mln_plt_Seeg_Demo

August 18, 2016

0.0.1 This file describes ipython codes which plots the final MULAN results of the calculated SEEG data which is included in the Example folder

Huifang Wang, INSERM U1106, Marseille, France

```
In [1]: import scipy.io
        import os.path
        import sys

        sys.path.append( os.path.expanduser('./Demonstration/') )
        sys.path.append( os.path.expanduser('./BasicTools/') )

        import mln_tools as mt
        import mln_plt_demonstration as md
        import mne
        import mln_circle as mc
```

```
In [2]: %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
In [3]: import matplotlib.gridspec as gridspec
```

```
In [4]: basicdir='./Examples/'
        ds=4
        iwins=1500
        datasets='SEEG_PS1001ts0'
```

```
In [5]: Toutdir=basicdir+'ToutResults/'
        prename=datasets+'ds'+str(ds)
        savedir = basicdir +'/plt/'
```

```
In [6]: fs=1000./ds
        itype='tak'
        iG=5
```

```
In [7]: toutfile='Adp_'+str(iwins)+'_'+prename+'FRN'+'_G'+str(iG)+'_'+itype+'.mat'
        filename=Toutdir+toutfile
        matdata = scipy.io.loadmat(filename)
```

```
In [8]: fromdata = scipy.io.loadmat(basicdir+'/data/'+prename+'.mat')
```

```
In [9]: arraystr=fromdata['Params']['str'][0][0]
```

```
In [10]: para=matdata['para']
         mlnMat=matdata['mlnMat']
```

```
In [11]: olabelname=arraystr.tolist()
         olabelname = [x.strip(’ ’) for x in olabelname]
```

```
In [12]: labelnamenum=olabelname
         for indch, ich in enumerate(olabelname):
             indchs=[i for i, x in enumerate(olabelname) if x ==ich]
             if np.size(indchs)>1:
                 for indd, iiid in enumerate(indchs):
                     labelnamenum[iiid]=ich+str(indd+1)
```

```
In [13]: area_color=[(’Frontal’,12,[0, 153,255,200]),
                     (’Temporal’,9,[0,204,255,200]),
                     (’Patial’,9,[0,255,255,200])]
         group_bd=[]
         igbd=0
         nc=30
         Color_ch=np.zeros([nc,4])
         for iac in area_color:
             group_bd.append(igbd)
             igbd=igbd+iac[1]
             iac4=np.array(iac[2])/255.
             Color_ch[group_bd[-1]:igbd,:]=np.tile(iac4,(iac[1],1))



         from mne.viz import circular_layout

         labelname=labelnamenum

         node_angles = circular_layout(labelname, labelname, start_pos=90,
                                       group_boundaries=group_bd)
```

```
In [14]: iBMed=’mlnMat’
         BMwins=matdata[iBMed]

         #BMMat=np.mean(BMwins[:,:,:2],axis=2)
         #BMMat=BMMat-np.diag(BMMat.diagonal())
```

```
In [15]: def mln_read_mlnMat(filename):
             matdata = scipy.io.loadmat(filename)
             mlnMat=matdata[’mlnMat’]
             para=matdata[’para’][0][0]

             return mlnMat, para
```

```
In [16]: nc=30
         fmlnMat=np.zeros([nc,nc])
         filename=basicdir+’/ToutResults/’+’Adp_’+str(iwins)+’_’+datasets+’ds’+str(ds)+’FRN’+’_G’+str(i
         mlnMat,para=mln_read_mlnMat(filename)
         methodlog=para[’methodlog’]
         theta=para[’theta’]
         dataname=basicdir+’data/’+datasets+’ds’+str(ds)
         matdata = scipy.io.loadmat(dataname)
```

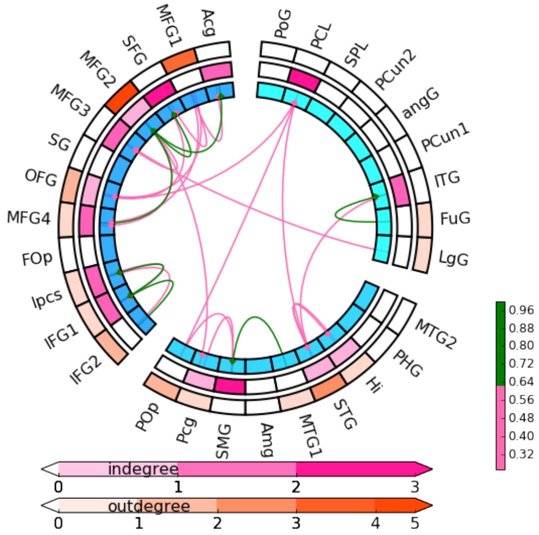
```
        thetamlnMat=mlnMat
        for iboot in np.arange(20):
            imlnMat=mlnMat[:,:,iboot]
            ithethod=para['theta'][iboot]
            imlnMat[imlnMat<ithethod[0]]=0
            imlnMat[imlnMat>=ithethod[1]]=2
            imlnMat[(imlnMat>=ithethod[0])*(imlnMat<ithethod[1])]=1
            thetamlnMat[:,:,iboot]=imlnMat
        imdMat=np.median(thetamlnMat,axis=2)
        imdMat[imdMat<0.3]=0
        fmlnMat=imdMat
        fmlnMat=fmlnMat/2.
```

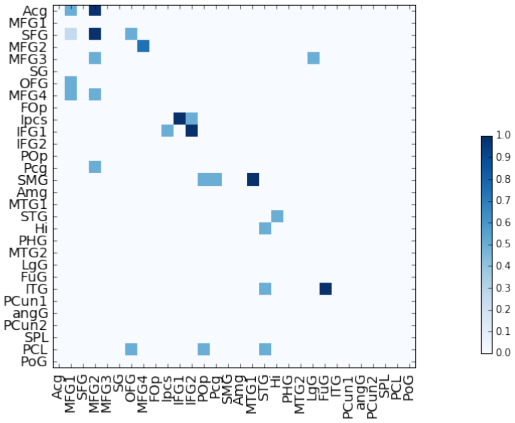
```
In [17]: import matplotlib.colors as mcolors
         def make_colormap(seq):
             """Return a LinearSegmentedColormap
             seq: a sequence of floats and RGB-tuples. The floats should be increasing
             and in the interval (0,1).
             """
             seq = [(None,) * 3, 0.0] + list(seq) + [1.0, (None,) * 3]
             cdict = {'red': [], 'green': [], 'blue': []}
             for i, item in enumerate(seq):
                 if isinstance(item, float):
                     r1, g1, b1 = seq[i - 1]
                     r2, g2, b2 = seq[i + 1]
                     cdict['red'].append([item, r1, r2])
                     cdict['green'].append([item, g1, g2])
                     cdict['blue'].append([item, b1, b2])
             return mcolors.LinearSegmentedColormap('CustomMap', cdict)


         c = mcolors.ColorConverter().to_rgb
         rvb = make_colormap(
         [c('Hotpink'),0.5,c('green')] )
         N = 100
         array_dg = np.random.uniform(0, 10, size=(N, 2))
         links_colors = np.random.uniform(0, 2, size=(N,))
         plt.register_cmap(name='mulan', cmap=rvb)
```

```
In [18]: fig= plt.figure(figsize = (20,10))

         fig, axes=mc.plot_connectivity_circle_dir_2para(fmlnMat, labelname, n_lines=None, facecolor='wh
                                  node_colors=Color_ch,colormap='mulan',node_angles=node_angles, flagcon
                                  title='',fig=fig,padding = 4.,fontsize_colorbar=12,fontsize_names=16)
         plt.savefig('ts0'+datasets+'.eps',pdi=600)
```


Acg
MFG1
SFG
MFG2
MFG3
SG
OFG
MFG4
FOp
Ipcs
IFG1
IFG2
POp
Pcg
SMG
Amg
MTG1
STG
Hi
PHG
MTG2
LgG
FuG
ITG
PCun1
angG
PCun2
SPL
PCL
PoG
0.96
0.88
0.80
0.72
0.64
0.56
0.48
0.40
0.32
indegree
0
1
2
3
outdegree
0
1
2
3
4
5


Acg
MFG1
SFG
MFG2
MFG3
SG
OFG
MFG4
FOp
Ipcs
IFG1
IFG2
POp
Pcg
SMG
Amg
MTG1
STG
Hi
PHG
MTG2
LgG
FuG
ITG
PCun1
angG
PCun2
SPL
PCL
PoG
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0