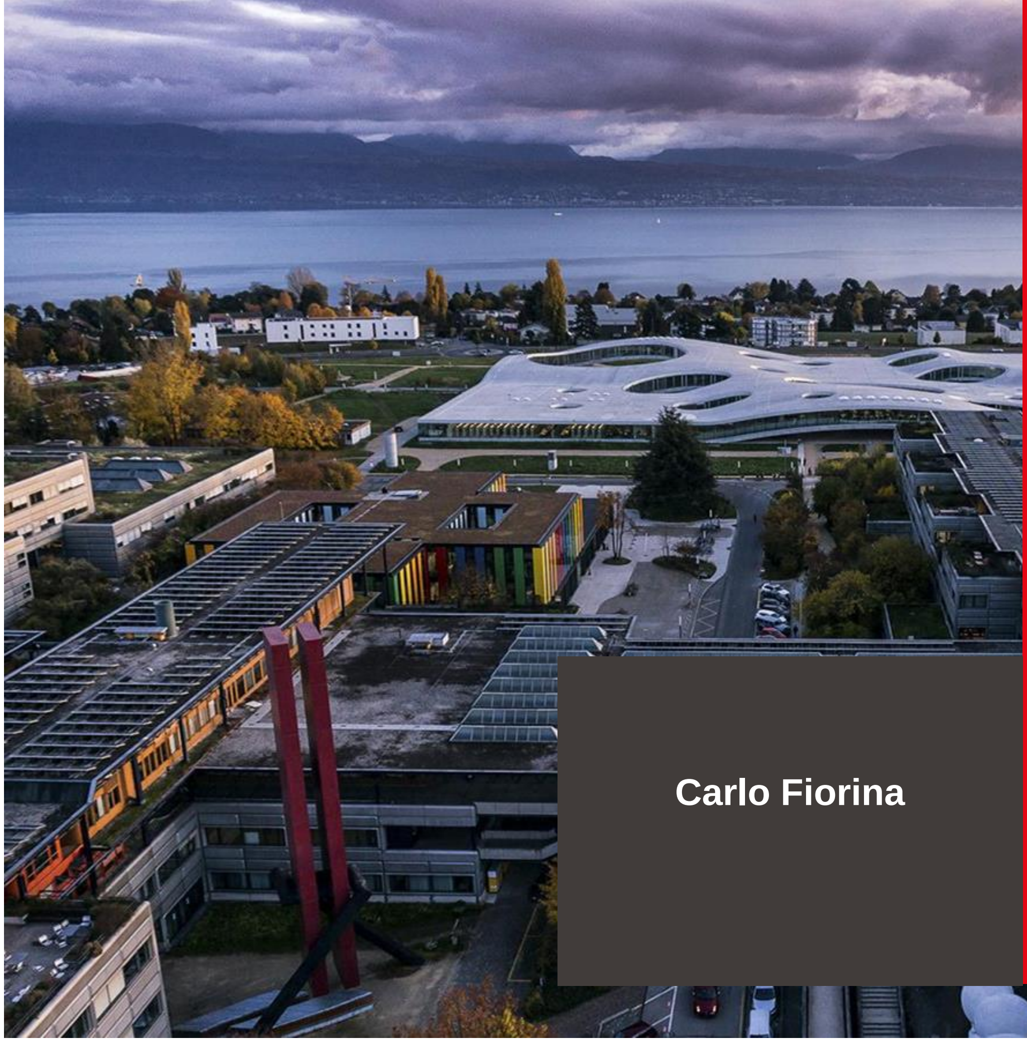
EPFL



Use of OpenFOAM for multi-physics in nuclear

Carlo Fiorina

École polytechnique fédérale de Lausanne

This workshop

What to expect

- Overview of the modelling capabilities of OpenFOAM and lessons learnt
- A crash introduction to OpenFOAM
- A crash introduction to GeN-Foam

What not to expect

- A full course on the use of OpenFOAM
- A full course on the use of OFFBEAT, GeN-Foam, or other OpenFOAM solvers

Objectives

- Understanding of modelling capabilities and pros & cons
- How to approach OpenFOAM-based tools
- References/keywords/best practices for autonomous learning of GeN-Foam (and other nuclear solvers)

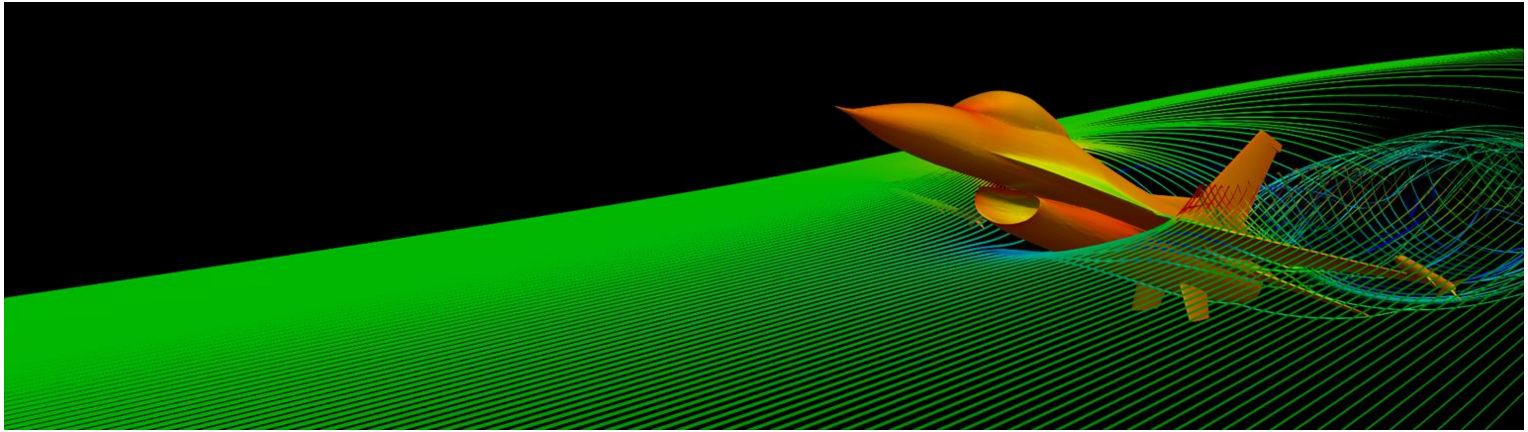
This workshop

- More slides than I can actually present!
- Objective to have consistent and readable material for you to keep

- What is OpenFOAM?
  - Distributed as CFD toolbox
  - ~10k to 20k estimated users worldwide

Open∇FOAM

The Open Source CFD Toolbox



OpenFOAM
The Open Source CFD Toolbox

- What is OpenFOAM?
  - ✓ Distributed as CFD toolbox
  - ✓ ~10k to 20k estimated users worldwide
  - ✓ OpenFOAM = Open Field Operation And Manipulation
  - ✓ Essentially a large, well organized, HPC-scalable, C++ library for the finite-volume discretization and solution of PDEs, and including several functionalities like ODE solvers, projection algorithms, and mesh search algorithms
  - ✓ Object-oriented, with a high-level “fail-safe” API

$$\frac{1}{v_i}\frac{\partial \varphi_i}{\partial t} - \Delta(D_i\varphi_i) = S$$

```
fvm::ddt(IV,flux_i])- fvm::laplacian(D,flux_i])= S
```

Open∇FOAM

The Open Source CFD Toolbox

- What is OpenFOAM?
  - ✓ Distributed as CFD toolbox
  - ✓ **~10k to 20k estimated users** worldwide
  - ✓ OpenFOAM = Open Field Operation And Manipulation
  - ✓ Essentially a large, well organized, **HPC-scalable**, C++ library for the **finite-volume** discretization and solution of PDEs, and **including several functionalities** like ODE solvers, projection algorithms, and mesh search algorithms
  - ✓ **Object-oriented**, with a **high-level "fail-safe" API**

$$\frac{1}{v_i}\frac{\partial \varphi_i}{\partial t} - \Delta(D_i \varphi_i) = S$$

```
fvm::ddt(IV,flux_i])- fvm::laplacian(D,flux_i])= S
```

Disclaimer

Most of the following is content taken from

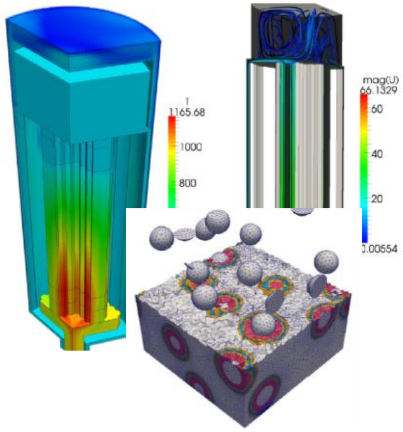
- Carlo Fiorina, Ivor Clifford, Stephan Kelm, Stefano Lorenzi, 2022. "On the development of multi-physics tools for nuclear reactor analysis based on OpenFOAM ®: state of the art, lessons learned and perspectives". Nuclear Engineering and Design 387, 111604. https://www.sciencedirect.com/science/article/pii/S0029549321005562

Use of OpenFOAM for multi-physics

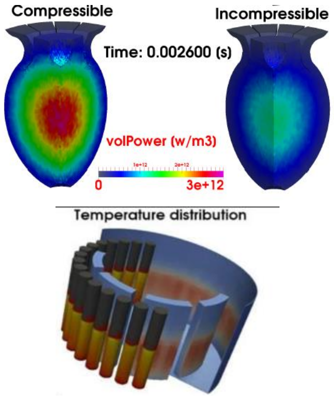
2000-2010
First activities

2010-2015
First widespread use

2015-2021
First coordinated and persistent developments

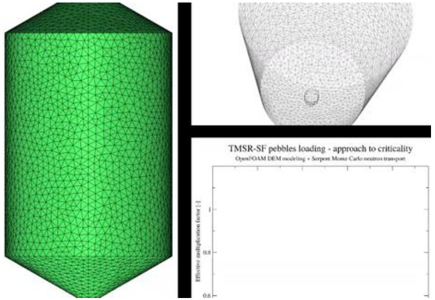


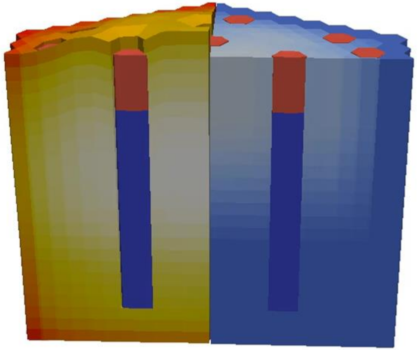
PBMRs and HTRs



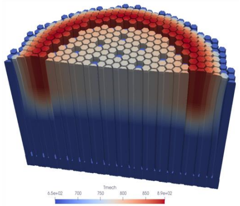
MSRs

FHRs

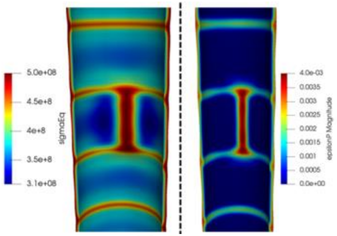




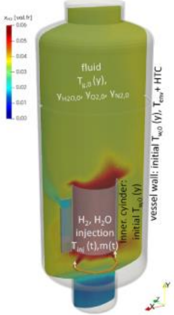
SFRs



GeN-Foam

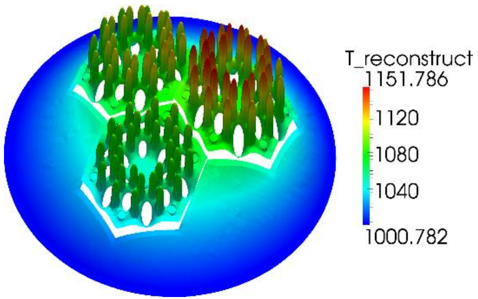


OFFBEAT



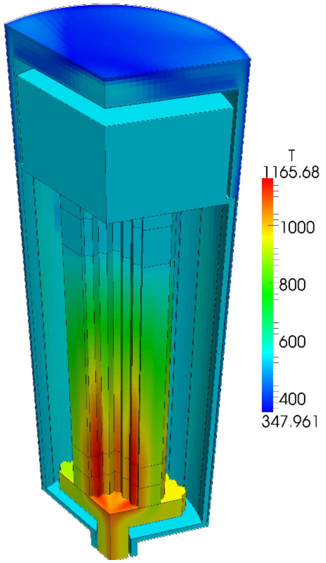
containmentFoam

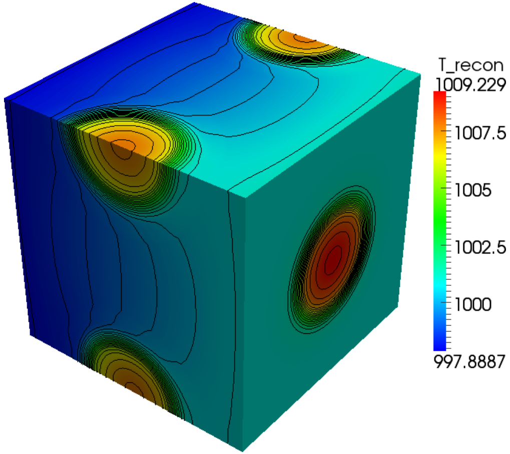
HTR modelling (PBMR -> I. Clifford)



ROM reconstructed solution for a fuel element

Full-core coarse-mesh thermal-hydraulics

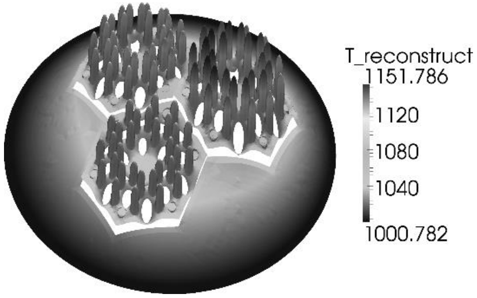




ROM reconstructed solution for TRISO coated particles

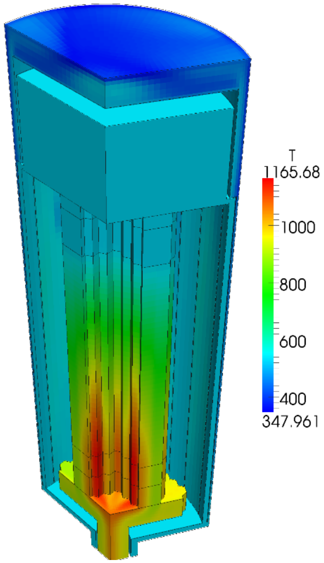
HTR modelling (PBMR -> I. Clifford)

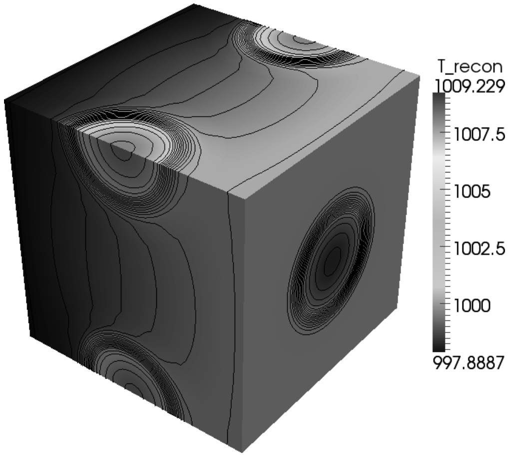
- Porous-medium thermal-hydraulics



ROM reconstructed solution for a fuel element

Full-core coarse-mesh thermal-hydraulics



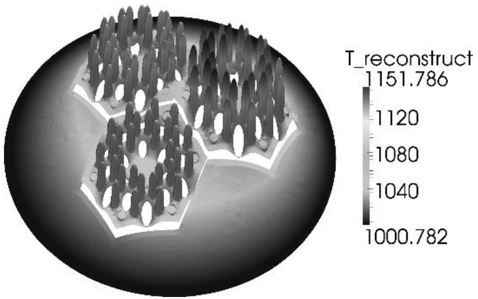


ROM reconstructed solution for TRISO coated particles

Carlo Fiorina

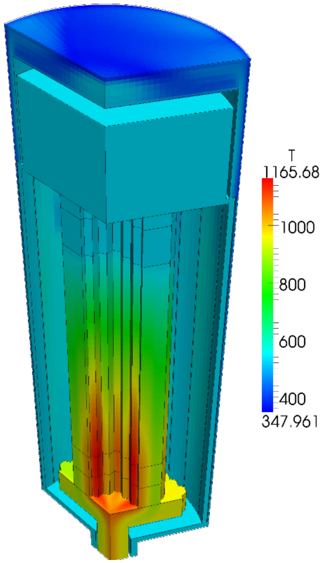
HTR modelling (PBMR -> I. Clifford)

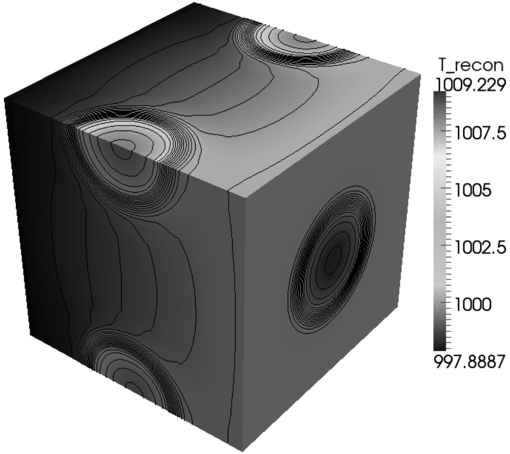
- Porous-medium thermal-hydraulics
  - ✓ Available CFD RANS



ROM reconstructed solution for a fuel element

Full-core coarse-mesh thermal-hydraulics



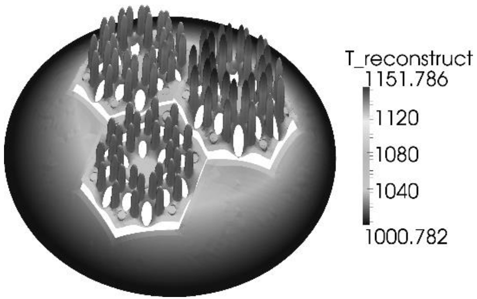


ROM reconstructed solution for TRISO coated particles

Carlo Fiorina

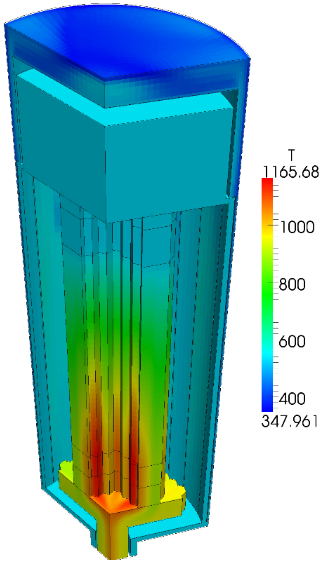
HTR modelling (PBMR -> I. Clifford)

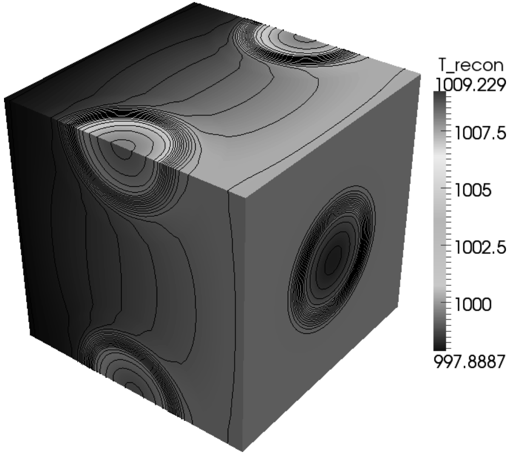
- Porous-medium thermal-hydraulics
  - ✓ Available CFD RANS plus source terms



ROM reconstructed solution for a fuel element

Full-core coarse-mesh thermal-hydraulics





ROM reconstructed solution for TRISO coated particles

Carlo Fiorina

Porous-medium thermal-hydraulics: governing equations

The coarse-mesh governing equations (Navier-Stokes and enthalpy) are:

$$\frac{\partial}{\partial t}(\alpha_i \rho_i) + \boldsymbol{\nabla} \cdot (\alpha_i \mathbf{u}_i \rho_i) = -\Gamma_{i \to j}$$

$$\frac{\partial}{\partial t}(\alpha_i \rho_i \mathbf{u}_i) + \boldsymbol{\nabla} \cdot (\alpha_i \rho_i \mathbf{u}_i \otimes \mathbf{u}_i) =$$

$$- \alpha_i \boldsymbol{\nabla} p + \boldsymbol{\nabla} \cdot (\alpha_i \boldsymbol{\sigma}_{d,i}) + \alpha_i \rho_i \mathbf{g} - \mathbf{S}_{\mathbf{u},i \to j}$$

$$\frac{\partial}{\partial t}(\alpha_i \rho_i h_i) + \boldsymbol{\nabla} \cdot (\alpha_i \mathbf{u}_i \rho_i h_i) =$$

$$\boldsymbol{\nabla} \cdot (\alpha_i \kappa_i \boldsymbol{T}_i \cdot \boldsymbol{\nabla} T_i) + \alpha_i \frac{\partial}{\partial t} p + \alpha_i \rho_i \mathbf{u}_i \cdot \mathbf{g} + \alpha_i q_{int,i} - S_{h,i \to j}$$

These reduce to traditional CFD approaches in clear fluid regions and a system-code-like approach in 1-D regions (multiple scales).

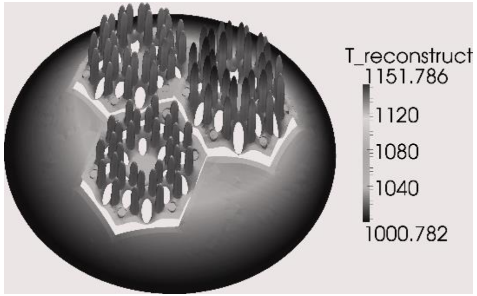
Porous-medium thermal-hydraulics: governing equations

```
    fvm::ddt(fixedRho_, UDarcy)
+   (1/alpha)*fvm::div(phiDarcy, UDarcy)
-   fvm::laplacian(fixedRho_*nuEff, UDarcy)
-   fvc::div
    (
        rho_*nuEff & dev2(T(fvc::grad(UDarcy)))
    )
+   fvm::Sp((1.0/3.0)*tr(Kds), UDarcy) + (dev(Kds) & UDarcy)
==
    alpha*fvc::reconstruct
    (
        (
          - ghf_*fvc::snGrad(fixedRho_*rhok_)
          - fvc::snGrad(p_rgh_)
        )*mesh_.magSf()
    )
```

HTR modelling (PBMR -> I. Clifford)

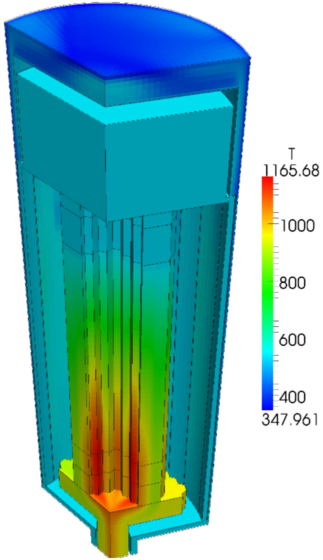
Carlo Fiorina

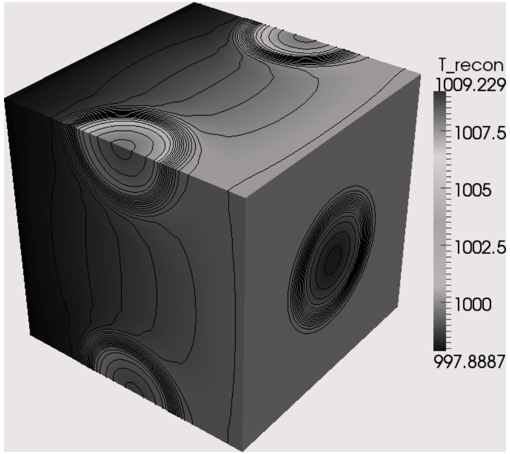
- Porous-medium thermal-hydraulics
  - ✓ Available CFD RANS plus source terms



ROM reconstructed solution for a fuel element

Full-core coarse-mesh thermal-hydraulics

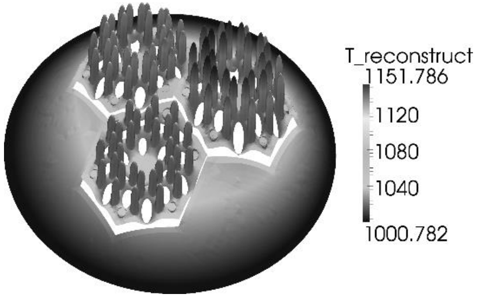




ROM reconstructed solution for TRISO coated particles

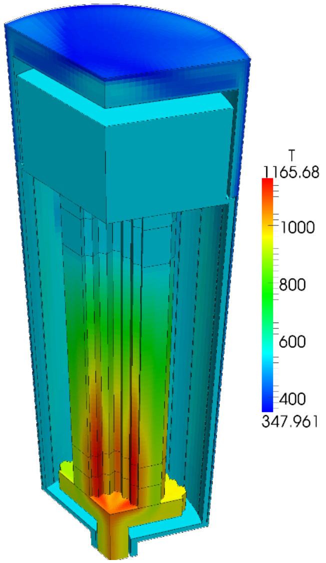
HTR modelling (PBMR -> I. Clifford)

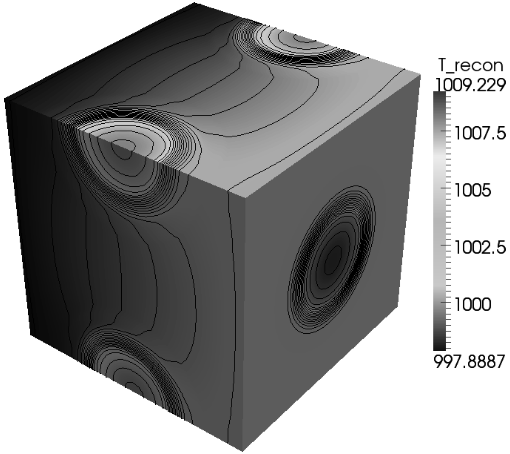
- Porous-medium thermal-hydraulics
  - ✓ Available CFD RANS plus source terms
  - ✓ Modified discretization to account for discontinuous pressure



ROM reconstructed solution for a fuel element

Full-core coarse-mesh thermal-hydraulics



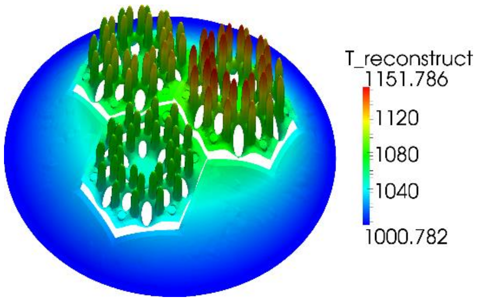


ROM reconstructed solution for TRISO coated particles

Carlo Fiorina

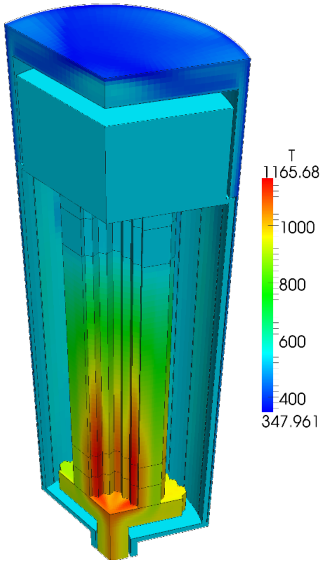
HTR modelling (PBMR -> I. Clifford)

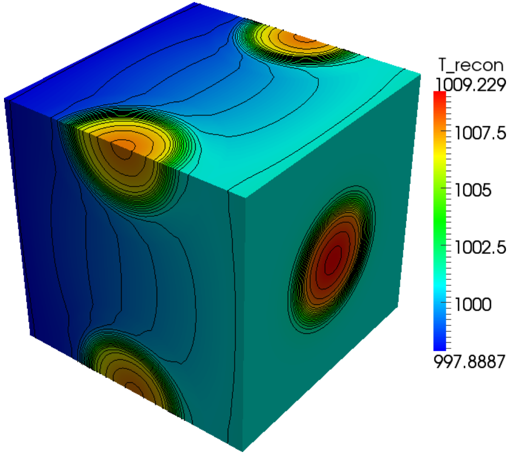
- Porous-medium thermal-hydraulics
  - ✓ Available CFD RANS plus source terms
  - ✓ Modified discretization to account for discontinuous pressure
- ROM reconstructed multi-scale temperature
  - ✓ Available ROM library
  - ✓ Multi-mesh
  - ✓ Mesh-to-mesh projections
  - ✓ Built-in ODE solvers



ROM reconstructed solution for a fuel element

Full-core coarse-mesh thermal-hydraulics





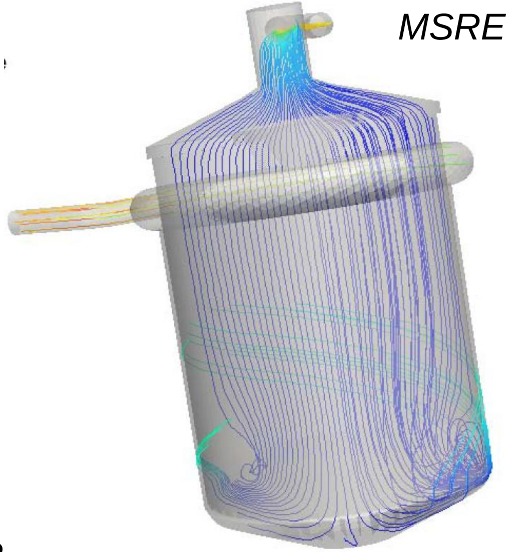
ROM reconstructed solution for TRISO coated particles

Carlo Fiorina

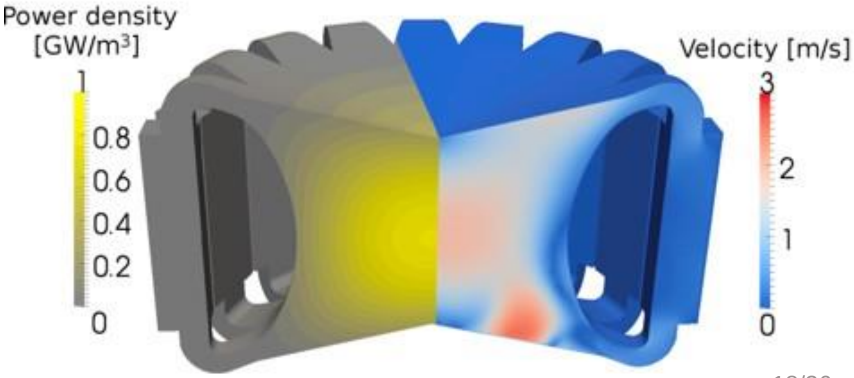
MSR modelling (M. Aufiero -> PoliMi + CNRS / GeN-Foam)

- Available CFD solvers
- Arbitrary geometries

MSRE



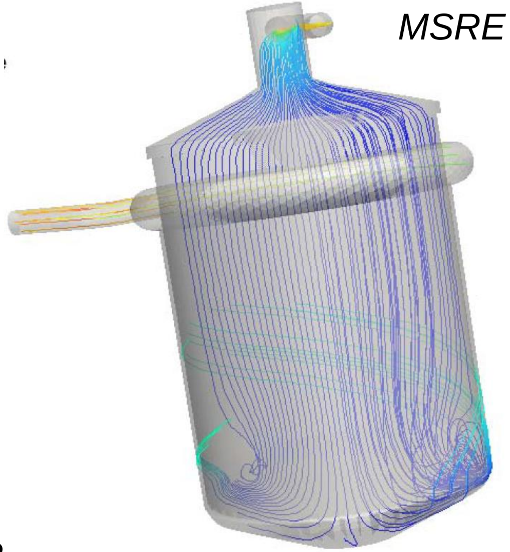
MSFR



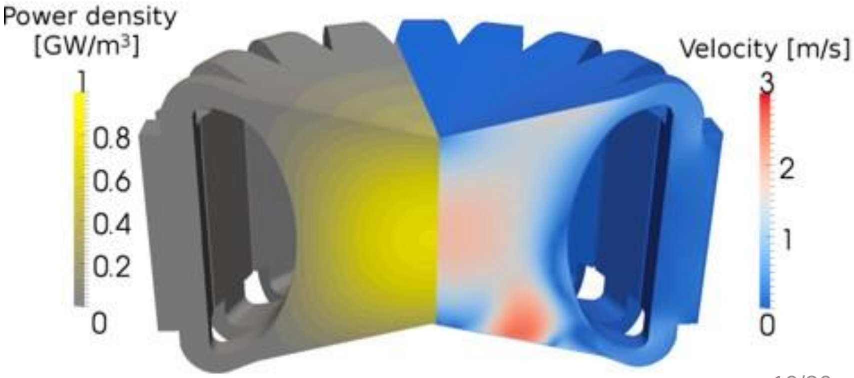
MSR modelling (M. Aufiero -> PoliMi + CNRS / GeN-Foam)

- Available CFD solvers
- Arbitrary geometries
- Streamlined implementation of diffusion and DNP equations

MSRE



MSFR

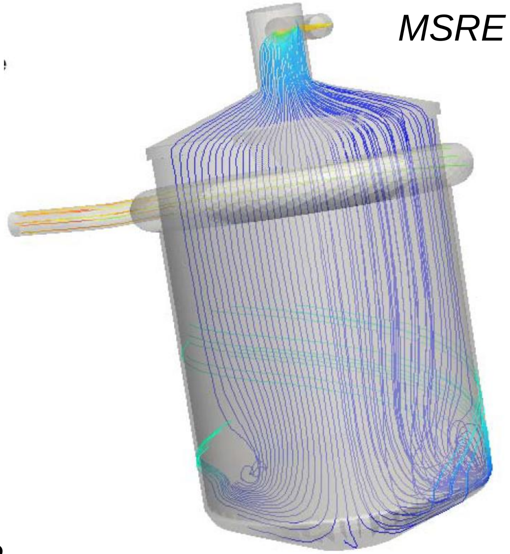


MSR modelling (M. Aufiero -> PoliMi + CNRS / GeN-Foam)

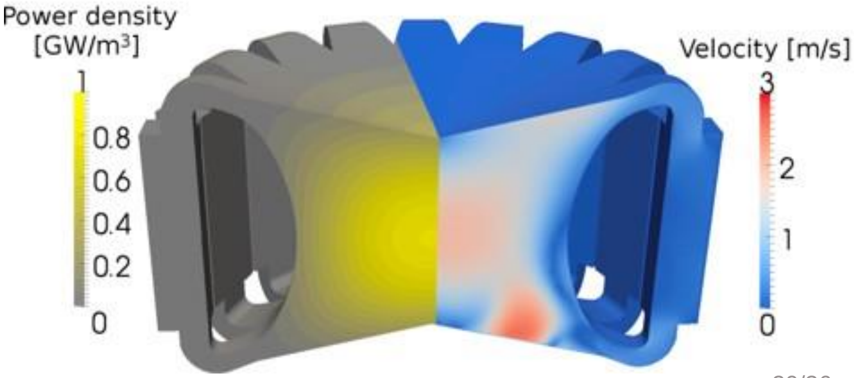
- Available CFD solvers
- Arbitrary geometries
- Streamlined implementation of diffusion and DNP equations

```
fvm::ddt(IV,flux_i])- fvm::laplacian(D,flux_i])= S
```

MSRE



MSFR



Carlo Fiorina

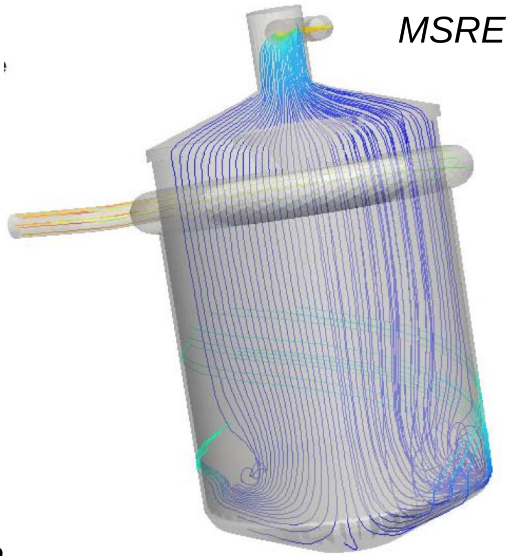
MSR modelling (M. Aufiero -> PoliMi + CNRS / GeN-Foam)

- Available CFD solvers
- Arbitrary geometries
- Streamlined implementation of diffusion and DNP equations

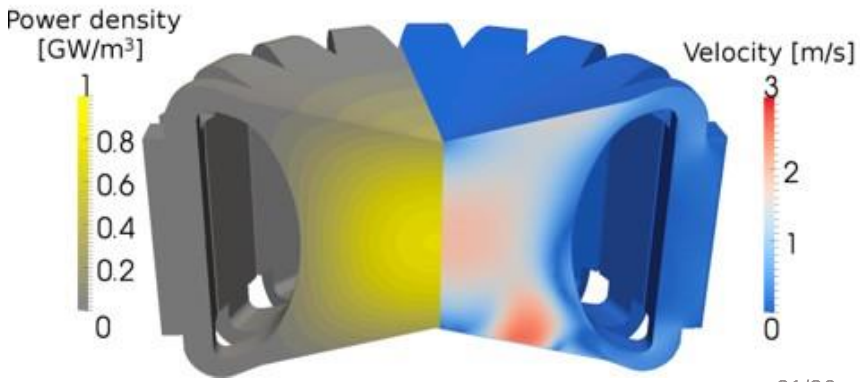
```
fvm::ddt(IV,flux_i])- fvm::laplacian(D,flux_i])= S
```

```
    fvm::ddt(alphaPtr_()*(1-eigenvalueNeutronics_), precStar_[precI])
+   fvm::Sp(lambda[precI]*alphaPtr_(), precStar_[precI])
-   neutroSource_/keff_*Beta[precI]
+   fvm::div(phiPtr_(), precStar_[precI])
-   fvm::laplacian(diffCoeffPrecPtr_(), precStar_[precI])
```

MSRE



MSFR

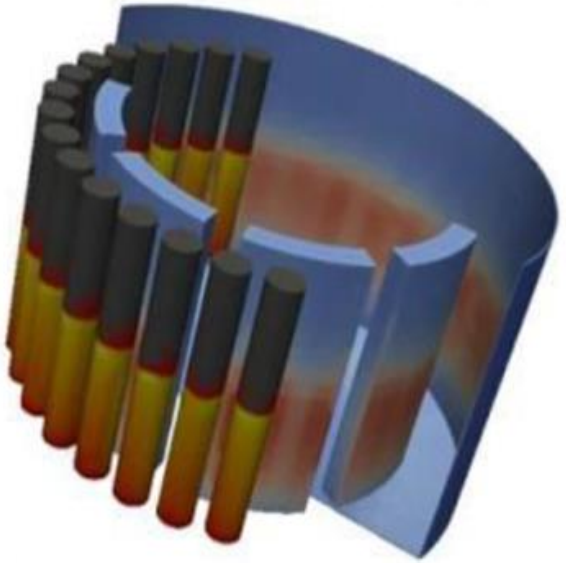


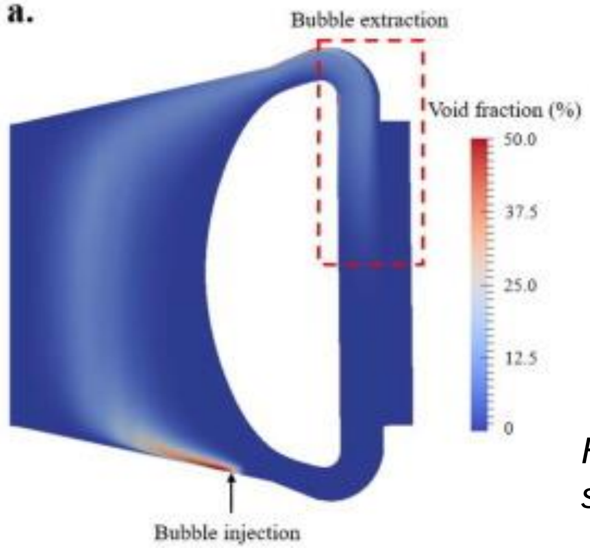
Carlo Fiorina

MSR modelling: advanced

- Available two-phase CFD solvers
- Radiative heat transfer
- Thermal-mechanics and moving mesh
- …

Dump tanks



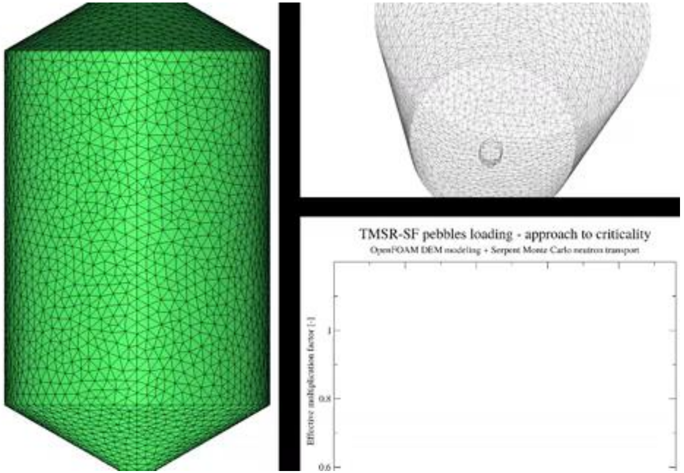


Helium sparging

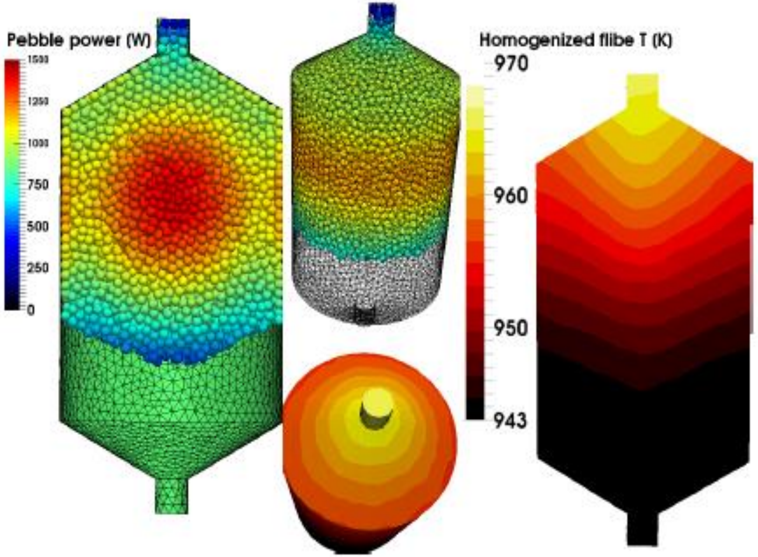
FHR modelling (UCB)

Carlo Fiorina

- Discrete Element Method + coarse-mesh thermal-hydraulics + Serpent Multi-physics interface



Helium sparging

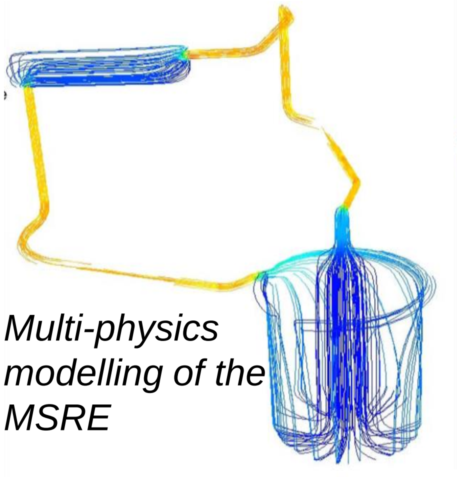


Dump tanks

GeN-Foam: Generalized Nuclear Field operation and manipulation

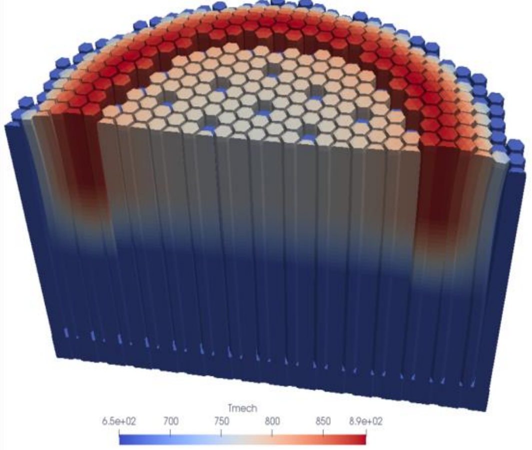
Carlo Fiorina

- First general solver for reactor safety based on OpenFOAM

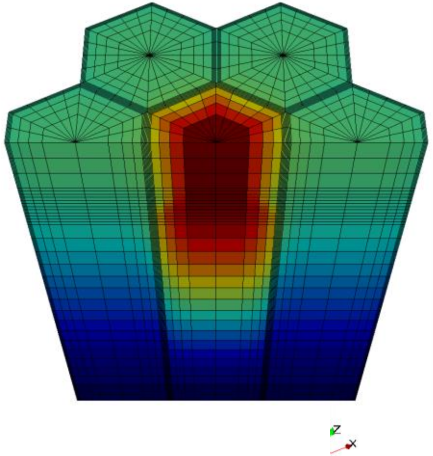


Multi-physics modelling of the MSRE

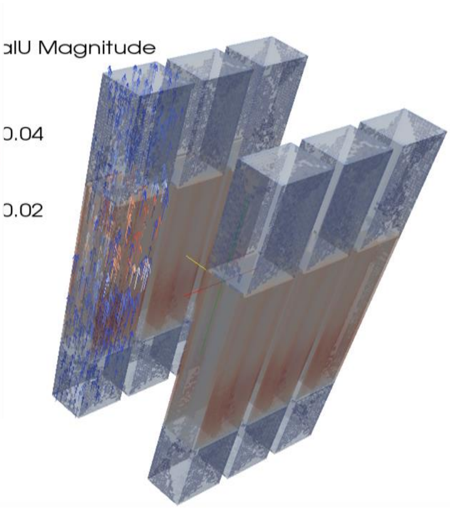
Core flowering in a SFR



Assembly windows in a SFR



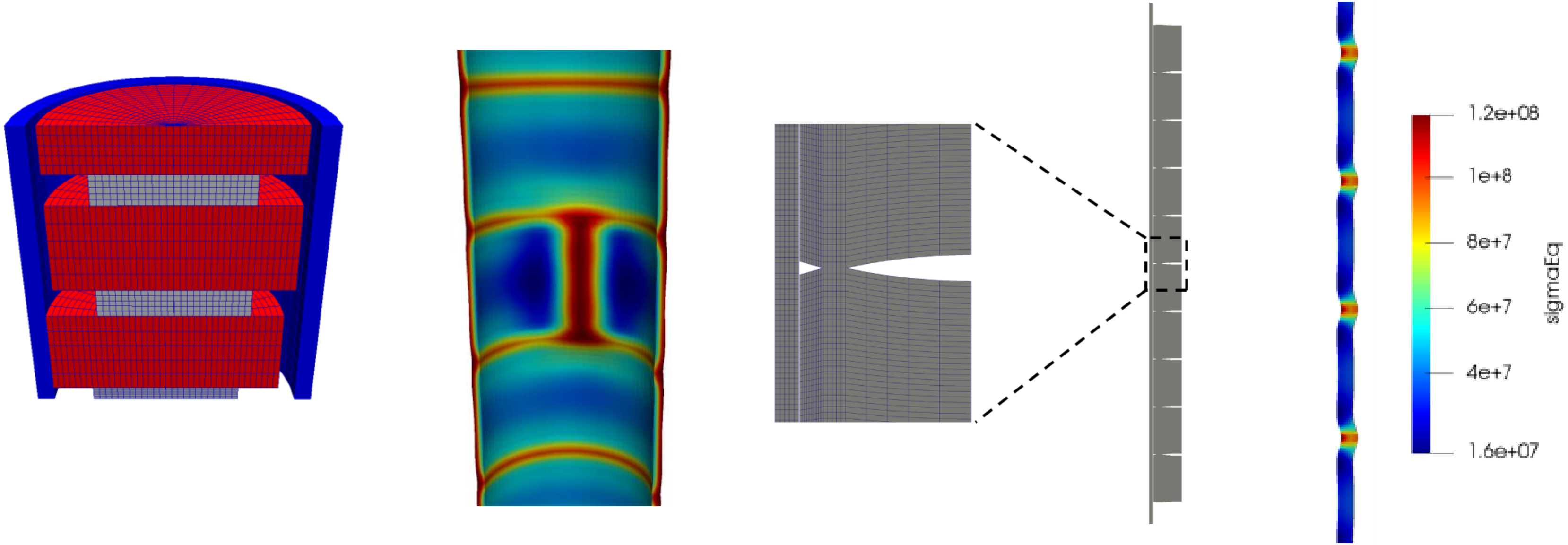
The Argonaut reactor



- Open-source + object -> use of previous work
- CFD solvers
- Thermal-mechanics solvers
- Multi-mesh with projection algorithms
- Multi-material
- Mesh deformations
- ….

OFFBEAT: OpenFoam Fuel BEhavior Tool

- Thermal-mechanics with finite volumes….

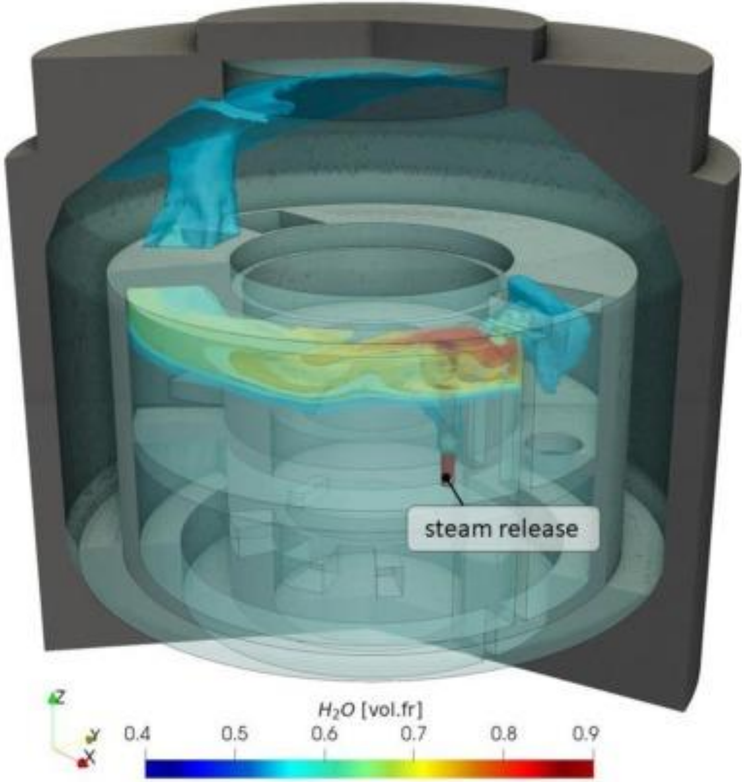


- Community contributions
- Region-coupled boundaries
- Multi-material
- …

HPC-oriented containment analysis - containmentFoam

From a general CFD tool to a nuclear-dedicated solver

- Available solvers (incl. Monte Carlo!)
- Turbulent models
- Conservative formulation
- Parallel scalability
- …



ISP-37 VANAM-M3 experiment
with containmentFOAM

Lessons learnt

One can model pretty much everything…

What's the effort?

What competences do I need?

What about the license?

What is the quality of the result?

Workflow

Downsides

- No graphical user interface (distributed with the code)
- Meshing and post-processing are performed with separate tools
- Meshing often requires proprietary tools
- Requires familiarity with Linux
- Limited documentation

Advantages

- Transparent
- Access to source code

} Better integrations of application and development

Structure of the base library

Very complete

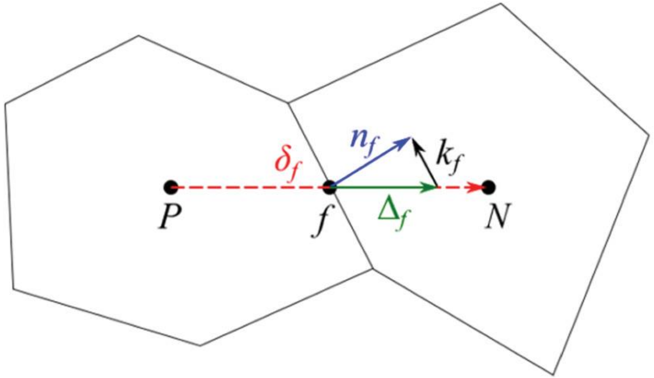
- discretization and linear system solution
- mesh-to-mesh projections
- mesh deformation
- mesh manipulation
- ordinary differential equations
- Monte Carlo methods
- octree-based mesh search
- methods for reduced-order modelling
- built-in and third-party code coupling schemes
- …

Object oriented

- encapsulation
- multi-level API

Finite volumes

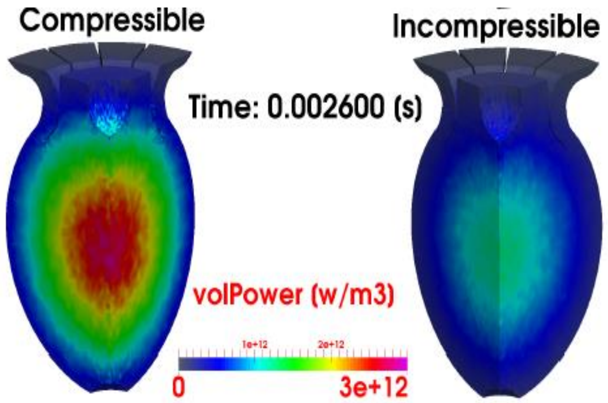
- Pros:
  - ✓ Flexible
  - ✓ Scalable
  - ✓ Conservative
  - ✓ Intuitive
  - ✓ CFD-friendly
  - ✓ Good for thermal-mechanics
  - ✓ Ok for neutronics

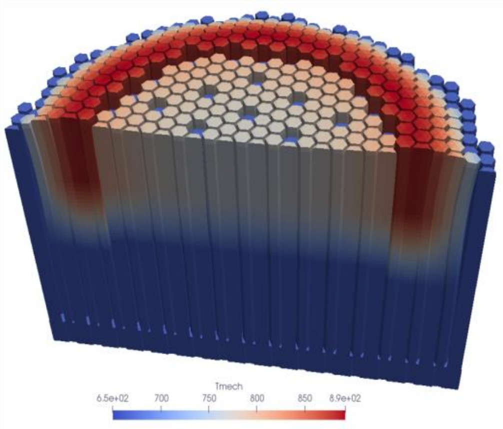


- Cons:
  - ✓ Still require familiarity with concepts associated with PDEs (well-posed problems, initial and boundary conditions), geometry creation, meshing, discretization, linear solution, etc.
  - ✓ Require good quality meshes
  - ✓ Max second order in space

Unstructured meshes

- Complete flexibility in terms of geometry -> non-traditional reactor designs and complex component
- Significant computational footprint
- First order, with all cell faces that are flat -> a high mesh resolution for curved surfaces





Operator-splitting

One matrix for each equation + iteration

Pros

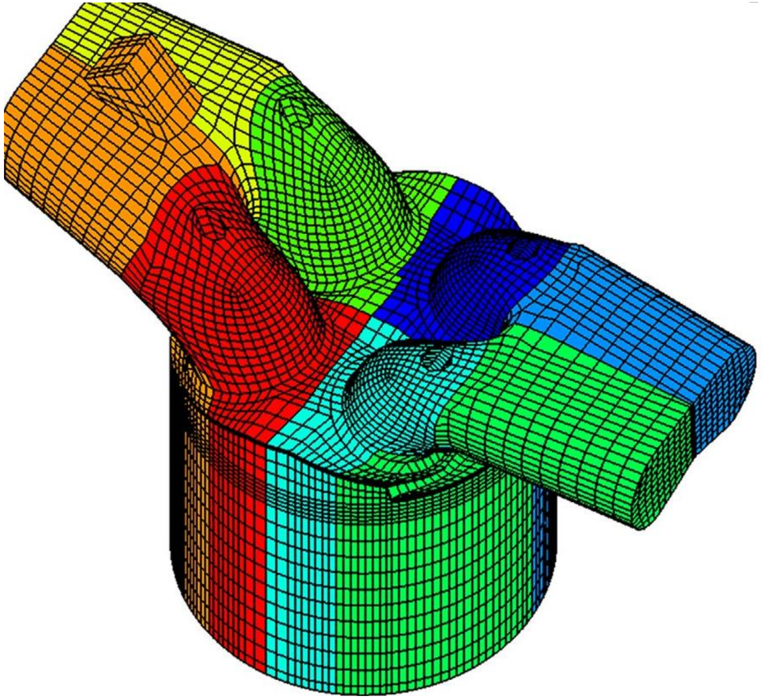
- Easier preconditioning and optimal choice of solution method
- No need to solve all physics at each coupling/time step

Cons

- Can be hard to converge for weakly-coupled / strongly non-linear equations

Parallelization

- Domain decomposition and the MPI
- Optimally scale up to few thousands of CPU cores
- Some bottlenecks
  - ✓ the sub-optimal sparse matrices storage format (LDU) that does not enable any cache-blocking mechanism (SIMD, vectorization)
  - ✓ the I/O data storage system
- The OpenFOAM HPC Technical Committee is currently working on the limitations
  - ✓ interface to external linear algebra libraries
  - ✓ recent work from NVIDIA



Computational requirements

CPU cores

- rule of thumb: 30’000 mesh cells per CPU core
- CFD
    - 2D RANS-> several hundred thousand cells -> 10 CPU cores
    - 3D RANS -> several hundred millions cells -> 5000 CPU cores
- coarse-mesh thermal-hydraulics and neutron diffusion
    - full-core models -> few hundred thousand to few million cells -> workstations or laptops

Runtime

- Steady-state simulations on the optimal number of CPU cores: several minutes to several hours
- Long-running time-dependent problems: up to a week
- In some specific applications, such as detailed containment simulations: up to a month

Memory requirements

- Single-phase RANS CFD simulation -> order of 10 fields -> 1 GB of memory per million cells
- 3D discrete ordinates neutron transport -> several thousand solution fields -> 200 GB of memory per million cells

License

- GNU GPLv3 license
  - copyleft type license: automatically affect derivative work
  - favors a collaborative development with minimal work duplication
  - limits investments from commercial players

EPFL

Thank you

Carlo Fiorina