Xueer Zhang HW4

In [ ]:
```
import plotly.graph_objects as go
import plotly.express as px
import pandas as pd
import numpy as np
```

In [23]:
```
# Load in the hierarchy information
url = "https://raw.githubusercontent.com/bcaffo/MRIcloudT1volumetrics/master/inst/extdata/multilevel_lookup_table.txt"
multilevel_lookup = pd.read_csv(url, sep = "\t").drop(['Level5'], axis = 1)
multilevel_lookup = multilevel_lookup.rename(columns = {
    "modify"   : "roi",
    "modify.1" : "level4",
    "modify.2" : "level3",
    "modify.3" : "level2",
    "modify.4" : "level1"})
multilevel_lookup = multilevel_lookup[['roi', 'level4', 'level3', 'level2', 'level1']]
multilevel_lookup.head()

# Now load in the subject data
id = 127
subjectData = pd.read_csv("https://raw.githubusercontent.com/smart-stats/ds4bio_book/main/book/assetts/kirby21AllLevels.csv")
subjectData = subjectData.loc[(subjectData.type == 1) & (subjectData.level == 5) & (subjectData.id == id)]
subjectData = subjectData[['roi', 'volume']]

# Merge the subject data with the multilevel data
subjectData = pd.merge(subjectData, multilevel_lookup, on = "roi")
subjectData = subjectData.assign(icv = "ICV")
subjectData = subjectData.assign(comp = subjectData.volume / np.sum(subjectData.volume))
subjectData.head()


# Create seperate tables that group by icv&level1, level1&level2, level2&level3, level3&level4
dat_icvl1 = subjectData.drop(['roi','volume','level4', 'level3', 'level2'],\
                      axis = 1)
dat_l1l2 = subjectData.drop(['roi','volume','level4', 'level3', 'icv'],\
                      axis = 1)
dat_l2l3 = subjectData.drop(['roi','volume','level4', 'level1','icv'],\
                      axis = 1)
dat_l3l4 = subjectData.drop(['roi','volume','level1', 'level2','icv'],\
                      axis = 1)
dat_icvl1.head()
dat_l1l2.head()
dat_l2l3.head()
dat_l3l4.head()
```

Out[23]:

	level4	level3	comp
0	SFG_L	Frontal_L	0.009350
1	SFG_R	Frontal_R	0.007270
2	SFG_L	Frontal_L	0.009247
3	SFG_R	Frontal_R	0.008324
4	SFG_L	Frontal_L	0.002227

In [27]:

```
t_list = [('icv','level1'),('level1','level2'),('level2','level3'),('lev
el3','level4')]
def df_to_sankey(df,cols_tuple_list):
    s = pd.DataFrame([])
    for t in cols_tuple_list:
        s1 = df.groupby(by=[t[0],t[1]],axis=0).count()
        s1 = s1.iloc[:,[0]]
        s1.columns = ['value']
        if s.shape[0]== 0:
            s = s1
        else:
            s = pd.concat([s,s1],axis=0)
    s.reset_index(inplace=True)
    s.columns = ['source','target','value']
    label_set = set(s['source'].unique()) | set(s['target'].unique())
    labels = {v: k for k, v in enumerate(label_set)}
    s.replace(labels, inplace=True)
    return s,list(label_set)

s,labels = df_to_sankey(subjectData[['icv','level1','level2','level3','l
evel4']],t_list)

fig = go.Figure(data=[go.Sankey(
   node = dict(
     pad = 3,
     thickness = 2,
     line = dict(color = "black", width = 1),
     label = labels,
   ),
   link = dict(
     source = s['source'].values,
     target = s['target'].values,
     value = s['value'].values
))])
```

In [28]:
```
fig.update_layout(title_text="Sankey", font_size=10)
 fig.show()
```

In [ ]:
```

```