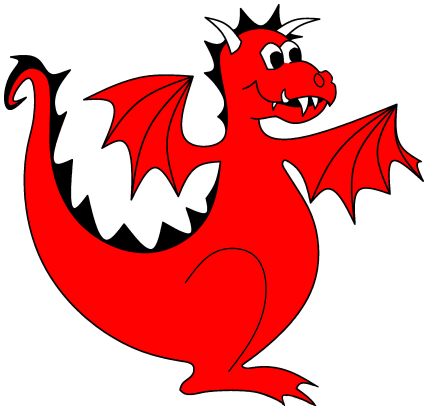
Imperial College London

Firedrake: towards generating an extruded dynamical core

David A. Ham, Andrew T. T. McRae, Gheorghe-Teodor Bercea, Lawrence Mitchell, Florian Rathgeber, Colin J. Cotter, Fabio Luporini, Nicolas Loriant, Michael Lange and Paul H. J. Kelly

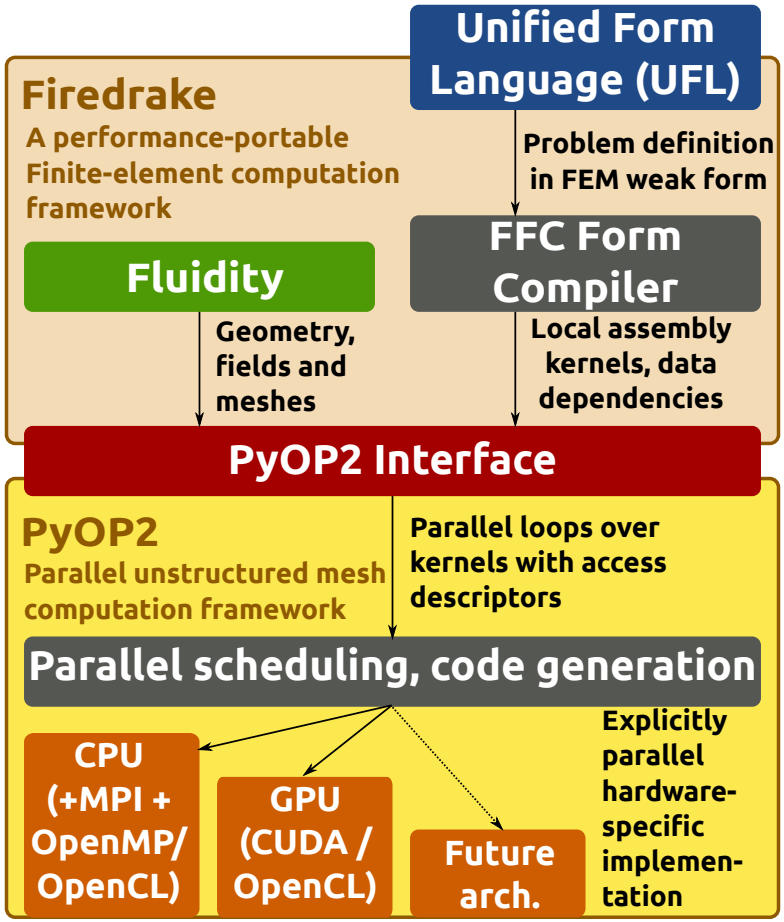
Departments of Mathematics, Computing, and Earth Science and Engineering
Imperial College London



Firedrake vs GungHo/LFRic

- Same data model.
- Essentially the same execution model.
- Firedrake generates code at runtime, GungHo is compile time.
- Firedrake generates code for discretisations, GungHo currently does not plan to.
- Firedrake currently only supports triangles in the horizontal. Primal quads on roadmap, primal-dual is harder.

The Firedrake system



The model problem

$$-\nabla^2 p = f \tag{1}$$

on $\Omega = [0,1] \times [0,1] \times [0,1]$ with:

$$p = 0 \quad \text{on } \Gamma_D = \{x = 0, x = 1, y = 0, y = 1\} \tag{2}$$

$$\nabla p \cdot n = g \quad \text{on } \Gamma_N = \{z = 0, z = 1\} \tag{3}$$

The model problem

$$-\nabla^2 p = f \tag{1}$$

on $\Omega = [0,1] \times [0,1] \times [0,1]$ with:

$$p = 0 \quad \text{on } \Gamma_D = \{x = 0, x = 1, y = 0, y = 1\} \tag{2}$$

$$\nabla p \cdot n = g \quad \text{on } \Gamma_N = \{z = 0, z = 1\} \tag{3}$$

In dual form:

$$u - \nabla p = 0 \tag{4}$$

$$-\nabla \cdot u = f \tag{5}$$

Model problem in weak form

$$u - \nabla p = 0 \tag{4}$$
$$-\nabla \cdot u = f \tag{5}$$

In weak form, find $(u, p) \in W_2' \times W_3$ such that:

$$\int_\Omega u \cdot v + p \nabla \cdot v \mathrm{d}x + \cancel{\int_{\Gamma_D} p \mathbf{n} \cdot v \mathrm{d}s} = 0 \quad \forall \mathbf{v} \in W_2' \tag{6}$$
$$\int_\Omega -(\nabla \cdot u) q \mathrm{d}x = \int_\Omega f q \mathrm{d}x \quad \forall q \in W_3 \tag{7}$$

where $W_2' = \{u \in W_2, u \cdot n = g \text{ on } \Gamma_N\}$.

Building a mesh

```
from firedrake import *

m = UnitSquareMesh(4, 4)
mesh = ExtrudedMesh(m, 11, layer_height=0.1)
```

It's also possible to specify a more complex expression for layer heights.

Building the function space

$$W_2 = H_{\mathrm{Div}}\left(\mathrm{BDM}_1 \times \mathrm{DG}_0 + \mathrm{DG}_1 \times \mathrm{CG}_1\right) \tag{8}$$

```
W2_a_horiz = FiniteElement("BDM", "triangle", 1)
W2_a_vert = FiniteElement("DG", "interval", 0)
W2_a = HDiv(OuterProductElement(W2_a_horiz,
                                W2_a_vert))

W2_b_horiz = FiniteElement("DG", "triangle", 1)
W2_b_vert = FiniteElement("CG", "interval", 1)
W2_b = HDiv(OuterProductElement(W2_b_horiz,
                                W2_b_vert))

W2_elt = W2_a + W2_b
W2 = FunctionSpace(mesh, W2_elt)
```

Building the function space

$$W_3 = DG_0 \times DG_0 W = W_2 \times W_3 \tag{9}$$

```
W3 = FunctionSpace(mesh, "DG", 0, vfamily="DG",
                                  vdegree=0)

return W2 * W3
```


Alternative function space

```
# Note the Norwegian counting.
def RT2():
    W2_a_horiz = FiniteElement("RT", "triangle", 2)
    W2_a_vert = FiniteElement("DG", "interval", 1)
    W2_a = HDiv(OuterProductElement(W2_a_horiz,
                                    W2_a_vert))

    W2_b_horiz = FiniteElement("DG", "triangle", 1)
    W2_b_vert = FiniteElement("CG", "interval", 2)
    W2_b = HDiv(OuterProductElement(W2_b_horiz,
                                    W2_b_vert))

    W2_elt = W2_a + W2_b
    W2 = FunctionSpace(mesh, W2_elt)

    W3 = FunctionSpace(mesh, "DG", 1, vfamily="DG",
                                      vdegree=1)

    return W2 * W3
```

Choose a right hand side

$$f = 3\pi^2 \sin(\pi x)\sin(\pi y)\sin(\pi z) \tag{10}$$

yielding solution

$$p = \sin(\pi x)\sin(\pi y)\sin(\pi z) \tag{11}$$

```
f = Function(W3)
exact = Function(W3)
f.interpolate(Expression("(3*pi*pi)*sin(x[0]*pi)*sin(x
                          [1]*pi)*sin(x[2]*pi)"))
exact.interpolate(Expression("sin(x[0]*pi)*sin(x[1]*pi
                          )*sin(x[2]*pi)"))
```

Set matching boundary conditions

$$u \cdot z = \pi \sin(\pi x) \sin(\pi y) \text{ on } z = 0 \tag{12}$$

$$u \cdot z = -\pi \sin(\pi x) \sin(\pi y) \text{ on } z = 1 \tag{13}$$

```
bcs = [DirichletBC(W[0], Expression(("0.0", "0.0", "pi
                                     *sin(x[0]*pi)*sin(x[1]*pi)"
                                     )), "bottom"),
       DirichletBC(W[0], Expression(("0.0", "0.0", "-
                                           pi*sin(x[0]*pi)*sin(
                                           x[1]*pi)")), "top")]
```

Now actually specify the variational problem

$$\int_\Omega u \cdot v + p\nabla \cdot v \mathrm{d}x = 0 \quad \forall \mathbf{v} \in W_2' \tag{14}$$

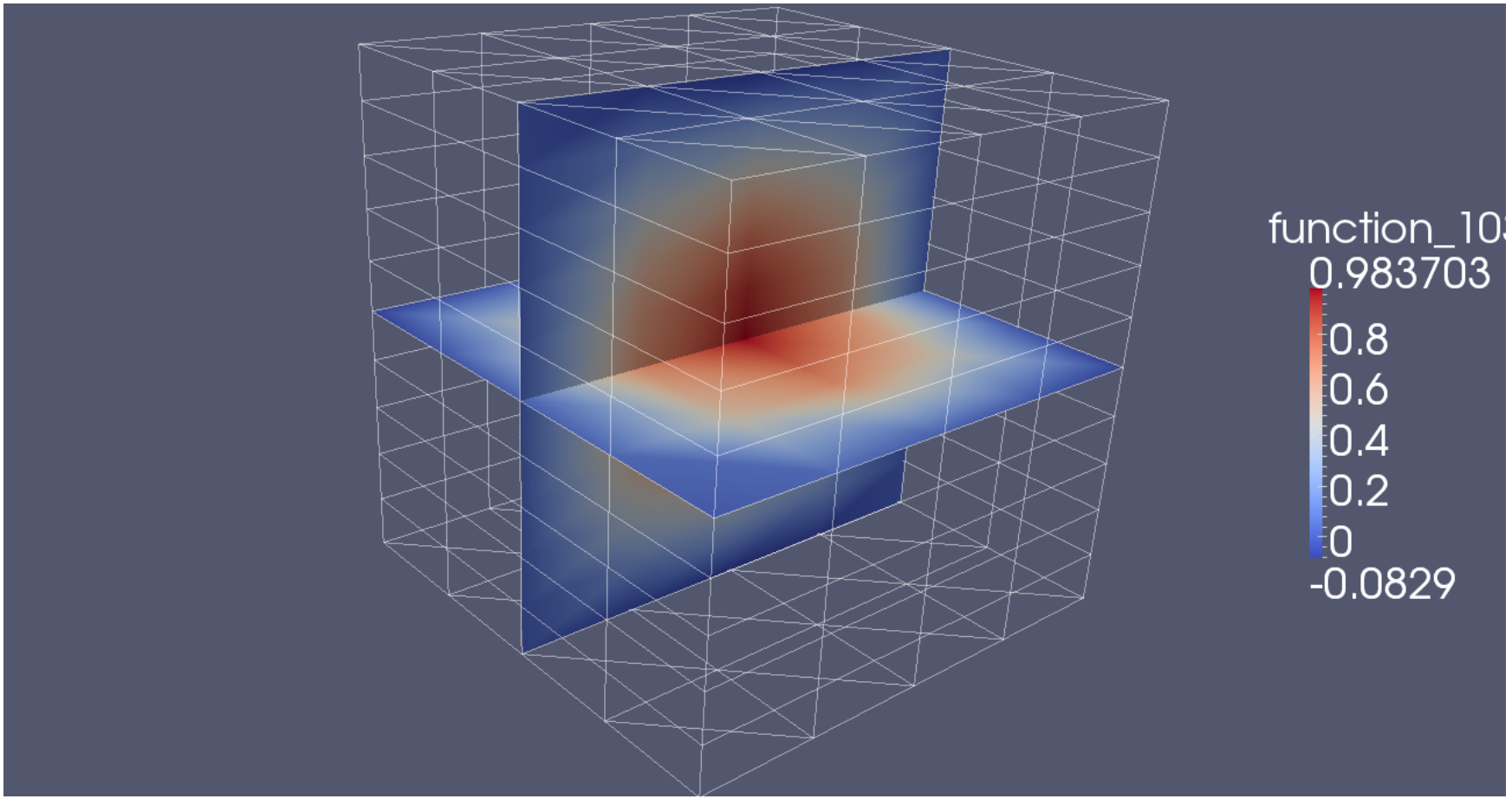
$$\int_\Omega -(\nabla \cdot u) q \mathrm{d}x = \int_\Omega f q \mathrm{d}x \quad \forall q \in W_3 \tag{15}$$

```
u, p = TrialFunctions(W)
v, q = TestFunctions(W)
a = (-q*div(u) + dot(v, u) + div(v)*p)*dx
L = f*q*dx
```


Define a solution function, and solve

```
out = Function(W)
solve(a == L, out, bcs=bcs, solver_parameters={
      'pc_type': 'fieldsplit',
      'pc_fieldsplit_type': 'schur',
      'ksp_type': 'cg',
      'pc_fieldsplit_schur_fact_type': 'FULL',
      'fieldsplit_0_ksp_type': 'cg',
      'fieldsplit_1_ksp_type': 'cg'})
```


The final result



Development roadmap

- Vertical slice (just bugs away)
- Lateral BCs (including periodic)
- Facet integrals (interior and exterior)
- Hybridisation
- Replace the mesh data structures with DMPLEX from PETSc
- Quads
- Somewhat less stupid tensor product evaluation
- Multigrid
- General performance engineering

Finally

http://firedrakeproject.org

http://wp.doc.ic.ac.uk/spo/finite-element/