Reynolds-robust solvers for incompressible flow problems

Patrick E. Farrell[1,2] Lawrence Mitchell[3] Ridgway Scott[4] Florian Wechsung[1]




[1]University of Oxford

[2]Charles University

[3]NVIDIA

[4]University of Chicago

A fundamental problem in fluid mechanics:

Stationary incompressible Navier–Stokes

For Reynolds number $\mathrm{Re} \in \mathbb{R}_+$, find $(u, p) \in [H^1(\Omega)]^d \times L^2(\Omega)$ such that

$$
\begin{aligned}
-\operatorname{div}\left(2\mathrm{Re}^{-1}\varepsilon(u)\right) + \operatorname{div}(u \otimes u) + \operatorname{grad} p &= f && \text{in } \Omega, \\
\operatorname{div} u &= 0 && \text{in } \Omega, \\
u &= g && \text{on } \Gamma_D, \\
2\mathrm{Re}^{-1}\varepsilon(u) \cdot n &= pn && \text{on } \Gamma_N.
\end{aligned}
$$

A fundamental problem in fluid mechanics:

Stationary incompressible Navier–Stokes

For Reynolds number $\mathrm{Re} \in \mathbb{R}_+$, find $(u, p) \in [H^1(\Omega)]^d \times L^2(\Omega)$ such that

$$\begin{aligned} -\mathrm{div}\left(2\mathrm{Re}^{-1}\varepsilon(u)\right) + \mathrm{div}\,(u \otimes u) + \mathrm{grad}\, p &= f && \text{in } \Omega, \\ \mathrm{div}\, u &= 0 && \text{in } \Omega, \\ u &= g && \text{on } \Gamma_D, \\ 2\mathrm{Re}^{-1}\varepsilon(u) \cdot n &= pn && \text{on } \Gamma_N. \end{aligned}$$

This talk

Preconditioner with *Reynolds-robust* GMRES performance in 2D & 3D.

Combines and develops many techniques that are useful for other difficult PDEs.

Section 1

Saddle point problems

These equations have a *saddle point structure*. Consider the following minimisation problem:

$$
\begin{aligned}
u = \underset{v \in H_0^1(\Omega;\mathbb{R}^n)}{\arg\min} \quad & \frac{1}{2}\int_\Omega 2\mathrm{Re}^{-1}\epsilon(v) : \epsilon(v)\ \mathrm{d}x - \int_\Omega f \cdot v\ \mathrm{d}x, \\
\text{subject to} \quad & \nabla \cdot v = 0.
\end{aligned}
$$

These equations have a *saddle point structure*. Consider the following minimisation problem:

$$
\begin{aligned}
u = \underset{v \in H_0^1(\Omega;\mathbb{R}^n)}{\arg\min} \quad & \frac{1}{2}\int_\Omega 2\mathrm{Re}^{-1}\epsilon(v) : \epsilon(v)\ \mathrm{d}x - \int_\Omega f \cdot v\ \mathrm{d}x, \\
\text{subject to} \quad & \nabla \cdot v = 0.
\end{aligned}
$$

Introducing a Lagrange multiplier $p \in L_0^2(\Omega)$ for the incompressibility constraint yields the Lagrangian

$$
L(u, p) = \frac{1}{2}\int_\Omega 2\mathrm{Re}^{-1}\epsilon(u) : \epsilon(u)\ \mathrm{d}x - \int_\Omega f \cdot u\ \mathrm{d}x - \int_\Omega p\nabla \cdot u\ \mathrm{d}x.
$$

These equations have a *saddle point structure*. Consider the following minimisation problem:

$$u = \operatorname*{arg\,min}_{v \in H_0^1(\Omega;\mathbb{R}^n)} \quad \frac{1}{2}\int_\Omega 2\mathrm{Re}^{-1}\epsilon(v) : \epsilon(v)\ \mathrm{d}x - \int_\Omega f \cdot v\ \mathrm{d}x,$$

$$\text{subject to} \qquad \nabla \cdot v = 0.$$

Introducing a Lagrange multiplier $p \in L_0^2(\Omega)$ for the incompressibility constraint yields the Lagrangian

$$L(u,p) = \frac{1}{2}\int_\Omega 2\mathrm{Re}^{-1}\epsilon(u) : \epsilon(u)\ \mathrm{d}x - \int_\Omega f \cdot u\ \mathrm{d}x - \int_\Omega p\nabla \cdot u\ \mathrm{d}x.$$

The solution of this problem (u,p) is a saddle point of the Lagrangian because it satisfies

$$L(u,q) \leq L(u,p) \leq L(v,p) \text{ for all } v \in H_0^1(\Omega;\mathbb{R}^n),\ q \in L_0^2(\Omega).$$

Taking the optimality conditions, we find exactly the Stokes equations:

$$\begin{aligned} -2\mathrm{Re}^{-1}\nabla\cdot(\epsilon(u)) + \nabla p &= f, \\ -\nabla\cdot u \qquad\qquad\qquad &= 0. \end{aligned}$$

Taking the optimality conditions, we find exactly the Stokes equations:

$$\begin{aligned} -2\mathrm{Re}^{-1}\nabla\cdot(\epsilon(u)) + \nabla p &= f, \\ -\nabla\cdot u \qquad\qquad &= 0. \end{aligned}$$

We want to build solvers for saddle point problems like

$$\begin{aligned} Au + B^\top p &= f, \\ Bu \qquad &= 0. \end{aligned}$$

Taking the optimality conditions, we find exactly the Stokes equations:

$$\begin{aligned} -2\mathrm{Re}^{-1}\nabla\cdot(\epsilon(u)) + \nabla p &= f, \\ -\nabla\cdot u \qquad\qquad &= 0. \end{aligned}$$

We want to build solvers for saddle point problems like

$$\begin{aligned} Au + B^\top p &= f, \\ Bu \qquad &= 0. \end{aligned}$$

If A is invertible, then we can left-multiply the first equation by A^{-1} to get

$$u = A^{-1}f - A^{-1}B^\top p,$$

Taking the optimality conditions, we find exactly the Stokes equations:

$$\begin{aligned} -2\mathrm{Re}^{-1}\nabla\cdot(\epsilon(u)) + \nabla p &= f, \\ -\nabla\cdot u \qquad\qquad &= 0. \end{aligned}$$

We want to build solvers for saddle point problems like

$$\begin{aligned} Au + B^\top p &= f, \\ Bu \qquad &= 0. \end{aligned}$$

If A is invertible, then we can left-multiply the first equation by A^{-1} to get

$$u = A^{-1}f - A^{-1}B^\top p,$$

and substituting this into the second equation yields

$$-BA^{-1}B^\top p = -BA^{-1}f,$$

where the new operator

$$S := -BA^{-1}B^\top$$

is called the *Schur complement*. The Schur complement is **dense**.

In fact, more generally, if A is invertible

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} I & 0 \\ CA^{-1} & I \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} I & A^{-1}B \\ 0 & I \end{bmatrix}.$$

where $S = D - CA^{-1}B$ again is the Schur complement.

In fact, more generally, if A is invertible

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} I & 0 \\ CA^{-1} & I \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} I & A^{-1}B \\ 0 & I \end{bmatrix}.$$

where $S = D - CA^{-1}B$ again is the Schur complement.

This is extremely useful, because we can write an explicit formula for the inverse:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} I & -A^{-1}B \\ 0 & I \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & S^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -CA^{-1} & I \end{bmatrix}.$$

This gives rise to four related theorems about block preconditioners.

Theorem (`full`)

The choice of preconditioner

$$\mathcal{P} = \begin{pmatrix} I & 0 \\ CA^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

will yield GMRES convergence in **1 iteration.**

Andy Wathen

Gene Golub

This gives rise to four related theorems about block preconditioners.

Theorem (`lower`)

The choice of preconditioner

$$\mathcal{P} = \begin{pmatrix} I & 0 \\ CA^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & S \end{pmatrix}$$

will yield GMRES convergence in **2 iterations.**

Andy Wathen

Gene Golub

This gives rise to four related theorems about block preconditioners.

Theorem (upper)

The choice of preconditioner

$$\mathcal{P} = \begin{pmatrix} A & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

will yield GMRES convergence in **2 iterations.**

Andy Wathen

Gene Golub

This gives rise to four related theorems about block preconditioners.

Theorem (diag)

The choice of preconditioner

$$\mathcal{P} = \begin{pmatrix} A & 0 \\ 0 & -S \end{pmatrix}$$

will yield GMRES convergence in **3 iterations**, **if** $D = 0$.

Andy Wathen

Gene Golub

This gives rise to four related theorems about block preconditioners.

Andy Wathen

Theorem (`diag`)

The choice of preconditioner

$$\mathcal{P} = \begin{pmatrix} A & 0 \\ 0 & -S \end{pmatrix}$$

will yield GMRES convergence in **3 iterations**, **if** $D = 0$.

Gene Golub

How do you use this?

We have to build solvers for A and S.

For Stokes,

$$A_{ij} = 2\mathrm{Re}^{-1} \int_\Omega \epsilon(\phi_j) : \epsilon(\phi_i) \, \mathrm{d}x,$$

a nice symmetric, coercive operator (with boundary conditions).

Andy Wathen

David Silvester

For Stokes,

$$A_{ij} = 2\mathrm{Re}^{-1} \int_{\Omega} \epsilon(\phi_j) : \epsilon(\phi_i)\ \mathrm{d}x,$$

a nice symmetric, coercive operator (with boundary conditions).

Multigrid/domain decomposition is the natural choice to approximate A^{-1}.

Andy Wathen

David Silvester

For Stokes,

$$A_{ij} = 2\mathrm{Re}^{-1} \int_\Omega \epsilon(\phi_j) : \epsilon(\phi_i) \, \mathrm{d}x,$$

a nice symmetric, coercive operator (with boundary conditions).

Multigrid/domain decomposition is the natural choice to approximate A^{-1}.

But what about the Schur complement S?

Andy Wathen

David Silvester

For Stokes,

$$A_{ij} = 2\mathrm{Re}^{-1} \int_\Omega \epsilon(\phi_j) : \epsilon(\phi_i) \, \mathrm{d}x,$$

a nice symmetric, coercive operator (with boundary conditions).

Multigrid/domain decomposition is the natural choice to approximate A^{-1}.

But what about the Schur complement S?

Andy Wathen

Theorem (Fortin, 1970s)

For a stable discretisation, the Schur complement is *spectrally equivalent* to the scaled pressure mass matrix:

$$\underline{c} x^\top Q_\nu x \le x^\top S x \le \bar{c} x^\top Q_\nu x,$$

where

$$(Q_\nu)_{ij} = \int_\Omega \frac{\mathrm{Re}}{2} \psi_j \psi_i \, \mathrm{d}x.$$

David Silvester

For the Stokes equations, this gives a solver like:

- Krylov solver: GMRES
  - Block preconditioner (upper)
    - A: velocity operator solve
      - geometric multigrid
    - S: Schur complement solve
      - Auxiliary operator: mass matrix Q_ν
        - geometric multigrid

Good news!

This approach works very well for the Stokes equations!

Good news!

This approach works very well for the Stokes equations!

Bad news!

This doesn't work at all for the Navier–Stokes equations! No control over Schur complement.

Good news!

This approach works very well for the Stokes equations!

Bad news!

This doesn't work at all for the Navier–Stokes equations! No control over Schur complement.

Previous attempts

Different approximations for the Schur complement. They all break down at Reynolds number in the hundreds.

Good news!

This approach works very well for the Stokes equations!

Bad news!

This doesn't work at all for the Navier–Stokes equations! No control over Schur complement.

Previous attempts

Different approximations for the Schur complement. They all break down at Reynolds number in the hundreds.

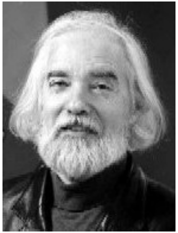
Challenge

How can we recover control of the Schur complement?

Section 2

Augmented Lagrangians

One idea is the *augmented Lagrangian* method.



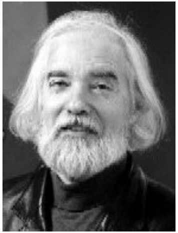
Michel Fortin

Roland Glowinski

One idea is the *augmented Lagrangian* method.

We augment the Lagrangian with a penalty term, $\gamma \geq 0$:

$$L_\gamma(u, p) = L(u, p) + \frac{\gamma}{2} \int_\Omega (\nabla \cdot u)^2 \, \mathrm{d}x.$$



Michel Fortin

Roland Glowinski

One idea is the *augmented Lagrangian* method.

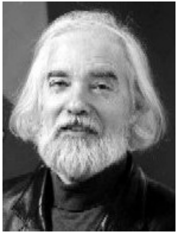
We augment the Lagrangian with a penalty term, $\gamma \geq 0$:

$$L_\gamma(u, p) = L(u, p) + \frac{\gamma}{2} \int_\Omega (\nabla \cdot u)^2 \, \mathrm{d}x.$$

The Schur complement is approximated by

$$S \sim (\frac{2}{\mathrm{Re}} + \gamma)^{-1} Q$$

with the spectral equivalence improving for larger γ.



Michel Fortin

Roland Glowinski

One idea is the *augmented Lagrangian* method.

We augment the Lagrangian with a penalty term, $\gamma \geq 0$:

$$L_\gamma(u, p) = L(u, p) + \frac{\gamma}{2} \int_\Omega (\nabla \cdot u)^2 \, \mathrm{d}x.$$

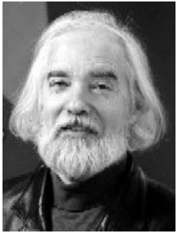
The Schur complement is approximated by

$$S \sim \left(\frac{2}{\mathrm{Re}} + \gamma\right)^{-1} Q$$

with the spectral equivalence improving for larger γ.

Augmented momentum equation

$$-\mathrm{div}\left(2\mathrm{Re}^{-1}\varepsilon(u)\right) + \mathrm{div}\,(u \otimes u) + \mathrm{grad}\, p - \underline{\gamma\, \mathrm{grad}\, \mathrm{div} u} = f$$



Michel Fortin

Roland Glowinski

This gives us control of the Schur complement, even for the Navier–Stokes equations:

γ	# iterations
0	$>$1000
1	10
10	6
100	4
1000	2
10000	2

This gives us control of the Schur complement, even for the Navier–Stokes equations:

γ	# iterations
0	>1000
1	10
10	6
100	4
1000	2
10000	2

Good news

The Schur complement approximation improves as γ increases.

The catch ...

... is that it makes the velocity solve **much** harder.

The catch ...

... is that it makes the velocity solve **much** harder.

The operator

$$A_{ij} = 2\mathrm{Re}^{-1} \int_\Omega \epsilon(\phi_j) : \epsilon(\phi_i) \; \mathrm{d}x$$

is very amenable to standard multigrid methods.

The catch ...

... is that it makes the velocity solve **much** harder.

The operator

$$A_{ij} = 2\mathrm{Re}^{-1} \int_\Omega \epsilon(\phi_j) : \epsilon(\phi_i) \ \mathrm{d}x$$

is very amenable to standard multigrid methods.

But even for Stokes, the augmented operator

$$(A_\gamma)_{ij} = 2\mathrm{Re}^{-1} \int_\Omega \epsilon(\phi_j) : \epsilon(\phi_i) \ \mathrm{d}x + \gamma \int_\Omega (\nabla \cdot \phi_j)(\nabla \cdot \phi_i) \ \mathrm{d}x$$

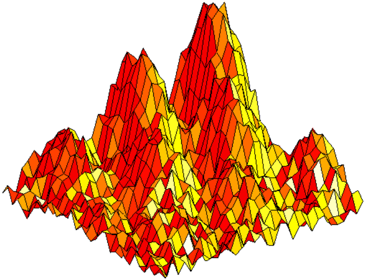
is very difficult to solve for $\gamma \gg \mathrm{Re}$.

Section 3

Solving the augmented block

Multigrid algorithm

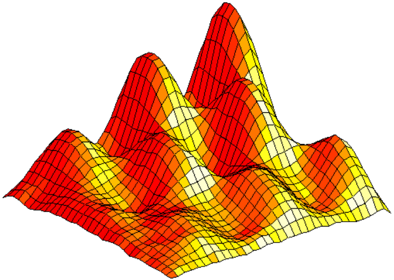
- Begin with an initial guess.



Error of initial guess.

Multigrid algorithm

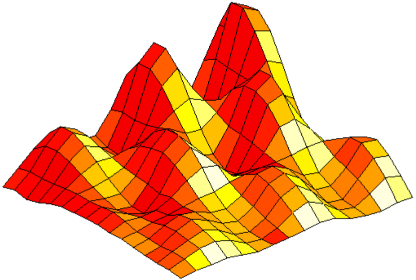
- Begin with an initial guess.
- Apply a *relaxation method* to smooth the error.



Error after relaxation.

Multigrid algorithm

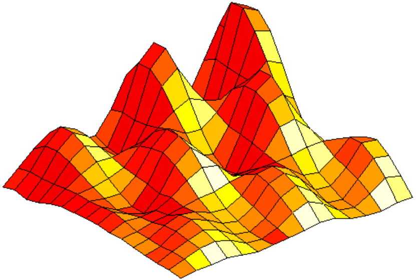
- Begin with an initial guess.
- Apply a *relaxation method* to smooth the error.
- Approximate the smooth error on a *coarse space*.



Error approximated on coarse grid.

Multigrid algorithm

- Begin with an initial guess.
- Apply a *relaxation method* to smooth the error.
- Approximate the smooth error on a *coarse space*.
- *Prolong* the error approximation to the fine grid and subtract.



Error approximated on coarse grid.

Building a geometric multigrid solver for A_γ *hinges on the kernel of* div.

Joachim Schöberl

Jinchao Xu

Building a geometric multigrid solver for A_γ *hinges on the kernel of* div.

Schöberl's theory (1999)

For a parameter-robust multigrid method, you need:

- *kernel-capturing multigrid relaxation*;
- *kernel-mapping prolongation*.

Joachim Schöberl

Jinchao Xu

Building a geometric multigrid solver for A_γ *hinges on the kernel of* div.

Joachim Schöberl

Schöberl's theory (1999)

For a parameter-robust multigrid method, you need:

- *kernel-capturing multigrid relaxation*;
- *kernel-mapping prolongation*.

Schöberl's theory applies to symmetric problems with singular terms. But amazingly **it works even for much harder problems**!

Jinchao Xu

Building a geometric multigrid solver for A_γ *hinges on the kernel of* div.

Joachim Schöberl

Schöberl's theory (1999)

For a parameter-robust multigrid method, you need:

- *kernel-capturing multigrid relaxation*;
- *kernel-mapping prolongation*.

Schöberl's theory applies to symmetric problems with singular terms. But amazingly **it works even for much harder problems**!

Today we will only discuss the relaxation, since that is all we need.

Jinchao Xu

Consider the variational problem: find $u \in V$, $\dim(V) < \infty$, such that

$$a(u, v) = (f, v) \quad \forall v \in V.$$

Consider the variational problem: find $u \in V$, $\dim(V) < \infty$, such that

$$a(u, v) = (f, v) \quad \forall v \in V.$$

The way we design relaxation methods is via *subspace correction*.

Subspace correction method

Choose an initial guess u_k and a space decomposition

$$V = \sum_i V_i.$$

Consider the variational problem: find $u \in V$, $\dim(V) < \infty$, such that

$$a(u, v) = (f, v) \quad \forall v \in V.$$

The way we design relaxation methods is via *subspace correction*.

Subspace correction method

Choose an initial guess u_k and a space decomposition

$$V = \sum_i V_i.$$

Solve for error approximations: for each i, find $V_i \ni e_i \approx u - u_k$ such that

$$a(e_i, v) = a(u, v) - a(u_k, v) = (f, v) - a(u_k, v) \quad \forall v \in V_i.$$

Consider the variational problem: find $u \in V$, $\dim(V) < \infty$, such that

$$a(u, v) = (f, v) \quad \forall v \in V.$$

The way we design relaxation methods is via *subspace correction*.

Subspace correction method

Choose an initial guess u_k and a space decomposition

$$V = \sum_i V_i.$$

Solve for error approximations: for each i, find $V_i \ni e_i \approx u - u_k$ such that

$$a(e_i, v) = a(u, v) - a(u_k, v) = (f, v) - a(u_k, v) \quad \forall v \in V_i.$$

Then combine the updates with weights:

$$u_{k+1} = u_k + \sum_i w_i(e_i).$$

Examples:

Jacobi

Let $V = \mathrm{span}(\phi_1, \ldots, \phi_N)$. Taking

$$V_i = \mathrm{span}(\phi_i)$$

yields the usual Jacobi iteration.

Examples:

Jacobi

Let $V = \mathrm{span}(\phi_1, \ldots, \phi_N)$. Taking

$$V_i = \mathrm{span}(\phi_i)$$

yields the usual Jacobi iteration.

Domain decomposition

If you partition the domain into overlapping $\Omega = \Omega_1 \cup \Omega_2 \cup \cdots \Omega_N$ and take

$$V_i = \{\text{functions in } V \text{ supported on } \Omega_i\}$$

you get a classical domain decomposition method.

Kernel-capturing multigrid relaxation

Now consider the problem: for $\alpha, \beta > 0$, find $u \in V$ such that

$$\alpha a(u, v) + \beta b(u, v) = (f, v) \quad \forall v \in V,$$

where a is symmetric coercive and b is symmetric positive semidefinite.

Kernel-capturing multigrid relaxation

Now consider the problem: for $\alpha, \beta > 0$, find $u \in V$ such that

$$\alpha a(u, v) + \beta b(u, v) = (f, v) \quad \forall v \in V,$$

where a is symmetric coercive and b is symmetric positive semidefinite.

For Stokes with augmented Lagrangian, we have

$$a(u, v) = \int_\Omega \epsilon(u) : \epsilon(v) \ \mathrm{d}x, \quad b(u, v) = \int_\Omega \operatorname{div} u \ \operatorname{div} v \ \mathrm{d}x.$$

Kernel-capturing multigrid relaxation

Now consider the problem: for $\alpha, \beta > 0$, find $u \in V$ such that

$$\alpha a(u, v) + \beta b(u, v) = (f, v) \quad \forall v \in V,$$

where a is symmetric coercive and b is symmetric positive semidefinite.

Theorem [Schöberl (1999), Lee, Wu, Xu, Zikatanov (2007)]

Define the kernel of the semidefinite term

$$\mathcal{N} = \{u \in V : b(u, v) = 0 \; \forall v \in V\}.$$

If *the decomposition captures the kernel*

$$\mathcal{N} = \sum_i \mathcal{N} \cap V_i,$$

in a stable way then the convergence will be robust wrt α and β.

How do we decompose the kernel of the divergence operator?

The function spaces arising in the Navier–Stokes equations form a *complex*:

$$\mathbb{R} \xrightarrow{\mathrm{id}} H^2 \xrightarrow{\mathrm{curl}} H^1 \times H^1 \xrightarrow{\mathrm{div}} L^2 \xrightarrow{\mathrm{null}} 0.$$

Doug Arnold

Ralf Hiptmair

How do we decompose the kernel of the divergence operator?

The function spaces arising in the Navier–Stokes equations form a *complex*:

$$\mathbb{R} \xrightarrow{\mathrm{id}} H^2 \xrightarrow{\mathrm{curl}} H^1 \times H^1 \xrightarrow{\mathrm{div}} L^2 \xrightarrow{\mathrm{null}} 0.$$

Doug Arnold

Ralf Hiptmair

In other words . . .

On a simply connected domain, $\ker(\mathrm{div}) = \mathrm{range}(\mathrm{curl})$.

How do we decompose the kernel of the divergence operator?

The function spaces arising in the Navier–Stokes equations form a *complex*:

$$\mathbb{R} \xrightarrow{\text{id}} H^2 \xrightarrow{\text{curl}} H^1 \times H^1 \xrightarrow{\text{div}} L^2 \xrightarrow{\text{null}} 0.$$

Doug Arnold

Ralf Hiptmair

In other words . . .

On a simply connected domain, $\ker(\text{div}) = \text{range}(\text{curl})$.

Consequence

By studying the space to the left, we can understand $\ker(\text{div})$.

In 2D, for velocity degree $p < 4$, we don't know what the potential space is.

John Morgan

Ridgway Scott

In 2D, for velocity degree $p < 4$, we don't know what the potential space is.

But for $p \geq 4$, we do: it is given by the *Morgan–Scott element*.

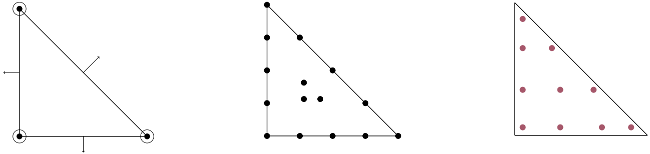
John Morgan

Ridgway Scott

In 2D, for velocity degree $p < 4$, we don't know what the potential space is.

But for $p \geq 4$, we do: it is given by the *Morgan–Scott element*.

$$
\begin{tikzcd}
H^2 \arrow[r, "\mathrm{curl}"] \arrow[d] & H^1 \times H^1 \arrow[r, "\mathrm{div}"] \arrow[d] & L^2 \arrow[d] \\
\mathbb{MS}_5 \arrow[r, "\mathrm{curl}"] \arrow[d] & \mathbb{CG}_4 \times \mathbb{CG}_4 \arrow[r, "\mathrm{div}"] \arrow[d] & \mathbb{DG}_3 \arrow[d] \\
{} & {} & {}
\end{tikzcd}
$$



John Morgan

Ridgway Scott

Why is this useful?

Why is this useful?

By exactness of the complex, if $u \in \mathbb{CG}_4$ and $\operatorname{div} u = 0$, then

$$u = \operatorname{curl} \phi, \quad \phi \in \mathbb{MS}_5.$$

Why is this useful?

By exactness of the complex, if $u \in \mathbb{CG}_4$ and $\operatorname{div} u = 0$, then

$$u = \operatorname{curl} \phi, \quad \phi \in \mathbb{MS}_5.$$

Let $\{\zeta_1, \ldots, \zeta_N\}$ be the (local) basis for $\mathbb{MS}_5$. Then we can write

$$\begin{aligned} u = \operatorname{curl} \phi &= \operatorname{curl} \sum_{i=1}^{N} c_i \zeta_i \\ &= \sum_{i=1}^{N} c_i \operatorname{curl} \zeta_i. \end{aligned}$$

Why is this useful?

By exactness of the complex, if $u \in \mathbb{CG}_4$ and $\operatorname{div} u = 0$, then

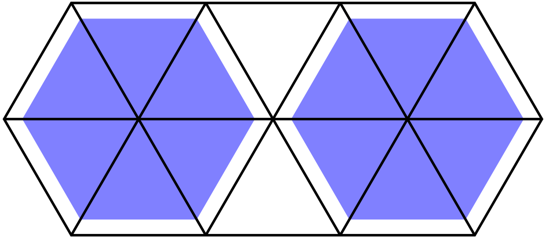
$$u = \operatorname{curl} \phi, \quad \phi \in \mathbb{MS}_5.$$

Let $\{\zeta_1, \ldots, \zeta_N\}$ be the (local) basis for $\mathbb{MS}_5$. Then we can write

$$\begin{aligned} u = \operatorname{curl} \phi &= \operatorname{curl} \sum_{i=1}^{N} c_i \zeta_i \\ &= \sum_{i=1}^{N} c_i \operatorname{curl} \zeta_i. \end{aligned}$$

This tells us that a good idea for a space decomposition is one that captures each ζ_i in a single subspace.

This motivates the *vertex-star* space decomposition.



In our space decomposition

$$V = \sum V_i,$$

we construct each V_i by

$$V_i = \{\text{all functions supported on the patch of cells around a vertex}\}.$$

With this knowledge, our solver diagram becomes

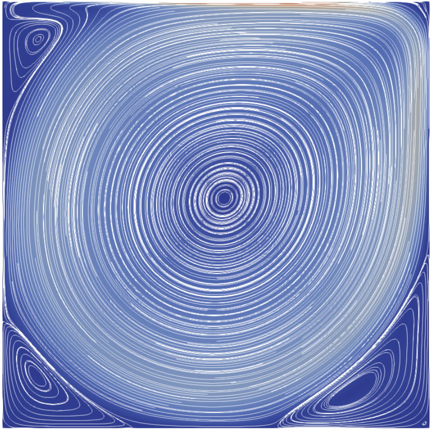
- Krylov solver: FGMRES
  - Block preconditioner (upper)
    - A_γ: geometric multigrid
      - h-fine
        - FGMRES
          - Vertex-star relaxation
      - h-coarse
        - LU factorisation
    - S_γ: Schur complement solve
      - Auxiliary pressure mass matrix
        - Exact inverse

Augmented Lagrangian multigrid solver for Navier–Stokes.

Section 4

Numerical results

2D lid-driven cavity



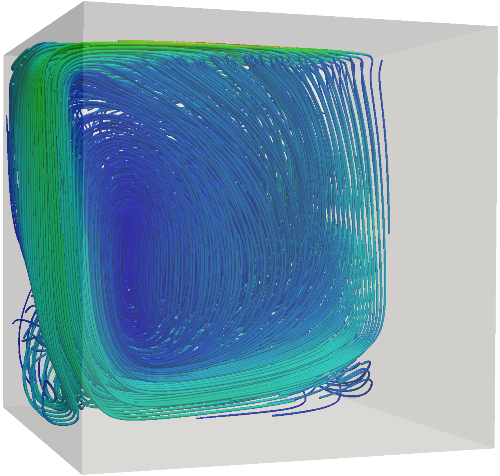
2D lid-driven cavity at $\mathrm{Re} = 5000$

Numerical results in 2D

# refinements	# dofs	Reynolds number				
		10	100	1000	5000	10000
Lid Driven Cavity						
1	9.3×10^4	2.50	2.33	2.33	5.50	8.50
2	3.7×10^5	2.00	2.00	2.00	4.00	6.00
3	1.5×10^6	2.00	1.67	1.67	2.50	3.50
4	5.9×10^6	2.00	1.67	1.50	1.50	4.00
Backwards Facing Step						
1	1.0×10^6	2.00	2.50	2.50	5.00	7.50
2	4.1×10^6	2.50	2.50	1.50	3.00	4.00
3	1.6×10^7	2.50	2.50	1.50	1.50	2.50

Table: Average outer Krylov iterations per Newton step for two 2D benchmark problems.

3D lid-driven cavity

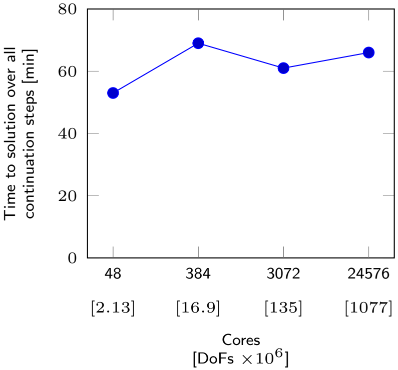


3D regularised lid-driven cavity at $\mathrm{Re} = 5000$

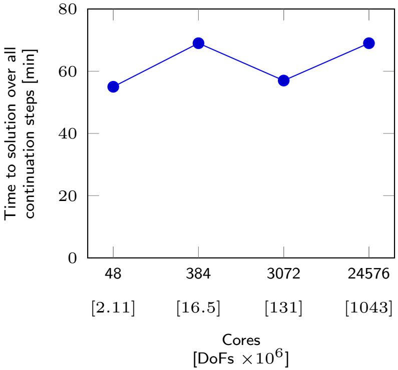
Numerical results in 3D

# refinements	# dofs	Reynolds number				
		10	100	1000	2500	5000
1	1.0×10^6	3.00	3.67	3.50	4.00	5.00
2	8.2×10^6	3.50	3.67	4.00	4.00	4.00
3	6.5×10^7	3.00	3.33	3.50	3.50	4.00

Table: Average outer Krylov iterations per Newton step for the 3D lid driven cavity.



(a) 3D lid-driven cavity



(b) 3D backwards-facing step

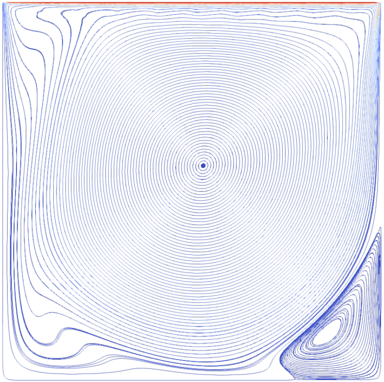
Weak scaling efficiency ...

...of 80% on ARCHER2 up to 25K cores with 1 billion degrees of freedom.

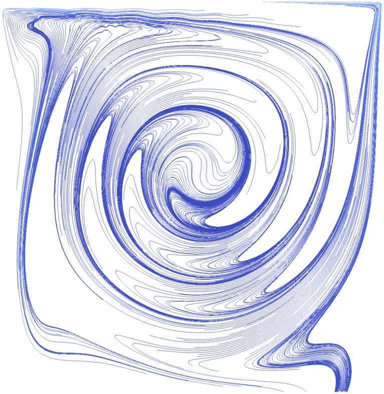
Section 5

Magnetohydrodynamics

2D lid-driven cavity



u



B

2D lid-driven cavity at $\mathrm{Rem} = 5000$, $\mathrm{Re} = 5000$

Numerical results for 3D lid-driven cavity

Rem\Re	1	1,000	10,000
1	6.0	4.3	4.3
1,000	4.5	3.0	3.0
10,000	4.5	5.5	5.7

Average outer Krylov iterations per Newton step.

Conclusions

Main toolkit

Block preconditioning + augmented Lagrangians + subspace correction + Hilbert complexes.

Can use these techniques to build preconditioners for

Conclusions

Main toolkit

Block preconditioning + augmented Lagrangians + subspace correction + Hilbert complexes.

Can use these techniques to build preconditioners for

- complex and coupled physical problems

Conclusions

Main toolkit

Block preconditioning + augmented Lagrangians + subspace correction + Hilbert complexes.

Can use these techniques to build preconditioners for

- complex and coupled physical problems
- with much greater parameter robustness than previously achieved.