Multiphysics Modeling Using the MOOSE Framework

Kathryn Huff
Advanced Reactors and Fuel Cycles Group
University of Illinois at Urbana-Champaign

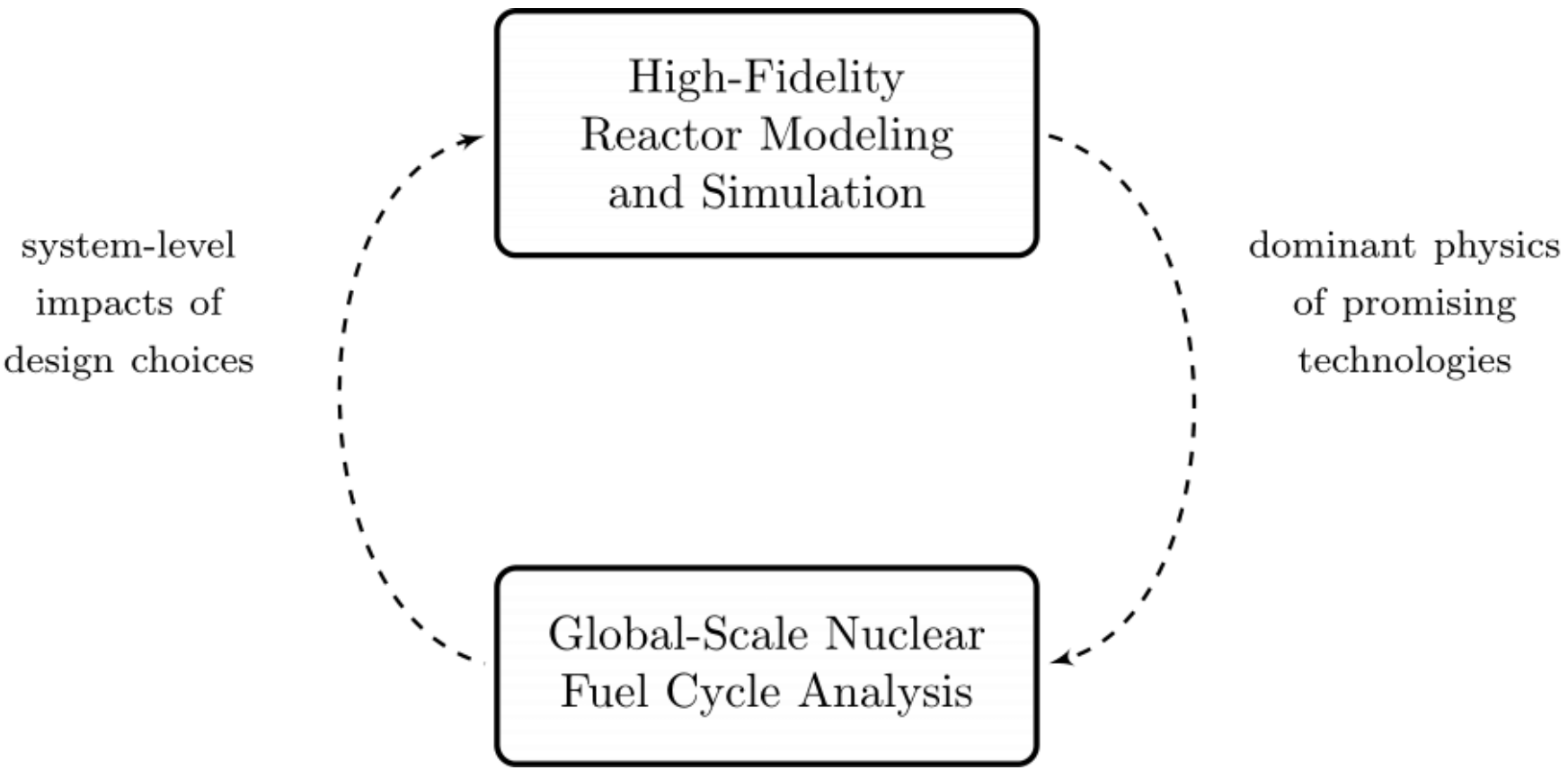
IAEA Workshop on Multiphysics Modelling to Optimize Design and Safety of Advanced Nuclear Reactors

March 17, 2021

ILLINOIS

Outline

Insights at Disparate Scales



Challenges in Liquid-Fueled Reactor Simulation

1. Contemporary burnup codes cannot treat fuel movement.
2. Neutron precursor locations drift before neutron emission.
3. Operational and safety parameters change during reactor operation.
4. Neutronics and thermal hydraulics are tightly interdependent.

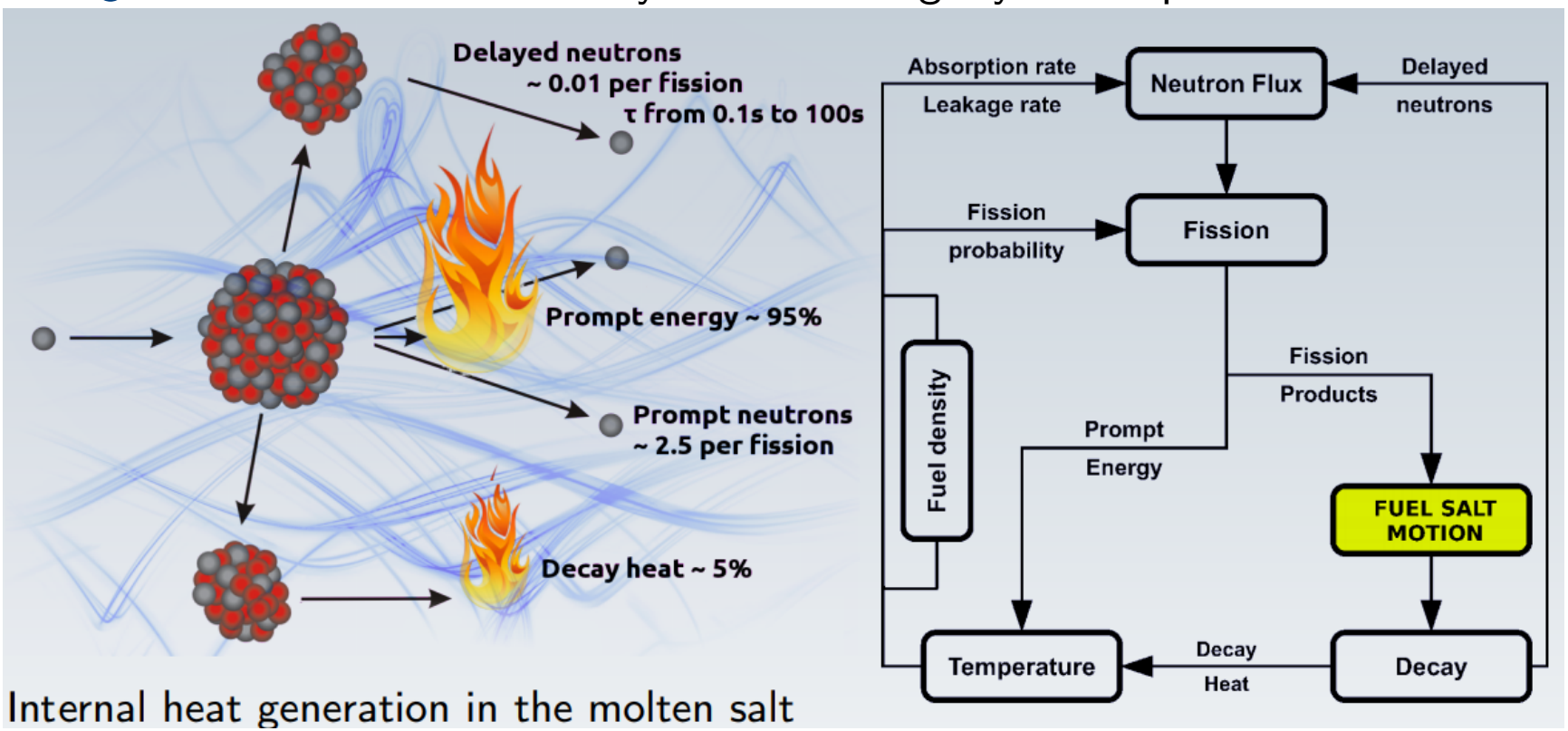


Figure: Challenges in simulating MSRs (Image courtesy of Manuele Aufiero, 2012).

Outline

MOOSE Framework

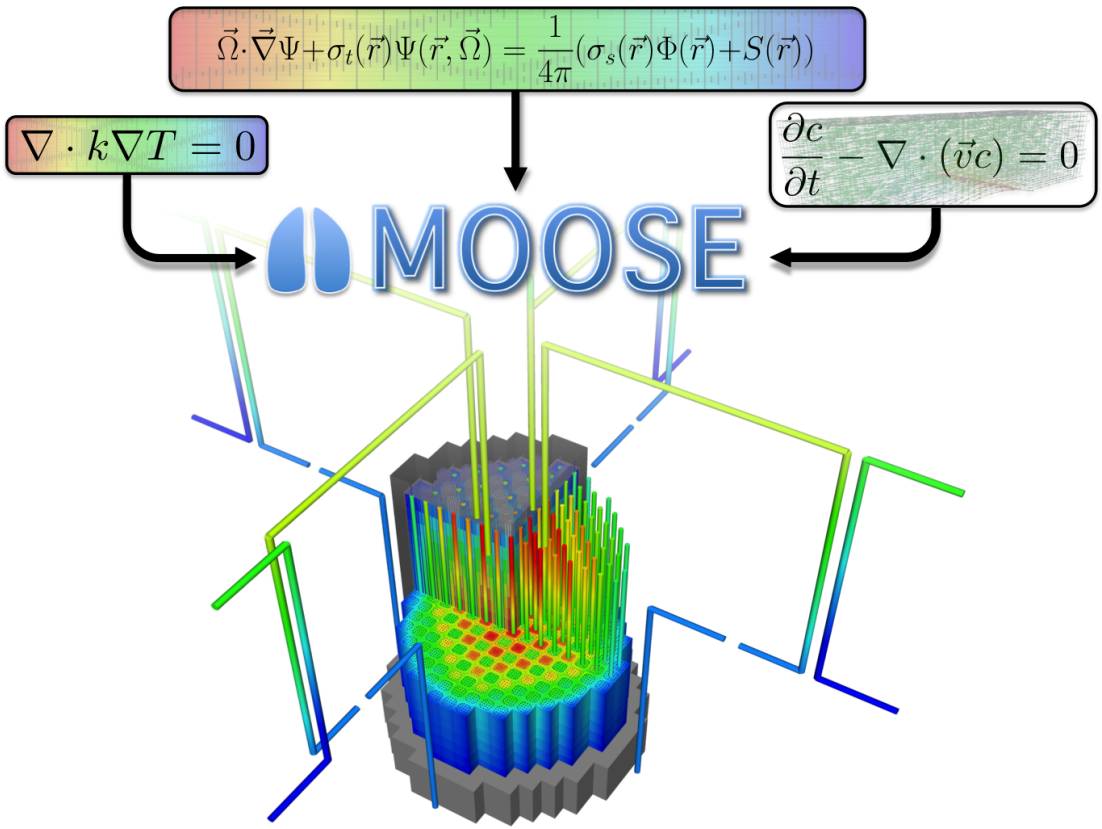


Figure: Multi-physics Object-Oriented Simulation Environment (MOOSE) [1, 5, 4].

- Developed by Idaho National Laboratory [1, 5, 4]
- Framework is truly open source (LGPL)
- Accessible docs and tutorials at https://mooseframework.inl.gov
- Source code at https://github.com/idaholab/moose

MOOSE Apps & Kernels

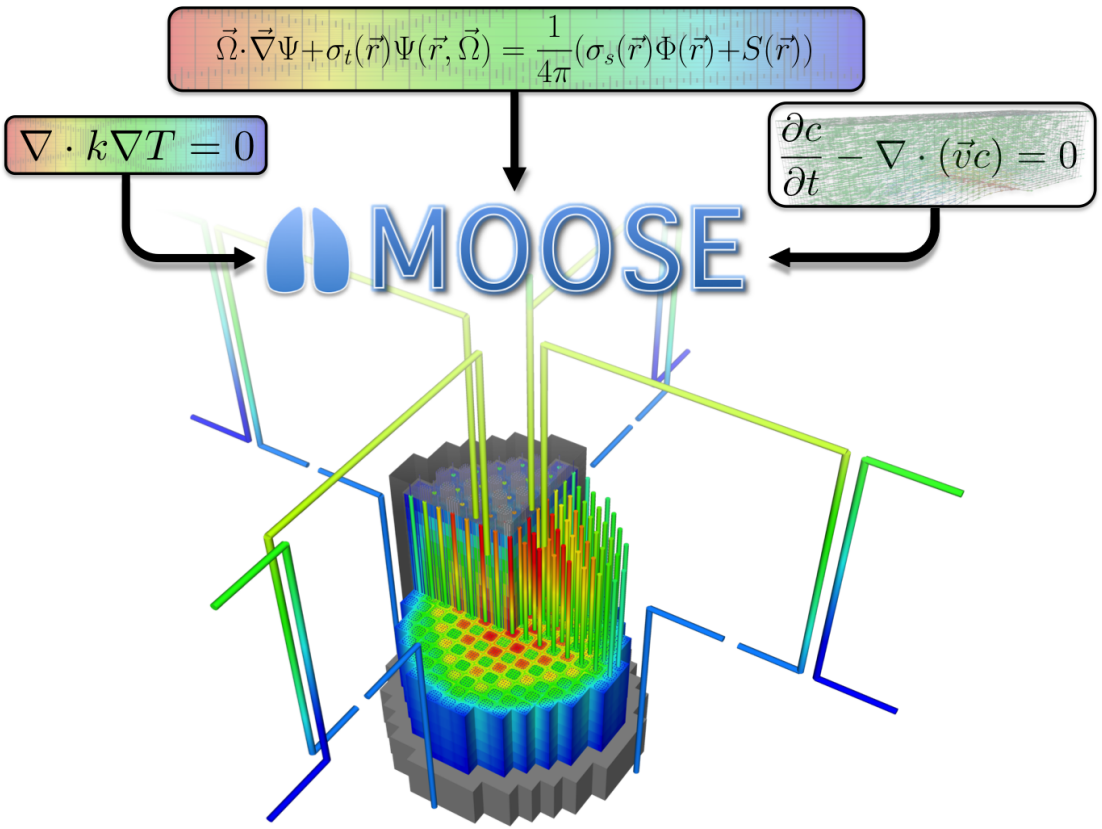


Figure: Multi-physics Object-Oriented Simulation Environment (MOOSE).

Lots of Open Kernels

- Chemical reactions
- Contact
- Fluid Properties
- Functional Expansion Tools
- Geochemistry
- Heat Conduction
- Level Set
- Navier-Stokes
- Peridynamics
- Phase Field
- Porous Flow
- Ray Tracing
- Reconstructed Discontinous Galerkin
- Tensor Mechanics
- ...

MOOSE Apps & Kernels

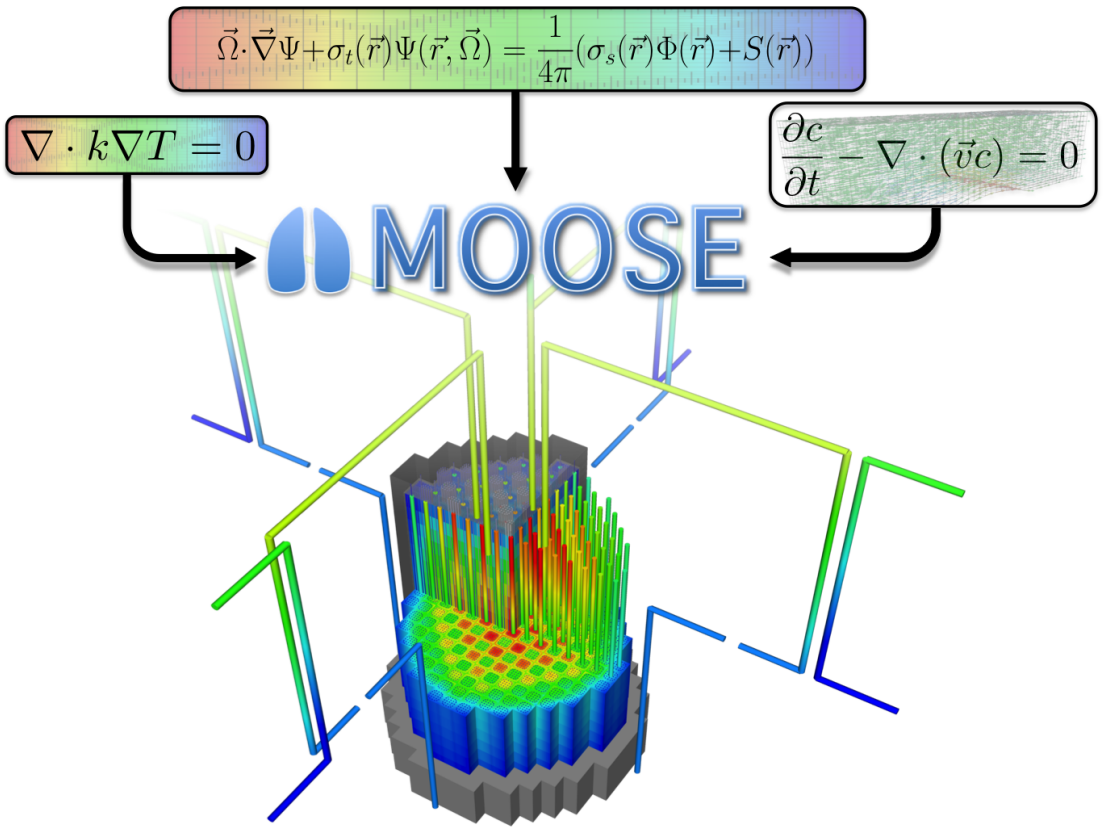


Figure: Multi-physics Object-Oriented Simulation Environment (MOOSE).

Lots of Open Apps

- Moltres (MSRs) [?]
- Squirrel (Utilities)
- Mastodon (structural dynamics, seismology)
- pika (microstructure)
- falcon
- blackbear
- crane (plasma chemistry)
- WhALE (fluid-structure mechanics)

MOOSE Apps & Kernels

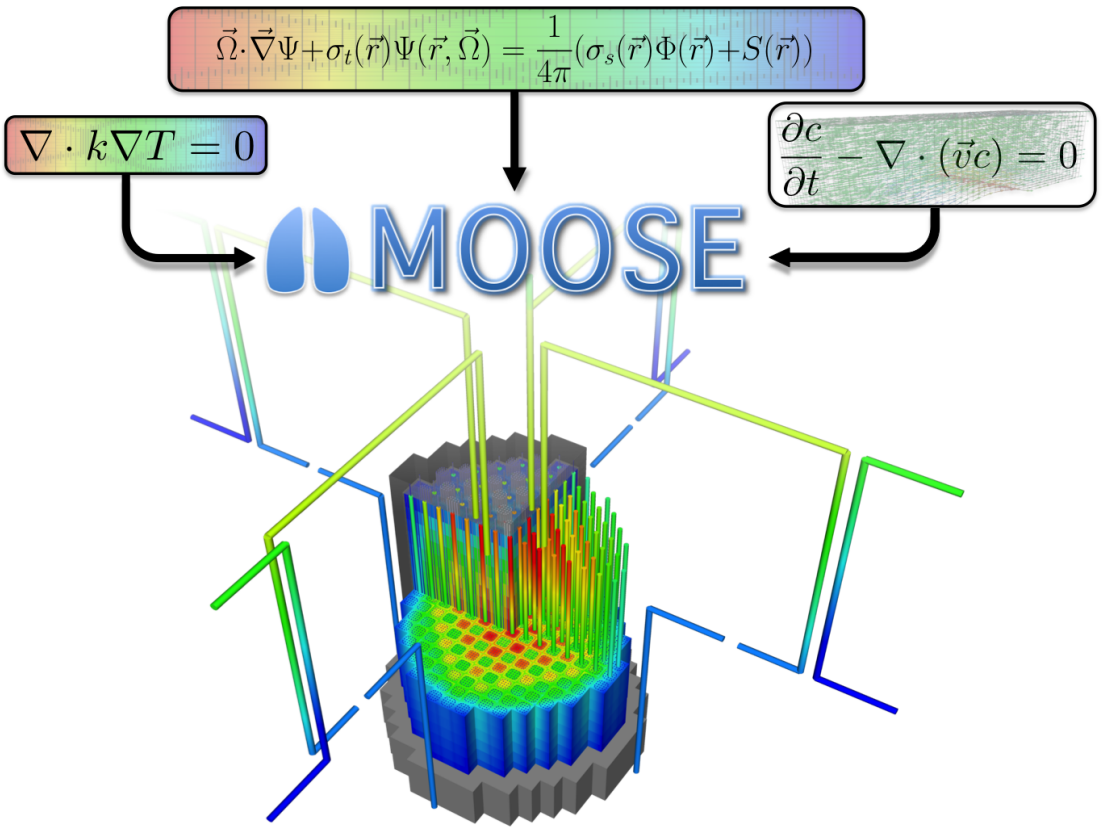


Figure: Multi-physics Object-Oriented Simulation Environment (MOOSE).

Lots of Restricted Apps

- BISON
- Marmot
- RattleSNake
- Pronghorn
- RELAP-7
- ...

How Does it Work?

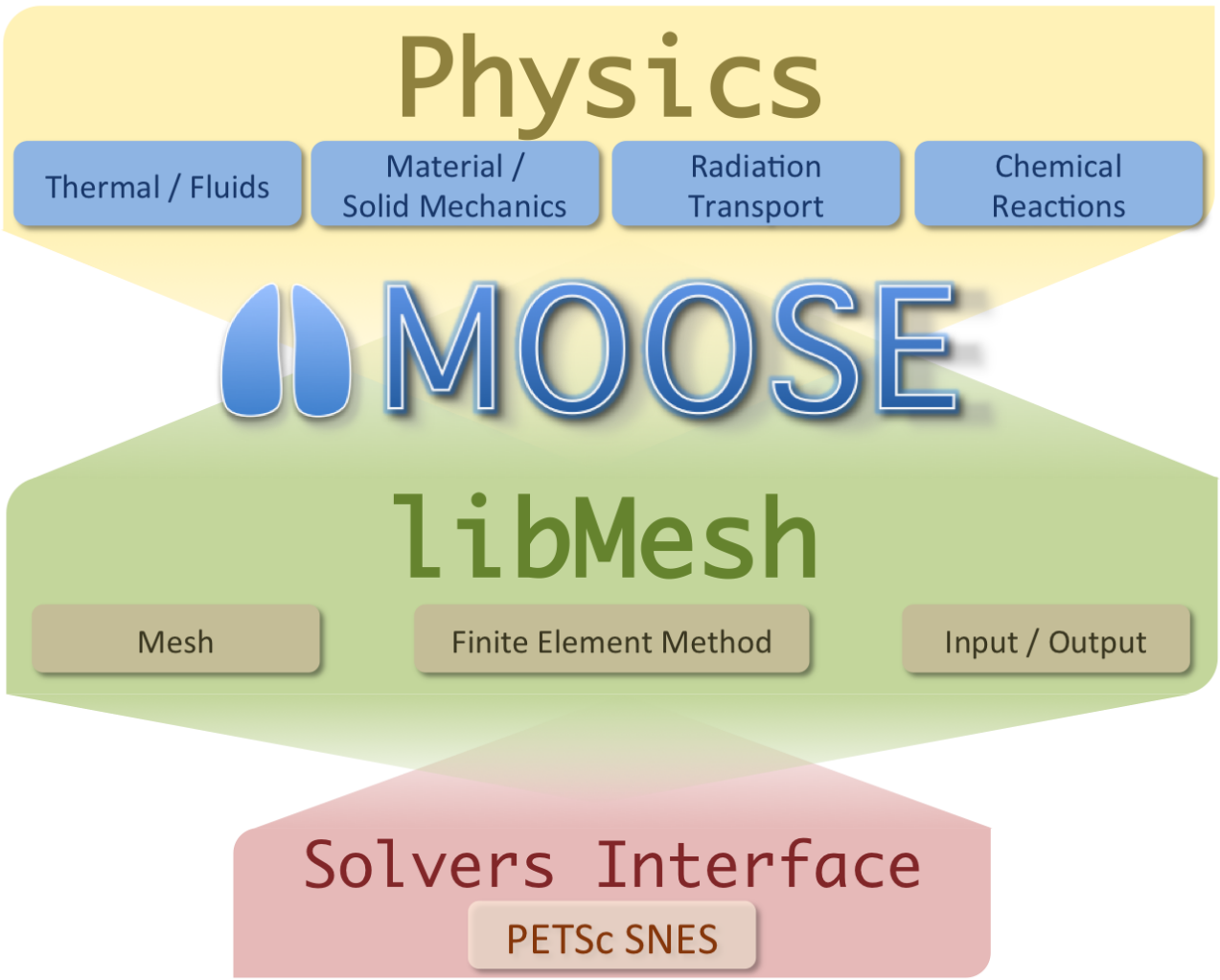


Figure: Shamelessly copied from the MOOSE Team Workshop slides.

How Does it Work?

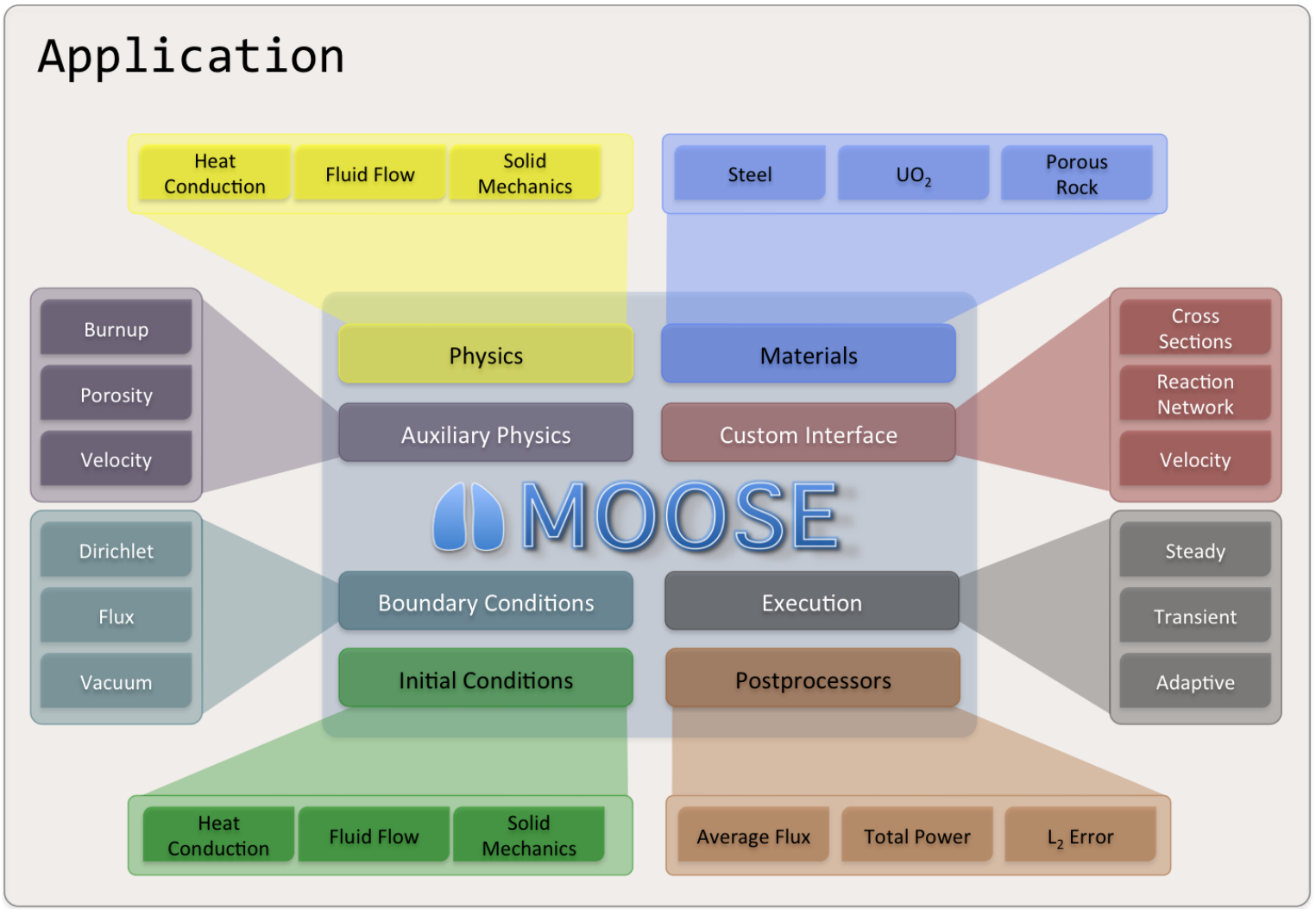


Figure: Shamelessly copied from the MOOSE Team Workshop slides.

How Does it Work?

Strong Form

$$\rho Cp\frac{\partial T}{\partial t} - \nabla \cdot k(T,B)\nabla T = f$$

Weak Form

$$\int_\Omega \rho Cp\frac{\partial T}{\partial t}\psi_i + \int_\Omega k\nabla T \cdot \nabla\psi_i - \int_{\partial\Omega} k\nabla T \cdot \mathbf{n}\psi_i - \int_\Omega f\psi_i = 0$$

Kernel Kernel BoundaryCondition Kernel

Actual Code

```
return _k[_qp]*_grad_u[_qp]*_grad_test[_i][_qp];
```

Figure: Shamelessly copied from the MOOSE Team Workshop slides.

MOOSE: Key Features

- MOOSE Framework is truly open source (LGPL)
- Developed initially for nuclear applications
- Signficant long-term support from US DOE
- Continuous integration support (CIVET)
- Intuitive parallel multiscale solves
- Easy developer onboarding

- Object Oriented, C++
- Interfaces with libMesh to discretize simulation volume into finite elements
- Residuals and Jacobians handed off to PetSc which handles solution of resulting non-linear system of algebraic equations
- Fully-coupled, fully-implicit multiphysics solver
- Automatically parallel (largest runs >100,000 CPU cores!)
- Built-in adaptive meshing & timestepping

Pros and Cons

Pros (+)

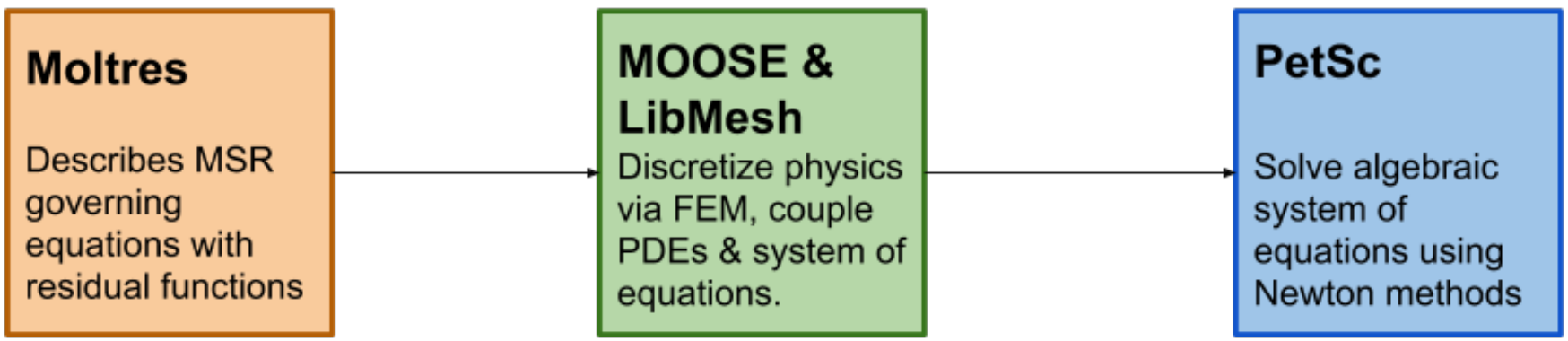
- LGPL means the Framework is open, but apps can be restricted
- Vast array of available apps and kernels
- Many solver and preconditioning options
- Finite Element Modeling
- Full coupling is optional
- Generates gorgeous visualizations

Cons (-)

- LGPL means the Framework is open, but apps can be restricted
- Vast array of available apps and kernels
- Many solver and preconditioning options
- Finite Element Modeling
- Full coupling is optional
- Generates gorgeous visualizations

Outline

Moltres: Coupling in MOOSE



Moltres: Basics

- Developed in ARFC group
- Fluid-fuelled, molten salt reactors
- Multi-group diffusion (arbitrary groups)
- Advective movement of delayed neutron precursors
- Navier-Stokes thermal hydraulics
- 3D unstructured
- 2D axisymmetric
- 3D structured
- Initial developer: Alexander Lindsay [3]

Acquiring Moltres

```
git clone https://github.com/arfc/moltres
cd moltres
git submodule init
git submodule update
```

Diffusion in Moltres

$$\frac{1}{v_g}\frac{\partial \phi_g}{\partial t} - \nabla \cdot D_g \nabla \phi_g + \Sigma_g^r \phi_g = \tag{1}$$

$$\sum_{g \neq g'}^{G} \Sigma_{g' \rightarrow g}^s \phi_{g'} + \chi_g^p \sum_{g'=1}^{G} (1-\beta)\nu \Sigma_{g'}^f \phi_{g'} + \chi_g^d \sum_{i}^{I} \lambda_i C_i \tag{2}$$

v_g = speed of neutrons in group g

ϕ_g = flux of neutrons in group g

t = time

D_g = Diffusion coefficient for neutrons in group g

Σ_g^r = macroscopic cross-section for removal of neutrons from group g

$\Sigma_{g' \rightarrow g}^s$ = macroscopic cross-section of scattering from g' to g

χ_g^p = prompt fission spectrum, neutrons in group g

G = number of discrete groups, g

ν = neutrons produced per fission

Σ_g^f = macroscopic fission cross section due to neutrons in group g

χ_g^d = delayed neutrons in group g

I = delayed neutron precursor groups

β = delayed neutron fraction

λ_i = average decay constant of delayed neutron precursors in group i

C_i = concentration of delayed neutron precursors in precursor group i

.

Moltres Delayed Neutrons

$$\frac{\partial C_i}{\partial t} = \sum_{g'=1}^{G} \beta_i \nu \Sigma_{g'}^{f} \phi_{g'} - \lambda_i C_i - \frac{\partial}{\partial z} u C_i \tag{3}$$

G = number of discrete groups, g

I = delayed neutron precursor groups

C_i = concentration of delayed neutron precursors in precursor group i

$.u$ = vertical fluid velocity

λ_i = average decay constant of delayed neutron precursors in group i

β = fraction of delayed neutron precursors in group i

Moltres Fuel Temperature

$$\rho_f c_{p,f} \frac{\partial T_f}{\partial t} + \nabla \cdot (\rho_f c_{p,f} \vec{u} \cdot T_f - k_f \nabla T_f) = Q_f \tag{4}$$

$$\rho_f = \text{density of fuel salt} \tag{5}$$

$$c_{p,f} = \text{specific heat capacity of fuel salt} \tag{6}$$

$$T_f = \text{temperature of fuel salt} \tag{7}$$

$$\vec{u} = \text{velocity of fuel salt} \tag{8}$$

$$k_f = \text{thermal conductivity of fuel salt} \tag{9}$$

$$Q_f = \text{source term} = \sum_{g=1}^{G} \epsilon_{f,g} \Sigma_{f,g} \phi_g \tag{10}$$

Moltres Moderator Temperature

$$\rho_g c_{p,g} \frac{\partial T_g}{\partial t} + \nabla \cdot (-k_g \nabla T_g) = Q_g \tag{11}$$

(12)

ρ_g = density of graphite moderator (13)

$c_{p,g}$ = specific heat capacity of graphite moderator (14)

T_g = temperature of graphite moderator (15)

k_g = thermal conductivity of graphite moderator (16)

Q_g = source term in graphite moderator (17)

(18)

Moltres MSRE Simulation

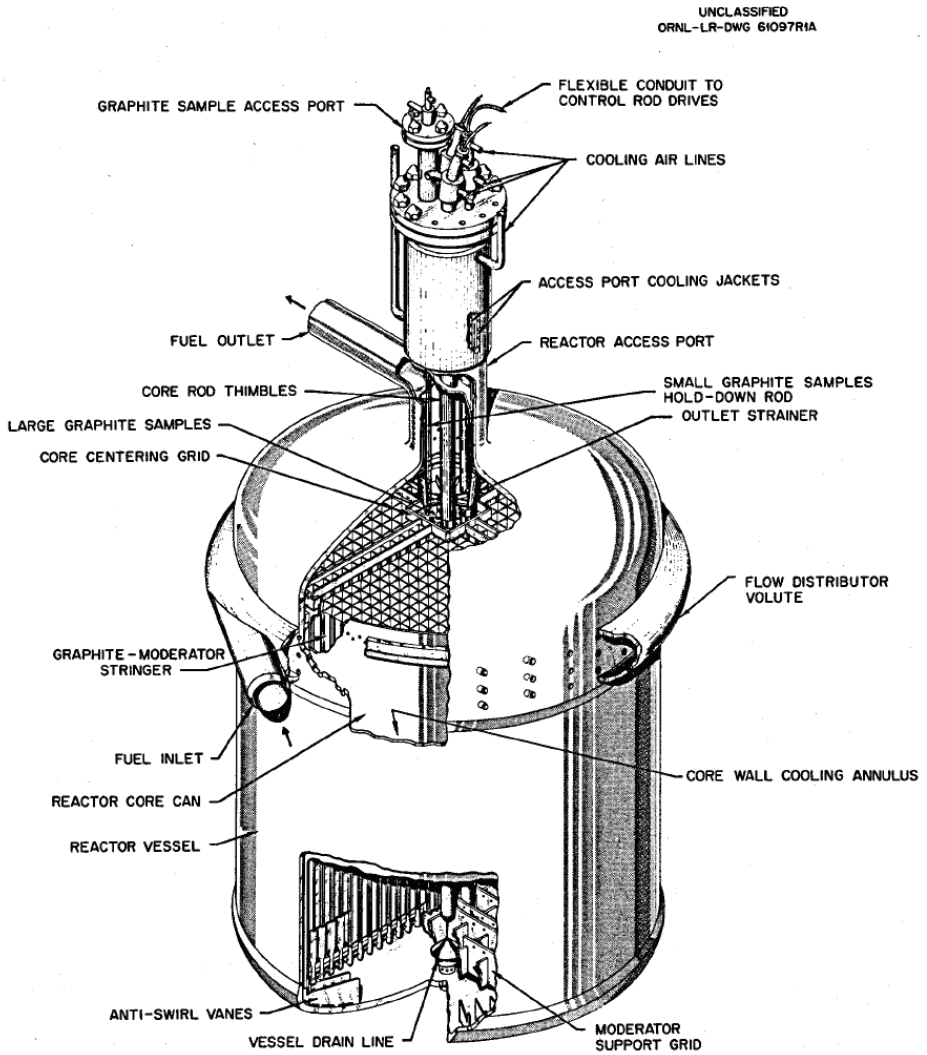
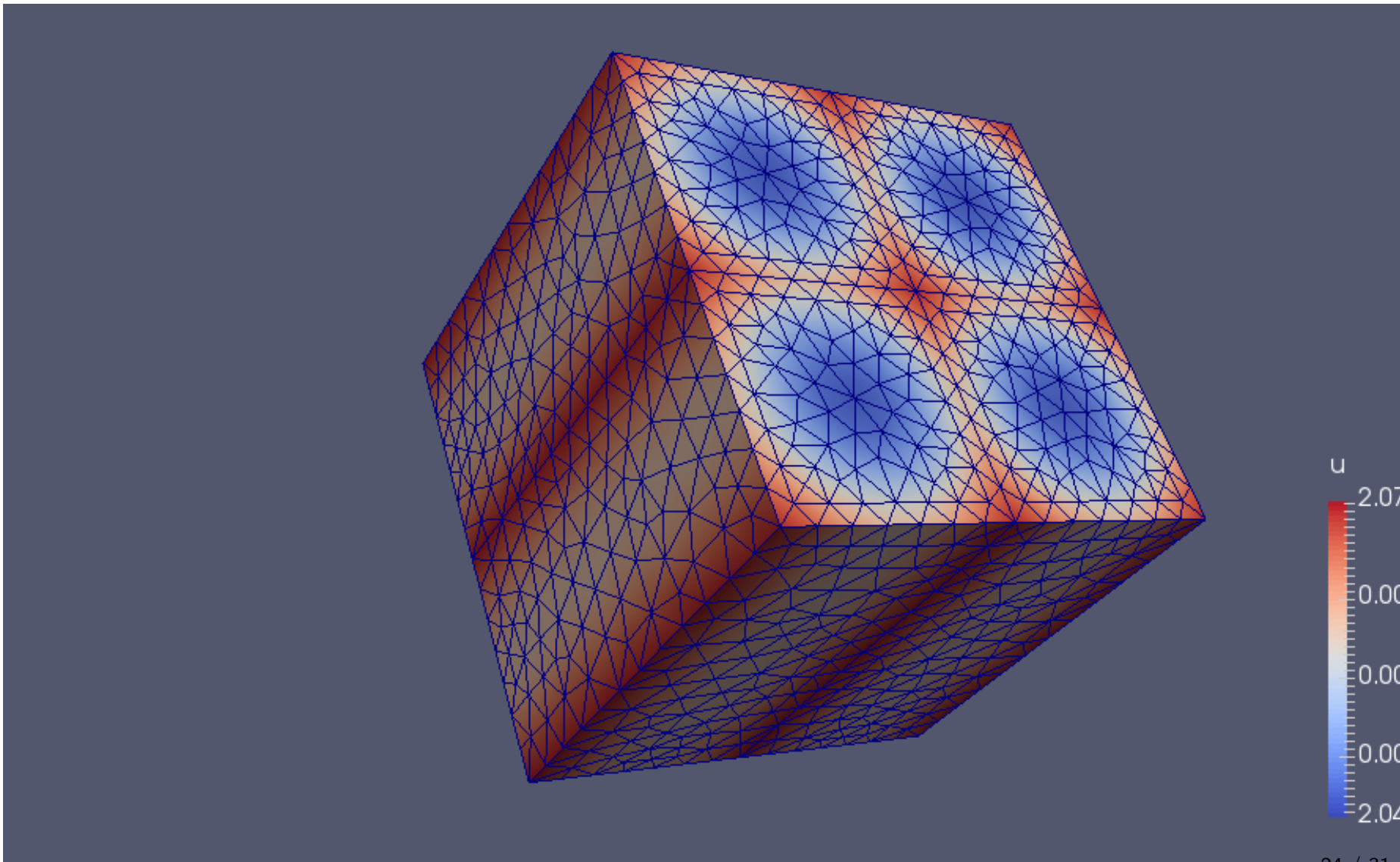
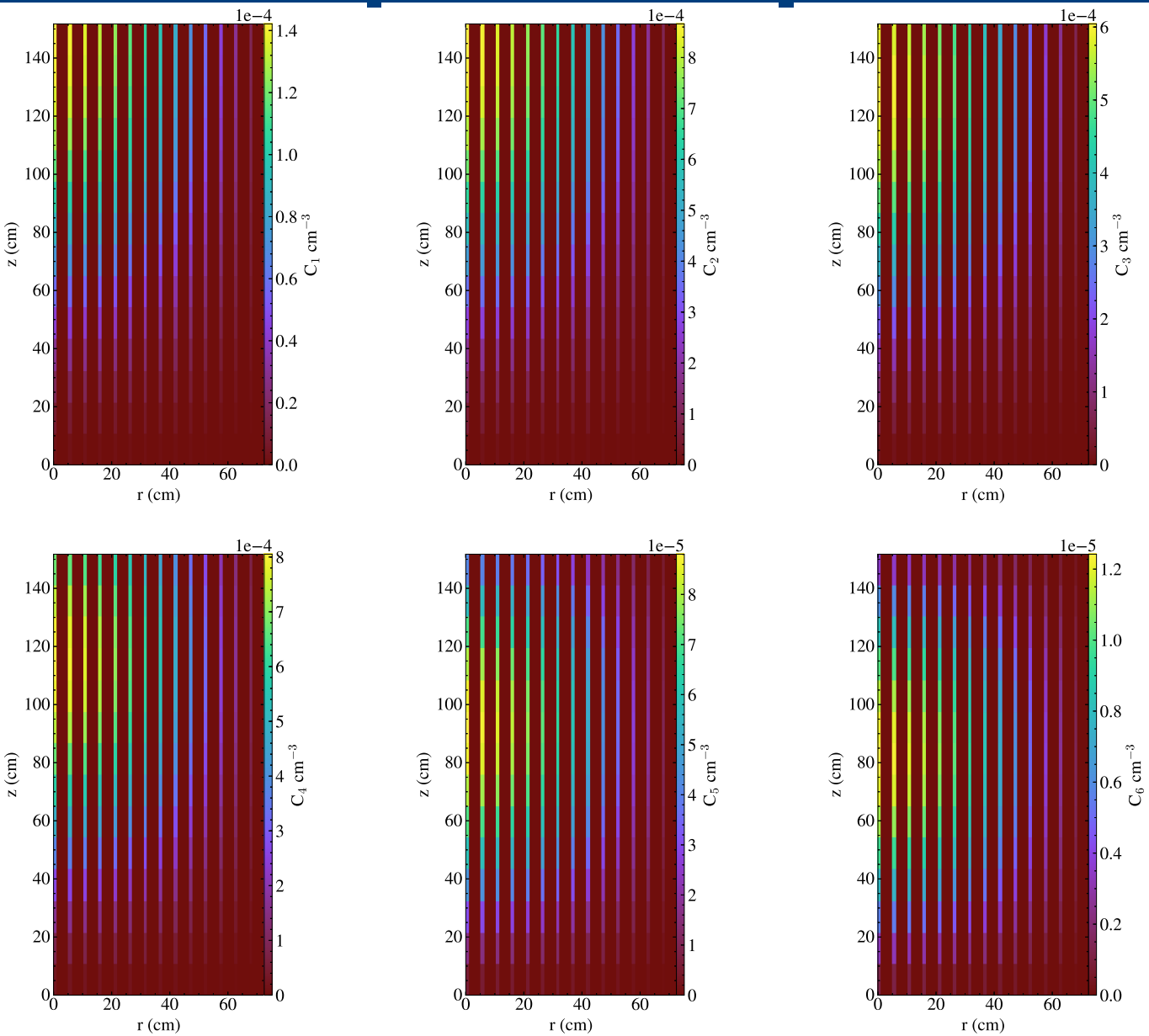


Fig. 6. MSRE Reactor Vessel.

Mesh Generation for MOOSE Apps like Moltres



Moltres Precursor Drift



Moltres: More Complex Mesh

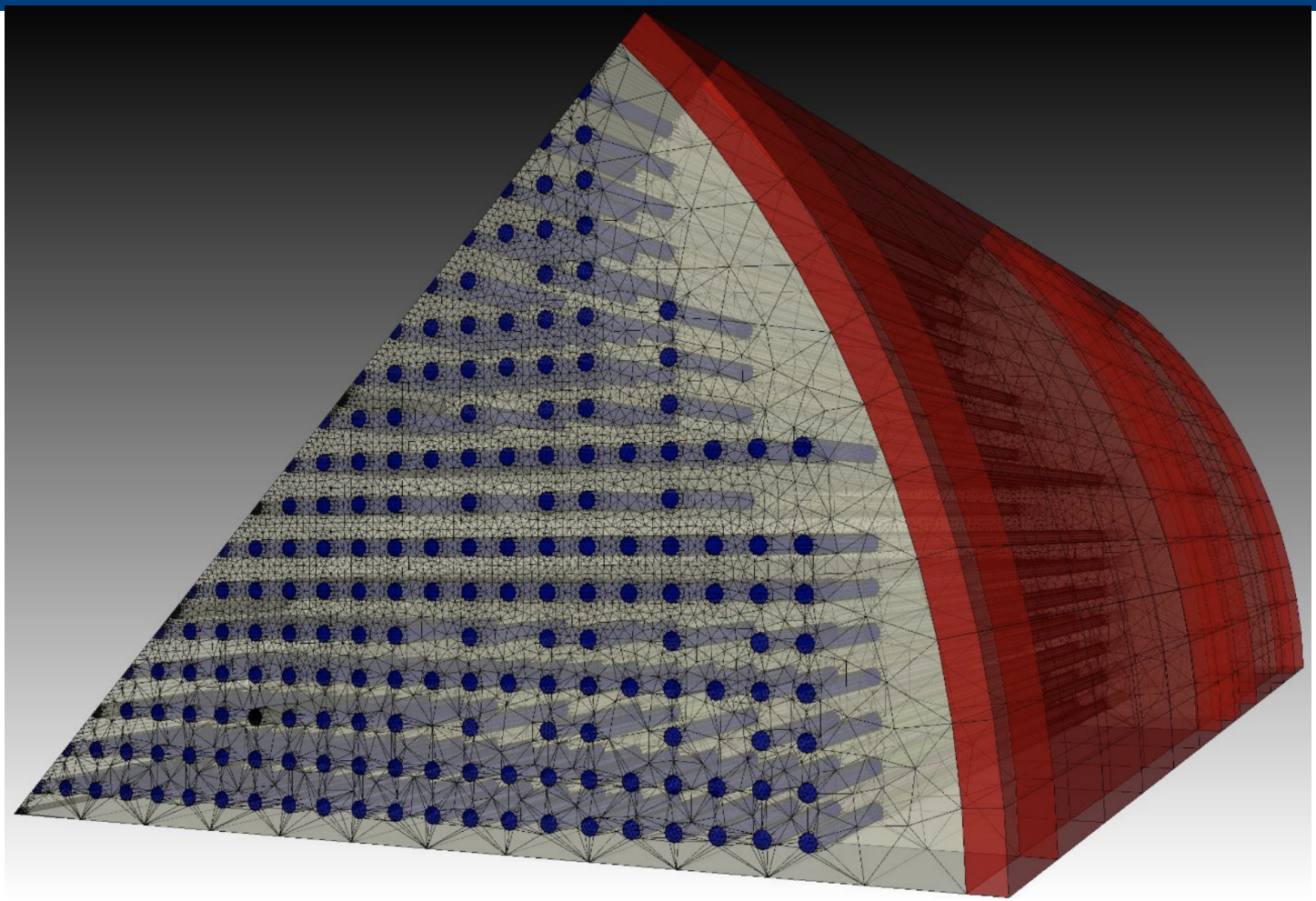


Figure: TAP Mesh generated by Alvin Lee [2]. Red = Reactor Vessel Wall, Light Yellow = Fuel Salt, Dark Gray = Control Rods, Blue = Fuel Salt radially co-located with the Moderator Rods.

Moltres: Multiphysics simulation (3D)

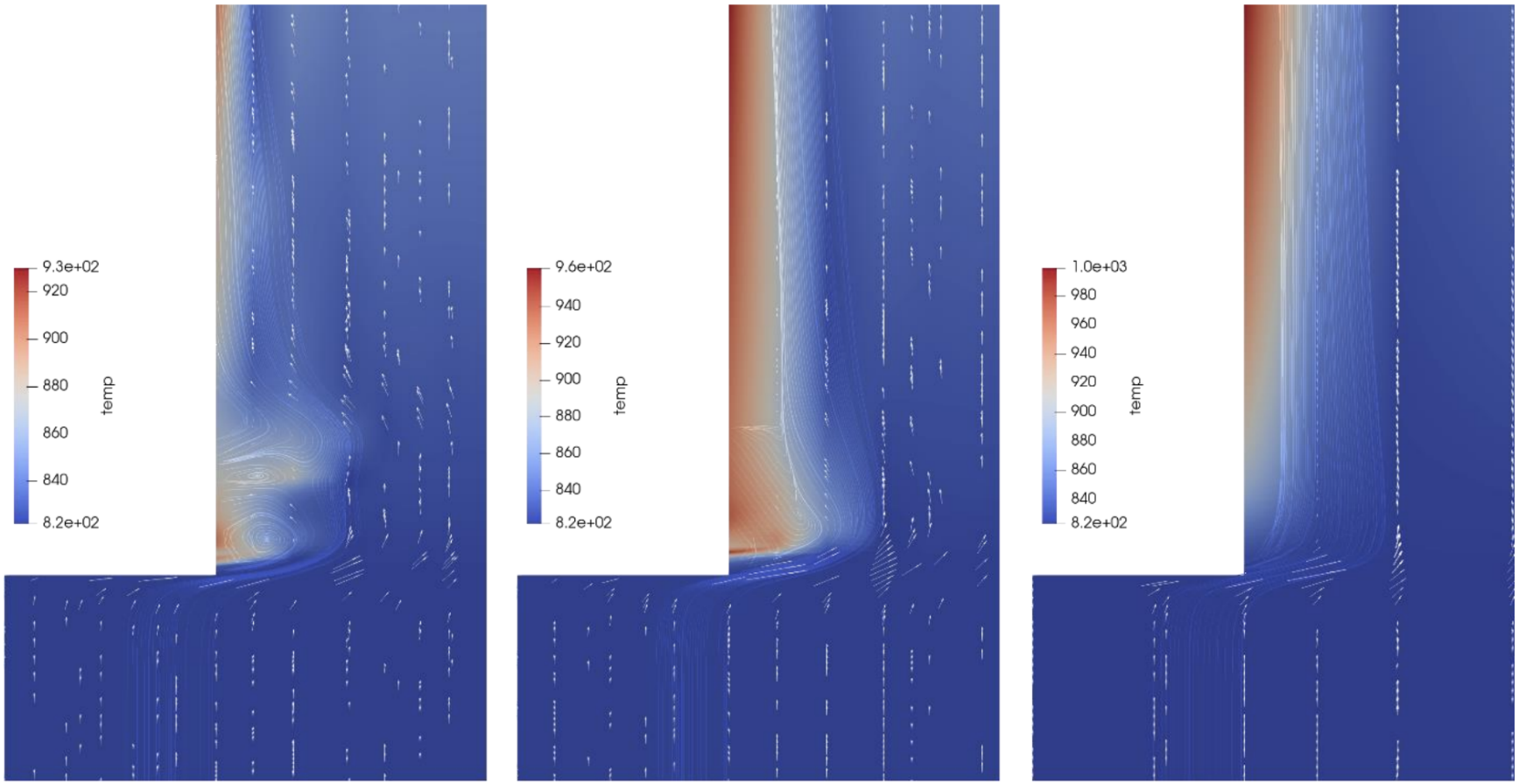


Figure: Meshing study by Alvin Lee [2] regarding KH instabilities and resolution of MSR fuel salt vortices.

What now?

Get Started

https://mooseframework.inl.gov/ https://github.com/idaholab/moose

https://github.com/arfc/moltres

Other Tools

https://gmsh.info

https://github.com/pyne/pyne

https://github.com/openmc-dev/openmc

https://www.paraview.org

Acknowledgements

- This research is part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois.
- Kathryn Huff is additionally supported by the NRC Faculty Development Program, the NNSA (awards DE-NA0002576 and DE-NA0002534), and the International Institute for Carbon Neutral Energy Research (WPI-I2CNER).
- The authors would like to thank members of Advanced Reactors and Fuel Cycles research group (ARFC) at the University of Illinois at Urbana Champaign who provided valuable code reviews and proofreading.
- This work is derived from the work of Alex Lindsay (Idaho National Laboratory), Gavin Ridley (University of Tennessee-Knoxville), Andrei Rykhlevskii (Argonne National Laboratory), Sun Myung Park (UIUC), and Alvin Lee (UIUC).

References I

[1] Derek R. Gaston, Cody J. Permann, John W. Peterson, Andrew E. Slaughter, David Andrš, Yaqi Wang, Michael P. Short, Danielle M. Perez, Michael R. Tonks, Javier Ortensi, Ling Zou, and Richard C. Martineau.

Physics-based multiscale coupling for full core nuclear reactor simulation.

Annals of Nuclear Energy, 84:45–54, October 2015.

[2] Alvin J. H. Lee.

Neutronics and Thermal-Hydraulics Analysis of TransAtomic Power Molten Salt Reactor (TAP MSR) Core Under Load Following Operations.

Master's thesis, University of Illinois at Urbana-Champaign, Urbana, IL, December 2020.

[3] Alexander Lindsay, Gavin Ridley, Andrei Rykhlevskii, and Kathryn Huff.

Introduction to Moltres: An application for simulation of Molten Salt Reactors.

Annals of Nuclear Energy, 114:530–540, April 2018.

[4] Alexander Lindsay, Roy Stogner, Derek Gaston, Daniel Schwen, Christopher Matthews, Wen Jiang, Larry K Aagesen, Robert Carlsen, Fande Kong, Andrew Slaughter, et al.

Automatic differentiation in metaphysicl and its applications in moose.

Nuclear Technology, pages 1–18, 2021.

References II

[5] Cody J. Permann, Derek R. Gaston, David Andrš, Robert W. Carlsen, Fande Kong, Alexander D. Lindsay, Jason M. Miller, John W. Peterson, Andrew E. Slaughter, Roy H. Stogner, and Richard C. Martineau.

MOOSE: Enabling massively parallel multiphysics simulation.

SoftwareX, 11:100430, 2020.