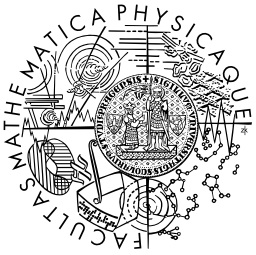


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

Kateřina Čížková

Comparing Two Main Community Detection Algorithms and Their Applications on Human Brains

Computer Science Institute of Charles University

Supervisor of the bachelor thesis: Mgr. Aneta Pokorná

Advisors: Ing. Mgr. Jaroslav Hlinka, Ph.D.
Ing. David Hartman, Ph.D.

Study programme: Computer Science

Study branch: General Computer Science

Prague 2022

I declare that I carried out this bachelor thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

Děkuji Mgr. Anetě Pokorné za vedení práce, všechny rady, průběžné konzultace, korektury všech verzí, včetně těch těsně před odevzdáním, a hlavně za to, že díky ní byla tvorba bakalářské práce mnohem příjemnější, než jsem si uměla představit. Také děkuji Ing. Davidu Hartmanovi, Ph.D. a Ing. Mgr. Jaroslavu Hlinkovi, Ph.D. za pomoc s výběrem zajímavého tématu, rady a konzultace, a panu Hlinkovi za korektury práce. Dále děkuji Matejovi a Páje za rady ohledně statistiky. Na závěr bych chtěla poděkovat své rodině a přátelům za jejich podporu (nejen) ve studiu.

Title: Comparing Two Main Community Detection Algorithms and Their Applications on Human Brains

Author: Kateřina Čížková

Department: Computer Science Institute of Charles University

Supervisor: Mgr. Aneta Pokorná, Computer Science Institute of Charles University

Abstract: Complex networks help us to understand complicated phenomena, including human brain. One of its key characteristics is its modular organization, also know as community structure. This thesis compares two main community detection algorithms, the Louvain algorithm and the label propagation algorithm. We prove some of their common properties and, on the other hand, we show that they make opposite decisions for certain graphs. The practical part of this thesis is devoted to community detection in human brain functional networks. It is known that the community structure of a human brain functional network changes during aging or due to some diseases. We compared the modularity and number of communities in functional networks of patients with the diagnosis of multiple sclerosis before and after a neurorehabilitation therapy. We did not find any significant change considering the whole dataset. However, the modularity changed in the functional networks of the six patients with a primary progressive course of multiple sclerosis. We show that there might be other minor changes in correlation with fMRI protocol or patients' gender.

Keywords: complex network, community detection, modularity, Louvain algorithm, label propagation, human brain

Contents

Introduction

We are surrounded by complex systems, including a society that requires cooperation between billions of individuals, intracellular protein interactions, a power grid that supplies energy for virtually all modern technology, or interactions of neurons in the human brain. It is nearly impossible to derive a collective behavior of a complex system just from the knowledge of its components. However, there are *complex networks* behind all these complex systems that encode the interactions between their components. [1]

Community is a cohesive group within a complex network. Roughly speaking, it is a part of the complex network in which nodes are more likely connected to each other than to nodes from other communities in the same network. Detection of communities helps us to understand the structure and dynamics of a complex network and, therefore, the underlying phenomenon.

There are plenty of community detection algorithms. They differ in resulting communities, running time, or ability to handle networks with weighted or directed edges. This work focuses on two of them, the *Louvain algorithm* and the *label propagation algorithm*. They are both among the fastest community detection algorithms, and they are used in practice. In the first chapter, we first introduce *modularity*, a network measure crucial for community detection. After that, we describe the algorithms and explain possible modifications. The second chapter is devoted to the comparison of the algorithms. We show some common properties and differences in the outputs of the algorithms.

An important field of network science is network neuroscience. Human brain can be represented as a complex network and studied from the perspective of graph theory. The third chapter reviews current knowledge about community structure and modularity changes in the *functional network*, which is used to investigate the neurophysiological dynamics of the brain. It also explains how to obtain the functional network using functional magnetic resonance imaging.

We applied the algorithms to real functional networks of patients with multiple sclerosis. Multiple sclerosis is an inflammatory disease attacking the brain. Previous studies reported changes in community structure and modularity caused by the disease. The second part of chapter three describes the course of the disease and summarizes the current knowledge about community structure and modularity in MS patients' functional networks.

The last chapter is devoted to experiments on human brain functional networks of patients diagnosed with multiple sclerosis. In the beginning, we introduce the dataset and network construction. Then we compare the results of the algorithms on the dataset. Afterwards, we use the algorithms to investigate the dataset itself, the difference between groups of patients regarding modularity, and the effect of neurorehabilitation therapy on modularity.

1. Community Detection

There is no universal definition of a community. However, there were some attempts to formally specify what a community is. The following two of them were provided by Radicchi et al. [2].

Definition 1 (Community in a Strong Sense)**.** *For an unweighted and undirected network, a subgraph of the network is a community in a strong sense if each node has more connections within the community than with the rest of the graph.*

Definition 2 (Community in a Weak Sense)**.** *For an unweighted and undirected network, a subgraph of the network is a weak community if the sum of all degrees within the subgraph is larger than the sum of all degrees toward the rest of the network.*

The notion of a community can be generalized to weighted networks. However, we must consider the meaning of weights in a particular network. Weights that indicate the strength of the connection, proximity, or similarity between vertices can give useful information about communities. On the other hand, there can also be other kinds of weights on edges that express the distance between vertices, i.e. the vertices are closer if the weight is smaller, or do not indicate any close relationship between the vertices. For example, if we use the traveling cost as an edge weight in a graph representing a transport system, it does not necessarily express the distance of route endpoints. Assuming that the weights express a proximity of vertices, the vertices connected with an edge with greater weight are more likely to be in the same community. Therefore we can consider the sum of weights of edges instead of the count of edges that is in Radicchi's definitions.

There are many approaches to the community detection problem. One can minimize cuts, maximize internal density, maximize the ratio between internal and external connections, etc. [3]. We will focus on two of them – maximization of *modularity*, a measure defined by Girvan and Newman in 2004 [4], and local optimization of the ratio between internal and external links. Specifically, we will focus on the multilayer modularity maximization Louvain algorithm and the label propagation algorithm.

1.1 Modularity

Modularity Q measures the density of connections within partitions in comparison to the density of connections between partitions. It is used to compare the quality of a partition of a network. It is defined as

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{i,j} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j). \tag{1.1}$$

Here, $A_{i,j}$ denotes the weight of the edge e_{ij} and $A_{i,i}$ is two times the weight of a self-loop e_{ii}. Note that for an undirected graph $A_{i,j} = A_{j,i}$. We denote the sum of the weights of the edges incident with vertex i as $k_i = \sum_y A_{i,y}$ and $m = \frac{1}{2} \sum_{i,j} A_{i,j}$ is the sum of weights of all edges in the graph. Kronecker delta function $\delta(c_i, c_j)$ is defined as 1 if $c_i = c_j$ and 0 otherwise, where c_i is the community of vertex i.

The sum iterates through vertices, but it will be useful to express modularity using a sum iterating through communities. We denote k_{i,in_C} the sum of weights of the edges incident to vertex i going to other vertices in C, so $k_{i,in_C} = \sum_{y,i\neq y} A_{i,y}\delta(C,c_y)$. We denote Σ_{in_C} the sum of weights of edges incident to vertices in community C going to vertices inside C. This can be expressed as

$$\Sigma_{in_C} = \sum_i (k_{i,in_C}\delta(C,c_i) + A_{i,i}) = \sum_{i,j} A_{i,j}\delta(C,c_i)\delta(C,c_j). \tag{1.2}$$

The sum of weights of the edges incident with vertices in a community C is denoted by $\Sigma_{tot_C} = \sum_i k_i\delta(C,c_i)$. Then, the modularity can also be formulated as

$$\begin{aligned}
Q &= \frac{1}{2m}\sum_C \left[\sum_{i,j}\left[A_{i,j} - \frac{k_i k_j}{2m}\right]\delta(C,c_i)\delta(C,c_j)\right] \\
&= \frac{1}{2m}\sum_C \left[\sum_{i,j} A_{i,j}\delta(C,c_i)\delta(C,c_j) - \frac{1}{2m}\sum_{i,j} k_i k_j \delta(C,c_i)\delta(C,c_j)\right] \\
&= \frac{1}{2m}\sum_C \left[\Sigma_{in_C} - \frac{1}{2m}\sum_i k_i\delta(C,c_i)\sum_j k_j\delta(C,c_j)\right] \\
&= \frac{1}{2m}\sum_C \left[\Sigma_{in_C} - \frac{1}{2m}\left(\Sigma_{tot_C}\right)\left(\Sigma_{tot_C}\right)\right] \\
&= \frac{1}{2m}\sum_C \left[\Sigma_{in_C} - \frac{1}{2m}\left(\Sigma_{tot_C}\right)^2\right].
\end{aligned} \tag{1.3}$$

1.1.1 Modularity Calculation – Example

Let us show an example of modularity calculation on a simple graph 1.1. Let us

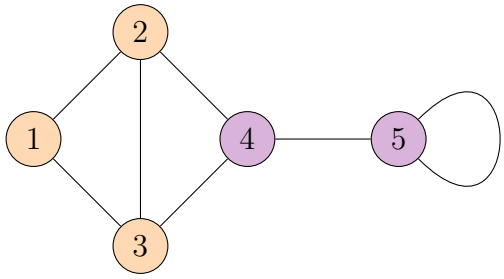


Figure 1.1: Graph G with communities $K = \{1,2,3\}$ and $L = \{4,5\}$

assume that all its edges have weight 1. The modularity of partition of G into communities K and L is

$$\begin{aligned}
Q &= \frac{1}{2m}\sum_C \left[\Sigma_{in_C} - \frac{1}{2m}\left(\Sigma_{tot_C}\right)^2\right] \\
&= \frac{1}{2m}\left[\left(\Sigma_{in_K} - \frac{1}{2m}\left(\Sigma_{tot_K}\right)^2\right) + \left(\Sigma_{in_L} - \frac{1}{2m}\left(\Sigma_{tot_L}\right)^2\right)\right] \\
&= \frac{1}{14}\left[\left((2+2+2) - \frac{1}{14}(2+3+3)^2\right) + \left((1+3) - \frac{1}{14}(3+3)^2\right)\right] = \frac{10}{49}
\end{aligned} \tag{1.4}$$

Note that the loop is counted twice because each edge is counted once for each end.

1.1.2 Properties of Modularity

The modularity value stays the same if we multiply all edge weights by a constant.

$$\begin{aligned} Q_k =& \frac{1}{\sum_{i,j} kA_{i,j}} \sum_C \left[\sum_{i,j} \left[kA_{i,j} - \frac{\sum_y kA_{i,y} \sum_y kA_{j,y}}{\sum_{i,j} kA_{i,j}} \right] \delta(C, c_i)\delta(C, c_j) \right] \\ =& \frac{1}{k\sum_{i,j} A_{i,j}} \sum_C \left[\sum_{i,j} \left[kA_{i,j} - \frac{k^2 \sum_y A_{i,y} \sum_y A_{j,y}}{k\sum_{i,j} A_{i,j}} \right] \delta(C, c_i)\delta(C, c_j) \right] \\ =& \frac{k}{k\sum_{i,j} A_{i,j}} \sum_C \left[\sum_{i,j} \left[A_{i,j} - \frac{\sum_y A_{i,y} \sum_y A_{j,y}}{\sum_{i,j} A_{i,j}} \right] \delta(C, c_i)\delta(C, c_j) \right] \\ =& \frac{1}{\sum_{i,j} A_{i,j}} \sum_C \left[\sum_{i,j} \left[A_{i,j} - \frac{\sum_y A_{i,y} \sum_y A_{j,y}}{\sum_{i,j} A_{i,j}} \right] \delta(C, c_i)\delta(C, c_j) \right] = Q \end{aligned} \tag{1.5}$$

If there is only one community containing all vertices, the modularity value is 0 because $\Sigma_{in_C} = \Sigma_{tot_C} = 2m$.

$$Q = \frac{\sum_C \Sigma_{in_C}}{2m} - \frac{\sum_C \left(\Sigma_{tot_C}\right)^2}{(2m)^2} = \frac{2m}{2m} - \frac{(2m)^2}{(2m)^2} = 0 \tag{1.6}$$

A single-vertex community without a loop, which is not an isolated vertex, always contributes with a negative value to modularity in a connected graph because Σ_{in_C} is 0 and Σ_{tot_C} is positive.

According to Brandes et al. [5], $-1/2 \leq Q < 1$ for unweighted and undirected networks. It also holds for weighted graphs; the proof follows the same structure.

Brandes et al. [5] also proved some interesting properties of maximum modularity:

- Clustering with maximum modularity has no cluster that consists of a single node with degree one.
- There always exists a clustering with maximum modularity, in which each cluster consists of a connected subgraph.
- Clustering of maximum modularity for clique of size n consists of a single cluster. For cycle of size n, the clustering of maximum modularity consists of approximately $\sqrt{n}$ clusters of size $\sqrt{n}$ each.

1.1.3 Limits of Modularity

Modularity optimization is an important technique in community detection, but we must be aware of some of its limitations.

Modularity Value

The modularity value of a partition does not carry much information because it may have very high values even for random graphs due to fluctuations. It is more interepretable when compared with the corresponding modularity expected for a random graph of the same size. [6]

The Resolution Limit

The resolution limit was described in detail by Fortunato and Barthélemy [7] for unweighted undirected networks and generalized by Berry et al. [8] for weighted networks. They claimed that the partition with the highest modularity does not necessarily capture the underlying community structure of the network because it merges small communities into larger ones.

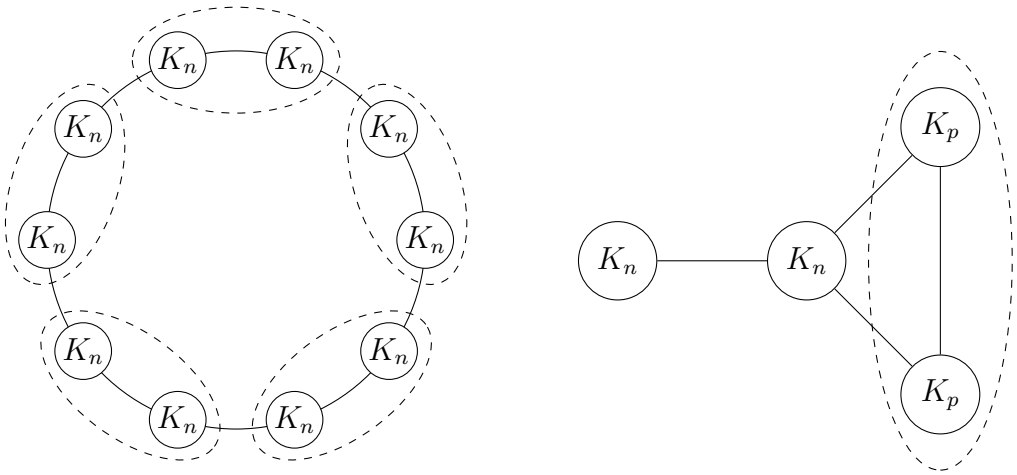
Let us have two communities C and D. The gain in modularity if we merge C and D together is

$$\begin{aligned}\Delta Q &= \frac{1}{2m}\left(\Sigma_{in_C} + 2e_{CD} + \Sigma_{in_D} - \frac{1}{2m}\left((\Sigma_{tot_C})^2 + 2\Sigma_{tot_C}\Sigma_{tot_D} + (\Sigma_{tot_D})^2\right)\right) \\ &\quad -\frac{1}{2m}\left(\left[\Sigma_{in_C} - \frac{1}{2m}(\Sigma_{tot_C})^2\right] + \left[\Sigma_{in_D} - \frac{1}{2m}(\Sigma_{tot_D})^2\right]\right) \\ &= \frac{1}{2m}\left(2e_{CD} - \frac{1}{2m}2\Sigma_{tot_C}\Sigma_{tot_D}\right) = \frac{1}{m}(e_{CD} - \frac{\Sigma_{tot_C}\Sigma_{tot_D}}{2m})\end{aligned} \tag{1.7}$$

where e_{CD} is a sum of weights of edges interconnecting C and D. We consider only the local change, the modularity of the rest of the network remains the same.

The gain should not be positive if C and D are two distinct communities because if it is positive and we maximize modularity, we should merge them. The problem is that it is positive in some cases.

The modularity change ΔQ is positive when $e_{CD} > \Sigma_{tot_C}\Sigma_{tot_D}/2m$. Assuming $\Sigma_{tot_C} \approx \Sigma_{tot_D} = \Sigma_{tot}$ we can derive the resolution limit $\Sigma_{tot} > \sqrt{2m \cdot e_{CD}}$. For unweighted network, $e_{CD} \geq 1$, thus $\Sigma_{tot} > \sqrt{2m}$. Therefore if $\Sigma_{tot} \leq \sqrt{2m \cdot e_{CD}}$, then modularity increases by merging C and D even if they are clearly distinct communities (e. g. cliques interconnected by one edge). [1]



(a) An unweighted network made out of identical cliques on n nodes, which are connected by single links. If the number of cliques is larger than about $\sqrt{2m}$, modularity is higher for partition consisting of pairs of cliques represented by dotted lines.

(b) An unweighted network with four pairwise identical cliques with n and p nodes, $p < n$, connected by single links. If n is large enough with respect to p (e.g., $n = 20$, $p = 5$), modularity is higher with the two smallest modules merged into one.

Figure 1.2: Resolution limit example by Fortunato and Barthélemy [7]

1.1.4 Mixing Parameter

Modularity is not the only network characteristic related to a community structure. *Mixing parameter* μ is another one, measuring how "clear" is the community structure in a specific network. It is defined as

$$\mu = \frac{\sum_i k_{i,out}}{\sum_i k_i} \tag{1.8}$$

where $k_{i,out}$ stands for the sum of weights of edges from a vertex i going to other communities than its own.

The intuition behind the definition is that if the mixing parameter is higher, the fraction of edges connecting communities is greater. Therefore the community structure is less clear and harder to detect.

Let us remind the Definition 1 of community in a strong sense. We see that for $\mu > 1/2$, communities in the strong sense tend to disappear. [9]

1.2 Louvain Method

There are many modularity maximization community detection methods. Louvain method was introduced by Blondel et al. in 2008 [10]. It is a hierarchical heuristic method based on a local maximization of modularity. The essential concepts of the method are fast evaluation of the change of modularity while moving a vertex from one community to another and contraction of communities. That are also the reason why it outperforms other modularity maximization algorithms.

1.2.1 Changes of Modularity

Let us show how to compute the gain in modularity obtained by moving a vertex from one community to another using the formulation of modularity (1.3). The loss in modularity $\Delta Q_{i,C\to i}$ obtained by moving a vertex i out of community C so it becomes an isolated vertex can be computed by (1.9). We subtract the modularity of the community C containing the vertex i from the modularity of C without i and an isolated vertex i. Let us remind that $A_{i,i}$ denotes two times the weight of a self-loop of the vertex i.

$$\begin{aligned}
\Delta Q_{i,C\to i} &= \frac{1}{2m}\left[\Sigma_{in_C} - 2k_{i,in_C} - A_{i,i} - \frac{(\Sigma_{tot_C} - k_i)^2}{2m} + A_{i,i} - \frac{k_i^2}{2m}\right] \\
&\quad - \frac{1}{2m}\left[\Sigma_{in_C} - \frac{(\Sigma_{tot_C})^2}{2m}\right] \\
&= \frac{1}{2m}\left[-2k_{i,in_C} - \frac{(\Sigma_{tot_C} - k_i)^2}{2m} - \frac{k_i^2}{2m} + \frac{(\Sigma_{tot_C})^2}{2m}\right] \\
&= \frac{1}{2m}\left[-2k_{i,in_C} - \frac{(\Sigma_{tot_C})^2 - 2\Sigma_{tot_C}k_i + k_i^2 + k_i^2 - (\Sigma_{tot_C})^2}{2m}\right] \\
&= \frac{1}{2m}\left[\left(\frac{k_i}{m}\right)(\Sigma_{tot_C} - k_i) - 2k_{i,in_C}\right]
\end{aligned} \tag{1.9}$$

The gain in modularity $\Delta Q_{i,i\to D}$ obtained by moving an isolated vertex i to a community D can be computed similarly. We subtract the modularity of the community D and the vertex i from the modularity of the community D including i.

$$\begin{aligned}
\Delta Q_{i,i\to D} &= \frac{1}{2m}\left[\Sigma_{in_D} + A_{i,i} + 2k_{i,in_D} - \frac{(\Sigma_{tot_D}+k_i)^2}{2m}\right] \\
&\quad - \frac{1}{2m}\left[\Sigma_{in_D} - \frac{(\Sigma_{tot_D})^2}{2m} + A_{i,i} - \frac{k_i^2}{2m}\right] \\
&= \frac{1}{2m}\left[2k_{i,in_D} - \frac{(\Sigma_{tot_D}+k_i)^2}{2m} + \frac{(\Sigma_{tot_D})^2}{2m} + \frac{k_i^2}{2m}\right] \\
&= \frac{1}{2m}\left[2k_{i,in_D} - \frac{(\Sigma_{tot_D})^2 + 2\Sigma_{tot_D}k_i + k_i^2 - (\Sigma_{tot_D})^2 - k_i^2}{2m}\right] \\
&= \frac{1}{2m}\left[2k_{i,in_D} - \left(\frac{k_i}{m}\right)\Sigma_{tot_D}\right]
\end{aligned} \tag{1.10}$$

This gives the formula (1.11) for computing the modularity change of moving vertex i from a community C to a community D.

$$\begin{aligned}
\Delta Q_{i,C\to D} &= \Delta Q_{i,C\to i} + \Delta Q_{i,i\to D} \\
&= \frac{1}{2m}\left[\left(\frac{k_i}{m}\right)(\Sigma_{tot_C} - k_i) - 2k_{i,in_C}\right] + \frac{1}{2m}\left[2k_{i,in_D} - \left(\frac{k_i}{m}\right)\Sigma_{tot_D}\right] \\
&= \frac{1}{2m}\left[\left(\frac{k_i}{m}\right)(\Sigma_{tot_C} - k_i) - 2k_{i,in_C} + 2k_{i,in_D} - \left(\frac{k_i}{m}\right)\Sigma_{tot_D}\right] \\
&= \frac{1}{2m}\left[\left(\frac{k_i}{m}\right)(\Sigma_{tot_C} - \Sigma_{tot_D} - k_i) - 2(k_{i,in_C} - k_{i,in_D})\right]
\end{aligned} \tag{1.11}$$

Changes of Modularity – Example

Here we show on graph G from Figure 1.1 how modularity changes when we move vertex 4 from community L to community K. The first step is to move the vertex out of community L as in Figure 1.3.

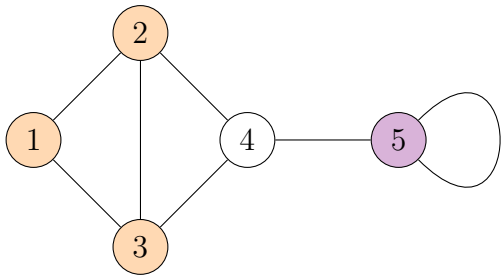


Figure 1.3: Graph G with communities $K = \{1, 2, 3\}$, $L = \{5\}$ and $M = \{4\}$

$$\Delta Q_{4,L\to M} = \frac{1}{2m}\left[\left(\frac{k_i}{m}\right)(\Sigma_{tot_L} - k_i) - 2k_{i,in_L}\right] = \frac{1}{14}\left[\left(\frac{3}{7}\right)(6-3) - 2\right] = -\frac{5}{98}$$

The modularity of a graph G from Figure 1.3 with communities K, L, M is 15/98 which corresponds to

$$Q + \Delta Q_{4,L\to 4} = \frac{10}{49} - \frac{5}{98} = \frac{15}{98}$$

where Q is the modularity (1.4) of the original graf G.

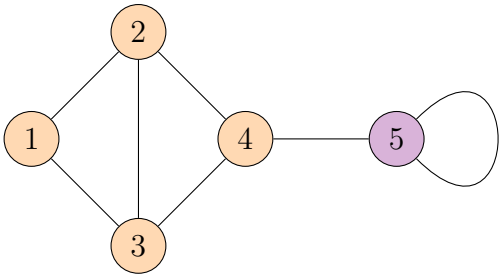


Figure 1.4: Graph G with communities $K = \{1, 2, 3, 4\}$ and $L = \{5\}$

The second step is to move the vertex 4 into community K as shown in Figure 1.4. The change of modularity is calculated as

$$\Delta Q_{4,M\to K} = \frac{1}{2m}\left[2k_{i,in_K} - \left(\frac{k_i}{m}\right)\Sigma_{tot_K}\right] = \frac{1}{14}\left[2\cdot 2 - \left(\frac{3}{7}\right)8\right] = \frac{2}{49}.$$

The overall change of modularity while moving vertex 4 from community L to community K is calculated using (1.11) as follows:

$$\begin{aligned}\Delta Q_{4,L\to K} &= \frac{1}{2m}\left[\left(\frac{k_i}{m}\right)(\Sigma_{tot_L} - \Sigma_{tot_K} - k_i) - 2(k_{i,in_L} - k_{i,in_K})\right] \\ &= \frac{1}{14}\left[\left(\frac{3}{7}\right)(6 - 8 - 3) - 2(1 - 2)\right] = -\frac{1}{98} \\ &= -\frac{5}{98} + \frac{1}{49} = \Delta Q_{4,L\to M} + \Delta Q_{4,M\to K}\end{aligned}$$

The modularity of G with the new communities from Figure 1.4 is

$$\frac{10}{49} - \frac{1}{98} = \frac{19}{98}$$

.

Now, let us also move vertex 5 from community L to community K:

$$\begin{aligned}\Delta Q_{5,L\to K} &= \frac{1}{2m}\left[\left(\frac{k_i}{m}\right)(\Sigma_{tot_L} - \Sigma_{tot_K} - k_i) - 2(k_{i,in_L} - k_{i,in_K})\right] \\ &= \frac{1}{14}\left[\left(\frac{3}{7}\right)(3 - 11 - 3) - 2(0 - 1)\right] = -\frac{19}{98}\end{aligned}$$

If we calculate the modularity of G with only one community containing all vertices, it is 0. It corresponds to the change of modularity while moving vertex 5 to community K because

$$\frac{19}{98} - \frac{19}{98} = 0$$

.

1.2.2 Louvain Algorithm

The algorithm proceeds as follows: Initially, every node forms its own community. The algorithm consists of two phases repeated iteratively. It is shown in a pseudo-code of Algorithm 1. We write $\sum\{i,j\}$ instead of $\sum_{i,j}$ and $k\{v\}$ ($k\{v, in_C\}$) instead of k_v (k_{v,in_C}) for readability.

Algorithm 1 Louvain algorithm – main function

```
1: function LOUVAIN(weighted graph G = (V, E))
2:     m ← (1/2) Σ{i,j} A_{i,j}
3:     for all v ∈ V do
4:         v.comm ← v                          ▷ initially each node in its own community
5:         tot[v.comm] ← Σ{i} e_{vi}.weight    ▷ e_{vi}.weight is a weight of edge e_{vi}
6:     end for
7:     Q ← (1/2m) Σ{v ∈ V} (e_{vv}.weight − (1/2m)(tot[v.comm])^2)
8:                                                ▷ initial modularity
9:     while modularity is increasing do
10:        Q ← MAXIMIZE-MODULARITY(G, Q, m)
11:        G ← COMMUNITIES-TO-GRAPH(G)
12:    end while
13: end function
```

During the first phase, we obtain a local maximum of the modularity by moving the individual vertices to the communities of their neighbors. See a pseudo-code of Algorithm 2. For each vertex i, we compute the gain in modularity $\Delta Q_{i,C\to D}$ coming from moving i from its community C to the community D of its neighbor j. We pick the community with the largest positive ΔQ with ties broken uniformly randomly. The community assignment is updated immediately. If no positive gain in modularity is possible, i remains in its own community C.

When no individual move can improve the modularity, we build a new network whose nodes are now the communities found during the first phase. There is one node created for each community. The weight of a self-loop of a vertex corresponding to a community C is Σ_{in_C} where C is the community – the sum of weights of self-loops of all vertices in C plus two times the weight of all edges interconnecting vertices in C. The weight of an edge between two new nodes (communities in the old graph) is the sum of the weights of edges between these two communities. [10] Algorithm 3 shows pseudo-code for this step.

It was proven by Brandes et al. [11] that modularity maximization for an unweighted undirected graph is an NP-complete problem. It follows that it is also NP-complete for a weighted undirected graph. However, the algorithm is fast for real networks; its running time is linear on typical and sparse data. The computation of ΔQ is fast due to the formula (1.11), and the number of nodes in the meta-graph decreases at each iteration, so most of the computing time is spent on the first iterations. We can also decrease the overall running time by stopping the first phase when ΔQ does not exceed some threshold. This modification was used in the original paper [10] to obtain all their numerical results.

Algorithm 2 Louvain algorithm – modularity maximization

```
14: function MAXIMIZE-MODULARITY(weighted graph G = (V, E), modularity Q, the sum of weights of all edges m)
15:     for all v ∈ V do
16:         k{v} ← ∑{i} e_vi.weight
17:     end for
18:     while modularity is increasing do
19:         for all v ∈ V do
20:             k{v, in_v.comm} ← ∑{i, i ≠ v, i.comm = v.comm} e_vi.weight
21:             ΔQ_best ← 0
22:             c_best ← v.comm
23:             for all n ∈ v.neighbors do
24:                 k{v, in_n.comm} ← ∑{i, i ≠ v, i.comm = n.comm} e_vi.weight
25:                 ΔQ ← 1/2m · ((k{v}/m)(tot[v.comm]−tot[n.comm] − k{v})
26:                                 −2(k{v, in_v.comm} − k{v, in_n.comm}))
27:                 if ΔQ > ΔQ_best then
28:                     ΔQ_best ← ΔQ
29:                     c_best ← n.comm
30:                 end if
31:             end for
32:             if ΔQ_best > 0 then
33:                 Q ← Q + ΔQ_best
34:                 tot[v.comm] ←tot[v.comm] − k{v}
35:                 tot[c_best] ←tot[c_best] + k{v}
36:                 v.comm ← c_best
37:             end if
38:         end for
39:     end while
40:     return Q
41: end function
```

1.2.3 Modifications of the Louvain Algorithm

There have been many attempts to improve the Louvain algorithm. Some of them are mentioned in this section.

Campigotto et al. [12] provided a generalized method for community detection based on the Louvain algorithm. Modularity is the most popular function for evaluating the quality of a partition, but there are many others. The authors claim that any linear quality function can be implemented efficiently in the generalized Louvain algorithm.

Traag et al. [13] came up with a modification called the Leiden algorithm, which guarantees well-connected communities. If there is a node that acts as a bridge between different parts of its community, it may be moved to a different community by the classical Louvain algorithm. Removing such a node from its community disconnects the community. However, the other nodes in the community may still be sufficiently strongly connected inside their parts, so they stay in the community, and the community stays disconnected until the end of the algorithm. The Leiden algorithm adds one more step between the local

Algorithm 3 Louvain algorithm – communities contraction

function COMMUNITIES-TO-GRAPH(weighted graph $G = (V, E)$)
  $V_{new} \leftarrow$ one vertex per community *comm* in the graph
    ▷ its community *comm* is the community *comm* of the original vertices
  $E_{new} \leftarrow \emptyset$
  for all $u \in V_{new}$ **do**
    for all $v \in V_{new}$ **do**
      if $u = v$ **then**
        $e_{uv}.weight \leftarrow \sum\{i, j \in V, i \neq j, i.comm = u.comm,$
                $j.comm = u.comm\}\, e_{ij}.weight \cdot 2$
            $+ \sum\{i \in V, i.comm = u.comm\}\, e_{ii}.weight$
      else
        $e_{uv}.weight \leftarrow \sum\{i, j \in V, i \neq j, i.comm = u.comm,$
                $j.comm = v.comm\}\, e_{ij}.weight$
      end if
      $E_{new} \leftarrow E_{new} \cup e_{uv}$
    end for
  end for
  return (V_{new}, E_{new}) ▷ contracted graph G_{new}
end function

modularity maximization and the network contraction. The communities found in the first step are refined, so one community could be represented by more than one node in the aggregated network. Each node in the new network represents a well-connected part of a community. Although the Leiden algorithm is considerably more complex than the Louvain algorithm, the authors showed that for larger networks and higher mixing parameter the Leiden algorithm is much faster than the Louvain algorithm and it finds better partitions for higher mixing parameter.

Another improved version of the Louvain algorithm was introduced recently by Zhang et al [14]. The authors reported slightly higher average modularity than obtained with the classical Louvain algorithm for the same graphs in a significantly shorter time. They did two main modifications. Firstly, they handle local tree structures separately. The local tree structure consists of leaf nodes with only one neighbor and other nodes, which become leaf nodes if we repeatedly remove the current leaves. The authors remove all local tree structures in the network before the modularity maximization iteration, partition the remaining network, and finally add the tree structures to the partition result. Secondly, they did not check the gain in the modularity of node movement between two communities if the communities did not change until the last check. This saves computation time while the result remains the same.

1.3 Label Propagation

The label propagation algorithm (LPA) was introduced by Raghavan et al. [15] in 2007. The idea is straightforward. Every node is initialized with a unique label. We iterate through all nodes in random order. Each node adopts the label which can be found on the majority of its neighbors. If more than one label has

the maximum frequency, one is chosen uniformly at random. When every node has the label that the maximal number of their neighbors have, densely connected groups of nodes form communities with the same labels. By construction, the number of neighbors of each vertex in its community is greater than in any other community. Although the definition is stricter because each vertex must have more neighbors in its community than in the rest of the network, it resembles Definition 1, the strong definition of community. [16] A pseudo-code for the label propagation algorithm is given in Algorithm 4.

Algorithm 4 Label propagation algorithm

```
1:  function LABEL-PROPAGATION(graph G = (V, E))
2:      for all v ∈ V do
3:          v.label ← v                    ▷ initially each node in its own community
4:      end for
5:      while labels are changing do
6:          for all v ∈ V in random order do      ▷ synchronous or asynchronous
7:              v.label ← arg max{l ∈ labels} Σ{u ∈ v.neighbors, v.label = l} 1
8:                                  ▷ edge weight instead of 1 for weighted variation
9:          end for
10:     end while
11: end function
```

The label updating process can be executed synchronously or asynchronously. In a synchronous setting, the new label of a node is determined based on the old labels of the neighbors, and then all the nodes are updated. In an asynchronous variation, when a label of node i is updated and then neighbor j of i is processed, the new label of i is already considered. The authors of the original paper used asynchronous updating because the synchronous approach leads to oscillations for bipartite or nearly bipartite structures in networks. [15]

The running time of the label propagation algorithm is near-linear. Each iteration takes linear time in the number of edges. The authors claim that their experiments show that irrespective of the number of nodes, 95% of the nodes or more have a label that the maximum of its neighbors has by the end of iteration 5.

1.3.1 Modifications of the LPA

Similar to the Louvain algorithm, many modifications of the label propagation algorithm have been suggested.

First, an easy alternation allows the algorithm to use weights. The only change is mentioned in the pseudo-code of Algorithm 4 in a comment on line 8. We consider the sum of weights of edges instead of their count.

Result of the LPA is highly affected by the randomness in breaking ties and in the ordering of nodes which is known as *label propagation strategy*. Non-random orderings have been designed, usually in combination with modified update rules. [3]

We can use smarter initialization than just assigning a unique label to each vertex. If we have prior knowledge about the communities, we can initially assign labels accordingly. We can also search for a particular structure in the network

(like some specific seed nodes or small subgraphs) and use it for the initialization. Another option is to assign particular labels manually; variants of this strategy are known as semi-supervised. [3]

The original LPA formulation is essentially local. However, there are modifications focusing on a definition of neighborhood, taking into account not only the directly adjacent nodes. These modifications can find better results but at the cost of increasing time complexity.

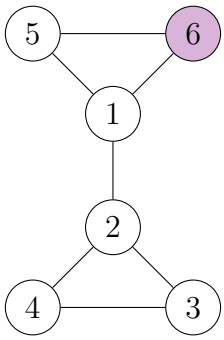
1.3.2 Limits of LPA

Monster Community

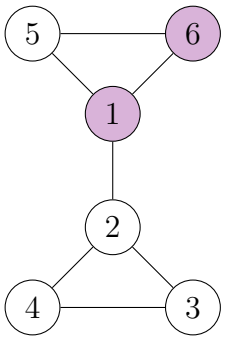
The LPA often results in a partition consisting of a single community. Such a dominant community is called a *monster community*.

Sometimes it indicates that the network is indivisible because its edges are so dense that there is no community structure. A complete graph is an example of a single community network, and a monster community obtained by the LPA is meaningful in this case. [17]

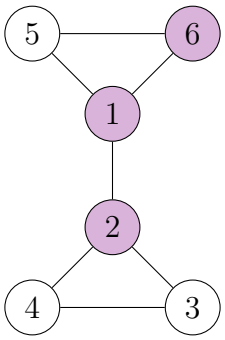
However, in some cases the trivial solution means that the detection process failed. There is a community structure in the network, but the LPA cannot find it. See a simple example of monster community occurrence in the LPA in Figure 1.5.



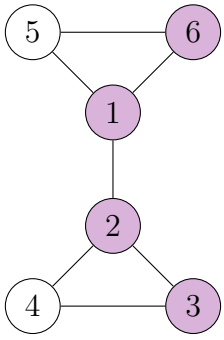
(a) Initially, each vertex has a different label



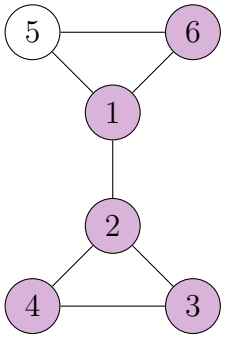
(b) Node 1 randomly adopts label of node 6



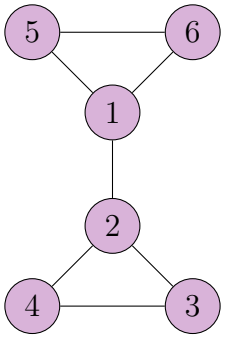
(c) Node 2 randomly adopts label of node 1



(d) Node 3 randomly adopts label of node 2



(e) Node 4 adopts label of nodes 2 and 3



(f) Node 5 adopts label of nodes 1 and 6

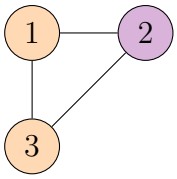
Figure 1.5: Monster community example. White color denotes that the node has its own initial label. Initial label of node 6 is marked with purple and it is propagated through the network.

1.4 Igraph

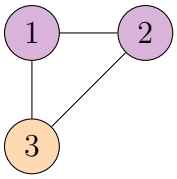
For all the experiments later, implementations of both Louvain and the label propagation algorithm from Igraph library `https://igraph.org/` were used. Igraph is free and open source network analysis tool with backend written in C. It can be installed as a Python package using `pip`. The algorithms are implemented there in both the unweighted and weighted versions.

I found an issue in the LPA implementation in Igraph 0.9.6. The iterative process of the LPA shloud be performed until every node in the network has a label to which the maximum number of its neighbors belongs. The problem was, that this implementation sometimes finished earlier because the goal condition was checked in the same iteration as the labels were changed.

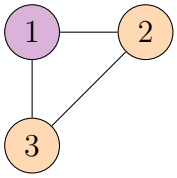
If at least one label was changed from non-dominant to dominant neighboring label, it suggested that another iteration is needed. Otherwise, the program finished. That caused the problem, because even if a node fulfiled the goal condition when it was processed, the dominant neighboring label could have changed later, if its neighbor had more dominant neighboring labels, and one was chosen randomly. Figure 1.6 shows an example of such unwanted behavior.



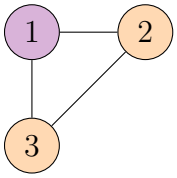
(a) New iteration starts, two nodes have the same label



(b) Node 1 randomly adopts label of node 2



(c) Node 2 randomly adopts label of node 3



(d) Node 3 randomly adopts label of node 2

Figure 1.6: Igraph bug example. After this iteration, the program finished, because the labels never changed from non-dominant to dominant. However, the goal condition is violated, node 1 does not have the same label as the majority of its neighbors.

I fixed the bug by separating the relabeling and the goal checking process. The patch was already merged, so the algorithm works correctly in Igraph 0.9.7 and newer.

2. Comparison of Community Detection Algorithms

There is no universal method for comparing the output quality of community detection algorithms. There are several reasons for that. There is no generally accepted definition of a community, and it can depend on the meaning of nodes and edges in a specific network. Therefore the communities resulting from different algorithms could be meaningful for different interpretations of the network, even if its topology is still the same. We want to estimate a community structure of a network, but the underlying community structure has not been clearly defined, so we do not have any *ground truth*, a reference solution to compare with. The algorithms can also be compared in terms of a running time in a typical situation. Both the Louvain and the label propagation algorithms are considered to be near-linear.

Community detection algorithms can also be compared experimentally on various real-world or computer-generated networks (see Section 2.2 for details), although again the ground truth concerning the correct community definition is not necessarily obvious, albeit in some systems it can be substituted by domain expert evaluation.

2.1 Theory

This section contains a theoretical comparison of the Louvain and the label propagation algorithm. We start by showing some differences and then we are discussing common properties of the algorithms.

2.1.1 Differences

In this section, we discuss what distinguishes the algorithms from each other. It is evident that the label propagation algorithm does not contract communities as the multilevel Louvain algorithm does. However, even if we consider only the first modularity-maximization phase of the Louvain algorithm, the algorithms could make opposite decisions in certain situations, as shown below. It is caused by the difference in the update rule changing the community assignment for each vertex.

The label propagation algorithm considers only direct neighbors of each node, neglecting the size or structure of the neighboring communities. On the other hand, the Louvain algorithm consideres the whole structure of the neighboring communities, and the sum of weights of all edges also plays a role in the modularity gain formula (1.11). That is caused by the fact that the Louvain algorithm maximizes modularity and the label propagation algorithm localy maximizes only the first term in the modularity definition (1.1).

Opposite Decisions

Let us show an example of where the algorithms make opposite decisions. We have a graph G consisting of two cliques K_4 and K_5 with one shared vertex v. The graph has two intuitive divisions to communities varying in an assignment of the shared vertex v to K_4 or K_5.

The Louvain algorithm can not end up with v assigned to community K_5 (Figure 2.1). The modularity of this partition is ≈ 0.217, while the modularity of a partition with v assigned to the community of K_4 (Figure 2.2) is 0.25. The end condition of the Louvain algorithm is not met in the former case because we can achieve a positive gain in modularity moving v to the other community.

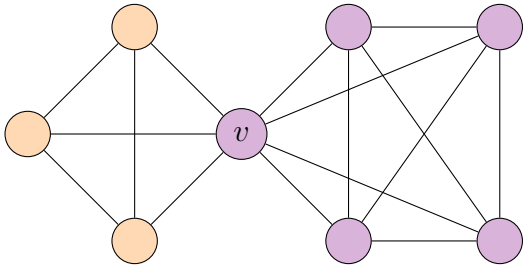


Figure 2.1: Graph G with two communities, modularity ≈ 0.217

On the other hand, the label propagation algorithm can not end up with v assigned to K_4. There are obviously more edges from v to K_5 than to K_4.

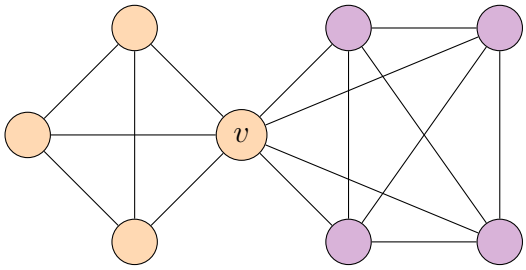


Figure 2.2: Graph G with two communities, modularity 0.25

It is possible to generalize the counterexample to all unweighted graphs consisting of two cliques of different sizes sharing one vertex v.

Proposition 1. *Let us have an unweighted graph consisting of two cliques of different sizes sharing one vertex v. Each of the cliques forms a community. In this situation, the Louvain algorithm and the label propagation algorithm assign the shared vertex v to different community.*

Proof. Let us denote the smaller community S and the greater one G. They have s and g nodes, $2 < s < g$. The label propagation algorithm always places the shared vertex to the community of the greater clique, because there are more edges from v to the greater clique than to the smaller one. However, the Louvain algorithm does the opposite.

Assume for contradiction that the shared vertex is a part of the larger clique community and the Louvain algorithm finishes, so it is not possible to obtain

higher modularity by moving the shared vertex to the smaller clique community.

$$\Delta Q_{v,G\to S} = \frac{1}{2m}\left[\left(\frac{k_v}{m}\right)(\Sigma_{tot_G} - \Sigma_{tot_S} - k_v) - 2(k_{v,in_G} - k_{v,in_S})\right] < 0$$
$$k_v\,(\Sigma_{tot_G} - \Sigma_{tot_S} - k_v) < 2m(k_{v,in_G} - k_{v,in_S}) \tag{2.1}$$

Let us express the individual terms of the equation using s and g.

$$\begin{aligned} 2m &= s(s-1) + g(g-1) \\ \Sigma_{tot_G} &= g(g-1) + (s-1) \\ \Sigma_{tot_S} &= s(s-1) - (s-1) \\ k_v &= k_{v,in_S} + k_{v,in_G} = (s-1) + (g-1) \end{aligned}$$

We plug it in the inequality (2.1).

$$\begin{aligned} \Big[(s-1)+(g-1)\Big]\Big(&\Big[g(g-1)+(s-1)\Big] \\ &- \Big[s(s-1)-(s-1)\Big] - \Big[(s-1)+(g-1)\Big]\Big) \\ &< \Big[s(s-1)+g(g-1)\Big]\Big[(g-1)-(s-1)\Big] \end{aligned}$$

We can simplify the inequality as follows.

$$\begin{aligned} &\Big[(s-1)+(g-1)\Big]\Big[g(g-1)-s(s-1)+s-g\Big] \\ &\qquad < \Big[s(s-1)+g(g-1)\Big]\Big[g-s\Big] \\ &\Big[g(g-1)(s-1) - s(s-1)(g-1) + g(g-1)^2 - s(s-1)^2 + s(s-1) \\ &\qquad - g(s-1) + s(g-1) - g(g-1)\Big] \\ &\qquad < \Big[gs(s-1) + g^2(g-1) - s^2(s-1) - sg(g-1)\Big] \\ &\Big[(g-s)(g-1)(s-1) + g(g-1)^2 - s(s-1)^2 + (s-g)(s-1) + (s-g)(g-1)\Big] \\ &\qquad < \Big[gs(s-g) + g^2(g-1) - s^2(s-1)\Big] \\ &\Big[(g-s)(g-1)(s-1) + g(g-1)^2 - s(s-1)^2 + (s-g)(s+g-2)\Big] \\ &\qquad < \Big[gs(s-g) + g^2(g-1) - s^2(s-1)\Big] \end{aligned}$$

Let us rearrange the left and right sides of the inequality.

$$\begin{aligned} &(g-s)(g-1)(s-1) + (s-g)(s+g-2) - gs(s-g) \\ &\qquad < g^2(g-1) - g(g-1)^2 - s^2(s-1) + s(s-1)^2 \\ &(g-s)(g-1)(s-1) + (s-g)(s+g-2-gs) \\ &\qquad < g(g-1)\Big[g-(g-1)\Big] + s(s-1)\Big[(s-1)-s\Big] \\ &(g-s)\Big[(g-1)(s-1) + (gs-s-g+2)\Big] \\ &\qquad < g(g-1) - s(s-1) \end{aligned}$$

In the following steps, we expand the brackets.

$$\begin{aligned}
(g-s)\Big[(g-1)(s-1)+(gs-s-g+2)\Big]-g(g-1)+s(s-1)&<0\\
(g-s)\Big[2gs-2s-2g+3\Big]-g^2+g+s^2-s&<0\\
2g^2s-2sg-2g^2+3g-2gs^2+2s^2+2gs-3s-g^2+g+s^2-s&<0\\
2g^2s-3g^2+4g-2gs^2+3s^2-4s&<0\\
2gs(g-s)-3(g^2-s^2)+4(g-s)&<0\\
(g-s)(2gs+4-3(g+s))&<0
\end{aligned}$$

We can divide the inequality by $(g-s)$, because $g > s$, so the term is positive. Then, we us use the fact that $s > 2$.

$$0 > 2gs+4+3s+3g > 4g+4+6+3g = 7g+10$$

It is a contradiction, $7g+10 > 0$ because g is positive.

□

Community with a Negative Contribution to Modularity

The label propagation algorithm can finish with a partition containing a community that contributes negatively to the resulting modularity. Figure 2.3 shows an example, such a community is denoted with purple.

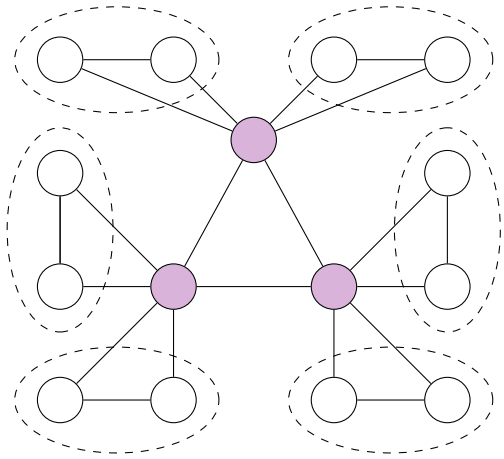


Figure 2.3: Unweighted graph with paritition with a community that contributes negatively to modularity. The community C with negative contribution to modularity is marked with purple, the other communities are marked with dashed ellipses, one ellipsis denotes one community.

The situation in Figure 2.3 is valid result of the label propagation algorithm. The label propagation process started with the purple label propagation in the purple community C. Then, the nodes outside of the purple community always had one neighbor in the purple community and one in other one. All of them randomly chose the non-purple.

The contribution of the community C to modularity is calculated as

$$Q_C = \frac{1}{2m}\left[\Sigma_{in_C} - \frac{1}{2m}\left(\Sigma_{tot_C}\right)^2\right].$$

Let us plug in the values $\Sigma_{in_C} = 6$, $\Sigma_{tot_C} = 18$ and $2m = 42$, then

$$Q_C = \frac{1}{42}\left[6 - \frac{1}{42}18^2\right] = \frac{1}{42}\left[\frac{256 - 324}{42}\right] < 0.$$

Now we show that the Louvain algorithm can not return a community with a negative contribution to modularity.

Proposition 2. *The Louvain algorithm can not return a community with a negative contribution to modularity.*

Proof. We can prove it by contradiction. We assume that there is a community with a negative contribution to modularity at the end of the algorithm. Consider the modularity-maximization phase in the last stage of the algorithm. Each vertex in the contracted graph represents a different community.

We suppose that the community C represented by a vertex i has negative modularity, which can be expressed using (1.3) as

$$e_{ii} = \Sigma_{in_C} < \frac{1}{2m}\left(\Sigma_{tot_C}\right)^2 = \frac{(k_i)^2}{2m} \tag{2.2}$$

where e_{ii} denotes the weight of the self-loop representing the edges which were inside of the community C (two times the sum of their weights) and k_i denotes the sum of all edges adjacent to vertices in C, so the sum of $e_{ii} = \Sigma_{in_C}$ and the weights of the edges going out of the community.

The algorithm finished, so it was not possible to merge the vertex i representing the negative-contribution community with another one and obtain a positive gain in modularity. It implies that

$$\Delta Q_{i,i\to D} = \frac{1}{2m}\left[2k_{i,in_D} - \left(\frac{k_i}{m}\right)\Sigma_{tot_D}\right] \leq 0 \tag{2.3}$$

holds for all $D \in NC_i$ where NC_i is the set of communities of neighbors of i.

As a result, for all $D \in NC_i$ holds the following inequality:

$$2k_{i,in_D} \leq \left(\frac{k_i}{m}\right)\Sigma_{tot_D}$$
$$\frac{2k_{i,in_D}}{k_i} \leq \frac{\Sigma_{tot_D}}{m}$$

By summing these terms for all communities of neighbors of vertex i, we get

$$\sum_{D\in NC_i}\frac{2k_{i,in_D}}{k_i} \leq \sum_{D\in NC_i}\frac{\Sigma_{tot_D}}{m}$$
$$\frac{2}{k_i}\sum_{D\in NC_i}k_{i,in_D} \leq \sum_{D\in NC_i}\frac{\Sigma_{tot_D}}{m}$$

Let use use the fact that $\sum_{D\in NC_i} k_{i,in_D} = k_i - e_{ii}$. It holds because k_i is the sum of weights of edges going to neighboring vertices plus the self-loop e_{ii}

$$\frac{2}{k_i}\left(k_i - e_{ii}\right) \leq \sum_{D\in NC_i}\frac{\Sigma_{tot_D}}{m}$$
$$2m\left(1 - \frac{e_{ii}}{k_i}\right) \leq \sum_{D\in NC_i}\Sigma_{tot_D}$$

The expression $\sum_{D \in NC_i} \Sigma_{tot_D}$ sums up weights of all edges adjacent to vertices in communities neighboring to i. This sum is smaller or equal to $2m$, which is the sum of weights of edges adjacent to all vertices, i.e. two times the sum of weights of all edges, minus k_i, because edges adjacent to i are not part of any $D \in NC_i$. That implies

$$\begin{aligned} 2m\left(1 - \frac{e_{ii}}{k_i}\right) \leq \sum_{D \in NC_i} \Sigma_{tot_D} &\leq 2m - k_i \\ -2m\left(\frac{e_{ii}}{k_i}\right) &\leq -k_i \\ e_{ii} &\geq \frac{(k_i)^2}{2m} \end{aligned}$$

Combining this with the inequality (2.2) we obtain a contradiction.

$$\frac{(k_i)^2}{2m} > e_{ii} \geq \frac{(k_i)^2}{2m}$$

□

The proof also confirms that the modularity value obtained using the Louvain algorithm is always non-negative because none of the communities contributes with a negative value. However that is not surprising, because the algorithm maximizes modularity and we know that modularity of a single-community partition is 0.

Number of Communities

In Section 1.3.2, the monster community problem of the label propagation algorithm was mentioned. Because of that, it is fairly common to get just one community as the output of the LPA. This can cause a difference in the resulting partition because the Louvain algorithm returns a single community only when it is not possible to reach any partition with positive modularity.

On the other hand, the Louvain algorithm tends to merge pairs of small communities because of the resolution limit described in Section 1.1.3. The LPA does not optimize modularity, so it does not suffer from this limitation. For example, for a circle of cliques in Figure 1.2a, the Louvain algorithm merges pairs of cliques, while LPA tends to separate them [7].

2.1.2 Common Properties

We showed differences between the algorithms in the previous section. On the other hand, thinking about the algorithms, we see some similarities. Especially label propagation seems similar to the first pass of the modularity maximization in the Louvain algorithm. The label propagation algorithm assigns each node to the community where the majority of its neighbors is. That is in fact local maximization of the first term in modularity definition (1.1). The algorithms also start in the same way, with single-vertex communities, proceeding one vertex after another. They iterate through all vertices again and again until convergence. Let us now show some other the common properties of the algorithms.

Decreasing Number of Communities

Both algorithms start with the same number of communities as the number of vertices. The number of communities decreases when the algorithm proceeds. It happens when a single-node community is incorporated into another community or, in later phases of the Louvain algorithm, if two communities are merged. The algorithms share the principle of moving nodes into neighboring communities. Thus if a community has already disappeared, it can not be restored.

No Single-vertex Communities

None of the algorithms can form a sigle-vertex community consisting of a non-isolated vertex without a loop.

This is obvious for the LPA. The algorithm terminates when each vertex has the same label as the majority of its neighbors. Because of that, a vertex can not have a different label from all its neighbors.

For the Louvain algorithm, this is a specific case of the Proposition 2. As we mentioned in Section 1.1.2, a single-vertex community without a loop, which is not an isolated vertex, always contributes with a negative value to modularity in a connected graph because Σ_{in_C} is 0 and Σ_{tot_C} is positive.

Disconnected Communities

Traag et al. [13] constructed the Leiden algorithm to adress an issue of internally disconnected communities in the Louvain algorithm. Intuitively, all communities should be connected. Does the label propagation algorithm suffer from disconnected communities as well? Figure 2.4 shows a graph and an ordering of vertices leading to a disconnected component.

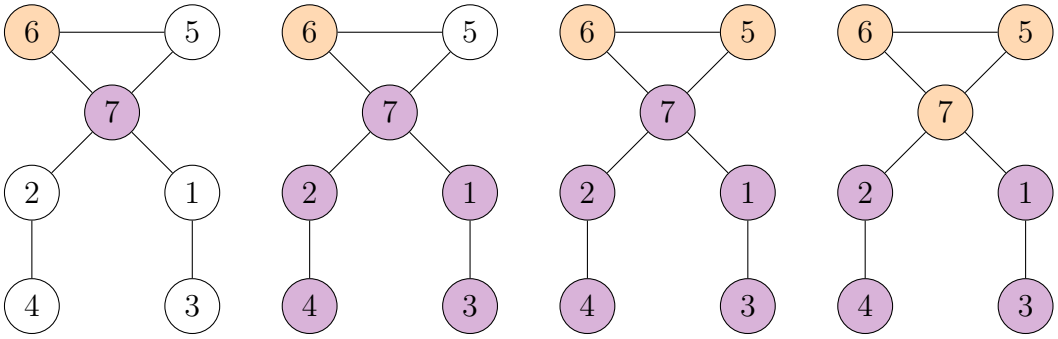


Figure 2.4: Label propagation resulting into a disconnected community. White color denotes that the node has its own initial label. Initial label of node 6 is marked with orange and initial label of node 7 with purple. First, nodes 1 and 2 randomly adopt the purple label of node 7. Then, nodes 3 and 4 adopt the purple label, because they have only one neighbor, so there is no choice. Node 5 randomly choses orange label of node 6. In the end, node 7 adopts the orange label as well. Every node has the same label as the majority of its neighbors, so the LPA finishes. The purple community stays disconnected.

2.2 Comparison on Artificial Networks

Community detection algorithms are often tested on artificial networks with known community stucture to determine which ones are faster or reconstruct the stucture better.

Aldecoa and Marín in 2013 [18] compared various community detection algorithms using a closed benchmark. A closed benchmark starts with a network with a known community structure. Then, edges in the network are rewired. The rewiring process is guided from the initial network towards a final network. The topology and community structure of the final network is identical to the initial one, with the labels of the nodes randomly reassigned. Aldecoa and Marín compared the algorithms based on their performance in reconstructing the initial or (in later stages of rewiring) the final communities. They used two different types of networks, called LFR and RC benchmark. The label propagation algorithm was among the best ones in the LFR benchmark, but performed poorly in the RC benchmark. The Louvain algorithm performed poorly in both of them.

In 2016, Yang et al. [19] conducted a great comparative analysis of the accuracy and computing time of eight different community detection algorithms. They tested the dependence of the algorithms' performance on mixing parameter $\mu \in [0, 0.8]$ across many different network sizes. Same as Aldecoa and Marín [18], they used the LFR benchmark networks, in which the node degrees and community sizes follow the power-law distributions. With 200 to 32000 nodes and a fixed average degree, they changed the structure of networks by using different values of the mixing parameter. They conclude that taking into account both accuracy and computing time, the Louvain algorithm outperforms all the others on the examined networks. It is possible to use it even for $0.5 \lessapprox \mu \lessapprox 0.6$, which is not possible for the label propagation algorithm, because it usually fails and detects only one community. However, the label propagation algorithm is able to successfully uncover the structure when μ is small, and it is the fastest one, thus they recomend it for large networks with $\mu \lessapprox 0.4$.[1]

[1] Igraph implementations of the algorithms were used for the benchmark, so the results might be slightly different for the label propagation algorithm after the correction of goal condition discussed in Section 1.4.

3. Community Structure of the Human Brain

The human brain is a highly complex system consisting of hundreds of billions of interlinked neurons. With the rise of network science came the idea of representing brain as a complex network [1]. Features of complex networks such as high clustering or small-world topology (which reflects an optimal balance between segregation and integration between regions) can be observed in both structural and functional systems of the human brain [20, 21, 22], although the correct interpretation of such graph properties is a matter of ongoing controversy [23, 24].

In 2011, Olaf Sporns wrote a review about the human brain as a complex network [21]. He claims that

> "Significant progress has already been made in discovering the network basis of common disorders of the brain, response to and recovery from brain injury, individual differences, heritability, normal development, and aging." [21, p. 121]

and the field is still developing since that.

This chapter explains what a functional network is and how to obtain it using magnetic resonance imaging. We discuss the current knowledge about community structure in human functional networks.

This thesis focuses on modularity and community structure changes in brain functional networks of multiple sclerosis patients. Therefore the last section of this chapter is devoted to multiple sclerosis and community detection applications on functional networks in multiple sclerosis.

3.1 From the Human Brain to a Network

Currently it is not possible to map the whole human brain on a neuron-level. Therefore comes the question of how to represent the brain as a network. There are two main types of such networks – structural and functional. Let us note that the resulting network for both of the approaches highly depends on the definition of the nodes. Various parcellations of the brain into discrete regions are used to overcome this issue, but we should be aware of their limitations. [20]

Structural networks represent the anatomical structure of the brain. Brain regions are represented as nodes linked by physical connections, such as the white matter tracts that bundle the axonal projections of individual neurons, as edges. The network can be obtained, for example, using diffusion weighted magnetic resonance imaging, which uses the tissue-specific restriction in diffusion direction to map the structure of white matter tracts in the brain. [20, 25]

Functional networks are used to investigate the neurophysiological dynamics of the brain. There are many methodological approaches based, for example, on functional magnetic resonance imaging, electroencephalography, magnetoencephalography, or multielectrode array data. Nodes in the functional network represent brain regions or recording sites. Edges then express synchronization between time series recordings of neural activity of the nodes. [20, 25]

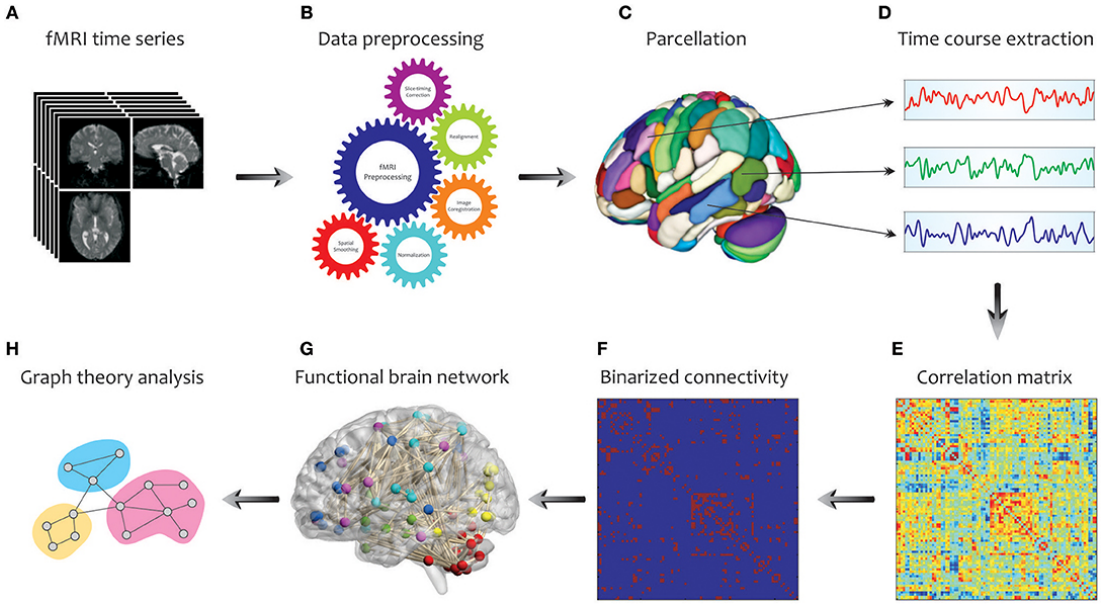


Figure 3.1: Schematic representation of brain network construction using fMRI data. The figure was created by Farahani et al. [26].

3.1.1 Communities in Functional Networks

Recently, much attention has been given to graph theory applications in human neuroscience. In this section, we mention some of the results obtained so far with emphasis on the community structure and modularity.

There is strong evidence that functional networks are modular. [22, 25, 27, 28] There is no agreement on the number of communities, but the majority of functional studies report that there are between three to ten main modules in the human brain. [29] The network structure, and therefore the number of detected communities and the specific value of modularity Q, highly depends on methodologic preferences such as the number of brain regions (brain parcellation) mentioned earlier, but also on neuroimaging pre-processing pipelines, treatment of connectivity weights and others. [29, 30]

There is some evidence that modularity decreases with aging, but the findings are ambiguous [26]. In 2008, Meunier et al. [27] observed changes in community structure, but they claim that there was no significant difference in the modularity of brain networks of young and older people. On the other hand, more recent studies indicate a decrease with aging. In 2014, Song et al. [28] observed lower mean modularity in the older group than in the younger group over a range of different connection densities. In 2020, Iordan et al. [31] concluded the same and also observed that modularity decreased when shifting from resting to task mode in both younger and older adults.

In a systematic review *Application of Graph Theory for Identifying Connectivity Patterns in Human Brain Networks* from 2019, Farahani et al. [26] discuss the effect of cognitive loads on the brain modularity. The researchers believe that functional networks adapt flexibly to the cognitive load while preserving the modular structure. In 2019, Gallen and D'Esposito [32] even formulated an opinion that modular organization in brain networks is beneficial for performance in various cognitive domains.

Community Detection Algorithms

We mentioned several findings about modularity and community structure in functional networks. However, we have not mentioned any community detection algorithm in this context. The community detection algorithms are often heuristic and randomized – both the label propagation and the Louvain algorithm are – and it could affect the resulting partition. In addition, there is no widely accepted definition of a community, so the algorithms could have slightly different goals. Because of that, we should consider the choice of the algorithm.

In 2015, Sporns and Betzel [33] reviewed contemporary approaches to community detection in functional networks. They discuss the pros and cons of various community detection algorithms used in network neuroscience, including the Louvain algorithm. Regarding its disadvantages, they list the resolution limit, but also the fact that there are usually more partitions with high modularity value Q, and it is difficult to choose a single representative partition because of that. However, they do not conclude that any of the algorithms is significantly better, so it is still reasonable to use the Louvain algorithm with the knowledge of its limitation. They did not include the label propagation algorithm in the review.

Most of the studies mentioned so far used some type of modularity maximization. The Louvain algorithm is a popular choice, it was used for example by Keown et al. (2017) [34] in a study about functional networks in autism, by Iordan et al. (2020) [31] and Chen et al. (2021) [29] in studies about aging, or by Servaas et al. (2014) [35] in a study about neuroticism.

Despite the fact that it is one of the fastest community detection algorithms and it is used in practice in other domains like social networks [36, 37, 38], I did not find any study using the label propagation algorithm for community detection or its modifications in functional networks. Therefore it is reasonable to find out if it might be a suitable choice of community detection algorithm in network neuroscience.

3.1.2 Functional Magnetic Resonance Imaging – fMRI

This work handles functional networks, specifically networks obtained from magnetic resonance imaging data. In this section, the fMRI technology is briefly introduced.

MRI

The human body, including the brain, is mainly made out of water molecules, consisting of two hydrogen atoms and one oxygen atom. Protons have a spin magnetic moment. Because of that, the protons in hydrogen nuclei have their own magnetic field. [39]

MRI is based on the usage of a strong static magnetic field generated by powerful magnets. The protons in the water nuclei are usually randomly oriented, but under the influence of the magnetic field, some of them align parallel with it. [39, 40]

A radiofrequency (RF) pulse with the same frequency as the spinning protons is sent through the patient during the imaging process. It stimulates the protons, so they spin out of equilibrium. The protons absorb the energy from the pulse

and change their orientation. Therefore they slightly change their magnetic field. When the RF pulse is over, the protons release the energy that they absorbed and align back. The MRI sensors detect the energy emitted by this process. [39, 40, 41]

The MRI process consists of repeated cycles of RF pulses. Protons in different tissues and fluids change their magnetization at different extent. Different intensity of emitted radio frequency signal at a specific "picture-snapping time" causes different brightness of the resulting image. Because of that, physicians can distinguish various types of tissues. [40, 41]

fMRI, BOLD fMRI and resting-state BOLD fMRI

The goal of functional MRI is to visualize the movement of blood and fluid. Several techniques can produce contrast image of blood flow. One of them is a blood oxygenation level-dependent (BOLD) contrast technique. [41]

Blood brings oxygen to the brain. The BOLD technique uses the fact that deoxyhemoglobin is paramagnetic, but oxyhemoglobin is not. Because of that, deoxyhemoglobin is more sensitive to a magnetic field. MRI uses this fact to determine the sites of higher brain activity because higher brain activity causes higher oxygen consumption. [41]

BOLD fMRI can measure brain activity when a patient performs some task. The resulting images show which parts of the brain were the most active during the task. When the patient is still, does not perform any activity during the imaging procedure, and does not think about anything specific, we call it resting-state MRI. Longer period recording produces time series of spontaneous fluctuations in the BOLD signal. The result reveals spontaneous brain activity, which is accompanied by BOLD fluctuations. [41, 42]

3.1.3 From resting-state fMRI to a Network

Correlations of resting-state fMRI time-series of brain regions indicate correlations between its spontaneous neuronal activation patterns. The brain could be divided into regions using a parcellation scheme. The correlation between resting-state spontaneous activity time-series of the resulting regions can be calculated. This operation forms a *functional connectivity (FC) matrix*, which is showing the level of functional connectivity between the regions. [42]

The FC matrix can be transformed into an adjacency matrix of a graph with brain regions as nodes. To do so, we can use a pre-defined cut-off threshold: two nodes are adjacent if the correlation between them is above the threshold. Another option is to take some percentage of the strongest correlations as edges and discard the others. [42] Unweighted and weighted version of the network can be created, the latter using the above-threshold correlations as edge weights. Both approaches are used in practice, for example Jajcay et al. [43] in 2022 used unweighted networks to compare modularity between hemispheres, and Power et al. [30] in 2011 used weighted networks in their study of functional networks organization.

3.2 Multiple Sclerosis

Multiple sclerosis (MS) is an inflammatory disorder of the central nervous system. In this section, we briefly explain how neurons work and then continue to the principle of the disease.

Billions of neurons in the brain provide a transfer of information. A neuron consists of a cell body (soma) like other cells, but also of dendrites and a single axon. The axon is a specialized structure for transferring information over long distances in the nervous system. On the contrary, dendrites are specialized to receive information from other neurons. Figure 3.2 shows a schematic illustration of a neuron. [44, 45]

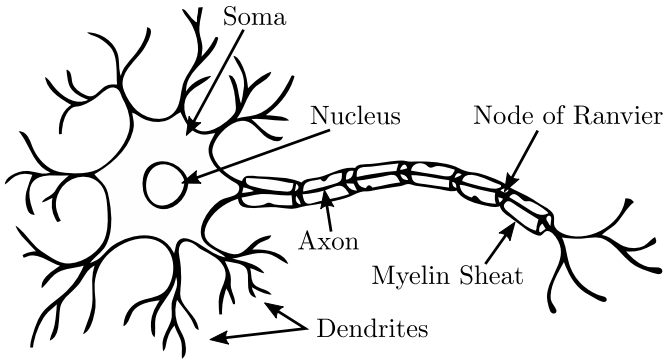


Figure 3.2: Structure of a neuron.

Within the axon, information is transferred through the movement of an electrical charge (impulse) called an action potential. The axons in the central nervous system are covered by a myelin sheath interrupted periodically at nodes of Ranvier, short sections where the axonal membrane is exposed (see Figure 3.2). The myelin sheath insulates the axons, and it allows current to spread faster and farther, speeding up action potential conduction from node to node. Because of that, myelin has an important role in the normal functioning of the central nervous system. [44, 45]

Multiple sclerosis is an autoimmune inflammatory disease causing the damage of myelin. The damage could be transient; remyelination occurs in such a case. The myelin degradation damage worsens the ability of the affected nerves to conduct signals. The symptoms of MS vary according to the affected brain area, including worsening of gait and postural control, problems with vision, depression, cognitive decline, impaired speech, and others. [45, 46, 47]

There are three disease phenotypes in multiple sclerosis: relapsing-remitting, primary progressive, and secondary progressive. Relapsing-remitting is the most common one. It is characterized by clear episodes of worsening neurological symptoms followed by partial or complete recovery periods. Secondary progressive MS is characterized by an initial relapsing-remitting period followed by persistent worsening of symptoms. Primary progressive MS is characterized by an accumulation of disability from the onset of symptoms. [48]

The disability in MS can be quantified and monitored over time using the Expanded Disability Status Scale (EDSS). The EDSS is ranging from 0 (normal neurologic examination) to 10 (death due to MS) in 0.5 increments. Scoring is based on a neurological examination.

3.2.1 Functional Network Structure in MS Patients

In 2019 Fleischer et al. [49] wrote a review of concepts of graph theory applications on brain networks in multiple sclerosis. They claim that fMRI network studies provided important insights into brain reorganization processes due to acute and chronic inflammatory activity. According to their findings, distinct connectivity patterns have been increasingly linked to specific MS phenotypes using graph theory. Therefore, the use of graph theory in the study of multiple sclerosis seems promising.

Regading community structure and modularity in functional networks, Gamboa et al. [50] presented a paper *Working memory performance of early MS patients correlates inversely with modularity increases in resting state functional connectivity networks* in 2014. The study included patients with only low disability status (EDSS < 3). The authors found that network reorganization was reflected by increased modularity. They suggest that it may be the result of focal myelin destruction, leading to an adaptive rewiring of previously interconnected areas. They were further able to differentiate patients from controls with an accuracy of 75% based on the modularity values. [49]

In 2018, Tahdel et al. [51] reported a decrease of within-network coherence, resulting in a formation of larger communities. However, they suggest this can also be explained by a statistical consequence of a decrease of modularity.

In 2020, Welton et al. [52] studied brain network organization in a relationships with cognitive impairment in 37 people with MS and 23 healthy controls. The MS patients in this study had a relapsing-remmiting or secondary progressive MS. The authors observed more graph theoretic network metrics including modularity. The result agrees with the study of Gamboa et al. [50] that the modularity in functional networks in MS is higher than in controls. They conclude that summary measures of network organization including modularity may be valid and reliable markers of cognitive impairment in MS.

4. Community Detection in MS Patients' Functional Networks

We compared the Louvain algorithm and the label propagation algorithm on a dataset of the resting-state fMRI data of patients with the diagnosis of multiple sclerosis. They underwent fMRI scanning before and after neurorehabilitation therapy, and we used the FC matrices obtained from these measuring sessions.

The chapter describes the data used in this work, the process of network construction, and the detection of communities. Afterward, we discuss the results.

4.1 Data

We used dataset created by Bučková et al. in 2021 and stored in an online repository `https://osf.io/p2kj7/`. The authors provided a detailed description of the dataset in the paper *Open Access: The Effect of Neurorehabilitation on Multiple Sclerosis—Unlocking the Resting-State fMRI Data* [46]. This section briefly summarizes how the patients were chosen and how the data were measured and preprocessed. For further information, see the above mentioned paper.

4.1.1 Patients

All patients in the dataset have a positive MS diagnosis of one of the three MS phenotypes – relapsing-remitting, primary progressive, and secondary progressive. Their EDSS is at most 7.5, and they have a neurologically confirmed stable clinical status for a minimum of 3 months prior to the study. The study period was spread over two subsequent projects in 2013-2014 and 2015-2017. Because of the original project purpose, people with notable spastic paraparesis among the symptoms and the ability to reach a rehabilitation center regularly were chosen. Patients with mobility disruption caused by reasons other than those related to the disease or other orthopedic, cardiovascular, or neurological conditions were excluded from the study.

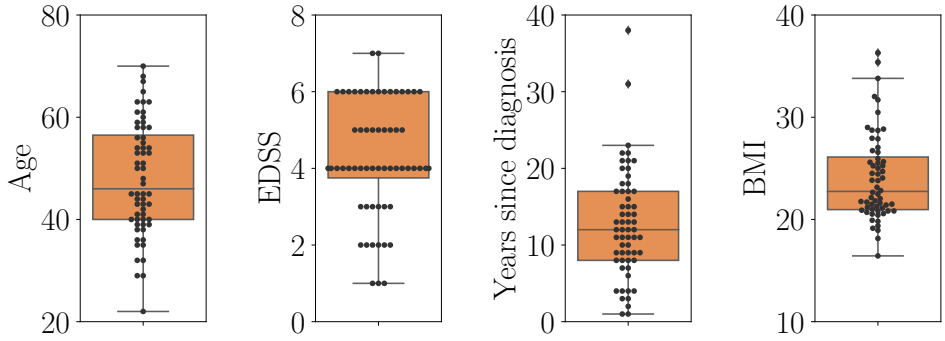


Figure 4.1: Dataset description – quantitative clinical variables

The resulting dataset consists of fMRI imaging data of 60 MS patients before (session 1) and after (session 2) two months of neurorehabilitation therapy. They underwent one of three specific variants of ambulatory "facilitation, inhibition" physical therapy. For more information concerning the detailed interventions and its effect on clinical status and white matter characteristics, see a paper written by Řasová et. al. [53].

There is no healthy or placebo control group in the dataset.

4.1.2 Data Acquisition

BOLD fMRI was performed using a standard MRI scanner Siemens Trio 3T. As mentioned above, the measurements were done in two consecutive projects. There were slight differences in the fMRI protocol in the projects, such as different acquisition times and times to repetition, i.e. time to get one whole-brain 3D image. Both settings are described in the paper describing the data. [46]

Raw fMRI signals were preprocessed using a minimal preprocessing pipeline from CONN toolbox [54] consisting of the following steps: functional realignment and unwarping, slice-timing correction, outlier identification, direct segmentation and normalization, and functional smoothing. A detailed description is beyond the scope of this work and it is included in the paper of Bučková et. al. [46] and also available on a CONN toolbox website `https://web.conn-toolbox.org`.

Preprocessed fMRI signals were used to calculate FC matrices. First, time series were extracted from 116 regions of the Automated Anatomical Labeling (AAL). They were further processed. The final FC matrices were created by correlating the time series.

The authors also did outlier and motion artifact detection. They estimated the two most common head motion metrics called DVARS and FD. DVARS measures changes in signal intensity within the whole brain between two consecutive volumes. FD tracks head movements between the volumes. [46]

4.1.3 Network Construction

Adjacency matrices were created from the FC matrices. The FC matrices were thresholded with a fixed desired edge density in graphs across subjects. We used this appproach to ensure comparability of the modularity value across subjects because edge density affects modularity. There is no universally used threshold. Because of that, FC matrices were thresholded by preserving a selected percentage of the strongest correlations ranging from 15% to 40% in increments of 5%.

Unweighted and weighted networks were created for each FC matrix and desired density. Their adjacency matrices were obtained by converting all below-threshold correlations to 0, and for the unweighted version, all above-threshold correlations to 1. The above-threshold correlations were used as weights for the weighted version. For lower densities, isolated vertices or pairs of vertices sometimes appeared. With the 15% density, it happened significantly more often. More than half of the graphs were affected, some with more than ten isolated vertices. Therefore we discarded this density because a high number of isolated vertices causes an artificially high number of detected communities.

As mentioned in Section 1.1.3, the modularity value itself is difficult to interpret. Therefore we created a randomized version of each unweighted graph to compare with by randomly rewiring edges using Igraph rewiring function `G.rewire(n=G.ecount())`. The resulting graphs have the same number of nodes and edges, degree distribution, and edge weights.

In addition, an average FC matrix was created for each session, and average brain networks were constructed following the same process.

4.2 Experiments

In this section, we denote for brevity the Louvain algorithm as MLM, which stands for multi-layer modularity optimization algorithm. As mentioned before, LPA stands for the label propagation algorithm.

For each of the graphs, communities were estimated by both the MLM and the LPA. Both algorithms are randomized; the tie-breaking and the order of nodes processing are random (for the LPA it changes after each iteration). Because of that, the algorithms were always run a hundred times. We obtained the number of communities, the modularity value, and an assignment of the nodes to communities for each run.

The run with the best modularity value was chosen for each graph for further processing. There are two reasons for this step. Firstly, the Louvain algorithm maximizes modularity, so choosing the result with maximum modularity is reasonable. Secondly, many of the runs of the label propagation algorithm returned just a single community with zero modularity, especially for lower network densities and unweighted networks. Therefore we wanted to choose a result with at least some detected structure. See Figure 4.2 showing the distribution of detected communities count in the weighted and unweighted average network for the first session.

We also tried to run each algorithm one thousand times for the avarage networks. There was no significant difference in the resulting best modularity in the one hundred runs and one thousand runs. Because of that, it is reasonable to run the algorithms one hundred times.

4.2.1 Average Networks: The Global Picture

Our first observation is that for both algorithms, the detected communities make sense concerning the current knowledge about functional networks. For the average graph for the first session, see the detected communities in Figure 4.3.

Both clusterings are right-left symmetric, which is caused by the fact that the functions are distributed in the brain nearly laterally symetrically. Cerebellum (green) and thalamus (cyan) are clearly distinguished in both clusterings. The Louvain algorithm formed one more community consisting of left and right caudate nuclei (black). The visual function is localized in the posterior part of the brain, which corresponds to blue color in the figures. Frontal lobe functionality is more complex. Thus it is reasonable that more communities are mixed in the area (red, purple, and yellow).

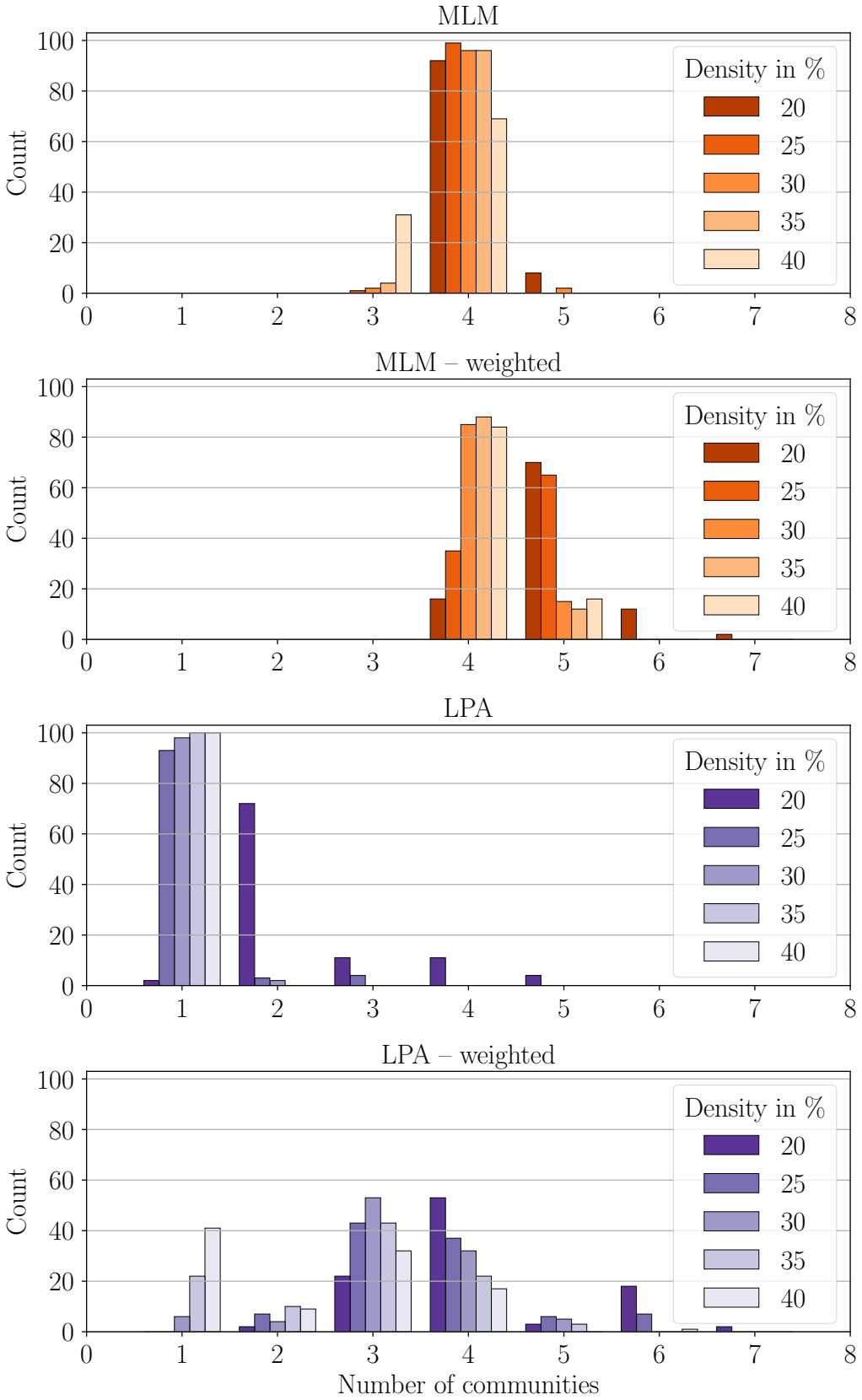
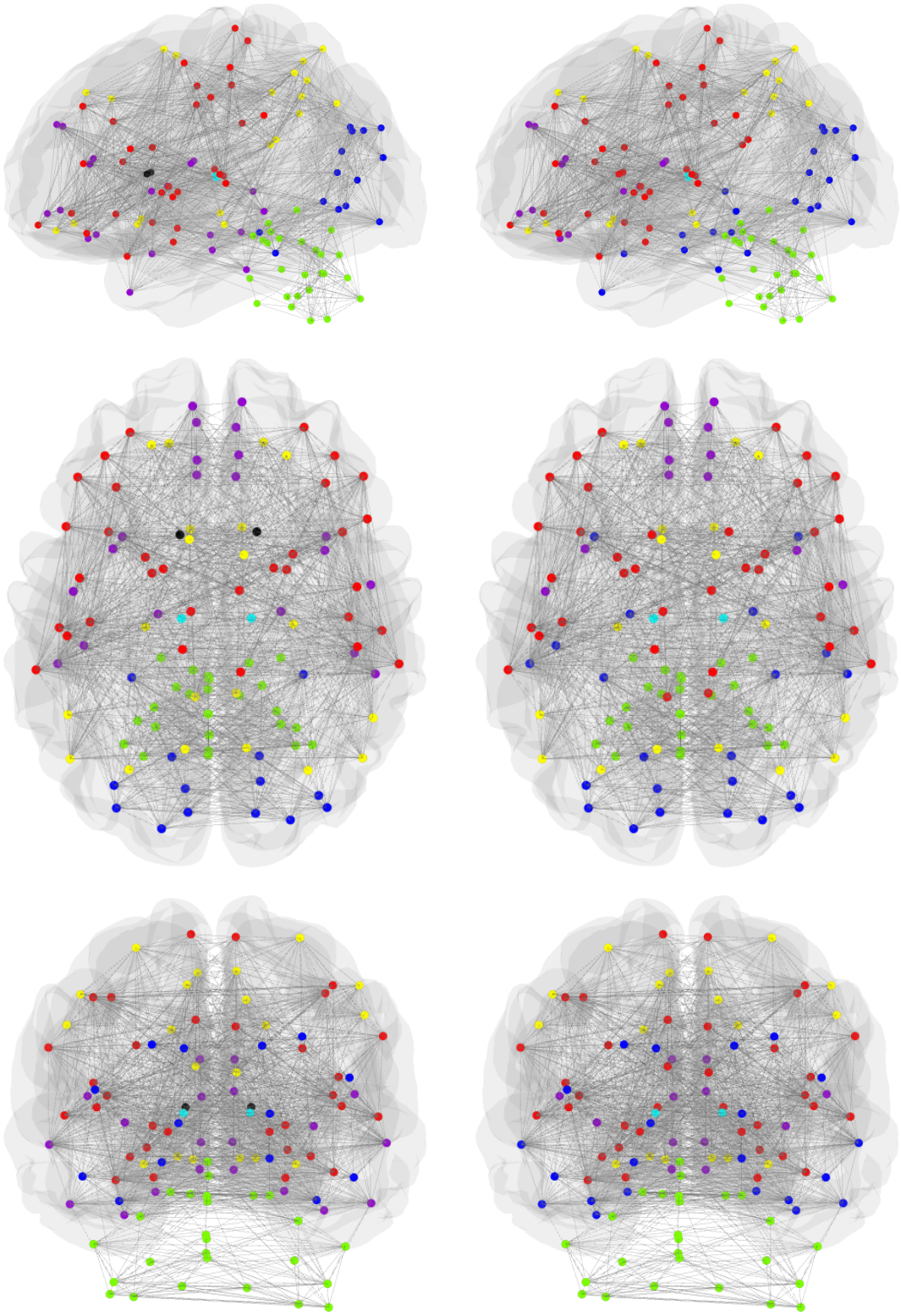


Figure 4.2: Number of communities detected by the LPA and the MLM in the 100 runs in unweighted and weighted average functional network for the first session.



(a) Louvain algorithm (b) Label propagation algorithm

Figure 4.3: Communities detected in the weighted average functional network from the first session with 20% density; left, top and back view for each algorithm. Green part corresponds to cerebellum, cyan to thalamus and blue in the posterior part of the brain to visual system.

4.2.2 Unweighted, Weighted, and Randomized Networks

This section compares the modularity value and the number of detected communities across all densities in unweighted, weighted and randomized networks. The figures in this section contain results for the first session. In all box plots, the box extends from the first quartile, i.e. median of the lower half of the dataset, to the third quartile, i.e. median of the upper half, with a line at the median. The whiskers extend from the box by 1.5× the inter-quartile range, i.e. the distance between the upper and lower quartiles. Outlier points are those past the end of the whiskers.

Modularity

In the Figure 4.4, we see that the modularity value of clusterings detected by both algorithms in unweighted functional networks is higher than that of clusterings detected in randomized networks. This confirms that the human functional network is modular.

The label propagation algorithm often returns one-community clustering for the randomized networks, for which modularity is zero. Since the networks are randomized, there is no underlying community structure, so this behavior is correct. On the other hand, the Louvain algorithm maximizes modularity. Therefore the value keeps in range $(0.1, 0.2)$. That corresponds to the modularity limitations pointed out in Section 1.1.3 because the modularity value is above zero even if there is no underlying community structure.

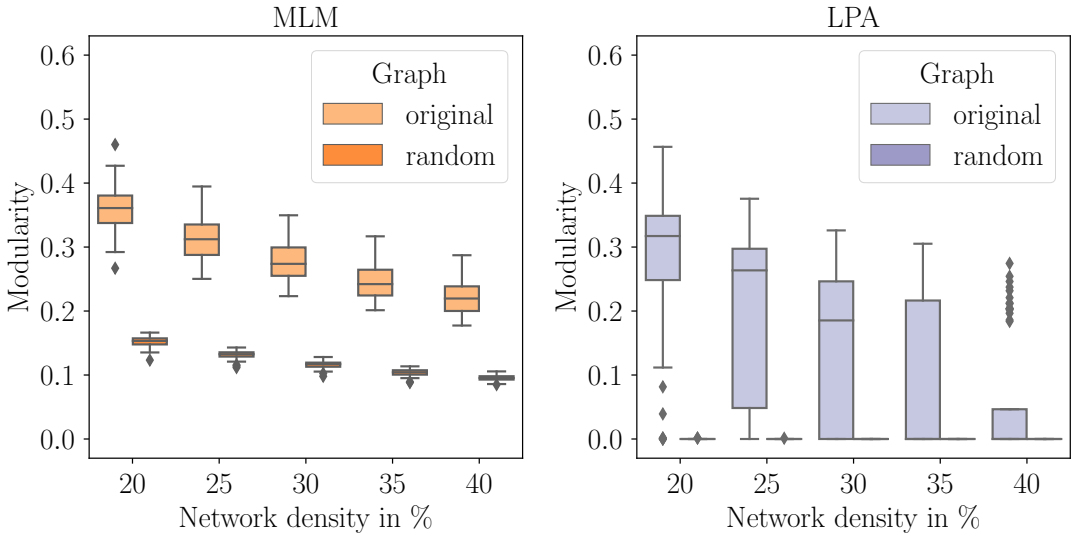


Figure 4.4: Comparison of the modularity with respect to the network density in unweighted networks and randomized networks with the same degree distribution; the distribution is across the 60 patients, for each of them the highest modularity across 100 runs of the algorithms is used. Modularity is clearly higher in the non-randomized networks for both the MLM (left) and the LPA (right).

For both algorithms, the modularity value is higher for the weighted versions of the networks; see Figure 4.5. It is probably because the weights, i.e. the strength of corresponding correlations, keep information about the network and

its community structure. Therefore, restoring the underlying community structure is easier.

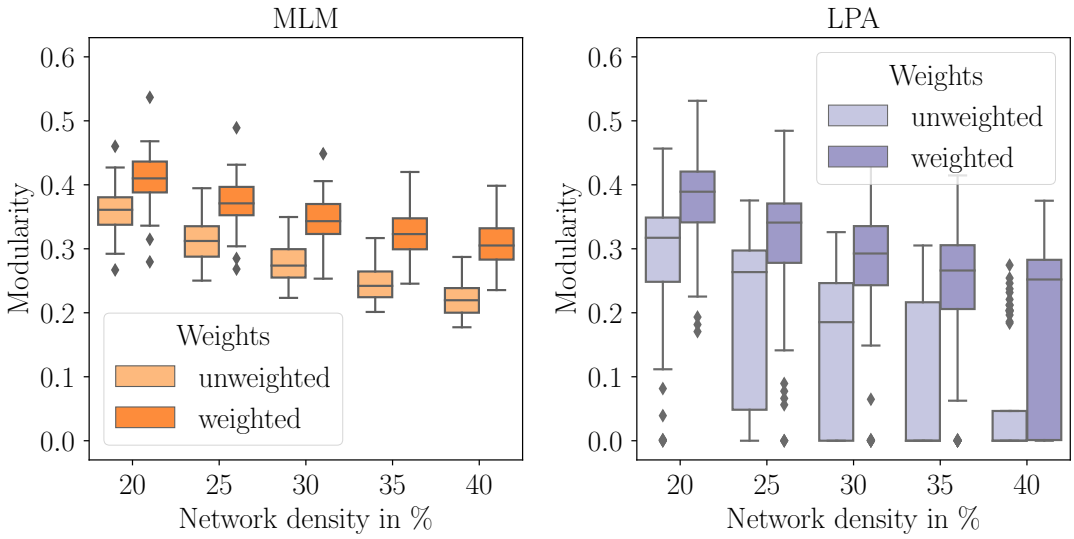


Figure 4.5: Comparison of the modularity with respect to the network density in unweighted and weighted networks. Modularity is higher for the weighted networks for both the MLM (left) and the LPA (right).

Comparing the algorithms, we see in the Figure 4.6 that the Louvain algorithm returns higher modularity with far lower variance than the label propagation algorithm. That is not surprising since the Louvain algorithm is based on modularity maximization. However, the difference in mean modularity for weighted graphs and lower graph densities is not too large.

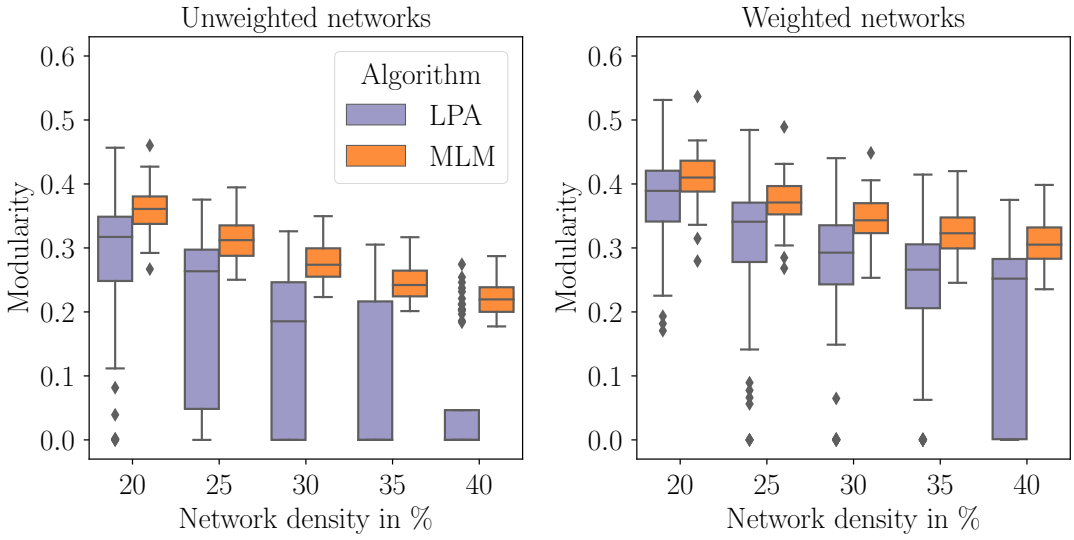


Figure 4.6: Comparison of the modularity with respect to the network density for the MLM and the LPA in unweighted and weighted networks. Modularity is higher and less variable for the MLM in both unweighted and weighted networks.

Number of communities

Besides modularity, let us look at the number of communities detected by the algorithms. Figures 4.7 and 4.8 show the number of networks out of the total 60 functional networks of the patients in which a certain number of communities was detected. The number of detected communities is higher with lower network density for both algorithms.

The algorithms tend to detect more communities in the weighted functional networks than in the unweighted ones across all network densities. However, they differ regarding the number of communities detected in the randomized networks. The Louvain algorithm tends to detect more communities in the randomized networks than in the weighted ones. On the other hand, the label propagation algorithm usually returns only one community for the randomized networks, which agrees with the modularity observations from the previous section.

If we compare the algorithms, the Louvain algorithm usually detects more communities than the label propagation algorithm. For the non-randomized unweighted networks and higher network densities, the label propagation algorithm often fails to detect any community structure and returns only one cluster. That does not match the fact that the human brain is modular, and that there probably is some underlying community structure in the functional network. On the contrary, the Louvain algorithm always returned at least two communities for unweighted networks and at least three for the weighted ones.

Network Density

The threshold and corresponding graph density highly affect the resulting number of communities and modularity of the detected clustering. Figure 4.6 shows that the modularity value increases with the decreasing graph density for both algorithms in both weighted and unweighted networks. Figures 4.7 and 4.8 show that the number of communities increases with decreasing density. Let us emphasize that for higher network density, the label propagation algorithm often detects only one or two communities, which does not help to unveil the functional network structure. It might be caused by an increase in the mixing parameter, see 2.2.

It seems that it is easier to detect community structure in graphs with lower density. On the other hand, we should be aware of the rising number of disconnected vertices in functional networks with lower density because they can not be assigned to any community. In practice, we can use more different network densities, like Welton et al. [52] or Jajcay et al. [43].

4.2.3 Multiple Sclerosis Data

The previous section compares the algorithms' results for both weighted and unweighted networks. In this section, we focus on the data itself. We use only the weighted networks in the whole section because the experiments in previous section suggest that this approach leads to more relevant results. The modularity of weighted network clusterings is higher, and the label propagation algorithm has a lower tendency to detect only one partition.

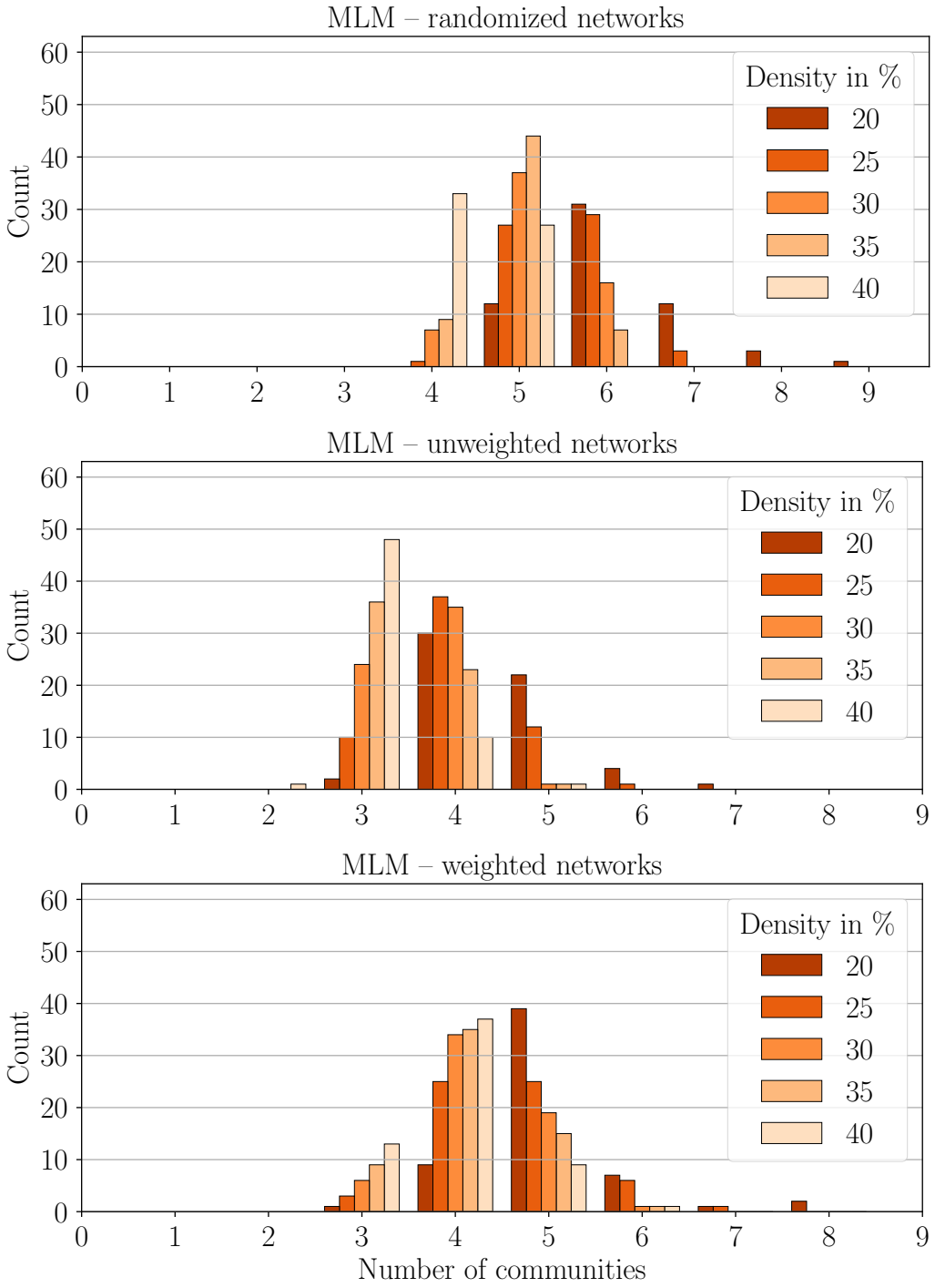


Figure 4.7: Number of communities detected using the Louvain algorithm in unweighted, weighted and randomized networks by network density. An outlier p028 is excluded for 20% density because the graph has 9 components, so the number of detected communities is artificially high.

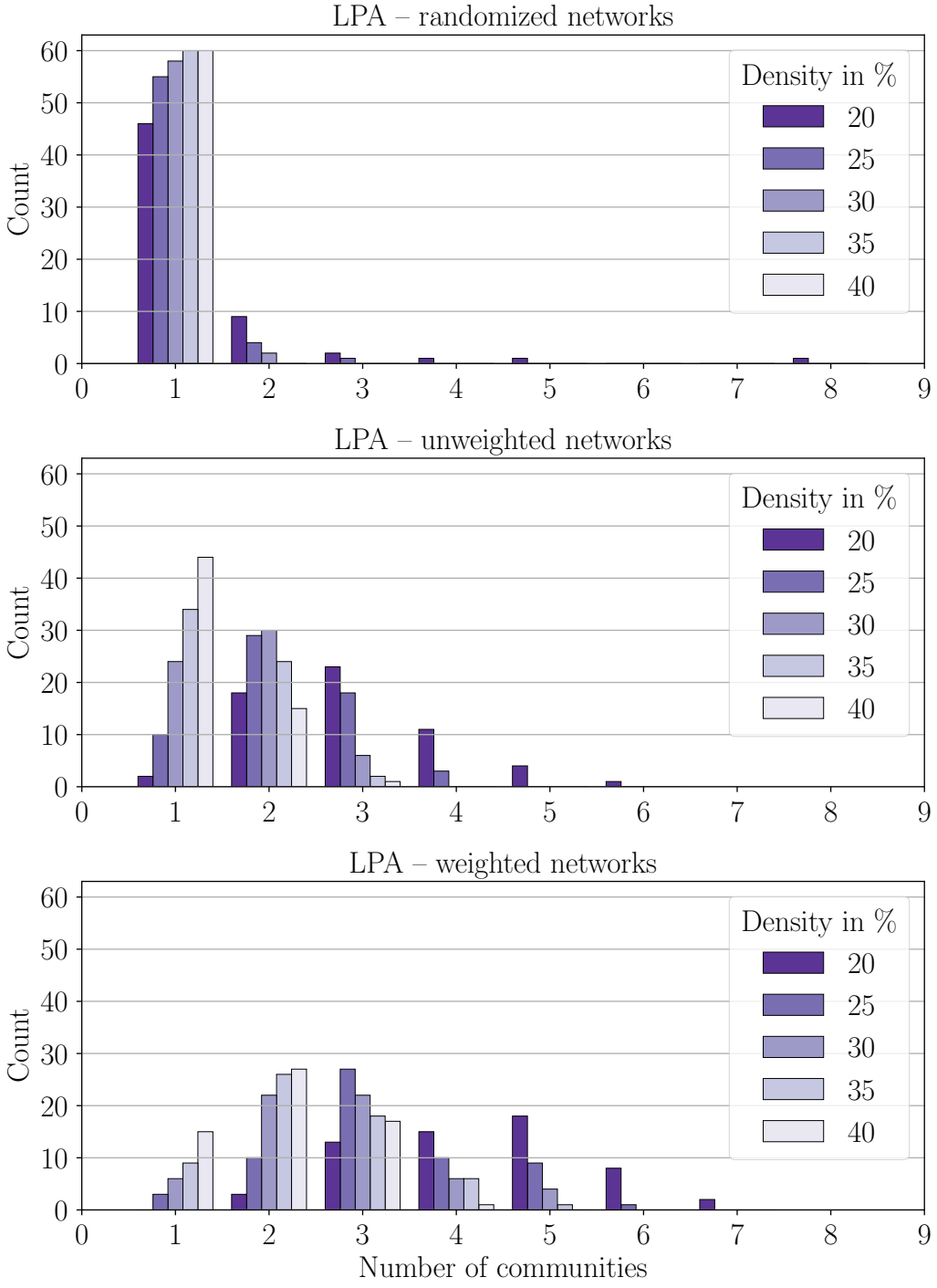


Figure 4.8: Number of communities detected using the label propagating algorithm in unweighted, weighted and randomized networks by network density. An outlier p028 is excluded for 20% density because the graph has 9 components, so the number of detected communities is artificially high.

Sessions Comparison

The most interesting question about the data is whether there is a difference between the sessions – whether the neurorehabilitation therapy changed the modularity and community structure of the patients' functional networks. Figure 4.9 compares modularity of the two sessions and Figure 4.10 compares the number of detected communities.

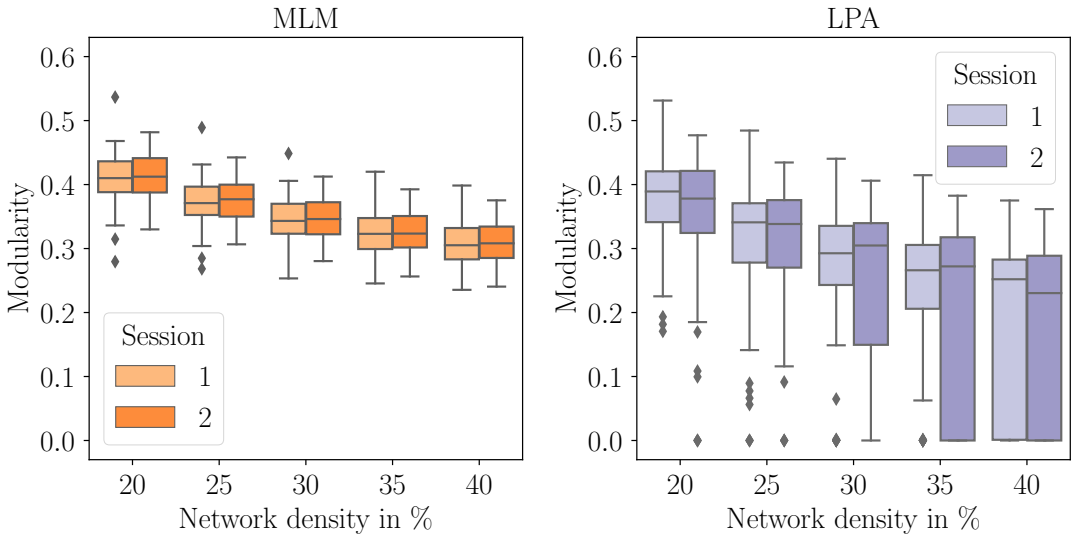


Figure 4.9: Comparison of the functional networks' modularity before (session 1) and after (session 2) the neurorehabilitation therapy. There is probably no significant difference between the sessions.

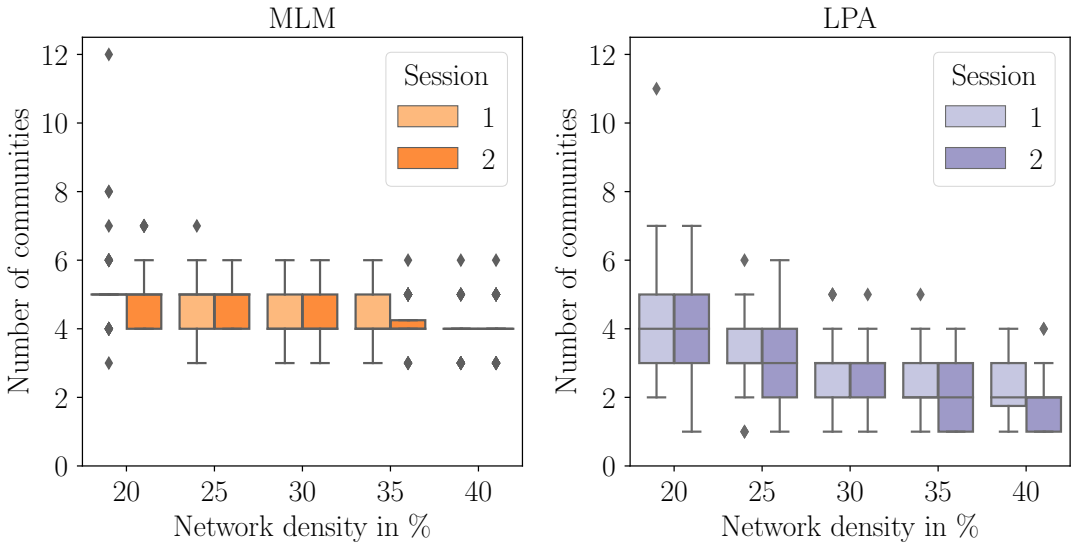


Figure 4.10: Comparison of the functional networks' number of communities before (session 1) and after (session 2) the neurorehabilitation therapy. There is probably no significant difference between the sessions.

We performed paired samples t-tests for each network density for modularity and number of communities. The null hypothesis is that the average modular-

ity/number of communities is identical in both sessions. The alternative hypothesis is that the means of modularity/number of communities observed in functional networks before and after the therapy are unequal. For both algorithms, all p-values are greater than 0.1, so there is no significant difference between the sessions.

MS Types Comparison

There are patients with all three MS types in the dataset – 6 primary progressive, 20 secondary progressive, and 34 relapsing-remitting. In this section, we focus on the MS types separately. Figures 4.11 and 4.12 show the modularity of detected clustering across all network densities.

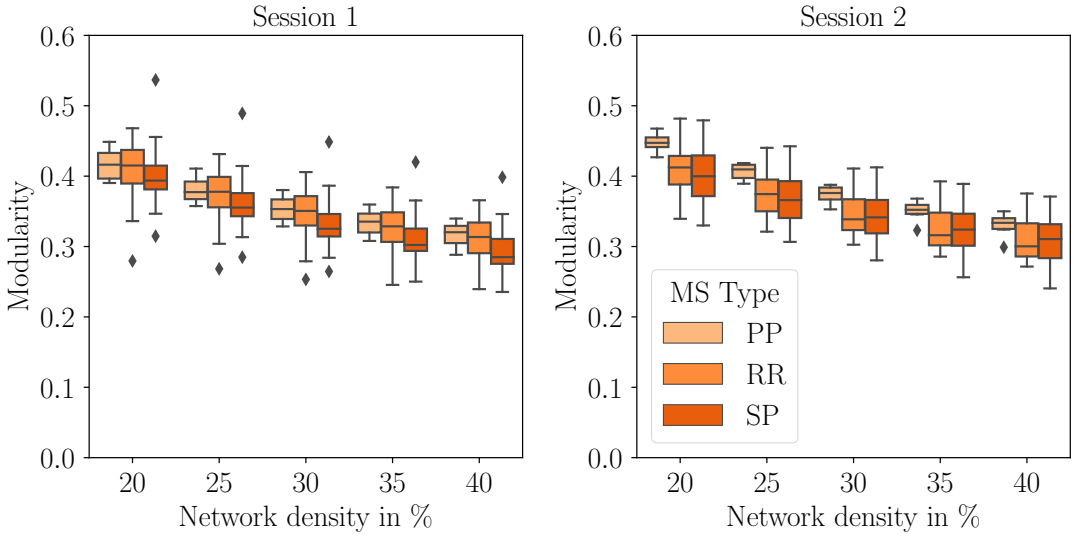


Figure 4.11: Comparison of the functional networks' modularity detected by the MLM in the two sessions for the three MS types – primary progressive (PP), relapsing-remitting (RR), and secondary progressive (SP).

We see that there is probably no systematic difference in the modularity of the MS types in the first session. However, in the second session, the modularity of primary progressive MS seems to be higher than the others and higher than in the first session. Figure 4.13 shows the session comparison for primary progressive MS in detail.

We performed a paired samples t-test for each MS type and network density. The null hypothesis is that the average modularity is identical in both sessions, and the alternative hypothesis is that the average modularity is lower in the first session than in the second session.

For the Louvain algorithm, $p < 0.05$ for primary progressive MS for all network densities except 40% ($p \approx 0.06$), and $p > 0.1$ for all other MS types and network densities. For the label propagation algorithm, p is higher than 0.1 for all MS types and network densities. As the sample is not particularly large and the distribution of modularity may nonnegligibly deviate from Gaussianity, we have repeated the analysis with the nonparametric Wilcoxon signed rank test, and obtained equivalent results, $p < 0.05$ for primary progressive MS for all network densities except 40%.

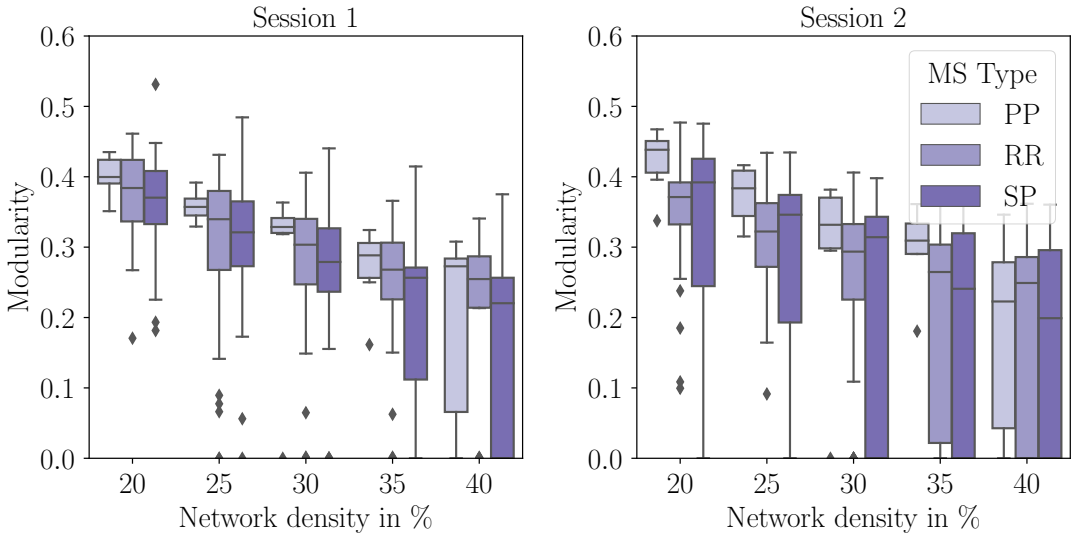


Figure 4.12: Comparison of the functional networks' modularity detected by the LPA in the two sessions for the three MS types – primary progressive (PP), relapsing-remitting (RR), and secondary progressive (SP).

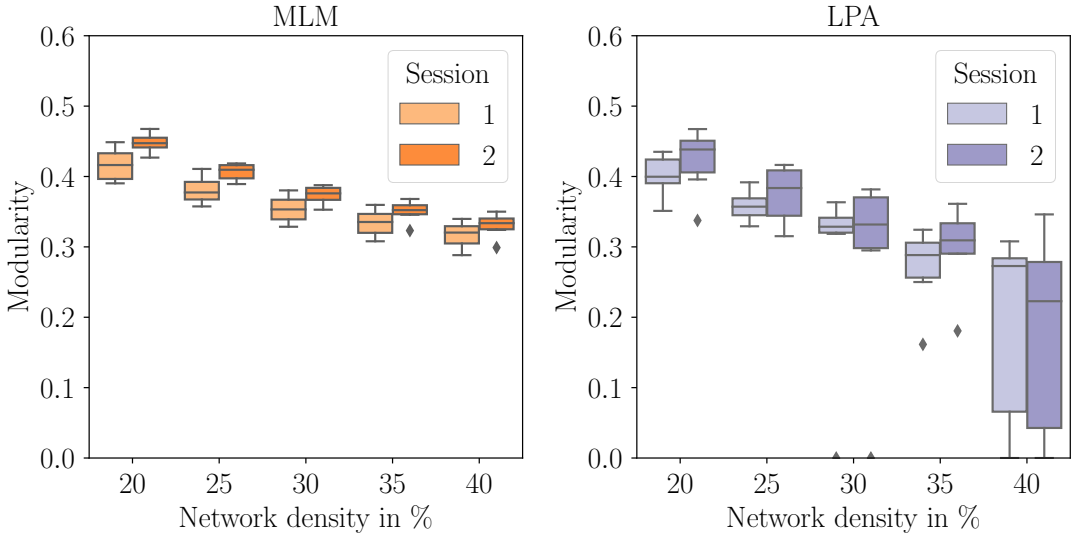


Figure 4.13: Comparison of the modularity of primary progressive MS patients' functional networks in first and the second session. Modularity is higher in the second session for both the MLM (left) and the LPA (right).

The modularity of the clustering detected by the Louvain algorithm in the functional networks of patients with primary progressive MS increased between sessions. It might be due to the therapy or due to the progression of the disease. We do not have a placebo control group, so we can not decide which one is more likely. Furthermore, there are only six patients with the primary progressive MS type. Therefore we should be careful to make conclusions from these observations.

We also tested the number of communities for each MS type with the paired samples t-test to see if there is a difference in number of communities between the sessions. The observed p-values are smaller than 0.05 for the secondary progressive MS and 20% density and for the primary progressive MS and 20%

density for the clusterings detected using the Louvain algorithm. However, it is not consistent across network densities, MS types, or algorithms, so it is probably caused by the fact that we performed the t-test many times (multiple testing problem).

Datasets Comparison

As written in Section 4.1.2, the 60 patients were measured in two projects with slightly different fMRI protocols. The resulting data could be divided into two datasets based on the project (28 patients in dataset 1 and 32 patients in dataset 2). Do the datasets differ regarding the resulting modularity? Figures 4.14 and 4.15 show that the modularity might be slightly higher in the second dataset.

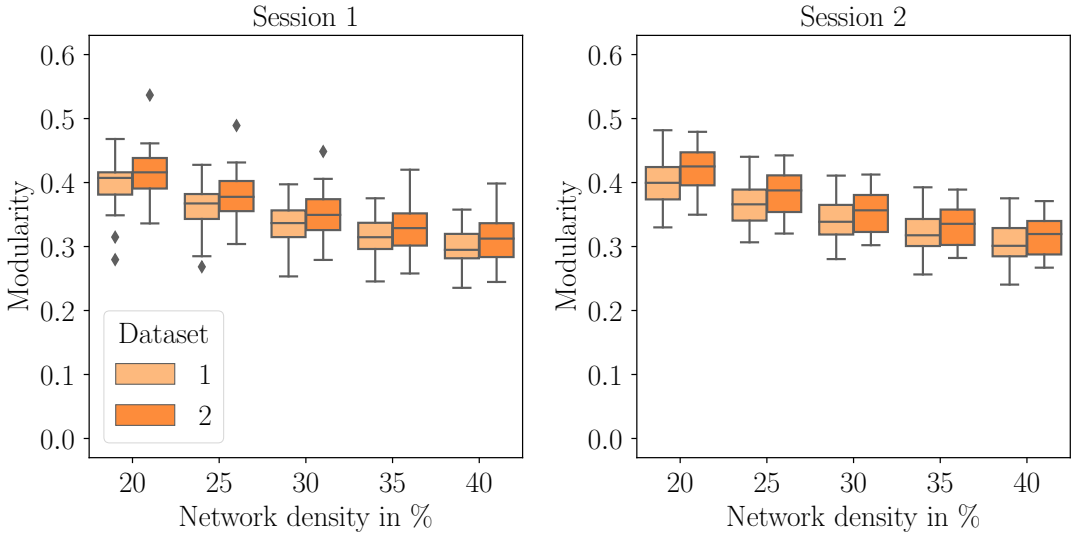


Figure 4.14: Modularity detected by the MLM in first (left) and second (right) session for the two datasets with different fMRI protocols.

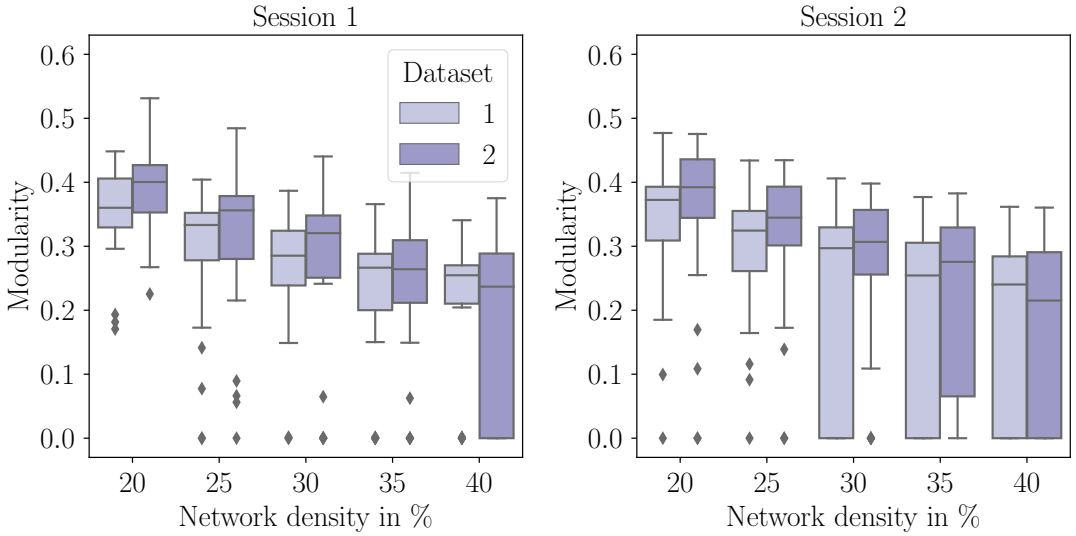


Figure 4.15: Modularity detected by the LPA in first (left) and second (right) session for the two datasets with different fMRI protocols.

We used independent samples Welch's t-test to examine if there is a significant difference. The null hypothesis is that the two datasets have identical average modularity and the alternative hypothesis is that the first dataset has lower average modularity than the second dataset.

The resulting p-values are smaller than 0.1 for all network densities except 40% and both sessions for the Louvain algorithm (≈ 0.1 for 40% density in the second session). It suggests that there might be a difference in modularity between the two datasets. However, we must consider that 5 of the 6 primary progressive MS patients are in the second dataset, which can influence the result. Therefore we performed the tests again with the primary progressive MS patients filtered out. The resulting p-values were a bit higher, but for densities 20% to 30% still smaller than 0.1 for the Louvain algorithm (≈ 0.1 for the network density 35% for both sessions). It could still indicate a small difference between datasets. In any case, if there is a difference, the label propagation algorithm cannot detect it.

We also compared the number of detected communities. There is no significant difference between datasets.

Gender Comparison

There are 37 female and 23 male patients in the data. Same as for the datasets comparison, we used independent samples Welch's t-test to examine if there is a significant difference between genders. The null hypothesis is that male and female patients' functional networks have identical average modularity. The alternative hypothesis is that female patients have lower average modularity than male patients.

The p-value is greater than 0.1 for all settings for the first session. However, it is lower than 0.05 for both algorithms and all network densities in the second session, except for 40% density for the LPA. Figure 4.16 shows the small difference. We must interpret these results carefully because 5 out of 6 primary progressive MS patients are male, but $p \lessapprox 0.05$ stays even when the primary progressive patients are filtered out.

There might be a difference in modularity changes between genders. However, the genders are also not equally distributed in the two datasets. There are 21 female and 7 male patients in the first one and 16 male and 16 female in the second one. Therefore the dependencies in the results are not clear. We can not tell for sure if they are caused by the different imaging protocols, patients' gender, or distribution of MS types in patients.

Other Investigations

We also tried to explore other dependencies in the data, albeit being aware that any statistically significant results might be due to extended multiple testing, and thus would only constitute exploratory findings warranting further independent validation. We compared the sessions for different types of therapy using the paired samples t-test, but there were no statistically significant differences.

We also calculated the Pearson correlation coefficient to determine if there is a correlation between modularity/number of communities and age, EDSS score, BMI, or the number of years since diagnosis. We tested each combination of

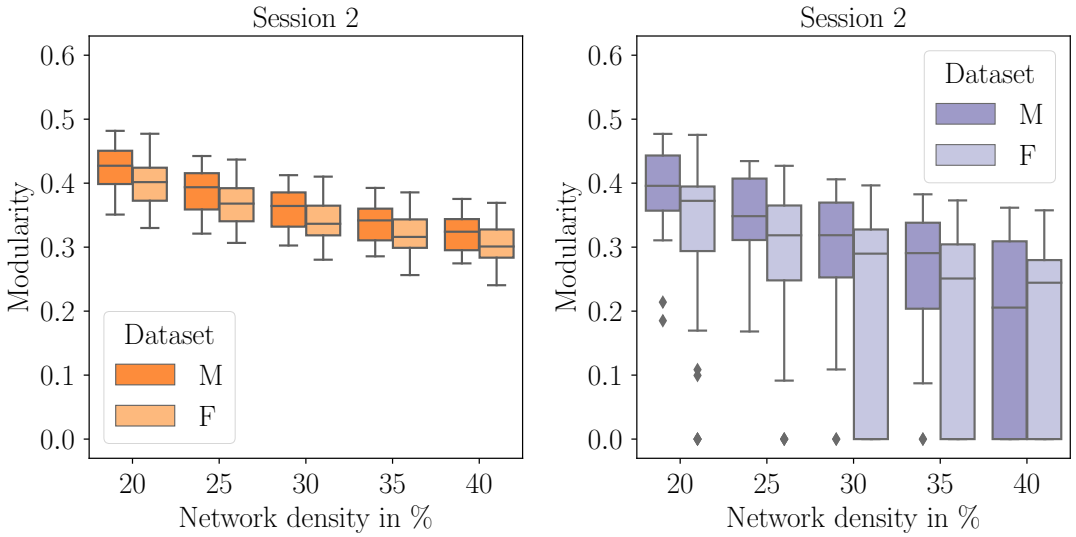


Figure 4.16: Modularity detected by the MLM (left) and the LPA (right) for male (M) nad female (F) patients in the second session.

session, algorithm, and network density. The Pearson's correlation coefficient is always in a range $[-0.2, 0.2]$, which indicates no or negligible relationship.

4.3 Discussion

Let us summarize the results of the experiments. We applied the Louvain algorithm and the label propagation algorithm to the unweighted, weighted and randomized functional networks with densities from 20% to 40%. We always run the algorithms hundred times and choose the result with the higest modularity because of the algorithms' randomization and because the label propagation algorithm often detected only one community in the unweighted networks.

The modularity of detected community structure is significantly higher in the functional networks than in the randomized networks for both algorithms. In addition, as illustrated in the Figure 4.3, the detected community structure corresponds to the modular structure of the human brain for both of them. This observation confirms the relevance of community detection in functional networks and both algorithms appear to be relevant for appliction on functional networks.

Both algorithms detect clusterings with more communities and higher modularity in the weighted networks than in the unweighted ones. This is caused by the fact that the weights encode information about the community structure, thus it is easier to distinguish the functional modules in weighted networks.

Comparing the algorithms, both modularity and the number of detected communities were higher accros all the network densities using the Louvain algorithm. The reason for the higher modularity probably is, that the Louvain algorithm optimizes modularity directly and the label propagation algorithm does not. The higer number of detected communities may be caused by the monster community problem in the label propagation algorithm discussed in Section 1.3.2.

For higher densities, the label propagation algorithm often fails to detect any community structure. This is probably caused by the inability of the label

propagation algorithm to detect community structure in networks with higher mixing parameter, which was observed by Yang et al. [19] and discussed in Section 2.2.

Regarding the data itself, the possible differences in modularity discussed below were always detected by the Louvain algorithm, the label propagation algorithm did not find any of them.

There is no significant difference between the first and the second session neither in modularity, nor in the number of detected communities. Therefore we did not inspect the community structure further, because its change would with high probability affect the modularity or number of communities.

However, it seems that there is an increase in modularity between the sessions in the small sample of six patients with primary progressive MS. We do not have a placebo control group, so we can not decide if it is due to the disease course or the therapy. There is also an evidence that there might be a difference in modularity between the two projects, the networks obtained from the second projects' data have slightly higher modularity. The last observation is that in the second session male patients have slightly higher modularity than female patients. However, the MS types and genders were not distributed equally in the projects and 5 out of 6 primary progressive patients were men. If we filtered out the primary progressive patients, the evidence for these differences were even weaker. Therefore it is neccessary to do more experiments with more MS patients before deriving a result regarding the influence of MS type, fMRI protocol or gender on modularity.

Conclusion

Community detection is an essential part of complex network analysis. This work focused on two main community detection algorithms, the Louvain algorithm, and the label propagation algorithm. The goal was to compare the algorithms theoretically and use them to detect changes in community structure in multiple sclerosis patients' functional networks. The first part was devoted to explaining the algorithms, their limits, and modifications. We also introduced modularity, a network measure essential for the Louvain algorithm.

In the theoretical part, we proved that the algorithms make opposite decisions in specific situations. We also show that the label propagation algorithm can result in a paritition containing a community with a negative contribution to modularity. This is not possible for the Louvain algorithm, and thus the modularity of a partition detected by the Louvain algorithm can not be negative. On the other hand, we show that only isolated vertices can form single-vertex communities in both algorithms, and they can both return a partition containing a disconnected community. The last section of the second chapter summarizes previous studies about the algorithms' performance in benchmarks. It shows that they are among the fastest community detection algorithms up to date.

The practical part of this thesis focused on the application of the algorithms in neuroscience, namely community detection in the human brain. Specifically, we detected communities in multiple sclerosis patients' functional networks. While the Louvain algorithm is widely used in network neuroscience, the label propagation algorithm is neglected in this field. Therefore it was interesting to compare their results.

We studied the resulting modularity and number of detected communities in MS patients' functional networks. We confirmed that the functional networks are modular, and the algorithms are able to unveil the community structure. The Louvain algorithm detected overall higher modularity and more communities and its results were less variable. This is probably caused by the monster community occurence in the label propagation algorithm and the inability of the label propagation algorithm to handle networks with higer mixing parameter. Therefore the Louvain algorithm is probably better for community detection in human brain functional networks and it is reasonable that the label propagation algorithm is not used in this field. Modularity and number of communities increase for both algorithms with decreasing network density.

We did not find any significant change in modularity between the two sessions considering the whole dataset. However, the modularity increased in the functional networks of the six patients with a primary progressive course of multiple sclerosis. We found that there might be other minor changes with respect to fMRI protocol or patients gender, but the evidence is weak. In any case, further experiments would be necessary to confirm these observations.

Bibliography

[1] A.-L. Barabási and M. Pósfai. *Network science*. Cambridge University Press, Cambridge, 2016. ISBN 9781107076266 1107076269. URL http://barabasi.com/networksciencebook/.

[2] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(9):2658–2663, 2004. ISSN 0027-8424. URL https://doi.org/10.1073/pnas.0400054101.

[3] S. E. Garza and S. E. Schaeffer. Community detection with the label propagation algorithm: A survey. *Physica A: Statistical Mechanics and its Applications*, 534:122058, 2019. ISSN 0378-4371. URL https://www.sciencedirect.com/science/article/pii/S0378437119312026.

[4] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69:026113, 2004. URL https://link.aps.org/doi/10.1103/PhysRevE.69.026113.

[5] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172–188, 2008. URL https://ieeexplore.ieee.org/document/4358966.

[6] R. Guimerà, M. Sales-Pardo, and L. A. N. Amaral. Modularity from fluctuations in random graphs and complex networks. *Phys. Rev. E*, 70:025101, 2004. URL https://link.aps.org/doi/10.1103/PhysRevE.70.025101.

[7] S. Fortunato and M. Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007. URL https://www.pnas.org/doi/abs/10.1073/pnas.0605965104.

[8] J. W. Berry, B. Hendrickson, R. A. LaViolette, and C. A. Phillips. Tolerating the community detection resolution limit with edge weighting. *Phys. Rev. E*, 83:056119, 2011. URL https://link.aps.org/doi/10.1103/PhysRevE.83.056119.

[9] Z. Yang, R. Algesheimer, and C. J. Tessone. A comparative analysis of community detection algorithms on artificial networks. *Scientific Reports*, 6(1): 30750, 2016. ISSN 2045-2322. URL https://doi.org/10.1038/srep30750.

[10] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):10008, 2008. URL https://doi.org/10.1088/1742-5468/2008/10/p10008.

[11] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner. On finding graph clusterings with maximum modularity. In *Graph-Theoretic Concepts in Computer Science*, pages 121–132, Berlin, Heidelberg, 2007XXX. Springer Berlin Heidelberg. ISBN

978-3-540-74839-7. URL https://link.springer.com/chapter/10.1007/978-3-540-74839-7_12.

[12] P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti. Generalized louvain method for community detection in large networks. *2011 11th International Conference on Intelligent Systems Design and Applications*, pages 88–93, 2011. URL https://ieeexplore.ieee.org/document/6121636.

[13] V. A. Traag, L. Waltman, and N. J. van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1):5233, 2019. ISSN 2045-2322. URL https://doi.org/10.1038/s41598-019-41695-z.

[14] J. Zhang, J. Fei, X. Song, and J. Feng. An improved louvain algorithm for community detection. *Mathematical Problems in Engineering*, 2021:1485592, 2021. URL https://doi.org/10.1155/2021/1485592.

[15] U. N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76 (3), 2007. URL https://doi.org/10.1103%2Fphysreve.76.036106.

[16] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3): 75–174, 2010. ISSN 0370-1573. URL https://www.sciencedirect.com/science/article/pii/S0370157309002841.

[17] A. Zhang, G. Ren, Y. Lin, B. Jia, H. Cao, J. Zhang, and S. Zhang. Detecting community structures in networks by label propagation with prediction of percolation transition. *The Scientific World Journal*, 2014:148686, 2014. ISSN 2356-6140. URL https://doi.org/10.1155/2014/148686.

[18] R. Aldecoa and I. Marín. Exploring the limits of community detection strategies in complex networks. *Scientific Reports*, 3(1):2216, 2013. ISSN 2045-2322. URL https://doi.org/10.1038/srep02216.

[19] Z. Yang, R. Algesheimer, and C. J. Tessone. A comparative analysis of community detection algorithms on artificial networks. *Scientific Reports*, 6(1): 30750, 2016. ISSN 2045-2322. URL https://doi.org/10.1038/srep30750.

[20] E. Bullmore and O. Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 10(3):186–198, 2009. ISSN 1471-0048. URL https://doi.org/10.1038/nrn2575.

[21] O. Sporns. The human connectome: a complex network. *Annals of the New York Academy of Sciences*, 1224, 2011. URL https://doi.org/10.1111/j.1749-6632.2010.05888.x.

[22] E. Gozdas, S. K. Holland, M. Altaye, and CMIND Authorship Consortium. Developmental changes in functional brain networks from birth through adolescence. *Human brain mapping*, 40(5):1434–1444, 2019. ISSN 1097-0193. URL https://doi.org/10.1002/hbm.24457.

[23] D. Papo, M. Zanin, J. H. Martínez, and J. M. Buldú. Beware of the small-world neuroscientist! *Frontiers in Human Neuroscience*, 10, 2016. ISSN 1662-5161. URL https://www.frontiersin.org/article/10.3389/fnhum.2016.00096.

[24] J. Hlinka, D. Hartman, N. Jajcay, D. Tomeček, J. Tintěra, and M. Paluš. Small-world bias of correlation networks: From brain to climate. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(3):035812, 2017. URL https://doi.org/10.1063/1.4977951.

[25] Da. Mears and H. B. Pollard. Network science and the human brain: Using graph theory to understand the brain and one of its hubs, the amygdala, in health and disease. *Journal of Neuroscience Research*, 94, 2016. URL https://doi.org/10.1002/jnr.23705.

[26] F. V. Farahani, W. Karwowski, and N. R. Lighthall. Application of graph theory for identifying connectivity patterns in human brain networks: A systematic review. *Frontiers in Neuroscience*, 13, 2019. ISSN 1662-453X. URL https://www.frontiersin.org/article/10.3389/fnins.2019.00585.

[27] D. Meunier, S. Achard, A. Morcom, and E. Bullmore. Age-related changes in modular organization of human brain functional networks. *NeuroImage*, 44(3):715–723, 2009. ISSN 1053-8119. URL https://doi.org/10.1016/j.neuroimage.2008.09.062.

[28] J. Song, R. M. Birn, M. Boly, T. B. Meier, V. A. Nair, M. E. Meyerand, and V. Prabhakaran. Age-related reorganizational changes in modularity and functional connectivity of human brain networks. *Brain connectivity*, 4(9): 662–676, 2014. ISSN 2158-0022. URL https://doi.org/10.1089/brain.2014.0286.

[29] X. Chen, J. Necus, L. R. Peraza, R. Mehraram, Y. Wang, J. T. O'Brien, A. Blamire, M. Kaiser, and J.-P. Taylor. The functional brain favours segregated modular connectivity at old age unless affected by neurodegeneration. *Communications Biology*, 4(1):973, 2021. ISSN 2399-3642. URL https://doi.org/10.1038/s42003-021-02497-0.

[30] J. D. Power, A. L. Cohen, S. M. Nelson, G. S. Wig, K. A. Barnes, J. A. Church, Alecia C. Vogel, T. O. Laumann, F. M. Miezin, B. L. Schlaggar, and S. E. Petersen. Functional network organization of the human brain. *Neuron*, 72(4):665–678, 2011. ISSN 1097-4199. URL https://doi.org/10.1016/j.neuron.2011.09.006.

[31] A. D. Iordan, K. D. Moored, B. Katz, K. A. Cooke, M. Buschkuehl, S. M. Jaeggi, T. A. Polk, S. J. Peltier, J. Jonides, and P. A. Reuter-Lorenz. Age differences in functional network reconfiguration with working memory training. *Human brain mapping*, 42(6):1888–1909, 2021. ISSN 1097-0193. URL https://doi.org/10.1002/hbm.25337.

[32] C. L. Gallen and M. D'Esposito. Brain modularity: A biomarker of intervention-related plasticity. *Trends in cognitive sciences*, 23(4):293–304,

2019. ISSN 1879-307X. URL https://doi.org/10.1016/j.tics.2019.01.014.

[33] O. Sporns and R. F. Betzel. Modular brain networks. *Annual review of psychology*, 67:613–640, 2016. ISSN 1545-2085. URL https://doi.org/10.1146/annurev-psych-122414-033634.

[34] Ch. L. Keown, M. C. Datko, C. P. Chen, J. O. Maximo, A. Jahedi, and R.-A. Müller. Network organization is globally atypical in autism: A graph theory study of intrinsic functional connectivity. *Biological psychiatry. Cognitive neuroscience and neuroimaging*, 2(1):66–75, 2017. ISSN 2451-9022. URL https://doi.org/10.1016/j.bpsc.2016.07.008.

[35] M. N. Servaas, L. Geerligs, R. J. Renken, J.-B. C. Marsman, J. Ormel, H. Riese, and A. Aleman. Connectomics and neuroticism: an altered functional network organization. *Neuropsychopharmacology : official publication of the American College of Neuropsychopharmacology*, 40(2):296–304, 2015. ISSN 1740-634X. URL https://doi.org/10.1038/npp.2014.169.

[36] M. Conover, J. Ratkiewicz, M. Francisco, B. Goncalves, F. Menczer, and A. Flammini. Political polarization on twitter. *Proceedings of the International AAAI Conference on Web and Social Media*, 5(1):89–96, 2021. URL https://ojs.aaai.org/index.php/ICWSM/article/view/14126.

[37] A. L. Schmidt, F. Zollo, M. Del Vicario, A. Bessi, A. Scala, G. Caldarelli, H. E. Stanley, and W. Quattrociocchi. Anatomy of news consumption on facebook. *Proceedings of the National Academy of Sciences*, 114(12):3035–3039, 2017. URL https://www.pnas.org/doi/abs/10.1073/pnas.1617052114.

[38] R Wan and J. Cai. Community detection using an optimized label propagation algorithm. In *2013 International Conference on Cloud Computing and Big Data*, pages 360–365, 2013. URL https://ieeexplore.ieee.org/abstract/document/6821016.

[39] J. Bulthé. *Of dots and digits. Advanced neuroimaging analyses applied to the numerical brain.* PhD thesis, 2017. URL https://lirias.kuleuven.be/retrieve/453549.

[40] National Institute of Biomedical Imaging, U.S. Department of Health Bioengineering, and Human Services. Magnetic resonance imaging (MRI). https://www.nibib.nih.gov/science-education/science-topics/magnetic-resonance-imaging-mri.

[41] P. Sprawls. *Magnetic Resonance Imaging: Principles, Methods, and Techniques*. Medical Physics Publishing Corporation, 2000. ISBN 9780944838976. URL https://books.google.cz/books?id=igVzPQAACAAJ.

[42] M. P. van den Heuvel and H. E. Hulshoff Pol. Exploring the brain network: A review on resting-state fmri functional connectivity. *European Neuropsychopharmacology*, 20(8):519–534, 2010. ISSN 0924-977X. URL https://doi.org/10.1016/j.euroneuro.2010.03.008.

[43] L. Jajcay, D. Tomeček, J. Horáček, F. Španiel, and J. Hlinka. Brain functional connectivity asymmetry: Left hemisphere is more modular. *Symmetry*, 14(4), 2022. ISSN 2073-8994. URL https://www.mdpi.com/2073-8994/14/4/833.

[44] The principles of nerve cell communication. *Alcohol health and research world*, 21(2):107–108, 1997. ISSN 0090-838X. URL https://pubmed.ncbi.nlm.nih.gov/15704344.

[45] M. F. Bear, B. W. Connors, and M. A. Paradiso. *Neuroscience: exploring the brain*. Lippincott Williams and Wilkins, 3 edition, 2007. ISBN 0-7817-6003-8.

[46] B. Bučková, J. Kopal, K. Řasová, J. Tintěra, and J. Hlinka. Open access: The effect of neurorehabilitation on multiple sclerosis—unlocking the resting-state fmri data. *Frontiers in Neuroscience*, 15, 2021. ISSN 1662-453X. URL https://www.frontiersin.org/article/10.3389/fnins.2021.662784.

[47] A. Compston and A. Coles. Multiple sclerosis. *The Lancet*, 372:1502–1517, 2008. URL https://doi.org/10.1016/S0140-6736(08)61620-7.

[48] National Multiple Sclerosis Society. Types of MS. https://www.nationalmssociety.org/What-is-MS/Types-of-MS. Accessed: 2022-04-19.

[49] V. Fleischer, A. Radetz, D. Ciolac, M. Muthuraman, G. Gonzalez-Escamilla, F. Zipp, and S. Groppa. Graph theoretical framework of brain networks in multiple sclerosis: A review of concepts. *Neuroscience*, 403:35–53, 2019. ISSN 0306-4522. URL https://www.sciencedirect.com/science/article/pii/S0306452217307613. Non-invasive MRI windows on brain inflammation.

[50] O.L. Gamboa, E. Tagliazucchi, F. von Wegner, A. Jurcoane, M. Wahl, H. Laufs, and U. Ziemann. Working memory performance of early ms patients correlates inversely with modularity increases in resting state functional connectivity networks. *NeuroImage*, 94:385–395, 2014. ISSN 1053-8119. URL https://doi.org/10.1016/j.neuroimage.2013.12.008.

[51] M. Tahedl, S. M. Levine, M. W. Greenlee, R. Weissert, and J. V. Schwarzbach. Functional connectivity in multiple sclerosis: Recent findings and future directions. *Frontiers in Neurology*, 9, 2018. ISSN 1664-2295. URL https://www.frontiersin.org/article/10.3389/fneur.2018.00828.

[52] T. Welton, C. S. Constantinescu, D. P. Auer, and R. A. Dineen. Graph theoretic analysis of brain connectomics in multiple sclerosis: Reliability and relationship with cognition. *Brain connectivity*, 10(2):95–104, 2020. ISSN 2158-0022. URL https://doi.org/10.1089/brain.2019.0717.

[53] K. Řasová, B. Bučková, T. Prokopiusová, M. Procházková, G. Angel, M. Marková, N. Hrušková, I. Štětkářová, Š. Špaňhelová, J. Mareš, J. Tintěra, P. Zach, V. Musil, and J. Hlinka. A Three-Arm parallel-group exploratory

trial documents balance improvement without much evidence of white matter integrity changes in people with multiple sclerosis following two months ambulatory neuroproprioceptive "facilitation and inhibition" physical therapy. *Eur J Phys Rehabil Med*, 57(6):889–899, February 2021. URL `https://pubmed.ncbi.nlm.nih.gov/33565742/`.

[54] S. Whitfield-Gabrieli and A. Nieto-Castanon. Conn: a functional connectivity toolbox for correlated and anticorrelated brain networks. *Brain Connect*, 2(3):125–141, 2012. URL `https://pubmed.ncbi.nlm.nih.gov/22642651/`.

List of Abbreviations

- **AAL** Automated Anatomical Labeling
- **BMI** Body Mass Index
- **EDSS** Expanded Disability Status Scale
- **FC** Functional Connectivity
- **LPA** Label Propagation Algorithm
- **MLM** Multi-Layer Modularity – denotes the Louvain algorithm
- **MRI** Magnetic Resonance Imaging
  - **fMRI** Functional MRI
  - **BOLD fMRI** Blood Oxygenation Level-Dependent fMRI
- **MS** Multiple Sclerosis
  - **PP** Primary Progressive (type of MS)
  - **RR** Relapsing-Remitting (type of MS)
  - **SP** Secondary Progressive (type of MS)
- **RF** Radio Frequency