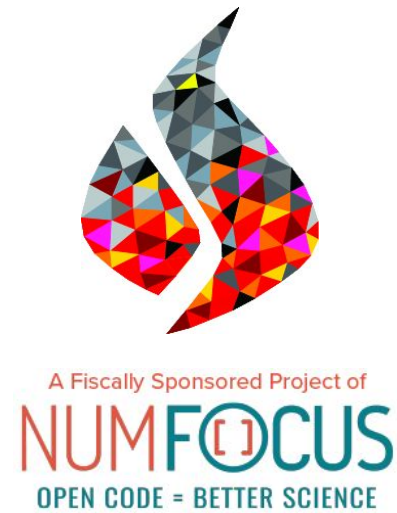


The FEniCS Project



Michal Habera, **Jack S. Hale**, Chris Richardson, Johannes Ring, Marie E. Rognes, Nathan Sime, Garth N. Wells and the FEniCS Project contributors.

<https://fenicsproject.org>

3M Presentation, Minnesota, 23rd February 2021.

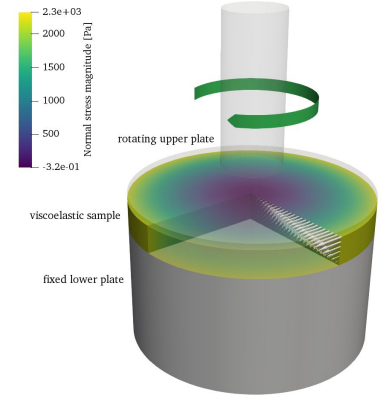
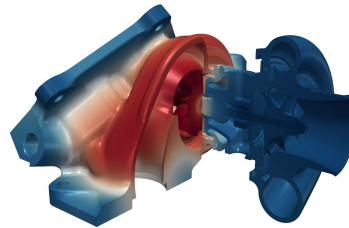
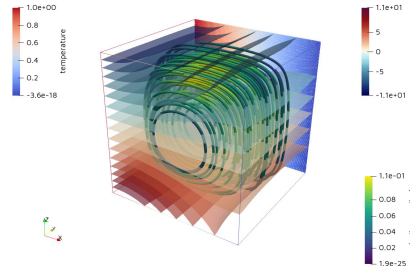
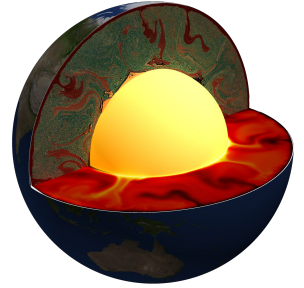
Outline

1. What is the FEniCS Project?
2. Key features
3. Live demo
4. Tour of applications
5. Summary

What is the FEniCS Project?

What is the FEniCS Project?

- A computing platform for translating *scientific models* into *finite element simulations*.
- 15 year history.
- Open Source (LGPLv3 and MIT).



What is the FEniCS Project for?

- For rapidly prototyping and testing novel finite element methods.
- For rapidly prototyping numerical solvers to test scientific hypotheses.
- For writing robust, scalable and maintainable finite element solvers in the context of larger engineering/scientific software projects.
- For teaching finite element methods to students.

Who is aimed at?

- Using the FEniCS Project requires domain specific knowledge of:
 - finite element methods,
 - partial differential equations,
 - the particulars of the problem you are trying to solve!
- Students, academics, engineers and scientists have all used the FEniCS Project successfully to solve problems of real-world interest.
- It is *not* an end-to-end finite element analysis package in the style of products from e.g. Ansys, Dassault or Comsol.
- The FEniCS Project has been used to write finite element solvers within end-to-end analysis packages aimed at users *without* domain specific knowledge.

Who is currently involved?

Legal

A Fiscally Sponsored Project of
NUMFOCUS
OPEN CODE = BETTER SCIENCE

Institutions

simula



UNIVERSITY OF
CAMBRIDGE



UNIVERSITÉ DU
LUXEMBOURG



CARNEGIE
SCIENCE

Community



How can I try it out?

- Docker containers

```
docker run -ti dolfinx/dolfinx
```

- Spack package manager (HPC)

```
spack add py-fenics-dolfinx ^petsc+mumps+hypre cflags="-O3"  
fflags="-O3"
```

```
spack install
```


Key features

Write mathematics as code

Find $u \in V$ such that
 $a(u, v) = L(v) \quad \forall v \in V,$

with

$$a(u, v) = \int_{\Omega} \nabla u \nabla v \, dx,$$

$$L(v) = \int_{\Omega} f v \, dx.$$

`u = TrialFunction(V)`

`v = TestFunction(V)`

`f = Coefficient(V)`

`a = inner(grad(u), grad(v)) * dx`

`L = inner(f, v) * dx`

Automatic differentiation

- Automatic differentiation capabilities remove tedious and error-prone manual differentiation steps.

$$\varphi(\underline{u}) = \frac{\mu}{2} (\mathcal{I}_c - 3) - \mu \ln(\mathcal{J}) + \frac{\lambda}{2} \ln(\mathcal{J})$$

$$\min_{u \in V} \int_{\Omega} \varphi(\underline{u}) \, dx - \int f \cdot \underline{u} \, dx := \pi(\underline{u})$$

$$D_{\underline{u}}[\pi(\underline{u})][\underline{v}] = 0 \quad \forall \underline{v} \in V$$

`u = Function(V)`

`psi = ...`

`Pi = psi*dx - f*u*dx`

`v = TestFunction(V)`

`F = derivative(Pi, u, v)`

`u_bar = TrialFunction(V)`

`J = derivative(F, u, u_bar)`

Automatic code generation

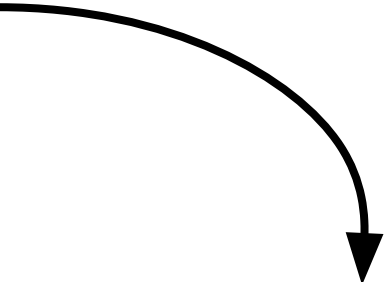
```
u = TrialFunction(V)
```

```
v = TestFunction(V)
```

```
f = Coefficient(V)
```

```
a = inner(grad(u), grad(v)) * dx
```

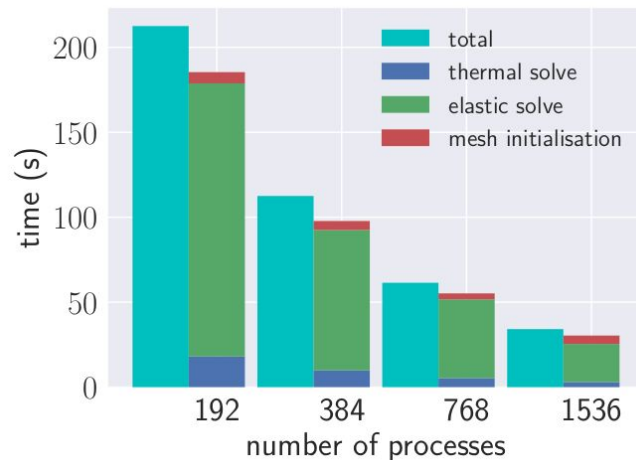
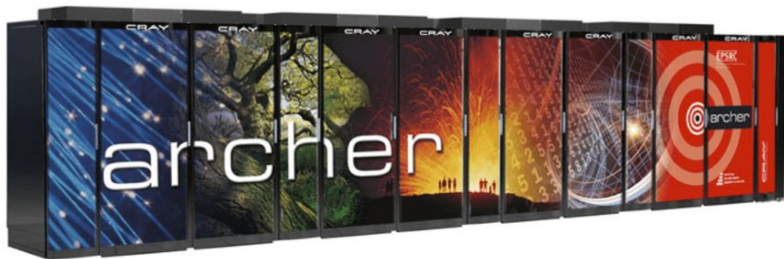
```
L = inner(f, v) * dx
```



```
void tabulate_tensor_poisson_cell_integral_43165b9e21c850b7e46d14f843t
... const ufc_scalar_t* c,
... const double* restrict coordinate,
... const int* unused_local_index,
... const int* cell_orientation)
{
    ...// Precomputed values of basis functions and precomputations
    ...// FE* dimensions: [entities][points][dofs]
    ...// PI* dimensions: [entities][dofs][dofs] or [entities][dofs]
    ...// PM* dimensions: [entities][dofs][dofs]
    ...alignas(32) static const ufc_scalar_t FE3_C0_D01_Q1[1][1][2] = {
    ...// Unstructured piecewise computations
    ...const double J_c0 = coordinate_dofs[0] * FE3_C0_D01_Q1[0][0][0] +
    ...const double J_c3 = coordinate_dofs[1] * FE3_C0_D01_Q1[0][0][0] +
    ...const double J_c1 = coordinate_dofs[0] * FE3_C0_D01_Q1[0][0][0] +
    ...const double J_c2 = coordinate_dofs[1] * FE3_C0_D01_Q1[0][0][0] +
```

High-performance computer ready

- FEniCS Project solvers run on high-performance computers using MPI.

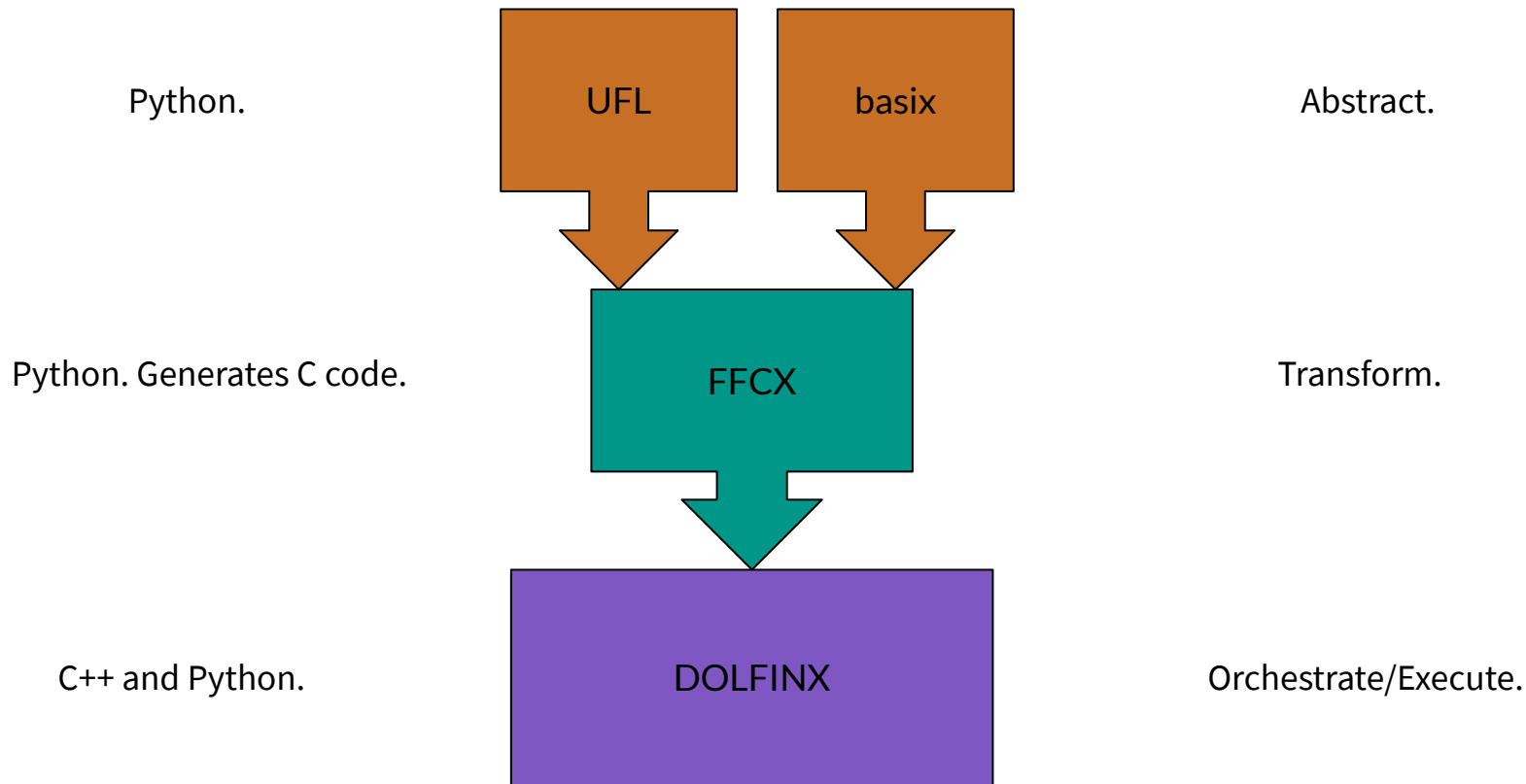


Boost productivity

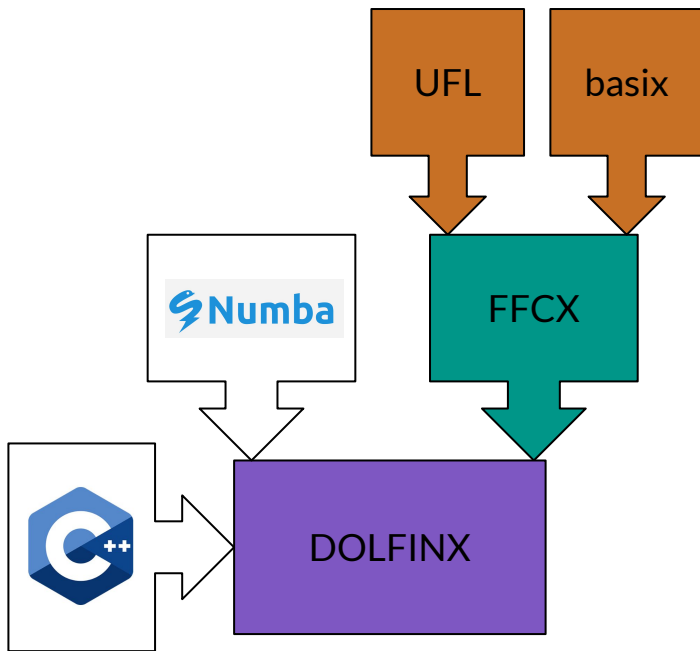
- For performance reasons, the FEniCS Project is written mostly in C++ and C.
- All main functionality is wrapped into Python.
 - All of the performance of C++ with the productivity of Python.
 - Interoperable with your favorite Scientific Python packages (numpy, scipy, matplotlib etc.).



Modular



Extendable



- The FEniCS Project has been designed to be extendable.
- Use automated tools 90% of the time.
- 10% of the time automated tools may not meet needs. Allows fully custom implementations of:
 - Finite element kernels.
 - Assemblers.
 - Input/Output.
 - Linear algebra backends.

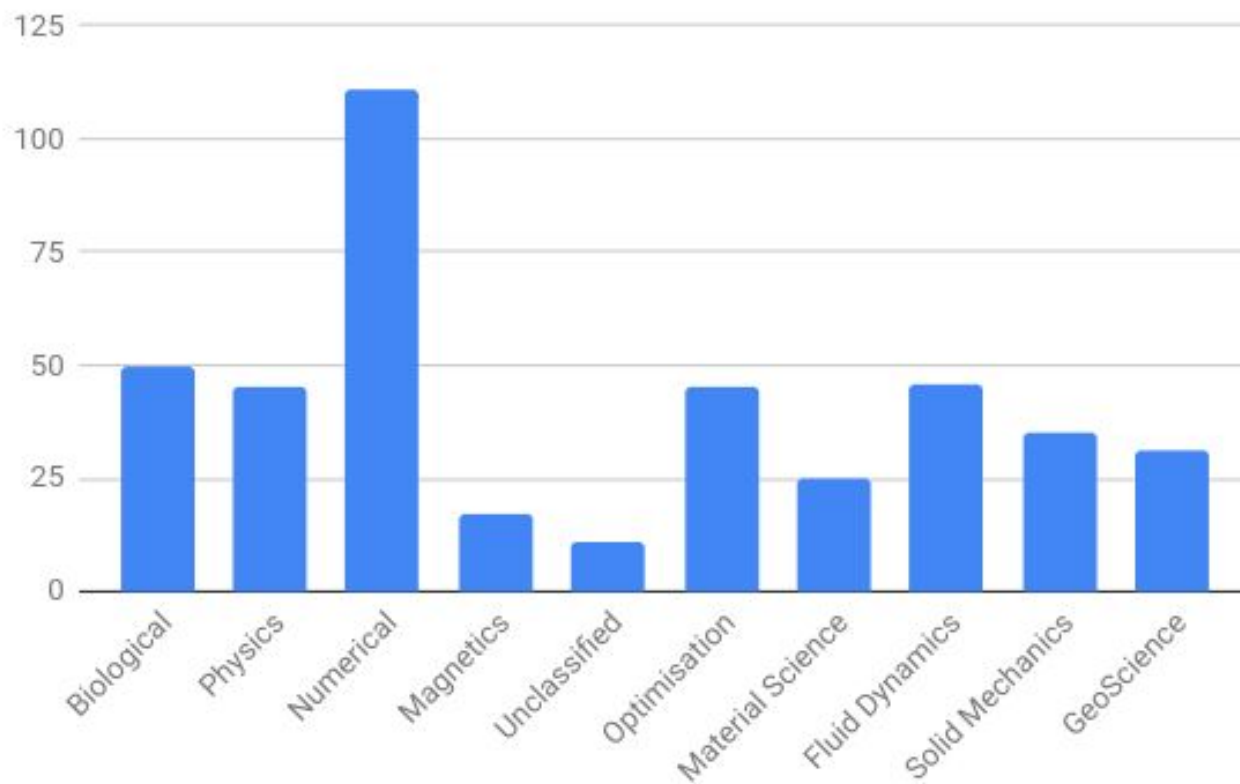
Live demo

Tour of applications

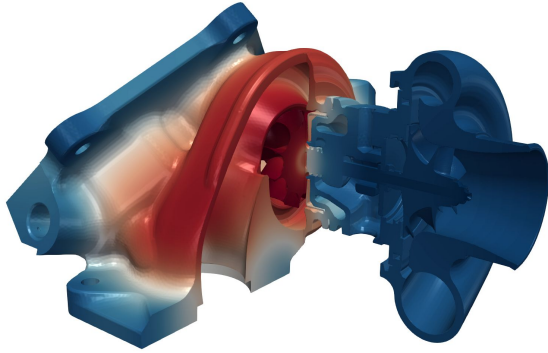
Popular and widely used

Automated solution of differential equations by the finite element method: The FEniCS book A Logg, KA Mardal, GN Wells Lecture Notes in Computational Science and Engineering 84	1893	2012
The FEniCS project version 1.5 M Alnæs, J Blechta, J Hake, A Johansson, B Kehlet, A Logg, ... Archive of Numerical Software 3 (100)	732	2015
DOLFIN: Automated finite element computing A Logg, GN Wells ACM Transactions on Mathematical Software (TOMS) 37 (2), 1-28	536	2010
Unified form language: A domain-specific language for weak formulations of partial differential equations MS Alnæs, A Logg, KB Ølgaard, ME Rognes, GN Wells ACM Transactions on Mathematical Software (TOMS) 40 (2), 1-37	231	2014

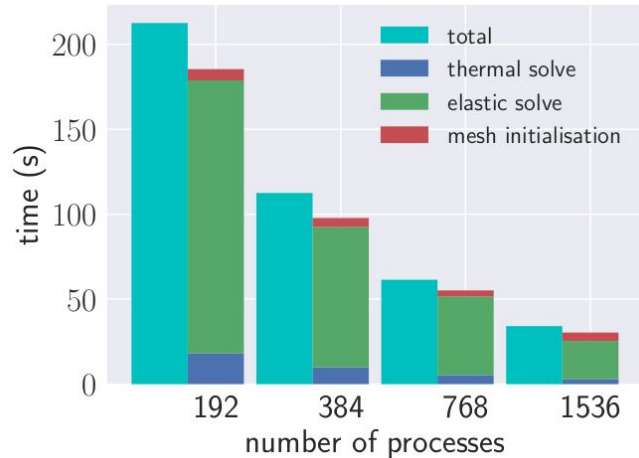
Publications citing FEniCS Book by Subject Area



Turbomachinery



- Scalable nonlinear thermomechanical analysis of turbomachinery components.
- Demonstrated computation of problems with more than 3×10^9 unknowns.

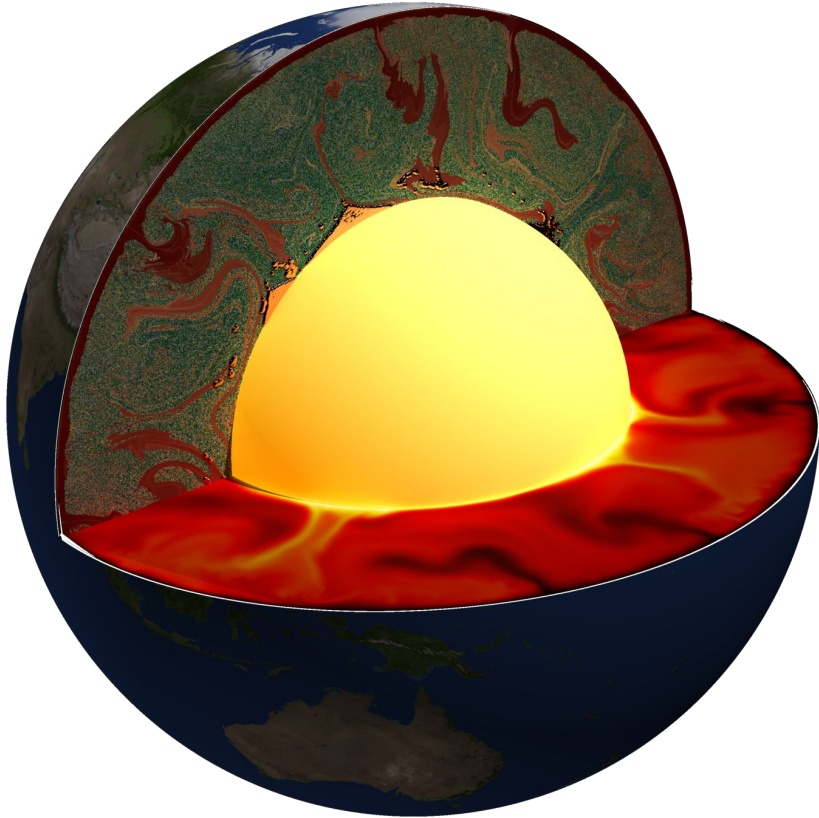


C. N. Richardson, N. Sime and G. N. Wells,

Scalable computation of thermomechanical turbomachinery problems,

Finite Elements in Analysis and Design 155, 32-42, (2019).

Geodynamics



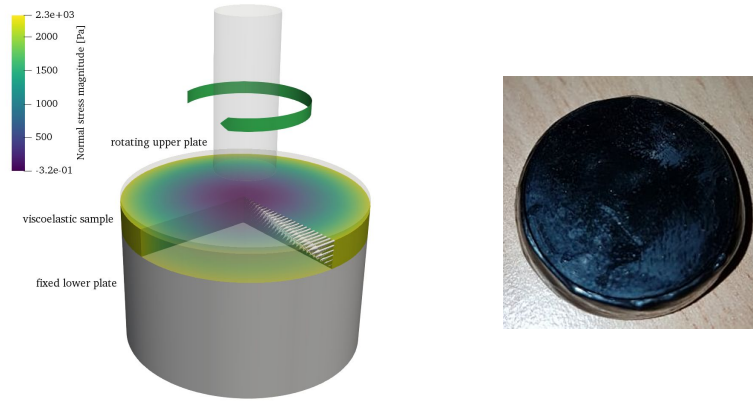
- Transient Boussinesq approximation of the Earth's compositional evolution.
- Tracer advection provided by LEoPart add-on to FEniCS.
- Age of compositional material (top), mantle temperature (bottom).
- Primordial material collects into persistent “piles” at the core-mantle boundary (specular highlight).

T. D. Jones, N. Sime and P. E. van Keken,

Burying Earth's primitive mantle in the slab graveyard

Geochemistry, Geophysics, Geosystems (accepted).

Viscoelastic flow characterisation

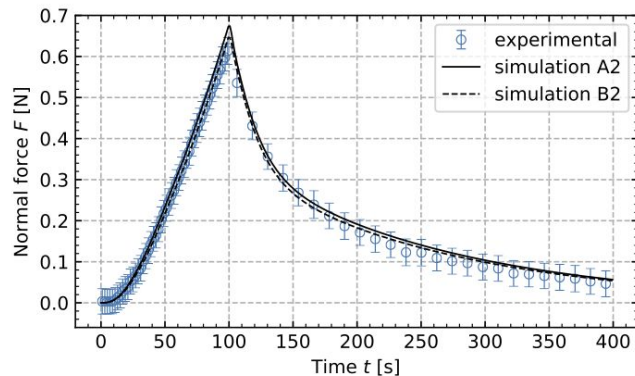


- Model of heated styrene butadiene rubber flowing in a rotational rheometer.
- Proposed three two time-scale viscoelastic constitutive models.
- Used for parameter estimation and model selection.

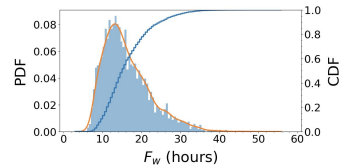
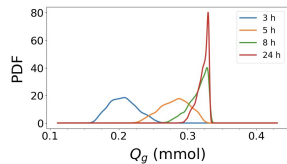
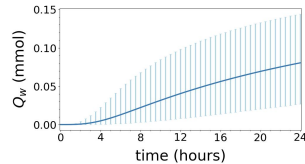
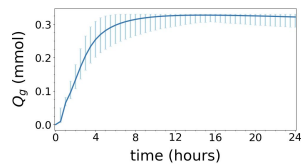
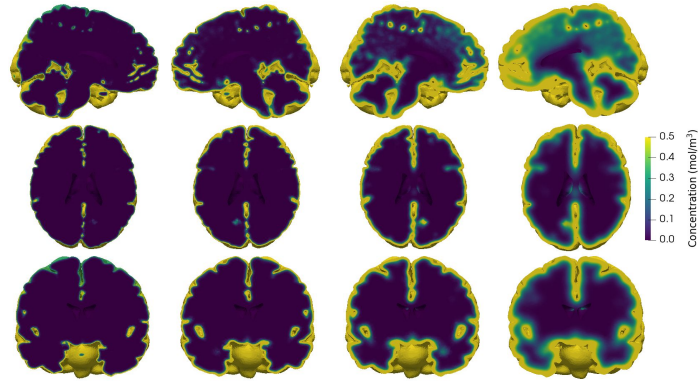
Řehoř, Gansen, Sill, Polińska, Westermann, Dheur, Baller, Hale,

A comparison of constitutive models for describing the flow of styrene-butadiene rubber,

Journal of Non-Newtonian Fluid Mechanics, Volume 286, 104398, (2020).



Biomechanics of the Human Brain



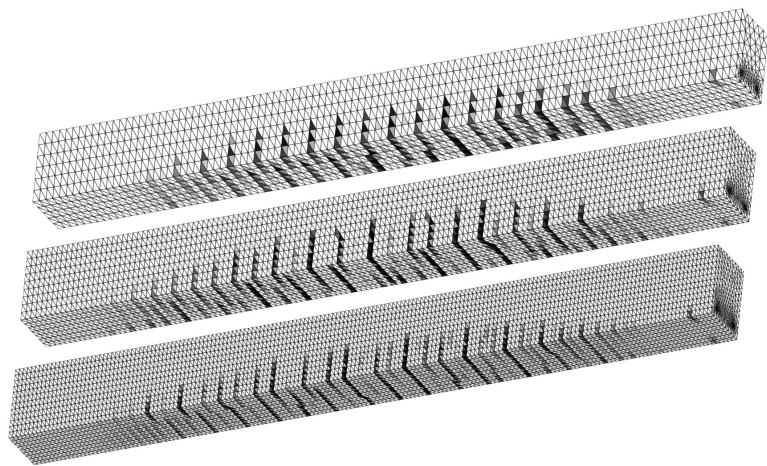
- The human brain contains a complex lymphatic system for draining unwanted substances.
- The precise mechanisms are poorly understood.
- Simulation of injected tracers in three-dimensional geometry.
- Quantified competing effects of advection and diffusion.

Croci, Vinje, Rognes,

Uncertainty quantification of parenchymal tracer distribution using random diffusion and convective velocity fields

Fluids and Barriers of the CNS, Volume 16, 32 (2019).

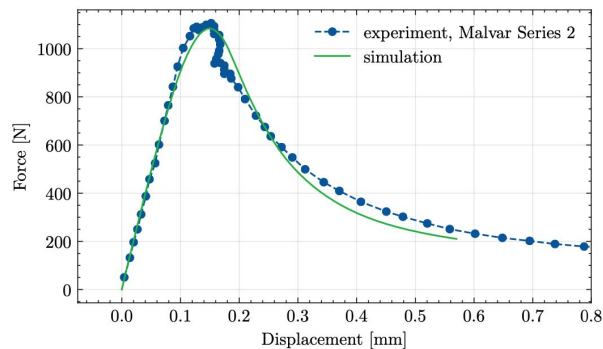
Damage of concrete structures



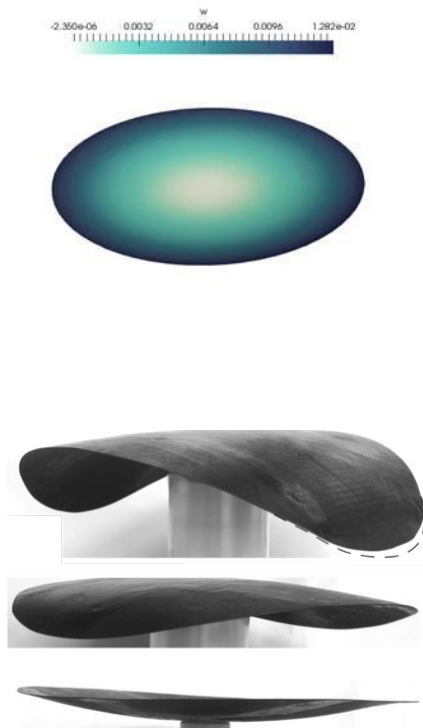
- Concrete is a typical material experiencing low tensile strength.
- Exact definition of failure mechanism involves complicated failure surfaces formulated in principal stress space.
- Used FEniCS AD to provide consistent tangent operator in damage mechanics.

Habera, Zilian

<https://github.com/michalhabera/fecoda>



Thin Structures



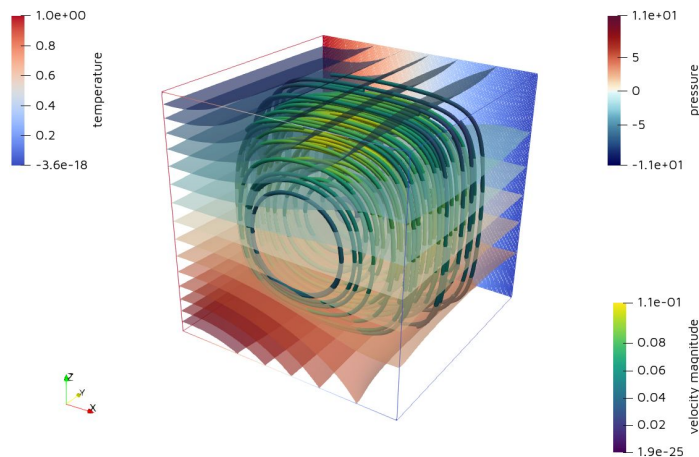
- FEniCS-Shells is a FEniCS based library for simulating thin structures.
- Bi-stable shell structure.
- Simulation.
 - Lenticular plate is heated.
 - Deforms with initially spherical curvature.
 - Snaps through to developable surface.

Hale, Brunetti, Bordas, Maurini,

Simple and extensible plate and shell finite element models through automatic code generation capabilities,

Computers and Structures, Volume 209, October 2018, 163-181.

Thermally-driven Navier-Stokes

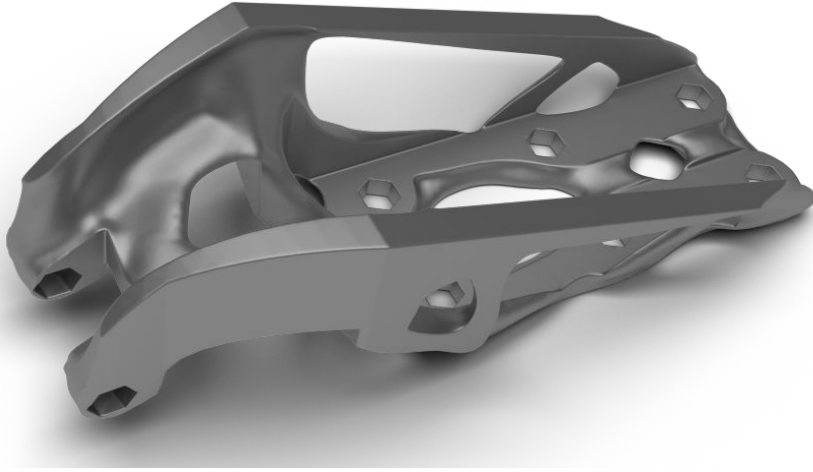


- Stationary Navier-Stokes/Temperature with Boussinesq approximation.
- Demonstrated computation up to 30 million degrees of freedom and ~800 MPI processes.
- Highly flexible robust block preconditioning.

Řehoř and Hale (unpublished).

DOF ($\times 10^6$)	MPI processes	Nonlinear iterations	Linear iterations	Navier-Stokes iterations	Temperature iterations	Time to solution (s)
0.7405	24	2	8	120 (15)	50 (6.2)	8.62
1.488	48	2	8	123 (15.4)	49 (6.1)	9.05
2.793	96	2	8	121 (15.1)	50 (6.2)	8.88
5.769	192	2	9	134 (14.9)	56 (6.2)	10.3
11.66	384	2	9	139 (15.4)	56 (6.2)	12.6
23.39	768	2	9	141 (15.7)	56 (6.2)	12.4

Stochastic Topology Optimisation



- Rafinex's *Stochastic Topology Optimisation Tool* can automatically produce robust designs that behave safely even when subjected to rare loading conditions.
- Use FEniCS Project as scalable and efficient solid mechanics solver at core of engineering-ready product.



Rafinex Sarl

<https://www.rafinex.com/>

Summary

Summary

- The FEniCS Project allows the quick, efficient and flexible translation of scientific models into finite element simulators.
- Key features include:
 - Writing mathematics as code.
 - Automatic differentiation.
 - HPC capable.
 - Modular and extendable.
- It can be used to write simulators for a huge range of problems.
- It is Open Source and available for use today.

Questions?