FEniCS PROJECT

Hybridized discontinuous Galerkin methods for the Stokes and Navier-Stokes equations in FEniCSx: non-simplex cells and curved geometries

Joseph P. Dean[1] Sander Rhebergen[2] Chris N. Richardson[1] Garth N. Wells[1]

[1]*University of Cambridge* [2]*University of Waterloo*

Outline

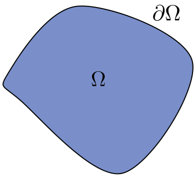
Problem statement

Stokes problem (weak form): *Given* $f \in [L^2(\Omega)]^d$, *find* $u \in V := [H^1_0(\Omega)]^d$ *and* $p \in Q := L^2_0(\Omega)$ *such that*

$$\begin{aligned} a(u, v) + b(v, p) &= F(v) \quad && \forall v \in V, \\ b(u, q) &= 0 && \forall q \in Q, \end{aligned}$$

where

$$a(u, v) := \int_\Omega \nu \nabla u : \nabla v \, \mathrm{d}x, \quad b(v, p) := -\int_\Omega p \nabla \cdot v \, \mathrm{d}x, \quad \textit{and} \quad F(v) := \int_\Omega f \cdot v \, \mathrm{d}x.$$



Some observations

1. The problem is well-posed and $\exists \beta > 0$ such that

$$\inf_{q \in Q} \sup_{v \in V} \frac{\int_\Omega q \nabla \cdot v \, dx}{||v||_{1,\Omega} ||q||_{0,\Omega}} \geq \beta$$

2. The following invariance property[1] holds:

$$f \to f + \nabla \phi \implies (u, p) \to (u, p + \phi)$$

[1] Volker John et al. "On the Divergence Constraint in Mixed Finite Element Methods for Incompressible Flows". In: *SIAM Review* 59.3 (2017), pp. 492–544. DOI: 10.1137/15m1047696.

Mass conservation?

Mass conservation (weak statement):

$$b(u, q) = 0 \quad \forall q \in Q$$

- The weak statement implies exact mass conservation, meaning $||\nabla \cdot u||_{0,\Omega} = 0$.

Mass conservation (discrete statement): *Let $u_h \in V_h \subset V$, then*

$$b(u_h, q_h) = 0 \quad \forall q_h \in Q_h \subset Q$$

- The discrete statement could imply global, local (cell), or exact mass conservation depending on V_h and Q_h. If $\nabla \cdot V_h \subseteq Q_h$, mass is conserved exactly.

With conforming methods, it is difficult to balance stability and incompressibility

Hybridized discontinuous Galerkin[2]

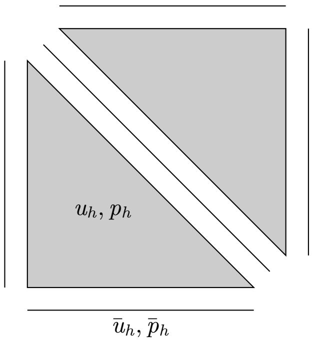
Let $\boldsymbol{u}_h := (u_h, \bar{u}_h) \in \boldsymbol{V}_h$ and $\boldsymbol{p}_h := (p_h, \bar{p}_h) \in \boldsymbol{Q}_h$, where $\boldsymbol{V}_h := V_h \times \bar{V}_h$, $\boldsymbol{Q}_h := Q_h \times \bar{Q}_h$, and

$$V_h := \left\{ v_h \in [L^2(\mathcal{T}_h)]^d ;\ v_h|_K \in V_h(K)\ \forall K \in \mathcal{T}_h \right\},$$

$$\bar{V}_h := \left\{ \bar{v}_h \in [L^2(\mathcal{F}_h)]^d ;\ \bar{v}_h|_F \in \bar{V}_h(F)\ \forall F \in \mathcal{F}_h,\ \bar{v}_h = 0 \text{ on } \partial\Omega \right\},$$

$$Q_h := \left\{ q_h \in L^2(\mathcal{T}_h);\ q_h|_K \in Q_h(K)\ \forall K \in \mathcal{T}_h \right\},$$

$$\bar{Q}_h := \left\{ \bar{q}_h \in L^2(\mathcal{F}_h);\ \bar{q}_h|_F \in \bar{Q}_h(F)\ \forall F \in \mathcal{F}_h \right\}.$$



[2] S. Rhebergen and G. N. Wells. "A hybridizable discontinuous Galerkin method for the Navier–Stokes equations with pointwise divergence-free velocity field". In: *J. Sci. Comput.* 76.3 (2018), pp. 1484–1501. DOI: 10.1007/s10915-018-0671-4.

HDG formulation

Stokes problem (HDG formulation): *Find* $(\boldsymbol{u}_h, \boldsymbol{p}_h) \in \boldsymbol{V}_h \times \boldsymbol{Q}_h$ *such that*

$$\begin{aligned} a_h(\boldsymbol{u}_h, \boldsymbol{v}_h) + b_h(v_h, \boldsymbol{p}_h) &= F(v_h) \quad && \forall \boldsymbol{v}_h \in \boldsymbol{V}_h, \\ b_h(u_h, \boldsymbol{q}_h) &= 0 && \forall \boldsymbol{q}_h \in \boldsymbol{Q}_h, \end{aligned}$$

where

$$\begin{aligned} a_h(\boldsymbol{u}_h, \boldsymbol{v}_h) := \sum_{K \in \mathcal{T}_h} \int_K \nu \nabla u_h : \nabla v_h \, \mathrm{d}x - \sum_{K \in \mathcal{T}_h} \int_{\partial K} \nu \left((u_h - \bar{u}_h) \cdot \partial_n v_h + \partial_n u_h \cdot (v_h - \bar{v}_h) \right) \mathrm{d}s \\ + \sum_{K \in \mathcal{T}_h} \int_{\partial K} \nu \frac{\alpha}{h_K} (u_h - \bar{u}_h) \cdot (v_h - \bar{v}_h) \, \mathrm{d}s, \end{aligned}$$

and

$$b_h(v_h, \boldsymbol{p}_h) := -\sum_{K \in \mathcal{T}_h} \int_K p_h \nabla \cdot v_h \, \mathrm{d}x + \sum_{K \in \mathcal{T}_h} \int_{\partial K} v_h \cdot n \bar{p}_h \, \mathrm{d}s.$$

Mapping functions

Let $\psi_K : V_h(K) \to V_h(\hat{K})$.

Lemma

If ψ_K is the pullback by the geometric mapping (as in the original method), and if $\nabla \cdot V_h(K) \subseteq Q_h(K)$ and $\bar{Q}_h(F) \supseteq \{v_h|_F \cdot n; v_h \in V_h(K)\}$, then the discrete velocity field is exactly divergence free.

Problem: what if the geometric mapping is not affine?

Lemma

If ψ_K is the contravariant Piola transform, then the above conditions can be relaxed; if $\nabla \cdot V_h(\hat{K}) \subseteq Q_h(\hat{K})$ and $\bar{Q}_h(\hat{F}) \supseteq \{\hat{v}_h|_{\hat{F}} \cdot \hat{n}; \hat{v}_h \in V_h(\hat{K})\}$ then the discrete velocity field is exactly divergence free.

A similar idea can be applied to Scott–Vogelius elements on curved domains.[3]

[3] Michael Neilan and M. Baris Otus. "Divergence–free Scott–Vogelius elements on curved domains". In: (2020), pp. 1–23. arXiv: 2008.06429. URL: http://arxiv.org/abs/2008.06429.

Suitable spaces

Simplex cells: *If $\hat{K}$ is the reference simplex and if ψ_K is the contravariant Piola transform, then the spaces*

$$V_h(\hat{K}) := [\mathbb{P}_k(\hat{K})]^d, \quad \bar{V}_h(\hat{F}) := [\mathbb{P}_k(\hat{F})]^d, \quad Q_h(\hat{K}) := \mathbb{P}_{k-1}(\hat{K}) \quad \textit{and} \quad \bar{Q}_h(\hat{F}) := \mathbb{P}_k(\hat{F})$$

give an exactly divergence free velocity field even if the geometric mapping is not affine.

Non-simplex cells: *If $\hat{K}$ is the reference quadrilateral or hexahedron and if ψ_K is the contravariant Piola transform, then the spaces*

$$V_h(\hat{K}) := \mathbb{RT}_k(\hat{K}), \quad \bar{V}_h(\hat{F}) := [\mathbb{Q}_k(\hat{F})]^d, \quad Q_h(\hat{K}) := \mathbb{Q}_k(\hat{K}), \quad \textit{and} \quad \bar{Q}_h(\hat{F}) := \mathbb{Q}_k(\hat{F})$$

give an exactly divergence free velocity field even if the geometric mapping is not affine.

More about the non-simplex case

- $H(\mathrm{div})$-conforming finite elements are introduced following the same ideas as divergence conforming DG[4] and HDG[5] methods.
- Other $H(\mathrm{div})$-conforming finite elements can be used, but care must be taken as some lose optimal order approximation in $[L^2(\Omega)]^d$ on general quadrilateral meshes.[6]

[4] Bernardo Cockburn, Guido Kanschat, and Dominik Schötzau. "A Note on Discontinuous Galerkin Divergence-free Solutions of the Navier-Stokes Equations". In: *Journal of Scientific Computing* 31.1-2 (2007), pp. 61–73. DOI: 10.1007/s10915-006-9107-7.

[5] Christoph Lehrenfeld and Joachim Schöberl. "High order exactly divergence-free Hybrid Discontinuous Galerkin Methods for unsteady incompressible flows". In: *Computer Methods in Applied Mechanics and Engineering* 307 (2016), pp. 339–361. DOI: 10.1016/j.cma.2016.04.025.

[6] Douglas N. Arnold, Daniele Boffi, and Richard S. Falk. "Quadrilateral H (div) Finite Elements". In: *SIAM Journal on Numerical Analysis* 42.6 (2005), pp. 2429–2451. DOI: 10.1137/S0036142903431924.

Static condensation

The block structure of the element tensor is of the form

$$\begin{bmatrix} A_{uu} & B_{pu}^T & A_{\bar{u}u}^T & B_{\bar{p}u}^T \\ B_{pu} & 0 & 0 & 0 \\ A_{\bar{u}u} & 0 & A_{\bar{u}\bar{u}} & 0 \\ B_{\bar{p}u} & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} U \\ P \\ \bar{U} \\ \bar{P} \end{pmatrix} = \begin{pmatrix} F_u \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Eliminating the cell degrees of freedom gives the condensed element tensor

$$\begin{bmatrix} A_{\bar{u}\bar{u}} - BA^{-1}B^T & -BA^{-1}C^T \\ -CA^{-1}B^T & -CA^{-1}C^T \end{bmatrix} \begin{pmatrix} \bar{U} \\ \bar{P} \end{pmatrix} = \begin{pmatrix} -BA^{-1}F \\ -CA^{-1}F \end{pmatrix},$$

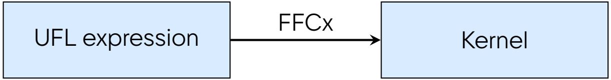
where

$$A = \begin{bmatrix} A_{uu} & B_{pu}^T \\ B_{pu} & 0 \end{bmatrix}, \; B = \begin{bmatrix} A_{\bar{u}u} & 0 \end{bmatrix}, \; C = \begin{bmatrix} B_{\bar{p}u} & 0 \end{bmatrix}, \text{ and } F = \begin{pmatrix} F_u \\ 0 \end{pmatrix}.$$

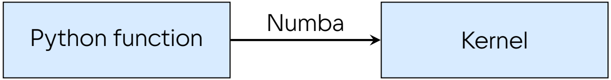
Implementation

Features of FEniCSx:

- Create kernels generated from UFL that are callable from python



- Create user defined kernels written in Python



- User defined kernels can call generated kernels

Implementation

Create kernels for each block of the element tensor (A_{uu}, ..., $A_{\bar{u}\bar{u}}$):

```
# UFL expressions for each block of the element tensor
A_uu_form = nu * inner(grad(u), grad(v)) * dx + nu * gamma * inner(u, v) * ds \
            - nu * (inner(u, dot(grad(v), n)) + inner(v, dot(grad(u), n))) * ds
...
A_ubar_ubar_form = nu * gamma * inner(ubar, vbar) * ds

# Compile forms with FFCx and expose to Python
forms = [A_uu_form, ..., A_ubar_ubar_form]
compiled_forms = ffcx.codegeneration.jit.compile_forms(forms)
A_uu_cell_kernel = compiled_forms[0][0].create_cell_integral().tabulate_tensor
A_uu_facet_kernel = \
    compiled_forms[0][0].create_exterior_facet_integral().tabulate_tensor
...

```

Implementation

Define a custom kernel to compute the top left block of the condensed element tensor $(K_{00} := A_{\bar{u}\bar{u}} - BA^{-1}B^T)$:

```python
@numba.cfunc(c_signature)
def tabulate_K00(K00_, w_, c_, coords_, entity_local_index, ...):
    K00 = numba.carray(K00_, (ubar_size, ubar_size))
    A_uu = np.zeros((u_size, u_size))
    ...
    # Compute cell integrals
    A_uu_cell_kernel(ffi.from_buffer(A_uu), w_, c_, coords_, entity_local_index, ...)
    ...
    for j in range(n_facets):
        # Compute facet integrals
        A_uu_facet_kernel(ffi.from_buffer(A_uu), w_, c_, coords_, fj, ...)
        ...
    # Static condensation
    K00 += A_ubar_ubar - B @ np.linalg.solve(A, B.T)

```

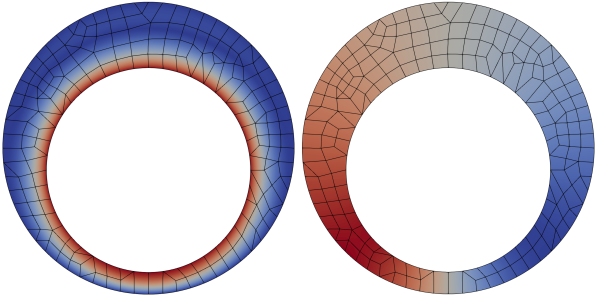
This kernel is passed to DOLFINx to assemble over the mesh.

Implementation: further work

- The above FEniCSx implementation has been tested on simplices.
- Until recently, FEniCSx did not have support for quadrilateral/hexahedral $H(\mathrm{div})$-conforming finite elements.
- Basix supports these elements, but some work is required to implement facet function spaces in a more general manner.
- To demonstrate the HDG scheme on meshes containing quadrilaterals, the method was also implemented in NGSolve.[7]

[7] Joachim Schöberl. "C++ 11 implementation of finite elements in NGSolve". In: *Technical Report ASC-2014-30, Institute for Analysis and Scientific Computing* (2014). URL: `https://www.asc.tuwien.ac.at/~schoeberl/wiki/publications/ngs-cpp11.pdf`.

Results: curved cells



(a) Velocity magnitude (b) Pressure

Figure: Computed solution

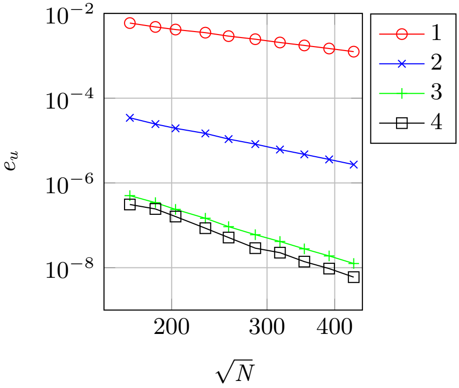


Figure: e_u against $\sqrt{N}$ for $k = 3$ with piecewise polynomial geometric mappings of degrees 1, 2, 3, and 4.

	N	e_u	$\boldsymbol{e_{\nabla\cdot u}}$	$e_{[\![u]\!]}$
Present method	3870	6.17×10^{-4}	$\mathbf{5.45 \times 10^{-15}}$	4.68×10^{-14}
Original method	3870	6.71×10^{-4}	$\mathbf{3.02 \times 10^{-2}}$	8.51×10^{-13}

Extension to the Navier-Stokes equations

- ✓ Straightforward extension to the Navier-Stokes equations
- ✓ Divergence free velocity field on affine and non-affine simplex and non-simplex cells
- ✓ Local momentum conservation
- ✓ Arbitrarily high order

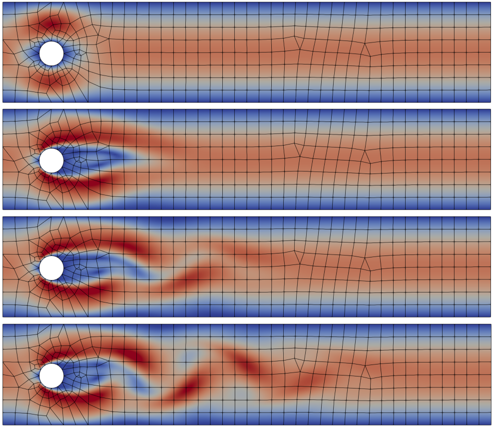


Figure: Velocity magnitude

Open questions

We are currently working on:

- Implementing a FEniCSx version of the method for meshes with quadrilateral and hexahedral cells.
- Rigorous proofs of the discrete inf-sup condition and error estimates on non-affine meshes.
- Optimal preconditioners and investigating the performance of the method at large scale.

Any suggestions/advice about these topics would be very much appreciated!

FEniCS PROJECT

Thank you. Any questions?