INL/CON-22-69936-Revision-0

Cross-Cutting Capability in MOOSE for Advanced Reactor Simulation

February 2024

Changing the World's Energy Future

Alexander D Lindsay, Guillaume Louis Giudicelli, Roy Hulen Stogner

Idaho National Laboratory

INL/CON-22-69936-Revision-0

Cross-Cutting Capability in MOOSE for Advanced Reactor Simulation

Alexander D Lindsay, Guillaume Louis Giudicelli, Roy Hulen Stogner

February 2024

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

U.S. DEPARTMENT OF ENERGY | Office of NUCLEAR ENERGY

Cross-Cutting Capability in MOOSE for Advanced Reactor Simulation

Alex Lindsay, Guillaume Giudicelli, Roy Stogner

NEAMS
Nuclear Energy Advanced Modeling and Simulation

November 2, 2022

NEAMS Annual Review Meeting

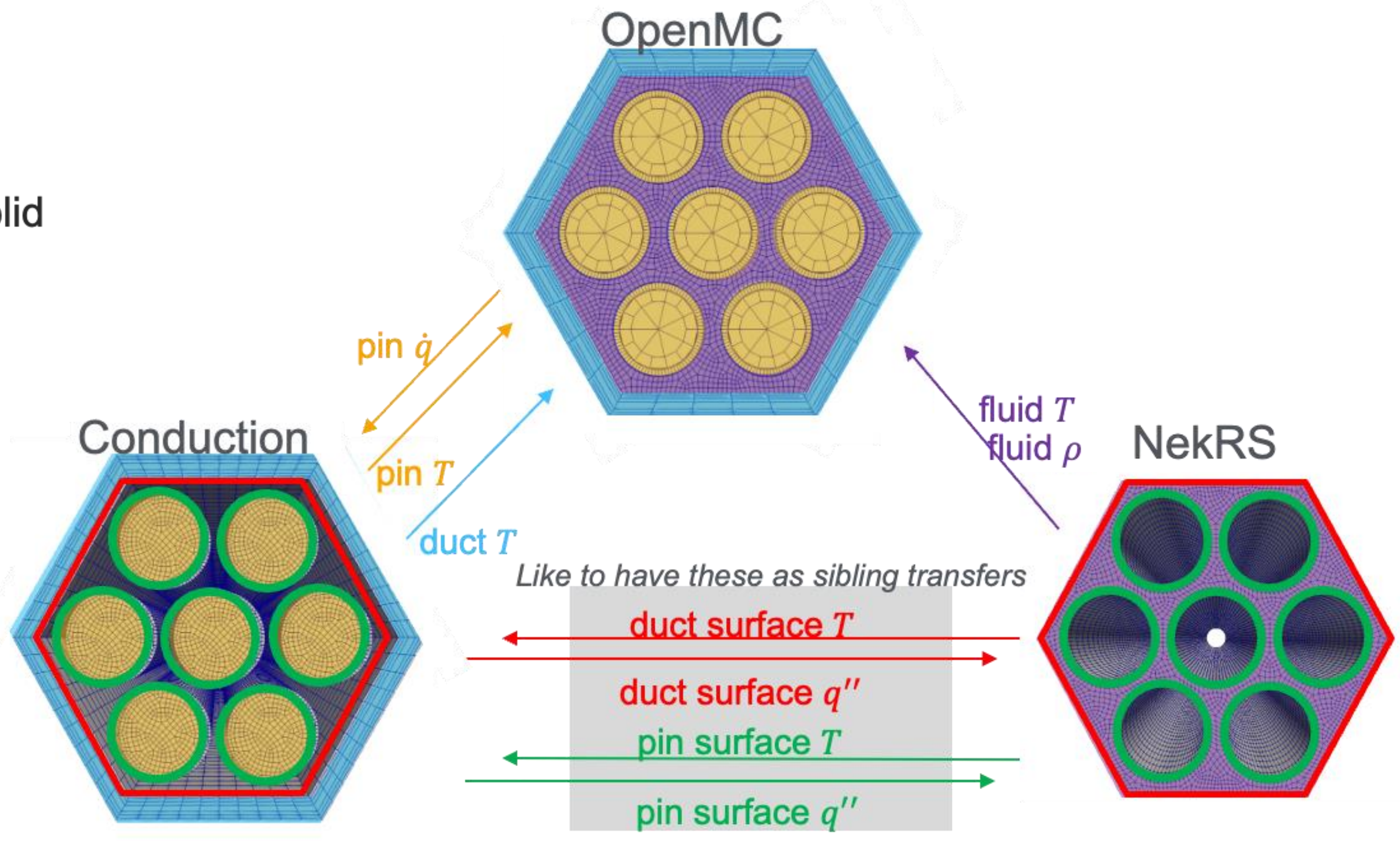
Outline

- User-Oriented Improvements and Optimizations
  - Sibling transfers – Transfer *between* MultiApps
  - General coordinate transformations – Rotations, scaling, different coordinate system types
  - Face variables – Useful for fluxes for hybrid FEM, finite volume methods
  - On-the-fly evaluation system – Construct residuals/Jacobians for highly arbitrary element stencils
  - Native Delaunay triangulation – Support more arbitrary meshes through the native MeshGenerator system
  - Reactor module meshing enhancements – Capable of meshing just about any advanced reactor type
- Technical Area Support
  - General support: Depth-first search for dependency resolution, within-group user object sorting, residual and Jacobian computed together
  - Fluids support: Mortar for finite volume, general advection schemes
  - Neutronics support: Matrix-only solve type for eigen problems, more robust lower-D element ghosting
  - Thermomechanics support: Selective reinitialization of materials for mortar
- Future FY 23 work

Siblings transfers: example use case 1

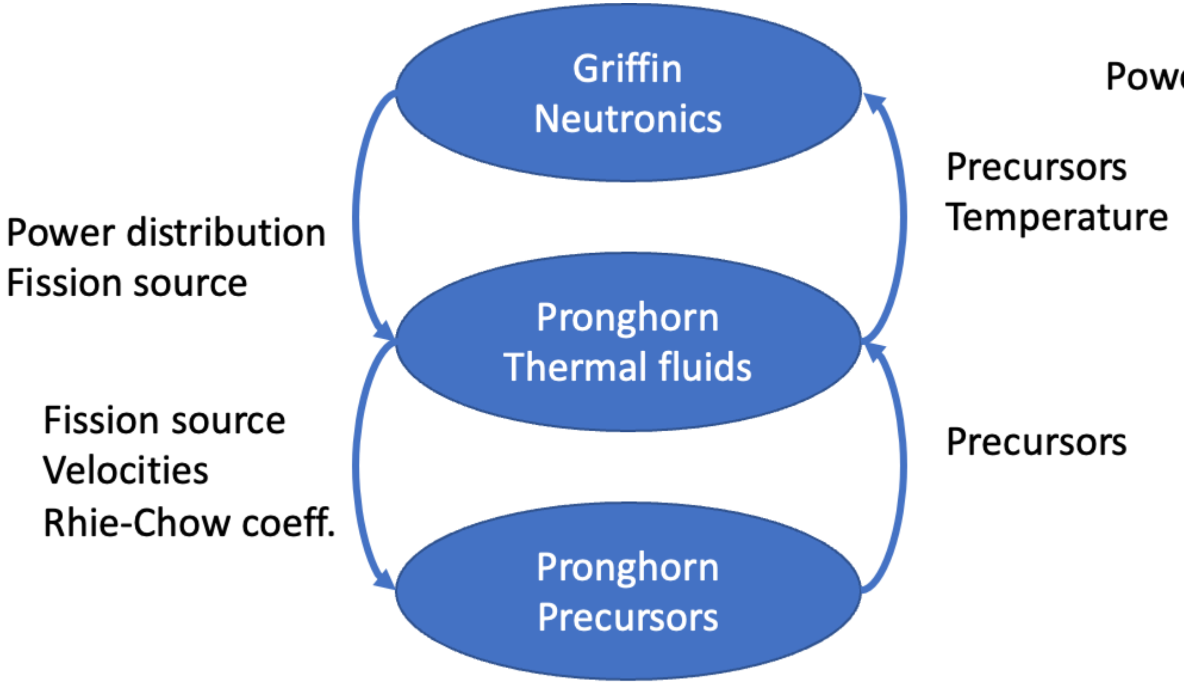
- A typical example from A. Novak
- **Cardinal**: common use case
  - OpenMC main application
  - MOOSE heat conduction sub-application (solid domains only)
  - NekRS sub-application (fluid domains only)

- **Requirements:**
  - Meshes are generally different
  - Data transfers between the two sub-apps typically are from surface ↔ surface, using a nearest node transfer
  - Data transfers between the main app and the two sub-apps are typically volume ↔ volume

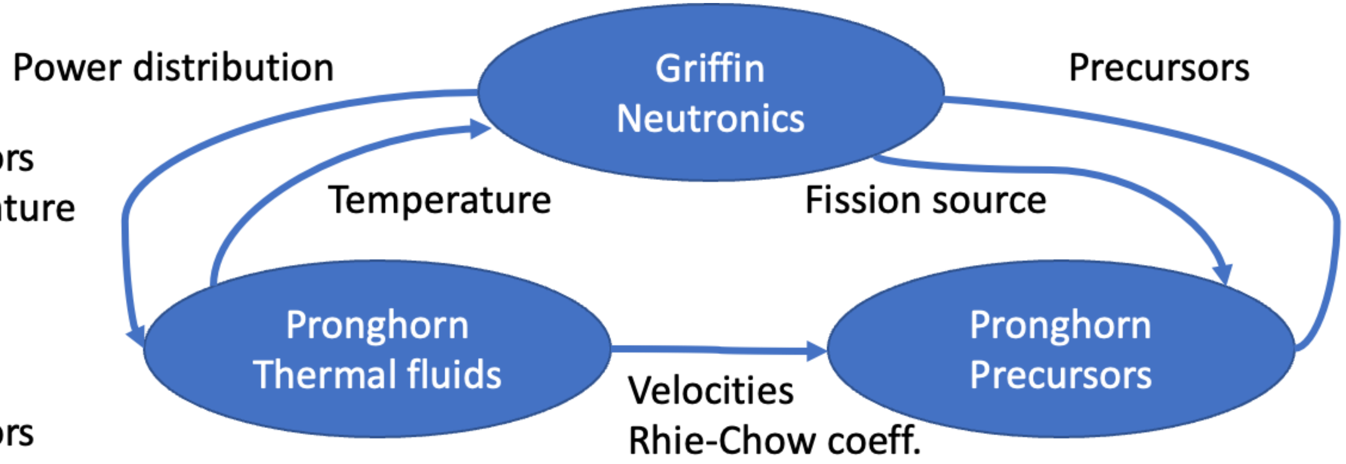


Siblings transfers: example use case 2

Molten Salt Reactor Multiphysics coupling schemes



Current



Future:
Less duplicated quantities

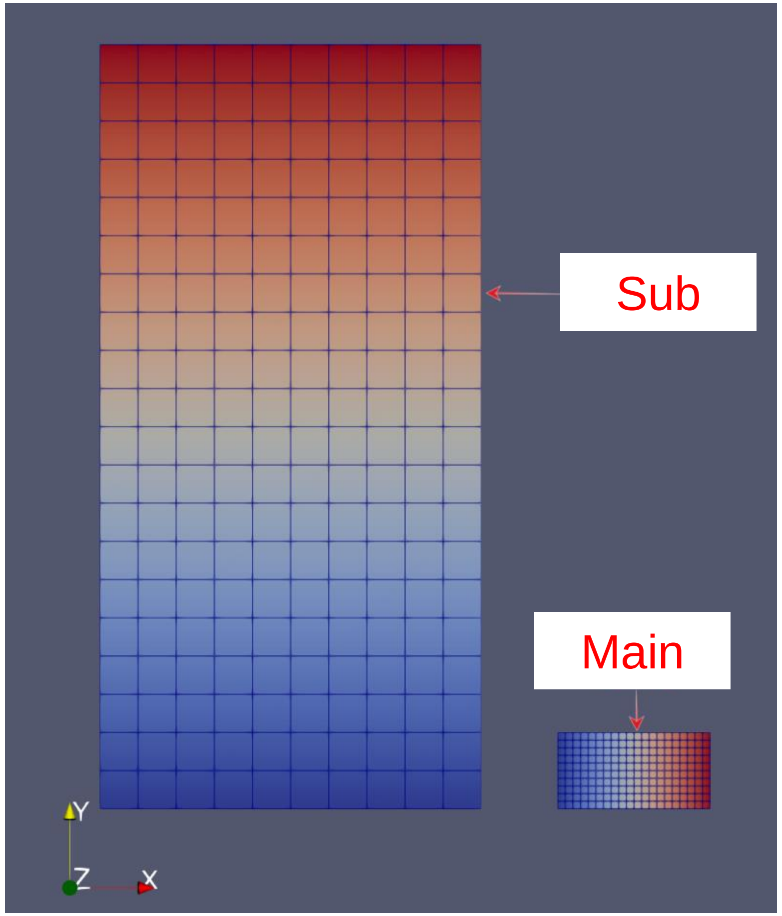
Siblings transfer: current status & future

Current capability allows use in MSFR coupled multiphysics calculations
Previous example requires additional considerations:

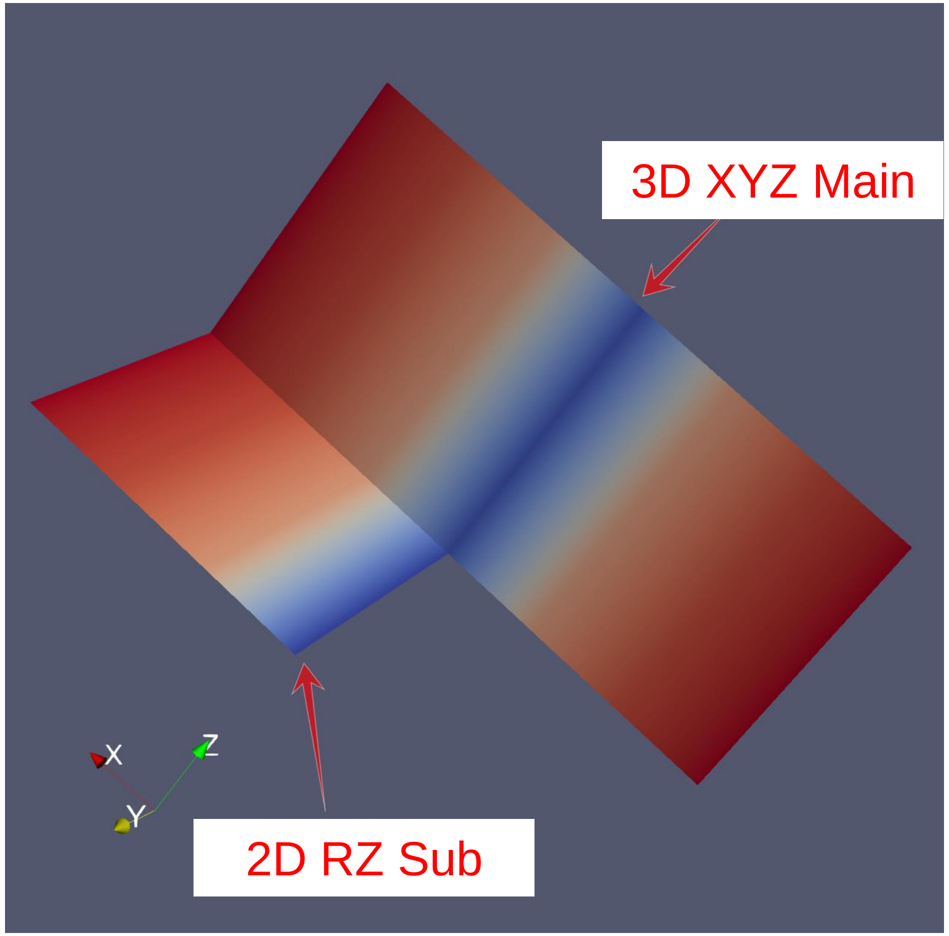
- how to order multiapps and transfers?
Dependency ordering
- transfers between sub-cycles of multiapps, within the main app timestep
Why? Dissimilar time steps:
Neutronics ~ few s
Heat conduction ~ < 1s
CFD ~ 0.1s or less

Transfer	Status
Copy	merged
Nearest-node	Under review
Shape-evaluation	Under review
Postprocessor	merged
Scalar variables	merged
PP <-> scalar	merged
Reporter	merged
User Objects	planned

Coordinate Transformations

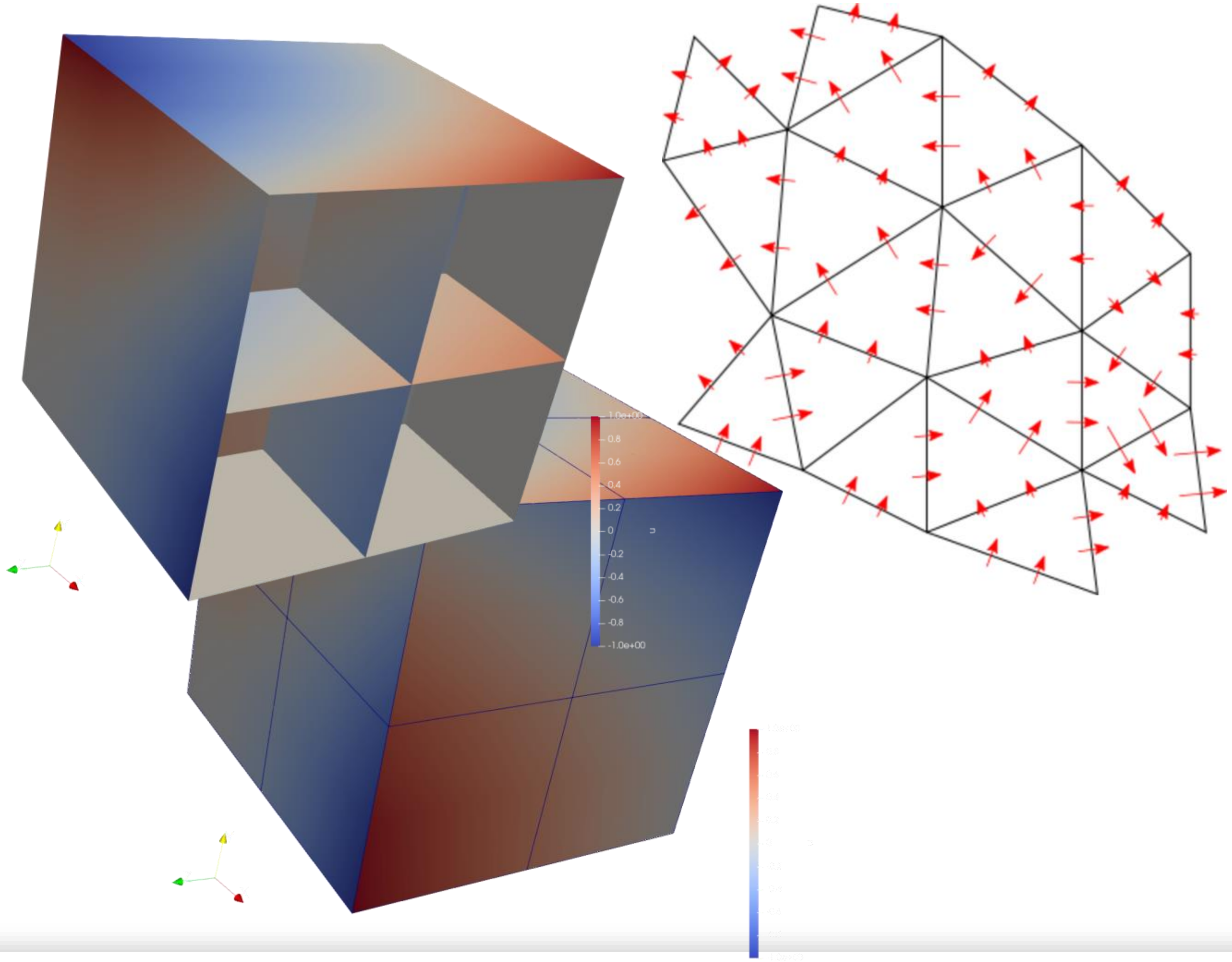


- Implemented generic coordinate transformation capability between applications
- Support
  - Rotations
  - Translation (always supported)
  - Scaling
  - “Coordinate collapsing”, e.g. transform XYZ coordinates into RZ, RZ into RSPHERICAL



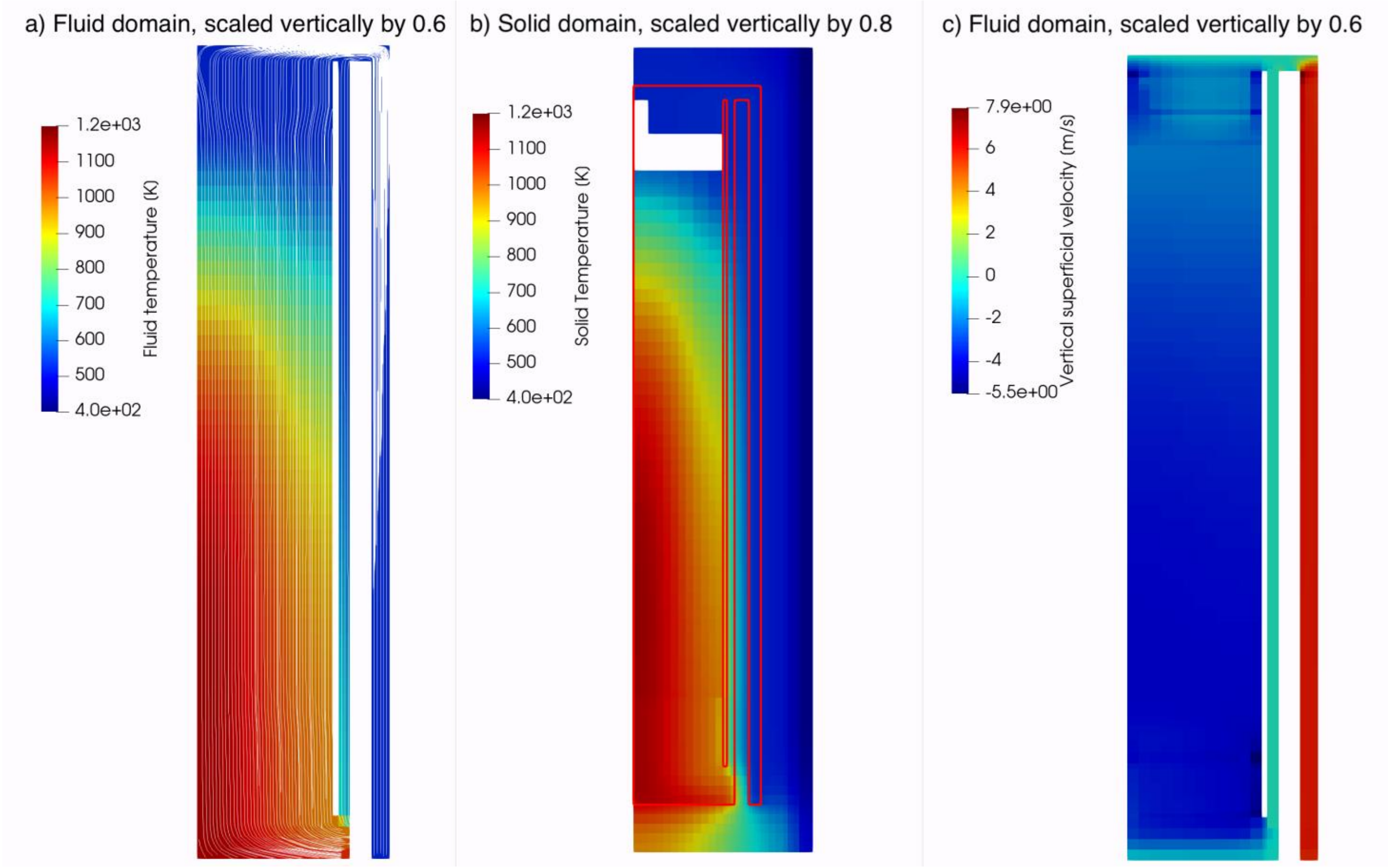
Side-discontinuous Variables

- Smooth on element sides (edges of 2D elements, faces of 3D), equal between neighboring elements, discontinuous at vertices (and in 3D, at edges)
- Originally requested to support arbitrary numeric vector data associated with each side in a mesh
- Designed to support hierarchic polynomial spaces on each face: usable for HFEM, DPG, and other discretizations that require e.g. jump, flux, or normal gradient terms in between elements
- Usable in both auxilliary and nonlinear systems
- Automatically projected in AMR/C, distributed in parallel, etc.
- Visualization via ExodusII output options



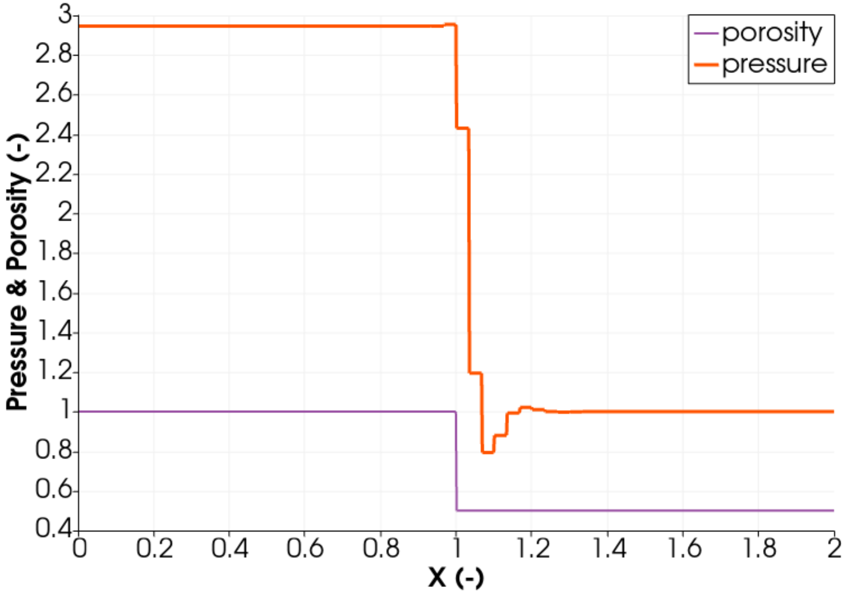
Functor System

- On-the-fly evaluations at arbitrary locations in space and time – no need to pre-init and store (potentially a lot of) data
- MOOSE systems that are functors:
  - Functions
  - Variables
  - Functor material properties
- Necessary for physics that involve large element stencils, e.g. satisfy need to evaluate non-constant density on many different elements when building Rhie-Chow velocity
- Leveraged in weakly-compressible simulation of HTR-PM (steady-state shown at right courtesy Sebastian Schunert)



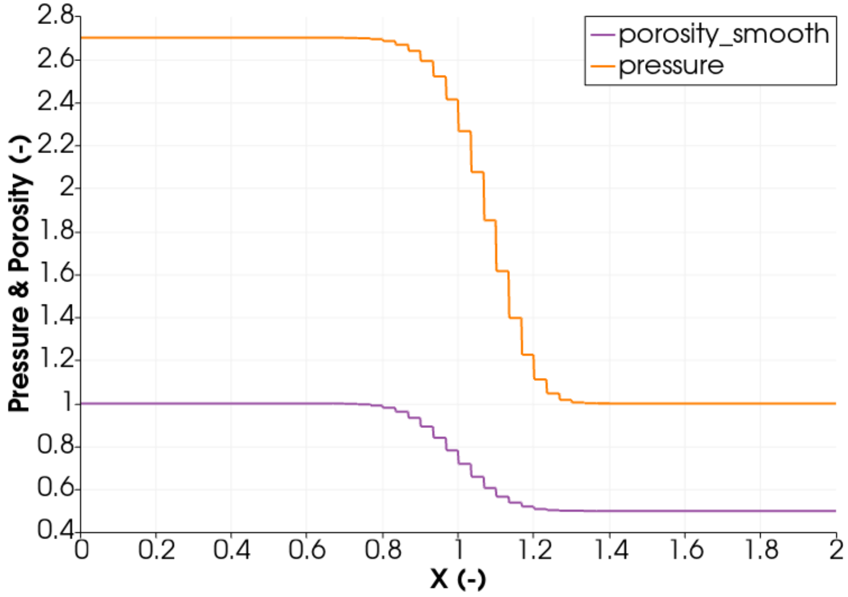
Functor System

- Developers can easily create their own functor classes; well-defined virtual interface
- `CellCenteredMapFunctor` created for Pronghorn
  - Used to hold repeatedly interpolated and reconstructed discontinuous porosity to form a smoothed porosity field with minimal effort from user (single integer parameter)
  - Leads to monotone solutions
  - As verified via MMS, retain second order convergence with respect to element size
  - (planned work in FY23 will lead to monotone solutions without smoothing)



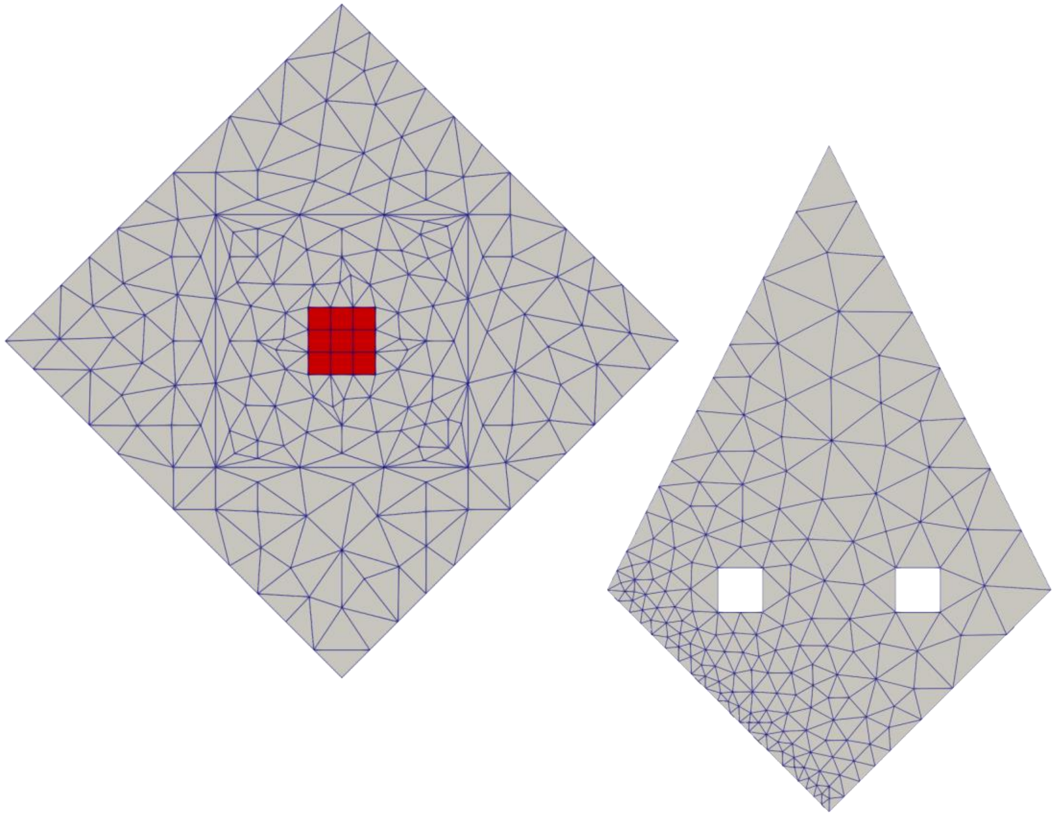
Non-smooth porosity

Smoothed porosity



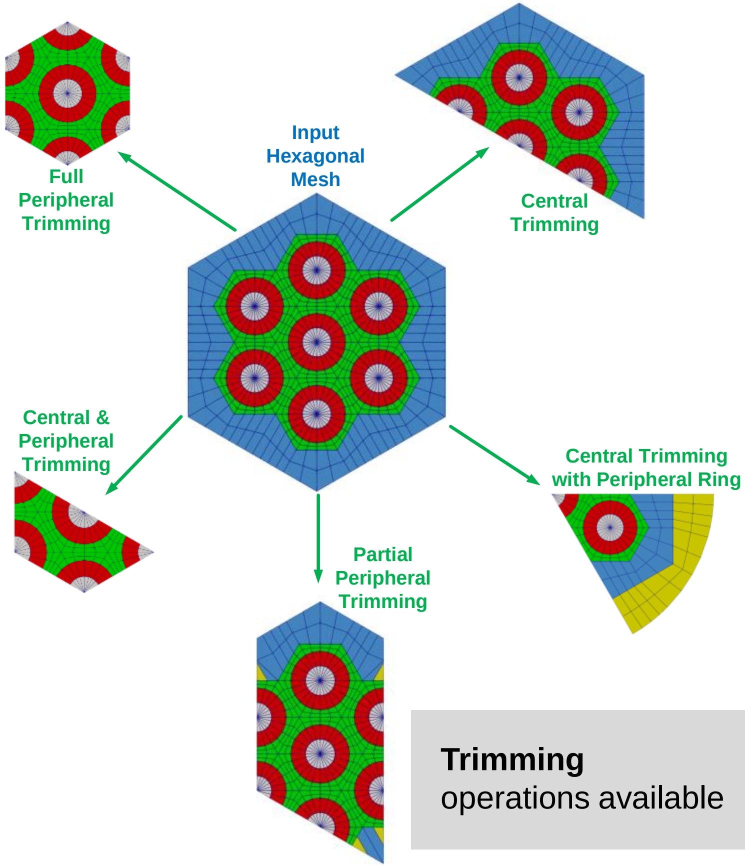
Delaunay Triangulation and Refinement

- XYDelaunay mesh generator, based on Poly2Tri for Delaunay triangulation and new code for refinement
- Inputs a series of boundary/hole meshes (identified by Edge elements or edges of 2D elements), triangulates domain in between
- Refines interior and/or edges based on user options, based on scalar maximum triangle size or spatially-varying size function
- Stitches input "hole" meshes into new triangulation if requested; MOOSE MeshGenerator system enables nested meshes this way

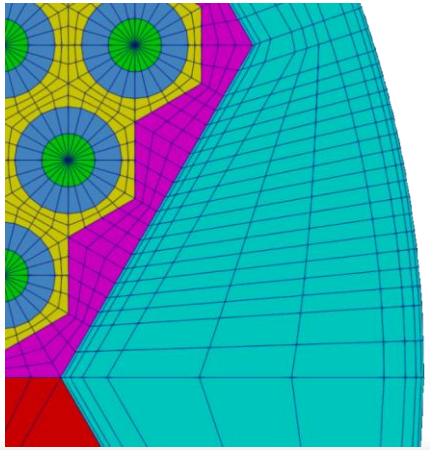


MOOSE Reactor Module:
Open Source, Reactor-Focused Meshing Capabilities

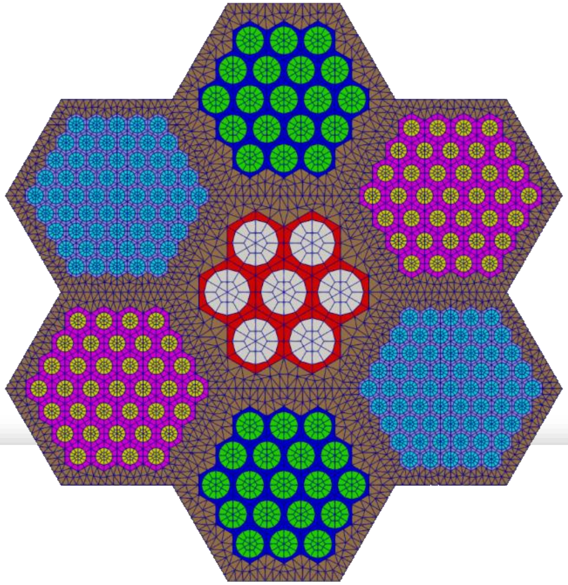
- **MOOSE Reactor Module** is an open-source reactor meshing capability
- Rapidly build up **Hexagonal and Cartesian** geometries with concise input. Can leverage advanced routines for more unique geometries.
- Supports rotating control drums, core periphery zone, **boundary layer meshing**, **mesh biasing**, **mesh trimming**, transition layer meshing, and more
- Supports **labeling of material/depletion/component zones**: **simplifies material mapping and output processing downstream** – user can query labels and integrate solutions over them (*e.g. axial pin power in pin 7 of assembly 3*).



Trimming operations available

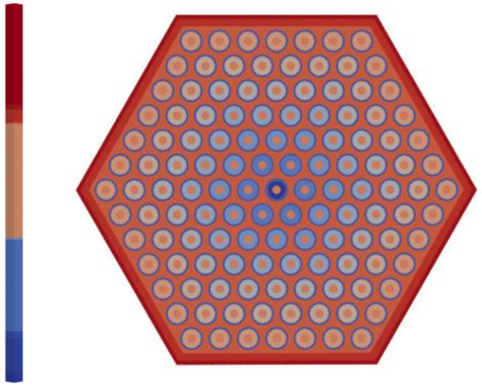


Boundary layer meshing in a core periphery

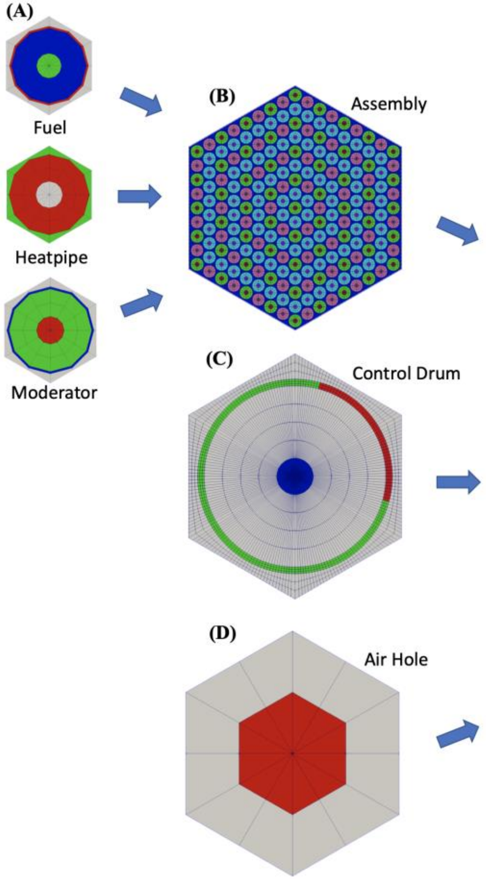


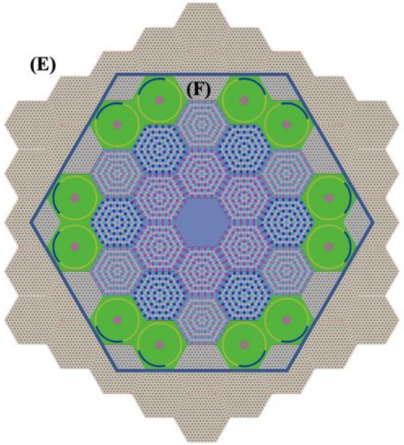
Stitching dissimilar assemblies by applying transition layers to maintain conformality

Application of the Reactor Module for Advanced Reactor Analysis



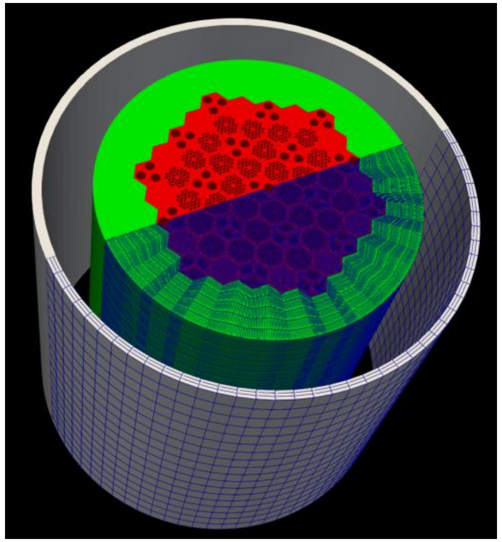
Lead-cooled fast reactor assembly with annular fuel [LFR]

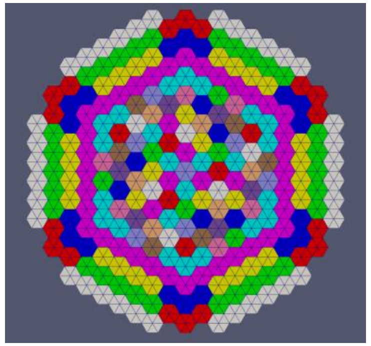


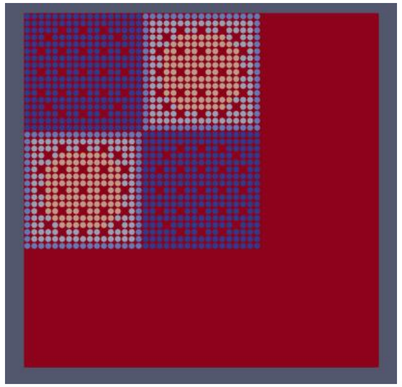


Empire heat-pipe cooled microreactor mesh with rotating control drums [**HP-MR**]

High Temperature Test Reactor mesh [**HTGR**]







Advanced Burner Test Reactor (ABTR) [SFR]

C5G7 Light Water Reactor OECD Benchmark [LWR]

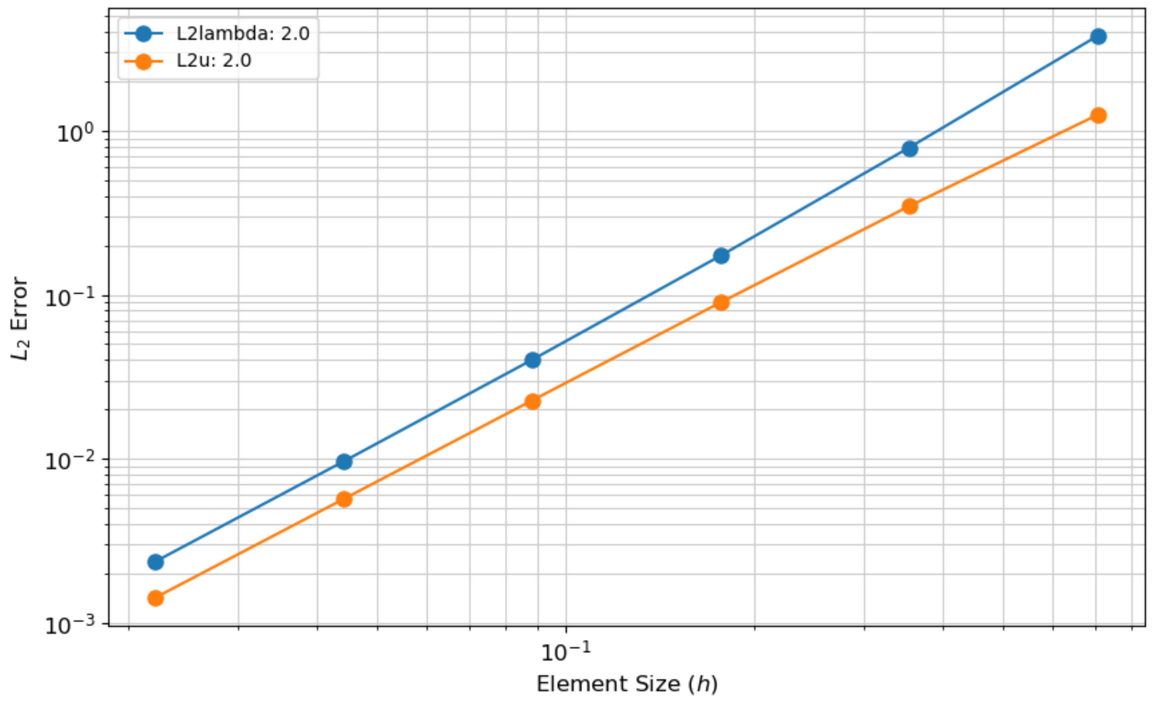
FY23: Tutorial and user training forthcoming, including examples on VTB

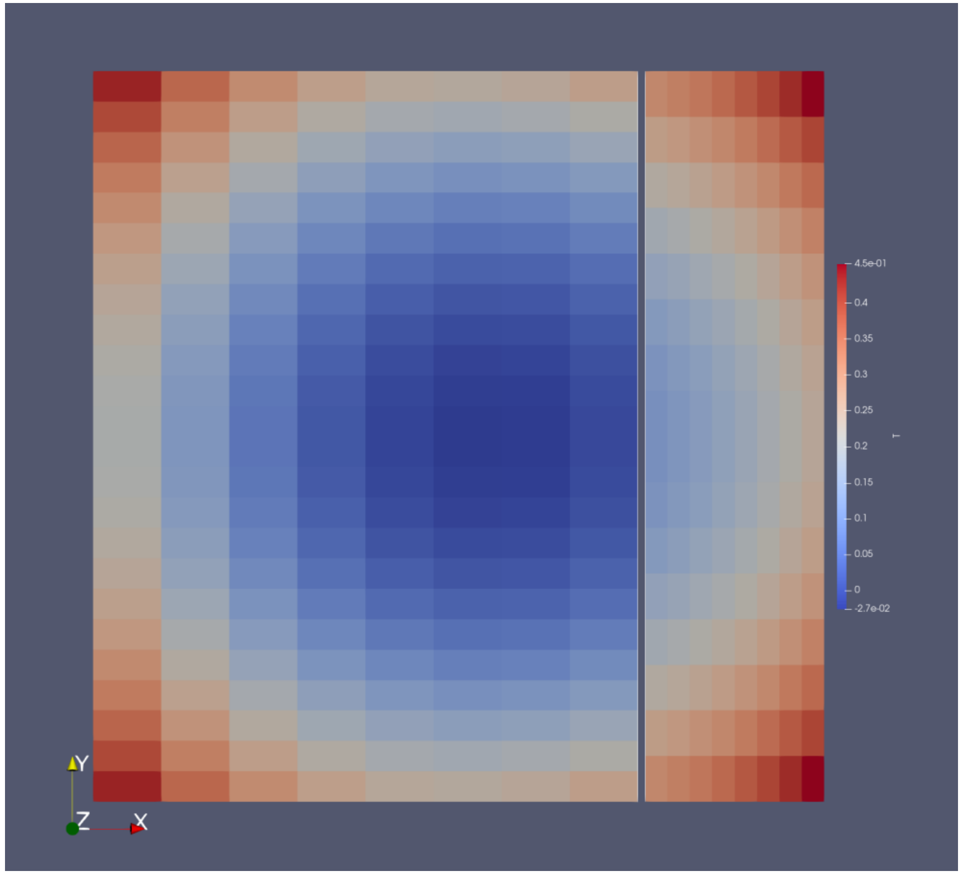
Technical Area General Support

- Moved to depth-first-search algorithm for dependency sorting
  - For THM simulation with many initial conditions, resulted in setup time reduction from ***76 hours to 5 seconds***
- Within-group user object sorting
  - Example: Two elemental user objects used to execute in the order specified in the input file. Now if one depends on another, the dependency will execute first
- Ability to compute residual and Jacobian together
  - Accelerates simulations with heavy residual evaluations (on the order of Jacobian expense). 20% speedup for finite-volume Navier-Stokes implementation

Fluids Support: Gap Heat Transfer via Mortar for Finite Volume

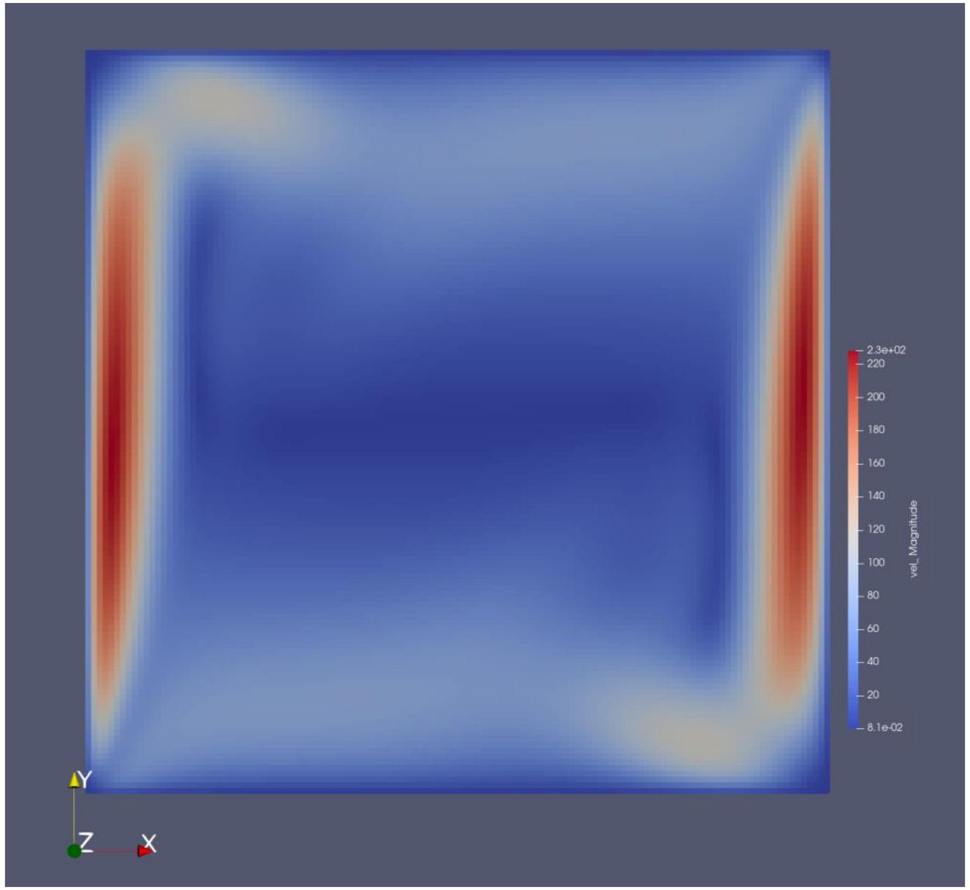
- Initial support for conduction and radiation gap heat transfer models





Fluids: Generalized advection schemes

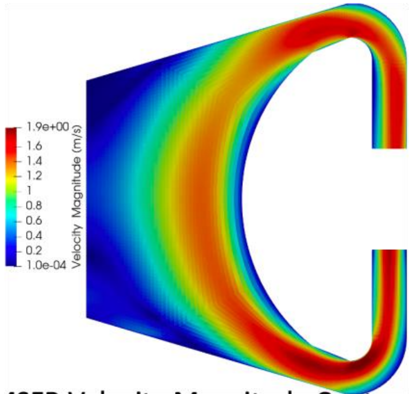
- Added general limiter framework and support for several specific interpolation limiting schemes within incompressible and weakly compressible flow formulations. Current schemes include
  - Minmod
  - Van Leer
  - Second Order Upwind (SOU)
  - Quadratic Upstream Interpolation for Convective Kinematics (QUICK)
- Higher order convergence (in smooth regions) compared to simple upwinding necessary for accurate results for high Rayleigh numbers
- These limiters are applicable to scalar fields, including vector components
- Limiting on a vector basis (adds more stability) planned for FY23



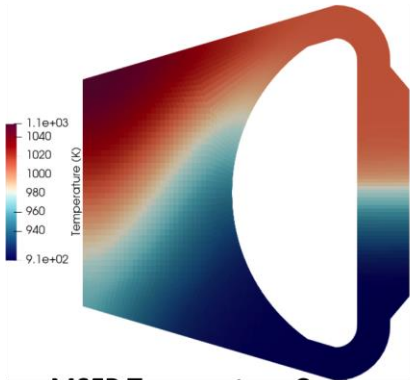
Natural convection, using a minmod interpolation limiter with Rayleigh number = 1e6

Fluids: segregation of scalar transport solve

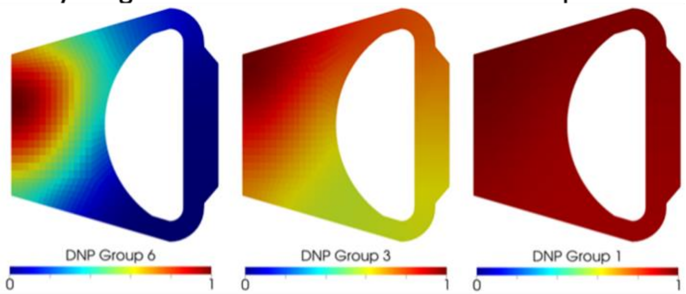
- For molten salt reactors: 6 precursors passively advected
- Up to 11 variables in fully coupled approach
- 2-3x more expensive than fluids only
- Separating 6 precursors makes for a trivial precursors-only solve
- Implemented in Action for easy & short syntax
- Leveraged for demonstrations to Terrapower of NEAMS capabilities
- Necessary for future 3D simulations
- Current: use MultiApps
- Future: use multiple nonlinear systems in same input capability



MSFR Velocity Magnitude Contours



MSFR Temperature Contours



MSFR Delayed Neutron Precursors in different Groupings

Neutronics / Thermomechanics support

- Enabled matrix-only solve type for eigenvalue executioner
  - Savings proportional to number of linear iterations within a nonlinear iteration: has led to 100-1000x speedups in some Griffin simulations
- Added more robust ghosting of lower dimensional variable degrees of freedom
  - Added two new relationship managers, `GhostLowerDElems` and `GhostHigherDLowerDPointNeighbors` to handle lower dimensional variable use cases
- Consumer based reinitialization of material properties for mortar constraints
  - Only compute materials needed by consumers
  - Avoids evaluating materials which depend on stateful properties not computed during mortar

Look ahead to FY23

- Scalar transport module merged – Hopefully common ground for code modeling scalar transport, e.g. tritium
- Multiple nonlinear systems on the same mesh merged
  - With Executor system allows arbitrary nonlinear solution techniques
  - Picard in one input, SIMPLE in one input, etc.
- Dynamic linking and loading of applications
- Enhance initial condition system to support more basis functions (e.g. Nedelec for electromagnetics)
- Automatic differentiation at the libMesh Node level
  - Allows propagation of displaced mesh dependence through all geometric calculations
  - Potential for more robust mechanics solves, particularly those involving contact
- Continued enhancement of checkpoint restart capabilities
  - Example: support restarting a transient simulation from previous eigenvalue calculation
- More transfers supported on distributed mesh (e.g. user object transfers)
- Dependency resolution for sibling transfers
- Stateful material properties with distributed mesh and adaptivity

U.S. DEPARTMENT OF ENERGY | Office of NUCLEAR ENERGY

Questions?

U.S. DEPARTMENT OF ENERGY

Office of NUCLEAR ENERGY

Flexible Executioners - Executor System

- “Execution” plan at the heart of every MOOSE simulation
- Some potential stages include
  - Mesh adaptivity
  - Time step selection
  - MultiApp execution
  - Nonlinear Solve A
  - Nonlinear Solve B (upcoming in FY23)
- Original simple executioners (Steady, Transient) ill-suited to handle complex execution sequence
- SolveObjects, added in last handful of years, also insufficient
- Experience and development has culminated in Executor system
- Complete replacement of Executioners
- Executor examples: `AdaptivityExecutor`, `FEProblemSolveExecutor`, `TransientExecutor`
- Enables highly-arbitrary user execution schemes
  - Necessary for new fluid solvers emulating canonical CFD algorithms like SIMPLE, PIMPLE, etc.