MTF_Interp_Polar_Data

September 23, 2015

1 !!WILL NOT WORK WITHOUT VOXEL DATA!!

If you have the voxel data then all code will work if the "files" variable contains string representations of the paths to each voxel file.

In [1]:
```
from IPython import parallel
clients = parallel.Client()
clients.block = True  # use synchronous computations
print(clients.ids)

lview = clients.load_balanced_view()
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 
```

In [2]:
```
import pandas as pd
```

In [11]:
```
%%px --local
 from scipy import io
 import h5py
 import numpy as np
 import itertools
 from scipy.interpolate import interp1d
 from scipy import ndimage
 from scipy.optimize import fsolve,brentq,fmin_cg,brute,bisect,newton
```

In [5]:
```
%%px --local
m = interp1d([-35,35],[0,300])
conv_vec = lambda vec: np.array([m(vec[0]),m(vec[1]),m(vec[2])])

vals = [-35,35]
x = 10

corners = [np.array(x) for x in itertools.product(vals,vals,vals)]
edge_t_corners = 30/np.linalg.norm(corners[0])
a = np.sqrt((x**2)/(4*np.dot(corners[0],corners[0])))
corner_samples_actual = [[vec*t for t in np.linspace(edge_t_corners-a,edge_t_corners+a,100)] \
                         for vec in corners]
corner_samples = [np.array([conv_vec(vec) for vec in sample]).T \
                  for sample in corner_samples_actual]

#poles
poles = [np.array([0,0,35]),np.array([0,0,-35])]
edge_t_poles = 30/np.linalg.norm(poles[0])
a = np.sqrt((x**2)/(4*np.dot(poles[0],poles[0])))
```

```
        pole_samples_actual = [[vec*t for t in np.linspace(edge_t_poles-a,edge_t_poles+a,100)] \
                               for vec in poles]
        pole_samples = [np.array([conv_vec(vec) for vec in sample]).T \
                        for sample in pole_samples_actual]

        # x,y and z planes 45deg
        zplane = [np.array([35,35,0]),np.array([-35,35,0]),np.array([35,-35,0]),np.array([-35,-35,0])]
        xplane = [np.array([0,35,35]),np.array([0,35,-35]),np.array([0,-35,35]),np.array([0,-35,-35])]
        yplane = [np.array([35,0,35]),np.array([-35,0,35]),np.array([35,0,-35]),np.array([-35,0,-35])]
        edge_t_plane = 30/np.linalg.norm(zplane[0])
        a = np.sqrt((x**2)/(4*np.dot(zplane[0],zplane[0])))
        zplane_samples_actual = [[vec*t for t in np.linspace(edge_t_plane-a,edge_t_plane+a,100)] \
                                 for vec in zplane]
        xplane_samples_actual = [[vec*t for t in np.linspace(edge_t_plane-a,edge_t_plane+a,100)] \
                                 for vec in xplane]
        yplane_samples_actual = [[vec*t for t in np.linspace(edge_t_plane-a,edge_t_plane+a,100)] \
                                 for vec in yplane]

        zplane_samples = [np.array([conv_vec(vec) for vec in sample]).T \
                          for sample in zplane_samples_actual]
        xplane_samples = [np.array([conv_vec(vec) for vec in sample]).T \
                          for sample in xplane_samples_actual]
        yplane_samples = [np.array([conv_vec(vec) for vec in sample]).T \
                          for sample in yplane_samples_actual]

        allsamples = zplane_samples + xplane_samples + yplane_samples
```

In [6]:
```
        %%px --local
        def MTF(samples,sphere):
            Y = ndimage.map_coordinates(sphere, samples, order=5)
            dY = np.diff(Y)
            fftdY = np.fft.fft(dY)
            X =np.abs(fftdY)
            return X
```

In [18]:
```
        %%px --local
         def half_width_calc(sphere):

             hlf_widths = []
             for sample in allsamples:

                 MTF_out = MTF(sample,sphere)
                 f = interp1d(np.linspace(-10,10,99),np.fft.fftshift(MTF_out))

                 try:
                     hlf_wdth = brentq(lambda x: f(x) - 0.5,0,5)
                 except:
                     hlf_wdth = 'error'

                 hlf_widths.append(hlf_wdth)

             return hlf_widths

         def MTF_calcs(fname,ind):
             sphere = h5py.File(fname)['img']['vox'].value
```

```
        vals = [fname.split('/')[-1]]
        vals += half_width_calc(sphere)

        return vals
```

In [ ]:
```
%%px --local
a = !ls /scratch/jdg1g14/all_resultspc1/vox*
b = !ls /scratch/jdg1g14/all_resultspc2/vox*
files = a+b
files
```

In [22]:
```
out = lview.map(MTF_calcs,files,range(len(files)))
```

In [56]:
```
MTF_res = pd.DataFrame(columns=['fname','X1','X2','X3','X4','Y1','Y2','Y3','Y4' \
                                ,'Z1','Z2','Z3','Z4'],index=range(len(files)))
for ind,item in enumerate(out):
    MTF_res.iloc[ind] = item


MTF_res["mag"] = MTF_res["fname"].apply(lambda x: float(x.split('_')[1]))
MTF_res["D"] = MTF_res["fname"].apply(lambda x: int(x.split('_')[3][1]))
MTF_res["S"] = MTF_res["fname"].apply(lambda x: int(x.split('_')[4][1]))
MTF_res["exp"] = MTF_res["fname"].apply(lambda x: int(x.split('_')[2]))

MTF_res[['X1','X2','X3','X4','Y1','Y2','Y3','Y4','Z1','Z2','Z3','Z4']] = \
MTF_res[['X1','X2','X3','X4','Y1','Y2','Y3','Y4','Z1','Z2','Z3','Z4']].astype(np.float64)

MTF_res.head()
```

Out[56]:

	fname	X1	X2	X3	X4	Y1
0	vox_1.5_1_D0_S0.mat	0.930477	0.930497	0.930508	0.930518	1.125532
1	vox_1.5_1_D0_S1.mat	0.678586	0.683349	0.683018	0.688339	0.746435
2	vox_1.5_1_D1_S0.mat	0.485466	0.473722	0.479446	0.481469	0.480553
3	vox_1.5_1_D1_S1.mat	0.437433	0.430403	0.426044	0.442343	0.437075
4	vox_1.5_2_D0_S0.mat	0.930531	0.930478	0.930503	0.930489	1.125583

	Y2	Y3	Y4	Z1	Z2	Z3	Z4	mag
0	1.125566	1.125579	1.125552	0.930488	0.930508	0.930488	0.930517	1.5
1	0.739818	0.737046	0.740083	0.682981	0.683390	0.685119	0.681977	1.5
2	0.480820	0.471945	0.487475	0.474332	0.474592	0.480154	0.466854	1.5
3	0.437571	0.436529	0.456245	0.437329	0.437040	0.431327	0.438650	1.5
4	1.125570	1.125572	1.125580	0.930520	0.930522	0.930465	0.930500	1.5

	D	S	exp
0	0	0	1
1	0	1	1
2	1	0	1
3	1	1	1
4	0	0	2

In [57]:
```
MTF_res.to_pickle('MTFHalfInterp.p')
```

In [86]:
```
from sortedcontainers import SortedSet
from scipy.optimize import curve_fit, fsolve
from scipy import interpolate
```

In [74]:
```
%%px --local
x = np.linspace(-35,35,301)
y = np.linspace(-35,35,301)

X,Y = np.meshgrid(x,y)

r = np.zeros_like(X)
theta = np.zeros_like(X)
for i in range(301):
    for j in range(301):
        r[i,j] = np.sqrt(X[i,j]**2+Y[i,j]**2)
```

In [77]:
```
%%px --local
def oversampled_edge(slice_,rs):

    rs = rs.flatten()
    sorted_set_rs = SortedSet(list(rs))
    index = np.argsort(rs)
    sortedr = rs[index]
    slice_ = slice_.flatten()
    slice_ordered = slice_[index]

    out = []
    for r in sorted_set_rs:
        out.append(slice_ordered[sortedr == r].mean())

    return np.array(sorted_set_rs),np.array(out)

def profiles(fname,r):

    sphere = h5py.File(fname)['img']['vox'].value
    slice_x = sphere[150,:,:]
    slice_y = sphere[:,150,:]
    slice_z = sphere[:,:,150]

    _,bx = oversampled_edge(slice_x,r)
    _,by = oversampled_edge(slice_y,r)
    _,bz = oversampled_edge(slice_z,r)



    return bx,by,bz

def dummy_foo(file):
    x,y,z =  profiles(file,r)
    return (x,y,z)
```

In [76]:
```
%%px --local
rs = r.flatten()
sorted_set_rs = SortedSet(list(rs))
index = np.argsort(rs)
sortedr = rs[index]
```

In [79]:
```
out = lview.map(dummy_foo,files)
```

```
In [81]: def sigmoid(x, a,x0, k,d):
             y = a / (1 + np.exp(-k*(x-x0))) + d
             return y

         sorted_rs = np.array(sorted_set_rs)

         def fit_sigmoid(profile):
             popt, pcov = curve_fit(sigmoid, sorted_rs, profile)
             return popt

In [82]: sig_coeffs = []
         for file in out:
             file_coeffs = []
             for slice_ in file:
                 file_coeffs.append(fit_sigmoid(slice_))
             sig_coeffs.append(file_coeffs)

In [87]: def MTF(coeffs):

             rss = np.linspace(25,35,100)

             Y = sigmoid(rss,*coeffs)
             dY = np.diff(Y)
             MTF = np.abs(np.fft.fftshift(np.fft.fft(dY)))

             f = interpolate.interp1d(np.linspace(-10,10,99),MTF)

             def opti_f(x):
                 return f(x) - 0.5

             half_width = brentq(opti_f,0,5)

             return half_width,MTF

In [88]: fout = open('halfwidths.dat','w')
         MTFS = []
         for ind,file in enumerate(files):
             fout.write(file)
             MTF_file = []
             for i in range(3):
                 width,modtransfunc = MTF(sig_coeffs[ind][i])
                 fout.write(','+str(width))
                 fout.write(','+str(sig_coeffs[ind][i][1]))
                 fout.write(','+str(sig_coeffs[ind][i][2]))
                 MTF_file.append(modtransfunc)
             fout.write('\n')
             MTFS.append(MTF_file)
         fout.close()

In [95]: df = pd.read_csv('halfwidths.dat',header=None,names=['name','xslice','xslice_radius' \
         ,'xslice_slope','yslice','yslice_radius','yslice_slope','zslice' \
         ,'zslice_radius','zslice_slope','Index'])

         df.Index = df.index
```

```
df['mag'] = df.name.apply(lambda x : float(x.split('/')[-1].split('_')[1]))
df['exp'] = df.name.apply(lambda x : float(x.split('/')[-1].split('_')[2]))
df['Soffset'] = df.name.apply(lambda x : 1.0 if x.split('/')[-1].split('_')[4][1] == '1' \
                              else 0.0)
df['Doffset'] = df.name.apply(lambda x : 1.0 if x.split('/')[-1].split('_')[3][1] == '1' \
                              else 0.0)
df.head()
```

Out[95]:

	name	xslice	xslice_radius	xslice_slope	yslice	yslice_radius	yslice_slope	zslice	zslice_radius	zslice_slope	Index	mag	exp	Soffset	Doffset
0	/scratch/jdg1g14/all_resultspc1/vox_1.5_1_D0_S...	1.117422	29.980251	4.962754	0.852496	29.993956	3.784164	0.852498	29.993955	3.784173	0	1.5	1	0	0
1	/scratch/jdg1g14/all_resultspc1/vox_1.5_1_D0_S...	0.752174	29.990553	3.332433	0.690748	29.990173	3.050705	0.690348	29.990156	3.048941	1	1.5	1	1	0
2	/scratch/jdg1g14/all_resultspc1/vox_1.5_1_D1_S...	0.481796	29.980785	2.108766	0.480884	29.979794	2.105463	0.479564	29.981178	2.099464	2	1.5	1	0	1
3	/scratch/jdg1g14/all_resultspc1/vox_1.5_1_D1_S...	0.443328	29.979673	1.946244	0.441921	29.978216	1.941053	0.441271	29.976192	1.938591	3	1.5	1	1	1
4	/scratch/jdg1g14/all_resultspc1/vox_1.5_2_D0_S...	1.117418	29.980252	4.962735	0.852496	29.993954	3.784162	0.852496	29.993956	3.784162	4	1.5	2	0	0

In [96]:
```
df.to_pickle('MTFHalfPolar.p')
```