Modeling high temperature superconducting magnetics with an open source finite element library

S. Shiraiwa, J. C. Wright, D. White[1], M. Stowell[1], and T. Kolev[1]
PSFC, MIT, LLNL[1]

6th International Workshop on Numerical Modeling of High Temperature Super Conductors
26-29 June 2018, Caparica-Portugal

PSFC

COMMONWEALTH FUSION SYSTEMS

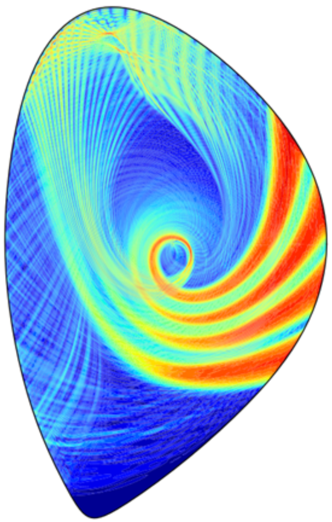
Commonwealth Fusion System (CFS)

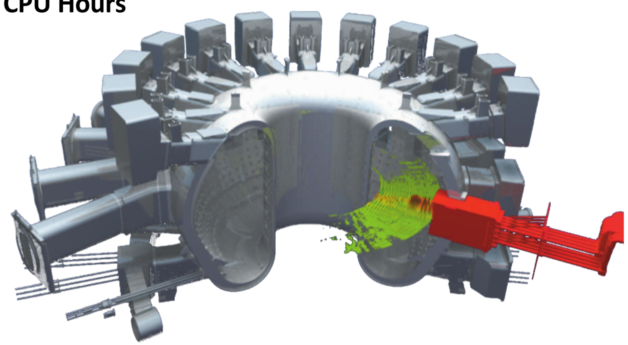
- CFS is an MIT spinout with the goal of developing high-field HTS magnets for fusion energy and other applications.
- CFS is sponsoring research at MIT to develop the magnet technology.
- Part of this work includes developing numerical models to predict magnet performance, which will be validated against small-scale test magnets.
- Many parallels between simulating magnetized plasmas and superconducting magnets.

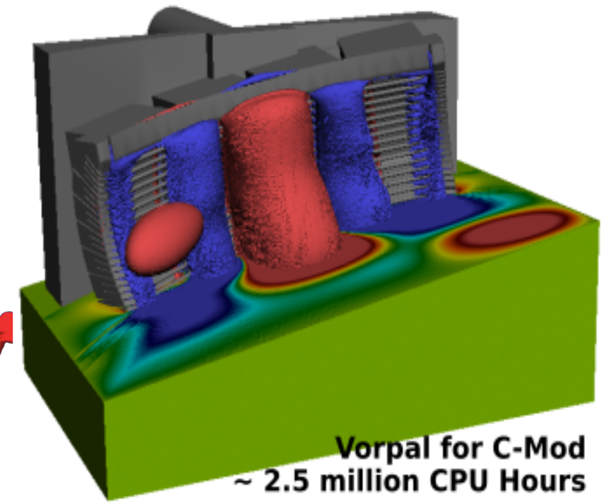
PSFC

COMMONWEALTH
FUSION
SYSTEMS

Leading-class computing facilities allow for accurate RF wave physics simulations in core and edge regions with great detail

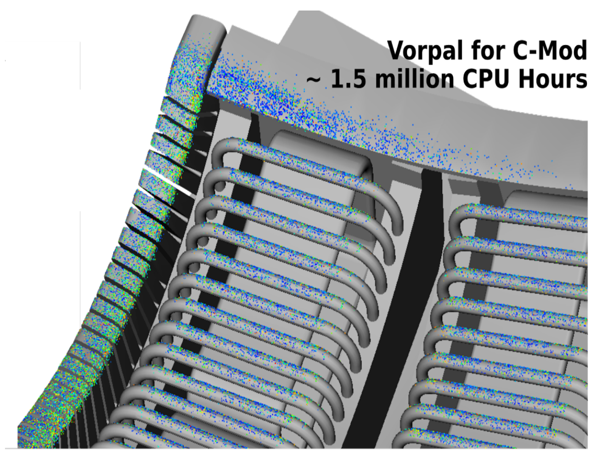






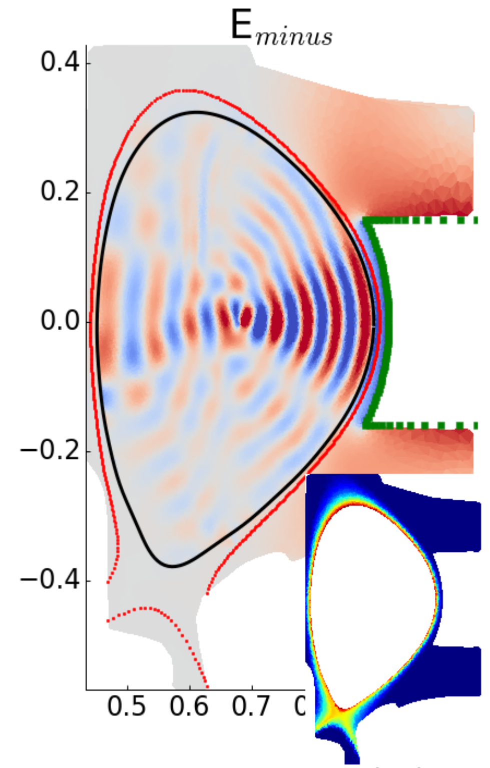
•These simulation models compute RF wave propagation and absorption including linear and non-linear effects using variety of numerical schemes

- Spectral code simulations of core LH and IC waves
- FDTD (finite difference time domain) simulation of ICRF antenna on C-Mod

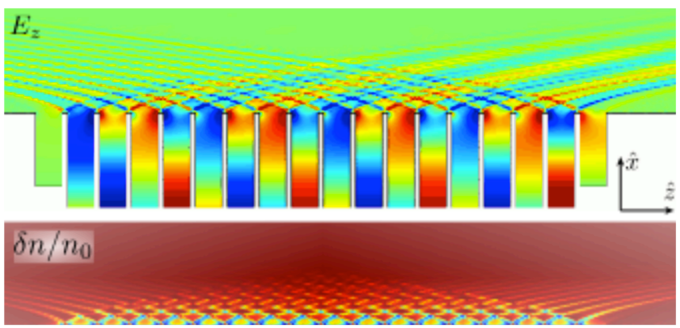


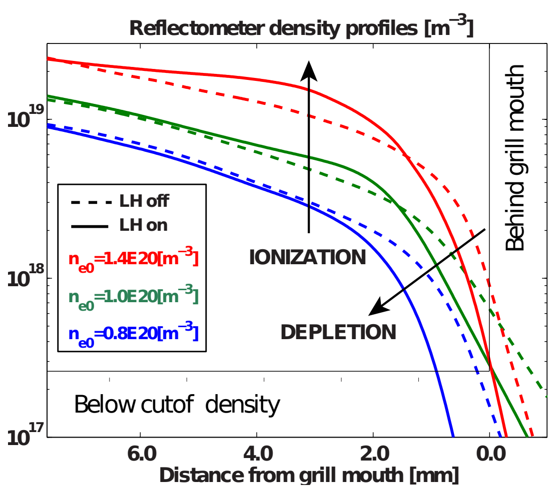
FEM has been used to handle complicated geometry, multi-physics/code integration

- (right) FEM solution in edge is coupled with the core spectral solver solution
- (bottom) LH wave propagation near the antenna including
  - Ponderomotive force acting on plasma
  - Modification of n_e



Density in SOL

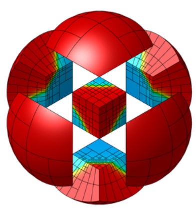




O. Meneghini, Ph.D Thesis (2011)
C. Lau, PPCF, (2013)

Petra-M is developed on the MFEM library and applied to various RF wave problems in fusion plasmas

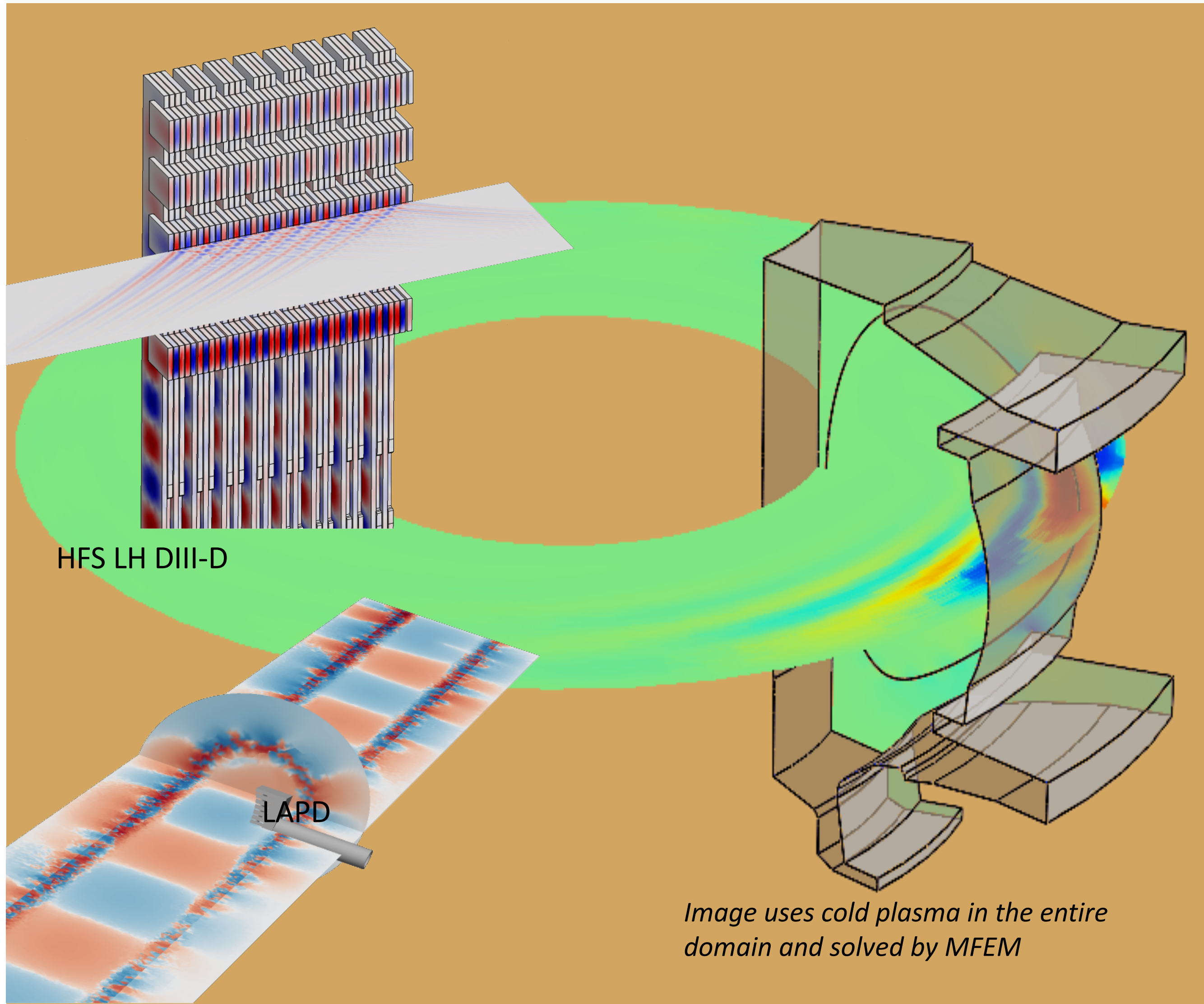
- Scalable MFEM library
  - http://mfem.org/features



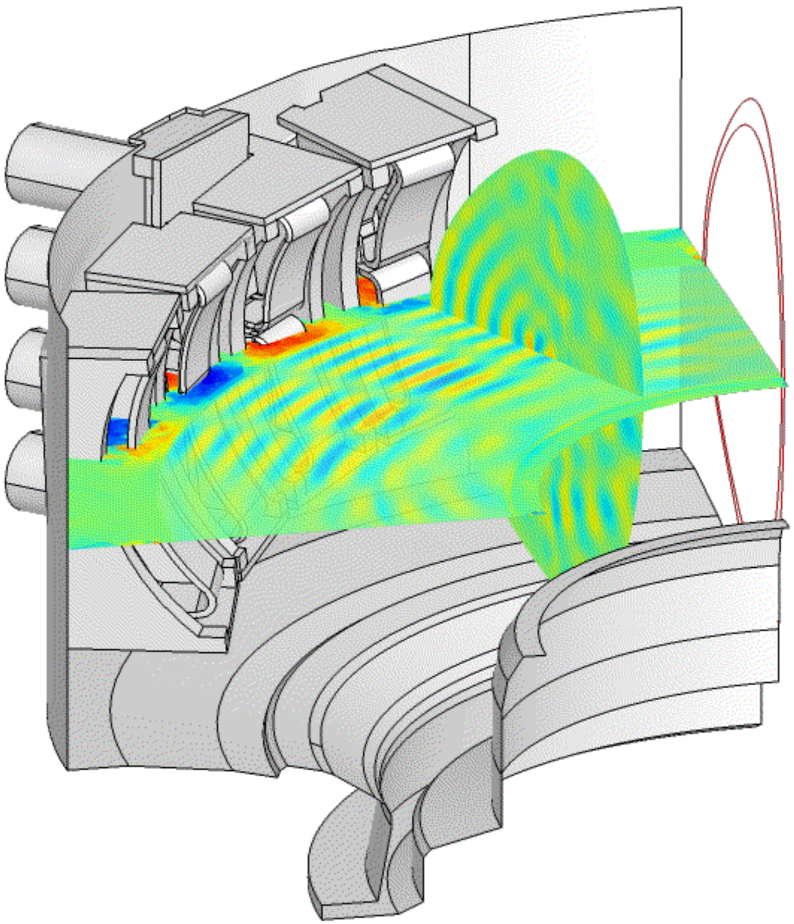
- Petra-M physics

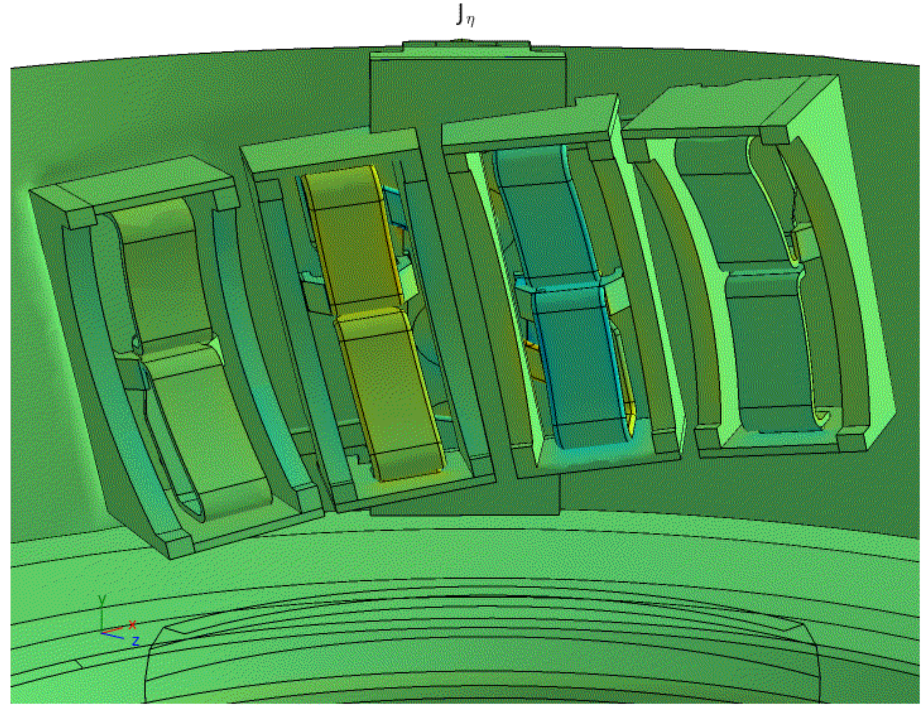
  based FEM modeling interface

- Workflow management using πScope
  - http://piscope.psfc.mit.edu



Petra-M was also coupled with the TORIC core spectral solver





- ICRF wave antenna propagation in the Alcator C-Mod tokamak
- Petra-M solves the RF field propagation in cold plasma near the antenna in 3D geometry
- TORIC solves the RF field propagation in hot core region

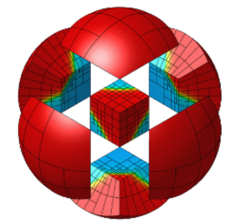
Petra-M: Physics Equation Translator for MFEM

MFEM

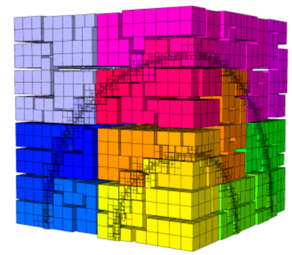
Lawrence Livermore National Laboratory

Free, lightweight, scalable C++ library for finite element methods. Supports arbitrary high order discretizations and meshes for a wide variety of applications.

- **Flexible discretizations on unstructured grids**
  - Triangular, quadrilateral, tetrahedral and hexahedral meshes.
  - Local conforming and non-conforming refinement.
  - High-order mesh optimization (ASCR Base).
  - Bilinear/linear forms for variety of methods: Galerkin, DG, DPG, …
- **High-order methods and scalability**
  - Arbitrary-order H1, H(curl), H(div)- and L2 elements. Arbitrary order curvilinear meshes.
  - MPI scalable to millions of cores. Enables application development on wide variety of platforms: from laptops to exascale machines.
- **Solvers and preconditioners**
  - Integrated with: HYPRE, SUNDIALS, PETSc, SUPERLU, …
  - Auxiliary-space AMG preconditioners for full de Rham complex
- **Open-source software**
  - Open-source (GitHub) with thousands of downloads/year worldwide
  - Part of FASTMath, ECP/CEED, xSDK, OpenHPC, ...



High order curved elements

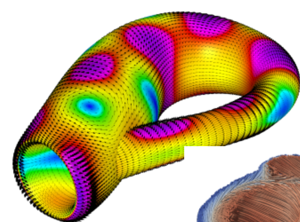


Parallel non-conforming AMR

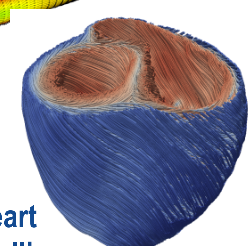
CEED EXASCALE DISCRETIZATIONS

FASTMATH

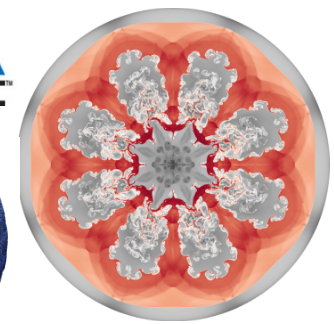
ASC



Surface meshes



Heart modelling



Compressible flow ALE simulations

http://mfem.org

MFEM provides building blocks to developing FEM application

```python
import mfem
from mfem import intArray
import numpy as np

order = 1
mesh = mfem.Mesh('/Users/shiraiwa/mfem-3.1/data/beam-tet.mesh', 1,1)

dim = mesh.Dimension()
sdim= mesh.SpaceDimension
ref_levels = int(np.floor(np.log(50000./mesh.GetNE())/np.log(2.)/dim))
for x in range(ref_levels):
    mesh.UniformRefinement();
mesh.ReorientTetMesh();
fec = mfem.ND_FECollection(order, dim)
fespace = mfem.FiniteElementSpace(mesh, fec)
print("Number of finite element unknowns: " + str(fespace.GetTrueVSize()))
ess_tdof_list = intArray();
if mesh.bdr_attributes.Size():
     ess_bdr = intArray(mesh.bdr_attrib
     ess_bdr = intArray([1]*mesh.bdr_attributes.Max())
     fespace.GetEssentialTrueDofs(ess_bdr, ess_tdof_list);

from mfem.examples.ex3 import f_exact_cb, set_dim, set_freq
set_freq(1)
set_dim(3)
b = mfem.LinearForm(fespace);
f = mfem.VectorFunctionCoefficient(sdim, f_exact_cb);
dd = mfem.VectorFEDomainLFIntegrator(f);
b.AddDomainIntegrator(dd)
b.Assemble()
from mfem.examples.ex3 import E_exact_cb
x = mfem.GridFunction(fespace);
E = mfem.VectorFunctionCoefficient(sdim, E_exact_cb);
x.ProjectCoefficient(E);
muinv = mfem.ConstantCoefficient(1.0);
sigma = mfem.ConstantCoefficient(1.0);
a = mfem.BilinearForm(fespace);
a.AddDomainIntegrator(mfem.CurlCurlIntegrator(muinv));
a.AddDomainIntegrator(mfem.VectorFEMassIntegrator(sigma));
static_cond = False
if (static_cond):  a.EnableStaticCondensation()
a.Assemble();
A = mfem.SparseMatrix()
B = mfem.Vector()
X = mfem.Vector()
a.FormLinearSystem(ess_tdof_list, x, b, A, X, B);
## Here, original version calls hegith, which is not
## defined in the header...!?
print("Size of linear system: " + str(A.Size()))
M = mfem.GSSmoother(A)
mfem.PCG(A, M, B, X, 1, 500, 1e-12, 0.0);
a.RecoverFEMSolution(X, b, x)
print("|| E_h - E ||_{L^2} = " + str(x.ComputeL2Error(E)))
```

Read Mesh File

Allocate FE space

Set Essential BC

Allocate Matrix and RHS

Fill Matrix and RHS

Solve

$$\nabla \times \nabla \times E + E = f$$

$$\downarrow$$

$$M\,x = b$$

In a short ~60 lines, it does...

- Define Finite Element Function Space
- Matrix Assembly/Solve

Changes required for parallelize the code is small.

PyMFEM (MFEM Python-binding) is developed for rapid development

- Allows for construct, manipulate MFEM c++ objects
- Supports both parallel (with MPI) and serial (w/o MPI) MFEM
- Allows for defining FunctionCoefficient using python class
- (Partial) Supports passing numpy array as argument and return value

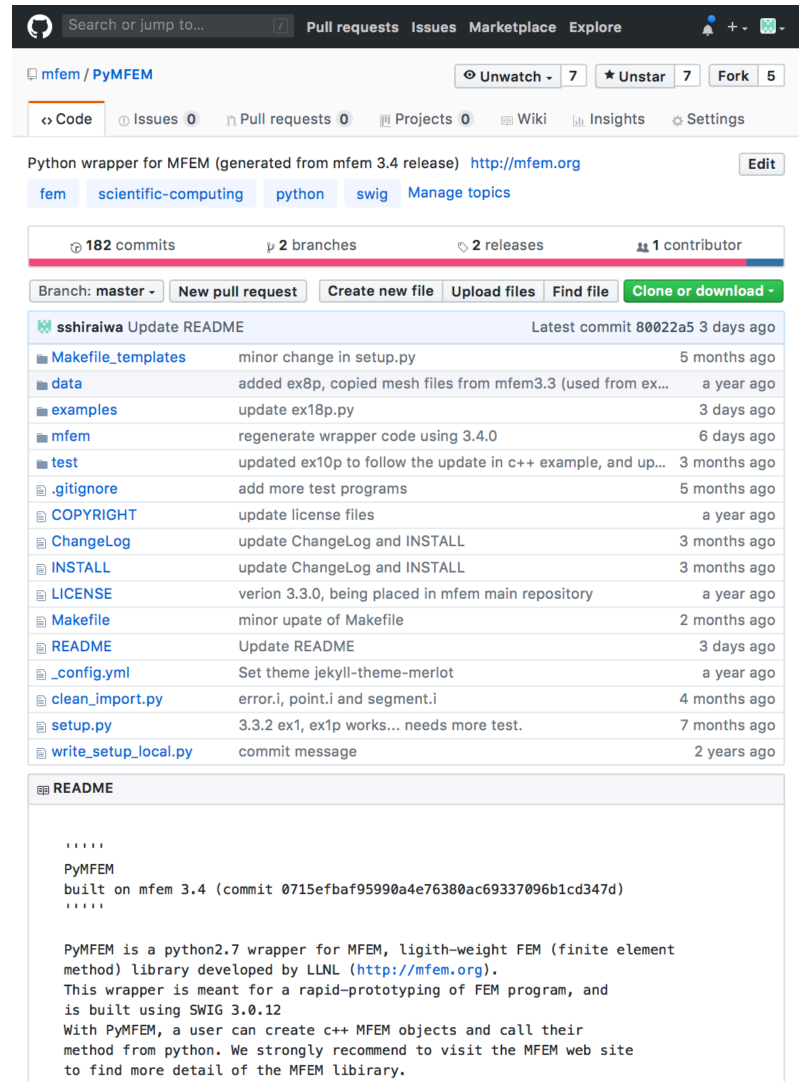
  (c++)

  ```
  double data[] = {1,2,3};
          o = Vector (data, 3);
  ```

  (python)

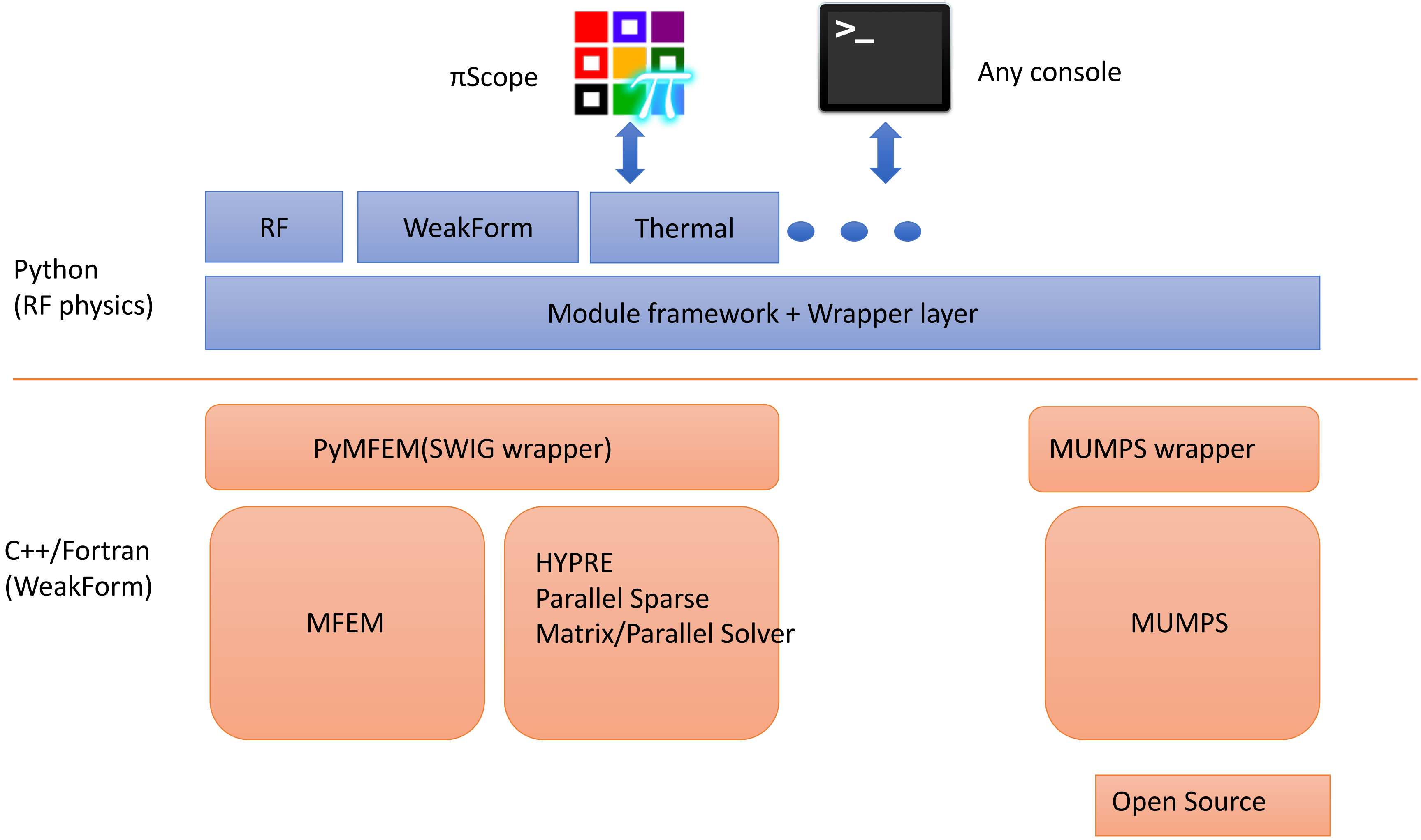
  ```
  v = mfem.Vector(np.array([1,2,3.])
  ```

- Create HypreParCSR/HypreVector using distributed scipy.sparse matrix



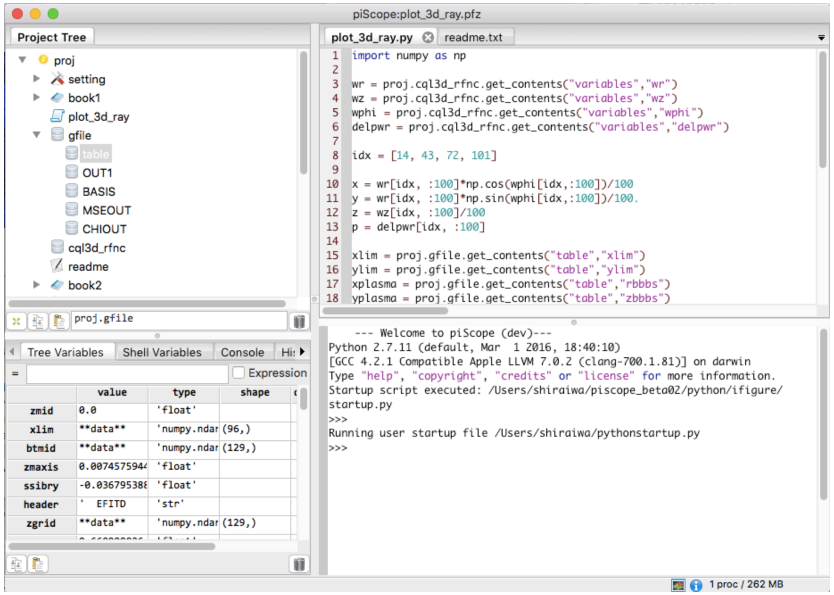
Availale from GitHub (LGPL)

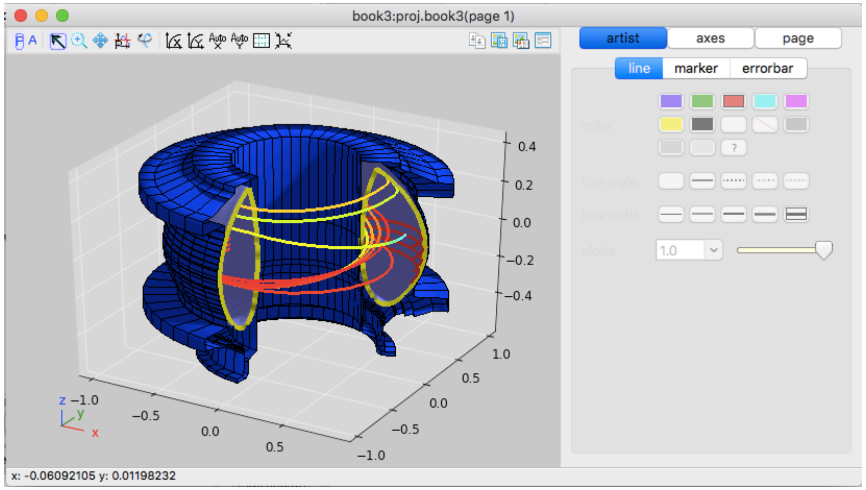
Petra-M uses Python for rapid physics module development, while using scalable FEM and solver libraries

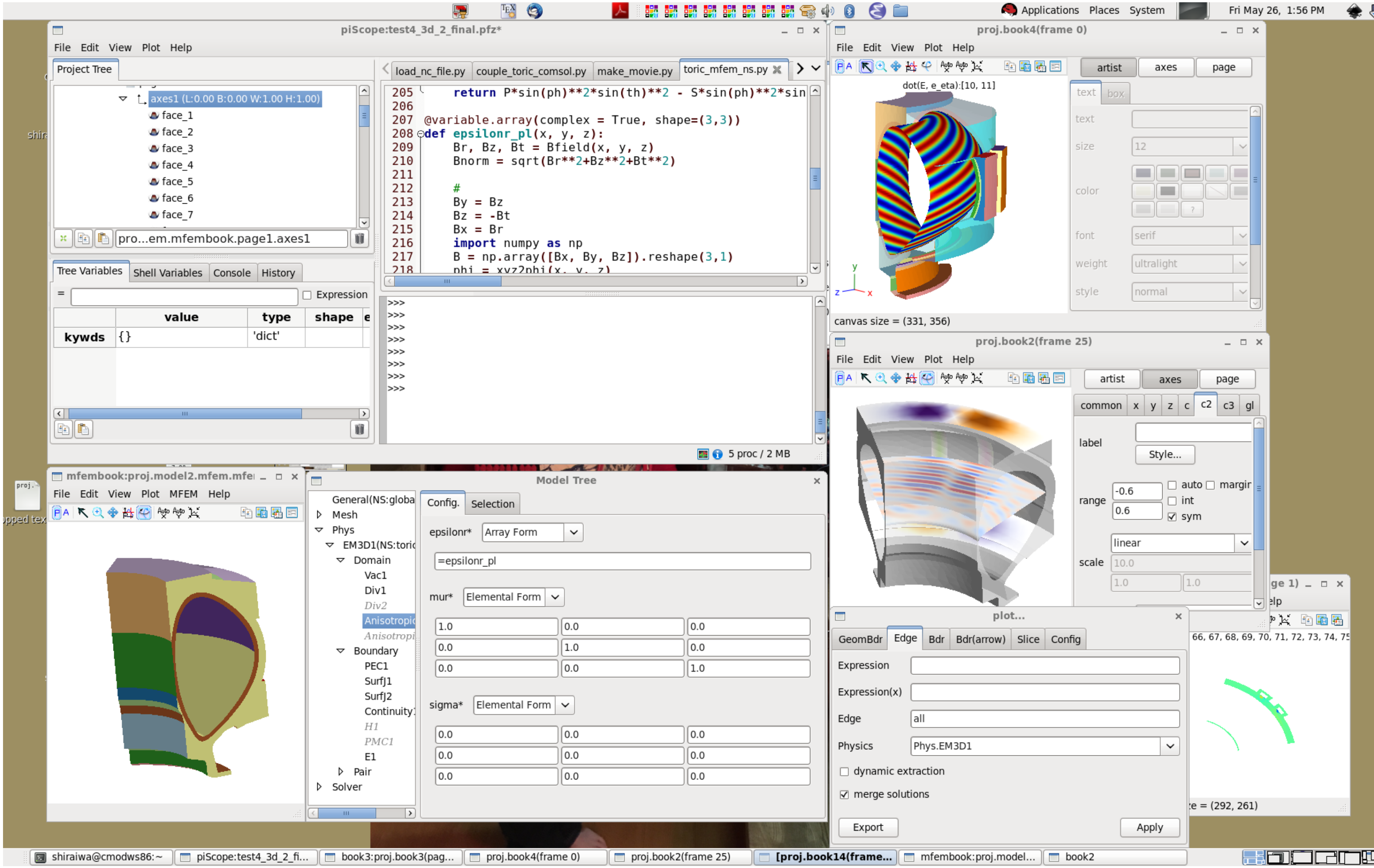


Model setup interface is built on πScope

- πScope
  - All-in-one style python data analysis environment
    - Shell
    - Editor
    - Debugger
    - Project management
    - Data Browser
  - Interactive plotting environment
    - Based on matplotlib + custom GUIs to edit figure
    - OpenGL based 3D graphics
  - Native support to browse MDSplus data system
  - Components to support code integration
  - Open Source (GNUv3)

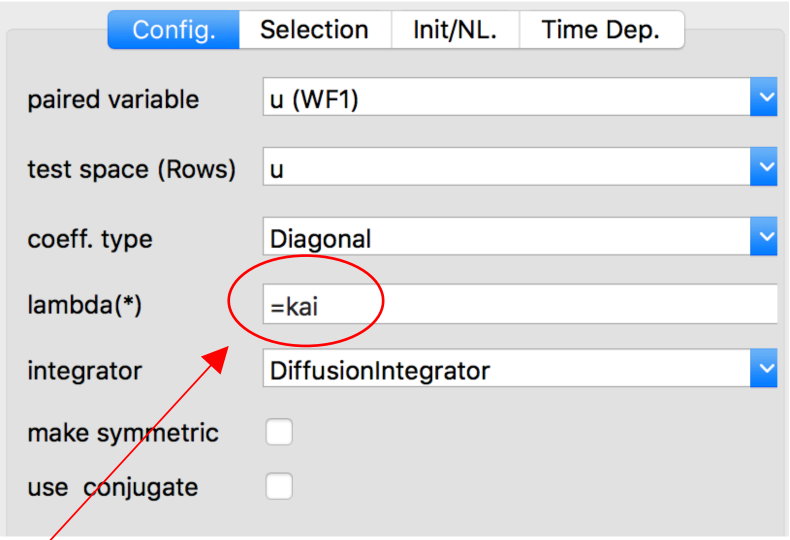





Applications Places System
Fri May 26, 1:56 PM
piScope:test4_3d_2_final.pfz*
File Edit View Plot Help
Project Tree
axes1 (L:0.00 B:0.00 W:1.00 H:1.00)
face_1
face_2
face_3
face_4
face_5
face_6
face_7
pro...em.mfembook.page1.axes1
Tree Variables
Shell Variables
Console
History
Expression
value
type
shape
kywds
{}
'dict'
load_nc_file.py
couple_toric_comsol.py
make_movie.py
toric_mfem_ns.py
return P*sin(ph)**2*sin(th)**2 - S*sin(ph)**2*sin
@variable.array(complex = True, shape=(3,3))
def epsilonr_pl(x, y, z):
Br, Bz, Bt = Bfield(x, y, z)
Bnorm = sqrt(Br**2+Bz**2+Bt**2)
#
By = Bz
Bz = -Bt
Bx = Br
import numpy as np
B = np.array([Bx, By, Bz]).reshape(3,1)
5 proc / 2 MB
proj.book4(frame 0)
artist
axes
page
dot(E, e_eta):[10, 11]
text
box
size
12
color
font
serif
weight
ultralight
style
normal
canvas size = (331, 356)
proj.book2(frame 25)
common
label
Style...
range
-0.6
0.6
auto
margin
int
sym
linear
scale
10.0
1.0
mfembook:proj.model2.mfem.mfe
File Edit View Plot MFEM Help
Model Tree
General(NS:globa
Mesh
Phys
EM3D1(NS:toric
Domain
Vac1
Div1
Div2
Anisotropic
Boundary
PEC1
SurfJ1
SurfJ2
Continuity1
H1
PMC1
E1
Pair
Solver
Config.
Selection
epsilonr*
Array Form
=epsilonr_pl
mur*
Elemental Form
sigma*
Elemental Form
plot...
GeomBdr
Edge
Bdr
Bdr(arrow)
Slice
Config
Expression
Expression(x)
Edge
all
Physics
Phys.EM3D1
dynamic extraction
merge solutions
Export
Apply
shiraiwa@cmodws86:~
piScope:test4_3d_2_fi...
book3:proj.book3(pag...
proj.book4(frame 0)
proj.book2(frame 25)
[proj.book14(frame...
mfembook:proj.model...
book2

Material properties in Petra-M

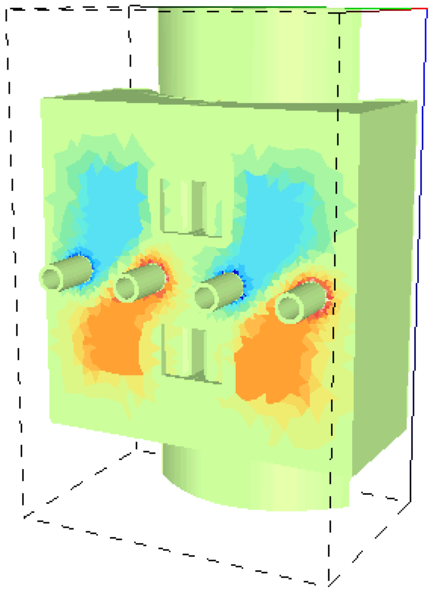
- Material property can be defined by using a Python function + decorator
  - A Python decorator specifies the type (float/complex) and shape of returned value
  - Supports Spatial dependence.
  - MFEM calls this function with the special coordinates at quadrature integration points.
  - Can use a value from a solution from the previous solve step.



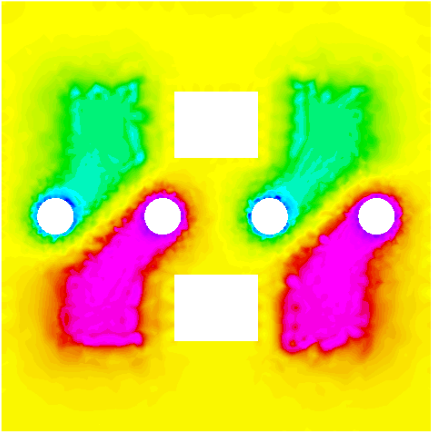

```python
import numpy as np
from petram.helper.variables import variable

@variable.array(complex=False, shape=(3,))
def kai(x, y, z):
    return np.array([1., 0.1, 0.1])

```

Mesh extension and DoF projection/mapping

- We often need to handle the coupling of PDEs defined over geometrically different regions
  - (Example) RF propagation in waveguide (RF) and heating of waveguide (heat)
- Petra-M generates automatically daughter meshes.
  - Daughter meshes inherit domain/boundary attributes for the mother mesh.
  - DoF mapping operator (can be either projection or linear solve.)
  - Works for H1/ND/RT elements.
  - Supports arbitrary polynomial order.
  - Supports a curved mesh.



RF wave field in induced on the metal wall of antenna box.



- Perpendicular electrci field on the surface, generated by mapping 3D RF fied

RF module (frequency domain Maxwell prob.)

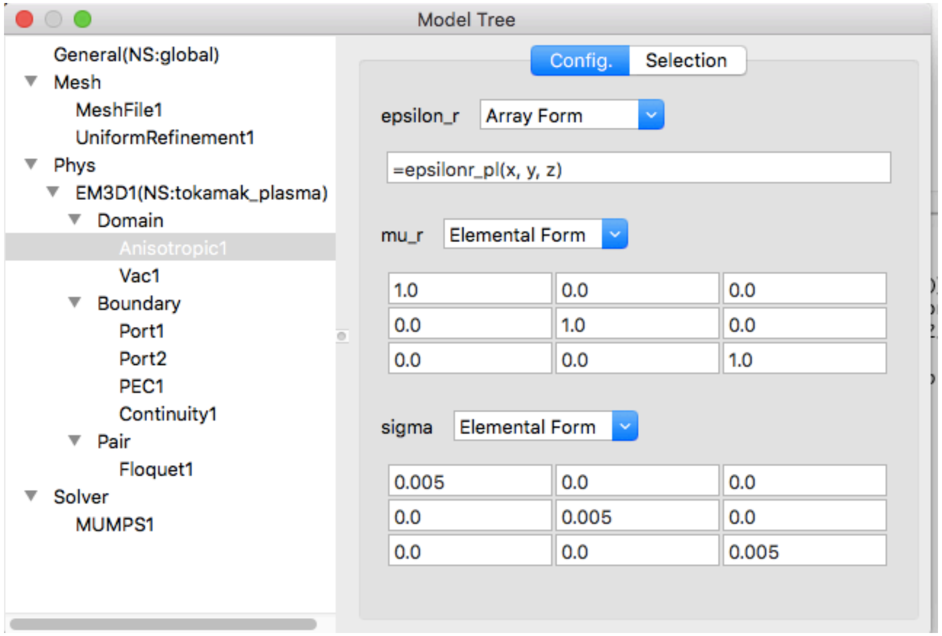
Maxwell eq.

$$
\begin{aligned}
\nabla \times \left(\mu^{-1}\nabla \times \boldsymbol{E}\right) - \left(\omega^2\epsilon - j\omega\sigma\right)\boldsymbol{E} &= -j\omega \boldsymbol{J}^s \text{ in } S,\\
\hat{\boldsymbol{n}} \times \boldsymbol{E} &= \boldsymbol{P} \text{ on } L_1,\\
\hat{\boldsymbol{n}} \times \left(\mu^{-1}\nabla \times \boldsymbol{E}\right) + \gamma\,\hat{\boldsymbol{n}} \times \hat{\boldsymbol{n}} \times \boldsymbol{E} &= \boldsymbol{Q} \text{ on } L_2.
\end{aligned}
$$

Weak form

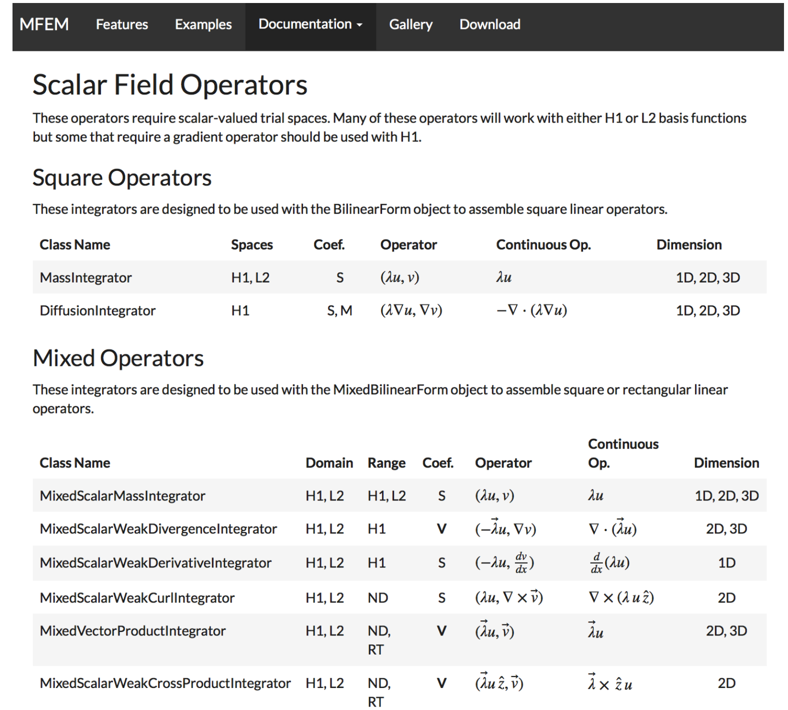
$$
\int_S \left[\mu^{-1}\left(\nabla \times \boldsymbol{W}_i\right)\cdot\left(\nabla \times \boldsymbol{E}\right) - \left(\omega^2\epsilon - j\omega\sigma\right)\boldsymbol{W}_i \cdot \boldsymbol{E}\right] dS
+ \int_{L_2} \boldsymbol{W}_i \cdot \left(\boldsymbol{Q} - \boxed{\gamma\,\hat{\boldsymbol{n}} \times \hat{\boldsymbol{n}} \times \boldsymbol{E}) \; dl}\right. = -j\omega \int_S \boldsymbol{W}_i \cdot \boldsymbol{J}^s dS.
$$

- Domain
  - Uniform dielectric media
  - Anisotropic (matrix) media
  - External J
  - DivJ constraints in vacuum
- Boundary
  - Perfect electric conductor (E_t=0)
  - Perfect magnetic conductor (B_t=0)
  - Waveguide port (TE, TEM modes)
  - Periodic boundary
  - Surface current/Magnetic field/Electric field



WF (weak form) module

- Used to construct an FEM model by specifying MFEM integrators (right)
- Can be coupled with other physics simulation model to define multi-physics coupling.


MFEM Features Examples Documentation Gallery Download

Scalar Field Operators

These operators require scalar-valued trial spaces. Many of these operators will work with either H1 or L2 basis functions but some that require a gradient operator should be used with H1.

Square Operators

These integrators are designed to be used with the BilinearForm object to assemble square linear operators.

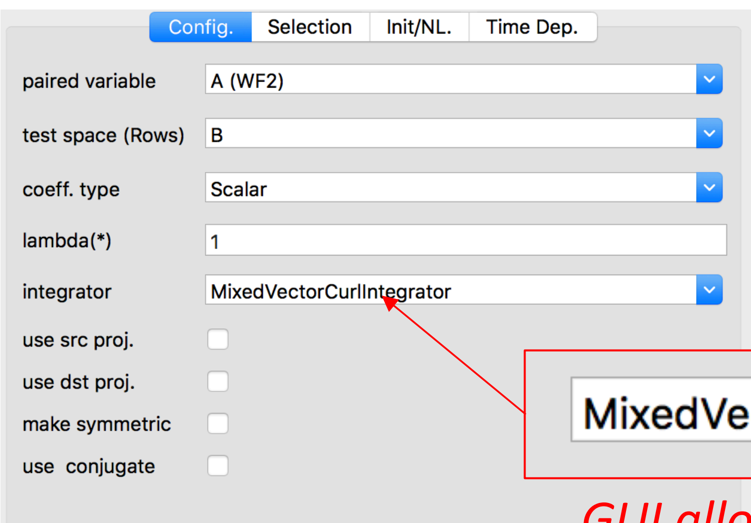
Class Name	Spaces	Coef.	Operator	Continuous Op.	Dimension
MassIntegrator	H1, L2	S	$(\lambda u, v)$	λu	1D, 2D, 3D
DiffusionIntegrator	H1	S, M	$(\lambda \nabla u, \nabla v)$	$-\nabla \cdot (\lambda \nabla u)$	1D, 2D, 3D

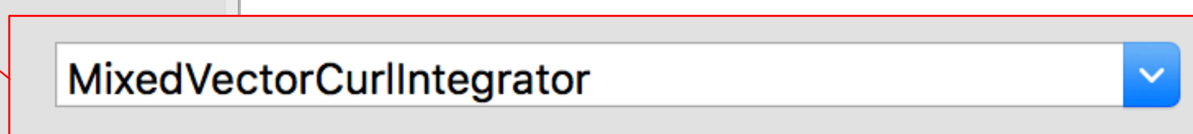
Mixed Operators

These integrators are designed to be used with the MixedBilinearForm object to assemble square or rectangular linear operators.

Class Name	Domain	Range	Coef.	Operator	Continuous Op.	Dimension
MixedScalarMassIntegrator	H1, L2	H1, L2	S	$(\lambda u, v)$	λu	1D, 2D, 3D
MixedScalarWeakDivergenceIntegrator	H1, L2	H1	V	$(-\vec{\lambda} u, \nabla v)$	$\nabla \cdot (\vec{\lambda} u)$	2D, 3D
MixedScalarWeakDerivativeIntegrator	H1, L2	H1	S	$(-\lambda u, \frac{dv}{dx})$	$\frac{d}{dx}(\lambda u)$	1D
MixedScalarWeakCurlIntegrator	H1, L2	ND	S	$(\lambda u, \nabla \times \vec{v})$	$\nabla \times (\lambda\, u\, \hat{z})$	2D
MixedVectorProductIntegrator	H1, L2	ND, RT	V	$(\vec{\lambda} u, \vec{v})$	$\vec{\lambda} u$	2D, 3D
MixedScalarWeakCrossProductIntegrator	H1, L2	ND, RT	V	$(\vec{\lambda} u\, \hat{z}, \vec{v})$	$\vec{\lambda} \times \hat{z}\, u$	2D

See http://mfem.org/bilininteg/ and http://mfem.org/lininteg/ for the full list

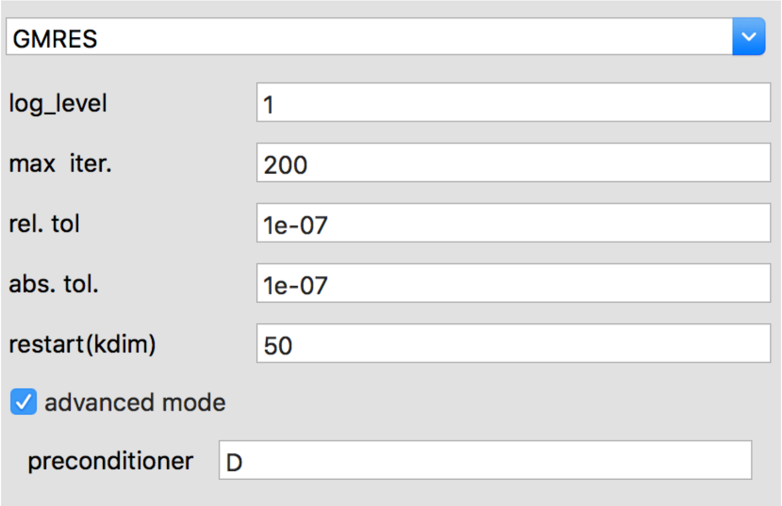




GUI allows to pick an MFEM integrator from menu

Solver

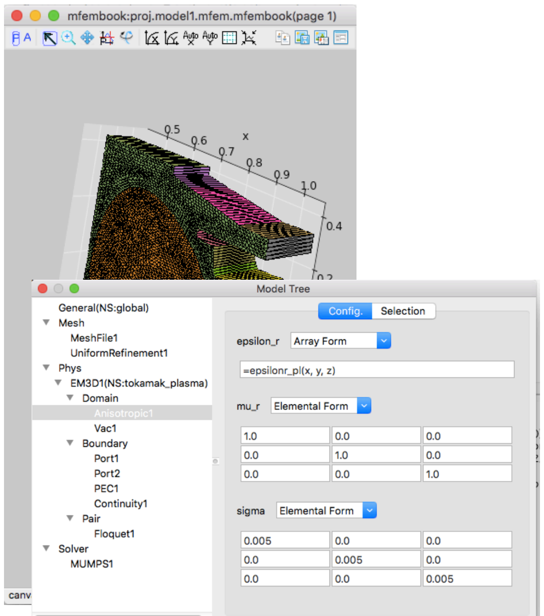
- Petra-M supports variety of linear solvers through MFEM library
- Linear Solvers
  - Direct solver:
    - MUMPS/Strumpack.
  - Iterative solver:
    - Krylov subspace solvers from Hypre (CG, MINRES, GMRES, FGMRES, BiCGSTAB…)
    - Block preconditioner GUI
- A user can construct a solution sequence, which is made from multiple linear-solve/time dependent solver runs.




```python
@prc.blk_diagonal
def D(p, g, *args, **kwargs):
    GSgen.set_param(g, "v")
    gs = GSgen()
    k = g.get_row_by_name("v")
    p.SetDiagonalBlock(k, gs) # set to block
    return p
```

Custom preconditioner can be defined using a Python decorator

Petra-M runs on a laptop, but also allows for submitting a job to a cluster



Model data files

Solutions/Processed data

- User frontend is used define a simulation mode using physics interface.
- Simulation model is "preprocessed" to make input files
    - Resolve boundary index
    - Analyze geometry data
- Input files are Python scripts, allowing for modifying the model on a server before running.

HTS modeling efforts

Goal: adopt a framework for simulating RF in plasmas to static and dynamic HTS magnet systems

- Many commonality between HTS and waves in plasmas
  - Highly anisotropic and non-linear material properties
  - Complex geometry
  - Maxwell's equations and heat transport
  - FEM Libraries demonstrated to run a large scale clusters
  - GUI sets supporting, multi-physics modeling
- Challenge
  - HTS tape is extremely thin compared to the size of magnet.
  - HTS exhibits strongly anisotropic and non-linear material property.

$$\nabla\times\left(\frac{1}{\mu}\nabla\times\vec{A}\right) = \overleftrightarrow{\sigma}\,(T,\vec{B},\vec{J})\vec{E}$$

$$\nabla(\overleftrightarrow{\sigma}\vec{E}) = 0, \nabla\cdot\vec{A} = 0$$

$$\vec{E} = -\frac{d\vec{A}}{dt} - \nabla\phi,\ \ \vec{B} = \nabla\times A$$

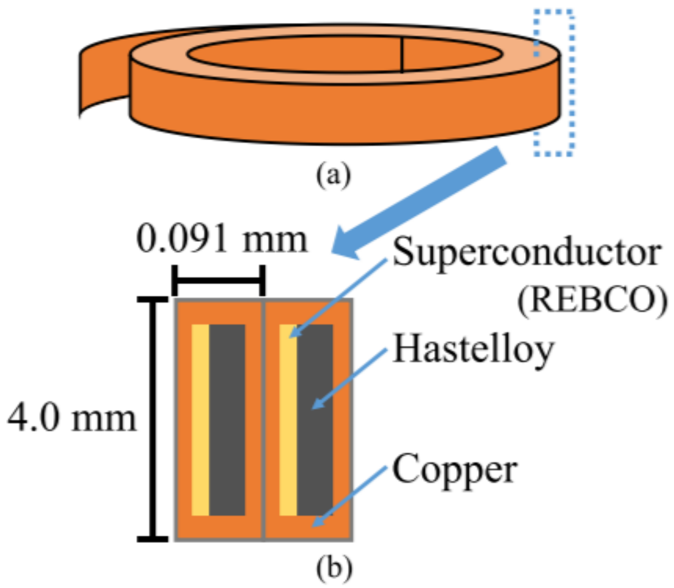
$$\rho C_p\frac{dT}{dt} - \nabla\overleftrightarrow{k}\nabla T = q \text{ , where } q = \vec{E}\cdot\vec{J}$$

PSFC

COMMONWEALTH FUSION SYSTEMS

Approach - Hierarchy of models

- Microscopic to Macroscopic properties
  - Investigate how stacks of tapes with different internal materials manifest thermal and electrical conductivity. See eg [Noguchi 2016 TAS]
  - This model to minimize peak voltages and heating
- Continuous media approximation
  - Circular cross section double pancake geometry
  - Tape not explicit in geometry.
    - Anisotropic electric conductivity
    - Rotate conductivity tensor to align with 'tape' position locally
      - Tape is on a spiral with finite pitch so there is a radial component
    - Uniform thermal conductivity from copper matrix



Summary

- Petra-M framework is being developed
  - Based on the scalable MFEM finite element library.
  - Developed originally to solve the frequency domain Maxwell problem in order to model RF propagation in a cold plasma.
  - Extended to run more complicated multi-physics type simulations.
- Application to HTS tape based magnet modeling has started, aiming towards,
  - Quench propagation analysis.
  - Magnet performance Analysis in whole tokamak device scale.

PSFC