SAND2021-4558C

Sandia National Laboratories

Exceptional service in the national interest

FLEXO: Development of a Portably Performant AMR Code for XMHD

Brian N. Granzow, Stephen D. Bond, Nathaniel D. Hamlin, Matthew M. Martin, Alan K. Stagg, Thomas E. Voth

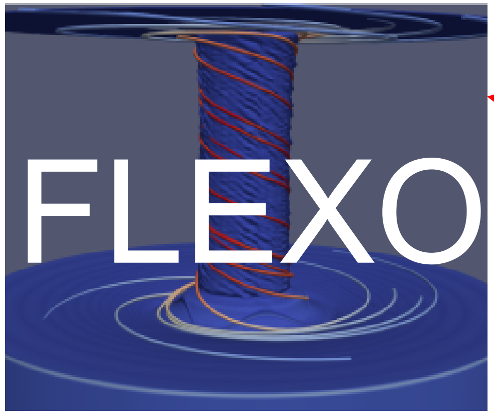
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

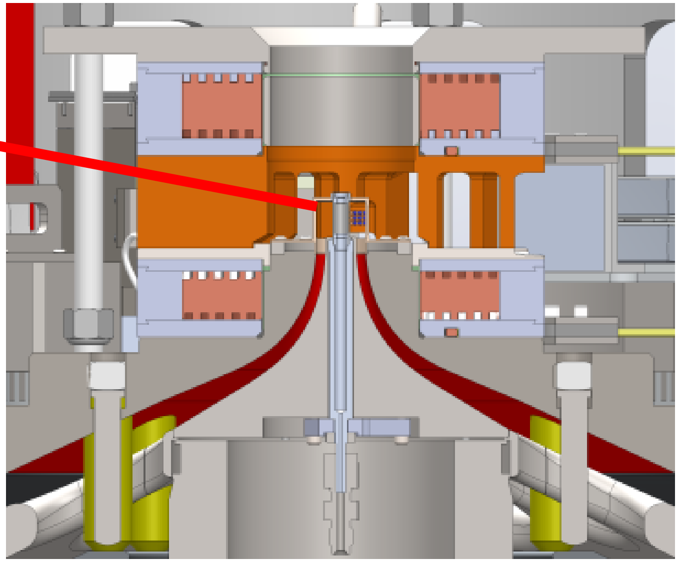
U.S. DEPARTMENT OF ENERGY

NNSA

Background and Motivation

- PERSEUS
  - Design code for experiments to investigate the origins of helical instabilities in MagLIF
  - Validation tests on Z experimental data bringing new insights into previously unexplained data
  - Results indicate the importance of Hall physics in target applications





- FLEXO
  - Next-generation of PERSEUS with expanded feature set for target design
  - Improve performance: NGP from inception
  - Improve fidelity: AMR to capture fine-scale features
  - Additional physics: multi-material to capture containment effects
  - Additional physics: tabular equations of state and conductivities for predictive material response

Governing Equations + Time Advancement Strategy

- Governing Equations

$$\partial_t[\rho] + \nabla \cdot [\rho \boldsymbol{v}] = 0,$$

$$\partial_t[\rho \boldsymbol{v}] + \nabla \cdot [\rho \boldsymbol{v}\boldsymbol{v} + p\boldsymbol{I}] = \boldsymbol{J} \times \boldsymbol{B},$$

$$\partial_t[\mathcal{E}] + \nabla \cdot [\boldsymbol{v}(\mathcal{E} + p)] = \boldsymbol{v} \cdot (\boldsymbol{J} \times \boldsymbol{B}) + \eta \boldsymbol{J}^2,$$

$$\partial_t[\boldsymbol{B}] + \nabla \times \boldsymbol{E} = \boldsymbol{0},$$

$$\partial_t[\boldsymbol{E}] - c^2 \nabla \times \boldsymbol{B} = -c^2 \mu_0 \boldsymbol{J},$$

Hall term

$$\partial_t[\boldsymbol{J}] = \frac{n_e e^2}{m_e}\left(\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B} - \frac{1}{n_e e}\boldsymbol{J} \times \boldsymbol{B} - \eta \boldsymbol{J}\right).$$

- Discretization Strategy
  - Space: DG P0/P1/P2
  - Time:
    - Fields explicitly advanced to intermediate stage (*), where all source terms except energy are neglected.
    - Implicit (element-local) correction:

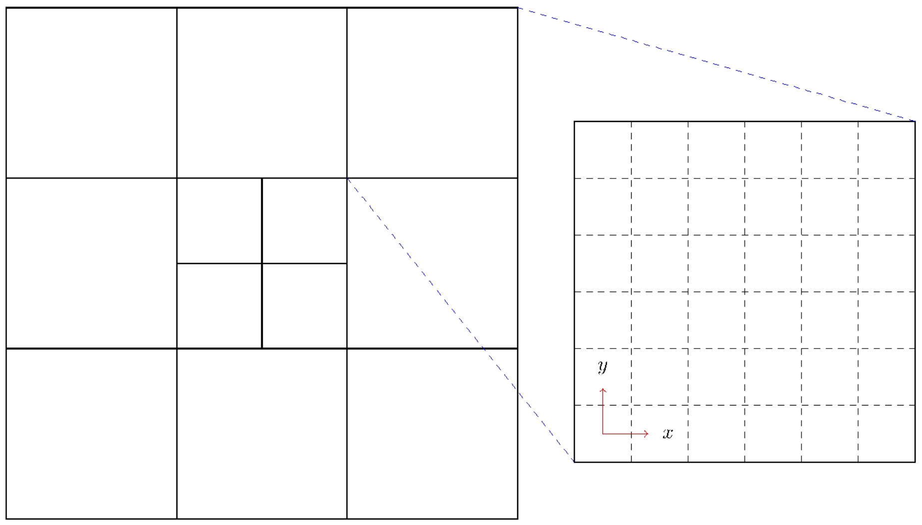
$$\boldsymbol{E}^{n+1} = \boldsymbol{E}^* - \Delta t\, c^2 \mu_0\, \boldsymbol{J}^{n+1},$$

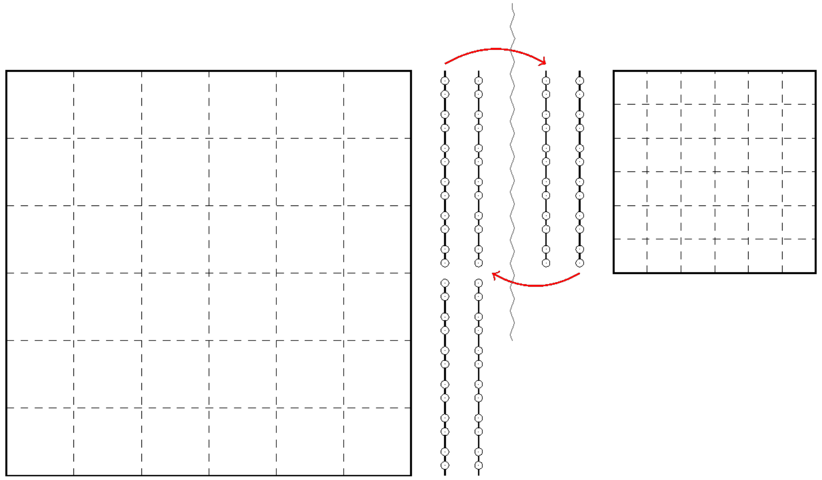
$$[\rho \boldsymbol{v}]^{n+1} = [\rho \boldsymbol{v}]^* + \Delta t[\boldsymbol{J} \times \boldsymbol{B}],$$

$$\boldsymbol{J}^{n+1} = \boldsymbol{J}^* + \Delta t \left(\frac{n_e e^2}{m_e}\right)\Big[\boldsymbol{E}^{n+1} + \boldsymbol{v}^* \times \boldsymbol{B}^* - \frac{1}{n_e e}\boldsymbol{J}^{n+1} \times \boldsymbol{B}^* - \eta^* \boldsymbol{J}^{n+1}\Big].$$

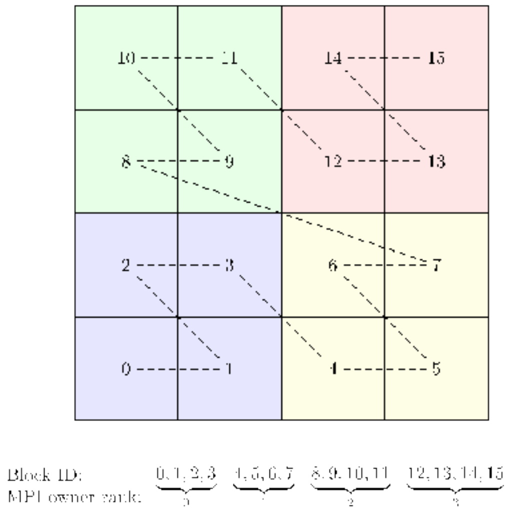
For details see: Seyler, Martin, Physics of Plasmas **18**, 012703 (2011)

DEVELOPMENT EFFORT: AMR Framework



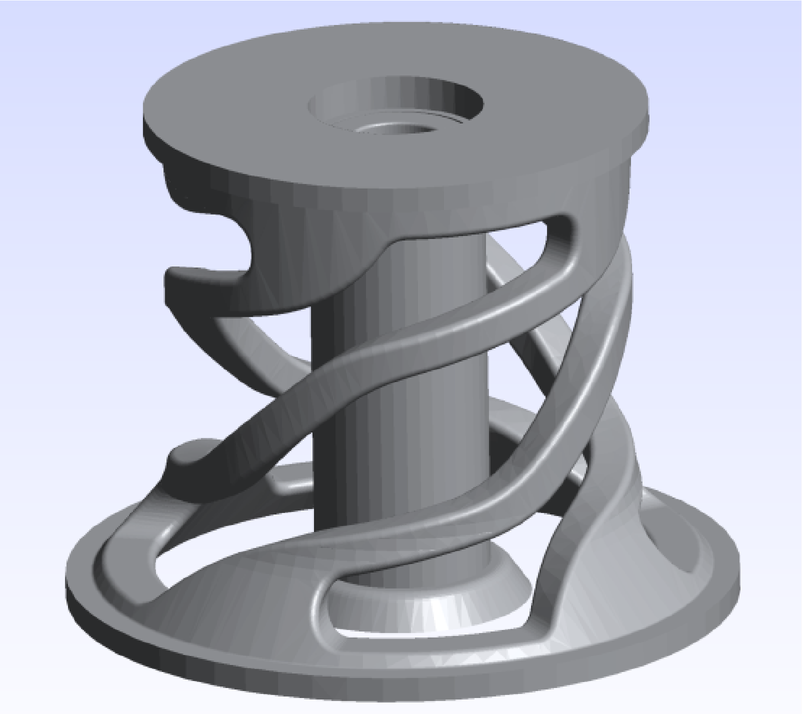


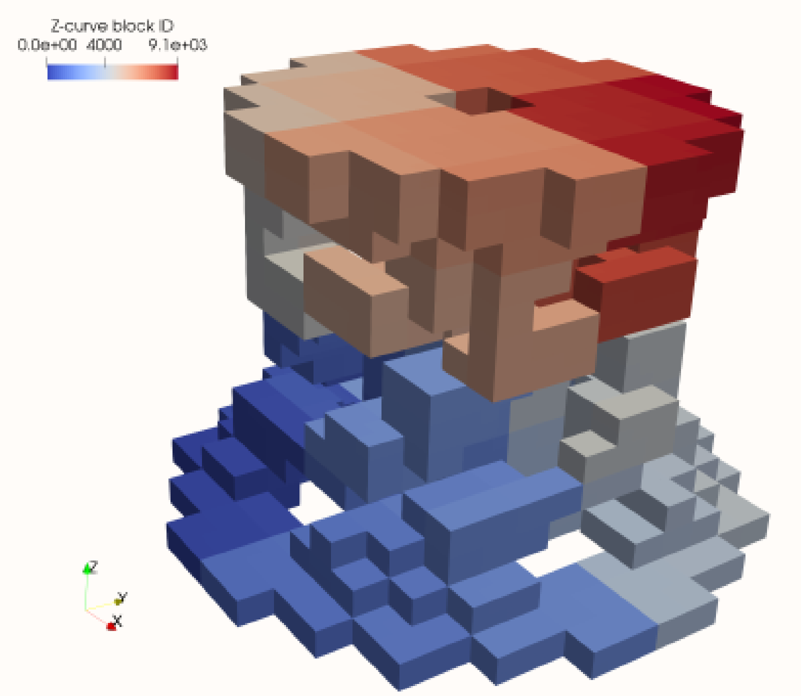
- Octree mesh data structure
  - MPI parallelized leaves
  - Each leaf is called a “block”
  - Each block represents a structured (i,j,k) Cartesian grid
  - Physics is evaluated over each block
  - Fluxes across block boundaries couple block physics

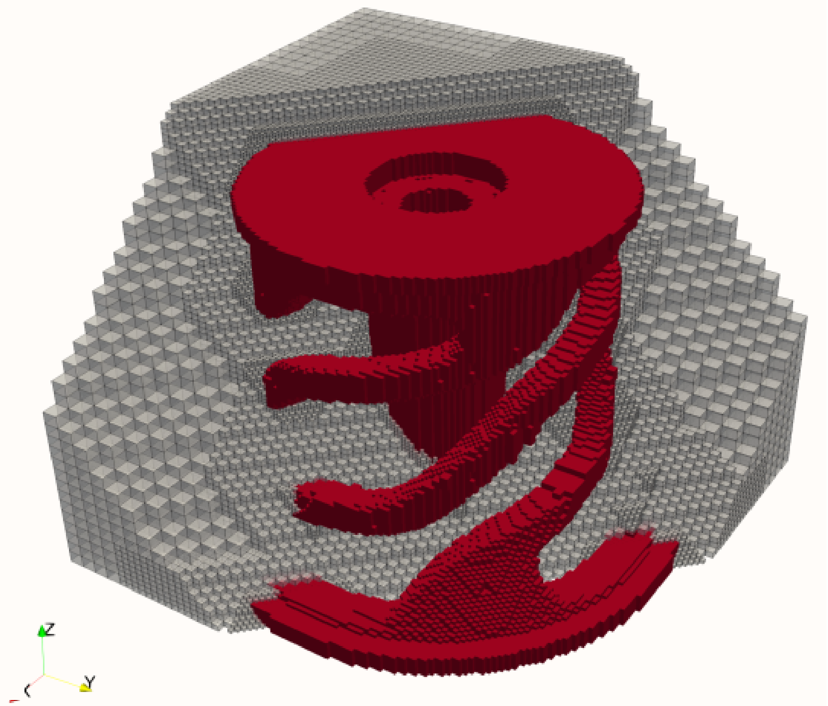


  - Shared memory parallelism -> kernels over block entities (cells, sides)
  - MPI parallelism -> solution information transferred at block borders, Z-curve block partitioning
  - Block adjacencies restricted to at most one finer or coarse level (2-1 interfaces)

AMR Framework Mesh Example







STL "CAD" geometry → Octree refined to STL geometry definition → Material initialization in refined octree mesh

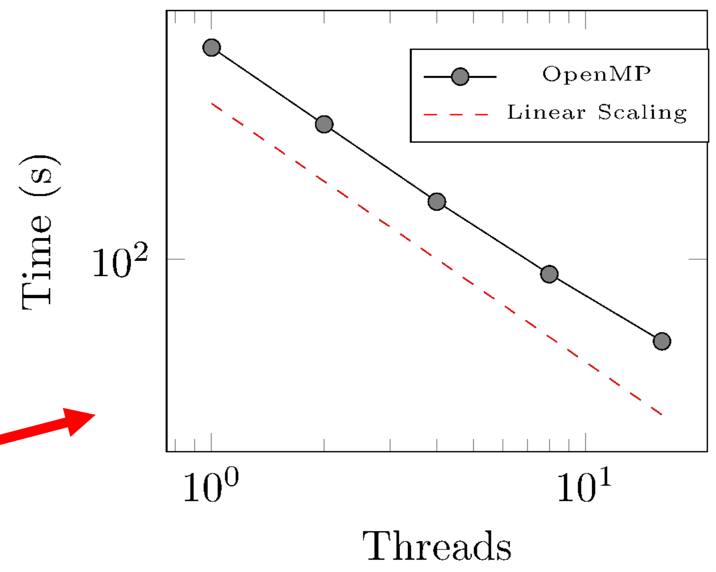
Illustrates the difference between "block-based" and "cell-based" AMR

DEVELOPMENT EFFORT: NGP Execution

- Shared-memory parallelism occurs over block entities (cells, sides)
  - A parallel-for construct (flexo::for_each) calls a functor that defines physics behavior
  - flexo::for_each
    - wrapper around Kokkos::parallel_for to parallelize over (i,j,k) grid indices
    - made possible with the use of Kokkos::MDRangePolicy
  - All functors run in a single execution space (GPU, OpenMP, etc…) depending on compilation
  - However, Kokkos makes mixing + matching execution spaces possible

```
grid3 const cell_grid = block.cell_grid();
auto f = KOKKOS_INLINE_FUNCTION (vector3<int> const& cell_loc) {
  int const cell = cell_grid.index(cell_loc);
  for (int basis = 0; basis < num_basis(polynomial_order); ++basis) {
    residual(cell, RHO, basis) += 1.0;
  }
};
flexo::for_each(cell_grid, f);
```

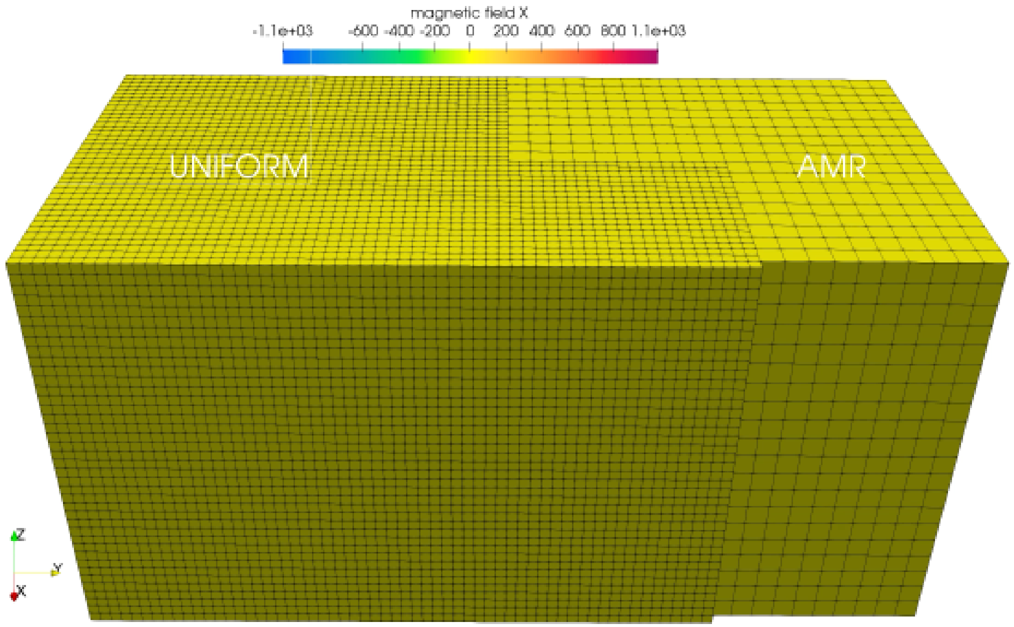
Timings for wire example (100x100x100 cells)

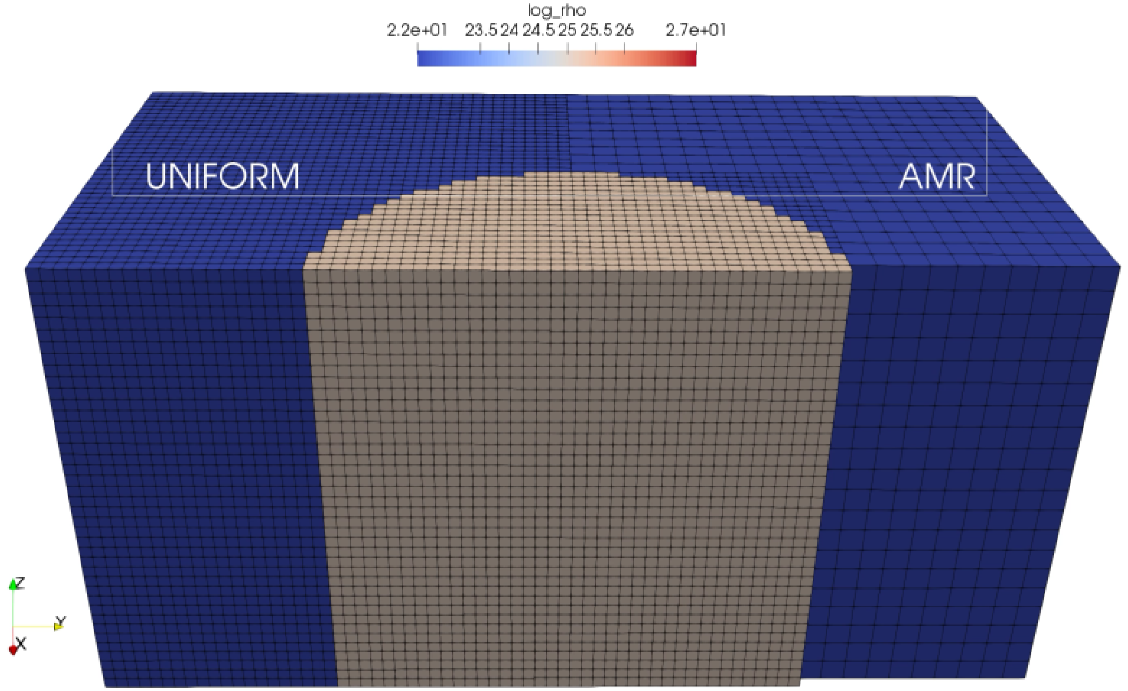


Nearly ideal shared-memory scaling for ENTIRE simulations, not just a single functor

NGP Execution Example

- Magnetically driven implosion
  - 80x80x80 cells in uniform grid cases
  - Cylinder with sinusoidally perturbed surface
  - Poynting inflow, momentum outflow





SMR (static mesh refinement) results agree nicely to uniform grid results

Table 1: Cylindrical Implosion Timings (80x80x80 cells, 10 steps)

PERSEUS	FLEXO
Intel(R) Xeon(R) CPU E7-4880	Tesla V100-SXM2-16GB
343.935s	2.177s

158 times faster!

DEVELOPMENT EFFORT: Multi-Material Extensions

- Governing equations
  - Per-material evolution equations: volume fraction, mass conservation, energy
  - Single mixture velocity, and thus momentum
  - Non-conservative source terms handled through Riemann solver * quantities

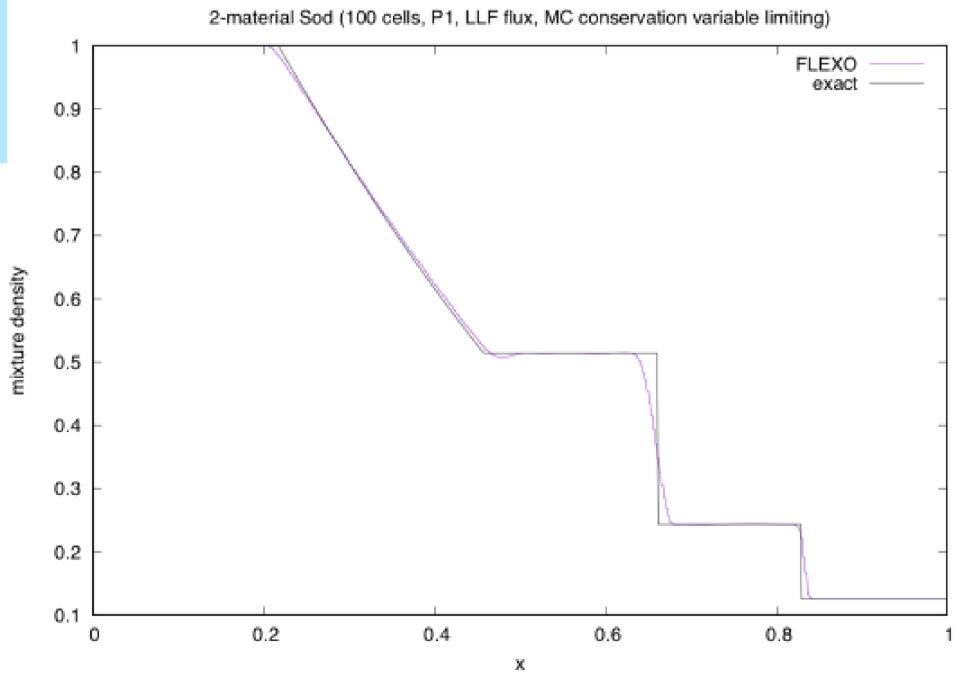
$$\partial_t[\alpha_k] + \boldsymbol{v} \cdot \nabla \alpha_k = 0,$$

$$\partial_t[\rho_k] + \nabla \cdot [\rho_k \boldsymbol{v}] = 0,$$

$$\partial_t[\bar{\rho}\boldsymbol{v}] + \nabla \cdot [\bar{\rho}\boldsymbol{v}\boldsymbol{v} + \bar{p}\boldsymbol{I}] = \boldsymbol{J} \times \boldsymbol{B}$$

$$\partial_t[\rho_k \mathcal{E}_k] + \nabla \cdot [\boldsymbol{v}(\rho_k \mathcal{E}_k + \alpha_k p_k)] = \boldsymbol{v}^* \cdot \boldsymbol{\nabla}[\alpha_k p_k]^* - \frac{\rho_k}{\bar{\rho}} \boldsymbol{v}^* \cdot \nabla[\bar{p}]^*$$

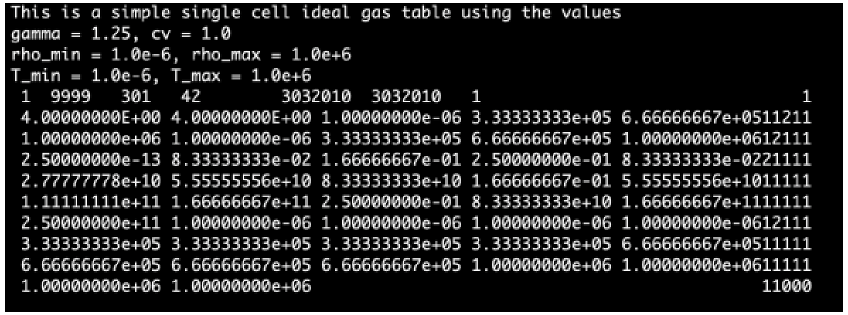
$$\rho_k := \alpha_k \tilde{\rho}_k, \qquad \bar{\rho} := \sum_{k=1}^{n_{\mathrm{mat}}} \rho_k, \qquad \bar{p} := \sum_{k=1}^{n_{\mathrm{mat}}} \alpha_k p_k.$$



For details see: Pandare et al., DOI: 10.1002/fld.4810 (2019)

DEVELOPMENT EFFORT: Tabular Equations of State

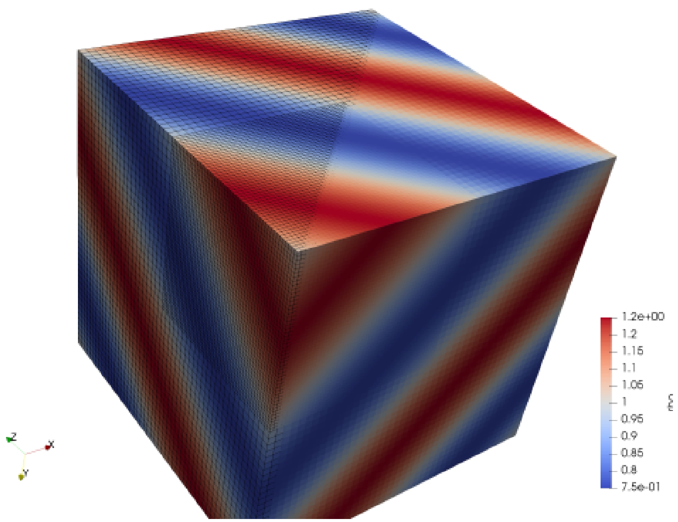
- Use tabular EOS’s (SESAME) for more predictive material response
- Requires modernization of tabular read / interpolation routines for NGP execution
  - SESAME table data must be moved to device memory for GPU execution
  - Interpolation / reverse interpolation methods have been re-written to execute on device
- Have an initial implementation of SESAME EOS tables for hydrodynamic variables
  - Higher-order interpolation / reverse interpolation routines
  - Will allow novel performance assessments for tabular EOS evaluations in NGP settings
  - Ongoing collaboration with EOS + plasma physics experts to improve:
    - tabular interpolation
    - response at density / temperature floors
- Mock ideal gas SESAME table (301) part of test harness
- TODO: conductivity + ionization state from SESAME tables (601)

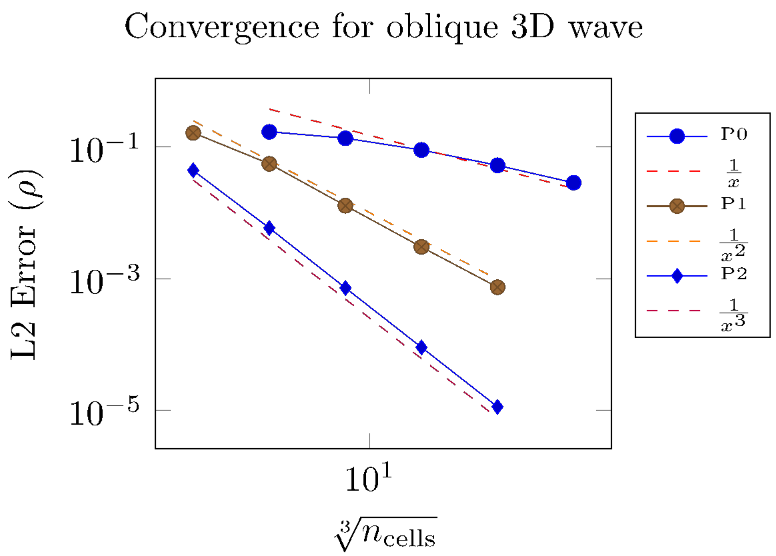


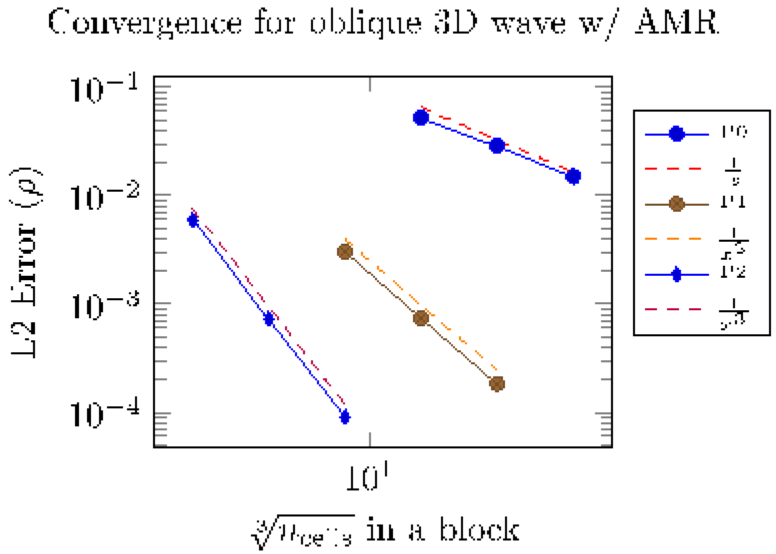
```
This is a simple single cell ideal gas table using the values
gamma = 1.25, cv = 1.0
rho_min = 1.0e-6, rho_max = 1.0e+6
T_min = 1.0e-6, T_max = 1.0e+6
 1  9999   301   42        3032010  3032010   1                    1
 4.00000000E+00 4.00000000E+00 1.00000000e-06 3.33333333e+05 6.66666667e+0511211
 1.00000000e+06 1.00000000e-06 3.33333333e+05 6.66666667e+05 1.00000000e+0612111
 2.50000000e-13 8.33333333e-02 1.66666667e-01 2.50000000e-01 8.33333333e-0221111
 2.77777778e+10 5.55555556e+10 8.33333333e+10 1.66666667e-01 5.55555556e+1011111
 1.11111111e+11 1.66666667e+11 2.50000000e-01 8.33333333e+10 1.66666667e+1111111
 2.50000000e+11 1.00000000e-06 1.00000000e-06 1.00000000e-06 1.00000000e-0612111
 3.33333333e+05 3.33333333e+05 3.33333333e+05 3.33333333e+05 6.66666667e+0511111
 6.66666667e+05 6.66666667e+05 6.66666667e+05 1.00000000e+06 1.00000000e+0611111
 1.00000000e+06 1.00000000e+06                                          11000
```

DEVELOPMENT EFFORT: Verification & Validation

- Desire for a strong V&V culture in development cycle
- TDD/integration in accordance with modern practices
- Broad computational physics space must be verified
  - DG -> P0/P1/P2 cells
  - Pure hydrodynamics
  - Pure electromagnetics (EM)
  - Coupled XMHD physics
  - AMR interfaces
  - Execution spaces (GPU/CPU)
- Validation efforts *forthcoming*
  - Match Z experimental results
  - Helical instability with XMHD
  - Integrate PERSEUS validation tests into development cycle







SUMMARY

- FLEXO is next-generation production effort based on PERSEUS
- Development efforts in FLEXO:
  - “Productionizing” current PERSEUS XMHD algorithms
  - Developing and demonstrating a portably performant (MPI + X) AMR framework
  - Including tabular equations of state for material response
  - Extending single material XMHD formulation to a 6-equation multi-material model
  - Rigorous verification & validation tests closely integrated with the development cycle using modern software tools and practices
- Some future work
  - Fully time-transient AMR (coarsening / refining blocks as the simulation evolves)
  - Demonstrate validity of multi-material XMHD via comparisons to Z experimental data
  - Right on the cusp of providing features for user-friendliness + transitioning to an eager user base