Working Notes

July 6, 2020

MEMS:testingclass.py

- two ways to get fields: solver.getselfe(), solver.getphi() and wp.getselfe(), wp.getselfe(). So far seem identical.

- when plotting in python, getting fields will give the values at endpoints on grid giving odd zero-contours. Be aware of this since gist graphics won’t show the same.

- so far python graphics haven’t matched with warp graphics. This is concerning since I’m pulling fields straight from warp. Example graphics below. Setup is as follows:

Two washers were created with 10 KV and -10 KV settings. Dimensions:

rmin=1mm

rmax=1.5mm

length= 2mm

z-center for washer 1 = 3mm

z-center for washer 2 = 8mm

Mesh details:

- x,y-extent = [-4mm, 4mm] with nx,ny = 100, 110. The different settings helps when slicing potential and electric fields arrays to be sure indexing is right.

- z-extent = [0m, 12mm] with nz =250

- solvergeom = XYZgeom

- boundary conditions = Dirichlet

- solver = MRBlock3D

Plots:

A close up of a logo

Description automatically generated

A close up of a logo

Description automatically generatedA close up of a logo

Description automatically generated

for the warp plot, green lines are Ex and blue correspond to the potential.

The red lines on the python plots are (left Ex, right potential) are the zero contours. How to do this is useful to right down.

As an example

fig, ax = plt.subplots()

cp = ax.contour()

cp is an object with a lot in it. The levels can be found by using

cp.levels

To find the zero contour np.where can be used. This returns a list of arrays and will require a double index for grabbing one index. For example

#--Find zero index

zeroindex = np.where(cp.levels == 0) #will be a list of arrays where this is true

zeroindex = zeroindex[0][0] #gives integer value of where the contour is zero

#--Set zero contour color and linestyle using collections

cp.collections[zeroindex].set\_linestyle(‘dashed’)

cp.collections[zeroindex].set\_color(‘r’)

Keep in mind this doesn’t change the color bar label. I’ll have to figure this out.

The warp plot is able to fill the conductor object with a color for visualization. Maybe there is a way to pull this data from warp and integrate with python.

--Update

Indexing!!! I was working in r-z geometry (a side effect of indexing with ORISS arrays for so long). The previous code was using wp.w3d.ymesh[1] thinking this was one grid cell above 0. This was actually one grid cell above -4mm. Thanks for Steve to pointing this out in about 2 minutes.

The code has been updated to find the 0 index as follows

#--Find index where y=0

#check existence

if (0 in wp.w3d.ymesh):

yindex = np.where(wp.w3d.ymesh==0)[0][0]

#if not find closest positive grid point

else:

yindex = np.where(wp.w3d.ymesh>0)[0][0]

With this addition the following python plots are produced: A close up of a logo

Description automatically generatedA close up of a logo

Description automatically generated

July 7, 2020

- Drawxy() is not specified for conductors resulting in a “Draw not implemented for \_\_\_” error message. A workout around will be to plot zero contours.

I attempted this first by finding where the electric field in x and y was zero. This made odd plots.

A screenshot of a computer

Description automatically generated

The code to accomplish this:

#--Plot conductors in xy using zero contours

# Find z-index where first conductor lives

x = wp.w3d.xmesh

y = wp.w3d.ymesh

z = wp.w3d.zmesh

zindexrange = np.where((z>2.5\*mm) & (z<3.5\*mm))

zindex = zindexrange[0][int(len(zindexrange)/2)] #take center point of range

Ex = wp.getselfe(comp='x')[1:-1, 1:-1, zindex]

Ey = wp.getselfe(comp='y')[1:-1, 1:-1, zindex]

E = Ex + Ey

X,Y = np.meshgrid(y[1:-1], x[1:-1])

fig, ax = plt.subplots()

ax.set\_xlabel('x [mm]')

ax.set\_ylabel('y [mm]')

ax.set\_title(r'$E\_x + E\_y$')

ax.set\_xlim(-5, 5)

ax.set\_ylim(-5, 5)

contx = ax.contourf(X/mm, Y/mm, E, levels=50)

#--Find 0 contour

if (0 in contx.levels):

zeroindex = np.where(contx.levels==0)[0][0]

else:

print("Zero contour doesn't exist")

contx.collections[zeroindex].set\_linestyle('dashed')

contx.collections[zeroindex].set\_color('r')

plt.tight\_layout()

plt.show()

Plotting the potential gives a better result:

A close up of a logo

Description automatically generated

The code to produce this:

V = 10\*kV

#--Plot conductors in xy using zero contours

# Find z-index where first conductor lives

x = wp.w3d.xmesh

y = wp.w3d.ymesh

z = wp.w3d.zmesh

zindexrange = np.where((z>2.5\*mm) & (z<3.5\*mm))

zindex = zindexrange[0][int(len(zindexrange)/2)] #take center point of range

potential = wp.getphi()[1:-1, 1:-1, zindex]

fig, ax = plt.subplots()

ax.set\_xlim(-5,5)

ax.set\_ylim(-5,5)

ax.set\_xlabel('x [mm]')

ax.set\_ylabel('y [mm]')

ax.set\_title(r'$\Phi(x,y)$')

X,Y = np.meshgrid(y[1:-1], x[1:-1])

cont = ax.contourf(X/mm, Y/mm, potential, levels=50)

if (V in cont.levels ):

index = np.where(cont.levels == V)[0][0]

else:

print("V not in levels")

cont.collections[-1].set\_linestyle('dashed')

cont.collections[-1].set\_color('r')

plt.tight\_layout()

plt.show()

Oddly, the index routine gives index=50. When running cont.collections[50] this index is out of range which is why [-1] index is used.