M3D-C1 ZOOM Meeting

05/01/2023

Upcoming Meetings

CS Issues

1. Adaptation update …. RPI
2. IDL Postprocessing error … Cesar
3. Perlmutter_cpu update
4. NERSC Time
5. Changes to github master since last meeting
6. Regression tests
7. Debug for Dingyun's application

Physics Studies

1. Magnetic boundary conditions at infinity
2. Anything else

In attendance

Steve Jardin
Hank Straus
Chang Liu
Jin Chen
Brendan Lyons
Cesar Clauser
Priyanjana Sinha
Min-Gu Yoo
Chen Zhao
Andreas Kleiner
Nate Ferraro
Dingyun Liu

Mark Shephard
Seegyoung Seol
Usman Riaz

Upcoming Meetings

Sherwood Theory	May 8-10	Knoxville, TN
EPS	July 3-7	Bordeaux, France
TSDW	July 19-21	Princeton, NJ
IAEA	Oct 16-21	London, UK
APS	Oct 30 – Nov 1	Denver, CO
AAPPS-DPP	Nov 12-17	Nagoya, JP

Adaption Update

RPI?

IDL Postprocessing error

From Cesar to Nate and myself on 4/17/23

I attach a set of slides to show how some scalars related to resistive walls are giving wrong values.
Perhaps this is related to the new wall model feature (I'm using the SPARC model).

I looked at the vertical wall force calculation and it looks fine to me. Does int1(a,b) include the "R" factor in the jaccobian (so it integrates R*dR*dZ)?

The only thing left that it is hard for me to check is the function "m3dc1_ent_getgeomclass" which assign the "izone" value to each element.

Some issues with IDL post-processing routines

- **I found wall forces and some currents to show wrong values during some VDE simulations**
- **I wrote some routines to 'integrate' quantities from fields to double check.**
  - **I loop over time slices to get scalar vs. time**
- **In next slides I show expected results compared to what M3D-C1 plot_scalar is showing.**

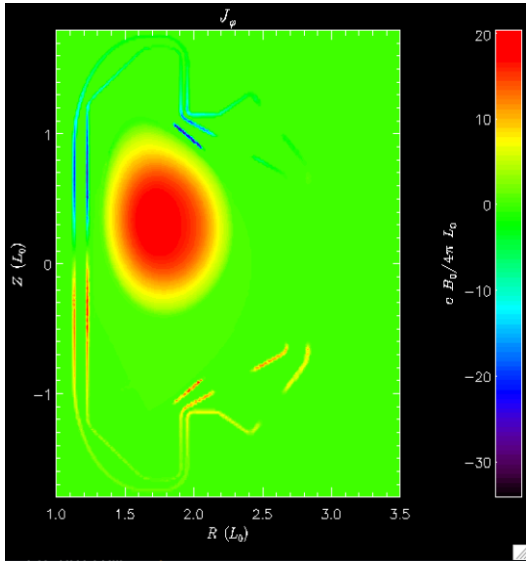
My diagnostic:
I looked at the wall force (in diagnostic.f90) and everything looks ok to me (force computation and integral).
So, the only thing left seems to be the assignment of which element is a conducting region.
This happens in
"call m3dc1_ent_getgeomclass(2, itri-1,izonedim,izone)" (L800. diagnostic.f90)
Not sure though.

Own routines to plot different fields

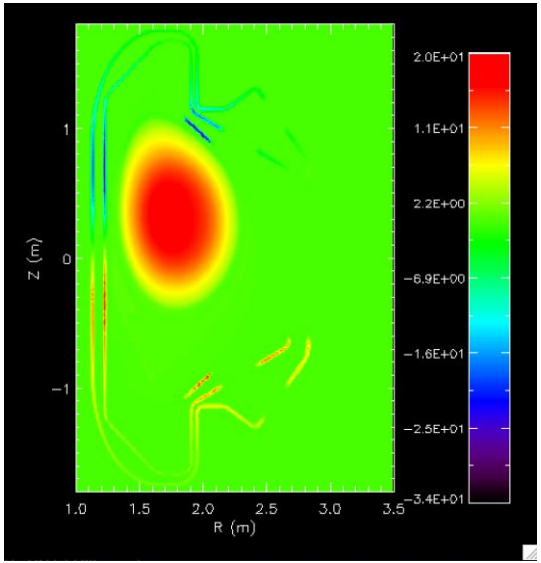
- Can plot show plasma and wall values,
- Only wall or
- Only plasma region

Example: Toroidal current density

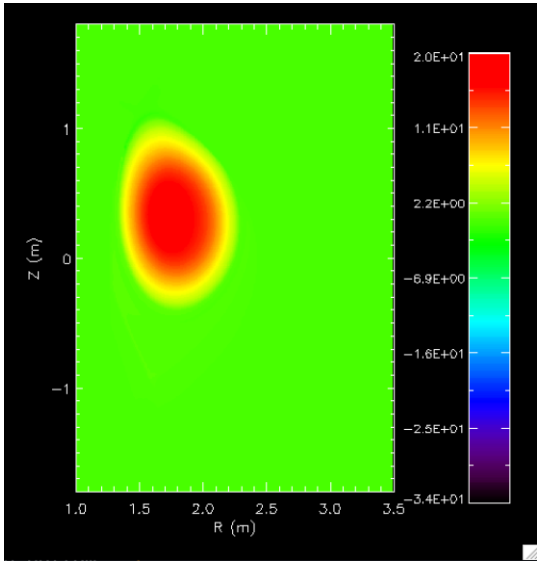
M3DC1 routine
plot_field,'Jy_plasma',…



My own plotting routine
'Jy_plasma',…



My own plotting routine (removing wall)
'Jy_plasma',…,/no_wall



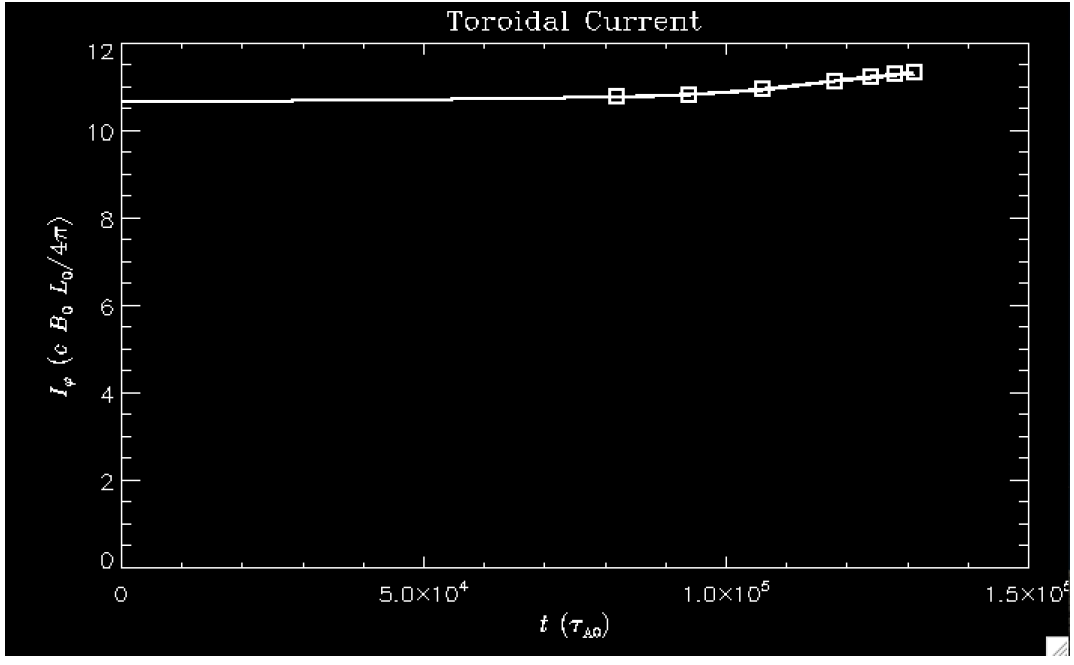
To verify agreement

Can integrate to get total toroidal current

Can “integrate” fields to calculate scalars

TEST: Toroidal current density

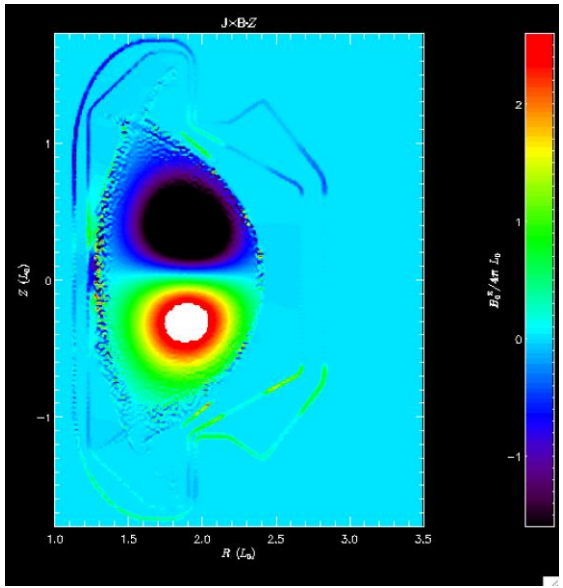
M3D-C1 plot_scalar, ‘it’ (line)
Compared to integration of *‘jy_plasma’, /no_wall* at different timeslices (squares)



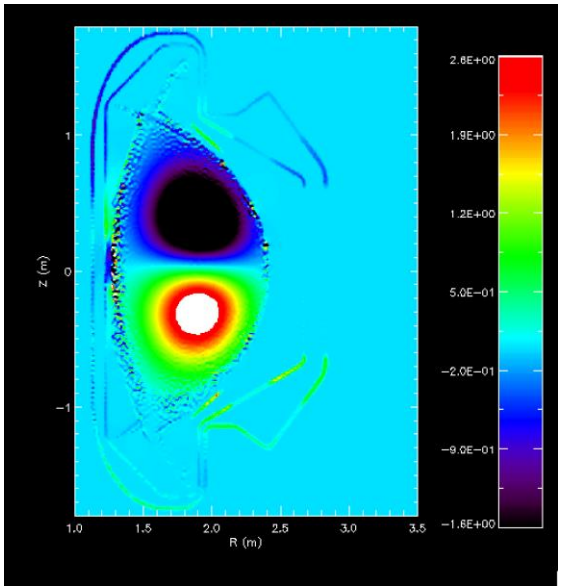
Some plot_scalar variables give wrong values

Example: Vertical force

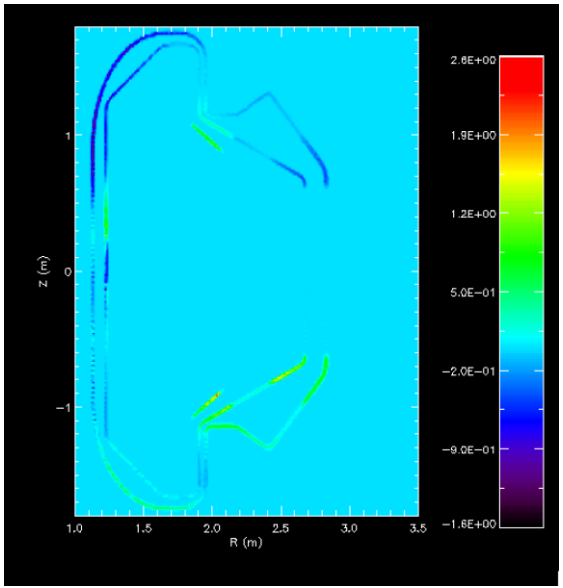
M3D-C1
plot_field,'JxB_z',…



My own plotting routine
'JxB_z',…



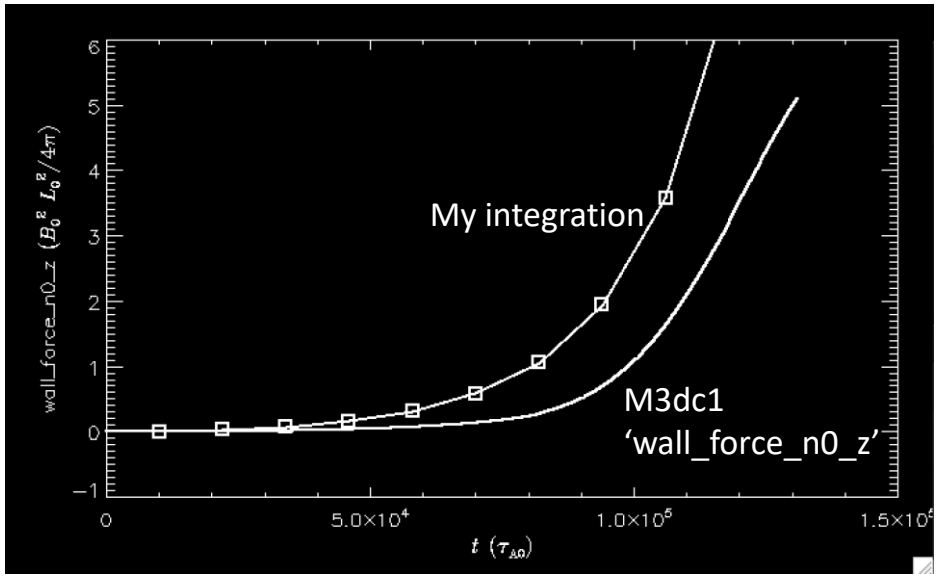
My own plotting routine (remove plasma)
'JxB_z',…,/only_wall



Verify identical agreement

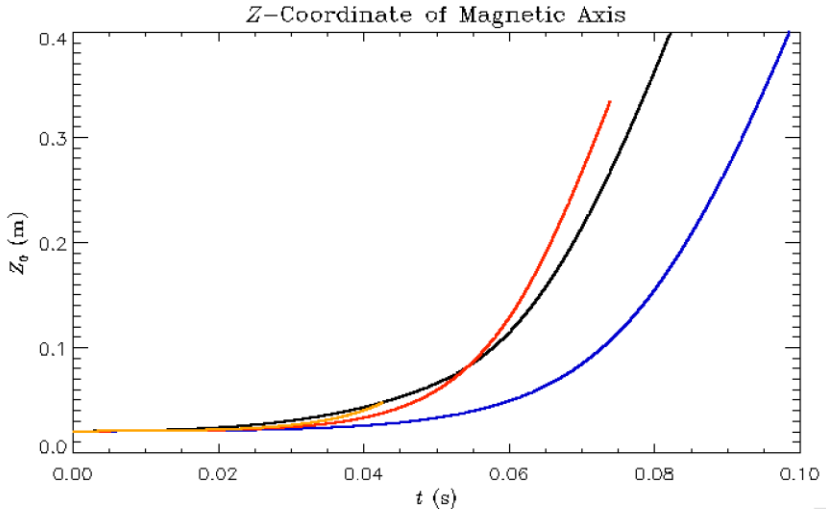
Can integrate to get total vertical force

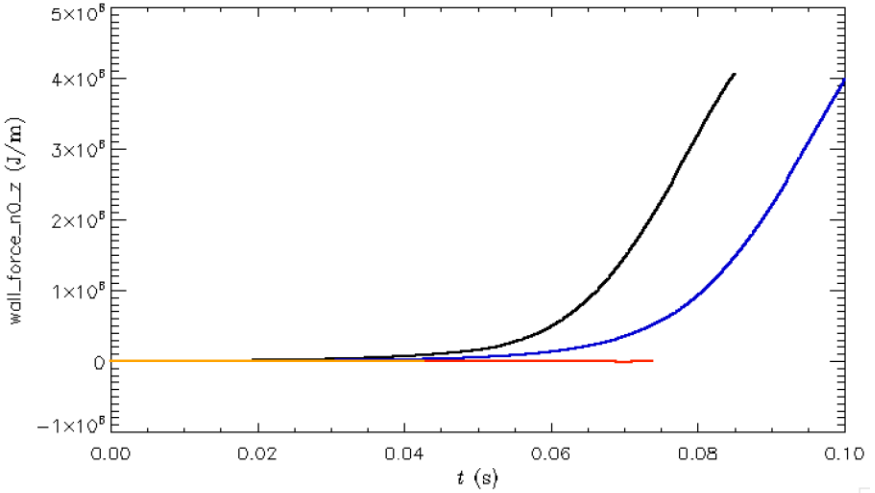
Vertical force shows significant discrepancy



Other examples when ”turning off” passive plates in SPARC by increasing their resistivity lead to almost no vertical force

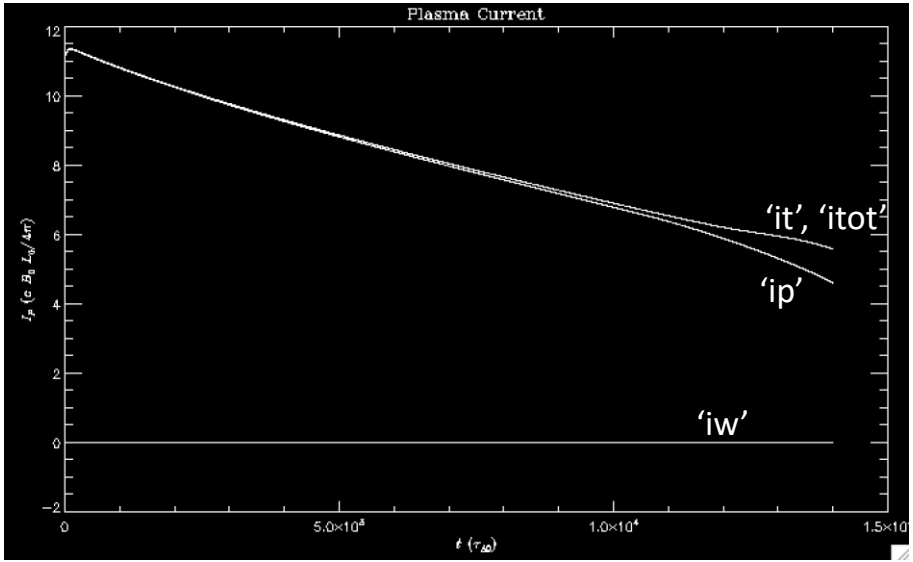
Red and yellow cases: Passive Plates with ‘vacuum’ resistivity so all the vessel forces are on inner and outer vessel. However, vertical forces seem to vanish



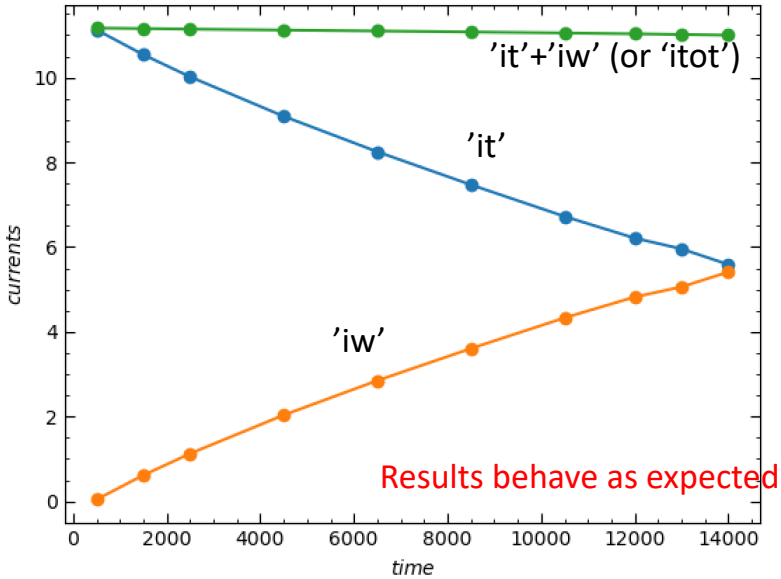


Another problem showing a CQ and wrong values for currents (at least ‘iw’ and ‘itot’)

This is a very fast current quench in a limit of a very conductive vessel.
So total current (plasma+wall), or *plot_scalar,‘itot’*, should remain constant



My integration routines



‘it’ is the same in both figures

perlmutter_cpu update

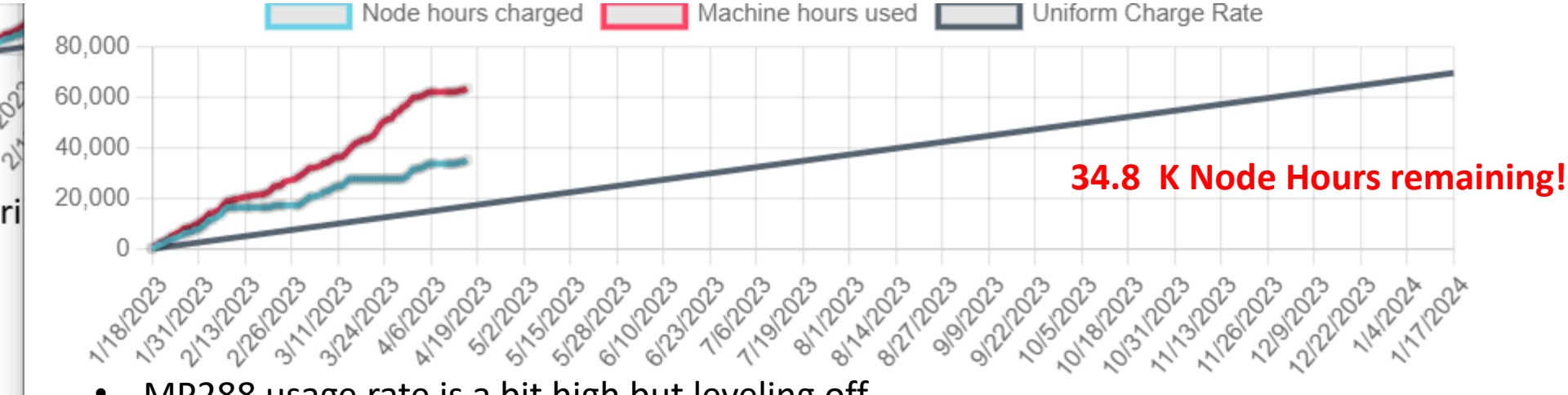
- Large jobs with 73 K vertices(N) and 98 K vertices (L) with 380 vertices/partition
  - I have now rerun these with the new 64bit PETSC: no more
  - 192-N-09j with 192 partitions, 380 v/p, 64 planes, 64 cpu/node: runs ok!
  - 192-N-10e with 192 partitions, 380 v/p, 32 planes, 96 cpu/node: oom
  - 128-K-09c with 128 partitions, 386 v/p, 64 planes, 64 cpu/node: **SUBPC Error**
  - 128-K-09d with 129 partitions, 386 v/p, 64 planes, 32 cpu/node: runs ok!
  - 256-L-09d with 256 partitions, 384 v/p, 64 planes, 64 cpu/node: **SUBPC Error**
  - 256-L-09e with 256 partitions, 384 v/p, 64 planes, 32 cpu/node: **SUBPC Error**

- Smaller meshes with 9 K vertices (B) and 19 K vertices (H)
  - Normally run ok with 128 cpu/node and 150-200 vertices/partition
  - Also run ok with 64 cpu/node and 300-400 vertices/partition
  - Too many vertices/partition gives oom error (not SUBPC)

Still not resolved, but Thanks to Jin and Seegyoung for installing the new PETSC libraries

NERSC Time 2023

mp288



34.8 K Node Hours remaining!

- MP288 usage rate is a bit high but leveling off
- Also, 8.9k GPU node hours
- Cori to go away May 31 2023, 12:00 noon PT

Changes to github master --after 2023-04-14

Jin Chen:

04/23/23: 64bit Perlmutter_cpu.mk

04/25/23: swap cray-mpich from default 8.1.25 to previous 8/1/24 used before major maintenance

04/27/23: modified matrix dump time

Local Systems

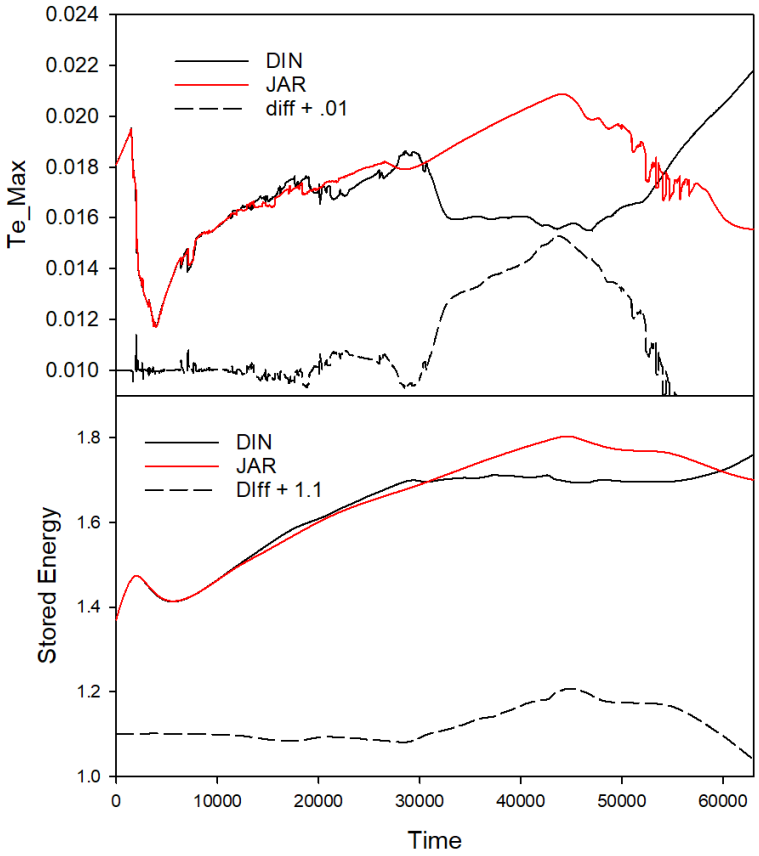
- PPPL centos7(04/28/23)
  - 7 jobs PASSED
- PPPL greene (04/28/23)
  - 5 jobs PASSED
- STELLAR (04/28/23)
  - 6 regression tests **PASSED** on stellar
  - NCSX failed due to small tolerance error
- TRAVERSE_gpu(11/04/22)
  - Compilation error (being looked at by Seegyound , Jin, and Chang)

NERSC

- Perlmutter_cpu (04/28/23)
  6 jobs **PASSED**
  NCSX failed with very small difference in C1ke

- Perlmutter_gpu (02/19/2023) **04/28/23: will not compile**
  - pellet, RMP, & RMP_nonlin, adapt all **PASSED**
  - KPRAD_2D, KPRAD_restart, NCSX all failed with very small differences
  - adapt not submitted

Dingyun is trying to reproduce a case in my 2020 paper



Need to find when these differences first occur:

git log --after 2018-06-30 > logout

git clone https://github.com/PrincetonUniversity/M3DC1

git checkout 8a6a036

Had to:

- remove wrrestartglobal from output.f90 and restart.f90
- Comment out MPI_Comm_split in newpar.f90
- Copy stellar.mk from current version

Progress on debug

Differences first appear on 11/17/20

- Mostly concerned with bf to bfp change $\mathrm{bf} = f, \quad \mathrm{bfp} = f'$

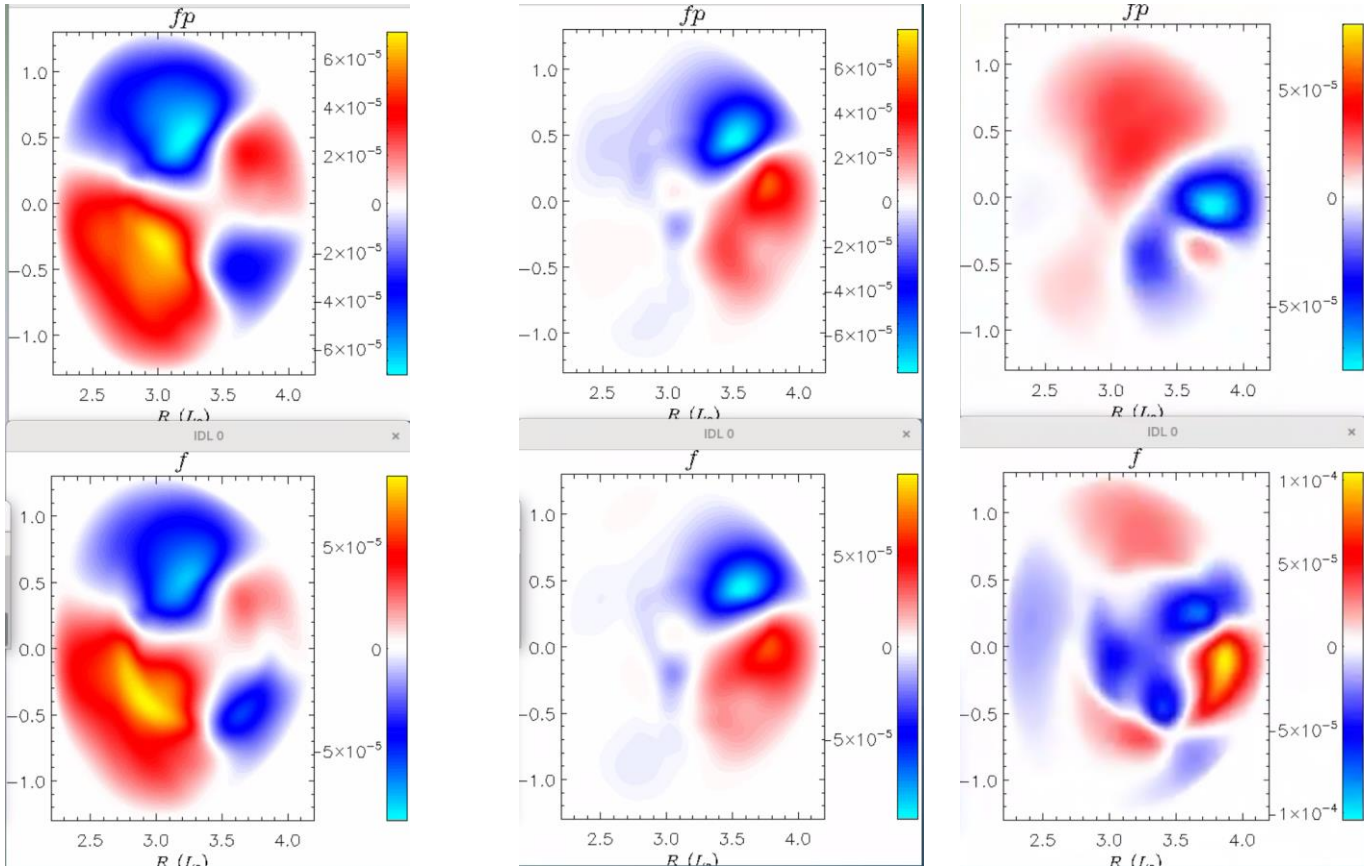
$$\mathbf{B} = \nabla\psi \times \nabla\varphi - \nabla_\perp f' + F\nabla\varphi \qquad R^2\nabla \bullet \nabla_\perp f = F - F_0$$

$$R^2\nabla \bullet \nabla_\perp f' = F'$$

- The two versions agree exactly in 2D !
- The two versions agree fairly well in 3D if $f = f' = 0$, and these results can be very different from f and f' nonzero
- Differences can appear in this and other problems at long time
- These differences likely due to differences in truncation error. Convergence studies should be performed

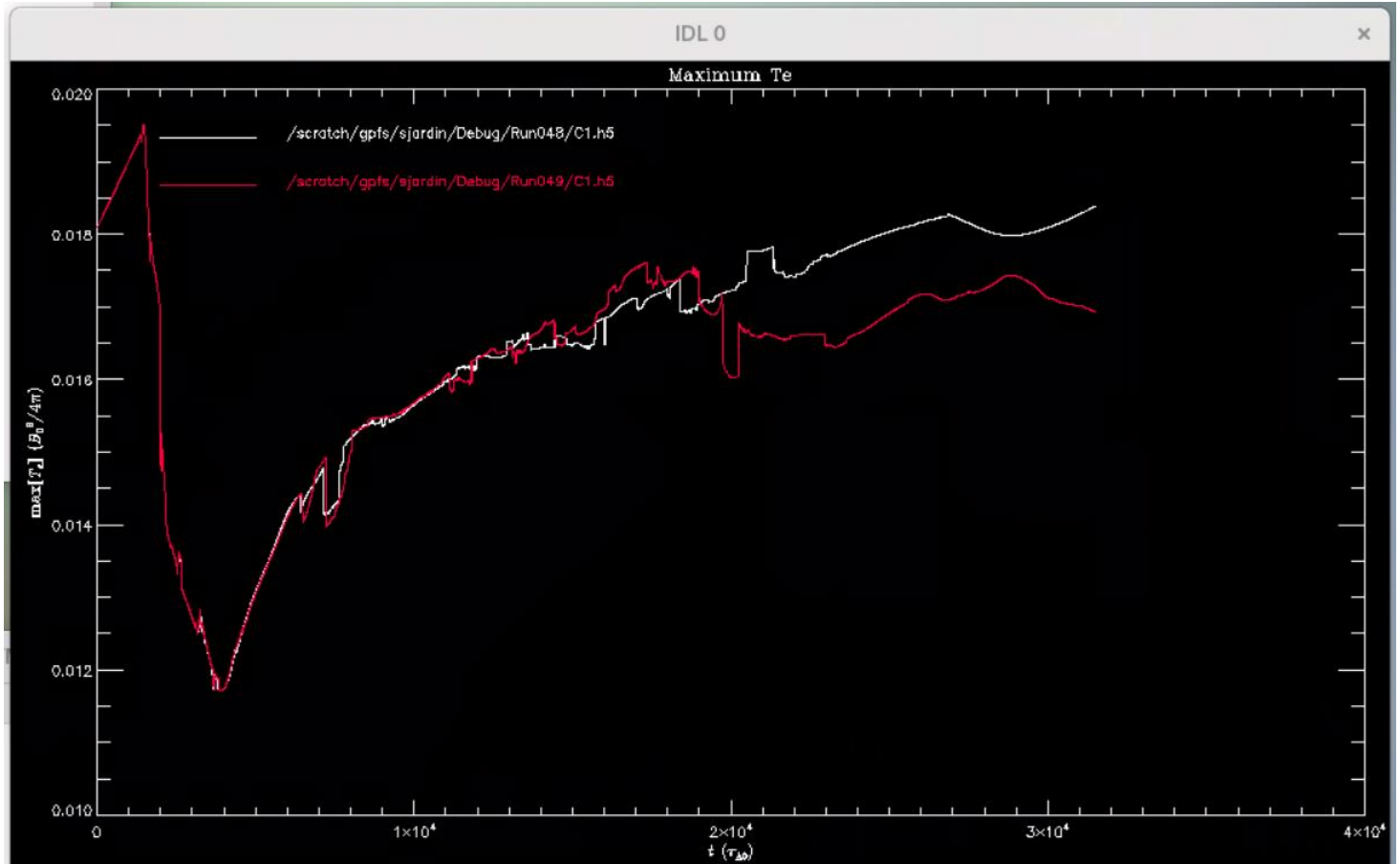
Run49, fp, op=1
11_17_20: 17:00
Commit: b572e1e7

Run48, f, op=11
11_17_20: 14:48
Commit: 43061986

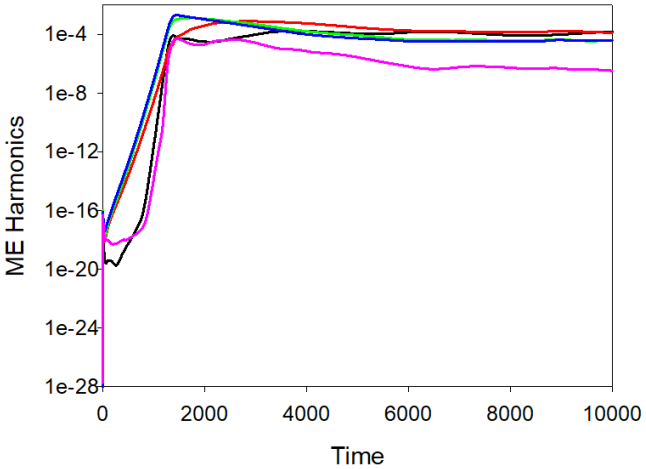


t=4500 t=9000 t=13500

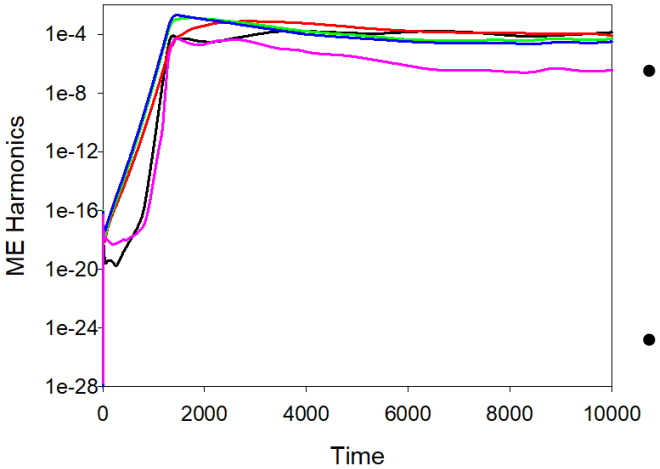
Comparison of TEMAX for versions in Run48 and Run49

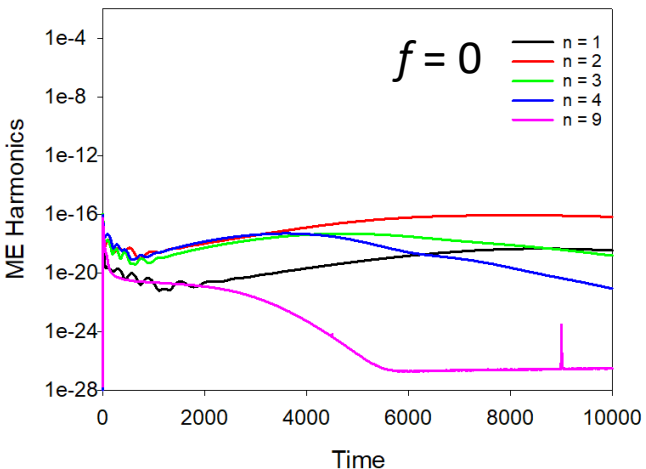


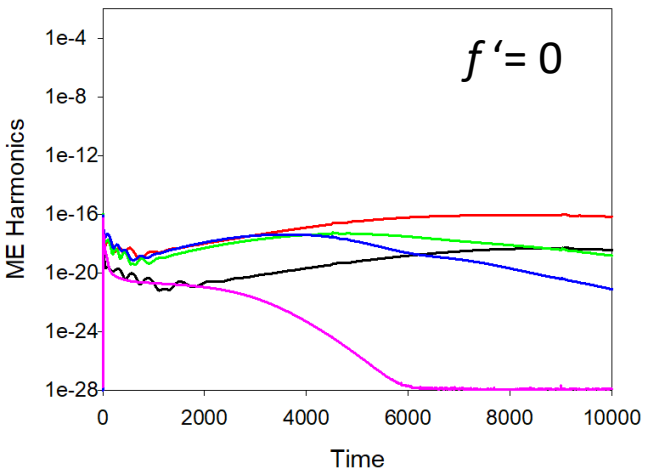
Run048 with f



Run049 with f'

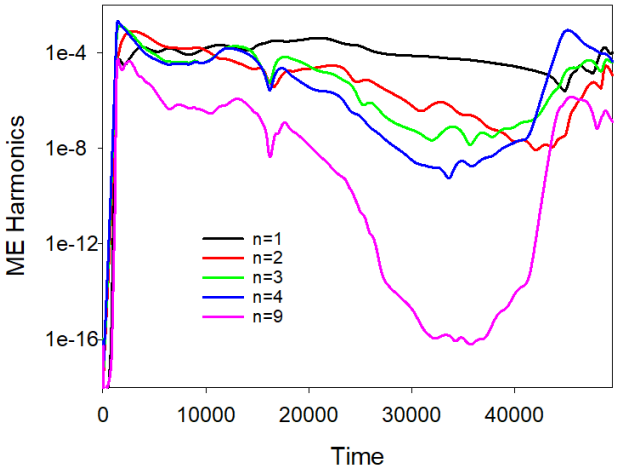


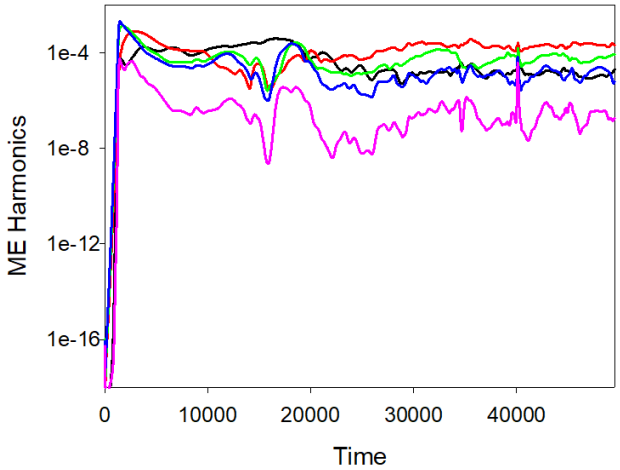




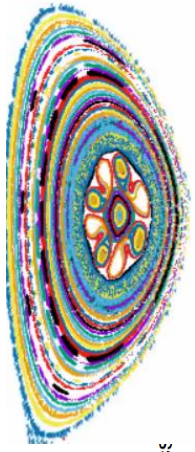
- The two runs agree in the magnetic harmonics for the first 10,000 τ_A
- Setting f (and f') to zero give very different results, no instability!

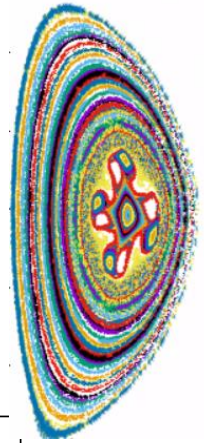
However, at longer times the magnetic harmonics differ considerably for the sawtooth run.



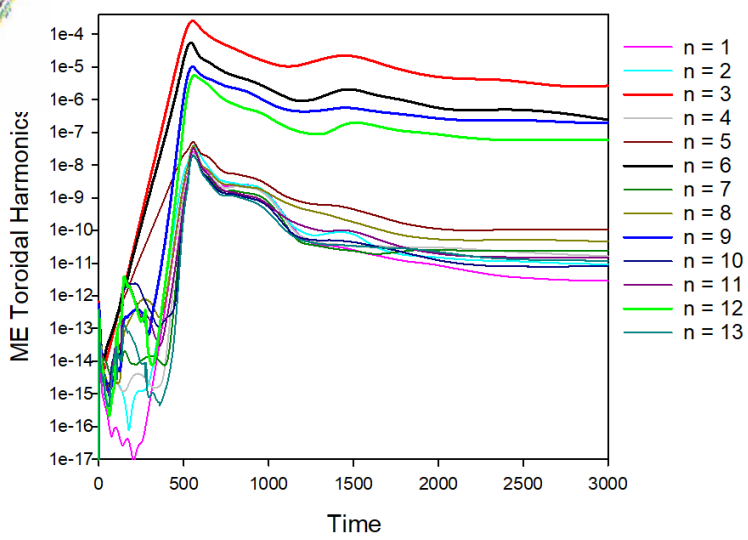


Results from infernal mode case are very similar but not identical

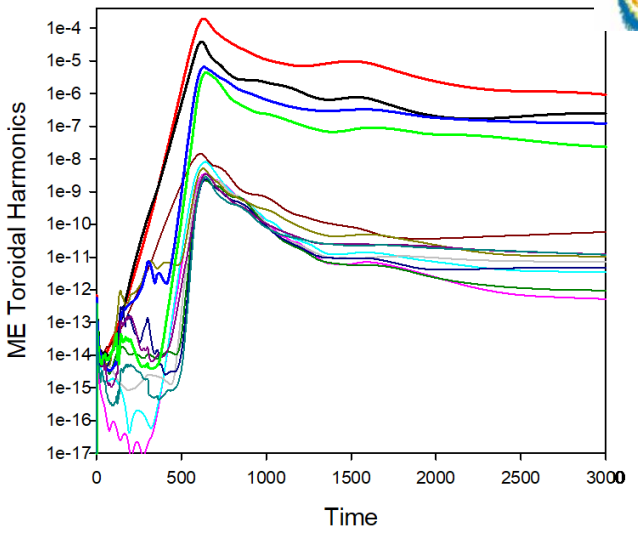




f version



f' version



Magnetic boundary conditions at infinity

- It would be useful to have the magnetic boundary conditions applied at infinity instead of at the computational boundary
- As a step in this direction, I propose to update Ψ on the computational boundary each timestep for 2D (axisymmetric) runs
- This would allow the computation of a conductors inductance (from a L/R test) and would also increase the accuracy of VDE caculations

Update Ψ on the computational boundary each timestep

$$\Psi(\vec{x}_b,t)=\frac{\mu_0}{2\pi}\iint_p G(\vec{x}_b,\vec{x})J_\phi(\vec{x},t)d^2\vec{x}+\sum_{i=1}^{N_c}G(\vec{x}_b,\vec{x}_i)I_i(t)$$

$$G(\vec{x}_b,\vec{x})=G(\vec{x}_b,\vec{x}_0)+(\vec{x}-\vec{x}_0)\bullet\nabla G(\vec{x}_b,\vec{x})\Big]_{x_0}$$
$$+\tfrac{1}{2}(\vec{x}-\vec{x}_0)(\vec{x}-\vec{x}_0):\nabla\nabla G(\vec{x}_b,\vec{x})\Big]_{x_0}+\ldots$$

$$I(t)=\iint_p J_\phi(\vec{x},t)d^2\vec{x}\qquad \vec{x}_0(t)=\frac{1}{I(t)}\iint_p \vec{x}\,J_\phi(\vec{x},t)d^2\vec{x}$$

$$\mathrm{K}(t)=\iint_p\left[\vec{x}-\vec{x}_0(t)\right]\left[\vec{x}-\vec{x}_0(t)\right]J_\phi(\vec{x},t)d^2\vec{x}$$

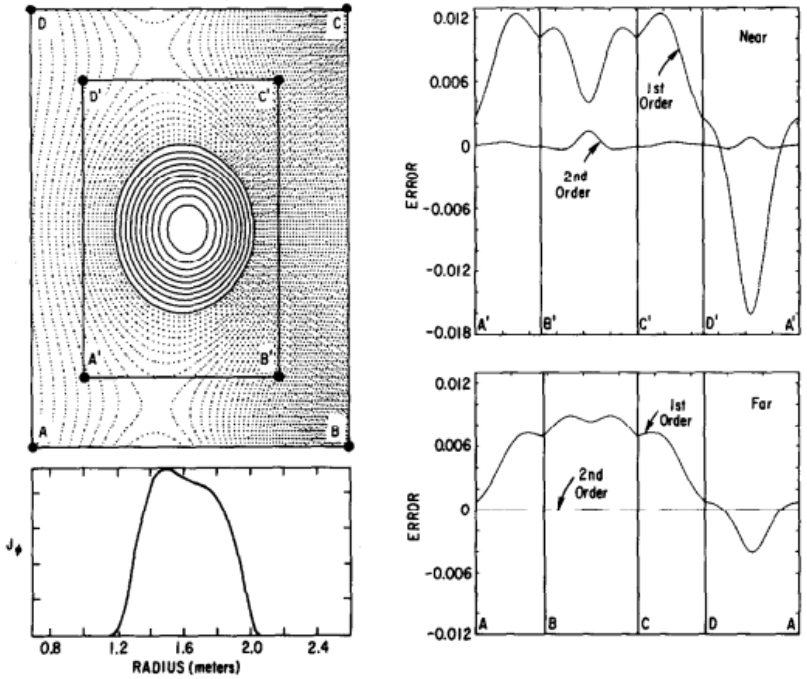
$$\Psi(\vec{x}_b,t)\simeq\frac{\mu_0}{2\pi}\left(G\left[\vec{x}_b,\vec{x}_0(t)\right]I(t)+\tfrac{1}{2}\mathrm{K}(t):\nabla\nabla G(\vec{x}_b,\vec{x})\Big]_{x_0(t)}\right)+\sum_{i=1}^{N_c}G(\vec{x}_b,\vec{x}_i)I_i(t)$$

- In 2D version only
- This will allow calculation of L/R time and thus L of conductors

Error in using greens function expansion vs total integral

$$\Psi_1(\mathbf{x}_b) = \frac{\mu_0}{2\pi} G(\mathbf{x}_b, \mathbf{x}_0)\, I(t), \tag{71}$$

$$\Psi_2(\mathbf{x}_b) = \Psi_1(\mathbf{x}_b) + \frac{1}{2}\frac{\mu_0}{2\pi}\, \mathbf{K}: \nabla\nabla G(\mathbf{x}_b, \mathbf{x})|_{\mathbf{x}_0}, \tag{72}$$



That’s All I have

Anything Else ?