Generalizing 2D Adaptation for M3D-C1

by

Brendan C. Lyons[1], N.M. Ferraro[2]

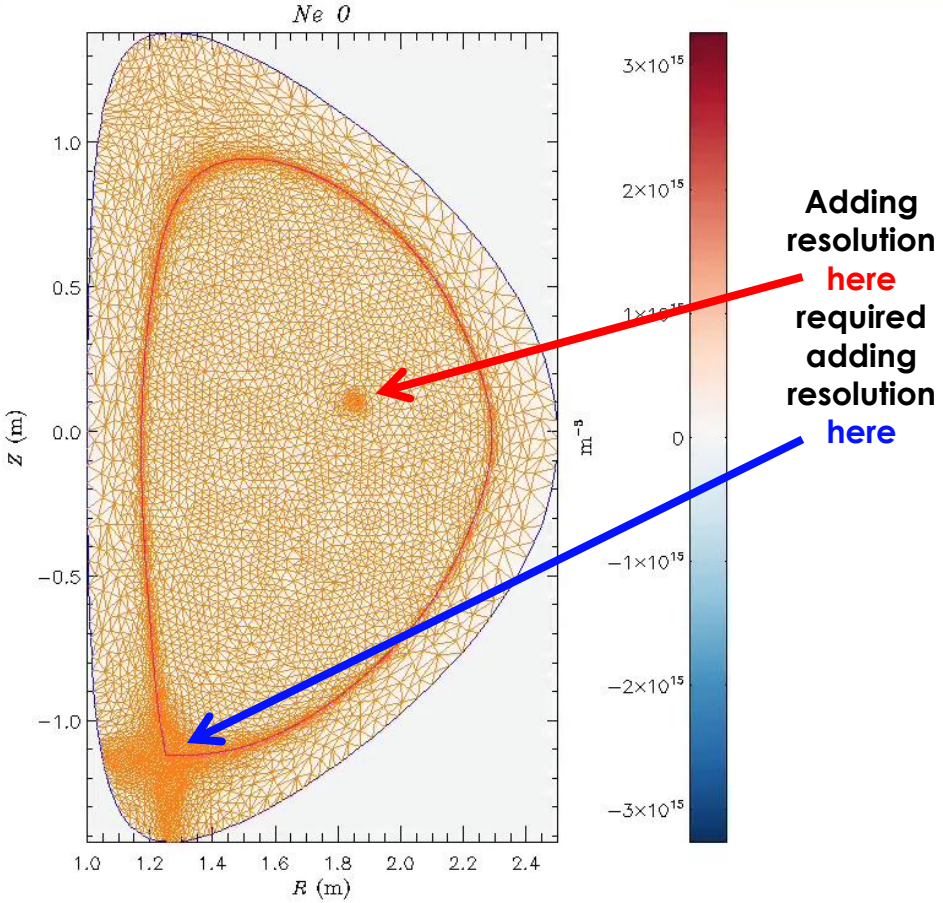
[1]General Atomics
[2]Princeton Plasma Physics Laboratory

August 11th, 2020

GENERAL ATOMICS

Newer Use Cases would Benefit from Generalized Adaptation

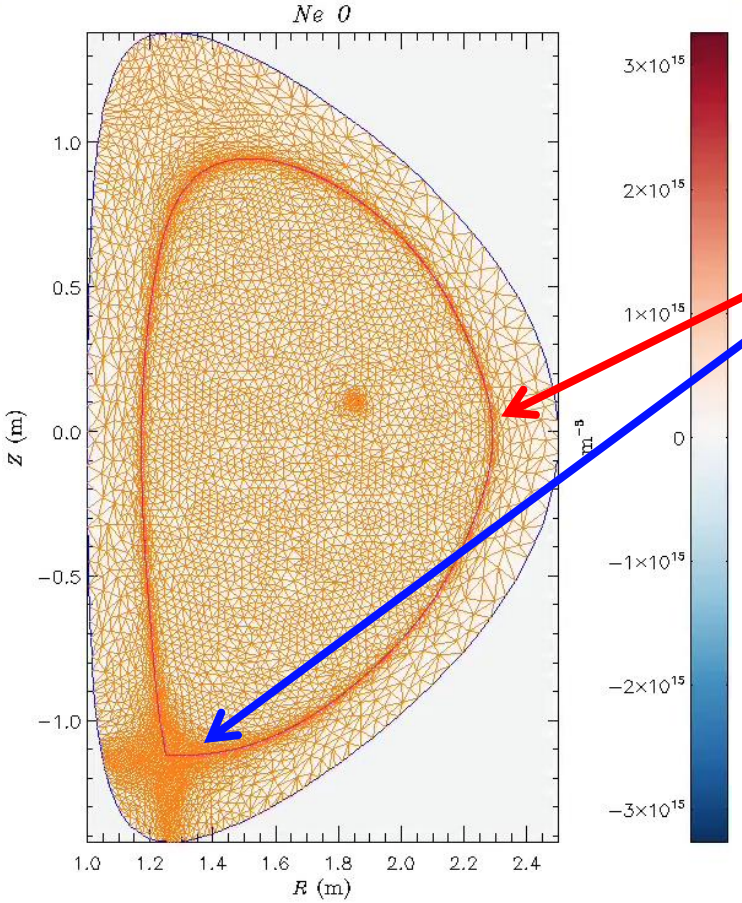
- **Currently, M3D-C1 has two adaptive meshing options available through the SCOREC libraries**
  - Adapt to Ψ_N
  - Adapt to error
- **We've also hijacked the Ψ_N adaptation to allow us to adapt to coil and pellet locations, by setting Ψ_N to 1 in those areas**
- **Generalized adaptation not linked to Ψ_N would be useful**



GENERAL ATOMICS

Proposal: M3D-C1 Defines the Sizes and Unit Vectors Directly

- **M3D-C1 defines anisotropic sizes**
  - Unit vectors for each dimension
  - Need to know how SCOREC sizes translate to real space
- **Permits many options on our end**
  - Define very fine adaptation in small area (removes linkage on previous slide)
  - Gradient scale lengths (of pressure, temperature, current) could be used to define element size, instead of implicitly through Ψ_N
  - Could also more easily resolve resistive wall, rational surfaces, etc.



Real-space gradients stronger here than here, since Ψ_N expands, but elements currently required to be same size

GENERAL ATOMICS

Proposed APIs

API #1

- **M3D-C1 sets size information for nodes or elements directly, whatever SCOREC uses**
- `call m3dc1_set_adapt_size(inode, size_x, size_y, unit_x, unit_y)`
- **Following** `m3dc1_scorec/src/m3dc1_sizeField.cc`
  - `h[0] = size_x` and `h[1] = size_y`
  - `R[0][0] = unit_x[0], R[1][0] = unit_x[1], R[0][1] = unit_y[0], R[1][1] = unit_y[1]`
- **Does SCOREC need a way of defining sizes on new nodes as they're created, or is that taken from the original nodes?**

API #2

- **M3D-C1 defines fields for size and unit vectors, which get evaluated by SCOREC**
- `call m3dc1_set_adapt_size(size_x, size_y, unit_xr, unit_xz, unit_yr, unit_yz)`
- **SCORE evaluates the** `size_` **and** `unit_` **fields and assigns them to** `h` **and** `R`

GENERAL ATOMICS