From symbolic mathematics to fast solvers for finite element problems

Lawrence Mitchell[1,*]

C.J. Cotter, P.E. Farrell, T.H. Gibson, D.A. Ham, P.H.J. Kelly, R.C. Kirby, M.G. Knepley, E.H. Müller, F. Wechsung, ...

6th January 2020

[1]Department of Computer Science, Durham University
[*]lawrence.mitchell@durham.ac.uk

Outline

- Overview
  - Automated finite elements
  - Multigrid preconditioning for numerical weather prediction
- In depth
  - Parameter-robust preconditioning for the Navier–Stokes equations
- Future directions

Automated finite elements

Code that captures mathematical structure

Find $(u, p, T) \in V \times W \times Q$ s.t.

$$\int \nabla u \cdot \nabla v + (u \cdot \nabla u) \cdot v$$

$$-p\nabla \cdot v + \frac{\mathrm{Ra}}{\mathrm{Pr}} T g\hat{z} \cdot v \,\mathrm{d}x = 0$$

$$\int \nabla \cdot u q \,\mathrm{d}x = 0$$

$$\int (u \cdot \nabla T) S + \mathrm{Pr}^{-1} \nabla T \cdot \nabla S \,\mathrm{d}x = 0$$

$$\forall\, (v, q, T) \in V \times W \times Q$$

```
from firedrake import *
mesh = Mesh(...)
V = VectorFunctionSpace(mesh, "CG", 2)
W = FunctionSpace(mesh, "CG", 1)
Q = FunctionSpace(mesh, "CG", 1)
Z = V * W * Q
Ra = Constant(...)
Pr = Constant(...)
upT = Function(Z)
u, p, T = split(upT)
v, q, S = TestFunctions(Z)
bcs = [...]

F = (inner(grad(u), grad(v))
     + inner(dot(grad(u), u), v)
     - inner(p, div(v))
     + (Ra/Pr)*inner(T*g, v)
     + inner(div(u), q)
     + inner(dot(grad(T), u), S)
     + (1/Pr)*inner(grad(T), grad(S)))*dx

solve(F == 0, upT, bcs=bcs)
```

The computer should work for *you*

- Mathematics just says “here is the integral to compute on each element, do that everywhere”
- Code specifies an *implementation*

Assertion(s)

Having chosen a discretisation, writing the element integral is “mechanical”.

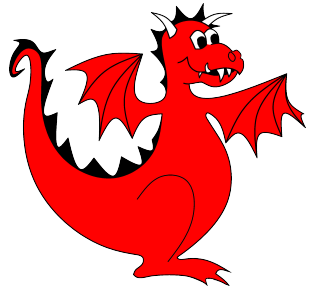
With an element integral in hand, integrating over a mesh is “mechanical”.

Corollary

Computers are good at mechanical things, why don’t we get the computer to write them for us?

Firedrake `www.firedrakeproject.org`

[...] *an automated system for the solution of partial differential equations using the finite element method.*



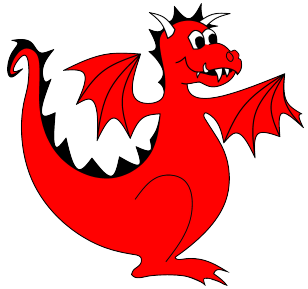
- Written in Python.
- Finite element problems specified with *embedded* domain specific language, UFL (Alnæs, Logg, Ølgaard, Rognes, and Wells 2014) from the FEniCS project.
- Domain-specifc optimising compiler.
- *Runtime* compilation to optimised, low-level (C) code.

F. Rathgeber, D.A. Ham, **L. Mitchell**, M. Lange, F. Luporini, A.T.T. McRae, G.-T. Bercea, G.R. Markall, P.H.J. Kelly.

ACM Transactions on Mathematical Software (2016). `arXiv:1501.01809[cs.MS]`

Firedrake `www.firedrakeproject.org`

[...] *an automated system for the solution of partial differential equations using the finite element method.*

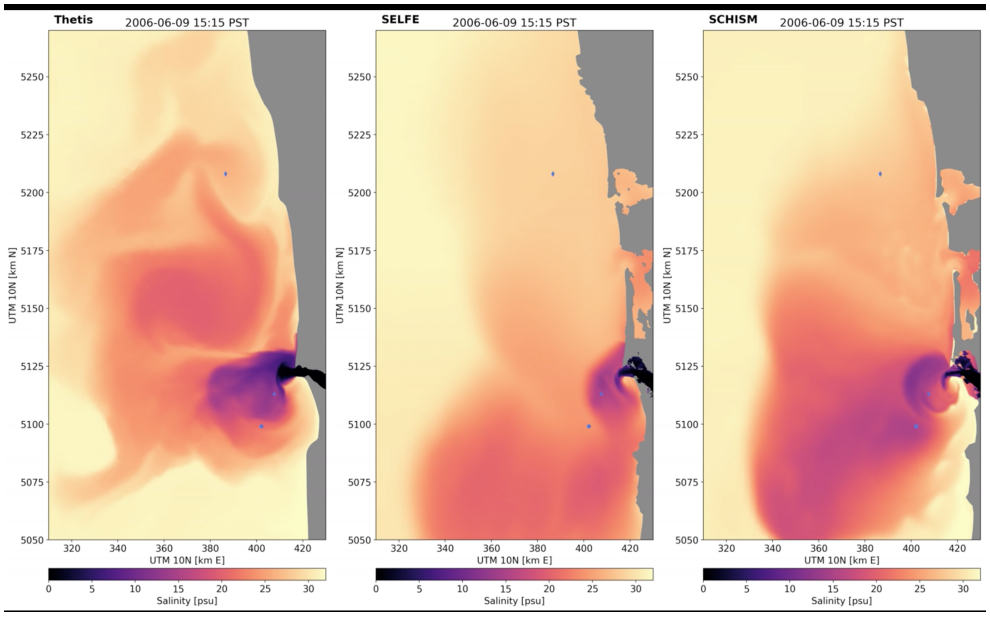


User groups at

Imperial, Oxford, Bath, Leeds, Durham, Kiel, Rice, Houston, Exeter, Buffalo, Waterloo, Washington, Baylor, Edinburgh, ...

Fourth annual (and first USA) user meeting this February (Seattle).

Application: coastal ocean modelling



Surface salinity of the Columbia river plume. Credit T. Kärnä, Finnish Meteorological Institute

- Lower numerical mixing, improved results.
- 3-8x faster than models with similar quality of results.
- Differentiable: efficient adjoint.
- Used at Finnish Met Institute; Earth Sciences at Imperial; Institute for Infrastructure and Environment (here).
- `thetisproject.org`

T. Kärnä, S.C. Kramer, **L. Mitchell**, D.A. Ham, M.D. Piggott, A.M. Baptista. Geoscientific Model Development (2018). `arXiv:1711.08552[physics.ao-ph]`

Application: numerical weather prediction

$$\begin{pmatrix} M_2 & -\frac{\Delta t}{2} D^T & -\frac{\Delta t}{2} Q \\ \frac{\Delta t}{2} c^2 D & M_3 & 0 \\ \frac{\Delta t}{2} N^2 Q^T & 0 & M_b \end{pmatrix} \begin{pmatrix} \vec{U} \\ \vec{P} \\ \vec{B} \end{pmatrix} = \begin{pmatrix} M_2 \vec{R}_u \\ M_3 \vec{R}_p \\ M_b \vec{R}_b \end{pmatrix}$$

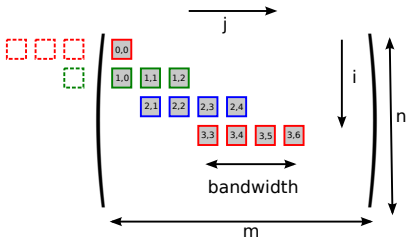
Challenge

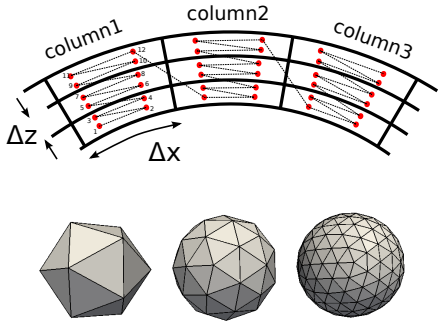
Invert elliptic operator fast enough for operational forecasting.

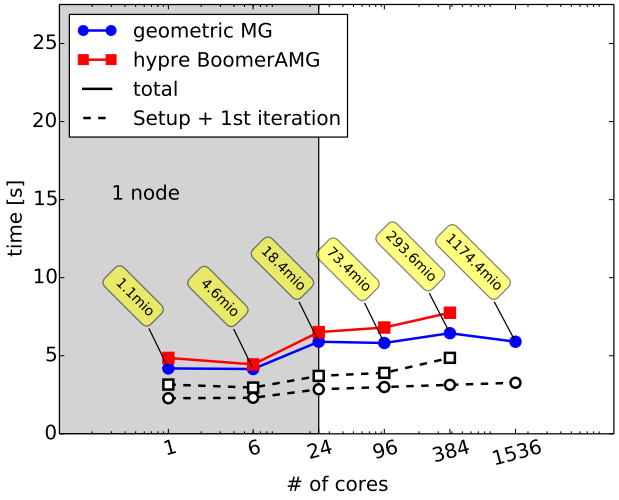
High aspect ratio: black-box numerical solvers struggle.

Need scalable, custom multigrid solver.

Multigrid approach







Developed custom tensor-product multigrid scheme, showed robustness and scalability.

L. Mitchell, E.H. Müller. JCP (2016). `arXiv:1605.00492[cs.MS]`

Future directions

Multilevel solvers for hybridised discretisations.

Approach is being explored by UK Met Office in their next generation dynamical core.

Preconditioning Navier–Stokes

Application challenge

Stationary incompressible Navier–Stokes

For $\nu \in \mathbb{R}_+$, find $(u, p) \in H^1(\Omega; \mathbb{R}^d) \times L^2(\Omega)$ such that

$$
\begin{aligned}
-\operatorname{div}(2\nu\varepsilon(u)) + (u \cdot \nabla)u + \nabla p &= f && \text{in } \Omega, \\
\operatorname{div} u &= 0 && \text{in } \Omega, \\
u &= g && \text{on } \Gamma_D, \\
2\nu\varepsilon(u) \cdot n &= pn && \text{on } \Gamma_N,
\end{aligned}
$$

Why?

Fully implicit discretisations of transient problem; bifurcation analysis of steady state.

Challenges for solvers

Desired properties

- Growth in time to solution is (at worst) $\mathcal{O}(n \log n)$ in number of degrees of freedom (resolution).
- Convergence does not degrade with $\nu \to 0$.

Direct methods

✓ Convergence independent of ν

✗ Time to solution $\mathcal{O}(n^2)$ (3D).

Krylov methods

✓ Time to solution $\mathcal{O}(n \log n)$ (with multilevel preconditioner)

✗ Convergence independent of ν challenging

Block preconditioning

Newton linearisation

$$P := \begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \delta u \\ \delta p \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}.$$

Block preconditioning

Newton linearisation

$$P := \begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \delta u \\ \delta p \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}.$$

Block factorisations

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

PDE-specific challenge

Find good approximations $\tilde{A}^{-1}$ for A^{-1} and $\tilde{S}^{-1}$ for S^{-1}.

Choosing $\tilde{A}^{-1}$ and $\tilde{S}^{-1}$

Stokes (no advection) Silvester and A. J. Wathen (1994)

Multigrid for $\tilde{A}^{-1}$, $\tilde{S}^{-1} = -\nu M_p^{-1}$

- ✓ h-independent
- ✗ Only effective up to $\mathsf{Re} \sim 10$ for Navier–Stokes.

PCD for Navier–Stokes Kay, Loghin, and A. Wathen (2002)

Multigrid for $\tilde{A}^{-1}$, approximate $\tilde{S}^{-1}$ with convection-diffusion solves on pressure space.

- ✓ h-independent
- ✗ Only effective up to $\mathsf{Re} \sim$ 100-1000

Performance of pressure convection-diffusion (PCD)

$1/h$	# degrees of freedom	Reynolds number		
		10	100	1000
2^4	8.34×10^2	22.0	40.4	103.3
2^5	3.20×10^3	23.0	41.3	137.7
2^6	1.25×10^4	24.5	42.0	157.0
2^7	4.97×10^4	25.5	42.7	149.0
2^8	1.98×10^5	26.0	44.0	137.0

Table 1: Average number of outer Krylov iterations per Newton step for the 2D regularized lid-driven cavity problem with PCD preconditioner. Obtained with IFISS v3.5, $Q_1 - P_0$ element pair.

Augmented Lagrangian approach

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2095–2113

AN AUGMENTED LAGRANGIAN-BASED APPROACH TO THE OSEEN PROBLEM*

MICHELE BENZI† AND MAXIM A. OLSHANSKII‡

Abstract. We describe an effective solver for the discrete Oseen problem based on an augmented Lagrangian formulation of the corresponding saddle point system. The proposed method is a block triangular preconditioner used with a Krylov subspace iteration like BiCGStab. The crucial ingredient is a novel multigrid approach for the (1,1) block, which extends a technique introduced by Schöberl for elasticity problems to nonsymmetric problems. Our analysis indicates that this approach results in fast convergence, independent of the mesh size and largely insensitive to the viscosity. We present experimental evidence for both isoP2-P0 and isoP2-P1 finite elements in support of our conclusions. We also show results of a comparison with two state-of-the-art preconditioners, showing the competitiveness of our approach.

Augmented Lagrangian approach

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2095–2113

AN AUGMENTED LAGRANGIAN-BASED APPROACH TO THE OSEEN PROBLEM*

MICHELE BENZI† AND MAXIM A. OLSHANSKII‡

Abstract. We describe an effective solver for the discrete Oseen problem based on an augmented Lagrangian formulation of the corresponding saddle point system. The proposed method is a block triangular preconditioner used with a Krylov subspace iteration like BiCGStab. The crucial ingredient is a novel multigrid approach for the (1,1) block, which extends a technique introduced by Schöberl for elasticity problems to nonsymmetric problems. Our analysis indicates that this approach results in fast convergence, independent of the mesh size and largely insensitive to the viscosity. We present experimental evidence for both isoP2-P0 and isoP2-P1 finite elements in support of our conclusions. We also show results of a comparison with two state-of-the-art preconditioners, showing the competitiveness of our approach.

Observation

✓ Viscosity-robust preconditioning

✗ Only 2D, no further implementation until 2018

Objectives

- First general implementation of the method. ✓
- Extend the solver and discretisation to three dimensions. ✓
- Extend approach to *divergence-free* discretisations. ✓

Three ideas

1. Control Schur complement with an augmented Lagrangian term
2. Kernel-capturing multigrid relaxation
3. Robust multigrid prolongation

P.E. Farrell, **L. Mitchell**, F. Wechsung. SIAM SISC (2019). `arXiv:1810.03315[math.NA]`

Controlling the Schur complement

Continuous augmentation

Add γ **grad div** u term to the momentum equation.

Doesn't change solution since **div** $u = 0$

Theorem (Hestenes, Fortin, Glowinski, Olshanksii, ...)

As $\gamma \to \infty$, the Schur complement is well approximated by $\tilde{S}^{-1} = -(\nu + \gamma) M_p^{-1}$.

Controlling the Schur complement

Continuous augmentation

Add γ **grad div** u term to the momentum equation.

Doesn't change solution since **div** $u = 0$

Theorem (Hestenes, Fortin, Glowinski, Olshanksii, ...)

As $\gamma \to \infty$, the Schur complement is well approximated by $\tilde{S}^{-1} = -(\nu + \gamma)M_p^{-1}$.

Discrete augmentation

$$\begin{pmatrix} A + \gamma B^T M_p^{-1} B & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}$$

Doesn't change solution since $Bu = 0$.

Again, as $\gamma \to \infty$, S^{-1} is well approximated by $\tilde{S}^{-1} = -(\nu + \gamma)M_p^{-1}$.

Conservation of misery

Good news

The Schur complement becomes easy to approximate as $\gamma \to \infty$

Bad news

Normal multigrid approaches for $\tilde{A}^{-1}$ don't work for $A_\gamma := A + \gamma B^T M_p^{-1} B$, and get *worse* as $\gamma \to \infty$.

Conservation of misery

Good news

The Schur complement becomes easy to approximate as $\gamma \to \infty$

Bad news

Normal multigrid approaches for $\tilde{A}^{-1}$ don't work for $A_\gamma := A + \gamma B^T M_p^{-1} B$, and get *worse* as $\gamma \to \infty$.

	LU for $\tilde{A}_\gamma^{-1}$	AMG for $\tilde{A}_\gamma^{-1}$
$\gamma = 10^{-1}$	15	18
$\gamma = 1$	6	40
$\gamma = 10^{1}$	3	107

Outer Krylov iterations for different choices of $\tilde{A}_\gamma^{-1}$.

Point smoother is not sufficient

- Ignorning advection, then the top-left block corresponds to discretisation of

$$a_\gamma(u, v) = \underbrace{\int_\Omega 2\nu\varepsilon(u) : \varepsilon(v)\, \mathrm{d}x}_{\text{sym. pos. def.}} + \underbrace{\int_\Omega \gamma\, \mathrm{div}\, u\, \mathrm{div}\, v\, \mathrm{d}x}_{\text{sym. pos. semi-def.}}$$

Point smoother is not sufficient

- Ignorning advection, then the top-left block corresponds to discretisation of

$$a_\gamma(u, v) = \underbrace{\int_\Omega 2\nu\varepsilon(u) : \varepsilon(v)\,\mathrm{d}x}_{\text{sym. pos. def.}} + \underbrace{\int_\Omega \gamma\,\mathrm{div}\,u\,\mathrm{div}\,v\,\mathrm{d}x}_{\text{sym. pos. semi-def.}}$$

- The semi-definite term is singular on all solenoidal fields $\Rightarrow$ the system becomes *nearly singular* as $\gamma \to \infty$

Point smoother is not sufficient

- Ignorning advection, then the top-left block corresponds to discretisation of

$$a_\gamma(u, v) = \underbrace{\int_\Omega 2\nu\varepsilon(u) : \varepsilon(v)\,\mathrm{d}x}_{\text{sym. pos. def.}} + \underbrace{\int_\Omega \gamma\,\mathrm{div}\,u\,\mathrm{div}\,v\,\mathrm{d}x}_{\text{sym. pos. semi-def.}}$$

- The semi-definite term is singular on all solenoidal fields $\Rightarrow$ the system becomes *nearly singular* as $\gamma \to \infty$
- To build a γ-robust scheme we need (Schöberl 1999)
  - a γ-robust smoother;
  - a prolongation operator with γ-independent continuity constant;

  $\Rightarrow$ overlapping Schwarz smoother with decomposition that decomposes the kernel. “The right blocks for block Jacobi”.

Discretisation requirements

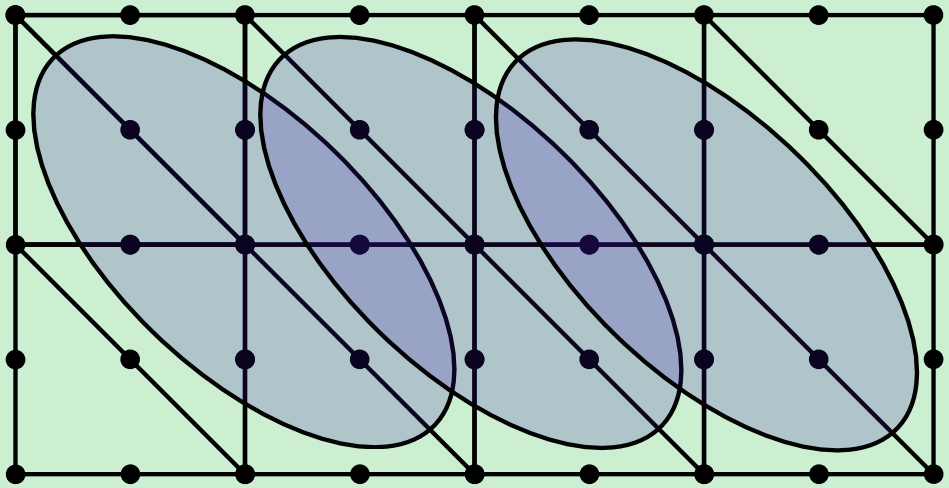
- Need inf-sup stable pair for velocity and pressure
- Need a *local basis* for the kernel of div (for efficiency).

Low-order

2D $P_2^2 - P_0$

3D $(P_1 \oplus \mathrm{FB})^3 - P_0$ (FB cubic face bubbles)

$\Rightarrow$ overlapping *star* patches



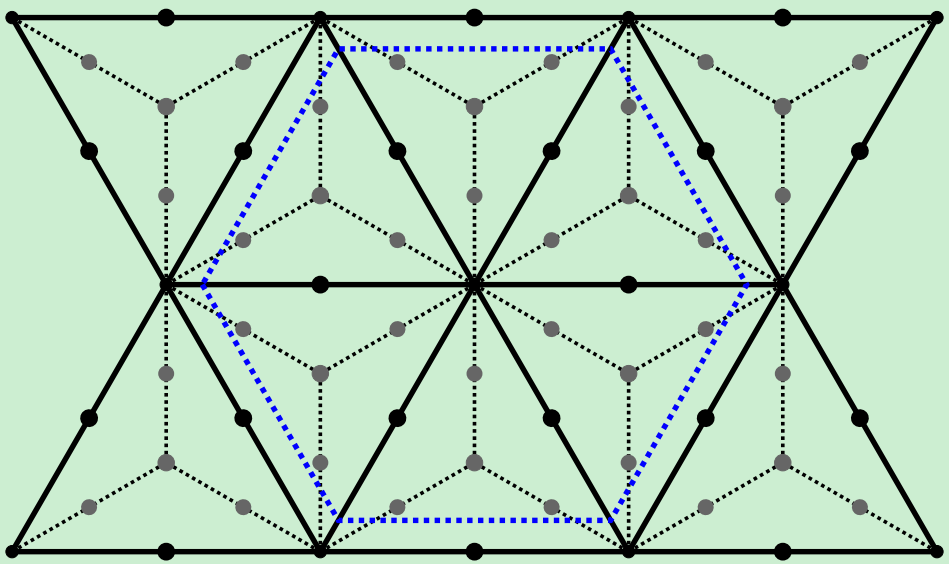
Divergence-free pair

Choose element pair from discrete subcomplex of Stokes complex

$$H^2 \xrightarrow{\mathsf{grad}} H^1(\mathsf{curl}) \xrightarrow{\mathsf{curl}} H^1 \xrightarrow{\mathsf{div}} L^2.$$

Divergence-free

- Scott–Vogelius pair: $P_k^d - P_{k-1}^{\mathrm{disc}}$ on barycentrically refined meshes ($k \geq d$).

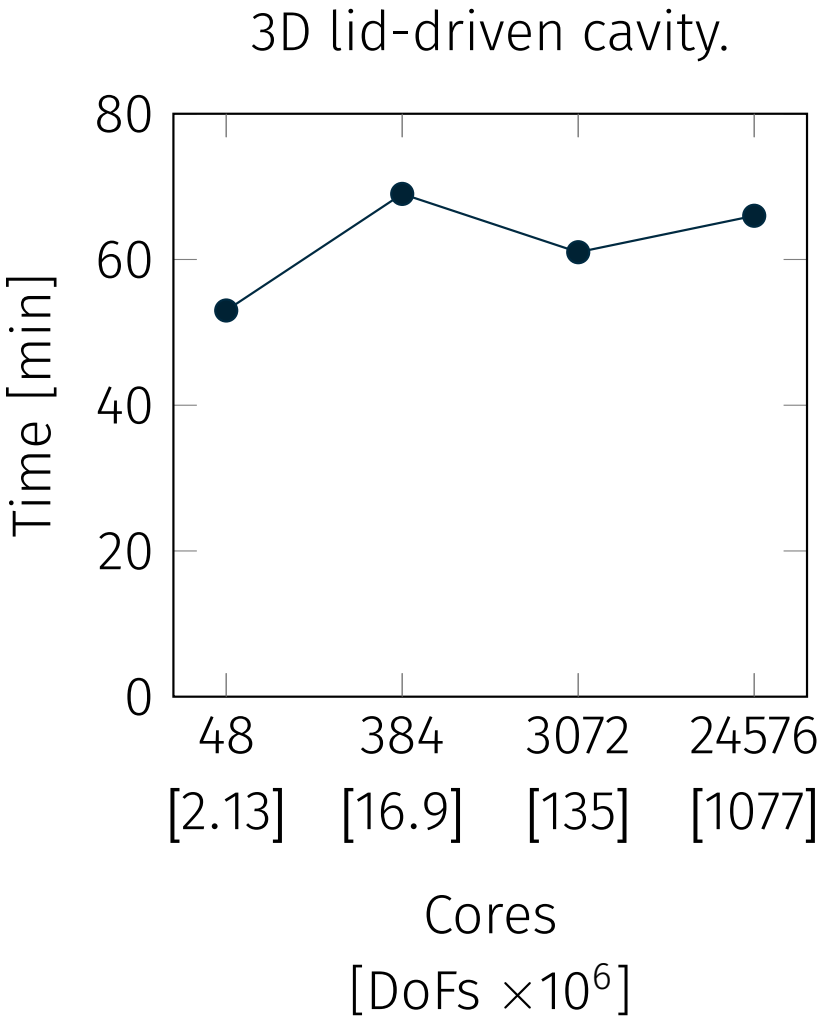


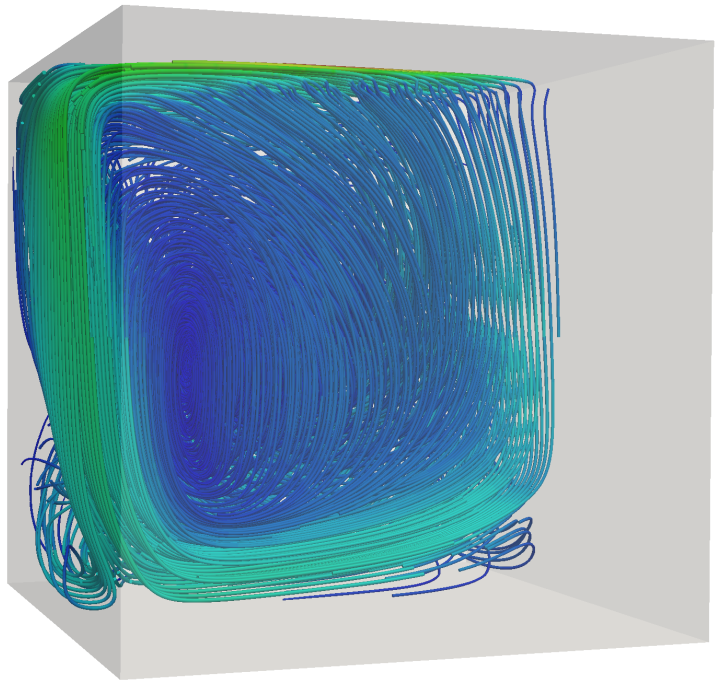
Numerical results — 3D $(P_1 \oplus \mathrm{FB})^3 - P_0$

# ref.	# dofs	Reynolds number				
		10	100	1000	2500	5000
Lid Driven Cavity						
1	2.1×10^6	4.50	4.00	5.00	4.50	4.00
2	1.7×10^7	4.50	4.33	4.50	4.00	4.00
3	1.3×10^8	4.50	4.33	4.00	3.50	7.00
4	1.1×10^9	4.50	3.66	3.00	5.00	5.00
Backwards facing step						
1	2.1×10^6	4.50	4.00	4.00	4.50	7.50
2	1.7×10^7	5.00	4.00	3.33	4.00	10.00
3	1.3×10^8	6.50	4.50	3.50	3.00	8.00
4	1.0×10^9	7.50	3.50	2.50	3.00	6.00

Table 2: Average number of outer Krylov iterations per Newton step for two 3D benchmark problems.

Computational performance — 3D





Velocity streamlines at Reynolds number 5000. Credit F. Wechsung, University of Oxford.

50 continuation steps

Numerical results — 2D $P_3^2 - P_2^{\text{disc}}$

# ref.	# 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 3: Average number of outer Krylov iterations per Newton step for two 2D benchmark problems.

Main result

It is possible to solve the Navier–Stokes equations in a Reynolds-robust way!

Even for exactly divergence-free discretisations.

Open source implementation at `https://github.com/florianwechsung/alfi`

Enabling mathematical software

General implementation enabled by new mathematical software developments:

- Extensible block-preconditioning framework in Firedrake

  R.C. Kirby, **L. Mitchell**. SIAM SISC (2018). `arXiv:1706.01346[cs.MS]`

- New preconditioner in PETSc and Firedrake for multigrid relaxation schemes based on subspace correction methods

  P.E. Farrell, M.G. Knepley, **L. Mitchell**, F. Wechsung. `arXiv:1912.08516[cs.MS]`

Future directions

Idea

- Mathematics is the language used to derive optimal solvers.
- It *should be* the language we use to describe their implementation.

Challenges

- Is there appropriate language to describe these problems to a computer?
- Not everything is a PDE, what then?

Rewards

- Bring state-of-the-art numerical solvers to the masses.
- Enable more exploratory, and creative, mathematics.

Summary

- Broad interdisciplinary research agenda spanning:
  - finite element discretisation and implementation;
  - linear and non-linear solvers/preconditioners;
  - high-performance computing;
  - domain-specific compilers.
- Application areas:
  - coastal ocean & freshwater outflow;
  - renewable energy;
  - numerical weather prediction.

References i

- Alnæs, M. S., A. Logg, K. B. Ølgaard, M. E. Rognes, and G. N. Wells (2014). "Unified Form Language: A Domain-specific Language for Weak Formulations of Partial Differential Equations". *ACM Trans. Math. Softw.* **40**. doi:10.1145/2566630. arXiv: 1211.4047 [cs.MS].
- Farrell, P. E., M. G. Knepley, L. Mitchell, and F. Wechsung (2019). *PCPATCH: software for the topological cosntruction of multigrid relaxation methods*. Submitted to ACM Transactions on Mathematical Software. arXiv: 1912.08516 [cs.MS].
- Farrell, P. E., L. Mitchell, and F. Wechsung (2019). "An augmented Lagrangian preconditioner for the 3D stationary incompressible Navier–Stokes equations at high Reynolds number". *SIAM Journal on Scientific Computing* **41**. doi:10.1137/18M1219370. arXiv: 1810.03315 [math.NA].
- Kärnä, T. et al. (2018). "Thetis coastal ocean model: discontinuous Galerkin discretization for the three-dimensional hydrostatic equations". *Geoscientific Model Development* **11**. doi:10.5194/gmd-11-4359-2018. arXiv: 1711.08552 [physics.ao-ph].
- Kay, D., D. Loghin, and A. Wathen (2002). "A preconditioner for the steady-state Navier-Stokes equations". *SIAM Journal on Scientific Computing* **24**. doi:10.1137/S106482759935808X.
- Kirby, R. C. and L. Mitchell (2018). "Solver composition across the PDE/linear algebra barrier". *SIAM Journal on Scientific Computing* **40**. doi:10.1137/17M1133208. arXiv: 1706.01346 [cs.MS].
- Mitchell, L. and E. H. Müller (2016). "High level implementation of geometric multigrid solvers for finite element problems: applications in atmospheric modelling". *Journal of Computational Physics* **327**. doi:10.1016/j.jcp.2016.09.037. arXiv: 1605.00492 [cs.MS].

References ii

- Rathgeber, F. et al. (2016). “Firedrake: automating the finite element method by composing abstractions”. *ACM Transactions on Mathematical Software* **43**. doi:10.1145/2998441. arXiv: 1501.01809 [cs.MS].
- Schöberl, J. (1999). “Multigrid methods for a parameter dependent problem in primal variables”. *Numerische Mathematik* **84**. doi:10.1007/s002110050465.
- Silvester, D. and A. J. Wathen (1994). “Fast iterative solution of stabilised Stokes systems. Part II: Using general block preconditioners”. *SIAM Journal on Numerical Analysis* **31**. doi:10.1137/0731070.