KEMP User Guide & Tutorials

Draw field profiles

Contributors: Min Gon Lee and SeokJae Yoo
Last Update: 09/25/2015

Draw Field Profiles

Contents

Draw Field Profiles

1) Basic Setting

```
import numpy as np
import h5py as h5
from KEMP import Basic_FDTD, Dielectric, structures, to_epr, to_SI, to_NU
c = 299792458.
nm = 1e-9
```

- 확장자명 ".py"로 된 python 파일을 만든다
  - Windows: 메모장 혹은 python 파일을 편집할 수 있는 프로그램 사용
  - Linux: vi editor 이용하여 편집
- Python 파일은 위 코드로 시작된다
  - Numpy: 수학 계산
  - h5: 데이터 추출
  - KEMP: KEMP에서 Electric field 측정에 필요한 module을 로드
  - 자주 사용하는 상수나 단위를 정의 (광속 c, 길이 nm)

Draw Field Profiles

2) Space Setting

```
dx, dy, dz = [5*nm, 5*nm, 5*nm]
nx, ny, nz = [200, 2, 600]
x = np.ones(nx, dtype=np.float64)*dx
y = np.ones(ny, dtype=np.float64)*dy
z = np.ones(nz, dtype=np.float64)*dz
space_grid = (x, y ,z)
lx, ly, lz = dx*nx, dy*ny, dz*nz
fdtd = Basic_FDTD('3D', space_grid, dtype=np.complex64, engine='nvidia_cuda')
pml_apply = {'x':'+-' , 'y':''  , 'z':'+-'  }
pbc_apply = {'x':False, 'y':True, 'z':False}
fdtd.apply_PML(pml_apply)
fdtd.apply_PBC(pbc_apply)
```

- dx, dy, dz: 한 칸의 크기
- nx, ny, dz: 총 시뮬레이션 공간의 칸 개수
  - Ex) nx=100이라면 x는 0~100까지 100+1 칸이 들어간다

Draw Field Profiles

2) Space Setting

```
dx, dy, dz = [5*nm, 5*nm, 5*nm]
nx, ny, nz = [200, 2, 600]
x = np.ones(nx, dtype=np.float64)*dx
y = np.ones(ny, dtype=np.float64)*dy
z = np.ones(nz, dtype=np.float64)*dz
space_grid = (x, y ,z)
lx, ly, lz = dx*nx, dy*ny, dz*nz
fdtd = Basic_FDTD('3D', space_grid, dtype=np.complex64, engine='nvidia_cuda')
pml_apply = {'x':'+-' , 'y':''   , 'z':'+-'  }
pbc_apply = {'x':False, 'y':True, 'z':False}
fdtd.apply_PML(pml_apply)
fdtd.apply_PBC(pbc_apply)
```

- 여기서는 nvidia_cuda를 통해 빠르게 계산하였다. 막약 intel cpu를 이용하고 싶다면 fdtd 부분을 engine='intel_cpu'로 바꾸어주면 된다.

Draw Field Profiles

2) Space Setting

```
dx, dy, dz = [5*nm, 5*nm, 5*nm]
nx, ny, nz = [200, 2, 600]
x = np.ones(nx, dtype=np.float64)*dx
y = np.ones(ny, dtype=np.float64)*dy
z = np.ones(nz, dtype=np.float64)*dz
space_grid = (x, y ,z)
lx, ly, lz = dx*nx, dy*ny, dz*nz
fdtd = Basic_FDTD('3D', space_grid, dtype=np.complex64, engine='nvidia_cuda')
pml_apply = {'x':'+-' , 'y':''   , 'z':'+-'  }
pbc_apply = {'x':False, 'y':True, 'z':False}
fdtd.apply_PML(pml_apply)
fdtd.apply_PBC(pbc_apply)
```

- pml_apply: PML 설정
  - Electric field의 polarized 방향(y)을 제외한 x, z축에 PML 설정 ('x':'+-', 'z':'+-')
- pbc_apply: PBC 설정
  - y축에 PBC 설정 (True는 PBC 설정, False는 PBC 미지정)

Draw Field Profiles

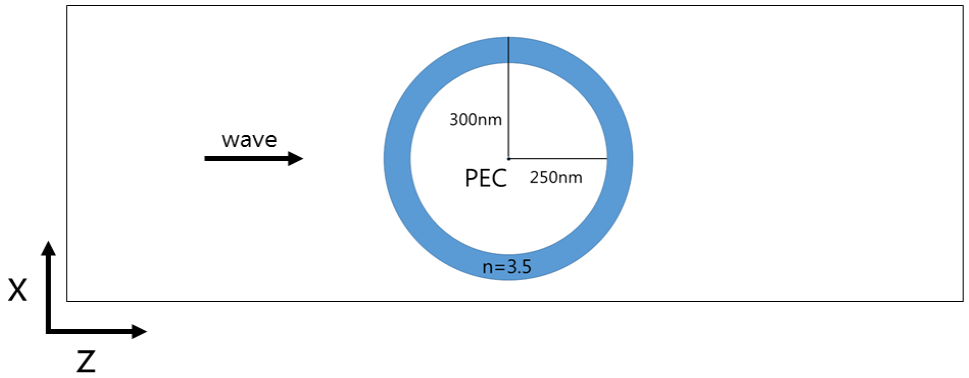
2) Space Setting

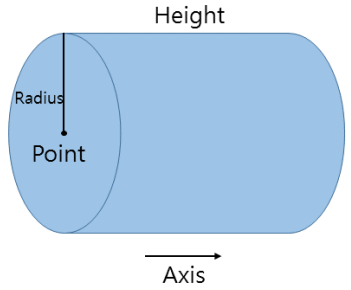
```
dx, dy, dz = [5*nm, 5*nm, 5*nm]
nx, ny, nz = [200, 2, 600]
x = np.ones(nx, dtype=np.float64)*dx
y = np.ones(ny, dtype=np.float64)*dy
z = np.ones(nz, dtype=np.float64)*dz
space_grid = (x, y ,z)
lx, ly, lz = dx*nx, dy*ny, dz*nz
fdtd = Basic_FDTD('3D', space_grid, dtype=np.complex64, engine='nvidia_cuda')
pml_apply = {'x':'+-' , 'y':''   , 'z':'+-'  }
pbc_apply = {'x':False, 'y':True, 'z':False}
fdtd.apply_PML(pml_apply)
fdtd.apply_PBC(pbc_apply)
```

- Tip
  - y 축으로 PBC가 들어가므로, 빠른 계산을 위해 ny 값을 낮게 잡음.
  - PML은 한 방향으로 10칸씩 들어가므로, 위 코드에서는 x축으로 0~9칸, 191~200칸 그리고 z축으로 0~9칸, 591~600칸에 PML이 설정된다.

Draw Field Profiles

3) Structure Setting



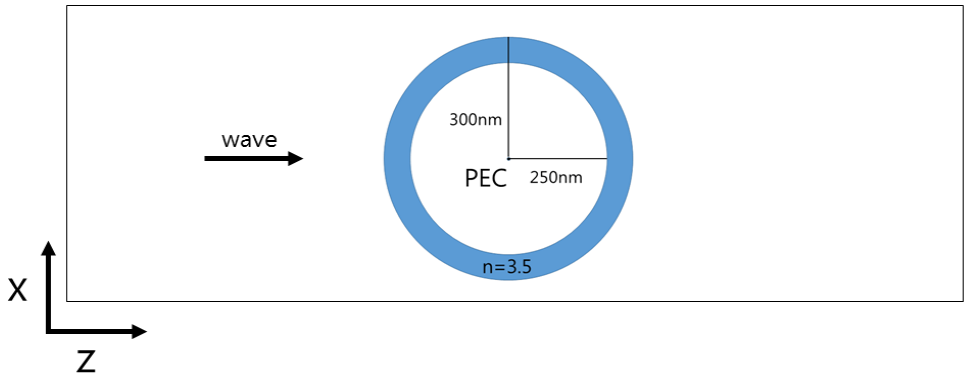


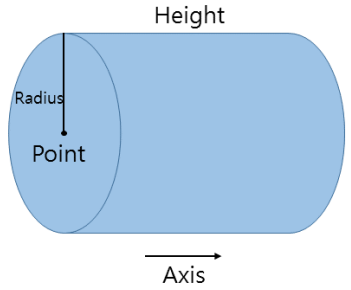

```
diel1 = Dielectric(to_epr(fdtd, n=1.e20))
diel2 = Dielectric(to_epr(fdtd, n=3.5))
cyl1=structures.Cylinder(diel1,(lx/2,0.*nm,lz/2), 250.*nm, 100.*nm, 'y')
cyl2=structures.Cylinder(diel2,(lx/2,0.*nm,lz/2), 300.*nm, 100.*nm, 'y')
fdtd.set_structures([cyl2,cyl1])
```

- 이번에 사용할 두 원형 구조는 반지름이 각각 250nm, 300nm이며 작은 원은 PEC(n=∞)이며 큰 원은 n=3.50이다.
- 원형 구조를 만들기 위해 KEMP의 structure 구현 기능 중 Cylinder를 사용한다.

Draw Field Profiles

3) Structure Setting



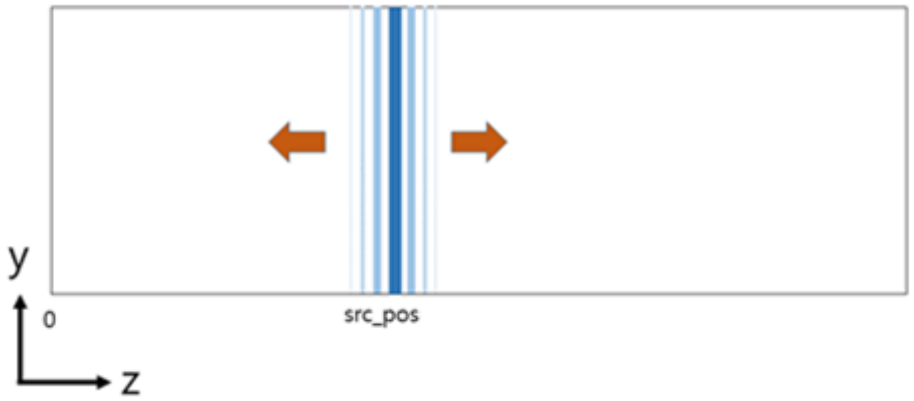



```
diel1 = Dielectric(to_epr(fdtd, n=1.e20))
diel2 = Dielectric(to_epr(fdtd, n=3.5))
cyl1=structures.Cylinder(diel1,(lx/2,0.*nm,lz/2), 250.*nm, 100.*nm, 'y')
cyl2=structures.Cylinder(diel2,(lx/2,0.*nm,lz/2), 300.*nm, 100.*nm, 'y')
fdtd.set_structures([cyl2,cyl1])
```

- structures.Cylinder(물질, 밑면의 원점, 밑면의 반지름, 높이, '높이 축')로 구조를 만든다. 밑면의 원점은 (x, y, z)의 array, 밑면의 반지름과 높이는 SI unit(meter 단위), axis는 높이에 해당되는 축을 적으면 된다. fdtd.set_structures를 통해 위에서 만든 구조를 적용한다.

Draw Field Profiles

4) Wave Setting

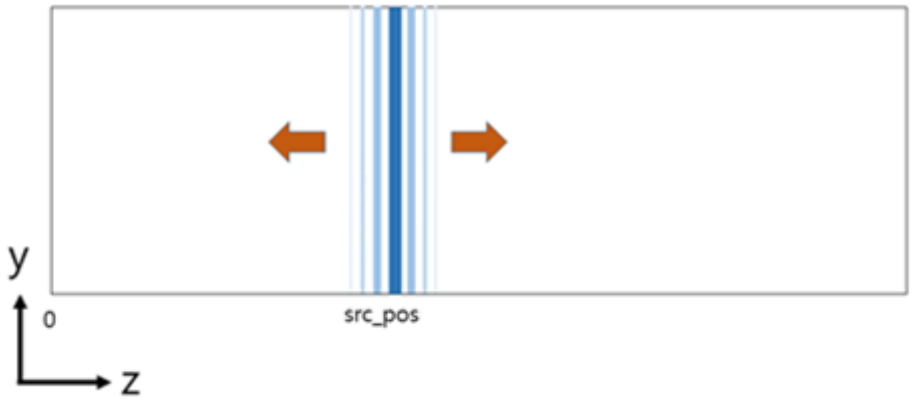



```
wavelength = 600*nm
freq = c/wavelength
wfreq_NU = to_NU(fdtd, 'angular frequency', 2.*np.pi*freq)
src_pos=60
inc    = fdtd.apply_direct_source('ey', ((1,0,src_pos),(-2,-1,src_pos)))
```

- Sine wave를 사용하기 위해 wavelength를 설정하고, 그에 해당되는 frequency를 계산한다. FDTD에서 계산하기 위해 frequency를 위와 같이 angular frequency의 Natural unit으로 바꾸어준다.

Draw Field Profiles

4) Wave Setting




```
wavelength = 600*nm
freq = c/wavelength
wfreq_NU = to_NU(fdtd, 'angular frequency', 2.*np.pi*freq)
src_pos=60
inc    = fdtd.apply_direct_source('ey', ((1,0,src_pos),(-2,-1,src_pos)))
```

- Plane Wave를 만들기 위해, Wave source의 z축 위치(src_pos)를 정해주고 fdtd.apply_direct_source로 xy-planewave를 적용한다. 여기서 xy 평면파를 x축으로 1칸씩 여유를 두고 넣어준 이유는, KEMP에서 PML 속으로 Source를 줄 때 최소 한 칸은 남겨둔 채로 실행하여야 오류가 나지 않는다.

Draw Field Profiles

5) FDTD Setting

```
print 'Setting Complete and READY to RUN'

tmax = 15000

for tstep in xrange(tmax):
    src = np.sin(wfreq_NU*tstep*fdtd.dt)

    inc.set_source(src)
    fdtd.updateE()
    fdtd.updateH()

print 'Simulation Complete'
```

- FDTD step(wave가 진행되는 시간)을 설정한다. Step을 진행할 for문을 생성하고 입사할 wave는 Sine wave로 넣고 FDTD updat를 한다.

Draw Field Profiles

6) Data Collection

```
f = h5.File('./save/%s.h5' %'fieldmap','w')
f.create_dataset('ey', data=fdtd.ey.real[:,ny/2,:])
f.close()
```

- hdf5로 y=ny/2 일 때의 xz평면에 대한 electric field를 저장한다. 여기서 저장되는 파일이름은 'fieldmap.h5' 이다.

Draw Field Profiles

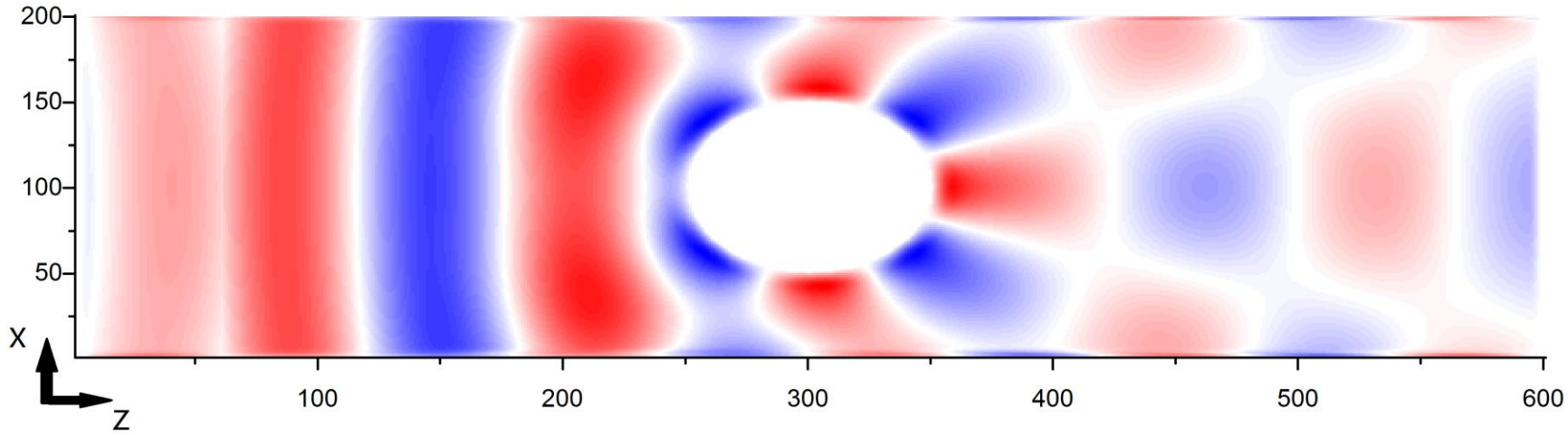
7) Execution

- Window
  - 저장된 파일을 실행한다.

- Linux
  - Python 파일이 있는 폴더에서 접속해 명령어 “python 파일명”로 실행한다.

Draw Field Profiles

8) Drawing Field map



- 저장된 결과 파일을 이용해 그래프를 그린다.