

# Multigrid Methods for Coupled Systems

Scott MacLachlan

Memorial University of Newfoundland and Labrador

June 10, 2021

# Big Picture

Efficient solvers for coupled  
multiphysics systems

# What multiphysics means to me

Thinking about coupled systems of

- Incompressible (nonlinear) fluid or solid dynamics
- Something else
  - ▶ Electromagnetics
  - ▶ Heat transfer

Consider linearization and finite-element discretization, resulting in linear systems of the form

$$\mathcal{A}x = \begin{bmatrix} F & Z & B \\ Y & D & 0 \\ B^T & 0 & 0 \end{bmatrix} \begin{bmatrix} x_u \\ x_a \\ x_p \end{bmatrix} = \begin{bmatrix} r_u \\ r_a \\ r_p \end{bmatrix} .$$

# Stokes and Navier-Stokes

Stokes flow is a prototypical model of incompressible fluids:

$$\begin{aligned}\rho \frac{\partial \mathbf{u}}{\partial t} - \nabla \cdot (\mu \nabla \mathbf{u}) + \nabla p &= \mathbf{f} \\ -\nabla \cdot \mathbf{u} &= 0\end{aligned}$$

- Fluid velocity,  $\mathbf{u}$ , and pressure,  $p$
- Often see in time-steady case
- $\mu$  is fluid viscosity (important in some applications),  $\rho$  is density

# Stokes and Navier-Stokes

Stokes flow is a prototypical model of incompressible fluids:

$$\begin{aligned}\rho \frac{\partial \mathbf{u}}{\partial t} - \nabla \cdot (\mu \nabla \mathbf{u}) + \nabla p &= \mathbf{f} \\ -\nabla \cdot \mathbf{u} &= 0\end{aligned}$$

- Fluid velocity,  $\mathbf{u}$ , and pressure,  $p$
- Often see in time-steady case
- $\mu$  is fluid viscosity (important in some applications),  $\rho$  is density

If inertial terms are incorporated, get Navier-Stokes

$$\begin{aligned}\rho \frac{\partial \mathbf{u}}{\partial t} + \rho (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot (\mu \nabla \mathbf{u}) + \nabla p &= \mathbf{f} \\ -\nabla \cdot \mathbf{u} &= 0\end{aligned}$$

# Thermal flows

Coupling between fluid viscosity and temperature is important in some settings

$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho (\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla \cdot (\mu(T) \nabla \mathbf{u}) + \nabla p = \mathbf{f}$$

$$-\nabla \cdot \mathbf{u} = 0$$

$$\rho c \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) - \alpha T \frac{\partial p}{\partial t} - \nabla \cdot K \nabla T = F(\mathbf{v}, T)$$

Arise in

- Mantle Convection
- Liquid cooling of electronics

Similar models are important in understanding behaviour of many *non-Newtonian* fluids

- Latex paint, Ketchup, Oobleck, ...

# Magnetohydrodynamics

$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho(\mathbf{u} \cdot \nabla \mathbf{u}) - \nabla \cdot (\mathbf{T} + \mathbf{T}_M) = \mathbf{f},$$

$$-\nabla \cdot \mathbf{u} = 0,$$

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \left( \frac{\eta}{\mu_0} \nabla \times \mathbf{B} \right) - \nabla \times (\mathbf{u} \times \mathbf{B}) = \mathbf{g},$$

$$\nabla \cdot \mathbf{B} = 0,$$

where the viscous and magnetic stress tensors are

$$\mathbf{T} = -p\mathbf{I} + \mu [\nabla \mathbf{u} + \nabla \mathbf{u}^T], \quad \mathbf{T}_M = \frac{1}{\mu_0} \mathbf{B} \otimes \mathbf{B} - \frac{1}{2\mu_0} \|\mathbf{B}\|^2 \mathbf{I},$$

with

- $\eta$  = magnetic resistivity
- $\mu_0$  = magnetic permeability of free space

# Flexoelectric nematic liquid crystals

Free-energy minimization of Lagrangian:

$$\int_{\Omega} w_F(\mathbf{u}) - \frac{1}{2} \mathbf{D} \cdot \mathbf{E} - \mathbf{P} \cdot \mathbf{E} + \lambda(\mathbf{u} \cdot \mathbf{u} - 1) dV.$$

where

$$w_F(\mathbf{u}) = \frac{1}{2} K_1 (\nabla \cdot \mathbf{u})^2 + \frac{1}{2} K_2 (\mathbf{u} \cdot \nabla \times \mathbf{u})^2 + \frac{1}{2} K_3 |\mathbf{u} \times \nabla \times \mathbf{u}|^2$$

$$\mathbf{D} = \epsilon_0 \epsilon_{\perp} \mathbf{E} + \epsilon_0 \epsilon_a (\mathbf{u} \cdot \mathbf{E}) \mathbf{u}$$

$$\mathbf{P} = e_s (\nabla \cdot \mathbf{u}) \mathbf{u} + e_b \mathbf{u} \times \nabla \times \mathbf{u}$$

plus Maxwell's equations for  $\mathbf{E}$ ,  $\mathbf{D}$ ,  $\mathbf{P}$



# (Lots of) Options

There are many issues associated with going from the nonlinear PDEs to the linearized and discretized system

- How to linearize?
  - ▶ Fixed point iteration, operator splitting, Newton
  - ▶ Can we guarantee nonlinear convergence?
  - ▶ Time-dependent or steady?
- How to discretize?
  - ▶ Staggered finite difference, mixed finite elements (stable or stabilized), finite volume
  - ▶ Accuracy, conserved quantities?
- Discretize-then-linearize or linearize-then-discretize?

These choices matter in building good solvers

# Discretization, Take One

Tempting to try and discretize without thinking about system structure.

Consider time-steady, iso-viscous Stokes:

$$\begin{aligned}-\Delta \mathbf{u} + \nabla p &= \mathbf{f} \\ -\nabla \cdot \mathbf{u} &= 0\end{aligned}$$

Could we use

- Centred finite-differences for both  $\mathbf{u}$  and  $p$ ?
- Piecewise (bi/tri)linear finite elements?

# Discretization, Take One

Tempting to try and discretize without thinking about system structure.

Consider time-steady, iso-viscous Stokes:

$$\begin{aligned}-\Delta \mathbf{u} + \nabla p &= \mathbf{f} \\ -\nabla \cdot \mathbf{u} &= 0\end{aligned}$$

Could we use

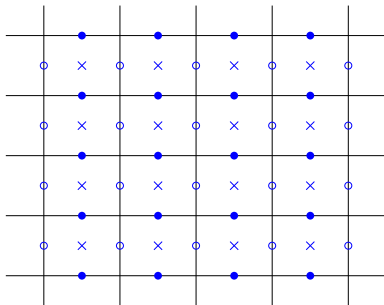
- Centred finite-differences for both  $\mathbf{u}$  and  $p$ ?
- Piecewise (bi/tri)linear finite elements?

NO!

- Simple discretizations are also unstable (solutions are not unique)

# Discretization, Take Two

Achieve stable finite-difference discretization by using *staggered grids* (Marker-and-Cell (MAC) scheme)



Discretize normal velocity on edges and pressure at cell centers

- Standard finite-difference approach on staggered unknowns
- Approach generalizes to other common systems of PDEs

# Discretization, Take Three

Finite-element approaches rely on *mixed finite-element* technology

- For Stokes, generally use higher-order finite-element for velocity than for pressure
  - ▶ Taylor-Hood:  $P_2 - P_1$  or  $Q_2 - Q_1$
  - ▶ Scott-Vogelius:  $P_k - DP_{k-1}$
  - ▶ Non-conforming:  $RT_k - DP_{k-1}$
- General framework for other equations, spaces
- Well-known *inf-sup* condition to guarantee stability

# Saddle-Point Systems

*Stable* discretizations lead to linear systems to solve of the form

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix}$$

where

- $A$  is symmetric and positive definite (vector Laplacian)
- $B^T$  is (nearly) full row rank (divergence)
- $\mathbf{u}$ ,  $p$  are discrete velocity and pressure unknowns
- $\mathbf{f}$  includes RHS data, BCs, etc.

# Multigrid Options

Two main approaches for using multigrid:

1. Preconditioning based on approximate block factorization, using multigrid as a solver for diagonal blocks
  - ▶ Well-established in theory and practice
  - ▶ Easier as a “black box”
  - ▶ Relies on knowing good approximation to Schur complement
2. Applying *monolithic* multigrid to entire system
  - ▶ Requires specialized relaxation schemes (point Jacobi doesn't work!)
  - ▶ Often more efficient in the end

# Block-Factorization Preconditioners

Notice that

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} = \begin{bmatrix} A & 0 \\ B^T & -S \end{bmatrix} \begin{bmatrix} I & A^{-1}B \\ 0 & I \end{bmatrix}$$

for  $S = B^T A^{-1} B$ .



# Block-Factorization Preconditioners

Notice that

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} = \begin{bmatrix} A & 0 \\ B^T & -S \end{bmatrix} \begin{bmatrix} I & A^{-1}B \\ 0 & I \end{bmatrix}$$

for  $S = B^T A^{-1} B$ . Three families of preconditioners:

- Full factorization

$$M = \begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix} \begin{bmatrix} I & \hat{A}^{-1}B \\ 0 & I \end{bmatrix}$$

- Approximate  $A$  and  $S$  by easily inverted  $\hat{A}$ ,  $\hat{S}$
- Take  $\hat{A}$  to be one multigrid cycle applied to velocity block
- Take  $\hat{S}$  to approximate mass/diagonal matrix

# Block-Factorization Preconditioners

Notice that

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} = \begin{bmatrix} A & 0 \\ B^T & -S \end{bmatrix} \begin{bmatrix} I & A^{-1}B \\ 0 & I \end{bmatrix}$$

for  $S = B^T A^{-1} B$ . Three families of preconditioners:

- Block-triangular approximation

$$M = \begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix}$$

- Approximate  $A$  and  $S$  by easily inverted  $\hat{A}$ ,  $\hat{S}$
- Take  $\hat{A}$  to be one multigrid cycle applied to velocity block
- Take  $\hat{S}$  to approximate mass/diagonal matrix

---

Elman, Sylvester, Wathen, Finite elements and fast iterative solvers, OUP 2005

# Block-Factorization Preconditioners

Notice that

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} = \begin{bmatrix} A & 0 \\ B^T & -S \end{bmatrix} \begin{bmatrix} I & A^{-1}B \\ 0 & I \end{bmatrix}$$

for  $S = B^T A^{-1} B$ . Three families of preconditioners:

- Block-diagonal approximation

$$M = \begin{bmatrix} \hat{A} & 0 \\ 0 & \pm \hat{S} \end{bmatrix}$$

- Approximate  $A$  and  $S$  by easily inverted  $\hat{A}$ ,  $\hat{S}$
- Take  $\hat{A}$  to be one multigrid cycle applied to velocity block
- Take  $\hat{S}$  to approximate mass/diagonal matrix

---

Elman, Sylvester, Wathen, Finite elements and fast iterative solvers, OUP 2005

# Block Factorization Works

Key point is that block methods with *exact* solves for  $A$  and  $S$  are ideal preconditioners

- At most 3 iterations for block diagonal
- At most 2 iterations for block triangular
- At most 1 iteration for full factorization

# Block Factorization Works

Key point is that block methods with *exact* solves for  $A$  and  $S$  are ideal preconditioners

- At most 3 iterations for block diagonal
- At most 2 iterations for block triangular
- At most 1 iteration for full factorization

Why do anything else?

- Monolithic approaches can be more efficient
- Don't always know how to approximate Schur complement
- Nested Schur complements add complication when considering larger systems

# Geometric Monolithic Multigrid

Basic idea: hierarchy of discretizations of coupled system

- Use geometric interpolation (e.g., canonical finite-element interpolation) between levels
- Don't interpolate between different components of the system

For Stokes,  $P = \begin{bmatrix} P_u & 0 \\ 0 & P_p \end{bmatrix}$

- Define coarse-grid operators either by rediscretization or Galerkin (sometimes equivalent)
- Focus energy on design and analysis of effective relaxation schemes for coupled systems

# Coupled Relaxation Schemes

Three common families of relaxation methods

## 1. Distributive Relaxation

- ▶ Idea: change the problem to one where pointwise relaxation works

---

Brandt, Multigrid techniques: 1984 guide with applications to fluid dynamics

# Coupled Relaxation Schemes

Three common families of relaxation methods

## 1. Distributive Relaxation

- ▶ Idea: change the problem to one where pointwise relaxation works

## 2. Approximate Block Factorization

- ▶ Idea: use a simple block-factorization preconditioner as relaxation
- ▶ Uzawa relaxation uses block-triangular preconditioner
- ▶ Braess-Sarazin relaxation uses full approximate Schur complement

---

Maitre, Musy, Nignon, A fast solver for the Stokes equations using multi-grid with a Uzawa smoother, 1984

Braess, Sarazin, Appl. Numer. Math. 23, 1997



# Coupled Relaxation Schemes

Three common families of relaxation methods

## 1. Distributive Relaxation

- ▶ Idea: change the problem to one where pointwise relaxation works

## 2. Approximate Block Factorization

- ▶ Idea: use a simple block-factorization preconditioner as relaxation
- ▶ Uzawa relaxation uses block-triangular preconditioner
- ▶ Braess-Sarazin relaxation uses full approximate Schur complement

## 3. Overlapping Schwarz Methods

- ▶ Idea: form overlapping blocks around constraint DoFs
- ▶ For fluids, use Vanka approach; for EM, use star relaxation

---

Vanka, J. Comput. Phys. 65, 1986

Arnold, Falk, Winther, Math. Comp. 66, 1997

# Distributive Relaxation

At continuum level, we can transform the Stokes equations into modified system:

Let  $\hat{p}$  be such that  $\Delta\hat{p} = p$ ,  $\hat{\mathbf{u}} = \mathbf{u} - \nabla\hat{p}$ :

$$\begin{aligned} -\Delta\mathbf{u} + \nabla p &= -\Delta(\hat{\mathbf{u}} + \nabla\hat{p}) + \nabla p \\ &= -\Delta\hat{\mathbf{u}} - \nabla\Delta\hat{p} + \nabla p = \mathbf{f} \\ -\nabla \cdot \mathbf{u} &= -\nabla \cdot \hat{\mathbf{u}} - \Delta\hat{p} = 0 \end{aligned}$$

At discrete level, approximate this:

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} I & B \\ 0 & -A_p \end{bmatrix} \approx \begin{bmatrix} A & 0 \\ B^T & B^T B \end{bmatrix},$$

where  $A_p \approx B^T B$  is the pressure Laplacian

- Relies on discrete commutator relationship,  $AB \approx BA_p$

# Practical Details, Part I

Suppose we had approximations,  $\hat{\mathbf{u}}^{(k)}$  and  $\hat{\mathbf{p}}^{(k)}$ , to  $\hat{\mathbf{u}}$  and  $\hat{\mathbf{p}}$ .  
Then compute updates as

$$\begin{bmatrix} \mathbf{r}_u^{(k+1)} \\ \mathbf{r}_p^{(k+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} I & B \\ 0 & -A_p \end{bmatrix} \begin{bmatrix} \hat{\mathbf{u}}^{(k)} \\ \hat{\mathbf{p}}^{(k)} \end{bmatrix}$$
$$\begin{bmatrix} \hat{\mathbf{u}}^{(k+1)} \\ \hat{\mathbf{p}}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{u}}^{(k)} \\ \hat{\mathbf{p}}^{(k)} \end{bmatrix} + \omega \begin{bmatrix} \hat{A} & 0 \\ B^T & \widehat{B^T B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{r}_u^{(k+1)} \\ \mathbf{r}_p^{(k+1)} \end{bmatrix}$$

Then could take

$$\begin{bmatrix} \mathbf{u}^{(k+1)} \\ \mathbf{p}^{(k+1)} \end{bmatrix} = \begin{bmatrix} I & B \\ 0 & -A_p \end{bmatrix} \begin{bmatrix} \hat{\mathbf{u}}^{(k+1)} \\ \hat{\mathbf{p}}^{(k+1)} \end{bmatrix}$$

## Practical Details, Part II

But note that if we know

$$\begin{bmatrix} \mathbf{u}^{(k)} \\ p^{(k)} \end{bmatrix} = \begin{bmatrix} I & B \\ 0 & -A_p \end{bmatrix} \begin{bmatrix} \hat{\mathbf{u}}^{(k)} \\ \hat{p}^{(k)} \end{bmatrix}$$

Then

$$\begin{bmatrix} \mathbf{r}_u^{(k+1)} \\ r_p^{(k+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k)} \\ p^{(k)} \end{bmatrix}$$

Multiplying correction on left by  $\begin{bmatrix} I & B \\ 0 & -A_p \end{bmatrix}$  gives

$$\begin{bmatrix} \mathbf{u}^{(k+1)} \\ p^{(k+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{u}^{(k)} \\ p^{(k)} \end{bmatrix} + \omega \begin{bmatrix} I & B \\ 0 & -A_p \end{bmatrix} \begin{bmatrix} \hat{A} & 0 \\ B^T & \widehat{B^T B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{r}_u^{(k+1)} \\ r_p^{(k+1)} \end{bmatrix}$$

⇒ Don't need  $\hat{\mathbf{u}}$ ,  $\hat{p}$  to do distributive relaxation!

# Limitations

Distributive Relaxation works well in monolithic MG for

- Simple staggered FD discretizations
- Linear systems of PDEs

Effectiveness relies on discrete commutator:  $\Delta \nabla p = \nabla \Delta p$

- More natural on structured uniform meshes
- Generally violated by BCs in  $AB \approx BA_p$

# Uzawa Iteration

Uzawa iteration aims to solve

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix}$$

iteratively, by computing

$$\begin{aligned} \mathbf{u}^{(k+1)} &= A^{-1} (\mathbf{f} - Bp^{(k)}) \\ p^{(k+1)} &= p^{(k)} + \alpha B^T \mathbf{u}^{(k+1)} \end{aligned}$$

Recognize this as a Richardson (stationary) iteration for solving

$$B^T A^{-1} B p = B^T A^{-1} \mathbf{f}$$

with  $\mathbf{u} = A^{-1} (\mathbf{f} - Bp)$ .

# Inexact Uzawa Iteration

Can also introduce inexact solves for  $A$ , leading to

$$\begin{aligned}\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \hat{A}^{-1} (\mathbf{f} - A\mathbf{u}^{(k)} - B\mathbf{p}^{(k)}) \\ \mathbf{p}^{(k+1)} &= \mathbf{p}^{(k)} + \alpha B^T \mathbf{u}^{(k+1)}\end{aligned}$$

# Inexact Uzawa Iteration

Can also introduce inexact solves for  $A$ , leading to

$$\begin{aligned}\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \hat{A}^{-1} (\mathbf{f} - A\mathbf{u}^{(k)} - B\mathbf{p}^{(k)}) \\ \mathbf{p}^{(k+1)} &= \mathbf{p}^{(k)} + \alpha B^T \mathbf{u}^{(k+1)}\end{aligned}$$

And better approximations to Schur complement solve, giving

$$\begin{aligned}\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \hat{A}^{-1} (\mathbf{f} - A\mathbf{u}^{(k)} - B\mathbf{p}^{(k)}) \\ \mathbf{p}^{(k+1)} &= \mathbf{p}^{(k)} + \hat{S}^{-1} (B^T \mathbf{u}^{(k+1)})\end{aligned}$$

Lots of analysis of efficiency of these as standalone iterations

- Can introduce Krylov acceleration
- Understand properties needed for robust convergence



# Uzawa Relaxation

For efficient multigrid, however, choose  $\hat{A}$  and  $\hat{S}$  to give good smoothing properties

Typical choices:

- Gauss-Seidel relaxation for  $\hat{A}$ , Richardson for  $\hat{S}$
- Weighted Jacobi relaxation for  $\hat{A}$
- Approximate  $S$  by  $B^T D^{-1} B$ , where  $D$  is (block) diagonal of  $A$ , then use weighted Jacobi on this for  $\hat{S}$

Can extend Local Fourier Analysis to understand these choices and relaxation weights

---

Maitre, Musy, Nignon, A fast solver for the Stokes equations using multigrid with a Uzawa smoother, 1984

He, MacLachlan, NLAA 2018, JCAM 2019

# Symmetrizing Uzawa

Can rewrite general Uzawa iteration as

$$\begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k+1)} - \mathbf{u}^{(k)} \\ \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k)} \\ \mathbf{p}^{(k)} \end{bmatrix}$$

Recognize  $\begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix}$  as approximation to lower-triangular

part of  $LU$  factorization of  $\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}$

Why not include upper-triangular solve as well?

# Symmetrizing Uzawa

Can rewrite general Uzawa iteration as

$$\begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k+1)} - \mathbf{u}^{(k)} \\ \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k)} \\ \mathbf{p}^{(k)} \end{bmatrix}$$

Recognize  $\begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix}$  as approximation to lower-triangular

part of  $LU$  factorization of  $\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}$

Approximate

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \approx \begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix} \begin{bmatrix} I & \hat{A}^{-1}B \\ 0 & I \end{bmatrix} = \begin{bmatrix} \hat{A} & B \\ B^T & \hat{C} \end{bmatrix}$$

for  $\hat{C} = B^T \hat{A}^{-1} B - \hat{S}$

# Symmetrizing Uzawa

Can rewrite general Uzawa iteration as

$$\begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k+1)} - \mathbf{u}^{(k)} \\ \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k)} \\ \mathbf{p}^{(k)} \end{bmatrix}$$

Recognize  $\begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix}$  as approximation to lower-triangular

part of  $LU$  factorization of  $\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}$

Replace Uzawa by

$$\begin{bmatrix} \hat{A} & B \\ B^T & \hat{C} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k+1)} - \mathbf{u}^{(k)} \\ \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k)} \\ \mathbf{p}^{(k)} \end{bmatrix}$$

# Symmetrizing Uzawa

Can rewrite general Uzawa iteration as

$$\begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k+1)} - \mathbf{u}^{(k)} \\ \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k)} \\ \mathbf{p}^{(k)} \end{bmatrix}$$

Recognize  $\begin{bmatrix} \hat{A} & 0 \\ B^T & -\hat{S} \end{bmatrix}$  as approximation to lower-triangular

part of  $LU$  factorization of  $\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}$

Componentwise iteration is

$$\mathbf{u}^{(k+1/2)} = \mathbf{u}^{(k)} + \hat{A}^{-1} (\mathbf{f} - A\mathbf{u}^{(k)} - B\mathbf{p}^{(k)})$$

$$\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + \hat{S}^{-1} (B^T \mathbf{u}^{(k+1/2)})$$

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(k+1/2)} - \hat{A}^{-1} B (\mathbf{p}^{(k+1)} - \mathbf{p}^{(k)})$$

# Braess-Sarazin Relaxation

Braess and Sarazin proposed using this as a relaxation, fixing  $\hat{S} = B^T \hat{A}^{-1} B$ :

$$\begin{bmatrix} \hat{A} & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k+1)} - \mathbf{u}^{(k)} \\ \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k)} \\ \mathbf{p}^{(k)} \end{bmatrix}$$

Idea: for simple  $\hat{A}$  (e.g., diagonal)

1. Compute  $B^T \hat{A}^{-1} B$  exactly
2. Solve  $\hat{S} (\mathbf{p}^{(k+1)} - \mathbf{p}^{(k)}) = B^T \mathbf{u}^{(k+1/2)}$  using multigrid (or something similar) to high tolerance

# Braess-Sarazin Relaxation

Braess and Sarazin proposed using this as a relaxation, fixing  $\hat{S} = B^T \hat{A}^{-1} B$ :

$$\begin{bmatrix} \hat{A} & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k+1)} - \mathbf{u}^{(k)} \\ \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(k)} \\ \mathbf{p}^{(k)} \end{bmatrix}$$

Idea: for simple  $\hat{A}$  (e.g., diagonal)

1. Compute  $B^T \hat{A}^{-1} B$  exactly
2. Solve  $\hat{S} (\mathbf{p}^{(k+1)} - \mathbf{p}^{(k)}) = B^T \mathbf{u}^{(k+1/2)}$  using multigrid (or something similar) to high tolerance

Zulehner later proposed to solve system with  $\hat{S}$  inexactly

- Multigrid to low tolerance (1 V-cycle)
- *For some discretizations* can even use small number of weighted Jacobi iterations

---

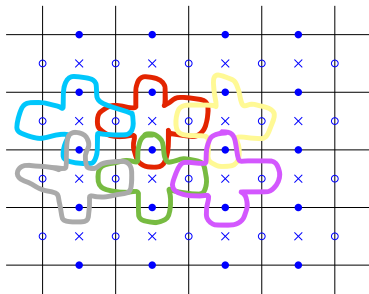
Braess, Sarazin, App Num Math 1997

Zulehner, Computing 2000

# Vanka Relaxation

First proposed for MAC scheme discretization of Navier-Stokes

- Overlapping Schwarz method as relaxation scheme
- Restrict residual and matrix to local subdomain, solve there and correct
- Construction of subdomains is key to achieving convergence





# Vanka for (Navier-)Stokes

Many extensions to many discretizations

- Central theme: “pressure-centric” relaxation
  - ▶ Define subdomains to include a single pressure degree of freedom and some subset of the velocities in its geometric/algebraic neighbourhood
- Multiplicative, additive, hybrid variants all work
- Typical use is in geometric multigrid
- Some sensitivity to parameter choice
- Generally do exact subdomain solves
  - ▶ Factoring submatrices can be expensive
  - ▶ Sometimes see “diagonal” variant

---

John, Tobiska, IJNMF 2000 & John, Matthies, IJNMF 2001

Rodrigo, Gaspar, Lisbona, App. Num. Math. 2016

Adler, Benson, MacLachlan, NLAA 2017

Farrell, He, MacLachlan, NLAA 2021

# “Vanka” for Maxwell’s Equations

Similar patch-based relaxation is common for both

$$\begin{aligned}-\nabla (K \nabla \cdot \mathbf{u}) + \mathbf{u} &= \mathbf{f} \\ \nabla \times (K \nabla \times \mathbf{v}) + \mathbf{v} &= \mathbf{g}\end{aligned}$$

No pressure!

- Implied constraints from vector calculus:

$$\nabla \times \mathbf{u} = \nabla \times \mathbf{f} \text{ and } \nabla \cdot \mathbf{v} = \nabla \cdot \mathbf{g}$$

- Exact sequence property that leads to effective subdomain construction for Raviart-Thomas and Nédélec elements

# Some Theory

Consider nearly singular systems,  $(A_0 + \epsilon A_1)u = f$ , where  $A_0$  is SPSD (and singular), and  $A_1$  is SPD

Need 2 ingredients to guarantee convergence

- Interpolation must coarse-grid kernel to fine-grid kernel
- Subdomains for Vanka-type relaxation must provide stable decomposition of the kernel
  - ▶ Again, cannot choose *any* set of Vanka patches, but must do so in compatible way to problem and discretization

This framework has been adapted in recent years to yield robust algorithms for new problems and new discretizations

---

Schöberl, PhD thesis 1999, Numer. Math. 1999

Lee, Wu, Xu, Zikatanov, MMMAS 2007

# Some Software

Vanka (and other) relaxation schemes easily implemented in Firedrake using PCPATCH

- Firedrake is a well-developed and fully featured FEM package
- Strong focus on code generation to enable HPC use
- Nonlinear and linear solvers enabled through interface to PETSc
- PCPATCH is tightly coupled library that implements “patch-based” relaxation within this framework

---

Rathgeber et al., ACM TOMS 2016

Kirby, Mitchell, SISC 2018

Balay et al., PETSc Users Manual 2021

Farrell, Knepley, Mitchell, Wechsung, ACM TOMS 2021 (to appear)

# FD for Stokes

Use weighted-Jacobi variants (allows parallelism)

- Fix 1 relaxation sweep per level (e.g.,  $W(1,0)$  cycles)
- Can prove (using LFA) optimal convergence factors of
  - ▶ Distributive Relaxation = 0.6
  - ▶ Inexact Braess-Sarazin = 0.6
  - ▶ Uzawa Relaxation =  $\sqrt{0.6}$
- Uzawa costs about  $1/2$  as much per iteration as other two
- See good agreement between measured performance and theory

# FEM for Stokes

Again focus on additive variants

- For stabilized Q1-Q1:
  - ▶ Distributive Relaxation convergence factor is  $1/3$   
(requires 2 sweeps of Jacobi on pressure)
  - ▶ Exact Braess-Sarazin convergence factor is  $1/3$
  - ▶ Inexact Braess-Sarazin convergence degrades; need MG on pressure equation to get close to exact results
  - ▶ Observe Uzawa convergence factors around  $1/2$

Generally find Vanka is less sensitive to parameter choices, but BSR with tuned parameters gives fastest time to solution

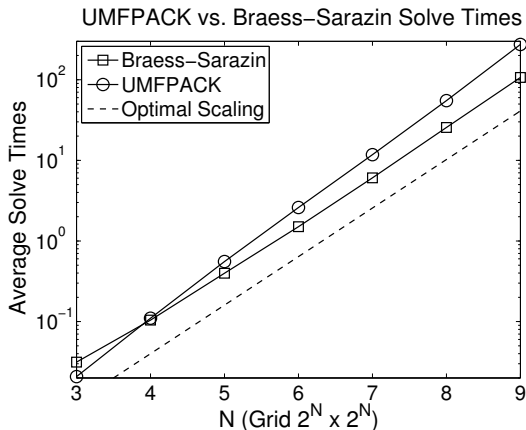
# FEM for Stokes

Again focus on additive variants

- For (stable) Q2-Q1:
  - ▶ Theory is harder
  - ▶ Observe convergence factors of 0.6 for distributive relaxation
  - ▶ With  $W(1,1)$  cycles and 3 sweeps of Jacobi on pressure, observe convergence factor of 0.25 for inexact Braess-Sarazin
  - ▶ Observe Uzawa convergence factor of about 0.55
  - ▶ LFA for Vanka gives convergence factor of 0.6

Generally find Vanka is less sensitive to parameter choices, but BSR with tuned parameters gives fastest time to solution

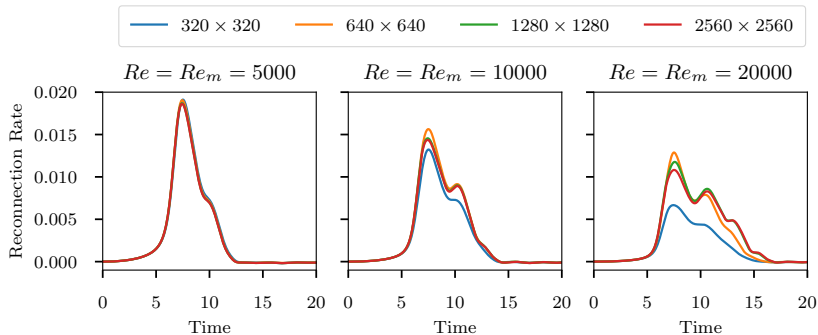
# FEM for Liquid Crystals



Flexoelectric LC, using Q2-Q2-P0 discretization

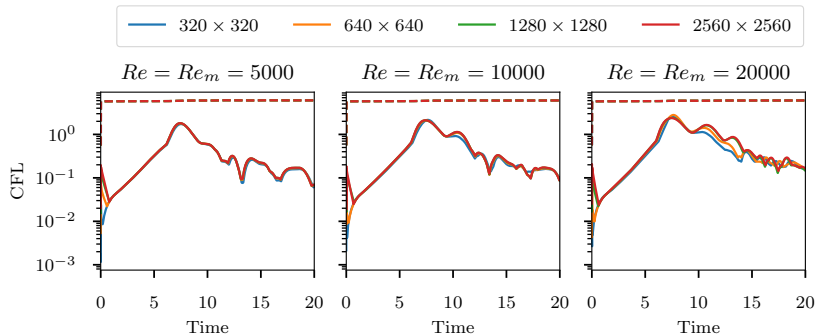


# MHD Island Coalescence



- See expected drop in peak reconnection rate with  $Re = Re_m$
- See expected increase in oscillations as  $Re = Re_m$  increases

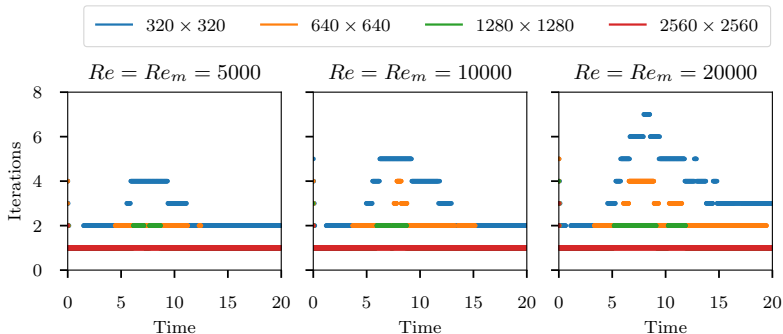
# MHD Island Coalescence



- Unstabilized discretization for convective terms, so CFL is limiting
- Alfvén CFL (dashed line) stays about 5
- Fluid CFL (solid) varies more, peaks about 2 or 3

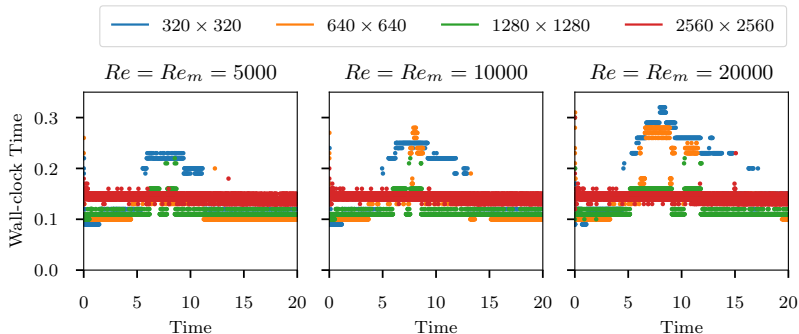
Adler et al., SISC 2021, to appear

# MHD Island Coalescence



- See advantage to timestepping as  $dt$  decreases, particularly at high  $Re = Re_m$

# MHD Island Coalescence



- Weak scaling from 10 to 640 cores with refinement

# Outlook

## Lots of Methods for MG for Systems

- Block-Factorization Preconditioners
  - ▶ Excel for high- $Re$  flows with augmented Lagrangian preconditioners (Farrell and co-workers, 2018–present)
- Monolithic Multigrid
  - ▶ Success with several relaxation schemes
  - ▶ Expanding number of problems and discretizations
  - ▶ Lots more work to do!