



**SAPIENZA**  
UNIVERSITÀ DI ROMA

BIOINFORMATICS CLASS – PROJECT PART 1

---

**Neuroscience application:  
Brain network study during  
resting states**

---

GIULIA CASSARÀ

1856973

`cassara.1856973@studenti.uniroma1.it`

IVAN COLANTONI

1704031

`colantoni.1704031@studenti.uniroma1.it`

# 1 Introduction

Goal of this project is to analyze two datasets of EEG data recorded from 64 electrodes with subject at rest in (i) eyes-open and (ii) eyes-closed conditions, respectively.

Analyses will span the following topics:

1. connectivity graphs
2. graph theory indices
3. motif analysis
4. community detection

Task	Class
1.1	Mandatory
1.3	A
1.5	C
2.1	Mandatory
2.2	D
2.4	C
2.7	C
3.1	Mandatory
3.3	C
4.1	Mandatory
4.2	B
4.3	C

Table 1: List of tasks chosen for the project

# 2 Dataset

The EEG data are available from PhysioNet, “EEG Motor Movement/Imagery Dataset” [1]. The whole dataset contains data acquired from 109 subjects, each containing 14 runs (files) of acquisition. Only the first two runs (SxxxR01 and SxxxR02) are relevant for this project: R01 is recorded during eyes-open (EO) resting state; R02 is recorded during eyes-closed (EC) resting state. We selected the subject 79. Data is provided in EDF files (European Data Format). This format includes metadata, among which the sampling frequency and the channel labels. We have used a Python library Pyedflib [2] to read EDF file.

With the help of this library we return the signals for each channel in a numpy array, from which we obtain the adjacency matrix. We perform this step for both runs.

### 3 Connectivity graph

Functional brain connectivity represents functional associations among brain regions and can be obtained by measuring the temporal correlations between spatially remote neurophysiological events from fMRI and EEG/MEG data.

To get the connectivity graph we have used a Python module, Connectivipy [3]. We fit a Multivariate autoregressive model using Yule-Walker algorithm to our physical data and, once the coefficients of the MVAR model were adequately estimated, we pass to frequency domain and compute Partial Directed Coherence (PDC) estimation at 160 Hz sample rate, 100 number of spectrum data points, and a model order estimated from the previous model (in our case, it was 4).

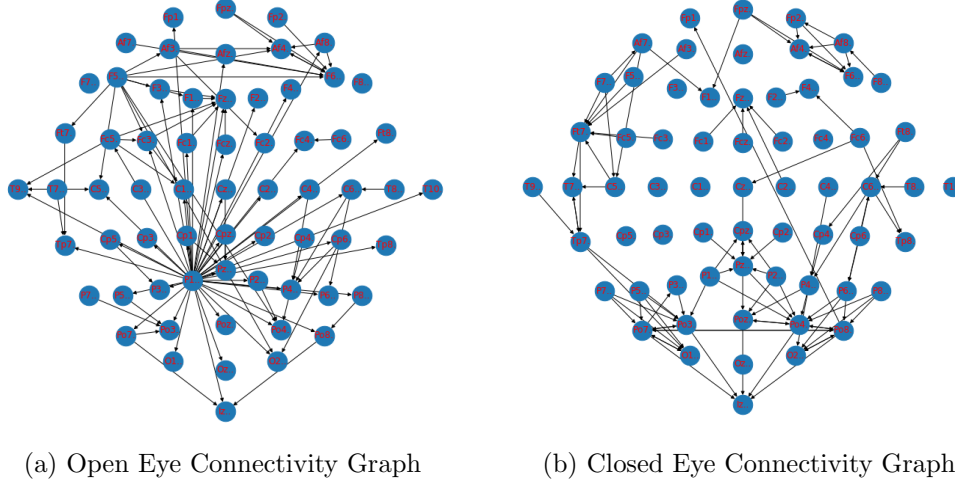
In order to get the adjacency matrix we must know in advance some parameters: density, the number of nodes, the number of edges and a threshold.

The density is defined as the actual number of connections within the model divided by its maximal capacity: we have tried with a density of 20%. At this point the number of edges is given by this equation [4]:

$$edges = \frac{density \times n \times (n - 1)}{2} \quad (1)$$

We then compute the threshold value in order to get the adjacency matrix which is composed by 1 if the element of the pdc matrix is greater than the threshold, otherwise 0. We set the diagonal to 0 by default.

We made a topographical representation of the networks with channels in positions specified in the file "channel\_locations.txt", along with the graphical representation of the binary adjacency matrix. We performed this task for the following density values: 1%, 5%, 10%, 20%, 30%, 50%, although we did the topographical representation only for densities set to 5% and 20%. (See the png files associated to this document). g

Figure 1: Topographical representation with density  $\leq 0.05$ 

## 4 Graph theory indices

From the adjacency matrix we build the graph structure using NetworkX library [5] in Python. We computed binary globals (clustering coefficient and path length) and locals (degree, in/out degree) graph indices. The file "graph\_indices.txt" contains a complete list of 10 highest channels for local indices and globals for open eye and closed eye graphs.

We computed the small-worldness index. A small-world network is a type of mathematical graph in which most nodes are not neighbors of one another, but the neighbors of any given node are likely to be neighbors of each other and most nodes can be reached from every other node by a small number of hops or steps. Network small-worldness has been quantified by a small-coefficient,  $\sigma$ , calculated by comparing clustering and path length of a given network to an equivalent random network with same degree on average [6].

$$\sigma = \frac{C/C_r}{L/L_r} \quad (2)$$

if  $\sigma > 1$  network is small-world.

Random network was generated from Erdős–Rényi model [7] giving different probabilities of building edges from vertices many times. Both graphs were found to be small-world in all cases.

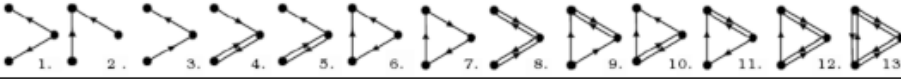
We also studied the behaviour of global indices in function of network density value. For densities  $< 0.15$  we were unable to compute globals because the graph was not weakly connected.

Finally, we computed global and locals of the weighted version of the graph where

the connections weights are provided by the connectivity estimator PDC that we used. In the file "graph\_indices.txt" it is contained all the results discussed so far in this paragraph.

## 5 Motif Analysis

In order to perform **Motif Analysis** we used iGraph Python package. We created the corresponding Igraph networks from the adjacency matrices obtained for tasks 1-2. The goal of point 3.1 is to find motifs and anti-motifs among all three-nodes configurations present in our networks. Motifs are those pattern which occurs with statistical significance and have supposed to be relevant for the detection of some functionalities. For the class of three-nodes isomorphism there are 13 different configuration to take into account (in the case of directed graphs). We firstly counted the presence of this configurations via the method *triad census*, performed on the algorithm defined by Davis and Leinhardt. It calculates how many times each isomorphism of three nodes occurs in the network. The results showed in Fig. 2 are compared to a set of  $M = 1000$  'Erdos Renyi' random networks.



<b>Motif Occurrence</b>	2053.0	1087.0	642.0	217.0	34.0	577.0	0.0	14.0	115.0	25.0	16.0	12.0	0.0
<b>Random Occurrence</b>	822.0	823.0	1645.0	182.0	182.0	182.0	61.0	10.0	10.0	10.0	20.0	2.0	0.0
<b>+ - SD</b>	45.0	44.0	81.0	36.0	35.0	18.0	8.0	5.0	4.0	4.0	6.0	2.0	0.0
<b>Z Score</b>	28.0	6.0	12.0	1.0	4.0	22.0	7.0	1.0	30.0	4.0	1.0	6.0	0.0

Figure 2: Eyes Open Motif Census

In order to detect a statistical significance we consider only the subgraphs which satisfy the metrics defined by [8] (Over Representation , Minimum Frequency, Minimum Deviation) with parameters  $[p, U, D, N] = [0.01, 4, 0.1, 1000]$  and assign for each motif a Z-score. The result is that subgraphs:  $[9, 1, 6, 12, 10, 4]$  appear to be motifs of our graph. While  $[3, 5, 7, 13]$  appear to be under-represented subnetworks and we classified it as anti-motifs.

<b>Motif Occurrence</b>	365	790	731	422	213	238	8	44	146	95	33	60	39
<b>Random Occurrence</b>	823	821	1646	182	182	183	61	10	10	10	20	2	0
<b>+ SD</b>	44	45	80	37	36	18	9	5	3	3	6	2	0
<b>Z Score</b>	11	1	11	7	1	3	6	6	39	25	2	34	168

Figure 3: Eyes Closed Motif Census

For the case of Eyes Closed graph, we applied the same method, the results are showed in Fig. 3. In that case the subnetworks [13, 9, 12, 10, 4, 8, 6] appeared to be over-represented and we classified it as motifs, with their relative Z-score. While [1, 3, 7] have to be considered as anti-motifs. The meaningful conclusion is that for the EC case, full subgraphs or almost full subgraphs appear more often than in EO, and in some cases motifs of one network appeared to be anti-motifs in the other one. For the point 3.3 we chose the channel 53 in the parieto-occipital scalp region and performed again *triad census* method to investigate the three-nodes configuration appearance. The result is that in this channel only configurations [1, 2, 6] are involved, respectively [1, 1, 2] times.

## 6 Community Detection

For the **Community Detection** we used the **Louvain Algorithm** implemented in the python-louvain external library in Python. [9]

In the code, we simply make a call to the `best_partition` function that computes the partition of the graph nodes which maximises the modularity (or try..) using the Louvain heuristics.

This is the partition of highest modularity, i.e. the highest partition of the dendrogram generated by the Louvain algorithm.

Curiously, every run of the algorithm finds four or five communities in both graphs. For graphical purpose, we decided to assign a color on the gray scale map to each node that belongs to a different community.

Out of curiosity we verified that there exist some channels that belongs to the same communities in both graphs: ['Fc5.', 'Fc3.', 'Fc1.', 'Fc4.', 'C3.', 'Fp1.', 'Fz.'].

We plotted the graphical representation of the two graphs. From this figures we can deduce that community seems to represent the different activated brain regions during the resting and open eye states.

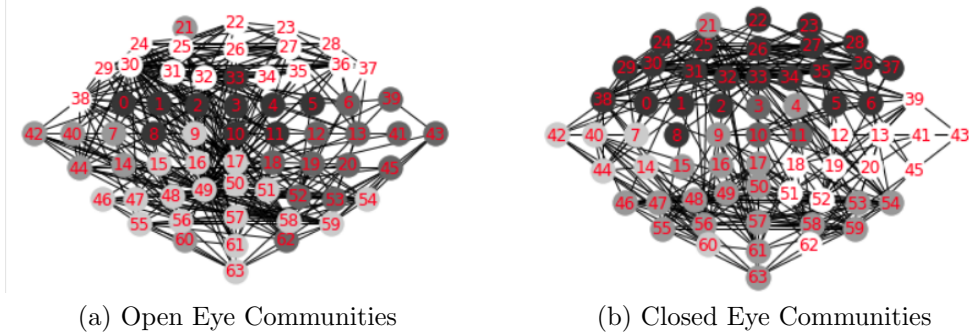


Figure 4: Community Detection with Louvain Algorithm

The file "channels.txt" contains a legend that maps numbers with their respective channels names.

We also tried to do community detection using an informative method like the Map Equation [10]. First, we tried to use the Infomap python package [11], unsuccessfully. The library detected zero communities in our network and also in the Zachary's Karate Club graph, used by the team as an example in the documentation [12]. We suppose that the package is broken. So we changed library using Python iGraph [13], a python wrapper of the iGraph R package. For the open eye graph it detected four communities, both in directed and undirected graph case. For the closed eye graph the infomap algorithm [11] detected three communities, both in directed and undirected graph case.

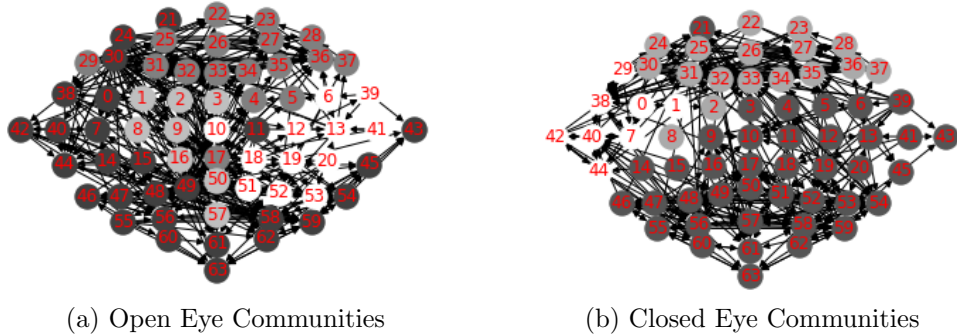


Figure 5: Community Detection with Infomap

## References

- [1] URL <https://physionet.org/content/eegmmidb/1.0.0/>.
- [2] Holgern. URL <https://github.com/holgern/pyedflib>.

- [3] dokato. URL <https://github.com/dokato/connectivitypy>.
- [4] Jorge J. (1983) Coleman, Thomas F.; Moré. Estimation of sparse jacobian matrices and graph coloring problems.
- [5] URL <https://networkx.github.io/>.
- [6] Humphries and Gurney. Network 'small-world-ness': A quantitative method for determining canonical network equivalence, 2008.
- [7] P.; Rényi Erdős. On random graphs. i, 1959.
- [8] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.
- [9] taynaud. URL <https://github.com/taynaud/python-louvain>.
- [10] URL <https://www.mapequation.org/>.
- [11] URL <https://mapequation.github.io/infomap/>.
- [12] URL <https://github.com/mapequation/infomap/blob/master/examples/python/example-networkx.py>.
- [13] URL <https://igraph.org/python/doc/igraph.Graph-class.html>.