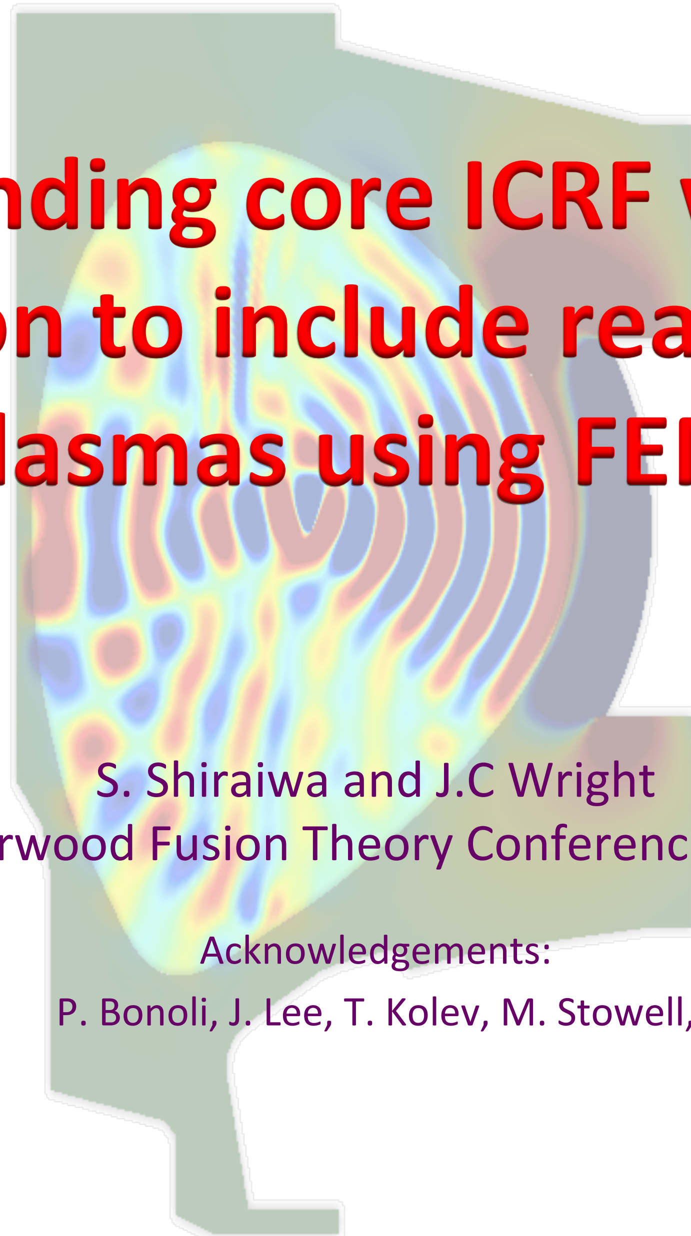


Extending core ICRF wave simulation to include realistic SOL plasmas using FEM

S. Shiraiwa and J.C Wright
Sherwood Fusion Theory Conference 2017

Acknowledgements:
P. Bonoli, J. Lee, T. Kolev, M. Stowell,

Motivation

- Need of self consistent simulation with realistic 3D antenna and plasma geometries.
- Various physics issues require coupling of edge (antenna and SOL) and core simulations:
  - Edge losses
  - Antenna coupling in 3D geometries (stellarators, C-Mod tilted ICRF) or small device plasmas
  - Multiple-pass absorption regimes
- Individually, in each (core or edge) region, good solvers are available. However, extending such a solver to the other region is difficult.
- This approach can extend to other frequency ranges, such as HHFW in NSTX, Helicon in DIIID, LHCD in C-Mod

Hybrid (core spectral / edge FEM) RF simulation. Why?

Spectral core:

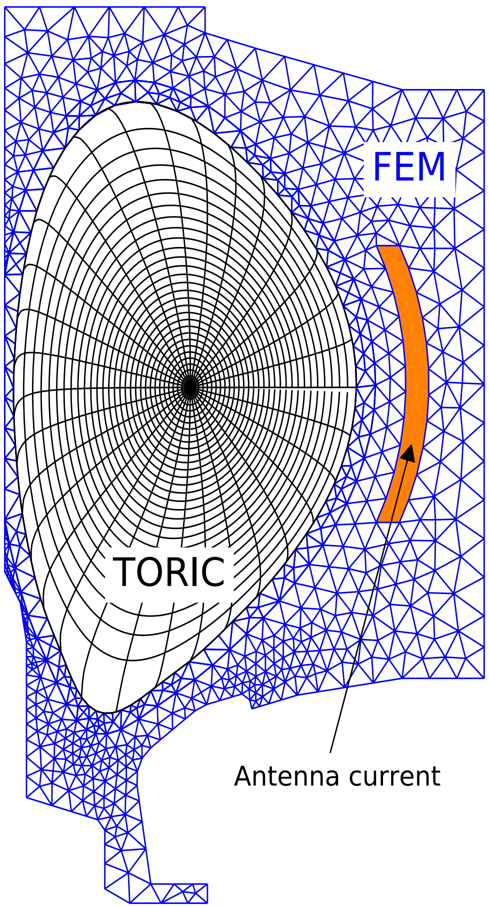
- Hot plasma formulation becomes algebraic
- Availability of mature scientific codes e.g. TORIC, AORSA, TASK, EVE,...
- 2D, 3D by single toroidal mode analysis

- Handling of sharp geometrical features is more difficult
- Dense matrices

FEM edge:

- Accurate geometry description (antenna, wall, SOL, ...)
- Cold plasma wave with collisions are straight-forward
- Not easy to deal with hot plasma effects (Needs a wave branch specific technique such as an iterative approach in LHEAF).
- Sparse matrices

2D TORIC-FEM coupling using COMSOL



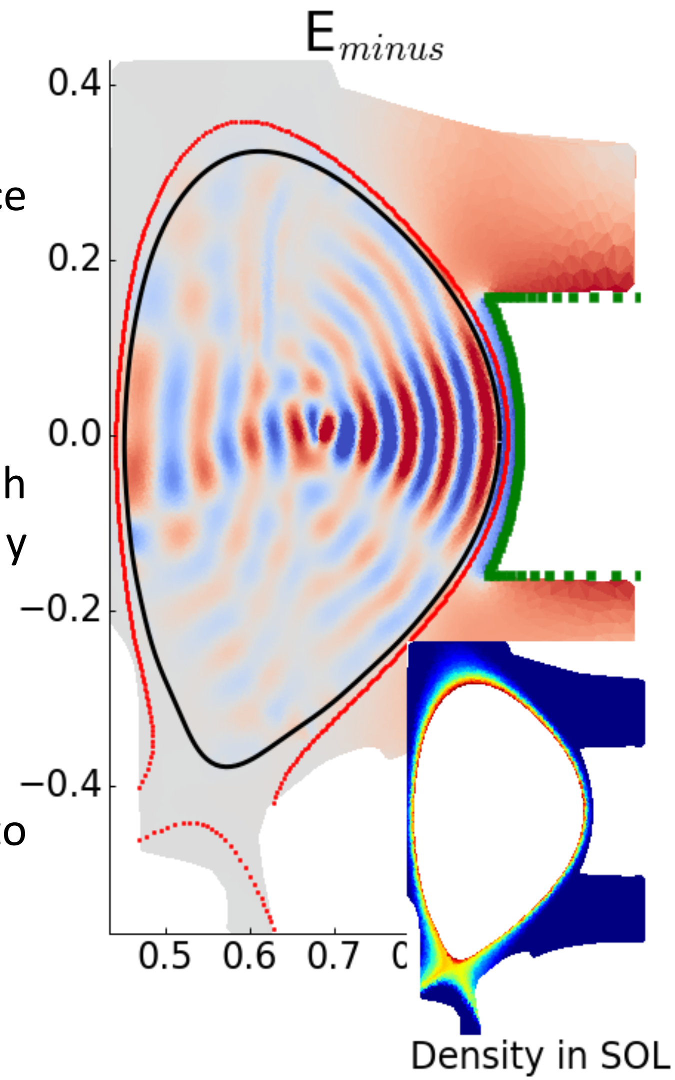
Core

- **Axisymmetric** flux surface regular grid
- Hot plasma conductivity
- Dense Matrix Solver

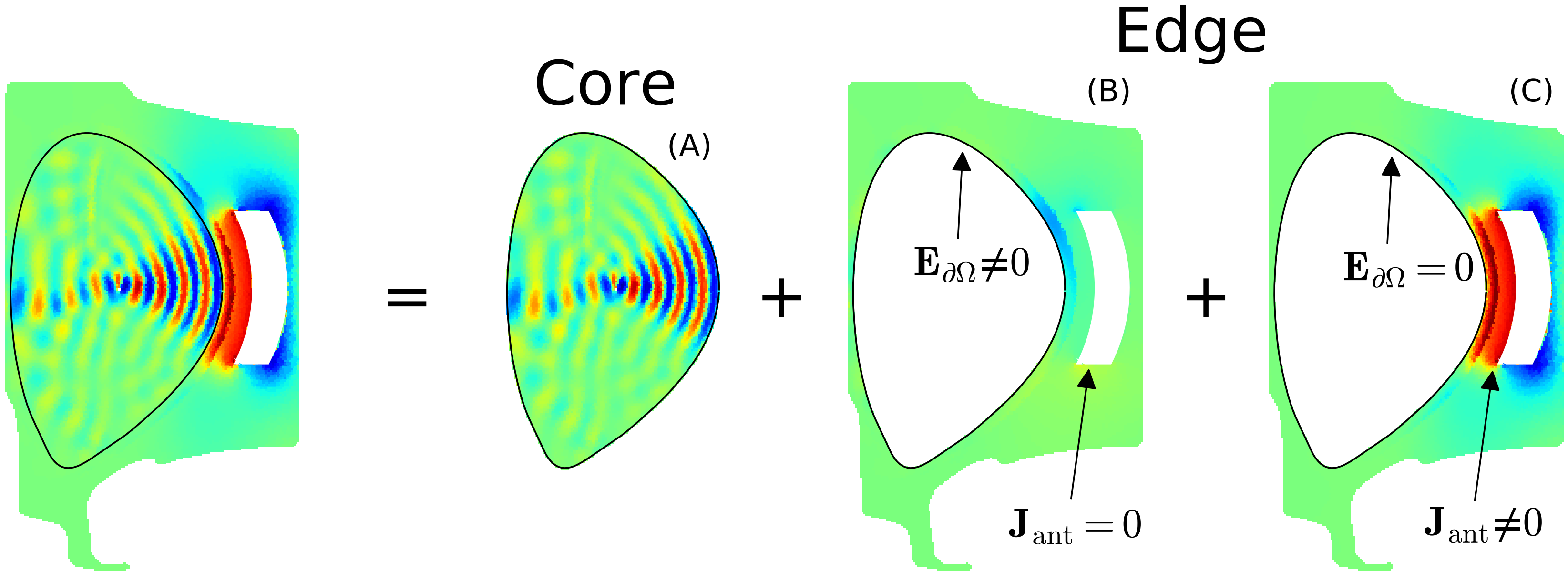
Edge

- Unstructured mesh with complicated geometry (either 2D or 3D)
- Cold plasma with collision.
- Tested using COMSOL

Boundary **matching technique** to build integrated solution

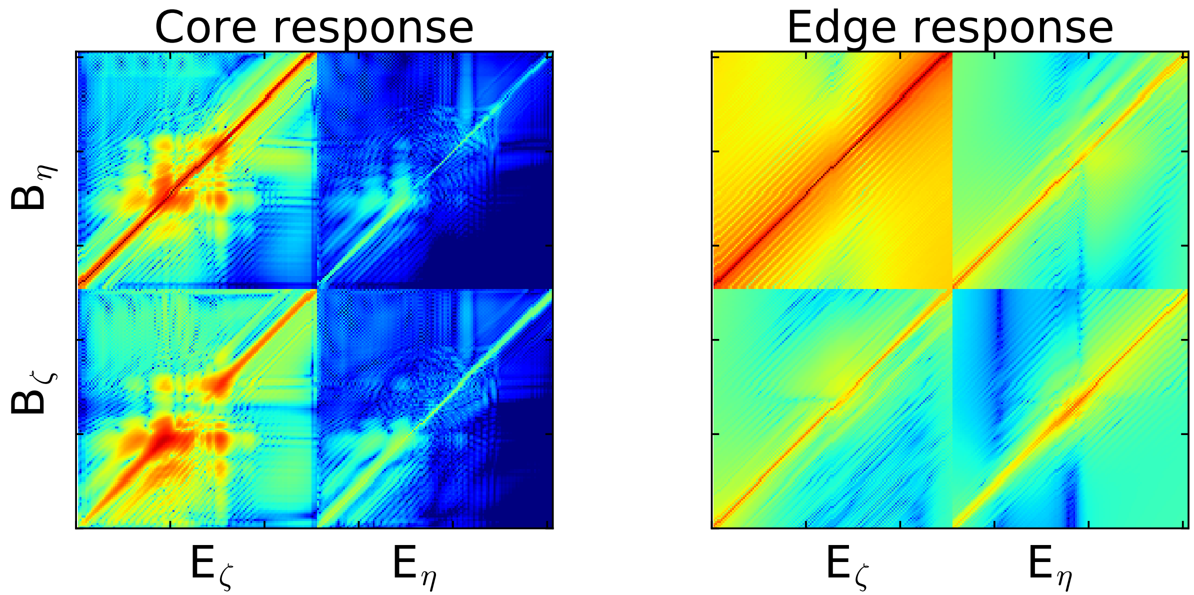


Domain partitioning



- Calculate the separate solutions in core (in) and edge (out) regions for different modal excitations.
- Superimpose the solutions, so that boundary conditions are satisfied.
- Note that linear system needs to be assembled so that mass matrices are the same

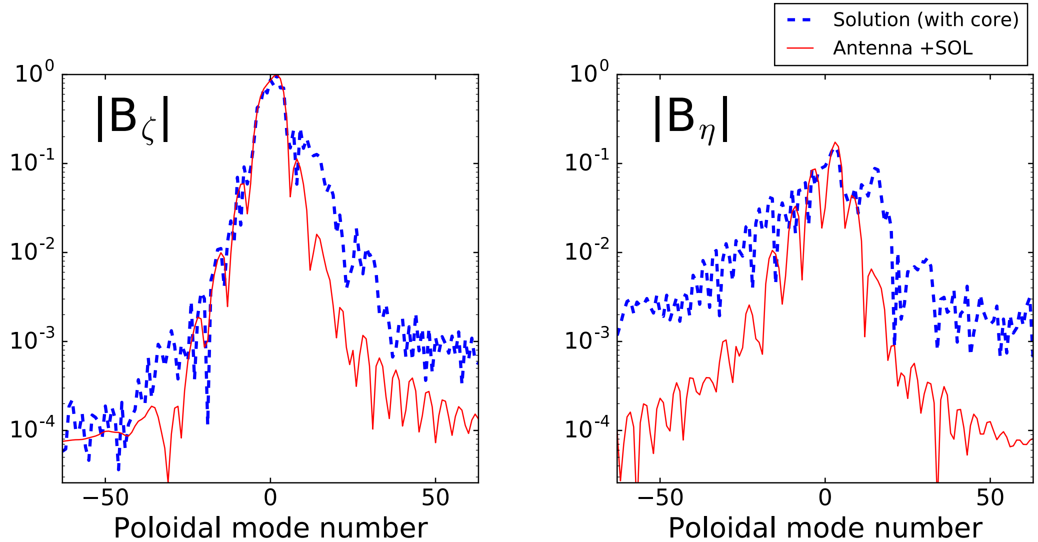
Mode superposition using response matrices

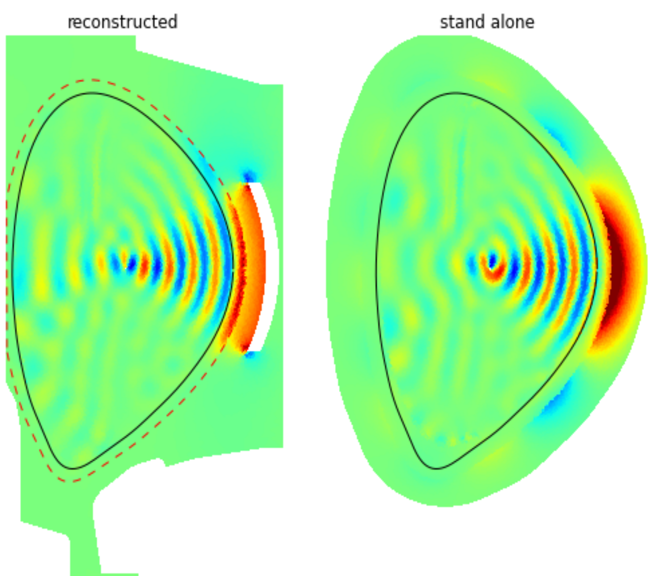


- Electric fields match for each mode at interface by construction
- Matching magnetic fields in presence of antenna currents given the matching amplitudes

$$\sum_m a_m {B_{core}}_m^{(k)} = \sum_m a_m {B_{edge}}_m^{(k)} + {B_{ant}}_m^{(k)}$$

Results and verification

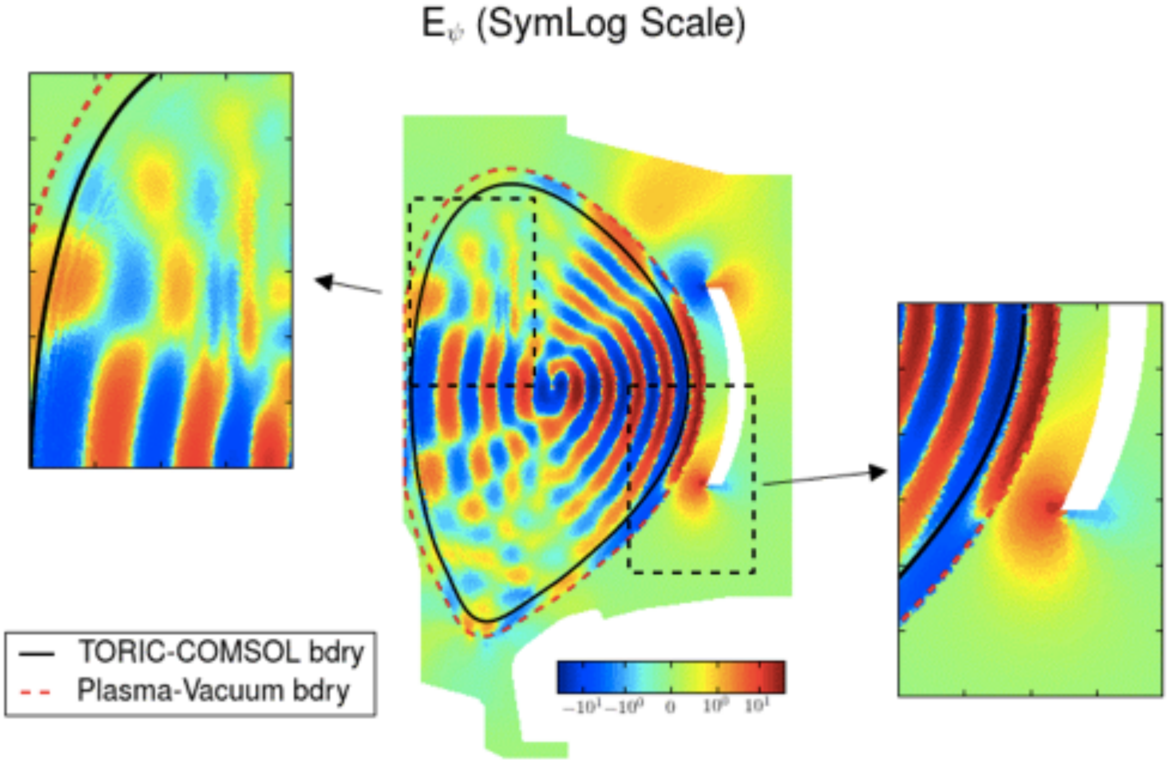




- Mode amplitude of superimposed solution (blue) spread wider than the antenna excitation amplitude (red)

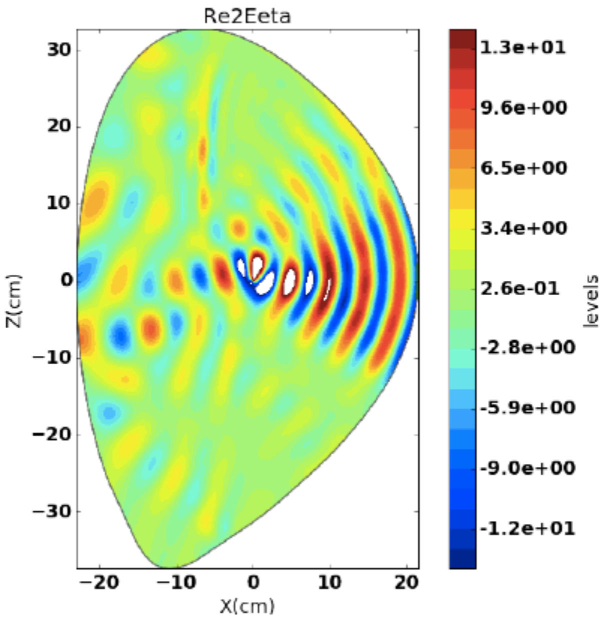
- In the core region, the superimposed solution (left) agreed well with the core solution of TORIC stand alone simulation (right)

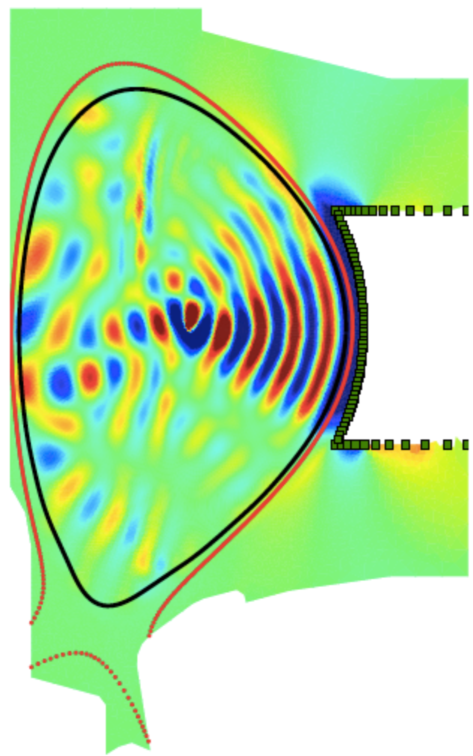
E_ψ continuity is not "built-in"

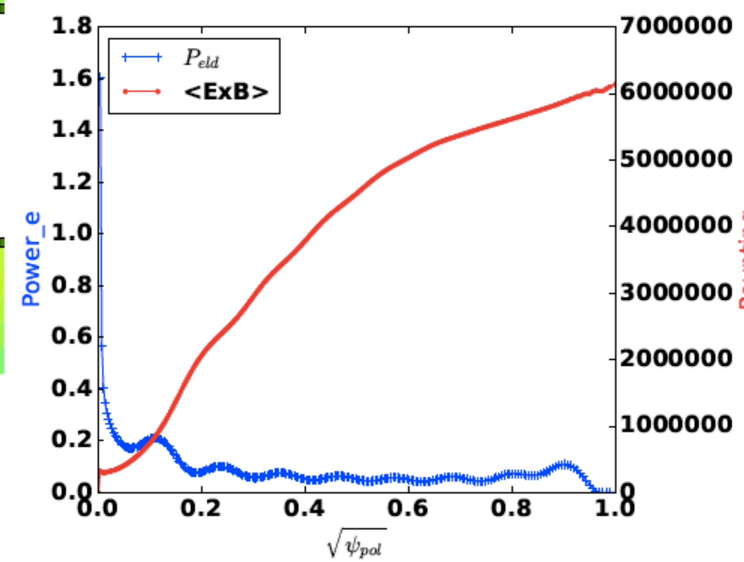


- Not given by construction provide a way to verify the approach
- Smoothly connected at TORIC/COMSOL boundary, but it is not at vacuum/ plasma boundary
- Consistent with a continuous dielectric at the former boundary, while it is not at the latter

Power absorption calculation needs re-run the codes using the obtained mode

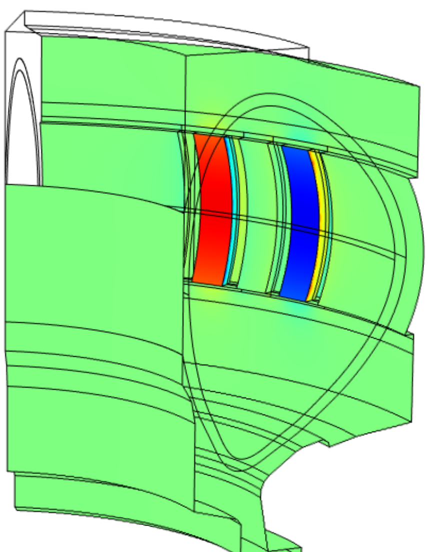


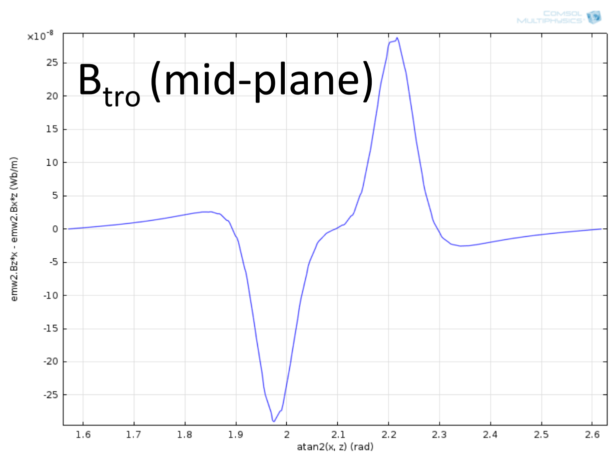


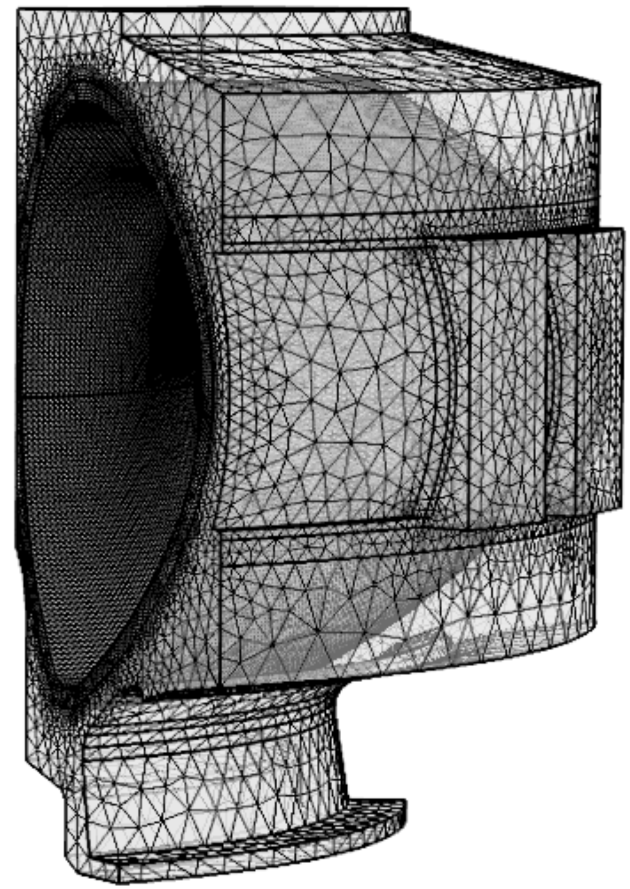


Extending 3D needs more computer resources

- Realistic Geometry
  - 60 deg vessel section
  - two strap antenna
  - LCSF from EFIT
- Quadratic EDGE elements yields a linear problem with 3M DoF (one model takes 30 min)
  - Good time for migrating to a cluster…

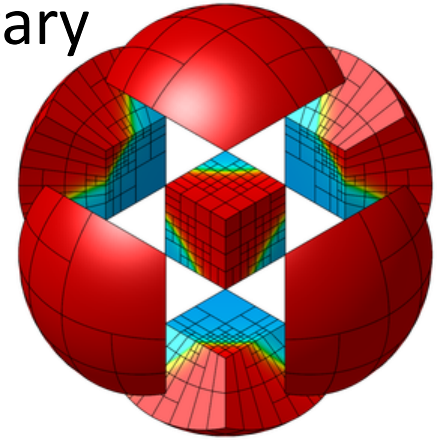






What is MFEM?

- *A free, lightweight, scalable* library for finite element methods (see http://mfem.org/features for detail)
  - Higher-order Finite Element Spaces: H1-, H(div)-, H(curl)-conforming spaces, and more
  - Triangular, quadrilateral, tetrahedral and hexahedral elements
  - Tightly integrated with Hypre scalable solver library
  - MPI-based parallelism throughout the library
  - Various examples including Maxwell. eq.
  - Written in C++.
- Powerful library to start from…



MFEM allows to develop FEM calculation in quick However, writing standalone program for every problem is inefficient

```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

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

$$\downarrow$$

$$M\,x = b$$

- Short ~60 lines
- Basis function
- Memory handling
- Solve

Problem needs to be defined in a weak form

Frequency domain Maxwell eq.

Maxwell eq.

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

Weak form

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

- $E_{//}= 0$ (Dirichlet) or $B_{//} = B_0$ are readily programmed.
- Needs user's coding to realize....
  - position depending parameters such as ε_{cold}
  - Port boundary : $E(x, y) = E_{inc} \exp(ikx) + E_{ref} \exp(-ikx)$
  - Periodic boundary : $E_{\Omega 1} = E_{\Omega 2}$
  - Finite impedance on surface: $\Upsilon \neq 0$
- 2D axisymmetric case requires different formulation.

PyMFEM = python wrapper for MFEM

- SWIG (simple wrapper interface generator)
- Allows for construct, manipulate MFEM c++ objects
- Allows for defining FunctionCoefficient using python class
- (Partial) Supports passing numpy array as argument and return value

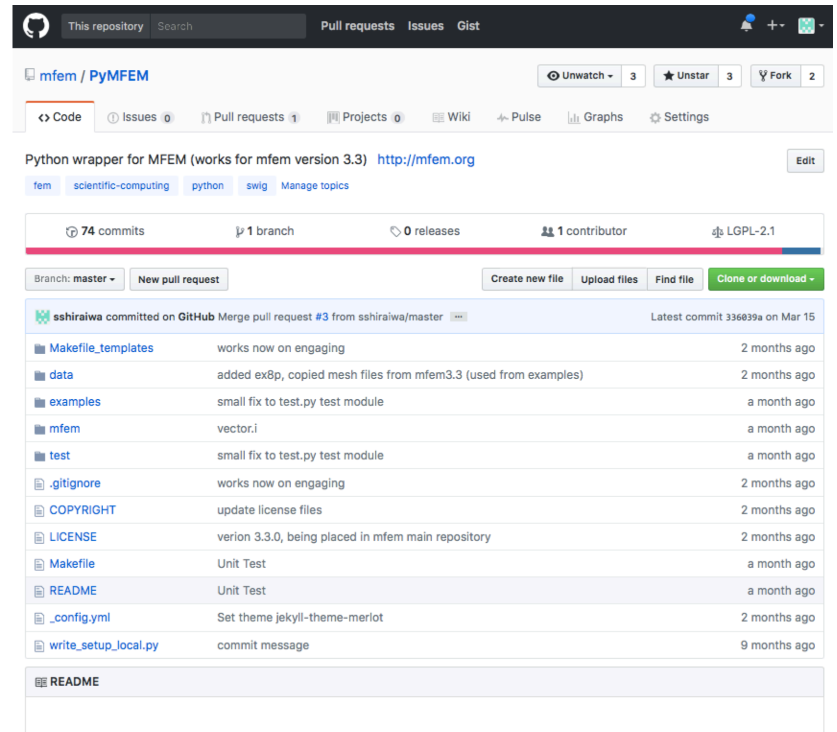
  (c++) double data[] = {1,2,3};

  o = Vector (data, 3);

  (python)

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

- Create HypreParCSR/HypreVector using distributed scipy.sparse matrix
- All 31 parallel/serial examples are translated in Python



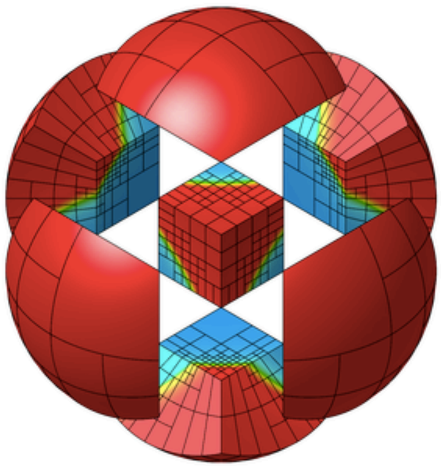
Jul. 2016 Put on GitHub for review

Sep. 2016 Released under LGPL v-2

Feb. 2017 Became part of MFEM repo.

.

MFEM Features Examples Documentation ▾ Gallery Download GitHub



MFEM is a *free*, *lightweight*, *scalable* C++ library for finite element methods.

Features

- Arbitrary high-order finite element meshes and spaces.
- Wide variety of finite element discretization approaches.
- Conforming and nonconforming adaptive mesh refinemen
- Scalable to hundreds of thousands of cores.
- ... and many more.

MFEM is used in many projects, including BLAST, XBraid and Vi

News

Jan 28, 2017	Version 3.3 released.
Dec 15, 2016	Postdoc position for exascale computing with MFEM.
Sep 12, 2016	PyMFEM - a Python wrapper for MFEM released.

Latest Release

New features ¦ Examples ¦ Code documentation ¦ Sources

Download mfem-3.3.tgz

For older releases see the download section.

Documentation

Twitter ->

Tzanio Kolev @TzanioKolev · Sep 19
A nice Python wrapper for **MFEM** by Syun'ichi Shiraiwa from MIT's Plasma Science and Fusion Center:

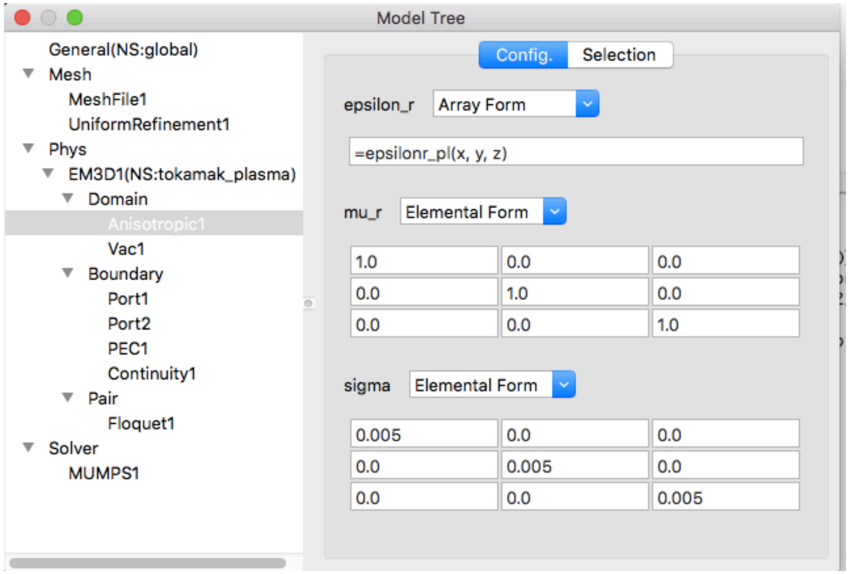
piScope/PyMFEM
PyMFEM - Python Wrapper for MFEM (Lightweight, general, scalable C++ FEM library, http://mfem.org)
github.com

2 4

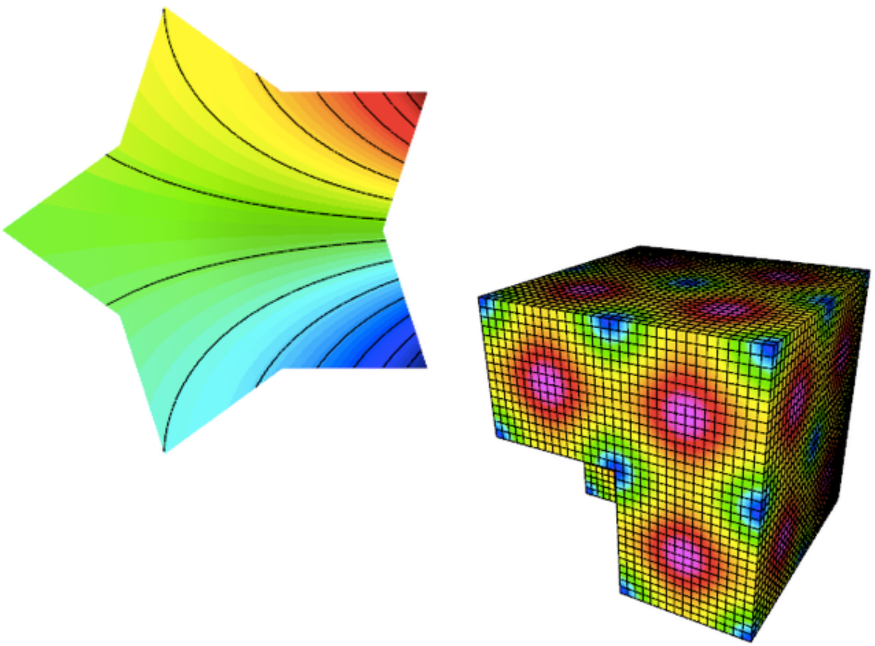
EM3D physics layer

- Solve inhomogeneous Maxwell eq. in 3D in frequency domain
  - Cartesian coordinate system
  - Time harmonics term follows the physics convention : ~exp(-iωt)

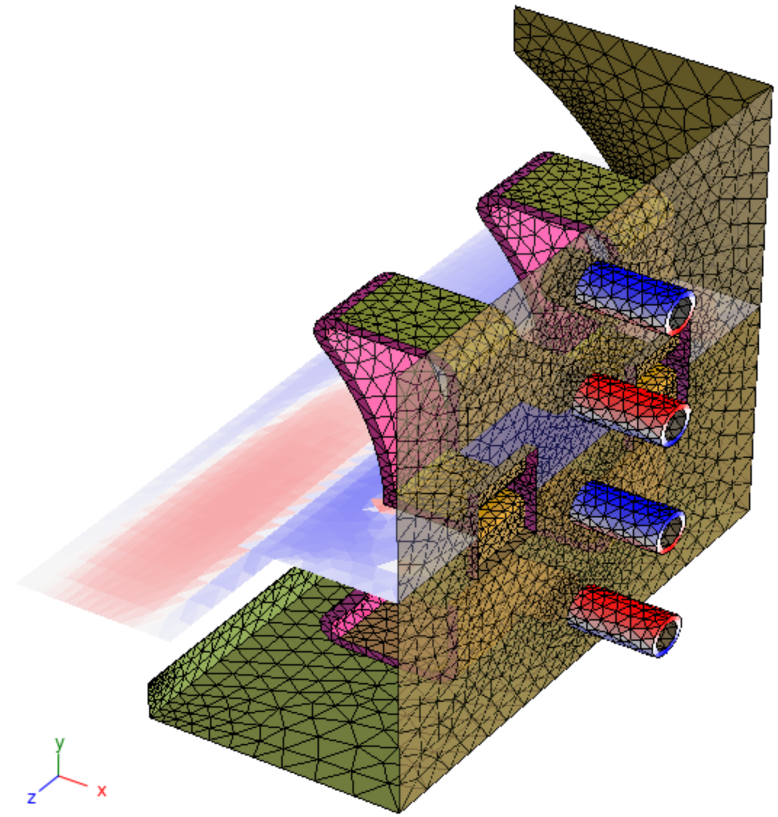
- Domain
  - Uniform dielectric media
  - Anisotropic (matrix) media
  - External J
  - DivJ constraints in vacuum
- Boundary
  - Perfect electric conductor (Et=0)
  - Perfect magnetic conductor (Bt=0)
  - Waveguide port (TE, TEM modes)
  - Periodic boundary
  - Surface current/Magnetic field/Electric field
  - Impedance Boundary (not yet implemented)



We also need a good user interface to tuckle a real world problem



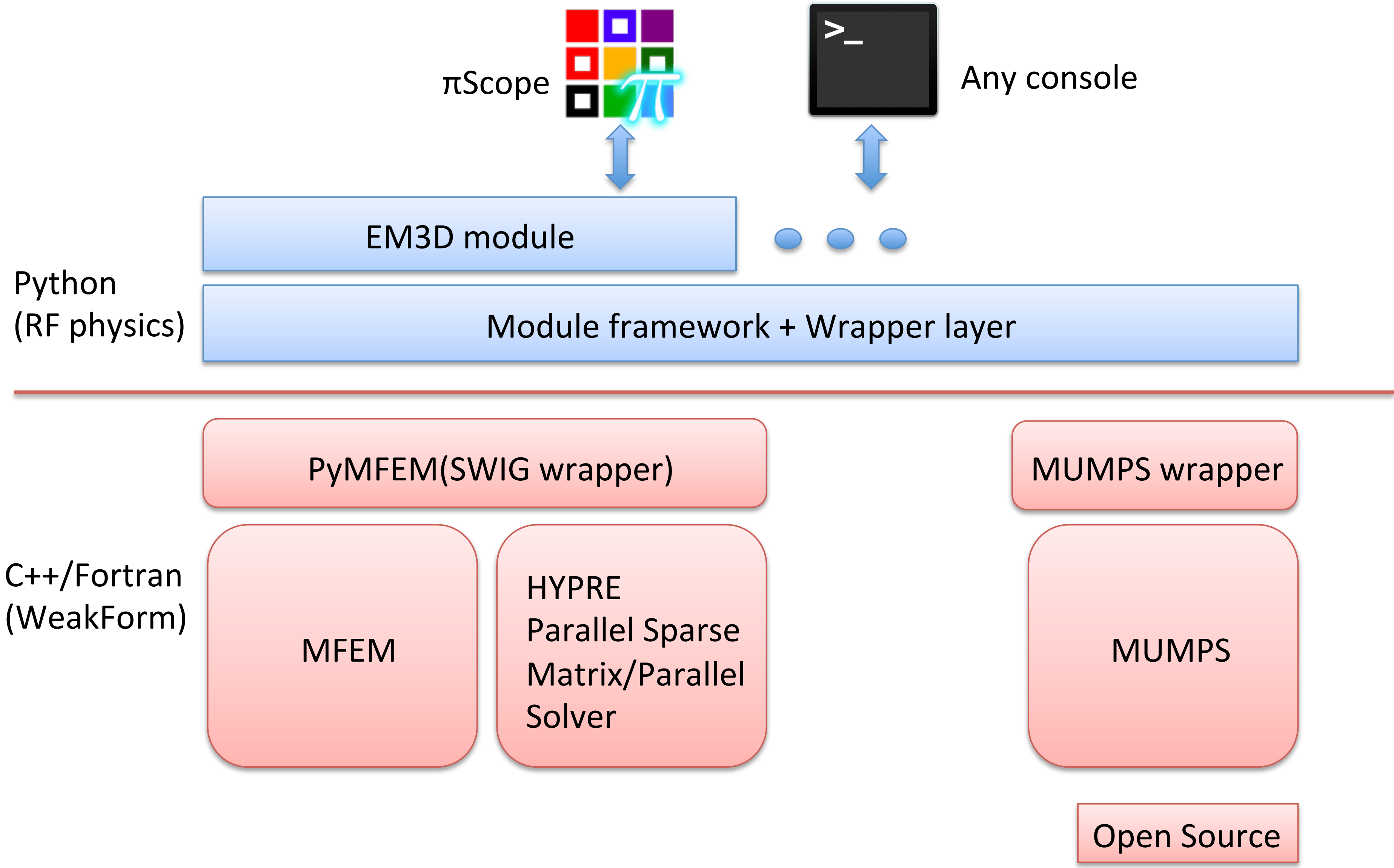
Examples on mfem.org



Our simulations

- Handle 3D geometry / mesh
- Data post-processing

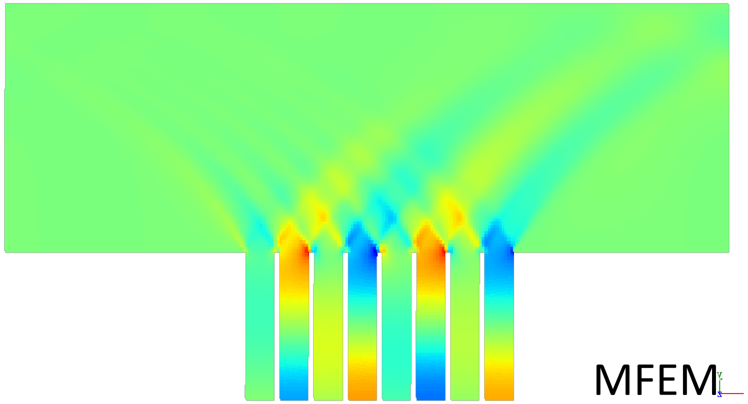
Software structure for physics module development and a user friendly model preparation is being developed



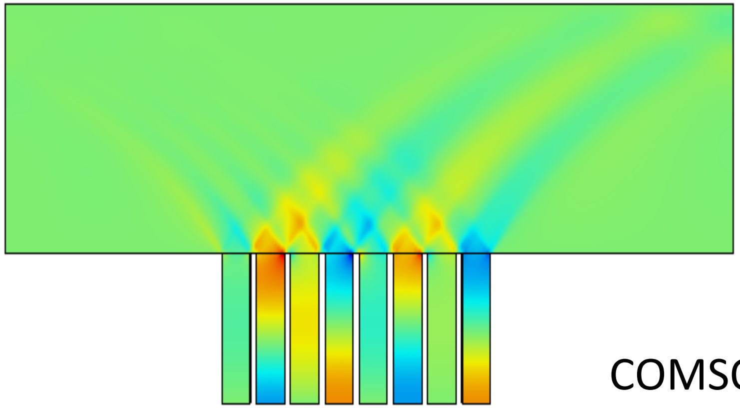
Verification with other codes: 2D lower hybrid (LH) grill launcher

- 2D stratified cold plasma model
- 8 wave guide with 90 deg phasing
- Linear density profile

Solution obtained using MFEM is nearly identical to a cold plasma model using COMSOL

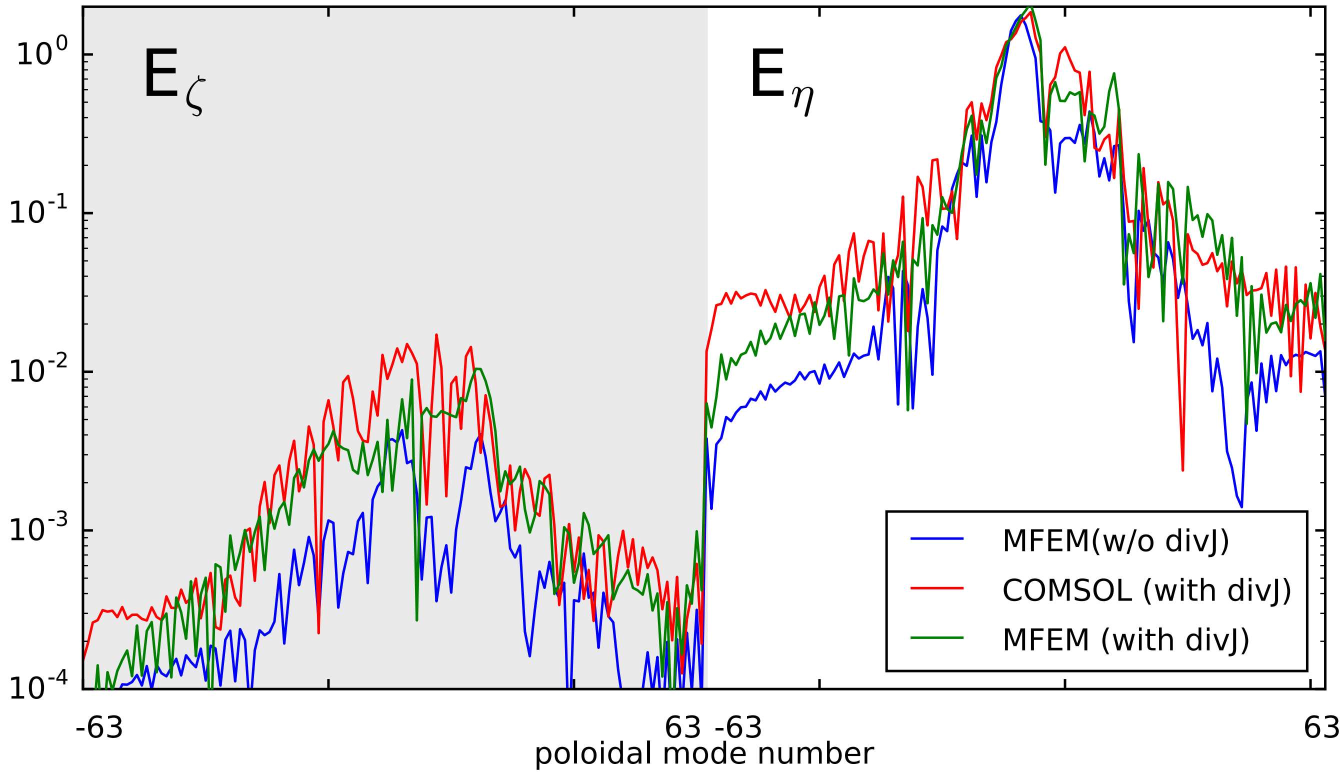


MFEM



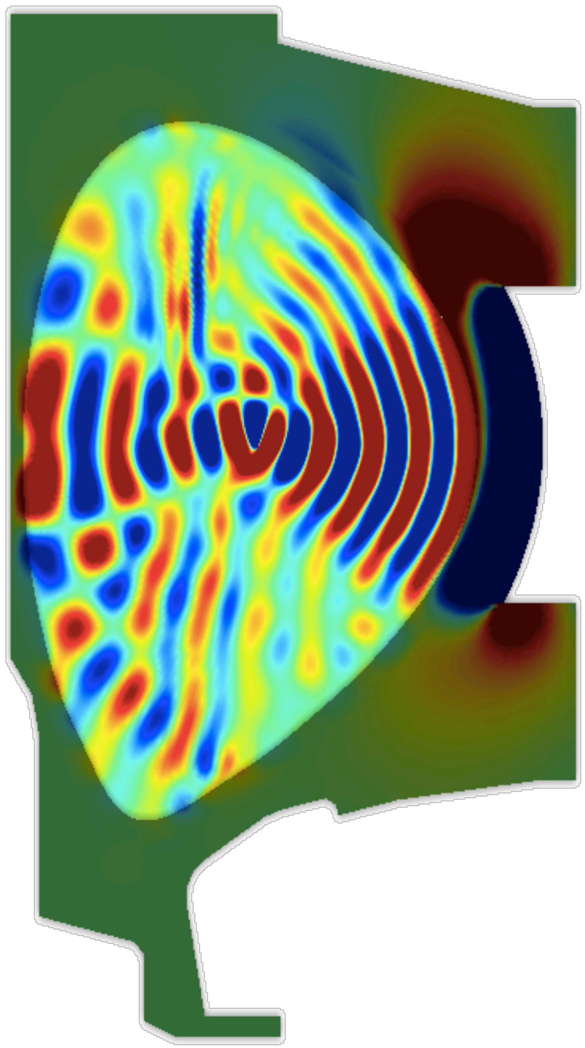
COMSOL

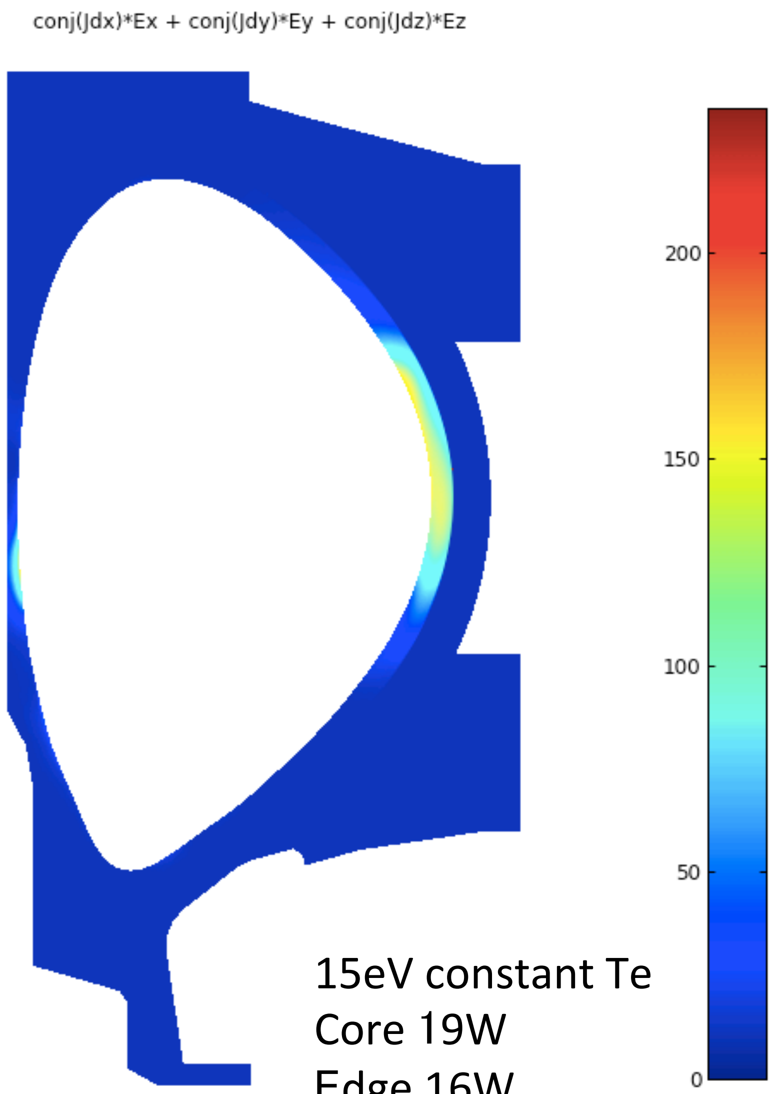
First coupling result with MFEM is on the way



- Comparison with TORIC-COMSOL coupling
- Note that MFEM using 3D geometry and different basis functions, therefore they don't need to agree with a numerical precision.

E_η (poloidal component) looks good and the evaluation of power absorption is on the way





15eV constant Te
Core 19W
Edge 16W

Summary

- Status of TORIC-FEM coupling (HIS-TORIC).
  - Proof-of-Principle using a cold plasma mode built on COMSOL is completed
  - Migration to a new edge FEM code based on MFEM library is making good progress
  - Will start modeling “real” 3D case next month.

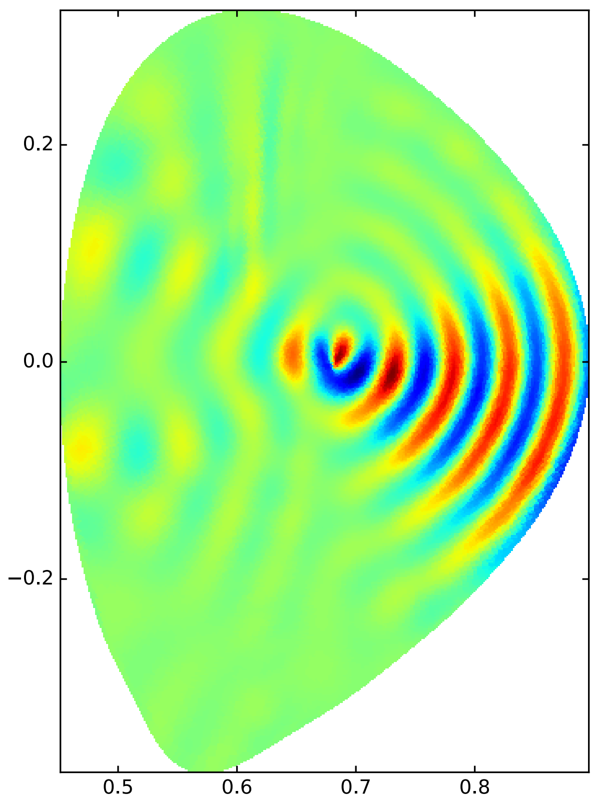
- Presentations & Collaborations
  - First paper submitted and got a very good reviewers’ comments
  - Sherwood (Shiraiwa) and RF topical conf (Wright, invited)

  - D. Green : Integration of Kinetic-J based hot plasma conductivity (FEM core)
  - C. Lau : Modeling 3D LHCD field in front of launcher (LH edge/antenna)
  - J. Myra : RF rectified sheath potential model (ICRF impurity)
  - Y. Takase : LH launcher modeling (LH core & ST)
  - N. Bertelli : HHFW on NSTC (HHFW)

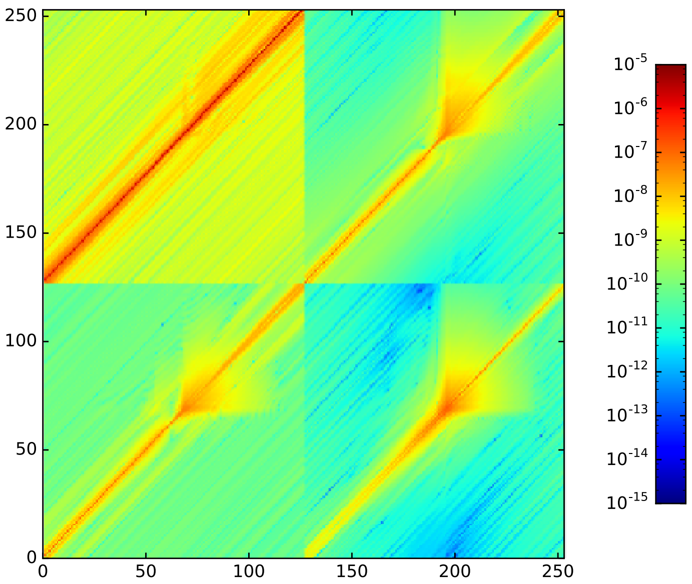
and many other possibility mentioned in a new SciDAC proposals

First coupling result with MFEM is on the way.

Reconstructed core TORIC RF field



Edge response matrix



- Reconstructed field looks reasonable.
- Reconstruction of edge is on-going
- Verification using previous TORIC-COMSOL simulations is on-going

E_η (poloidal component) looks good and the evaluation of power absorption is on the way