:**
  - $\hookrightarrow$ Stabilize flow profiles.
  - $\hookrightarrow$ Attenuate external perturbations.
  - $\hookrightarrow$ Influence flow via boundary conditions.
- Riccati-based feedback stabilization with boundary control input.
  - $\hookrightarrow$ Use *linear quadratic regulator* **(LQR)** approach.
  - $\hookrightarrow$ Influence the model via **boundary control**.
  - $\hookrightarrow$ **Stabilize** the flow around a desired flow profile (stationary trajectory) that is used as **linearization point**.
- Analytical approach by [RAYMOND since 2005].
  - $\hookrightarrow$ Uses *Leray projector* to project onto the correct subspace.
  - $\hookrightarrow$ Extended to finite dimensional controllers [RAYMOND/THEVENET '10].
- Ideas for numerical treatment based on [BÄNSCH/BENNER '10].
  - $\hookrightarrow$ Consider **linearized Navier-Stokes equations for 2D**.
  - $\hookrightarrow$ Discrete **projection** idea by [HEINKENSCHLOSS/SORENSEN/SUN '08].
  - $\hookrightarrow$ Use *Newton-ADI* method to compute **optimal control**.

Origin of Saddle Point Systems

Finite Element Discretization

- Applying a standard finite element discretization to the linearized flow/coupled flow problems yields

$$
\begin{aligned}
M\frac{d}{dt}\mathbf{x}(t) &= A\mathbf{x}(t) + G\mathbf{p}(t) + \mathbf{f}(t),\\
\mathbf{0} &= G^T\mathbf{v}(t).
\end{aligned}
$$

Scenario 1	**Scenario 2**
$\mathbf{x}(t) = \mathbf{v}(t)$	$\mathbf{x}(t) = \begin{bmatrix} \mathbf{v}(t) \\ \mathbf{c}(t) \end{bmatrix}$
$A = A_v$	$A = \begin{bmatrix} A_v & 0 \\ -R & A_c \end{bmatrix}$

Origin of Saddle Point Systems

Finite Element Discretization

- Applying a standard finite element discretization to the linearized flow/coupled flow problems yields

$$\begin{aligned} M\frac{d}{dt}\mathbf{x}(t) &= A\mathbf{x}(t) + G\mathbf{p}(t) + \mathbf{f}(t), \\ \mathbf{0} &= G^T\mathbf{v}(t), \\ \mathbf{y}(t) &= C\mathbf{x}(t). \end{aligned}$$

Scenario 1	**Scenario 2**
$\mathbf{x}(t) = \mathbf{v}(t)$	$\mathbf{x}(t) = \begin{bmatrix} \mathbf{v}(t) \\ \mathbf{c}(t) \end{bmatrix}$
$A = A_v$	$A = \begin{bmatrix} A_v & 0 \\ -R & A_c \end{bmatrix}$

Origin of Saddle Point Systems

Finite Element Discretization

- Applying a standard finite element discretization to the linearized flow/coupled flow problems yields

$$
\begin{aligned}
M\frac{d}{dt}\mathbf{x}(t) &= A\mathbf{x}(t) + G\mathbf{p}(t) + B\mathbf{u}(t),\\
\mathbf{0} &= G^T\mathbf{v}(t),\\
\mathbf{y}(t) &= C\mathbf{x}(t).
\end{aligned}
$$

Scenario 1	**Scenario 2**
$\mathbf{x}(t) = \mathbf{v}(t)$	$\mathbf{x}(t) = \begin{bmatrix} \mathbf{v}(t) \\ \mathbf{c}(t) \end{bmatrix}$
$A = A_v$	$A = \begin{bmatrix} A_v & 0 \\ -R & A_c \end{bmatrix}$

Origin of Saddle Point Systems

Finite Element Discretization

- Applying a standard finite element discretization to the linearized flow/coupled flow problems yields

$$\begin{aligned} M\frac{d}{dt}\mathbf{x}(t) &= A\mathbf{x}(t) + G\mathbf{p}(t) + B\mathbf{u}(t), \\ \mathbf{0} &= G^T\mathbf{v}(t), \\ \mathbf{y}(t) &= C\mathbf{x}(t). \end{aligned}$$

Properties

- Differential algebraic system (DAE) of D-index 2 (if $\tilde{G}$ has full rank).
- Matrix pencil:

$$\left(\begin{bmatrix} A & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \right).$$

Scenario 1	Scenario 2
$\mathbf{x}(t) = \mathbf{v}(t)$	$\mathbf{x}(t) = \begin{bmatrix} \mathbf{v}(t) \\ \mathbf{c}(t) \end{bmatrix}$
$A = A_v$	$A = \begin{bmatrix} A_v & 0 \\ -R & A_c \end{bmatrix}$
$\tilde{G} = G$	$\tilde{G} = \begin{bmatrix} G \\ 0 \end{bmatrix}$

Origin of Saddle Point Systems

Finite Element Discretization

- Applying a standard finite element discretization to the linearized flow/coupled flow problems yields

$$\boxed{\begin{aligned} M\frac{d}{dt}\mathbf{x}(t) &= A\mathbf{x}(t) + G\mathbf{p}(t) + B\mathbf{u}(t), \\ \mathbf{0} &= G^T\mathbf{v}(t), \\ \mathbf{y}(t) &= C\mathbf{x}(t). \end{aligned}}$$

Properties

- Differential algebraic system (DAE) of D-index 2 (if $\tilde{G}$ has full rank).
- Matrix pencil:

$$\left(\begin{bmatrix} A & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}\right).$$

- Descriptor system with multiple inputs and outputs (MIMO).

Origin of Saddle Point Systems

Finite Element Discretization

- Applying a standard finite element discretization to the linearized flow/coupled flow problems yields

$$\begin{aligned} M\frac{d}{dt}\mathbf{x}(t) &= A\mathbf{x}(t) + G\mathbf{p}(t) + B\mathbf{u}(t), \\ \mathbf{0} &= G^T\mathbf{v}(t), \\ \mathbf{y}(t) &= C\mathbf{x}(t). \end{aligned}$$

Properties

- Differential algebraic system (DAE) of D-index 2 (if $\tilde{G}$ has full rank).
- Matrix pencil:

$$\left(\begin{bmatrix} A & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \right).$$

- Descriptor system with multiple inputs and outputs (MIMO).
- Index reduction to apply LQR approach [Heinkenschloss/Sorensen/Sun '08].

Origin of Saddle Point Systems

LQR Approach for Projected System

[Benner/Saak/Stoll/W. 12]

Minimize

$$\mathcal{J}(\mathbf{y},\mathbf{u}) = \frac{1}{2}\int_0^\infty \lambda||\mathbf{y}||^2 + ||\mathbf{u}||^2 \,\mathrm{dt},$$

subject to

$$\begin{aligned} \mathcal{M}\frac{d}{dt}\tilde{\mathbf{x}}(t) &= \mathcal{A}\tilde{\mathbf{x}}(t) + \mathcal{B}\mathbf{u}(t), \\ \mathbf{y}(t) &= \mathcal{C}\tilde{\mathbf{x}}(t). \end{aligned} \tag{1}$$

Origin of Saddle Point Systems

LQR Approach for Projected System

[Benner/Saak/Stoll/W. 12]

Minimize

$$\mathcal{J}(\mathbf{y},\mathbf{u}) = \frac{1}{2}\int_0^\infty \lambda||\mathbf{y}||^2 + ||\mathbf{u}||^2 \,\mathrm{dt},$$

subject to

$$\begin{aligned} \mathcal{M}\frac{d}{dt}\tilde{\mathbf{x}}(t) &= \mathcal{A}\tilde{\mathbf{x}}(t) + \mathcal{B}\mathbf{u}(t), \\ \mathbf{y}(t) &= \mathcal{C}\tilde{\mathbf{x}}(t). \end{aligned} \tag{1}$$

Riccati Based Feedback Approach

[e.g., Locatelli '01]

- Optimal control: $\mathbf{u}(t) = -\mathcal{K}\tilde{\mathbf{x}}(t)$.
- Feedback: $\mathcal{K} = \mathcal{B}^T X \mathcal{M}$,

  where X is the solution of the generalized algebraic Riccati equation

$$\mathcal{R}(X) = \mathcal{C}^T\mathcal{C} + \mathcal{A}^T X\mathcal{M} + \mathcal{M}^T X\mathcal{A} - \mathcal{M}^T X\mathcal{B}\mathcal{B}^T X\mathcal{M} = 0.$$

Origin of Saddle Point Systems

Nested Iteration

Compute feedback matrix $\mathcal{K} = \mathcal{B}^T X \mathcal{M}$ with X solves:

$$\mathcal{R}(X) = \mathcal{C}^T\mathcal{C} + \mathcal{A}^T X \mathcal{M} + \mathcal{M}^T X \mathcal{A} - \mathcal{M}^T X \mathcal{B}\mathcal{B}^T X \mathcal{M} = 0$$

Newton Kleinman method

Origin of Saddle Point Systems

Nested Iteration

Compute feedback matrix $\mathcal{K} = \mathcal{B}^T X \mathcal{M}$ with X solves:

$$\mathcal{R}(X) = \mathcal{C}^T\mathcal{C} + \mathcal{A}^T X \mathcal{M} + \mathcal{M}^T X \mathcal{A} - \mathcal{M}^T X \mathcal{B}\mathcal{B}^T X \mathcal{M} = 0$$

Step $\mathbf{m+1}$: solve Lyapunov equation

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)})^T X^{(m+1)} \mathcal{M} + \mathcal{M}^T X^{(m+1)} (\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)}) = -(\mathcal{W}^{(m)})^T \mathcal{W}^{(m)}$$

Newton Kleinman method

low rank ADI method

Origin of Saddle Point Systems

Nested Iteration

Compute feedback matrix $\mathcal{K} = \mathcal{B}^T X \mathcal{M}$ with X solves:

$$\mathcal{R}(X) = \mathcal{C}^T\mathcal{C} + \mathcal{A}^T X \mathcal{M} + \mathcal{M}^T X \mathcal{A} - \mathcal{M}^T X \mathcal{B}\mathcal{B}^T X \mathcal{M} = 0$$

Step $\mathbf{m+1}$: solve Lyapunov equation

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)})^T X^{(m+1)} \mathcal{M} + \mathcal{M}^T X^{(m+1)} (\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)}) = -(\mathcal{W}^{(m)})^T \mathcal{W}^{(m)}$$

Step i: solve the projected linear system

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)} + q_i \mathcal{M})^T \mathcal{V}_i = \mathcal{Y} \qquad (2)$$

Newton Kleinman method

low rank ADI method

Krylov solver

Origin of Saddle Point Systems

Nested Iteration

Compute feedback matrix $\mathcal{K} = \mathcal{B}^T X \mathcal{M}$ with X solves:

$$\mathcal{R}(X) = \mathcal{C}^T\mathcal{C} + \mathcal{A}^T X \mathcal{M} + \mathcal{M}^T X \mathcal{A} - \mathcal{M}^T X \mathcal{B}\mathcal{B}^T X \mathcal{M} = 0$$

Newton Kleinman method

Step $\mathbf{m+1}$: solve Lyapunov equation

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)})^T X^{(m+1)} \mathcal{M} + \mathcal{M}^T X^{(m+1)} (\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)}) = -(\mathcal{W}^{(m)})^T \mathcal{W}^{(m)}$$

low rank ADI method

Step i: solve the projected linear system

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)} + q_i \mathcal{M})^T \mathcal{V}_i = \mathcal{Y} \qquad (2)$$

Krylov solver

Avoid explicit projection using [Heinkenschloss/Sorensen/Sun '08]:

Origin of Saddle Point Systems

Nested Iteration

Compute feedback matrix $\mathcal{K} = \mathcal{B}^T X \mathcal{M}$ with X solves:

$$\mathcal{R}(X) = \mathcal{C}^T\mathcal{C} + \mathcal{A}^T X \mathcal{M} + \mathcal{M}^T X \mathcal{A} - \mathcal{M}^T X \mathcal{B}\mathcal{B}^T X \mathcal{M} = 0$$

Newton Kleinman method

Step $\mathbf{m+1}$: solve Lyapunov equation

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)})^T X^{(m+1)} \mathcal{M} + \mathcal{M}^T X^{(m+1)} (\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)}) = -(\mathcal{W}^{(m)})^T \mathcal{W}^{(m)}$$

low rank ADI method

Step i: solve the projected linear system

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)} + q_i \mathcal{M})^T \mathcal{V}_i = \mathcal{Y} \qquad (2)$$

Krylov solver

Avoid explicit projection using [Heinkenschloss/Sorensen/Sun '08]:
Replace (2) and **solve instead** the saddle point system (SPS)

$$\begin{bmatrix} A^T - (K^{(m)})^T B^T + q_i M^T & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix} \begin{bmatrix} V_i \\ * \end{bmatrix} = \begin{bmatrix} Y \\ 0 \end{bmatrix}$$

for different ADI shifts $q_i \in \mathbb{C}^-$ for a couple of rhs Y.

Origin of Saddle Point Systems

Nested Iteration

Compute feedback matrix $\mathcal{K} = \mathcal{B}^T X \mathcal{M}$ with X solves:

$$\mathcal{R}(X) = \mathcal{C}^T\mathcal{C} + \mathcal{A}^T X \mathcal{M} + \mathcal{M}^T X \mathcal{A} - \mathcal{M}^T X \mathcal{B}\mathcal{B}^T X \mathcal{M} = 0$$

Step $\mathbf{m+1}$: solve Lyapunov equation

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)})^T X^{(m+1)} \mathcal{M} + \mathcal{M}^T X^{(m+1)} (\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)}) = -(\mathcal{W}^{(m)})^T \mathcal{W}^{(m)}$$

Step i: solve the projected linear system

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)} + q_i \mathcal{M})^T \mathcal{V}_i = \mathcal{Y} \qquad (2)$$

Avoid explicit projection using [Heinkenschloss/Sorensen/Sun '08]:
Replace (2) and **solve instead** the saddle point system (SPS)
(using Sherman Morrison Woodbury formula)

$$\begin{bmatrix} A^T - (K^{(m)})^T B^T + q_i M^T & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix} \begin{bmatrix} V_i \\ * \end{bmatrix} = \begin{bmatrix} Y \\ 0 \end{bmatrix}$$

for different ADI shifts $q_i \in \mathbb{C}^-$ for a couple of rhs Y.

Newton Kleinman method

low rank ADI method

Krylov solver

Origin of Saddle Point Systems

Nested Iteration

Compute feedback matrix $\mathcal{K} = \mathcal{B}^T X \mathcal{M}$ with X solves:

$$\mathcal{R}(X) = \mathcal{C}^T \mathcal{C} + \mathcal{A}^T X \mathcal{M} + \mathcal{M}^T X \mathcal{A} - \mathcal{M}^T X \mathcal{B}\mathcal{B}^T X \mathcal{M} = 0$$

Newton Kleinman method

Step m + 1: solve Lyapunov equation

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)})^T X^{(m+1)} \mathcal{M} + \mathcal{M}^T X^{(m+1)} (\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)}) = -(\mathcal{W}^{(m)})^T \mathcal{W}^{(m)}$$

low rank ADI method

Step i: solve the projected linear system

$$(\mathcal{A} - \mathcal{B}\mathcal{K}^{(m)} + q_i \mathcal{M})^T \mathcal{V}_i = \mathcal{Y} \qquad (2)$$

Krylov solver

Avoid explicit projection using [Heinkenschloss/Sorensen/Sun '08]:
Replace (2) and **solve instead** the saddle point system (SPS) (using Sherman Morrison Woodbury formula)

$$\begin{bmatrix} A^T + q_i M^T & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix} \begin{bmatrix} V_i \\ * \end{bmatrix} = \begin{bmatrix} \tilde{Y} \\ 0 \end{bmatrix}$$

for different ADI shifts $q_i \in \mathbb{C}^-$ for a couple of rhs $\tilde{Y}$.

Iterative Solver for Saddle Point Systems

Block-Preconditioner

Preconditioned GMRES [Elman/Silvester/Wathen '05]

- Use iterative methods because size becomes quite large.
- Using GMRES for non-symmetric SPS and a block-preconditioner **P**.
- Ideas for NSE can be adapted to coupled flow problems.

$$\mathbf{F} = \begin{bmatrix} A^T + q_i M^T & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

Block-Preconditioner

Preconditioned GMRES [Elman/Silvester/Wathen '05]

- Use iterative methods because size becomes quite large.
- Using GMRES for non-symmetric SPS and a block-preconditioner **P**.
- Ideas for NSE can be adapted to coupled flow problems.

$$\mathbf{F} = \begin{bmatrix} F & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

Block-Preconditioner

Preconditioned GMRES

[Elman/Silvester/Wathen '05]

- Use iterative methods because size becomes quite large.
- Using GMRES for non-symmetric SPS and a block-preconditioner **P**.
- Ideas for NSE can be adapted to coupled flow problems.

$$\mathbf{F} = \begin{bmatrix} F & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P} = \begin{bmatrix} P_F & 0 \\ \tilde{G}^T & -P_{SC} \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

Block-Preconditioner

Preconditioned GMRES [Elman/Silvester/Wathen '05]

- Use iterative methods because size becomes quite large.
- Using GMRES for non-symmetric SPS and a block-preconditioner **P**.
- Ideas for NSE can be adapted to coupled flow problems.

$$\mathbf{F} = \begin{bmatrix} F & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P} = \begin{bmatrix} P_F & 0 \\ \tilde{G}^T & -P_{SC} \end{bmatrix}$$

- Using algebraic *multigrid* approximation of F for P_F.

  (Yvan Notay, AGMG software, documentation; `http://homepages.ulb.ac.be/~ynotay/AGMG`)

Iterative Solver for Saddle Point Systems

Block-Preconditioner

Preconditioned GMRES [Elman/Silvester/Wathen '05]

- Use iterative methods because size becomes quite large.
- Using GMRES for non-symmetric SPS and a block-preconditioner **P**.
- Ideas for NSE can be adapted to coupled flow problems.

$$\mathbf{F} = \begin{bmatrix} F & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P} = \begin{bmatrix} P_F & 0 \\ \tilde{G}^T & -P_{SC} \end{bmatrix}$$

- Using algebraic *multigrid* approximation of F for P_F.
  (Yvan Notay, AGMG software, documentation; `http://homepages.ulb.ac.be/~ynotay/AGMG`)
- Using *least-squares commutator* (LSC) approach for Schur complement approximation P_{SC}. [Stoll/Wathen '11]

Iterative Solver for Saddle Point Systems

Block-Preconditioner

Preconditioned GMRES [Elman/Silvester/Wathen '05]

- Use iterative methods because size becomes quite large.
- Using GMRES for non-symmetric SPS and a block-preconditioner **P**.
- Ideas for NSE can be adapted to coupled flow problems.

$$\mathbf{F} = \begin{bmatrix} A^T + q_i M^T & \tilde{G} \\ \tilde{G}^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P} = \begin{bmatrix} P_F & 0 \\ \tilde{G}^T & -P_{SC} \end{bmatrix}$$

- Using algebraic *multigrid* approximation of F for P_F.
  (Yvan Notay, AGMG software, documentation; `http://homepages.ulb.ac.be/~ynotay/AGMG`)
- Using *least-squares commutator* (LSC) approach for Schur complement approximation P_{SC}. [Stoll/Wathen '11]
- **Problems:** - Preconditioner changes in every ADI step,
  - SPS has to be solved for a number of right hand sides.

Iterative Solver for Saddle Point Systems

NSE

Preconditioner for Scenario 1

$$\mathbf{F}_{NSE} = \begin{bmatrix} A_v^T + q_i M_v^T & G \\ G^T & 0 \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

NSE

Preconditioner for Scenario 1

$$\mathbf{F}_{NSE} = \begin{bmatrix} F_v & G \\ G^T & 0 \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

NSE

Preconditioner for Scenario 1

$$\mathbf{F}_{NSE} = \begin{bmatrix} F_v & G \\ G^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P}_{NSE} = \begin{bmatrix} P_{F_v} & 0 \\ G^T & -P_{SC} \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

NSE

Preconditioner for Scenario 1

$$\mathbf{F}_{NSE} = \begin{bmatrix} F_v & G \\ G^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P}_{NSE} = \begin{bmatrix} P_{F_v} & 0 \\ G^T & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{NSE}$ solve:

$$\mathbf{P}_{NSE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_b \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_p \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

NSE

Preconditioner for Scenario 1

$$\mathbf{F}_{NSE} = \begin{bmatrix} F_v & G \\ G^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P}_{NSE} = \begin{bmatrix} P_{F_v} & 0 \\ G^T & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{NSE}$ solve:

$$\mathbf{P}_{NSE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_b \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_p \end{bmatrix}$$

Precondition steps

Step I: $\mathbf{x}_v = P_{F_v}^{-1}\mathbf{b}_v$

Step II: $\mathbf{x}_p = P_{SC}^{-1}\left(G^T\mathbf{x}_v - \mathbf{b}_p\right)$

Iterative Solver for Saddle Point Systems

NSE

Preconditioner for Scenario 1

$$\mathbf{F}_{NSE} = \begin{bmatrix} F_v & G \\ G^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P}_{NSE} = \begin{bmatrix} P_{F_v} & 0 \\ G^T & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{NSE}$ solve:

$$\mathbf{P}_{NSE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_b \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_p \end{bmatrix}$$

Precondition steps

Step I: $\mathbf{x}_v = P_{F_v}^{-1} \mathbf{b}_v$

Step II: $\mathbf{x}_p = P_{SC}^{-1} (G^T \mathbf{x}_v - \mathbf{b}_p)$

LSC: $P_{SC}^{-1} \approx M_p^{-1} F_p S_p^{-1}$

Iterative Solver for Saddle Point Systems

NSE

Preconditioner for Scenario 1

$$\mathbf{F}_{NSE} = \begin{bmatrix} F_v & G \\ G^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P}_{NSE} = \begin{bmatrix} P_{F_v} & 0 \\ G^T & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{NSE}$ solve:

$$\mathbf{P}_{NSE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_b \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_p \end{bmatrix}$$

Precondition steps

Step I: $\mathbf{x}_v = P_{F_v}^{-1} \mathbf{b}_v$

Step II: $\mathbf{x}_p = P_{SC}^{-1} (G^T \mathbf{x}_v - \mathbf{b}_p)$

LSC: $P_{SC}^{-1} \approx M_p^{-1} F_p S_p^{-1}$

Implemented Solvers

- algebraic *multigrid* approximation (Yvan Notay, AGMG software + documentation)

Iterative Solver for Saddle Point Systems

NSE

Preconditioner for Scenario 1

$$\mathbf{F}_{NSE} = \begin{bmatrix} F_v & G \\ G^T & 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{P}_{NSE} = \begin{bmatrix} P_{F_v} & 0 \\ G^T & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{NSE}$ solve:

$$\mathbf{P}_{NSE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_b \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_p \end{bmatrix}$$

Precondition steps

Step I: $\mathbf{x}_v = P_{F_v}^{-1}\mathbf{b}_v$

Step II: $\mathbf{x}_p = P_{SC}^{-1}(G^T\mathbf{x}_v - \mathbf{b}_p)$

LSC: $P_{SC}^{-1} \approx M_p^{-1} F_p S_p^{-1}$

Implemented Solvers

- algebraic *multigrid* approximation (Yvan Notay, AGMG software + documentation)
- *Chebyshev semi-iteration* [Stoll/Wathen '11]

Iterative Solver for Saddle Point Systems

NSE+DCE

Preconditioner for Scenario 2

$$\mathbf{F}_{DCE} = \begin{bmatrix} A_v^T + q_i M_v & -R^T & G \\ 0 & A_c^T + q_i M_c & 0 \\ G^T & 0 & 0 \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

NSE+DCE

Preconditioner for Scenario 2

$$\mathbf{F}_{DCE} = \begin{bmatrix} F_v & -R^T & G \\ 0 & F_c & 0 \\ G^T & 0 & 0 \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

NSE+DCE

Preconditioner for Scenario 2

$$\mathbf{F}_{DCE} = \begin{bmatrix} F_v & -R^T & G \\ 0 & F_c & 0 \\ G^T & 0 & 0 \end{bmatrix} \Rightarrow \mathbf{P}_{DCE} = \begin{bmatrix} P_{F_v} & -R^T & 0 \\ 0 & P_{F_c} & 0 \\ G^T & 0 & -P_{SC} \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

NSE+DCE

Preconditioner for Scenario 2

$$\mathbf{F}_{DCE} = \begin{bmatrix} F_v & -R^T & G \\ 0 & F_c & 0 \\ G^T & 0 & 0 \end{bmatrix} \Rightarrow \mathbf{P}_{DCE} = \begin{bmatrix} P_{F_v} & -R^T & 0 \\ 0 & P_{F_c} & 0 \\ G^T & 0 & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{DCE}$ solve:

$$\mathbf{P}_{DCE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_c \\ \mathbf{x}_p \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_c \\ \mathbf{b}_p \end{bmatrix}$$

Iterative Solver for Saddle Point Systems

NSE+DCE

Preconditioner for Scenario 2

$$\mathbf{F}_{DCE} = \begin{bmatrix} F_v & -R^T & G \\ 0 & F_c & 0 \\ G^T & 0 & 0 \end{bmatrix} \Rightarrow \mathbf{P}_{DCE} = \begin{bmatrix} P_{F_v} & -R^T & 0 \\ 0 & P_{F_c} & 0 \\ G^T & 0 & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{DCE}$ solve:

$$\mathbf{P}_{DCE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_c \\ \mathbf{x}_p \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_c \\ \mathbf{b}_p \end{bmatrix}$$

Precondition steps

Step I: $\mathbf{x}_c = P_{F_c}^{-1}\mathbf{b}_c$

Step II: $\mathbf{x}_v = P_{F_v}^{-1}(R^T\mathbf{x}_c + \mathbf{b}_v)$

Step III: $\mathbf{x}_p = P_{SC}^{-1}(G^T\mathbf{x}_v - \mathbf{b}_p)$

Iterative Solver for Saddle Point Systems

NSE+DCE

Preconditioner for Scenario 2

$$\mathbf{F}_{DCE} = \begin{bmatrix} F_v & -R^T & G \\ 0 & F_c & 0 \\ G^T & 0 & 0 \end{bmatrix} \Rightarrow \mathbf{P}_{DCE} = \begin{bmatrix} P_{F_v} & -R^T & 0 \\ 0 & P_{F_c} & 0 \\ G^T & 0 & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{DCE}$ solve:

$$\mathbf{P}_{DCE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_c \\ \mathbf{x}_p \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_c \\ \mathbf{b}_p \end{bmatrix}$$

Precondition steps

Step I: $\mathbf{x}_c = P_{F_c}^{-1}\mathbf{b}_c$

Step II: $\mathbf{x}_v = P_{F_v}^{-1}(R^T\mathbf{x}_c + \mathbf{b}_v)$

Step III: $\mathbf{x}_p = P_{SC}^{-1}(G^T\mathbf{x}_v - \mathbf{b}_p)$

LSC: $P_{SC}^{-1} \approx M_p^{-1}F_pS_p^{-1}$

Iterative Solver for Saddle Point Systems

NSE+DCE

Preconditioner for Scenario 2

$$\mathbf{F}_{DCE} = \begin{bmatrix} F_v & -R^T & G \\ 0 & F_c & 0 \\ G^T & 0 & 0 \end{bmatrix} \Rightarrow \mathbf{P}_{DCE} = \begin{bmatrix} P_{F_v} & -R^T & 0 \\ 0 & P_{F_c} & 0 \\ G^T & 0 & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{DCE}$ solve:

$$\mathbf{P}_{DCE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_c \\ \mathbf{x}_p \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_c \\ \mathbf{b}_p \end{bmatrix}$$

Precondition steps

Step I: $\mathbf{x}_c = P_{F_c}^{-1} \mathbf{b}_c$

Step II: $\mathbf{x}_v = P_{F_v}^{-1} (R^T \mathbf{x}_c + \mathbf{b}_v)$

Step III: $\mathbf{x}_p = P_{SC}^{-1} (G^T \mathbf{x}_v - \mathbf{b}_p)$

LSC: $P_{SC}^{-1} \approx M_p^{-1} F_p S_p^{-1}$

Implemented Solvers

- algebraic *multigrid* approximation (Yvan Notay, AGMG software + documentation)

Iterative Solver for Saddle Point Systems

NSE+DCE

Preconditioner for Scenario 2

$$\mathbf{F}_{DCE} = \begin{bmatrix} F_v & -R^T & G \\ 0 & F_c & 0 \\ G^T & 0 & 0 \end{bmatrix} \Rightarrow \mathbf{P}_{DCE} = \begin{bmatrix} P_{F_v} & -R^T & 0 \\ 0 & P_{F_c} & 0 \\ G^T & 0 & -P_{SC} \end{bmatrix}$$

To apply preconditioner $\mathbf{P}_{DCE}$ solve:

$$\mathbf{P}_{DCE} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_c \\ \mathbf{x}_p \end{bmatrix} = \begin{bmatrix} \mathbf{b}_v \\ \mathbf{b}_c \\ \mathbf{b}_p \end{bmatrix}$$

Precondition steps

Step I: $\mathbf{x}_c = P_{F_c}^{-1}\mathbf{b}_c$

Step II: $\mathbf{x}_v = P_{F_v}^{-1}(R^T\mathbf{x}_c + \mathbf{b}_v)$

Step III: $\mathbf{x}_p = P_{SC}^{-1}(G^T\mathbf{x}_v - \mathbf{b}_p)$

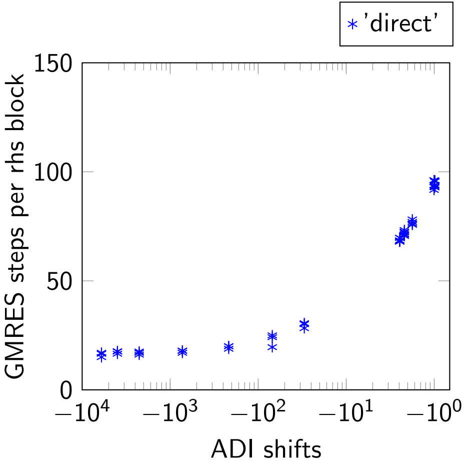
LSC: $P_{SC}^{-1} \approx M_p^{-1} F_p S_p^{-1}$

Implemented Solvers

- algebraic *multigrid* approximation (Yvan Notay, AGMG software + documentation)
- *Chebyshev semi-iteration* [Stoll/Wathen '11]

Iterative Solver for Saddle Point Systems

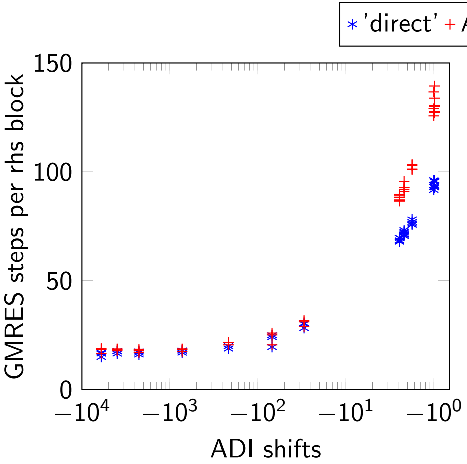
Numerical Examples – NSE



First Newton step with 46 ADI steps, Re $= 100$, $tol_{SPS} = 10^{-10}$, $tol_{ADI} = 10^{-4}$

Iterative Solver for Saddle Point Systems

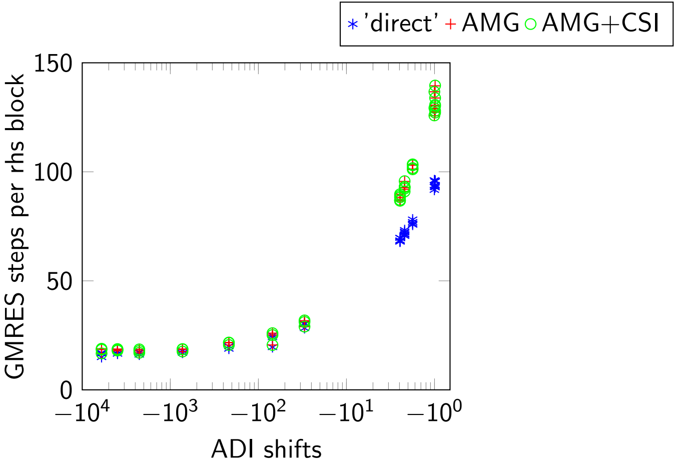
Numerical Examples – NSE



First Newton step with 46 ADI steps, Re = 100, $tol_{SPS} = 10^{-10}$, $tol_{ADI} = 10^{-4}$

Iterative Solver for Saddle Point Systems

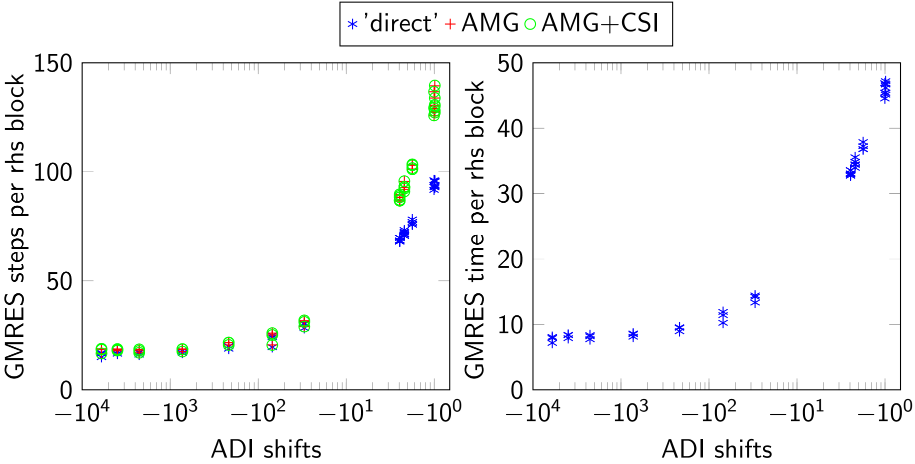
Numerical Examples – NSE



First Newton step with 46 ADI steps, Re = 100, $tol_{SPS} = 10^{-10}$, $tol_{ADI} = 10^{-4}$

Iterative Solver for Saddle Point Systems

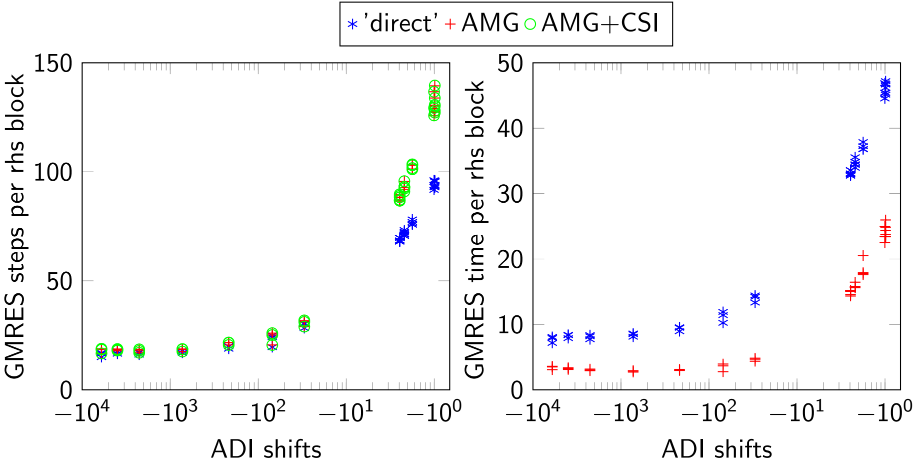
Numerical Examples – NSE



First Newton step with 46 ADI steps, Re = 100,
$tol_{SPS} = 10^{-10}$, $tol_{ADI} = 10^{-4}$

Iterative Solver for Saddle Point Systems

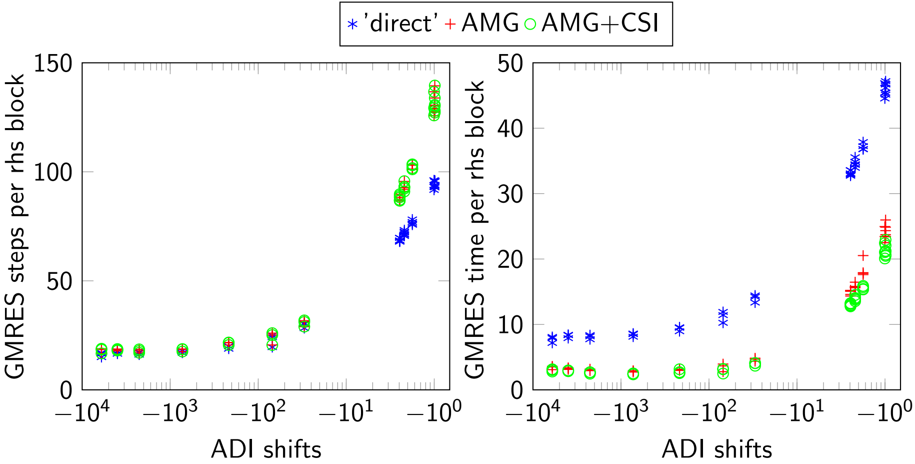
Numerical Examples – NSE



First Newton step with 46 ADI steps, Re $= 100$,
$tol_{SPS} = 10^{-10}$, $tol_{ADI} = 10^{-4}$

Iterative Solver for Saddle Point Systems

Numerical Examples – NSE



First Newton step with 46 ADI steps, $\mathrm{Re} = 100$,
$tol_{SPS} = 10^{-10}$, $tol_{ADI} = 10^{-4}$

Conclusion

Review

- Showed flow problems and coupled systems.
- Applied boundary feedback stabilization approach to DAE.
- Newton-ADI of projected system leads to nested iteration with SPS in the innermost loop.
- Investigated block-preconditioner depending on problem structure.

Conclusion

Review

- Showed flow problems and coupled systems.
- Applied boundary feedback stabilization approach to DAE.
- Newton-ADI of projected system leads to nested iteration with SPS in the innermost loop.
- Investigated block-preconditioner depending on problem structure.

Outlook

- Improve *multigrid* solver for complex ADI shifts q_i.
- Improve Krylov solver via the use of recycling or block techniques.
- Investigated the relations inside the threefold nested iteration. (3 stopping parameters, ADI shifts, parameters: $\mathrm{Re}, \mathrm{Sc}, \mathrm{Pr}$)

Conclusion

Review

- Showed flow problems and coupled systems.
- Applied boundary feedback stabilization approach to DAE.
- Newton-ADI of projected system leads to nested iteration with SPS in the innermost loop.
- Investigated block-preconditioner depending on problem structure.

Outlook

- Improve *multigrid* solver for complex ADI shifts q_i.
- Improve Krylov solver via the use of recycling or block techniques.
- Investigated the relations inside the threefold nested iteration. (3 stopping parameters, ADI shifts, parameters: $\mathrm{Re}, \mathrm{Sc}, \mathrm{Pr}$)

Many thanks for your attention!

Literature

E. Bänsch, P. Benner, J. Saak, and H. K. Weichelt, *Riccati-based boundary feedback stabilization of incompressible Navier-Stokes flow*, Preprint SPP1253-154, DFG-SPP1253, 2013, accepted for publication in SISC.

E. Bänsch, P. Benner, J. Saak, and H. K. Weichelt, *Optimal control-based feedback stabilization of multi-field flow problems*, in Trends in PDE Constrained Optimization, Birkhäuser, 2014, pp. 173–188.

P. Benner, J. Saak, M. Stoll, and H. K. Weichelt, *Efficient solution of large-scale saddle point systems arising in Riccati-based boundary feedback stabilization of incompressible Stokes flow*, SIAM J. Sci. Comput., 35 (2013), pp. S150–S170.

———, *Efficient solvers for large-scale saddle point systems arising in feedback stabilization of multi-field flow problems*, in System Modeling and Optimization, Springer, 2014, pp. 11–20.

H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*, Oxford University Press, 2005.

M. Heinkenschloss, D. C. Sorensen, and K. Sun, *Balanced truncation model reduction for a class of descriptor systems with application to the Oseen equations*, SIAM J. Sci. Comput., 30 (2008), pp. 1038–1063.

J.-P. Raymond, *Feedback boundary stabilization of the two-dimensional Navier-Stokes equations*, SIAM J. Control Optim., 45 (2006), pp. 790–828.

Appendix

Scenario 1: NSE on "von Kármán Vortex Street"

PDE: NSE

Goal: $\vec{z} = \vec{v} - \vec{w} \to 0$

$\rightsquigarrow$ Linearized Navier-Stokes equations:

$$\frac{\partial \vec{z}}{\partial t} - \frac{1}{\mathrm{Re}}\Delta \vec{z} + (\vec{z}\cdot\nabla)\vec{w} + (\vec{w}\cdot\nabla)\vec{z} + \nabla p = 0$$

$$\operatorname{div} \vec{z} = 0$$

defined on $(0,\infty)\times\Omega$

$+$ boundary and initial conditions

LQR

Minimize

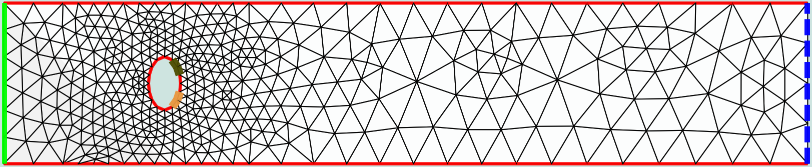
$$\mathcal{J}(\mathbf{y},\mathbf{u}) = \frac{1}{2}\int_0^\infty \lambda||\mathbf{y}||^2 + ||\mathbf{u}||^2 \,\mathrm{dt}$$

s.t.

$$\begin{bmatrix} M_z & 0 \\ 0 & 0 \end{bmatrix} \frac{d}{dt}\begin{bmatrix} \mathbf{z} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} A_z & G \\ G^T & 0 \end{bmatrix}\begin{bmatrix} \mathbf{z} \\ \mathbf{p} \end{bmatrix} + \begin{bmatrix} B_z \\ 0 \end{bmatrix}\mathbf{u}$$

$$\mathbf{y}(t) = C_z\mathbf{z}(t)$$

Domain Ω: *von Kármán vortex street*



Appendix

Scenario 1: NSE – "von Kármán Vortex Street"

NSE

stationary NSE

PDE: NSE

Goal: $\vec{z} = \vec{v} - \vec{w} \to 0$

$\rightsquigarrow$ Linearized Navier-Stokes equations:

$$\frac{\partial \vec{z}}{\partial t} - \frac{1}{\mathrm{Re}}\Delta\vec{z} + (\vec{z}\cdot\nabla)\vec{w} + (\vec{w}\cdot\nabla)\vec{z} + \nabla p = 0$$

$$\operatorname{div}\vec{z} = 0$$

defined on $(0,\infty)\times\Omega$

\+ boundary and initial conditions

LQR

Minimize

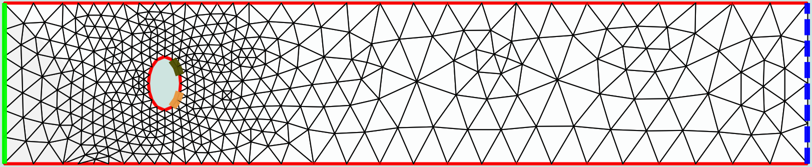
$$\mathcal{J}(\mathbf{y},\mathbf{u}) = \frac{1}{2}\int_0^\infty \lambda||\mathbf{y}||^2 + ||\mathbf{u}||^2\,\mathrm{dt}$$

s.t.

$$\begin{bmatrix} M_z & 0 \\ 0 & 0 \end{bmatrix}\frac{d}{dt}\begin{bmatrix}\mathbf{z}\\ \mathbf{p}\end{bmatrix} = \begin{bmatrix} A_z & G \\ G^T & 0 \end{bmatrix}\begin{bmatrix}\mathbf{z}\\ \mathbf{p}\end{bmatrix} + \begin{bmatrix} B_z \\ 0 \end{bmatrix}\mathbf{u}$$

$$\mathbf{y}(t) = C_z\mathbf{z}(t)$$

Domain Ω: *von Kármán vortex street*



Appendix

Scenario 1: NSE on "von Kármán Vortex Street"

PDE: NSE

Goal: $\vec{z} = \vec{v} - \vec{w} \to 0$

$\rightsquigarrow$ Linearized Navier-Stokes equations:

$$\frac{\partial \vec{z}}{\partial t} - \frac{1}{\mathrm{Re}}\Delta\vec{z} + (\vec{z}\cdot\nabla)\vec{w} + (\vec{w}\cdot\nabla)\vec{z} + \nabla p = 0$$

$$\operatorname{div}\vec{z} = 0$$

defined on $(0,\infty)\times\Omega$

$+$ boundary and initial conditions

LQR

Minimize

$$\mathcal{J}(\mathbf{y},\mathbf{u}) = \frac{1}{2}\int_0^\infty \lambda||\mathbf{y}||^2 + ||\mathbf{u}||^2\ \mathrm{dt}$$

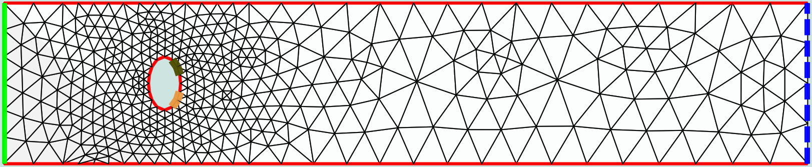
s.t.

$$\mathcal{M}\frac{d}{dt}\tilde{\mathbf{z}} = \mathcal{A}\tilde{\mathbf{z}} + \mathcal{B}\mathbf{u}$$

$$\mathbf{y}(t) = \mathcal{C}\tilde{\mathbf{z}}$$

[Heinkenschloss/Sorensen/Sun '08]

Domain Ω: *von Kármán vortex street*



Appendix

Scenario 2: NSE Coupled with DCE in Reactor Model

PDE: NSE+DCE

Goal: $\vec{z} = \vec{v} - \vec{w} \to 0, c = c^{(\vec{v})} - c^{(\vec{w})} \to 0$

$\rightsquigarrow$ Linearized coupled system:

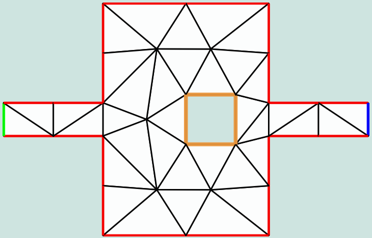
$$\frac{\partial \vec{z}}{\partial t} - \frac{1}{\mathrm{Re}}\Delta \vec{z} + (\vec{z}\cdot\nabla)\vec{w} + (\vec{w}\cdot\nabla)\vec{z} + \nabla p = 0$$

$$\frac{\partial c}{\partial t} - \frac{1}{\mathrm{Re}\,\mathrm{Sc}}\Delta c + (\vec{w}\cdot\nabla)c + (\vec{z}\cdot\nabla)c^{(\vec{w})} = 0$$

$$\operatorname{div}\vec{z} = 0$$

defined on $(0,\infty)\times\Omega$ plus BC,IC

Domain Ω: Reactor Model



LQR

Minimize

$$\mathcal{J}(\mathbf{y},\mathbf{u}) = \frac{1}{2}\int_0^\infty \lambda||\mathbf{y}||^2 + ||\mathbf{u}||^2 \,\mathrm{dt}$$

s.t.

$$\begin{bmatrix} M_\mathbf{z} & 0 & 0 \\ 0 & M_\mathbf{c} & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{d}{dt}\begin{bmatrix} \mathbf{z} \\ \mathbf{c} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} A_\mathbf{z} & 0 & G \\ R & A_\mathbf{c} & 0 \\ G^T & 0 & 0 \end{bmatrix}\begin{bmatrix} \mathbf{z} \\ \mathbf{c} \\ \mathbf{p} \end{bmatrix} + \begin{bmatrix} B_z \\ 0 \\ 0 \end{bmatrix}\mathbf{u}$$

$$\mathbf{y}(t) = C_c\mathbf{c}$$

Appendix

Scenario 2: NSE Coupled with DCE Reactor Model

DCE

stationary DCE

PDE: NSE+DCE

Goal: $\vec{z} = \vec{v} - \vec{w} \to 0$, $c = c^{(\vec{v})} - c^{(\vec{w})} \to 0$

$\rightsquigarrow$ Linearized coupled system:

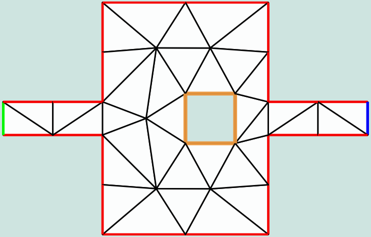
$$\frac{\partial \vec{z}}{\partial t} - \frac{1}{\mathrm{Re}}\Delta \vec{z} + (\vec{z}\cdot\nabla)\vec{w} + (\vec{w}\cdot\nabla)\vec{z} + \nabla p = 0$$

$$\frac{\partial c}{\partial t} - \frac{1}{\mathrm{Re}\,\mathrm{Sc}}\Delta c + (\vec{w}\cdot\nabla)c + (\vec{z}\cdot\nabla)c^{(\vec{w})} = 0$$

$$\operatorname{div} \vec{z} = 0$$

defined on $(0,\infty)\times\Omega$ plus BC,IC

Domain Ω: Reactor Model



LQR

Minimize

$$\mathcal{J}(\mathbf{y},\mathbf{u}) = \frac{1}{2}\int_0^\infty \lambda||\mathbf{y}||^2 + ||\mathbf{u}||^2\,\mathrm{d}t$$

s.t.

$$\begin{bmatrix} M_z & 0 & 0 \\ 0 & M_c & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{d}{dt}\begin{bmatrix} \mathbf{z} \\ \mathbf{c} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} A_z & 0 & G \\ R & A_c & 0 \\ G^T & 0 & 0 \end{bmatrix}\begin{bmatrix} \mathbf{z} \\ \mathbf{c} \\ \mathbf{p} \end{bmatrix} + \begin{bmatrix} B_z \\ 0 \\ 0 \end{bmatrix}\mathbf{u}$$

$$\mathbf{y}(t) = C_c\mathbf{c}$$

Appendix

Scenario 2: NSE Coupled with DCE in Reactor Model

PDE: NSE+DCE

Goal: $\vec{z} = \vec{v} - \vec{w} \to 0$, $c = c^{(\vec{v})} - c^{(\vec{w})} \to 0$

$\rightsquigarrow$ Linearized coupled system:

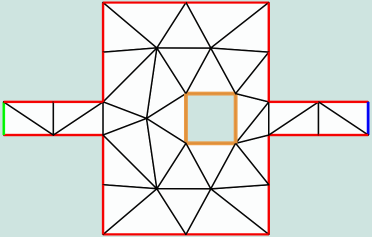
$$\frac{\partial \vec{z}}{\partial t} - \frac{1}{\mathrm{Re}}\Delta\vec{z} + (\vec{z}\cdot\nabla)\vec{w} + (\vec{w}\cdot\nabla)\vec{z} + \nabla p = 0$$

$$\frac{\partial c}{\partial t} - \frac{1}{\mathrm{Re\,Sc}}\Delta c + (\vec{w}\cdot\nabla)c + (\vec{z}\cdot\nabla)c^{(\vec{w})} = 0$$

$$\operatorname{div}\vec{z} = 0$$

defined on $(0,\infty)\times\Omega$ plus BC,IC

Domain Ω: Reactor Model



LQR

Minimize

$$\mathcal{J}(\mathbf{y},\mathbf{u}) = \frac{1}{2}\int_0^\infty \lambda||\mathbf{y}||^2 + ||\mathbf{u}||^2 \,\mathrm{d}t$$

s.t.

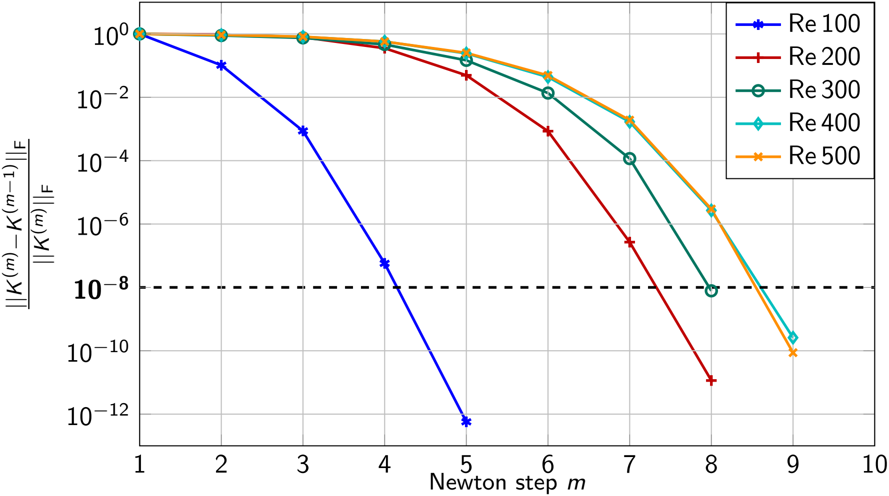
$$\mathcal{M}\frac{d}{dt}\begin{bmatrix}\tilde{\mathbf{z}}\\ \mathbf{c}\end{bmatrix} = \mathcal{A}\begin{bmatrix}\tilde{\mathbf{z}}\\ \mathbf{c}\end{bmatrix} + \begin{bmatrix}\mathcal{B}\\ 0\end{bmatrix}\mathbf{u}$$

$$\mathbf{y}(t) = C_c\mathbf{c}$$

[Heinkenschloss/Sorensen/Sun '08]

Appendix

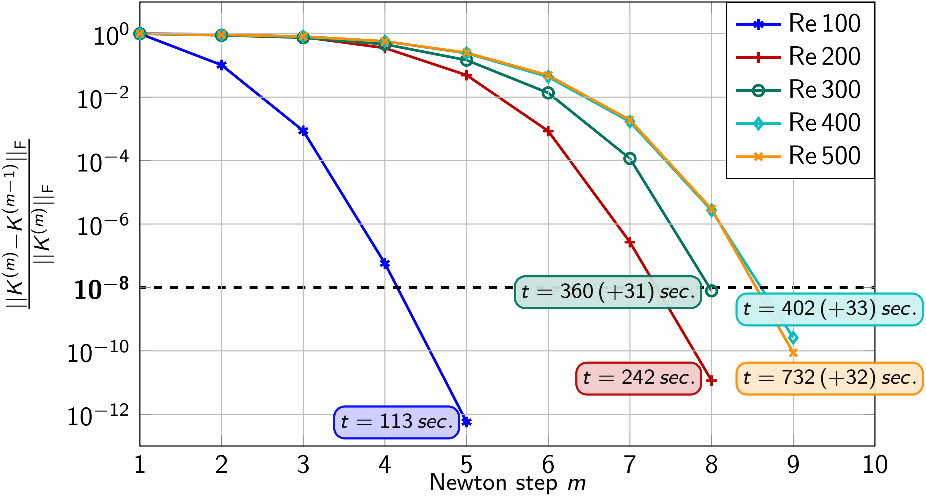
Newton-ADI: NSE on von Kármán Vortex Street



Relative change of feedback matrix K for different Reynolds numbers ($\lambda = \rho = 1$, $n = 5468$, direct solver, $tol_{\mathrm{NM}} = 10^{-8}$, $\mathbf{tol_{ADI} = 10^{-4}}$).

Appendix

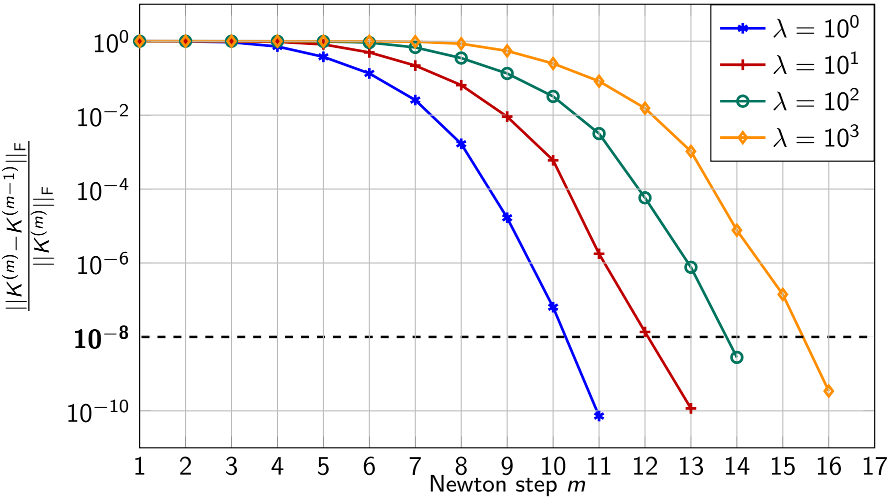
Newton-ADI: NSE on von Kármán Vortex Street



Relative change of feedback matrix K for different Reynolds numbers ($\lambda = \rho = 1$, $n = 5468$, direct solver, $tol_{\mathrm{NM}} = 10^{-8}$, $\mathbf{tol_{ADI}} = \mathbf{10^{-4}}$).

Appendix

Newton-ADI: NSE Coupled with DCE in Reactor Model



Relative change of feedback matrix K for different output weightings λ
($\mathrm{Re} = Sc = 10$, $\rho = 1$, $n = 6515$, direct solver, $tol_{\mathrm{NM}} = 10^{-8}$, $\mathbf{tol_{ADI} = 10^{-6}}$).