GCPDS - Visualizations

Yeison Cardona

Jul 18, 2023

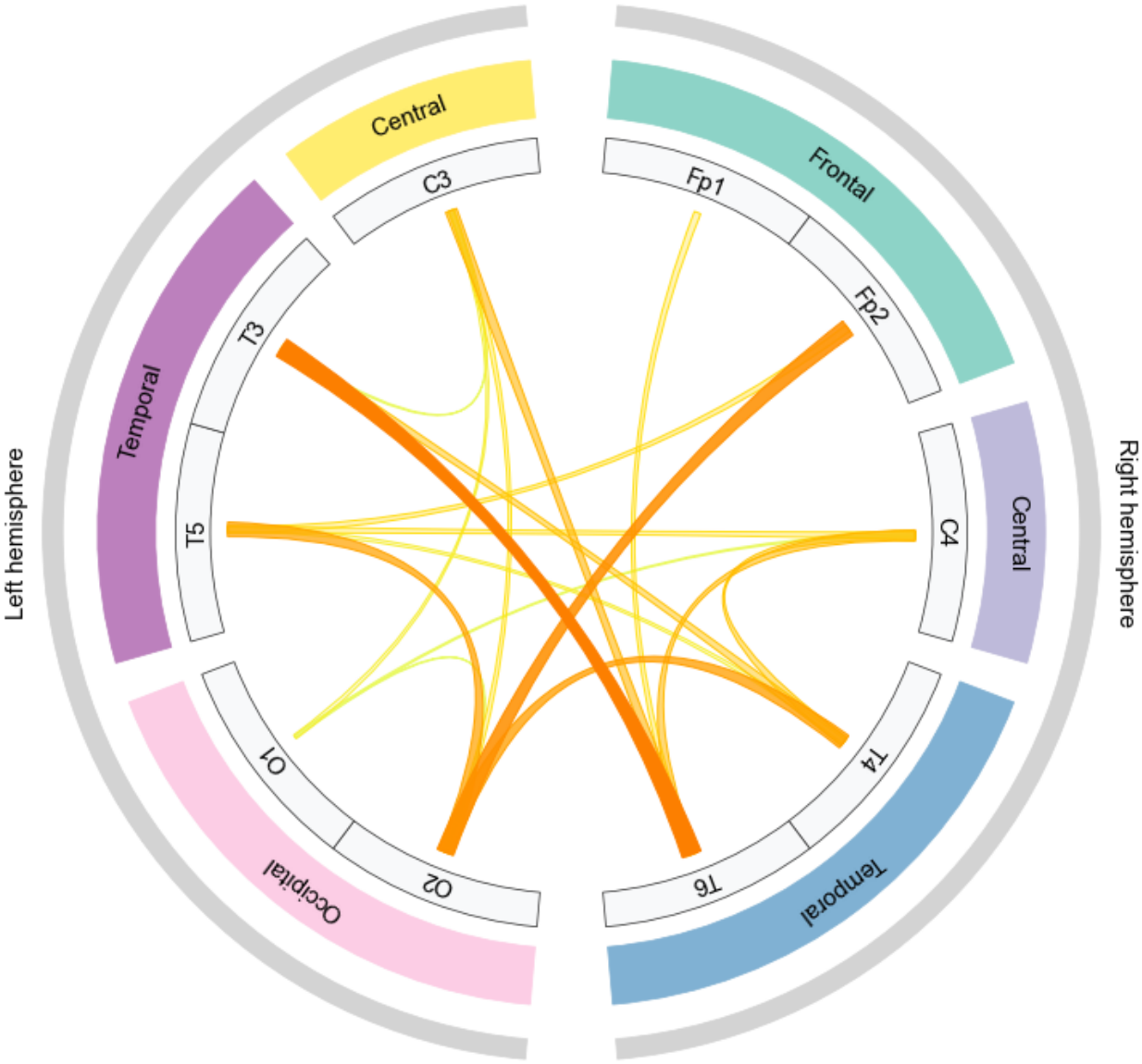
CONTENTS

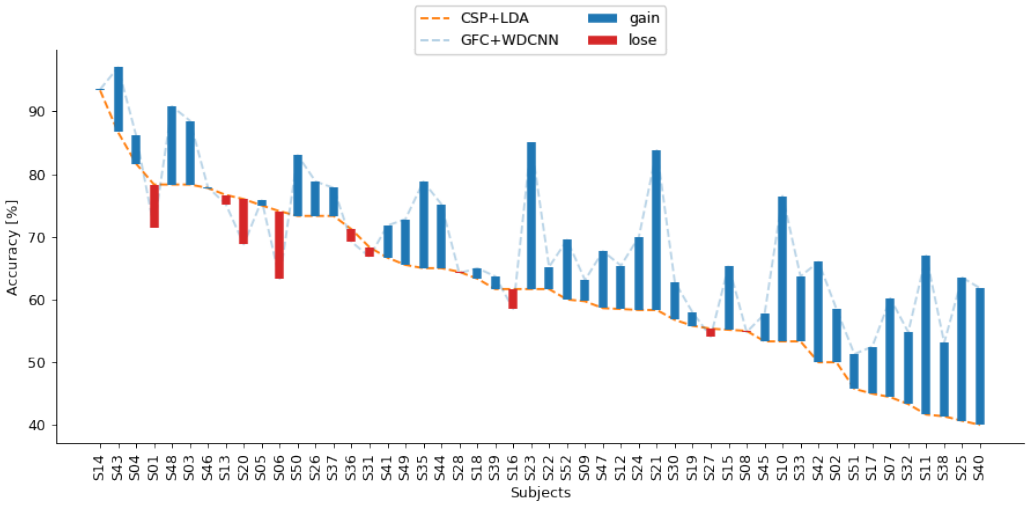
CHAPTER

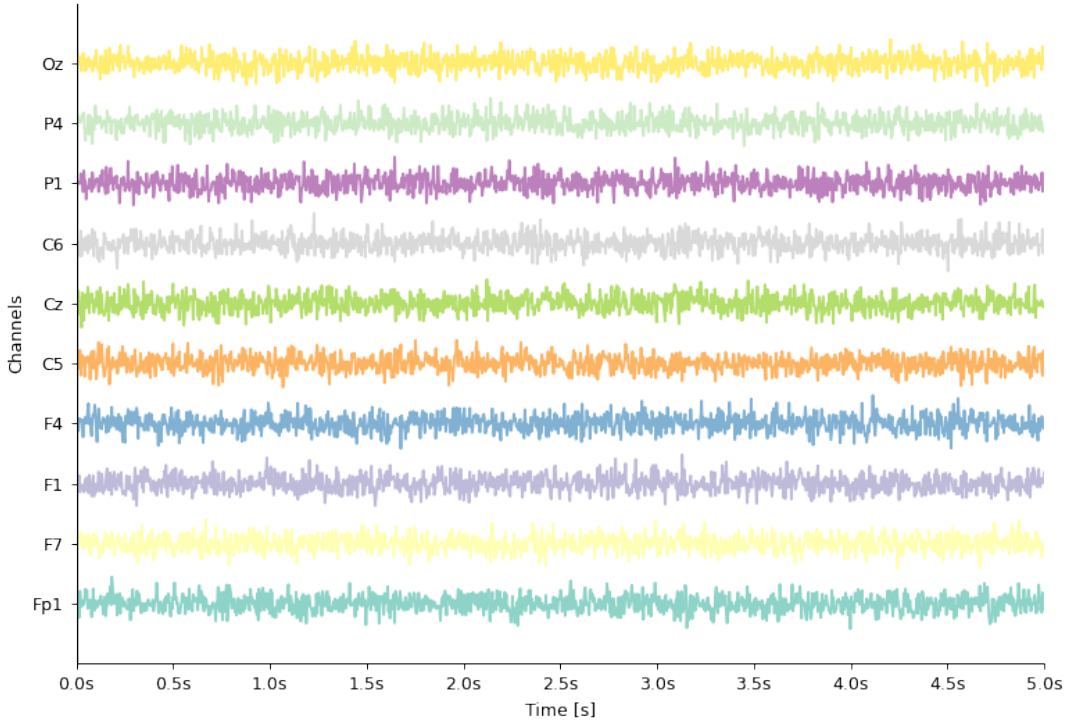
ONE

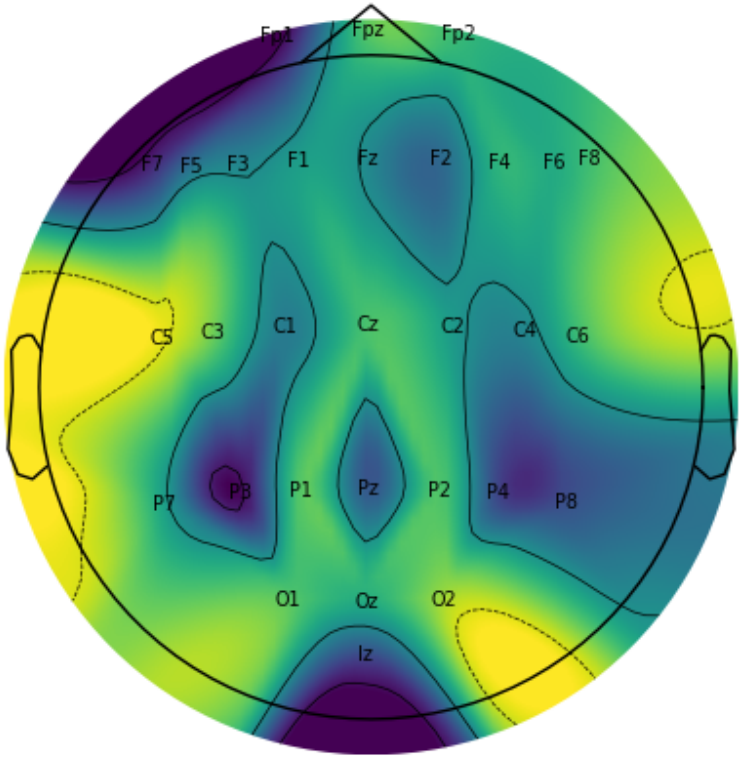
INSTALLATION

```
pip install -U git+https://github.com/UN-GCPDS/python-gcpds.visualizations.git
```


Central
Frontal
Temporal
Occipital
Central
Temporal
Left hemisphere
Right hemisphere
C3
Fp1
Fp2
T3
T5
C4
T4
T6
O1
O2


CSP+LDA
GFC+WDCNN
gain
lose
Accuracy [%]
90
80
70
60
50
40
S14 S43 S04 S01 S48 S03 S46 S13 S20 S05 S06 S50 S26 S37 S36 S31 S41 S49 S35 S44 S28 S18 S39 S16 S23 S22 S52 S09 S47 S12 S24 S21 S30 S19 S27 S15 S08 S45 S10 S33 S42 S02 S51 S17 S07 S32 S11 S38 S25 S40
Subjects


Channels
Oz
P4
P1
C6
Cz
C5
F4
F1
F7
Fp1
0.0s 0.5s 1.0s 1.5s 2.0s 2.5s 3.0s 3.5s 4.0s 4.5s 5.0s
Time [s]


Fp1
Fpz
Fp2
F7
F5
F3
F1
Fz
F2
F4
F6
F8
C5
C3
C1
Cz
C2
C4
C6
P7
P3
P1
Pz
P2
P4
P8
O1
Oz
O2
Iz

CHAPTER

TWO

NAVIGATION

2.1 Brain connectivities

Note:

Colab users must ensure to install *matplotlib* via *apt-get* with the following command:
`!apt-get install python3-matplotlib`

[3]:
```
from gcpds.visualizations.connectivities import CircosConnectivity
```

The channels that are used to display the connectivities:

[4]:
```
channels = ['Fp2', 'C4', 'T4', 'T6', 'O2', 'O1', 'T3', 'T5', 'C3', 'Fp1']
```

The areas are used to group a set of electrodes:

[5]:
```
areas = {
    'Frontal': ['Fp2'],
    'Central': ['C4'],
    'Temporal': ['T4', 'T6'],
    'Occipital': ['O2'],

    'Occipital_': ['O1'],
    'Temporal_': ['T5', 'T3'],
    'Central_': ['C3'],
    'Frontal_': ['Fp1'],
}
```

Random conectivities values:

[6]:
```
N = len(channels)  # channels

connectivities_s = np.random.normal(size=(N, N))  # Matrix form
connectivities_s.shape
```

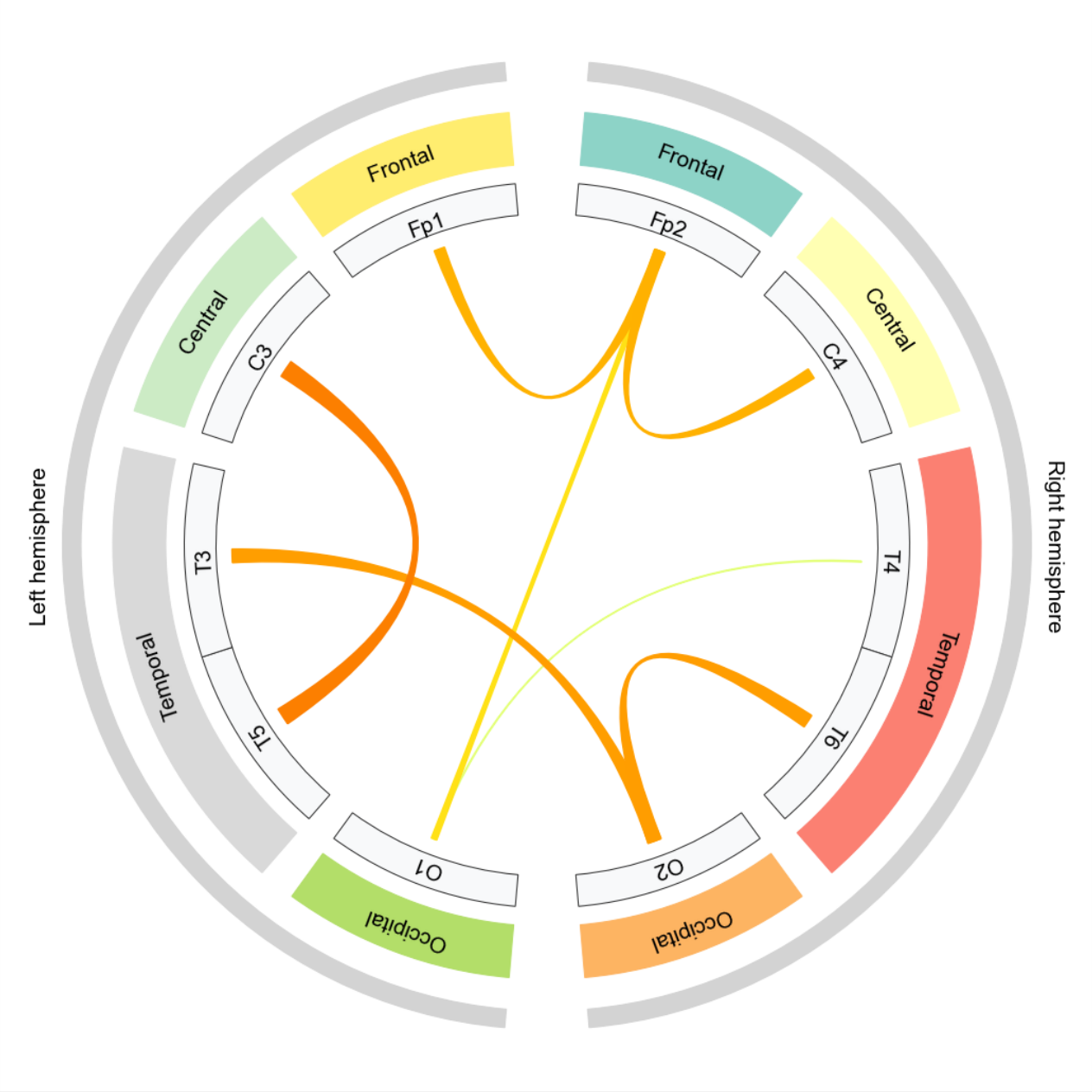
[6]:
```
(10, 10)
```

[7]:
```
connectivities_v = connectivities_s[np.triu_indices(N, k=1)].reshape(-1)  # vector form
connectivities_v.shape
```

[7]:
```
(45,)
```

2.1.1 Inter-hemispheric plots

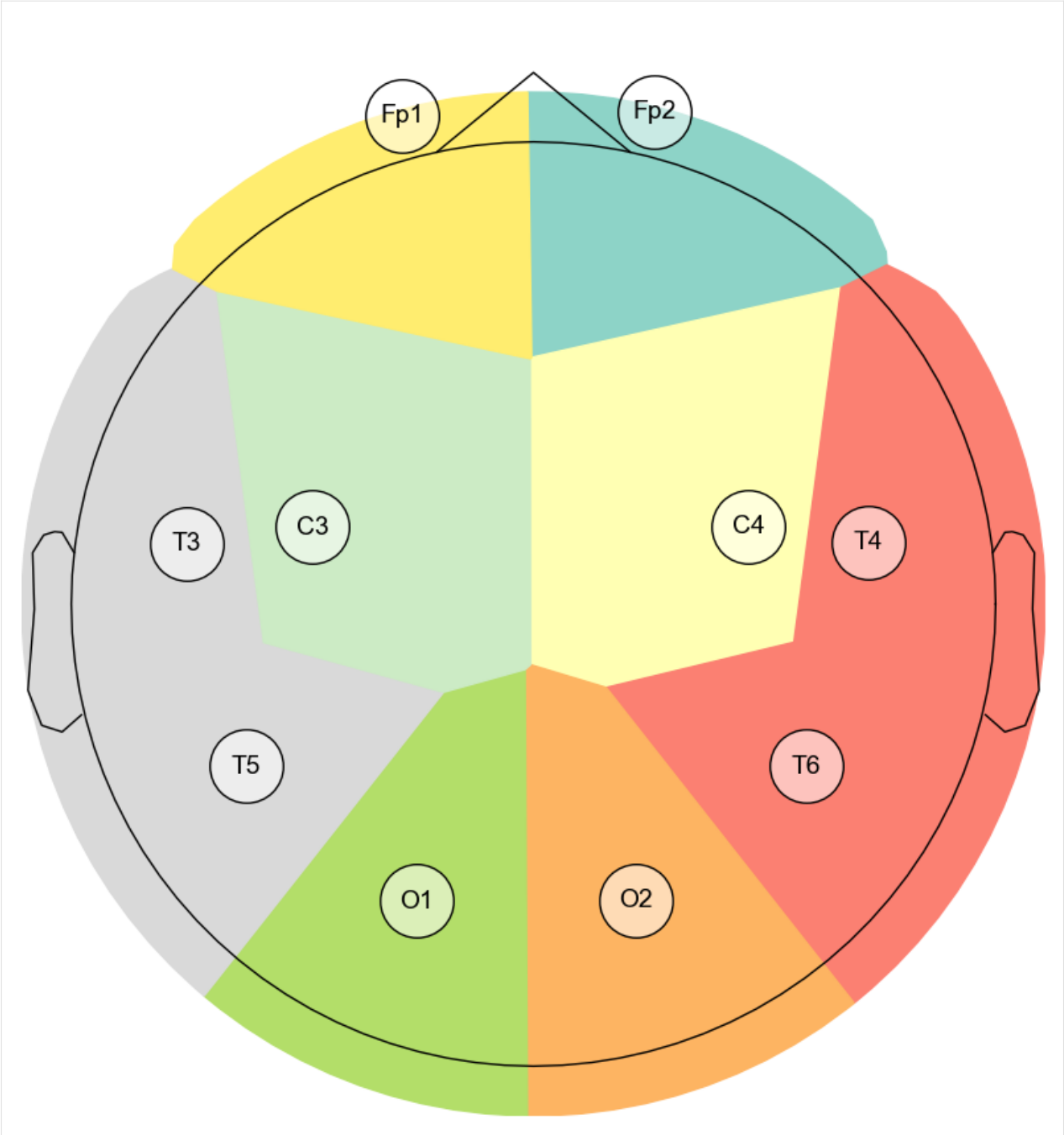
[8]:
```
conn = CircosConnectivity(
    connectivities_v, channels, areas=areas, threshold=0.7,

    # cmaps and themes
    areas_cmap='Set3',
    arcs_cmap='Wistia',
    hemisphere_color='lightgray',
    channel_color='#f8f9fa',
    min_alpha=0,


    # Texts
    width={'hemispheres':35, 'areas':100, 'channels':60},
    text={'hemispheres':40, 'areas':20,  'channels':40},
    separation={'hemispheres':10, 'areas':-30, 'channels':5},
    labelposition={'hemispheres':60, 'areas':0, 'channels':-10},
    size=10,
    labelsize=15,


    # Shapes
    show_emisphere=True,
    arcs_separation=30,
    connection_width=0.1,
    small_separation=5,
    big_separation=10,
    offset=0,
)

conn.figure;
```



Topoplot reference:

[9]:
```
conn.topoplot_reference(
    montage_name='standard_1005',
    size=10,
    fontsize=20,
    markersize=40,
    markerfacecolor='#ffffff88',
    markeredgecolor='#000000',
)
```

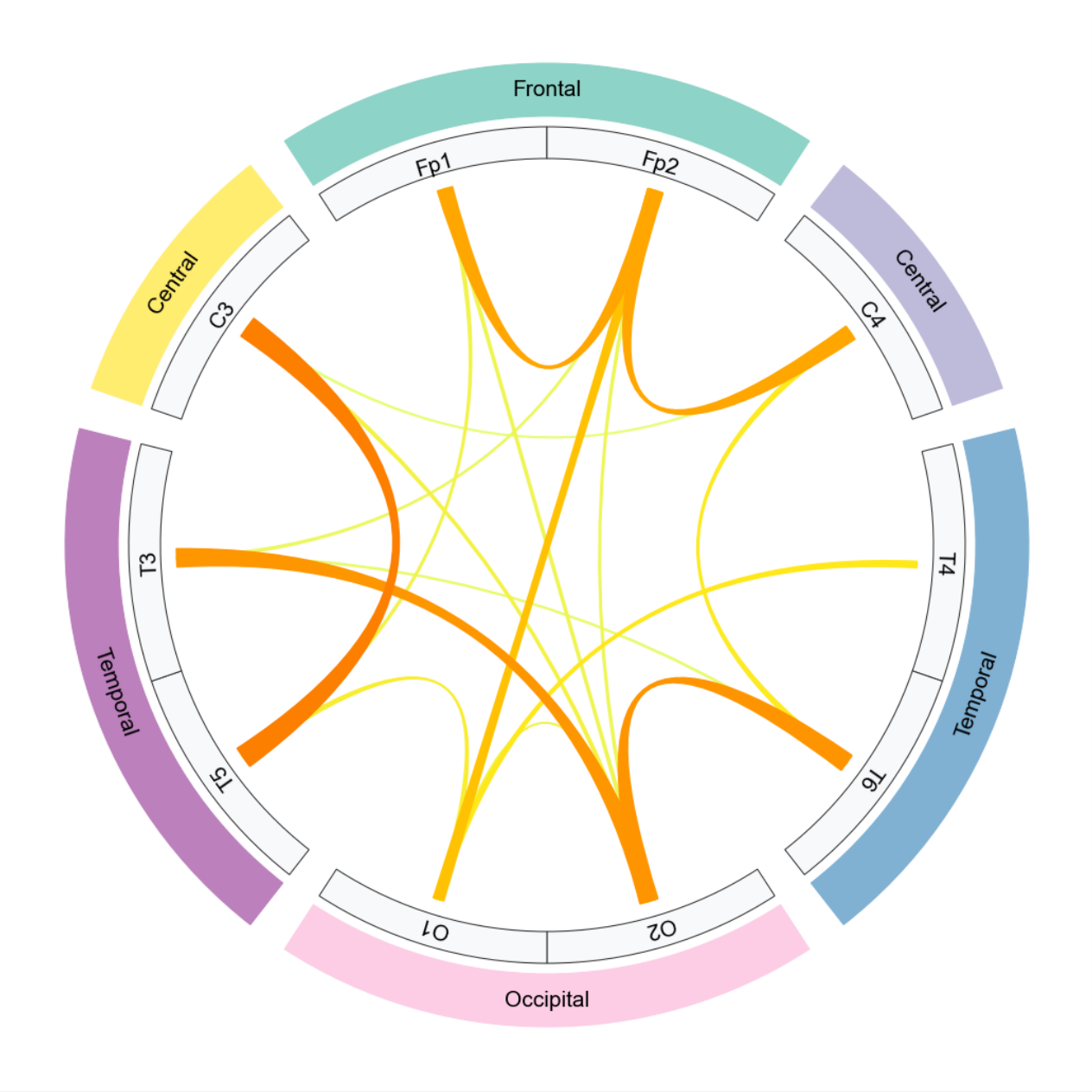
[9]:
```
<Axes: >
```


Fp1
Fp2
T3
C3
C4
T4
T5
T6
O1
O2

2.1.2 Cross-hemispheric plot

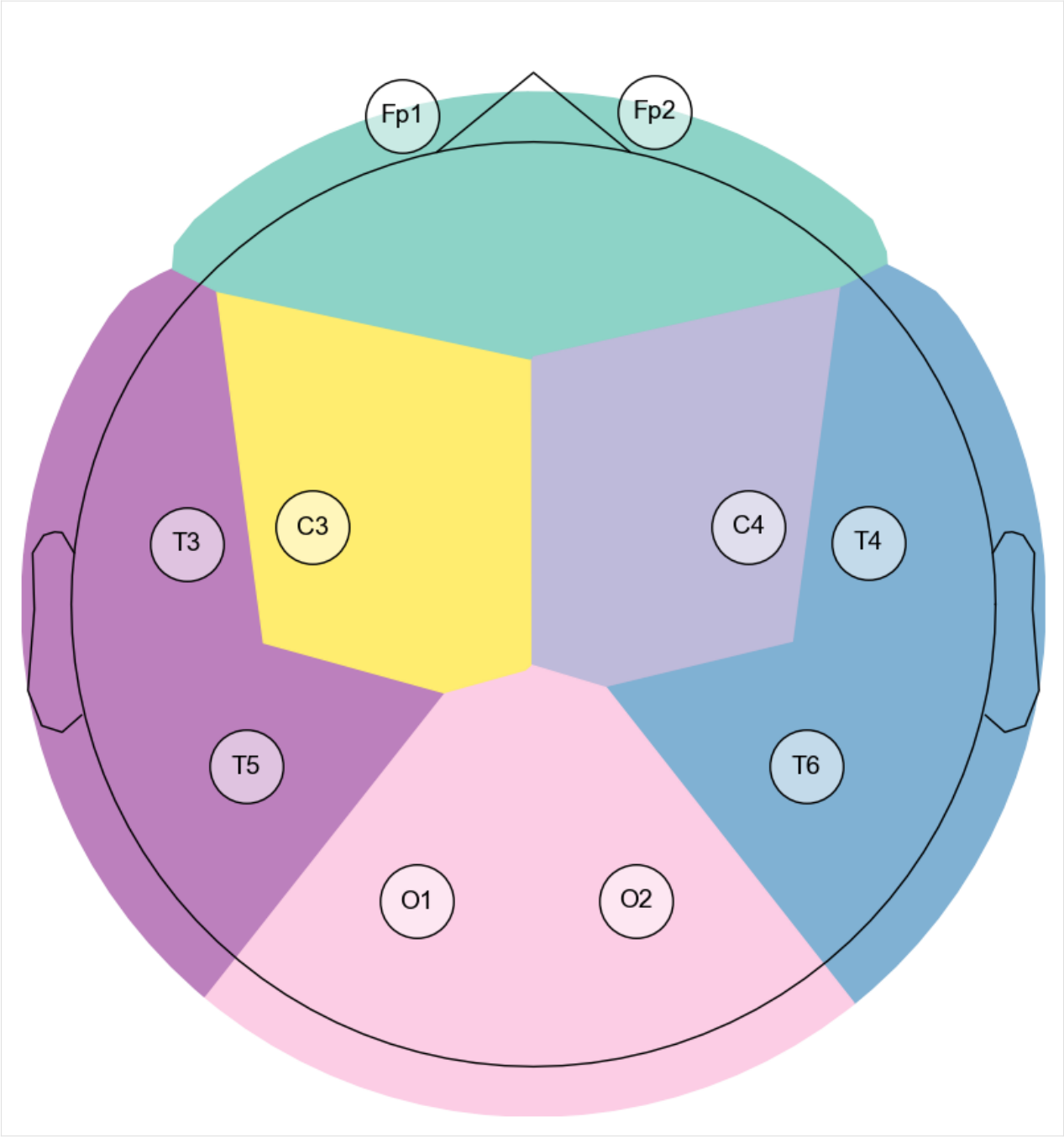
[10]:
```
areas = {
    'Frontal': ['Fp1', 'Fp2'],
    'Central': ['C4'],
    'Temporal': ['T4', 'T6'],
    'Occipital': ['O2','O1'],
    'Temporal_': ['T5', 'T3'],
    'Central_': ['C3'],
}
```

[11]:
```
conn = CircosConnectivity(
    connectivities_v, channels, areas=areas, threshold=0.3,

    # cmaps and themes
    areas_cmap='Set3',
    arcs_cmap='Wistia',
    hemisphere_color='lightgray',
    channel_color='#f8f9fa',
    min_alpha=0,


    # Texts
    width={'areas':100, 'channels':60},
    text={'areas':20,  'channels':40},
    separation={'areas':-30, 'channels':5},
    labelposition={'areas':0, 'channels':-10},
    size=10,
    labelsize=15,


    # Shapes
    show_emisphere=False,
    arcs_separation=30,
    connection_width=0.1,
    small_separation=5,
    big_separation=10,
    offset=-1,
)

conn.figure;
```



[12]:
```
conn.topoplot_reference(
    montage_name='standard_1005',
    size=10,
    fontsize=20,
    markersize=40,
    markerfacecolor='#ffffff88',
    markeredgecolor='#000000',
)
```

[12]:
```
<Axes: >
```


Fp1
Fp2
T3
C3
C4
T4
T5
T6
O1
O2

2.2 Accuracy and Gain Comparison (AGCO)

[7]:
```
from gcpds.visualizations.accuracy import agco
```

Input data:

[8]:
```
ticks = np.array(['S01', 'S02', 'S03', 'S04', 'S05', 'S06', 'S07', 'S08', 'S09', 'S10',
                  'S11', 'S12', 'S13', 'S14', 'S15', 'S16', 'S17', 'S18', 'S19', 'S20',
                  'S21', 'S22', 'S23', 'S24', 'S25', 'S26', 'S27', 'S28', 'S30', 'S31',
                  'S32', 'S33', 'S35', 'S36', 'S37', 'S38', 'S39', 'S40', 'S41', 'S42',
                  'S43', 'S44', 'S45', 'S46', 'S47', 'S48', 'S49', 'S50', 'S51', 'S52'])

method_1 = np.array([78.33, 50.  , 78.33, 81.67, 75.  , 74.07, 44.44, 55.  , 59.72,
                     53.33, 41.67, 58.49, 76.67, 93.33, 55.17, 61.67, 45.  , 63.33,
                     55.81, 76.  , 58.33, 61.67, 61.67, 58.33, 40.68, 73.33, 55.36,
                     64.41, 56.76, 68.33, 43.33, 53.33, 65.  , 71.19, 73.33, 41.38,
                     61.67, 40.  , 66.67, 50.  , 86.67, 65.  , 53.33, 77.78, 58.62,
                     78.33, 65.52, 73.33, 45.76, 60.])

method_2 = np.array([71.5, 58.5, 88.5, 86.2, 75.8, 63.3, 60.2, 54.8, 63.1, 76.5, 67. ,
                     65.4, 75.2, 93.5, 65.4, 58.5, 52.5, 65. , 57.9, 68.8, 83.8, 65.2,
                     85. , 70. , 63.5, 78.8, 54.1, 64.2, 62.8, 66.8, 54.8, 63.7, 78.8,
                     69.2, 77.8, 53.1, 63.7, 61.8, 71.8, 66. , 97. , 75.2, 57.7, 77.9,
                     67.7, 90.8, 72.8, 83. , 51.3, 69.5])

labels = ['CSP+LDA', 'GFC+WDCNN']
```

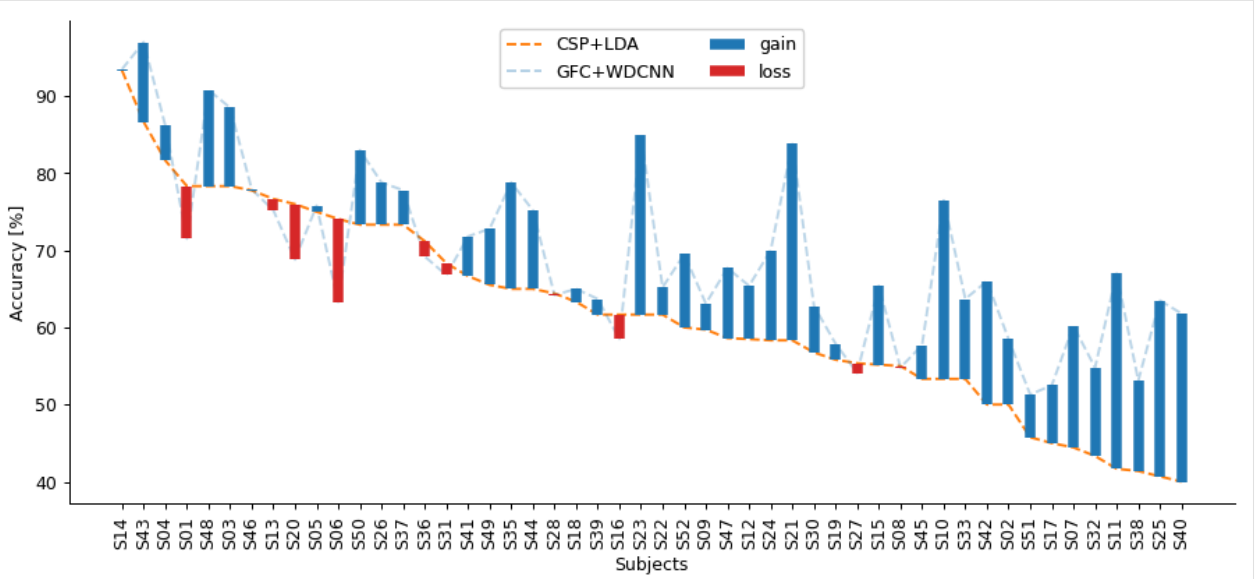
2.2.1 Plot arguments

[9]:
```
fig = agco(
    method_1, method_2, ticks, labels, sort='method_1',

    # styles
    reference_c='C1',
    gain_c='C0',
    loss_c='C3',
    barwidth=6,

    # labels
    ylabel='Accuracy [%]',
    xlabel='Subjects',
    gain_labels=['gain', 'loss'],

    # figure options
    size=(12, 5),
    fig=None,
    ax=None,
    dpi=90,
)

## To change the legend position
```

(continues on next page)

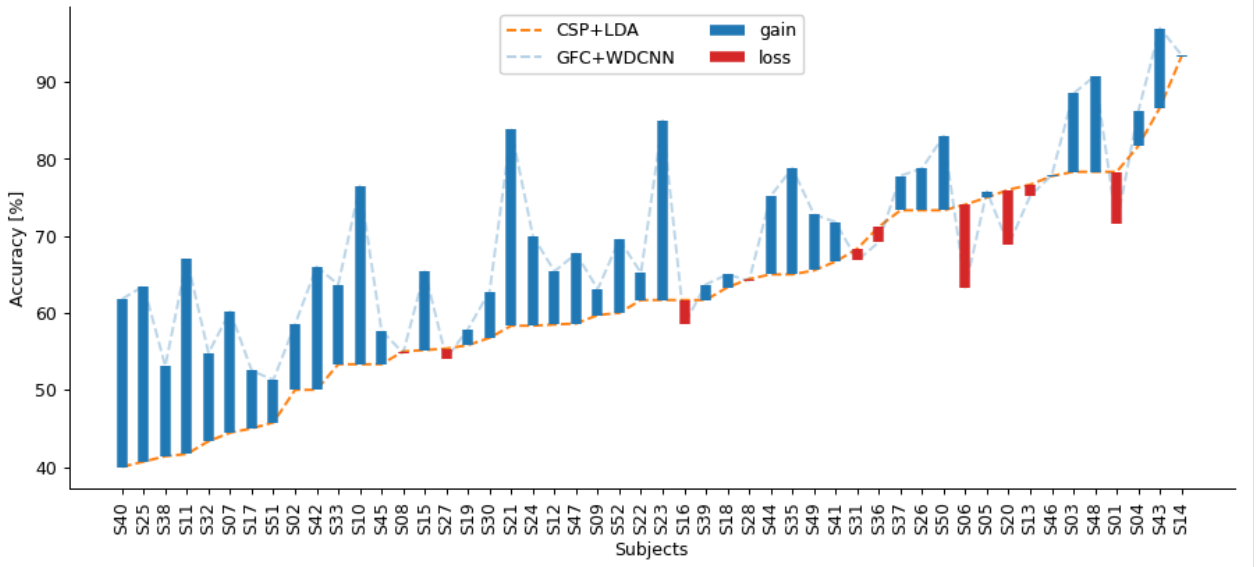
(continued from previous page)

```
# legend = plt.gca().artists[0]
# l2 = plt.legend(plt.gca().get_children(), labels[:4], loc='upper right', ncol=2)
# plt.gca().add_artist(l2)
# legend.remove()
```



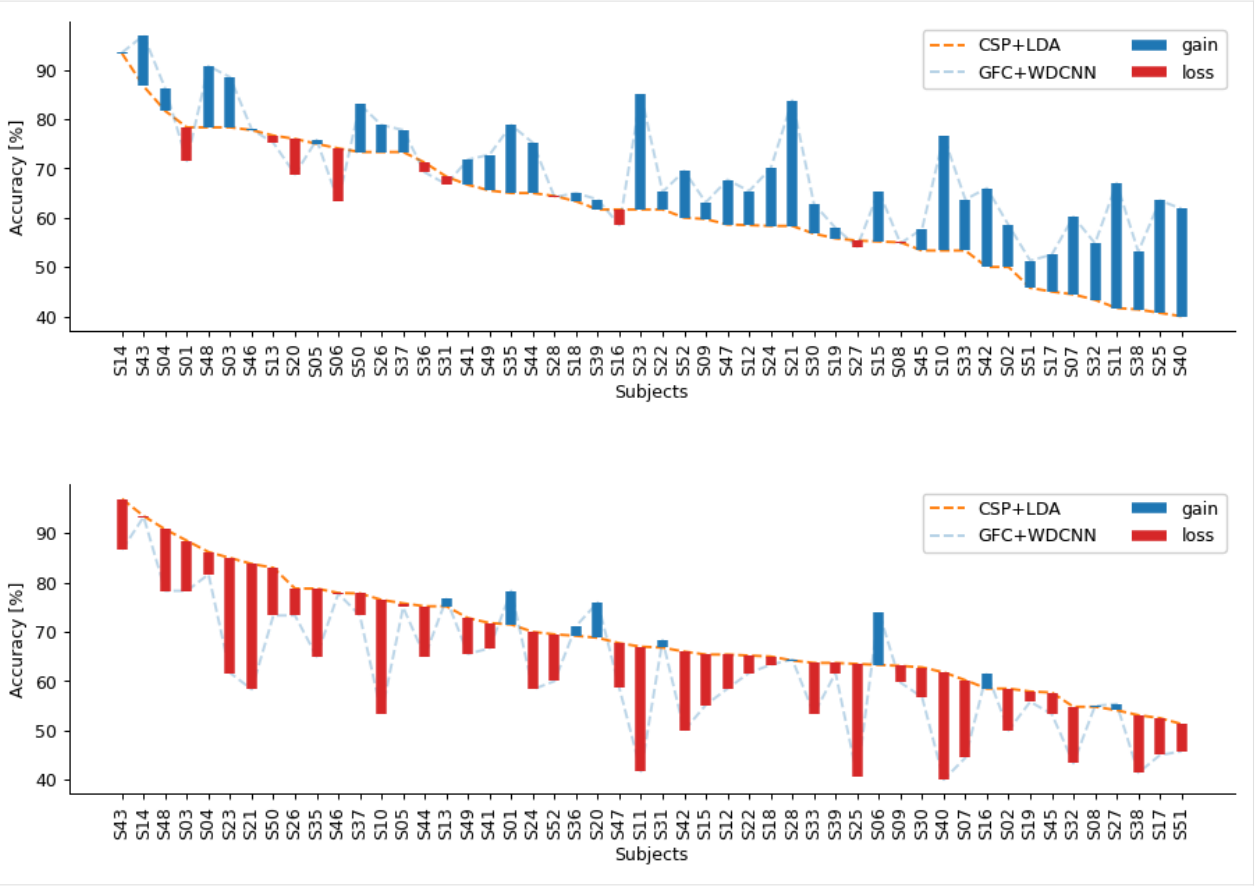
2.2.2 Reverse sort

[6]:
```
agco(method_1, method_2, ticks, labels, sort='method_1r', size=(12, 5));
```



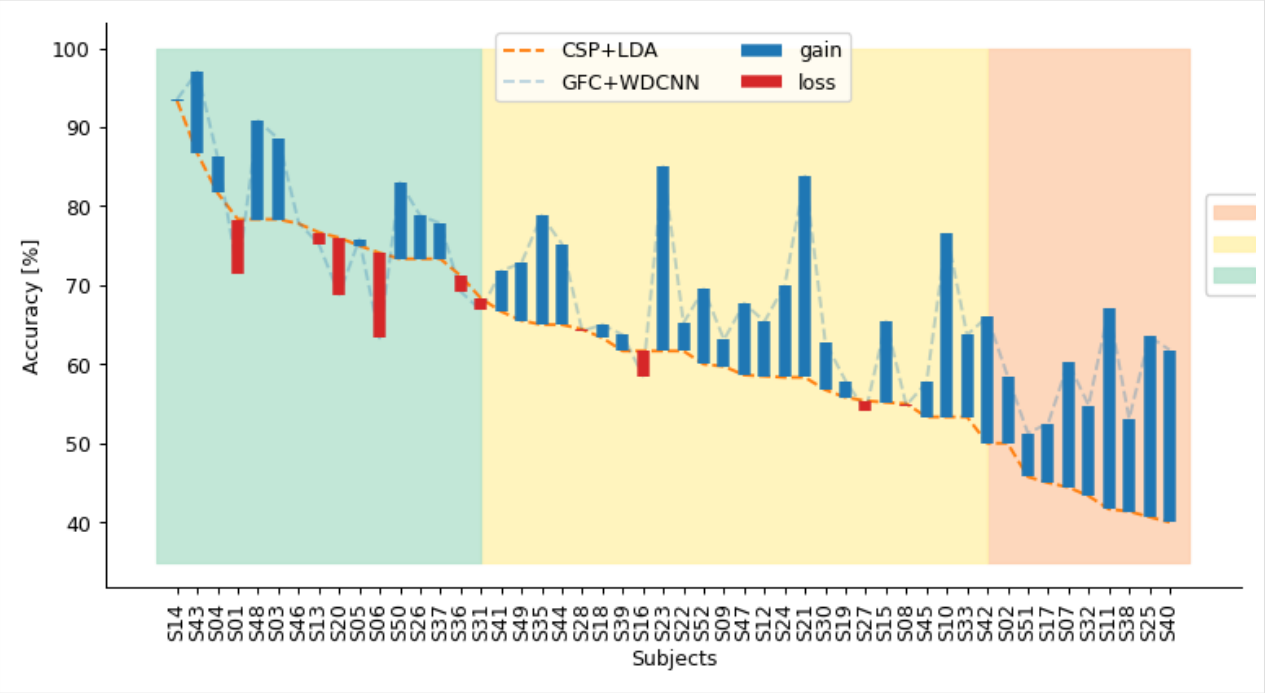
2.2.3 Subplots

[11]:
```
plt.figure(figsize=(12, 8), dpi=90)

ax1 = plt.subplot(211)
agco(method_1, method_2, ticks, labels, sort='method_1', ax=ax1, fig=plt.gcf())

ax2 = plt.subplot(212)
agco(method_1, method_2, ticks, labels, sort='method_2', ax=ax2, fig=plt.gcf())

plt.subplots_adjust(hspace=0.5)
```



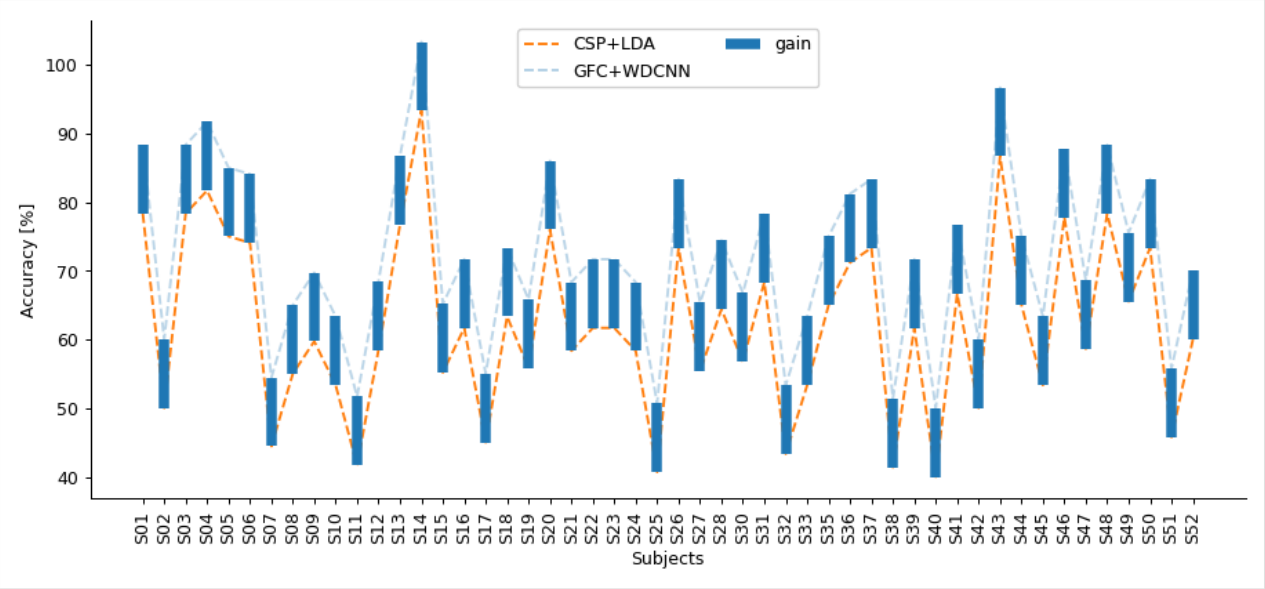
2.2.4 Plot modifications

[8]:
```
fig = agco(method_1, method_2, ticks, labels, sort='method_1', size=(10, 5))

pastel2 = plt.get_cmap('Pastel2')
f1 = plt.fill_betweenx([35, 100], 40, 50, color=pastel2(1), alpha=0.8, label='4444')
f2 = plt.fill_betweenx([35, 100], 15, 40, color=pastel2(5), alpha=0.8)
f3 = plt.fill_betweenx([35, 100], -1, 15, color=pastel2(0), alpha=0.8)
l2 = plt.legend([f1, f2, f3], ["2%", '3%', '5%'], loc=4, ncol=1, bbox_to_anchor =(1.07,␣
↪0.5))

plt.gca().add_artist(l2);
```



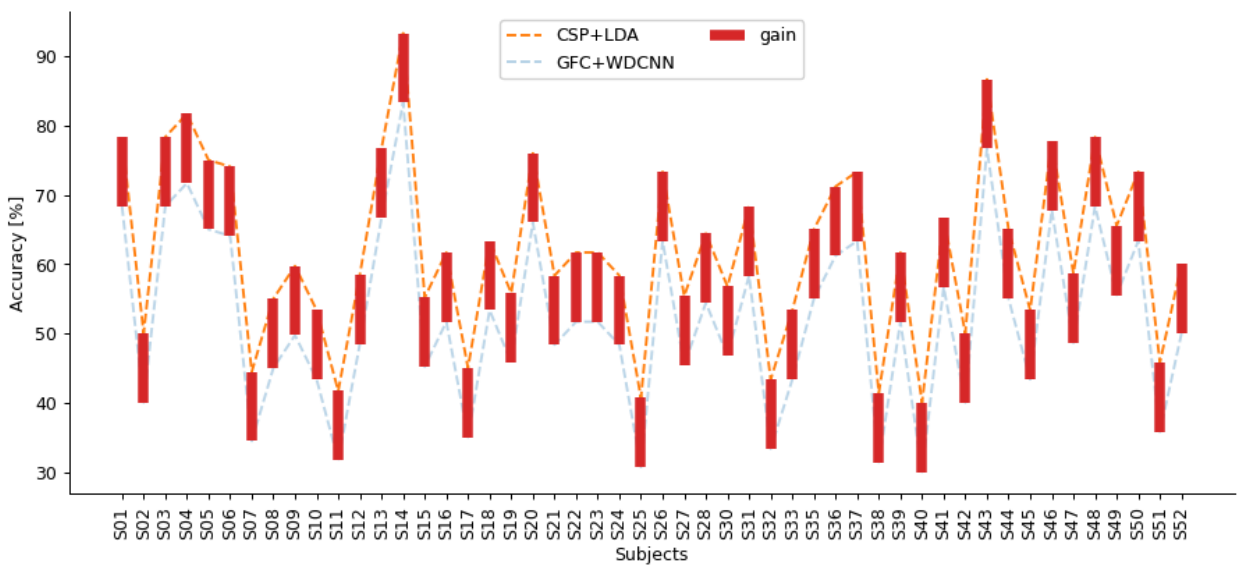
2.2.5 All gains

[9]:
```
agco(method_1, method_1+10, ticks, labels, sort=None, size=(12, 5));
```



2.2.6 All loss

[10]:
```
agco(method_1, method_1-10, ticks, labels, sort=None, size=(12, 5));
```



2.3 EEG plot

[3]:
```
from gcpds.visualizations.series import plot_eeg
```

Input data:

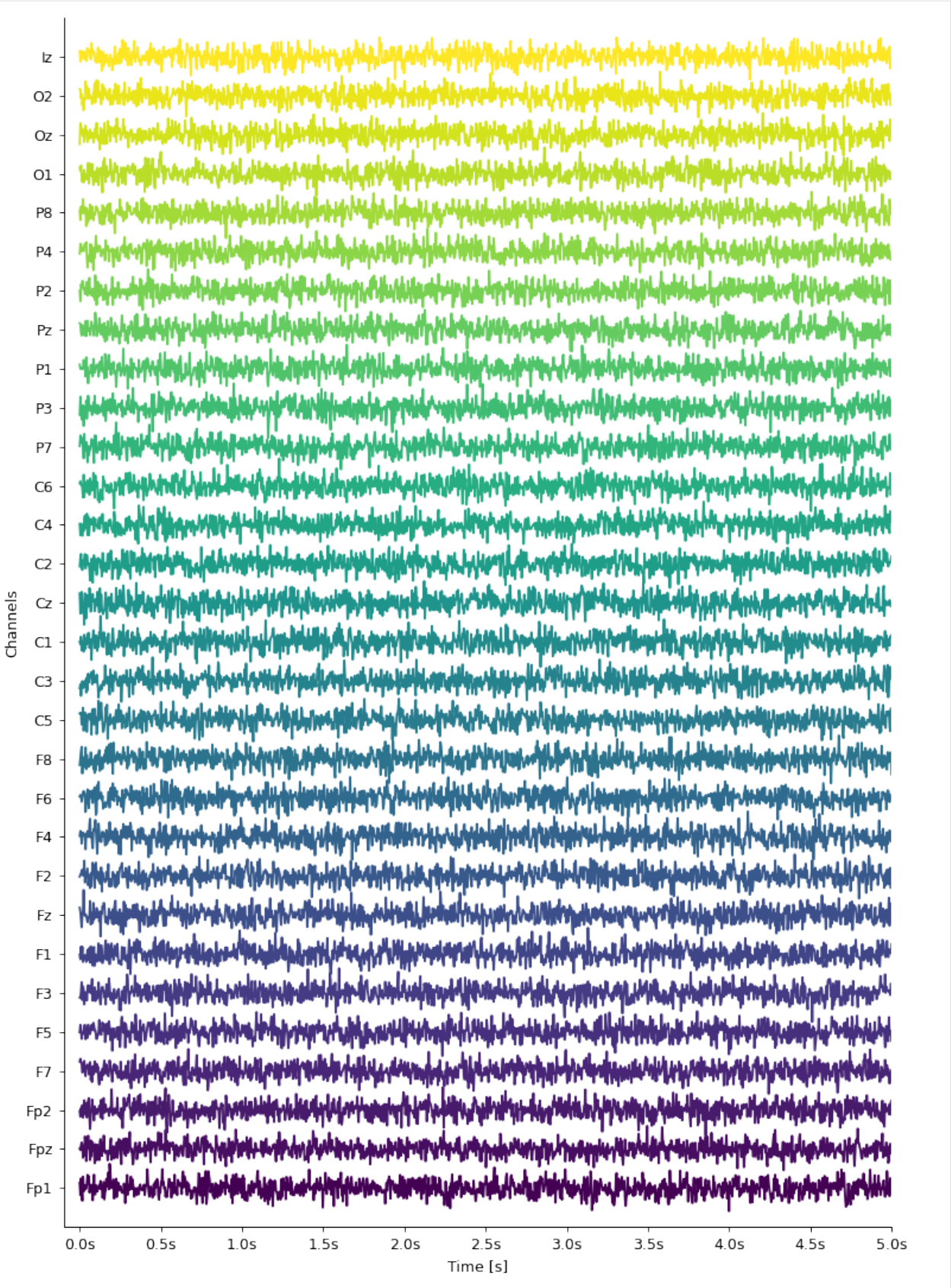
[4]:
```
channels = ['Fp1','Fpz','Fp2',
            'F7','F5','F3','F1','Fz','F2','F4','F6','F8',
            'C5','C3','C1','Cz','C2','C4','C6',
            'P7', 'P3','P1','Pz','P2','P4','P8',
            'O1','Oz','O2',
            'Iz']

sample_rate = 250
```

[5]:
```
data = np.random.normal(size=(len(channels), sample_rate*5))
data.shape
```

[5]:
```
(30, 1250)
```

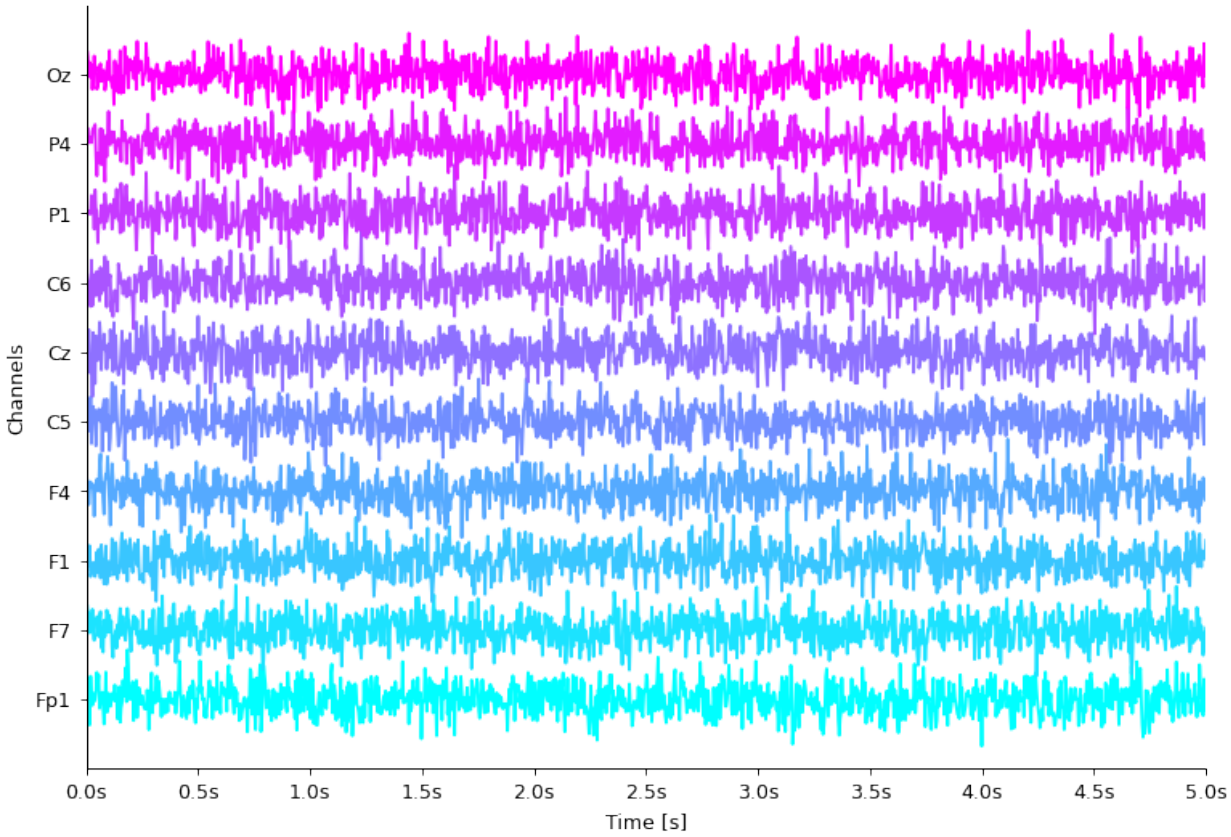
2.3.1 Plot arguments

[10]:
```
plot_eeg(
    data, channels, sample_rate,

    # color theme
    cmap='viridis',

    # axis options
    nxlabel=10,
    sca=0.7,
    offset=-0.1,

    # figure options
    fig=None,
    ax=None,
    size=(10, 15),
    dpi=90

);
```

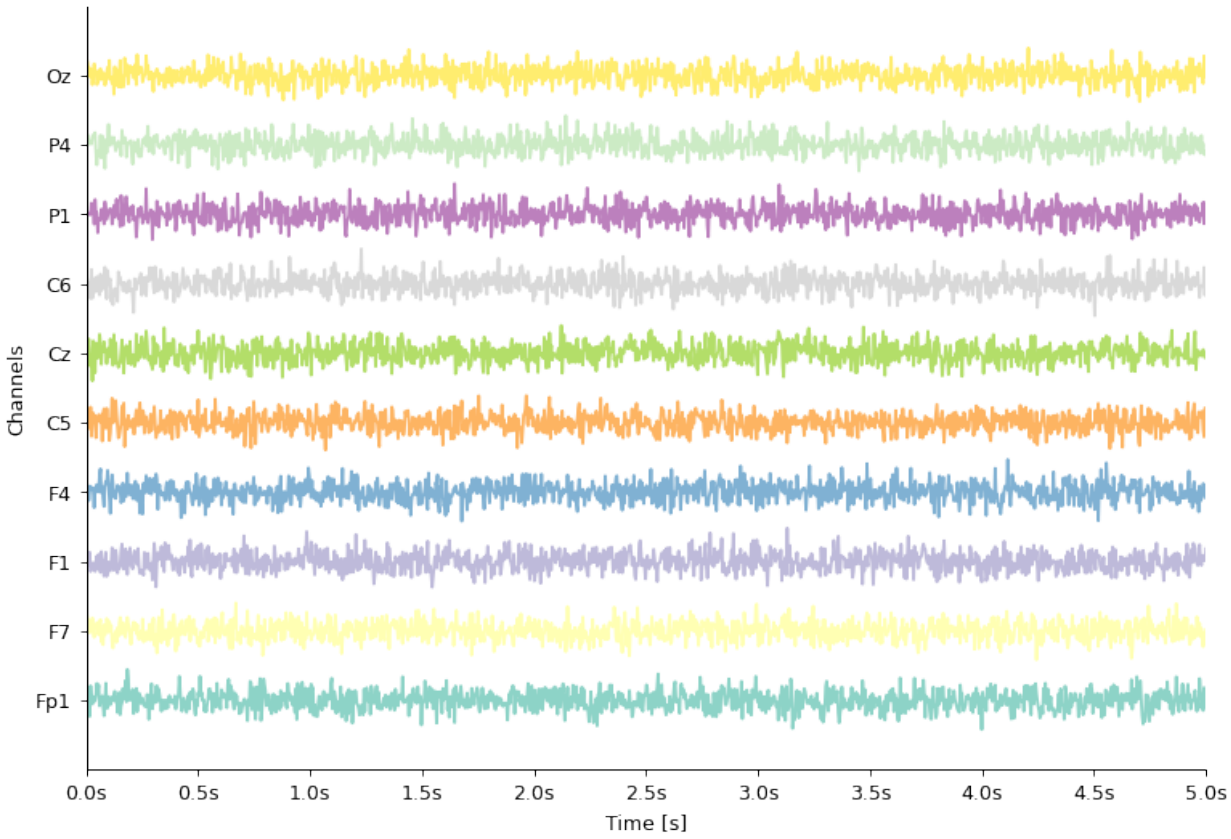

Iz
O2
Oz
O1
P8
P4
P2
Pz
P1
P3
P7
C6
C4
C2
Cz
C1
C3
C5
F8
F6
F4
F2
Fz
F1
F3
F5
F7
Fp2
Fpz
Fp1
Channels
0.0s
0.5s
1.0s
1.5s
2.0s
2.5s
3.0s
3.5s
4.0s
4.5s
5.0s
Time [s]

2.3.2 Colors

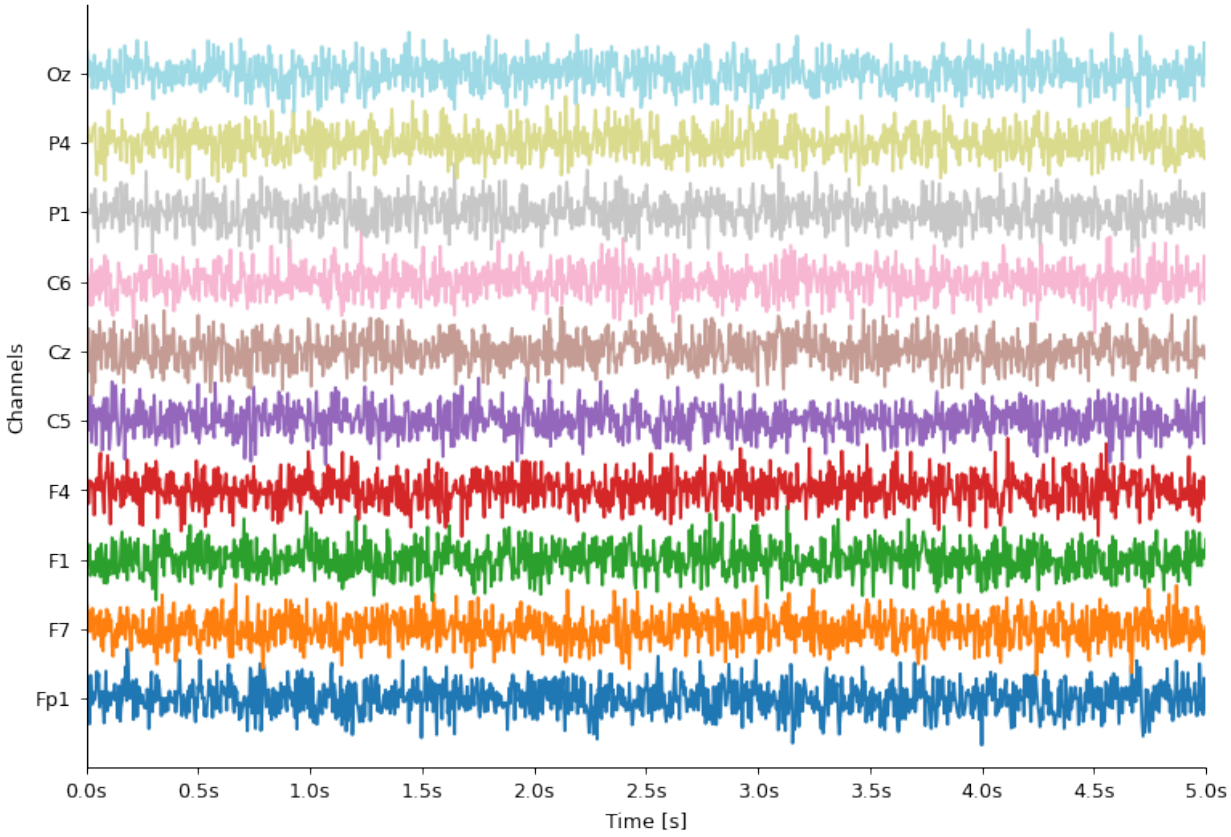
[11]:
```
plot_eeg(
    data[::3], channels[::3], sample_rate, size=(10, 7), sca=0.8,

    # color theme
    cmap='cool',
);
```



[15]:
```
plot_eeg(
    data[::3], channels[::3], sample_rate, size=(10, 7), sca=0.5,

    # color theme
    cmap='Set3',
);
```



[13]:
```
plot_eeg(
    data[::3], channels[::3], sample_rate, size=(10, 7), sca=0.8,

    # color theme
    cmap='tab20',  # default
);
```

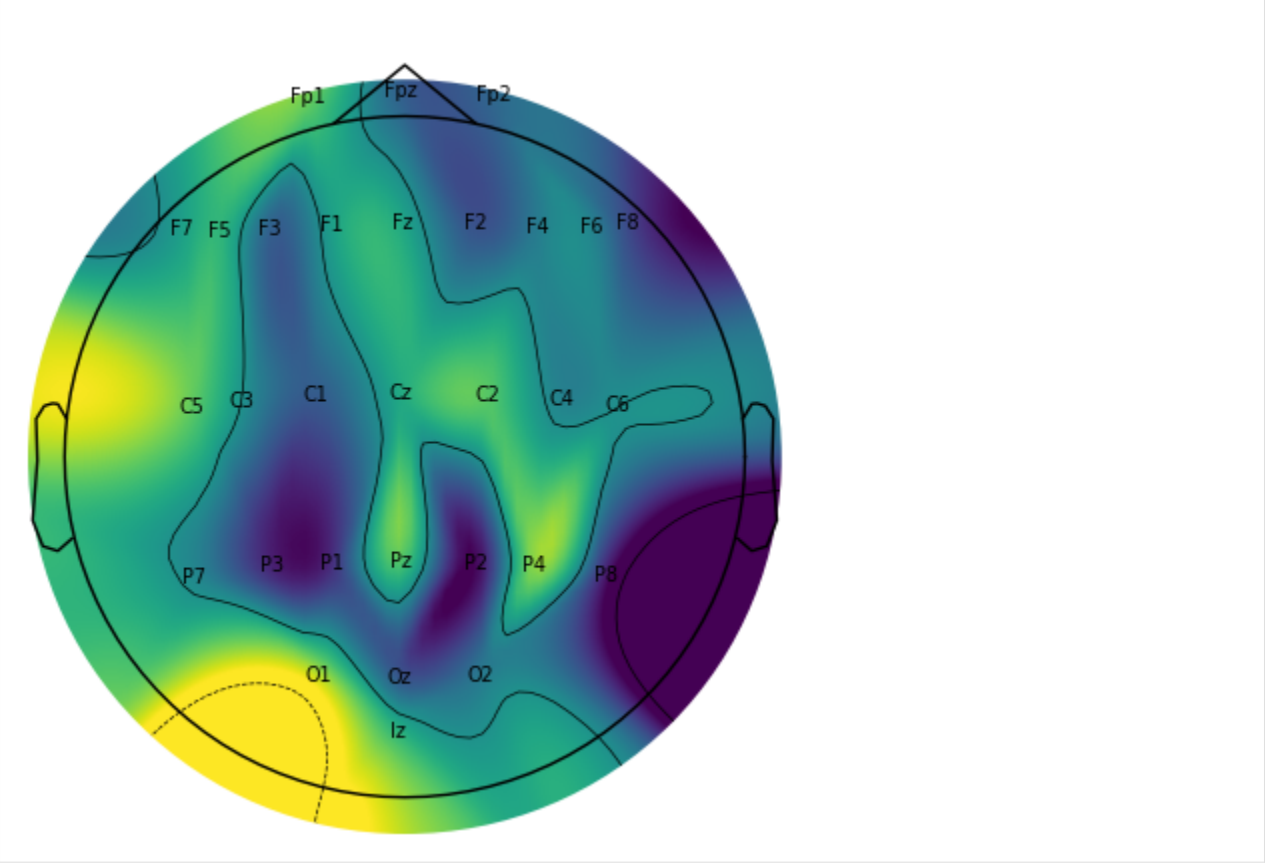


2.4 EEG topoplots

[3]:
```
from gcpds.visualizations.topoplots import topoplot
```

[4]:
```
channels = ['Fp1','Fpz','Fp2',
            'F7','F5','F3','F1','Fz','F2','F4','F6','F8',
            'C5','C3','C1','Cz','C2','C4','C6',
            'P7', 'P3','P1','Pz','P2','P4','P8',
            'O1','Oz','O2',
            'Iz']

data = np.random.normal(size=len(channels))
```

[5]:
```
ax = topoplot(data, channels, contours=3, cmap='viridis_r', names=channels,␣
↪sensors=False)
ax.get_figure();
```



[18]:
```
ax = topoplot(data, channels, cmap='viridis_r', res=10, sensors=False, show=False,␣
↪contours=0, outlines=None)
ax.get_images()[0].set_interpolation('none')
ax.get_figure();
```