Community detection and motifs discovery for the characterization of transcriptional regulatory networks of yeast species

Fábio Cruz[1]
fabio.rocha.cruz@tecnico.ulisboa.pt

Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal

Abstract. Genomic expression of living organisms is controlled by complex transcriptional regulation. Transcriptional regulatory networks are responsible for representing and controlling this gene expression. These networks are composed of associations between transcription factors and target genes, which can regulate the response of an organism to environmental changes. In this proposal, we aim at studying different strategies to successfully characterize closely-related yeast transcriptional regulatory networks through the use of Network Science methods. We give basic concepts and present several community detection algorithms for single-layer networks, as well as many measures to evaluate them. We also present methods to extract structure from multilayer networks, and finally we study relevant algorithms and tools to find motifs, significant subgraphs in networks.

Keywords: Complex Networks · Transcriptional regulatory networks · Community detection · Network motifs · Multilayer networks.

Summary

1 Introduction

Transcriptional regulation is an important mechanism for the control of gene expression, which allows the organism to respond to changes in the environment by altering the production of specific gene products. This control is managed by transcription factors (TF) [2] and other proteins, who can activate or repress a wide variety of target genes (TG) to make sure they are expressed at the right time and in the right amount throughout the lifetime of a cell. When a single gene is regulated by two or more transcription factors, who can function independently or with others, we say that we have a combinatorial regulation of gene expression given by a combination of factors to control transcription.

These interactions between transcription factors and target genes define transcriptional regulatory networks, that contain information about the biological functionality of the organisms. These networks have been the subject of several studies [3, 4] where they have been analyzed, thus facilitating the understanding of differential gene expression. Despite this, there is still plenty of uncertainty regarding the structures and dynamics of this type of networks.

In network science, networks that result from complex real-life systems from many areas can be represented by graphs [5], such as citation networks, social networks, biological networks, among others. Furthermore, it is possible to study the structure of those graphs to discover underlying functionality and dynamics that are not obvious at first sight. This can be done by detecting community structure through the use of algorithms for this purpose [6].

In this thesis, we want to characterize transcriptional regulatory networks to expand the scientific knowledge available in this field. We want to do this by detecting important groups of nodes in these networks and understanding their relevance to the biological functions of the organisms that are represented by the networks.

1.1 Objectives

The main goal in this thesis is to characterize the transcriptional regulatory networks of 10 closely-related species, provided by YEASTRACT+ [7], a wide-scope tool for the analysis and prediction of transcription regulatory associations at the gene and genomic levels in yeasts of biotechnological or human health relevance. We will use network science approaches to accomplish this.

The first objective will be to analyze the performance of several community detection algorithms on the identification of clusters on each of these biological networks, as well as assess the impact of the algorithm parameters on the quality of the identified clusters. After that, we want to characterize the functionality of the communities found, with the goal of verifying if the communities are well-defined and what is the relation between them and the different biological functions present in the genetic information of the species.

The next objective is to use a multilayer approach to detect structures and functions across different species. The aim is to analyze the similarities between the closely-related species, and also to verify the possibility of conservation of subnetworks across different species by inferring structure on networks with less available data. Lastly, we want to apply algorithms that aim at discovering motifs with significant representation in the different yeast networks as a complementary approach to traditional community methods, and we want to compare the obtained results from the discovered subgraphs with the ones obtained from other approaches.

The remainder of this proposal is organized as follows: in Section 2, we talk about basic concepts of Network Science, including simple notions of multilayer networks and network motifs; in Section 3, we present and analyze several community detection algorithms for single-layer and multilayer networks, as well as some evaluation metrics and network motifs discovery algorithms; and in Section 4, we describe a solution proposal for the problem of this thesis, and metrics to better quantify the quality of those solutions.

2 Background

In this section, we will introduce some basic concepts of network science that we will use later in the document, and briefly present the networks that we will be analysing and as well as some of their properties.

2.1 Network science

Network science is a field that studies complex systems in a wide range of areas, such as technological, biological and social ones. The main purpose of network science is to design common principles, algorithms and tools that can define and explain certain phenomena that are implicit in the networks that represent those systems. The most famous reference for network science is the Barabási book about this subject [5].

2.1.1 Basic concepts These system networks can be represented by graphs, mathematical structures used to model interactions between its components. Graph theory is the mathematical field that is responsible for the study of graphs.

A graph is composed of a set of *vertices*, and a set of *edges* that connect one vertex to another. The equivalent nomenclature for vertices and edges in networks is *nodes* and *links*, respectively. The meaning of the nodes and their connections can vary depending on the type of system we are trying to represent. The number of nodes of a graph represents the number of components in the system, and the number of edges is the total number of interactions between the nodes. These links can be *undirected* or *directed*, depending on if the interaction is symmetrical between the nodes, or if the set of edges is represented by an ordered pair of nodes, with an origin node and a destination node. A network is called directed (*digraph*) if all of its connections are directed, as shown in Fig.1, undirected if none of the links are directed and can even be considered both directed and undirected in some particular cases.

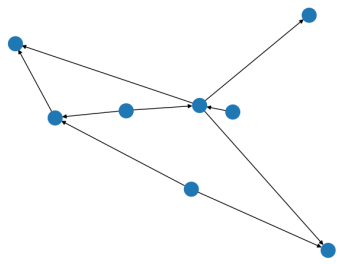

Fig. 1. Digraph generated in the Python package NetworkX. The blue dots represent the graph nodes, and the arrows are the directed links.

Some systems may need to be represented as *weighted graphs*, where edges have weights associated with them, meaning the importance of links can differ in a given context instead of all having the same weight. Graphs can also be *signed*, when the weight of the edges only has two values, usually -1 or 1, and the relation is considered either *negative* or *positive*, respectively. *Bipartite graphs* are somewhat similar to signed ones by also separating the graph into two groups of nodes. However, in bipartite networks, these sets of nodes need to be disjoint and independent of each other, and each node from one set connects only to nodes from the opposite set.

Graphs can be divided into several components or *subgraphs*. Subgraphs are sets of nodes and edges that are contained in the original graph. A fully connected subgraph, where each node has a link to every other of the group, is called a *clique*. A *k-clique* graph is a fully connected subgraph with k nodes.

Alternatively to graphs, we can also represent networks as a *list of edges*, an *adjacency matrix* or *adjacency list*. Depending on the data structure we choose, the performance when accessing or adding/removing elements might be different. While the list of edges structure is self-explanatory, the adjacency matrix consists of a square matrix whose elements $A_{i,j}$, where i and j represent nodes from the network, can have the values 0 if the nodes are not connected, or 1 otherwise. As for adjacency lists, each node has associated a list with all the nodes that are connected with itself. The adjacency matrix can be very efficient when searching, inserting or removing links, with a time complexity of $O(1)$ in the worst-case scenario. However, the main drawback is that this data structure requires plenty of memory space ($O(N^2)$, where N is the number of nodes) when representing large graphs to create or even copy the data. On the other hand, adjacency lists are more efficient regarding this and inserting edges, but are worse at finding and removing them.

2.1.2 Network properties A key property of each network node is the *degree*, often represented by k, which is the number of links that are connected between itself and other nodes. In an undirected network, the total number of links can be expressed as the sum of all node degrees divided by 2. This node property allows for the existence of another property, this time directly related to the network, the *average degree*. In undirected networks, the average degree can be calculated as follows, where N is the total number of nodes of the network, L is the number of links and k_i is the degree of a node:

$$\langle k \rangle = \frac{1}{N} \sum_{i=1}^{N} k_i = \frac{2L}{N} \tag{1}$$

As for directed graphs, we can separate the degree in internal and external. The *internal degree* or *indegree* of a node is the number of links that point towards itself, while the *external degree* or *outdegree* is the number of links that point to other nodes. The total average degree of a directed network can be calculated by the sum of the average in-degree and the average outdegree.

Another important notion to have is the *degree distribution* of a network. The degree distribution, p_k, is the probability that a random node has degree k, and can be represented through a function graph, where the x-axis is the degree k and the y axis is p_k. The shape of the degree distribution graphic representation can tell us some interesting things about our network. This is particularly relevant in *scale-free networks*, which we will briefly discuss later.

In networks, the "distance" between two nodes is given through the concept of *path length*. A network *path* is the sequence of links that one has to traverse in order to reach the desired destination node from the origin point. The *shortest path* between nodes i and j is the path with the smallest length, that is, the least amount of links one has to cross from one node to another. In undirected networks, the distance from i to j is the same as from j to i, but this is not true for directed networks. The *diameter* of a network is defined as the longest shortest path in a graph or the distance between the two furthest nodes. The most important measure related to distances in networks is the Average Path Length (APL), $\langle d \rangle$. It is the average distance between all pairs of nodes in a network. That distance d can be calculated with the Breadth-First Search (BFS) algorithm [50]. APL is calculated as follows:

$$\langle d \rangle = \frac{1}{N(N-1)} \sum_{i \neq j} d(i,j) \tag{2}$$

The *clustering coefficient* of a node, C_i, captures the degree to which the neighbours of some node link to each other. The degree of clustering of a whole network can also be used to characterize it, and is given by the average clustering coefficient $\langle C \rangle$:

$$\langle C \rangle = \frac{1}{N} \sum_{i=1}^{N} C_i \tag{3}$$

Sometimes it can be useful to characterize singular nodes instead of the whole network to understand which vertices are the most important or central in the network. There are several measures for this purpose, depending on the context. The most obvious one is the *Degree centrality*, which is simply the degree of the node. Instead of giving each neighbour node the same importance, the *Eigenvector centrality* gives each node a score proportional to the sum of the scores of its neighbours. Another approach is the *Closeness centrality* which measures how close an individual is to the rest of the nodes, which is based on the APL distance measure. Finally, the last strategy to understand the centrality of a node is the *Betweeness centrality*. This measures the extent to which a vertex lies on paths between other vertices. With σ_{st} being the total number of shortest paths from node s to t and $\sigma_{st}(i)$ the number of shortest paths between those vertices that pass through node i, the Betweenness centrality C_B for this node is given by:

$$C_B(i) = \frac{1}{N^2} \sum_{s \neq t \neq i} \frac{\sigma_{st}(i)}{\sigma_{st}} \tag{4}$$

2.1.3 Random networks Network science aims to build models that reproduce the properties of real networks. At first look, most networks that we find in nature do not have a clear-cut structure, but rather look like they were created randomly. Random network theory takes this into account to model and characterizes random graphs.

The simplest model to create random networks consists of N nodes where each pair is connected with a probability p. Another way of doing this is to connect the nodes by E randomly placed links. This is the *Erdős–Rényi model* [8], also called the *null model* of network science. The ER network follows an interesting property, the *small-world effect* [9]. This property states that if you choose any two individuals anywhere on Earth, you will find a path of at most six hops between them. This means a small-world network is a graph in which most nodes are not neighbours of one another, but the neighbours of any given node are likely to be neighbours of each other and most nodes can be reached from every other node by a small number of hops or steps. The problem with this model is that it does not generate graphs with high clustering coefficients and they do not account for the formation of hubs. To solve this, the *Watts-Strogatz model* [10], a random graph generator that produces graphs specifically with small-world properties, including short average path lengths and high clustering coefficients, was then introduced. This was accomplished by initially creating a ring of nodes connected only to their direct neighbours, and then rewiring each link to a randomly chosen node with a probability p, which can range between 0 and 1.

A *scale-free network* is one whose degree distribution follows a *power-law*, that is, where $p_k = k^{-\gamma}$ where γ is a parameter whose value is typically between 2 and 3. This means that this type of networks have degree distributions where a small number of nodes have a very high degree, also called *hubs*, and a large number of nodes have small degrees, as shown in Fig. 2.

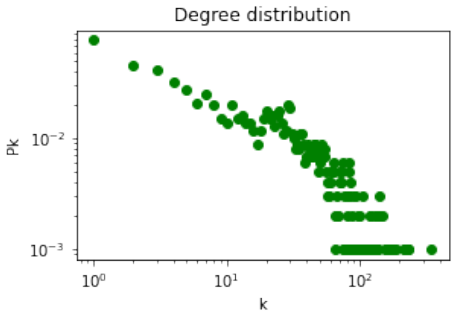

Fig. 2. Power-law degree distribution in a log scale. k represents the degree of the nodes.

The *Barabási–Albert model* [11] is another algorithm that generates random graphs, in particular, scale-free networks using a *preferential attachment* mechanism. Initially, a ring with m_0 nodes is created and at each time step, a node is added with m links. Then, preferential attachment comes into play. When each new node is being created, they are connected to older nodes with a probability proportional to the degree of the older nodes. The higher the degree, the more likely it is to be connected with a new node. This model is very interesting because many large real-life networks are approximately scale-free networks, such as the World Wide Web, citation networks and some protein interaction networks. It also takes into account that most real networks are not static, they can grow over time.

2.1.4 Modularity *Modularity* is a metric of the structure of networks that measures the quality of a network division into modules. Its values usually vary between -1 and 1, depending on how it is calculated. Networks with a high modularity value have groups of nodes that are densely connected with each other and sparsely connected with nodes from other modules or clusters. Partitions, where every node belongs to the same cluster, have a modularity of 0, and bad partitions have negative values.

Modularity is often used in the implementation of several algorithms that aim at detecting *communities* [12], well-defined groups of nodes that tend to play similar roles in real networks, similar to what we see in Fig. 3. We will analyze some of these methods in the related work section. The main drawback of modularity is that it suffers from a *resolution limit*, which means that it has problems when detecting small-sized communities.

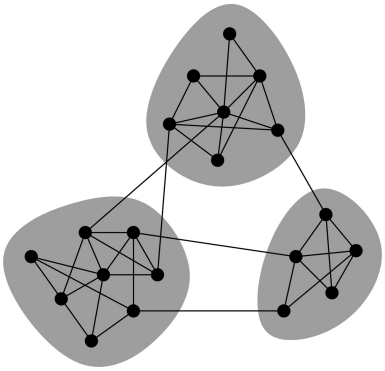

Fig. 3. Community structure example. Image taken from [12].

Considering a network with N nodes, L links and n_c being the number of clusters of a partition, and a network cluster c with N_c nodes and L_c links. If k_c is the total degree of the nodes in a cluster, the modularity value of the network

can be calculated as:

$$M_c = \sum_{c=1}^{n_c} \left[\frac{L_c}{L} - \left(\frac{k_c}{2L} \right)^2 \right] \tag{5}$$

2.1.5 Multilayer In most natural and engineered systems, a set of entities interact with each other in complicated patterns that can encompass multiple types of relationships. Such systems include multiple subsystems and layers of connectivity, and it is important to take such *multilayer* features into account to try to improve our understanding of complex systems. Nowadays, the study of multilayer networks has become one of the most important fields in network science [13, 14]. In this section, we briefly present some fundamental concepts about multilayer networks, as well as a basic definition of *network motifs*.

Multilayer networks are composed by *layers*, each having different types of relations. In the most general multilayer network concept, nodes are allowed to belong to any subset of the layers, and edges can be connected to any node from any layer. Edges connecting nodes from the same layer are named *intra-layer* edges, while the ones connecting nodes from different layers of the same network are called *inter-layer* edges. When nodes from different layers are the equal and they are connected, we say they are *couplings*.

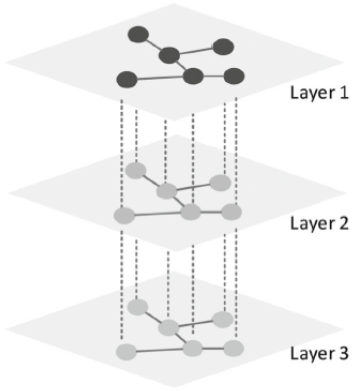

Fig. 4. Multilayer network with three connected layers, representing inter-layer and intra-layer edges. Image taken from [80].

Differently from the traditional single-layer approach, where node properties are described by scalar variables, node features in multiplex networks are naturally described in vectorial terms. In single-layer networks, one of the main centrality measures is the degree of each node: the more links a node has, the more important the node is. If l represents a layer and L the full set of layers, the centrality of node i , e.g. the degree of a node i in a multilayer network is

defined as:

$$k_i = \sum_{l=1}^{L} k_i^l \tag{6}$$

A crucial empirical evidence is that in many multilayer networks not all nodes have connections at all layers. As a consequence, a node i is defined as active on a layer α if it is connected to at least another node at that layer, i.e. if $k_i^{[\alpha]} > 0$. The activity-pattern of each node can be compactly stored into the following node activity vector, where each vector entry is 1 if node i is active in layer α, and 0 otherwise, where M is the number of layers:

$$b_i = \{b_i^{[1]}, ..., b_i^{[M]}\} \tag{7}$$

Many other node, edge and layer properties were created or extend for multilayer networks, namely the node overlapping degree, participation coefficient, clustering coefficient, eigenvector centrality, degree correlations, among many others. Complex networks are usually characterised by non-trivial structural patterns not only at the level of node properties but also, and more importantly, at the level of subgraphs. A lot of attention has been devoted to the analysis of statistically significant subgraphs in single-layer and multilayer networks, also known as *network motifs*. Network motifs were first defined by Milo *et al.* as patterns of interconnections occurring in complex networks in numbers that are significantly higher than those in similar randomized networks [15]. The most common way of calculating the statistical significance of a network motif is by using the Z-score measure:

$$Z - score = \frac{f_{original} - \langle f_{random} \rangle}{\sigma(\langle f_{random} \rangle)}, \tag{8}$$

where $f_{original}$ is the frequency of a given motif in the original network, $\langle f_{random} \rangle$ is the average of the frequency of the motif in the random networks and σ is the standard deviation. Z-score, or standard score, is the number of standard deviations by which the value of the observed score is above or below the mean value of what is being measured. One interesting type of motifs is *triads*, which are 3 node subgraphs. In multilayer networks, *m-triads* are triads that are composed by edges from m different layers.

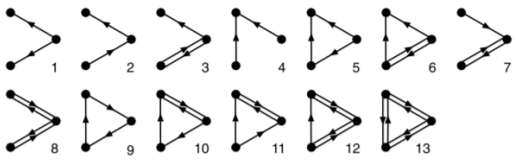

Fig. 5. All 13 types of three-node connected subgraphs (triads). Image taken from [15].

2.2 Yeast networks

In this section, we will introduce the networks from yeast species that we will be working on during this thesis, as well as briefly analyze some of their structural characteristics.

These are 10 transcriptional regulatory networks of closely-related yeast species, retrieved from YEASTRACT+. The networks are directed, where each origin node represents a transcription factor that is associated with a target gene. These regulatory associations are essentially supported by DNA binding evidence retrieved from experimental data available, and/or expression evidence. Most of the links of the last-mentioned group can still be divided into four types, that are represented by a "signed" annotation in the network. Depending on the effect the TF has on the TG, edges can be positive if the TF is an activator, negative if it is a repressor or dual/unknown when it has a dual effect or the directionality of the effect is unknown [16].

Network	Nodes	Edges	TF	TG	$\langle k \rangle$	$\langle k_{in} \rangle$	$\langle k_{out} \rangle$	CC	APL
S. cerevisiae	6 886	195 498	220	6 886	56.78	28.39	28.39	0.47	0.059
C. albicans	6 015	35 687	118	6 015	11.87	5.93	5.93	0.28	0.034
Y. lypolytica	5 288	9 238	5	5 288	3.49	1.75	1.75	0.36	0.001
C. parapsilosis	3 381	6 986	11	3 380	4.13	2.07	2.07	0.25	0.003
C. glabrata	2 133	3 508	40	2 116	3.29	1.64	1.64	0.05	0.002
C. tropicalis	665	698	16	663	2.10	1.05	1.05	0.01	0.041
K. pastoris	561	581	4	559	2.07	1.04	1.04	0.01	0.002
K. lactis	111	126	10	106	2.27	1.14	1.14	0.15	0.014
Z. bailii	32	31	1	31	1.94	0.97	0.97	0.00	0.031
K. marxianus	4	3	1	3	1.50	0.75	0.75	0.00	0.250

Table 1. Yeast network properties. The properties shown are the number of nodes of each network, number of edges, number of transcription factors, number of target genes, average degree, internal average degree, external average degree, average clustering coefficient and average path length (the value of the largest component was chosen when applied), in this order.

In Table 1, we can observe some basic properties of the yeast networks, organized by the number of nodes, that help us understand how they are composed. The first thing that pops up is that some of the networks are severely underdocumented, as we can see from the number of links of the bottom five networks, all below a thousand. We can also see that, even for the networks with a high number of nodes and edges, some might not be as relevant as others when it comes to genetic information, due to the low number of TFs. One example of this is the *Y.lypolytica* network that only has five TFs, meaning that all the associations (almost ten thousand) come from the same five origin nodes.

Another interesting observation is that the average indegrees and average outdegrees are equal in all networks. This makes sense since, for example con-

sidering the smallest network, if one TF is connected to three TGs, the TF has an outdegree value of three and all the TGs have an outdegree of zero giving an average outdegree of $3/4 = 0.75$, and the same value appears for the in-degree since the indegree of the TF is zero and of the three TGs is 1, giving again an average indegree of 0.75. Even if the TFs have a connection to themselves or the TGs connect to other nodes, this would still happen. This means that maybe it would be more interesting to verify the degrees of specific individual nodes that might be relevant for a network during the implementation of this thesis. As for the clustering coefficient, we can conclude that only the four largest networks have well-clustered groups of nodes while the others are small and sparse. The average path length has very low values in general since they are directed and mostly composed by big hubs where the TF can reach the TGs with distance one, but the TGs can not connect to each other, giving a distance of zero. With that being said, the number of TGs in all networks is much larger than the number of TFs, so the average path length will always tend to zero.

3 Related Work

In the previous section, we introduced some basic concepts of network science and briefly described the yeast networks we are going to work with. In this section, we will cover state-of-the-art work that has been published regarding algorithms for community detection and measures to evaluate them, as well as multilayer network approaches and motifs discovery techniques and tools. We would like to highlight the work done by Paulo Dias in his master's thesis [1] where he also researched on the same networks.

3.1 Community detection

In network science, communities are groups of nodes that are more likely to connect with each other than to nodes from other communities [5], and in some cases, communities can overlap, which means some nodes can belong to more than one. We can find community structure in many real networks from different areas such as social and biological networks. The study of this network property is important because nodes from the same community tend to play similar roles or functions in the network, so this allows us to better understand their underlying structural importance for the network. For instance, communities in social networks can represent real social groups between entities and in biological networks they can represent biological functions composed of different genes [6].

There is a big collection of algorithms proposed in recent years for the detection of communities in real networks that use different strategies. Table 2 shows us some of the algorithms that we will now look at in more detail. Some of them have been analysed and compared by Fortunato and Lancichinetti [17, 18], so we will base some of our analysis on their papers.

Since the community structure of a network is often unknown, we cannot use brute-force cluster methods; its time complexity would not be practical. We need

Algorithm	Directed	Weighted	Overlapping	Signed
Modularity optimization				
Greedy Modularity [20]		X		
Clauset *et al.* [21]	X	X		
Guimerà-Amaral [22]	X	X		
GA-Net [23]				
Louvain [25]		X		
Leiden [26]		X		
AGDL [27]	X	X		
Paris [28]				
EdMot [29]				
Divisive				
Girvan-Newman [6, 30]	X			
Radicchi *et al.* [31]				
Donetti-Muñoz [32]				
EigenVector [33]				
Ronhovde-Nussinov [34]	X	X		
Overlapping				
CFinder [35]			X	
Link Clustering [36]			X	
CONGO [38]			X	
WalkSCAN [39]			X	
Signed				
Anchuri *et al.* [40]				X
Esmailian-Jalili [41]				X
Others				
Markov Clustering [42]				
Walktrap [43]				
Label Propagation [44]				
Newman-Leicht [45]	X			
Infomap [46]	X	X	X	
Surprise [47]	X	X		
Ricci [48]				
CONDOR [49]		X		

Table 2. Community detection algorithms, inspired by P.Dias [1]. The implementations of most the algorithms are available in the Community Discovery Library CDlib [102].

algorithms that can discover communities without iterating through all partitions of a network. Besides this, networks may have an inherited hierarchical structure where some groups of nodes can be inside others. This brings us to the first group of algorithms that we present. *Hierarchical clustering* is a way of grouping data/nodes into clusters with the objective of iteratively producing a set of clusters with a hierarchical structure where nodes from the same cluster have a high similarity between each other. Usually, this is accomplished using a similarity matrix, whose entries represent the distance between two nodes of a network, which is calculated using a similarity measure that can vary depending on the algorithm used. There are two types of strategies for hierarchical clustering: *agglomerative* and *divisive* [5, 17].

3.1.1 Modularity optimization methods Modularity is a measure of the quality of a given community partition of a network. If a partition has a high modularity value (ranges between -1 and 1), the better the community structure of the network is. Partitions with only one community have a value of 0, and bad partitions have a negative modularity [67].

The first modularity based algorithm proposed by Newman, the *greedy modularity maximization algorithm* [20], aimed at choosing a partition that had the maximum possible modularity without having to check all possible partitions of a network, since this would not be computationally feasible. Instead, it starts by assigning a community to each node, and iteratively joins pairs of communities if this improves the modularity of the partition. In the end, it selects the partition for which the modularity is maximized. Another way of calculating modularity, and currently the most popular one, is the Newman-Girvan modularity [30], which is used in the *Girvan-Newman algorithm* for community detection. We will talk about this in the next class of algorithms.

Clauset *et al.* [21] introduced the *fast greedy modularity optimization algorithm*, which is basically a faster implementation of the Newman algorithm previously referred, by using more efficient data structures. Guimerà and Amaral also tried to improve the Newman greedy solution by optimizing modularity via *simulated annealing* with the *Guimerà-Amaral algorithm* [22]. Simulated annealing is a stochastic optimization technique for approximating the global optimum of a given function, that allows the discovery of good partitions by introducing a computational temperature. Pizzuti with the *GA-Net algorithm* [23] and Shuzhuo *et al.* [24] used genetic-based approaches to maximize modularity. GA-Net optimizes a fitness function known as community score in order to identify well-connected nodes.

Both the simulated annealing and genetic algorithm approaches aimed at improving the accuracy of the communities found, however, this happens at the expense of computational speed. Blondel *et al.* introduced the *Louvain algorithm* [25] in 2008. This method became the most popular algorithm based on modularity optimization because it offered a great compromise between finding well-connected groups of nodes and also being very efficient from a computational point of view ($O(m)$ for sparse graphs, where m is the number of edges).

Initially, the algorithm assigns a different community for each node of the network. Then for each node, it calculates the Newman-Girvan modularity value if the node is changed from the current community to a neighbourhood community. The change will occur if modularity is maximized that way, otherwise nothing changes. This process is repeated until no more improvements to the modularity of the network can be done. After this, the algorithm transforms the current communities into nodes and reapplies the previous procedure. Again, this is repeated until no more improvements are accomplished in the network, thus allowing the algorithm to find communities inside other communities. However, one of the biggest problems of the Louvain algorithm is that it can find weakly connected communities. In order to solve this, the *Leiden algorithm* [26] was proposed by V.A. Traag *et al.*, which ensures well-connected communities and even is much faster than Louvain. Both methods work very similarly, but Leiden has an additional phase to try to refine the discovered partitions. Communities found in the first step of the Louvain algorithm may split again into more partitions in the next phase, but in the additional phase of Leiden, nodes can be randomly merged with communities, which increases the quality function. Besides this, after the first check of all nodes has been done, Leiden only revisits nodes whose neighbours have been changed, instead of revisiting all nodes again.

Other hierarchical agglomerative methods based on the optimization of modularity were proposed in recent years, such as the *AGDL algorithm* [27] in 2012, the *Paris algorithm* [28] in 2018 and *EdMot* [29] in 2019. The AGDL algorithm uses indegrees and outdegrees to characterize the affinity between two clusters, so it supports directed and also weighted graphs. The Paris algorithm is also modularity-based and is based on a simple distance metric between clusters induced by the probability of sampling node pairs. Finally, the EdMot algorithm uses higher-order motifs to initially create a new graph and then uses the Louvain algorithm to create the partitions.

3.1.2 Divisive methods Divisive methods aim at removing network edges/links that connect different communities, which results in a set of isolated clusters usually represented by a dendrogram. In order to do this, we need a measure to decide what edge we want to remove. Several algorithms were proposed in the last few years that use different ways of calculating this measure.

One of the most popular divisive methods is the *Girvan-Newman algorithm* (GN) [6, 30], proposed by Michelle Girvan and Mark Newman in 2004. It consists in calculating a centrality measure for the edges of the network and then removing the one with the highest centrality consecutively since high centrality links tend to connect well-separated groups of nodes. More specifically, they used the notion of edge betweenness, which consist of how frequently an edge is used in the shortest path of a pair of nodes. Using the *Breadth-First Search* (BFS) algorithm for the calculation of the shortest paths [50], the Girvan-Newman has, in the worst case, a time complexity of $O(nm^2)$(where n is the number of nodes of the network, and m the number of links). After this step, we choose where we want to cut the dendrogram to obtain the optimal partition of the graph.

This choice is made by calculating the modularity for each possible partition and picking the one with the highest value.

Radicchi *et al.* also proposed a divisive algorithm [31] similar to the Girvan-Newman, but instead of using the edge betweenness measure to cut the links of the communities, it uses the edge clustering coefficient, which is the ratio between the number of loops based on the link and the maximum possible number of loops. The time complexity for sparse graphs for this method is $O(m^2)$, thus being slightly faster than the Girvan-Newman algorithm. In the same year, Donetti and Muñoz introduced the *spectral algorithm* [32]. This method assumes that if a network has well-connected clusters, the eigenvector components of the nodes should have similar values. In particular, this algorithm focuses on the eigenvectors of the *Laplacian matrix*. The clustering is done using traditional hierarchical clustering techniques, and it is chosen the partition that maximizes the Newman-Girvan modularity measure. A few years later, Newman proposed a new divisive method based on *eigenvectors* [33] where, again, each split of the communities is done by maximizing the modularity of the network.

Another approach worth mentioning is the *Ronhovde-Nussinov* [34], a *Potts model* [51] community detection algorithm based on minimizing the Hamiltonian of a Potts-like spin model, where the spin state represents the membership of the node in a given community. This method is known to have a complexity almost linear to the number of links, and to give good results since it avoids the "resolution limit" that is well-known to be one of the biggest problems of modularity-based methods [18].

3.1.3 Overlapping methods So far we have presented algorithms where each node can only belong to one community, however, real networks often have vertices that can be shared by different communities. Concerning this, in the last few years this topic has become popular and many algorithms that consider these overlapping communities have been developed, as well as papers comparing the performance and accuracy of some of these methods [19].

The most popular overlapping technique is the *Clique Percolation Method* (CPM) [52], which assumes that a community consists of overlapping sets of subgraphs and detects communities by searching for adjacent cliques. It initially identifies all k-cliques (cliques of size k) and then connects nodes if their k-cliques share *k-1* members. Vertices can belong to several cliques, thus allowing overlapping between communities. *CFinder* is the algorithm that implements this technique [35], but its main issue is that it can be very slow, sometimes unfeasible for large networks, having a time complexity of $O(e^n)$.

Ahn *et al.* proposed a different solution to find overlapping communities, the *Link Clustering algorithm* [36]. This method consists of partitioning links instead of nodes, using edge similarity with the *Jaccard index*. Single-linkage hierarchical clustering then builds a link dendrogram, which is cut at some threshold yielding link communities. The selection of the best partition of the dendrogram is done using a measure called *partition density*, which calculates the link density of a partition.

More recently, Gregory extend the Girvan-Newman divisive algorithm by allowing a node to split into multiple copies with the *CONGA method* [37]. It uses edge betweenness and *splitting betweenness*, which is defined by the number of shortest paths that would run between two parts of a node if it was split. The algorithm iteratively removes the edge with the highest edge betweenness or splits the vertex with maximum split betweenness. However, this algorithm suffers from the same problem as many overlapping methods, having a high time complexity for sparse graphs, $O(n^3)$ in this case. To solve this, Gregory proposed an optimized solution for the same algorithm, *CONGA Optimized*, or CONGO [38]. The main difference is that CONGO uses local betweenness to optimize the speed of the algorithm. A different solution but also recent is the *WalkSCAN algorithm* [39], based on random walks. It uses a simple efficient version of the *PageRank algorithm* and the *DBSCAN algorithm* to detect communities that can overlap.

3.1.4 Alternative methods In this section, we will briefly talk about other community detection algorithms that do not fit the sections mentioned above. Two sets of methods that are worth mentioning are the ones meant to cluster signed networks and bipartite networks. In signed networks, a positive or negative link can be established between two nodes, which can weigh on the importance an edge may have, possibly resulting in a different partition of the network against clustering the same network but with unsigned links. Although this type of community detection has not been thoroughly researched in the scientific community, some work has been done lately. Anchuri *et al.* [40] developed a spectral approach augmented with iterative optimization by trying to minimize *frustration*, the number of edges that participate in unstable configurations which disturb the stability of the network and maximize modularity. Esmailian-Jalili proposed a solution for this [41] that measured the quality of partitions by introducing a new *Map Equation* [53] for signed networks, based on the assumption that negative relations weaken positive flow from a node towards a community, which means good partitions have fewer negative links inside communities. Regarding bipartite networks, nodes can be divided into two disjoint and independent sets, meaning one node can only belong to one of those sets and every edge connects a node from one set to one from the other. *CONDOR* [49] is an algorithm for bipartite community structure detection that identifies groups of SNP (Single-nucleotide polymorphism) that are linked to groups of genes in order to understand their biological function. Some other popular already existing algorithms have also been extended to detect communities in bipartite networks, such as the *Infomap algorithm* that will mention later in this section.

The first community detection method proposed in our list of methods was the *Markov clustering algorithm* (MCA) [42], developed in 2000 by Dongen. The main idea is that if we randomly walk in the network we are more likely to stay inside clusters instead of crossing different ones, so the method uses a probability matrix of a random walk after a number of steps and then raises each element of the matrix by some power to increase the chances of the walker being trapped

within a community. These steps are repeated until we obtain a forest, whose components are the communities. The *Walktrap algorithm* [43] is also based on random walks that are used to compute distances between nodes, which are then assigned into groups with small intra- and larger inter-community distances via bottom-up hierarchical clustering.

In addition to these, *Label Propagation* [44] is a simple and fast algorithm where initially each node is given a different label and then at each iteration, each vertex takes the label shared by the majority of its neighbours, or a random label in case of a tie, until every node has the majority label. Groups of nodes with the same label belong to the same community. Newman and Leicht also introduced a method for community discovery in the same year, the *Expectation-maximization method* (EM) [45]. This algorithm uses *Bayesian inference* to deduce the best fit of a given model to the data represented by the graph structure. The quality of the model is maximized through the expectation-maximization technique. One possible problem of this method is that we need to give the number of communities *a priori*.

One of the most accurate and best-performing algorithm that we present in this section is the *Infomap algorithm* [46], created by Rosvall and Bergstrom. This method is based on optimal compression of information on random walks in the network. Every node is labelled with a code. If a random walker walked inside our network, we would want to describe its path with the least number of symbols possible using those codes. But we know that normally a random walker tends to stay "trapped" for longer when it is inside a community, so we can partition the network and give a code to every community as well so we can optimize and find the shortest message possible with those community codes. Essentially, we associate the paths that are more used by the random walker with edges between nodes that belong to the same community, and paths or edges that are not travelled a lot are probably connecting different clusters of nodes. If we partition the network into too many modules the message encoded becomes bigger than what it needs to be, so this way we can find an optimal partition that assigns nodes to modules in such a way that the information needed to compress the movement of our random walker is minimized. This quality function that we try to optimize to find the best compression is called Minimum Description Length. The time complexity of Infomap is $O(m)$.

The two last methods that we present are the *Surprise algorithm* [47] and the recently proposed *Ricci algorithm* [48]. The Surprise community detection method is similar to Louvain but uses a measure based on classical probabilities known as Surprise, instead of modularity. Nodes are switched between communities such that surprises are greatly improved. The Ricci algorithm presents a novel geometric approach by considering networks as geometric objects and communities as geometric decomposition. It applies curvature and discrete *Ricci flow* to discover community structure in networks.

3.1.5 Evaluation metrics The evaluation of community discovery algorithms is not an easy task. One way to formalize this process is to design a *scoring*

function that outputs a quality score to characterize how much the connectivity structure of a given set of nodes resembles a community [54]. Next, we look at a few strategies to evaluate communities internally and externally, as well as synthetic benchmark networks that allow us to apply and compare the accuracy and performance of different algorithms with networks with ground-truth communities.

Internal evaluation

Regarding the internal evaluation of communities, we present some quality score functions, including variations of the Newman-Girvan modularity measure, that help us summarize the cluster characteristics and even compare these score values with communities generated by different methods.

The simplest measures to characterize communities are the *size*, the number of nodes, and the *average path length*. Both can be computed in each individual community or as an average for the set of all communities, which is also true for most of the measures presented here. The *average transitivity*, *average embeddedness*, *hub dominance* and *surprise* [47] are also scoring functions that can be used to characterize communities. The average transitivity of a community is defined as the average clustering coefficient of its nodes regarding their connection within the community itself, the embeddedness of a node is the ratio of its degree within the community and the overall degree of the node and hub dominance is calculated as the ratio of the degree of the most connected node of a community with relation to the theoretical maximum degree within the community.

We can divide most of the rest of the scoring functions into four groups: scoring functions based on *internal connectivity*, based on *external connectivity*, based on both internal and external connectivity and based on modularity. The *average internal degree*, the *average number of edges*, the *internal edge density*, the *fraction over a median degree* (FOMD) and the *triangle participation ratio* (TPR) are some of the measures related to internal connectivity [31, 54]. The first three are self-explanatory, while the FOMD quality score function is the fraction of community nodes having a higher internal degree than the median degree value, and the TPR is the fraction of community nodes that belong to a triad (a triangle composed of connected nodes). As for external connectivity based scoring functions, we have the *expansion* [31] and the *cut ratio* [17]. The first one measures the number of edges of each node that points outside the cluster, and the second is the fraction of existing edges leaving the community, out of all possible edges. Regarding examples based on both types of methods mentioned above, we have the *conductance* and the *normalized cut scoring* functions [55]. Conductance is defined as the fraction of total edge volume that points outside the community, while the normalized cut is a normalized variant of the cut ratio measure.

Regarding the last of these four groups of measures, we have modularity. As we discussed previously, the most popular modularity metric proposed is the Newman-Girvan modularity [30], calculated by the difference between the fraction of intra-community edges of a partition with the expected number of such

edges if distributed according to a null model. However, this modularity has a few limitations, as the largest modularity value does not necessarily correspond to the best community structure, because the measure has a preferential community scale. Furthermore, the maximum modularity of partitions of random graphs without group structure, e.g. Erdős–Rényi graphs, can still attain surprisingly high modularity values. According to GN modularity, then, such random networks do have a community structure, against intuition. Other modularity quality functions were proposed to try and solve these issues, or just as alternatives for specific types of networks. The first one we present is the *Erdős–Rényi modularity*, which is a variation of the GN modularity that assumes that vertices in a network are connected randomly with a constant probability p. Another alternative is the *link modularity* [58], a modularity quality function designed for directed graphs and overlapping communities. Finally, we have the *Z-modularity measure* [59], another variant of the standard modularity recently proposed to avoid the resolution limit. The concept of this version consists in that the difference between the fraction of edges inside communities and the expected number of such edges in a null model should not be considered as the only contribution to the final quality of community structure. It is called Z-modularity because it measures the Z-score of a given partition with respect to the fraction of the number of edges within communities.

The concept of *significance* of a given partition is also relevant, since high values of modularity do not always translate to a good community structure. The significance measure estimates how likely a partition of dense communities appears in a random graph. Lancichinetti *et al.* [56] addressed this by comparing the community structure with one from a random graph with similar properties. This approach, called *C-score*, calculates the probability that the internal degree of the vertex with the lowest degree in the correspondent random graph is equal to or larger than the node with the lowest internal degree in the original network. In order to improve the performance of C-score, the same authors proposed the *B-score* which used a list of low internal degree nodes instead of only the worst one to compute the statistical significance of a community. This metric is calculated as the minimum probability that the sum of the worst nodes in a random graph is lower than the ones from the original community.

Significance has also been related to the *robustness/stability* of a partition against random perturbations of the network structure. Gradually modifying the structure of the graph via random perturbations, consisting of random rewiring a growing fraction of links, until communities are disrupted and the network becomes equivalent to a random graph is one way to do this. Karrer *et al.* [57] considered this in 2008 when they proposed a strategy where an edge is removed with probability α and replaced it with another one chosen at random with another given probability, creating a perturbation in the original network without changing the degree sequence of the graph. By varying this variable from 0 to 1 we interpolate between the original graph without perturbations and the Newman-Girvan modularity null model. Then the partitions of the original and final graph are compared using the variation of information metric, thus assess-

ing the stability of the network community structure. A similar approach was recently proposed by Silva *et al.* [68]. Modularity can be defined as the likelihood that, as the network is disturbed, a given community detection algorithm finds partitions different from the ones detected in random graphs. They created the *Robustness Modularity measure* (RM), which consists in perturbing the structure of the graph by rewiring a random edge with a probability p that ranges between 0 (original graph) and 1(Erdős–Rényi model). For each value of p they calculate the *Trivial Partition Ratio*, representing the probability that there is no community structure for a given p. The higher this ratio, the more random the graph structure is, so networks with a high RM only break at high p values.

External evaluation

It is often relevant to compare and evaluate community partitions generated by different algorithms. Checking the performance of a partition may involve defining a formula to decide how similar a partition given by an algorithm is to the desired optimal partition. There are several different ways of defining these measures, so we will now look at some of the most popular ones.

The *Normalized Mutual Information* (NMI) measure [60] is probably the most popular one to evaluate how similar two clusterings are. NMI is a normalization of the Mutual Information score (MI) based on information theory, which quantifies the "amount of information" observed by one of the variables by the other random variable. The NMI values vary between 0, where there is no mutual information between the two partitions, and 1, where there is a perfect correlation. The opposite measure of NMI would be *Variation of Information* (VI) [61], which returns how different two cluster partitions are by measuring the amount of information lost and gained in changing from one partition to another. Lancichinetti *et al.* also extended this measure in the same paper to cope with overlapping communities, with the *Overlapping Normalized Mutual Information* function. Another solution similar to NMI was proposed in 2010, the *Adjusted Mutual Information* (AMI) [62]. This is an adjustment of Mutual Information score that considers chance. It accounts for the fact that the Mutual Information is generally higher for two partitions with a larger number of clusters, regardless of whether there is actually more information shared.

As alternatives for the NMI strategy of comparing partitions, we have the *Adjusted Rand Index* (ARI) [63], *Normalized F1* [64] and *Omega* [65] measures. ARI is based on the original Rand Index (RI) score [66], but is also adjusted to chance. Rand Index computes a similarity measure by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the two partitions. The higher this value, the more identical the two partitions are. It is then adjusted for chance into the ARI score in the following manner:

$$ARI = (RI - Expected\_RI)/(max(RI) - Expected\_RI) \tag{9}$$

The Omega index metric was recently proposed to compare non-disjoint clustering solutions. It was built on both the Rand Index and ARI by accounting for

disjoint solutions and correcting for chance agreement and working on overlapping community detection algorithms.

Benchmark networks
External evaluation scores can be used to compare alternative clusterings of the same network, but also to assess to what extent an identified node clustering matches a known ground truth partition. We will now present three models that represent benchmark networks with annotated communities in order to compare and evaluate the different community discovery solutions.

The first model designed to generate graphs with community structure was the *Girvan-Newman benchmark* network [6], which produces approximately the form of small interconnected Erdős–Rényi networks. It generates a graph with 128 nodes divided into 4 equally sized communities. Each node is connected to the nodes within the same community with a given probability p_{int}, and is connected to nodes from other communities with a probability p_{ext}. This allows the creation of a control parameter for this model, where k^{ext} is the degree of links connecting to nodes exterior to its community, and k^{int} is the degree of nodes connecting with nodes from the same community:

$$\mu = \frac{k^{ext}}{k^{ext} + k^{int}} \tag{10}$$

Although very well known in the scientific community, this model has limitations, mainly related to the fact that the communities are small, few and even-sized, which is not realistic. To solve this, a different approach appeared with Lancichinetti *et al.* introducing the *LFR benchmark* [69]. The main difference is that this model generates networks that follow a power-law distribution for the node degrees and the community sizes, thus solving some of the problems we had with the GN benchmark. It also allows controlling several parameters for the generated graph, such as the number of nodes, desired average degree, exponents for the degree and community size distributions and the mixing coefficient μ. The latter represents the desired average proportion of links between a node and nodes located outside its community. Consequently, this proportion is $1 - \mu$. A node of degree k has therefore an external degree of $k^{ext} = \mu k$ and an internal degree of $k^{int} = (1 - \mu)k$. Finally, another different strategy to generate networks with ground truth would be the *stochastic block model* (SBM) [70]. This is a generative and probabilistic model for random graphs that partitions the nodes in blocks of arbitrary sizes, and places edges between pairs of nodes independently, with a probability that depends on the blocks.

3.2 Multilayer

The study of multilayer networks has become extremely popular in recent years since most real and engineered systems include multiple subsystems and layers of connectivity. Analyzing multilayer networks is of great importance because many interesting patterns cannot be obtained by analyzing single-layer networks.

In this section, we will talk about community detection algorithms that were created or extended for multilayer networks and work that has been done in some areas.

The paper written by Kivela *et al.* [13] tries to summarize and generalize the huge amount of information that has been exploding recently about this subject of network science. One notion that is very similar to multilayers is that of multiplex networks. Multiplex networks are graphs where each link is categorized by the coexistence of several types of interactions among the constituents of a complex system. The main difference between the two terms is that multiplex layers require the nodes in each layer to be aligned.

Biological systems, from a cell to the human brain, are intrinsically complex. Multilayer networks, described by an intricate network of relationships across multiple scales are most widely employed in representing such systems. Protein-protein interaction (PPI) networks contain the communications among the protein groups that communicate with each other closely, which can be used to predict the complexity of the function of normal proteins [71]. Zitnik and Leskovec [72] created the *OhmNet algorithm*, which aimed at studying proteins in different tissues to understand their features, and applied it to a PPI multilayer network of human tissues. Zhao *et al.* [73] and Liang *et al.* [74] also worked on yeast PPI multilayer networks and successfully tried to predict protein functions from the modules retrieved from a community detection algorithm. Chen *et al.* [75] proposed an MLPCD algorithm and reconstructed the weighted protein-protein interaction (WPPI) network by combining PPI network and related Gene Expression Data. They utilized an improved Louvain algorithm to achieve the goal of detecting protein function modules.

Regarding human brain networks, Cantini *et al.* [76] proposed a multi-network strategy to integrate different layers of genomic information and use them in a coordinated way to identify cancer genes. The multi-networks focus, combine transcription factor co-targeting, microRNA co-targeting, PPIs, and gene co-expression networks. They were able to extract information on the functional roles of different cancer-driving genes. Stam *et al.* [77] investigated the representation of brain networks with minimum spanning trees (MST), a unique acyclic subgraph that connects all nodes and maximizes a property of interest such as synchronization between brain areas. Puxeddu *et al.* [78] performed a comprehensive analysis of three algorithms at the state-of-the-art for multilayer community detection (*genLouvain* [86], *DynMoga* and *FacetNet*) for application to EEG-Based Brain Networks and compared them with an approach based on the application of a single-layer clustering algorithm to each slice of the multilayer network. In 2021, Presigny and Fallani [79] released a paper where they presented recent advances for the characterization of the multiscale brain organization in terms of structure-function, oscillation frequencies and temporal evolution. In particular, they identified multimodal network-based biomarkers of brain pathologies, such as Alzheimer's disease.

New challenges arise for community detection in multilayer graphs, in comparison to single graph clustering. Intuitively, every single layer has a piece of

meaningful information from its own perspective, however, one can expect improved community detection results through the merging of each layer's information. Thus, it is important to understand how to combine the multiple aspects of information to better understand nodes and their associations. In addition, since we are confronted with managing multiple layers, scalability remains a significant challenge because of the larger resulting search spaces. Huang *et al.* [80] recently released a survey where they provided a comprehensive understanding of community detection methods in multilayer networks, by comparing and analysing some of them. Generally speaking, the strategies are mainly classified into three categories: flattening methods, aggregation methods and direct methods.

Flattening methods
Flattening methods collapse the information of the layers into one individual layer and then apply the traditional single-layer detection algorithms. This strategy is very common in multiplex networks.

The first community detection strategy we present in this section was proposed by Berlingerio *et al.* [81] in 2011. It is a flattening method that can be applied to multi-dimensional networks, where many connections can reside between two nodes to reflect different kinds of relationships. They used a mapping function to convert the original multi-dimensional network into a mono-dimensional one, applied traditional detection algorithms in the latter and finally assessed the quality of the result by introducing a new measure called redundancy, ρ, which captures the phenomenon for which a set of nodes that constitute a community in a dimension tend to constitute a community also in other dimensions. Pan *et al.* [82] proposed more recently another flattening based method, this time on multiplex networks, that constructs a consensus network from multiple networks and then applies existing communities detection algorithms on this graph. An aggregated/consensus graph is defined by the sum of all edge weight matrices of the layers, followed by the removal of low valued entries. The time complexity of this method is $O((n + m)L)$, where n is the total number of nodes, m is the number of edges and L is the number of layers. The major limitation in these methods is when the layers are diverse, since it is only possible to merge them when their structure is similar.

Aggregation methods
Aggregation methods discover communities in each individual layer and then merge those clusters by a certain aggregation mechanism, which could be useful for removing redundant information.

Tang *et al.* introduced one of the first aggregation based methods, the *Principal Modularity Maximization* (PMM) algorithm [83], which aimed at detecting communities in multi-dimensional networks by maximizing modularity it two phases. In the first, they extract structural features from each dimension of the network via modularity maximization, and in the second they apply *PCA* on the concatenated data to select the top eigenvectors, and finally perform k-means

to find out the discrete community structure. This method has a time complexity of $O(n^3L)$. In 2017, Tagarelli *et al.* proposed the *EMCD algorithm* [84], novel modularity and ensemble-based approach to multiplex community detection. This method finds consensus community structures and also preserves the multilayer topology information and optimizes the edge connectivity in the consensus via modularity analysis. It can also be faster than the previous solution, having a time complexity of $O(I(m + LC))$, where I is the number of iterations and C is the number of clusters. A different strategy was introduced by Huang *et al.* with the creation of the *M-Motif algorithm* [85]. This approach discovers communities with good intra-layer higher-order community quality while preserving inter-layer higher-order community consistency in multilayer networks. It identifies statistically significant sets of motifs in each layer and uses them as community structure.

Direct methods

Direct methods aim to detect the community structures directly on the multilayer network by optimizing some quality function, without flattening it. This is a group of methods that has been thoroughly explored recently, with numerous strategies being proposed to accomplish this.

Kuncheva and Montana proposed in 2015 the *Locally Adaptive Random Transitions* (LART) community detection algorithm [87] for the detection of communities that are shared by either some or all the layers in multiplex networks. The algorithm is based on random walks on the multiplex, but the transition probabilities defining the random walk are tuned depending on the local topological similarity between layers at any given node to facilitate the exploration of communities across layers. Based on this random walk, a node dissimilarity measure is derived and nodes are clustered based on this distance in a hierarchical fashion. It has a complexity of $O(m^3)$.

Some single-layer community detection methods have been extended for multiplex networks. One example is *Multiplex-Infomap* developed by De Domenico *et al.* [88], a method based on compression of network flows that can identify modular flows both within and across layers in non-aggregated multilayer networks. They extended the Infomap map equation for multiplex networks, which can identify modules that independently span across any number of layers. The complexity of Multiplex-Infomap is $O(n^2)$. Pramanik *et al.* [89] defined a multilayer modularity index, Q_M, and combined it with the improved Girvan-Newman and Louvain algorithms, namely GN-Q_M and Louvain-Q_M, respectively.

The final direct methods we present are the *Multi-Layer Many-objective Optimization algorithm* (MLMaOP) [90] by Pizzuti and Socievole, and the *Multilayer Network Label Propagation Algorithm* (MNLPA) [91] by Alimadadi *et al.*. The first authors formulated the community detection problem in multilayer networks as a many-objective optimization problem where a given objective is simultaneously optimized on all the network layers. MNLPA is an extension of the traditional Label Propagation and works on weighted and directed multi-

layer networks. It is also one of the most efficient algorithms that we presented ($O(nk)$, where k is the average degree of the nodes).

Finally, the recent work done by De Bacco *et al.* [92] proposed a generative model for multilayer networks that extends and generalizes the stochastic block model. It allows the communities to overlap and can be applied to networks with directed, undirected, or weighted links. In addition, the methods in this paper incorporate a framework for link prediction, which they use to measure how much the knowledge of one layer, or a set of layers, improves the accuracy of link prediction in another layer.

3.3 Motifs

Network motifs are recurrent and statistically significant subgraphs or patterns of a larger graph. They are of notable importance largely because they may reflect functional properties in networks. Over recent years some studies have been done on this matter.

This concept first appeared in 2002 when Milo *et al.* [15] defined motifs as patterns of interconnections occurring in complex networks in numbers that are significantly higher than those in similar randomized networks. They used a method based on the Z-score measure to detect motifs and better understand structural information in networks from several areas, such as biochemistry, neurobiology, ecology, and engineering. Since then, network motifs have been applied and studied in many fields, one of them being biology.

Albert and Albert [93] detected motifs in the *S. cerevisiae* protein-protein interaction data available at the time, to identify and validate existing interactions. Shen-Orr *et al.* [94] applied algorithms for the detection of network motifs on one of the best-characterized transcriptional regulation networks, *Escherichia coli*. They find that a big part of the network is composed of repeated appearances of three highly significant motifs, and that each one of them has a specific function in determining gene expression, such as generating temporal expression programs and governing the responses to fluctuating external signals. Choobdar *et al.* [95] proposed a mining method capable of extracting information from weighted gene co-expression networks, related to different types of cancer and healthy datasets. They studied the impact of network motifs in these networks by using the *Kolmogorov-Smirnov test* to calculate the significance score of the subgraph, and came to the conclusion that their method was capable of distinguishing different types of networks and that the discovered weighted motifs are more biologically relevant when compared to the discovered traditional binary motifs. More recently, Monteiro *et al.* [16] assessed the motif profile in the *S. cerevisiae* transcriptional network, extracted from YEASTRACT+, by applying 13 triad motifs. They concluded that these motif profiles vary as the network is enriched with novel regulatory associations, and that they also depend on the environmental conditions in which the regulatory associations take place.

Various solutions have been proposed for the challenging problem of network motif discovery. Most of them essentially consist of two main steps: first, calculating the number of occurrences of a subgraph, and then evaluating its

significance. Ribeiro *et al.* [96] surveyed three of the most popular strategies for motifs discovery: *MFinder* [97], *FanMod* [98] and *Grochow-Kellis* [99].

MFinder allows the estimation of subgraph concentrations and detection of network motifs at a runtime that is independent of the network size [97]. This algorithm is based on a random sampling of subgraphs and can be utilized for motif discovery in directed or undirected networks. *FanMod* is a tool for fast network motif detection that relies on recently developed algorithms to improve the efficiency of network motif detection over existing tools, which facilitates the detection of larger motifs in bigger networks [98]. It is one of the most useful algorithms, as it supports visual options and also is efficient with respect to time. However, it has a limitation on motif size as it does not allow searching for motifs of size 9 or higher because of the way the tool is implemented. *Grochow and Kellis* proposed a motif-centric approach, which means that the frequency of a given subgraph, called the query graph, is exhaustively determined by searching for all possible mappings from the query graph into the larger network [99]. They applied this algorithm to the PPI network and transcription regulatory network of *S. cerevisiae*, and discovered several large network motifs, which were previously inaccessible to existing methods, including a 29-node cluster of 15-node motifs corresponding to the key transcription machinery of *S. cerevisiae*. A year after releasing the survey, Ribeiro and Silva introduced *G-Tries* [100], the fastest algorithm of the ones that are here presented. It consists of a data structure that stores collections of subgraphs, each called a *g-trie*. This data structure is a *multiway tree* that can store subgraphs according to their structures and find occurrences of each of these subgraphs in a larger graph.

4 Solution Proposal

The goal of this thesis is to study the principles behind the topological structure of regulatory networks of 10 closely-related yeast species retrieved from YEASTRACT+, characterize the networks regarding their biological functions and understand how the structure is conserved across different species. Thus in this section, we will explain how this will be accomplished with three main phases.

In an initial phase of the development of this thesis, the goal is to analyze, evaluate and compare several single-layer community detection algorithms and the impact of their parameters on synthetic benchmark networks and, if possible, on real networks with ground-truth communities [101] that have a similar structure to the yeast networks. The goal is to understand which of the algorithms have better performance and are more suited to be applied in our networks. After this, we will apply the smaller chosen set of methods on the yeast networks and analyze the quality of the resulting structures. Then, we will assess the quality of the retrieved modules from a biological point of view by associating the retrieved modules with the annotated species biological functionalities.

In the next phase, we will use a multilayer approach to detect structures and functions across different species, where each one is represented by a different network layer. The aim is to analyze the similarities between the closely-related

species, and also to verify the possibility of conservation of subnetworks across different species by replicating community structure from larger networks to smaller ones, with the objective of predicting/completing information about the biological functionality of yeast species with less available data.

Finally, we will apply algorithms that aim at discovering motifs with significant representation in the different identified modules on the yeast networks as a complementary approach to traditional community methods. We will analyze their importance to the networks and compare the obtained results with the ones obtained from previous approaches.

4.1 Evaluation methodology

In this section, we will explain in more detail how the objectives will be accomplished and evaluated.

In the first phase, we will apply the majority of the methods presented in Table 2 on the benchmark and real ground-truth networks, and eliminate the ones that either have a very poor performance, or can not be applied to our networks for some reason. For instance, the methods based on bipartite graphs will not be considered since in our networks a node can be simultaneously a transcription factor (origin node) and a target gene (destination node), which goes against the principles of bipartite networks.

The evaluation of the modules retrieved by the algorithms on the yeast networks will be mainly done by using several internal evaluation measures, for example the different significance and the modularity measures. For the comparison of the communities retrieved from the ground-truth networks, we will use some external evaluation metrics such as Normalized Mutual Information, Adjusted Rand Index and others. The main benchmark network we will be focusing on is the LFR benchmark, since it is an improved version of the Girvan-Newman benchmark and has several parameters that can be tuned to better represent the desired structure of the network. Alternatively, the SBM benchmark can be used to evaluate stochastic methods.

Most of the algorithms and evaluation methods are already implemented and available to use in the *Community Discovery Library CDlib* [102], a Python software package that allows to extract, compare and evaluate communities from complex networks. After the extraction of the modules, in order to relate them with the annotated species biological functions and to better characterize each module, we will use YEASTRACT+ available data and the *Gene Ontology resource* [103], the most comprehensive resource about the functions of genes and also used to support modern biological research.

As for the multilayer phase, the networks from each yeast species will be represented as layers from the network. The associations between species are available through a mapping between their nodes. We will be focusing on aggregation methods and direct methods for community detection that work on network layers that have different structures. Flattening methods would not be possible to use because they require that the layers have their nodes aligned, which does not happen in our yeast networks. The evaluation metrics used in

this phase depend on the chosen strategy to tackle this problem. For aggregation methods the community detection is applied in the initial individual layers, so the already mentioned single-layer evaluation metrics can be applied. For the evaluation of direct methods there are several extensions of single-layer metrics, and novel methodologies to evaluate specific algorithms.

Regarding the network motifs discovery, we will be using the *gtrieScanner* tool [100] to find significant motifs in our networks, since this tool uses the g-trie data structure to find significant motifs in networks, and is very efficient compared to other solutions for this purpose. To evaluate the significance of the retrieved subgraphs we intend to use the normalized Z-score measure and other novelty evaluation metrics that might be proposed in papers related to this subject.

5 Work Schedule

In this section, a schedule for the time needed for each of the stages of the thesis will be presented in Fig. 6. The writing of the thesis's respective section on each stage will be done in parallel with the implementation. Therefore, the time specified for each one of the tasks comprises both implementation and writing.

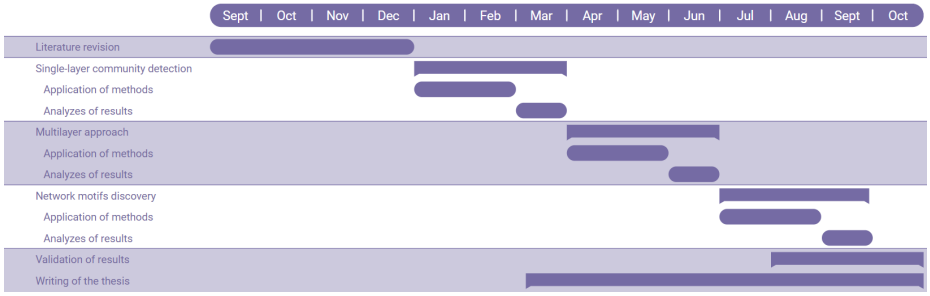

Fig. 6. Gantt chart with the proposed work schedule for the thesis implementation.

References

1. Dias, P. (2021). Functional characterization of transcriptional regulatory networks of yeast species (Unpublished master's dissertation). Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal
2. Latchman, D. S. (1997). Transcription factors: An overview. The International Journal of Biochemistry & Cell Biology, 29(12), 1305–1312. https://doi.org/10.1016/s1357-2725(97)00085-x
3. Lee, T. I., Rinaldi, N. J., Robert, F., Odom, D. T., Bar-Joseph, Z., Gerber, G. K., Hannett, N. M., Harbison, C. T., Thompson, C. M., Simon, I., Zeitlinger, J., Jennings, E. G., Murray, H. L., Gordon, D. B., Ren, B., Wyrick, J. J., Tagne,

J. B., Volkert, T. L., Fraenkel, E., . . . Young, R. A. (2002). Transcriptional Regulatory Networks in Saccharomyces cerevisiae. Science, 298(5594), 799–804. https://doi.org/10.1126/science.1075090
4. Babu, M. M., Luscombe, N. M., Aravind, L., Gerstein, M., & Teichmann, S. A. (2004). Structure and evolution of transcriptional regulatory networks. Current Opinion in Structural Biology, 14(3), 283–291. https://doi.org/10.1016/j.sbi.2004.05.004
5. Barabasi, A. (2016). Network Science (1st ed.). Cambridge University Press.
6. Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. Proceedings of the National Academy of Sciences, 99(12), 7821–7826. https://doi.org/10.1073/pnas.122653799
7. Monteiro, P. T., Oliveira, J., Pais, P., Antunes, M., Palma, M., Cavalheiro, M., ... & Teixeira, M. C. (2020). YEASTRACT+: a portal for cross-species comparative genomics of transcription regulation in yeasts. Nucleic acids research, 48(D1), D642-D649.
8. Erdos, P., & Rényi, A. (1960). On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci, 5(1), 17-60.
9. Travers, J., & Milgram, S. (1969). An Experimental Study of the Small World Problem. Sociometry, 32(4), 425. https://doi.org/10.2307/2786545
10. Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of 'small-world'networks. nature, 393(6684), 440-442.
11. Albert, R., & Barabási, A. L. (2002). Statistical mechanics of complex networks. Reviews of modern physics, 74(1), 47.
12. Newman, M. E. (2006). Modularity and community structure in networks. Proceedings of the national academy of sciences, 103(23), 8577-8582.
13. Kivela, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y., & Porter, M. A. (2014). Multilayer networks. Journal of Complex Networks, 2(3), 203–271. https://doi.org/10.1093/comnet/cnu016
14. Battiston, F., Nicosia, V., & Latora, V. (2017). The new challenges of multiplex networks: Measures and models. The European Physical Journal Special Topics, 226(3), 401–416. https://doi.org/10.1140/epjst/e2016-60274-8
15. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., & Alon, U. (2002). Network motifs: simple building blocks of complex networks. Science, 298(5594), 824-827.
16. Monteiro, P. T., Pedreira, T., Galocha, M., Teixeira, M. C., & Chaouiya, C. (2020). Assessing regulatory features of the current transcriptional network of Saccharomyces cerevisiae. Scientific Reports, 10(1). https://doi.org/10.1038/s41598-020-74043-7
17. Fortunato, S. (2010). Community detection in graphs. Physics Reports, 486(3–5), 75–174. https://doi.org/10.1016/j.physrep.2009.11.002
18. Lancichinetti, A., & Fortunato, S. (2009). Community detection algorithms: A comparative analysis. Physical Review E, 80(5). https://doi.org/10.1103/physreve.80.056117
19. Xie, J., Kelley, S., & Szymanski, B. K. (2013). Overlapping community detection in networks. ACM Computing Surveys, 45(4), 1–35. https://doi.org/10.1145/2501654.2501657
20. Newman, M. E. J. (2004). Fast algorithm for detecting community structure in networks. Physical Review E, 69(6). https://doi.org/10.1103/physreve.69.066133
21. Clauset, A., Newman, M. E. J., & Moore, C. (2004). Finding community structure in very large networks. Physical Review E, 70(6). https://doi.org/10.1103/physreve.70.066111

22. Guimerà, R., & Nunes Amaral, L. A. (2005). Functional cartography of complex metabolic networks. Nature, 433(7028), 895–900. https://doi.org/10.1038/nature03288
23. Pizzuti, C. (2008a). GA-Net: A Genetic Algorithm for Community Detection in Social Networks. Parallel Problem Solving from Nature – PPSN X, 1081–1090. https://doi.org/10.1007/978-3-540-87700-4_107
24. Li, S., Chen, Y., Du, H., & Feldman, M. W. (2009). A genetic algorithm with local search strategy for improved detection of community structure. Complexity, NA. https://doi.org/10.1002/cplx.20300
25. Blondel, V. D., Guillaume, J. L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, 2008(10), P10008. https://doi.org/10.1088/1742-5468/2008/10/p10008
26. Traag, V. A., Waltman, L., & van Eck, N. J. (2019). From Louvain to Leiden: guaranteeing well-connected communities. Scientific Reports, 9(1). https://doi.org/10.1038/s41598-019-41695-z
27. Zhang, W., Wang, X., Zhao, D., & Tang, X. (2012). Graph Degree Linkage: Agglomerative Clustering on a Directed Graph. Computer Vision – ECCV 2012, 428–441. https://doi.org/10.1007/978-3-642-33718-5_31
28. Bonald, T. (2018, June 5). Hierarchical Graph Clustering using Node Pair Sampling. ArXiv.Org. https://arxiv.org/abs/1806.01664
29. Li, P. Z., Huang, L., Wang, C. D., & Lai, J. H. (2019). EdMot. Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. https://doi.org/10.1145/3292500.3330882
30. Newman, M. E. J., & Girvan, M. (2004). Finding and evaluating community structure in networks. Physical Review E, 69(2). https://doi.org/10.1103/physreve.69.026113
31. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., & Parisi, D. (2004). Defining and identifying communities in networks. Proceedings of the National Academy of Sciences, 101(9), 2658–2663. https://doi.org/10.1073/pnas.0400054101
32. Donetti, L., & Muñoz, M. A. (2004). Detecting network communities: a new systematic and efficient algorithm. Journal of Statistical Mechanics: Theory and Experiment, 2004(10), P10012. https://doi.org/10.1088/1742-5468/2004/10/p10012.a
33. Newman, M. E. J. (2006). Finding community structure in networks using the eigenvectors of matrices. Physical Review E, 74(3). https://doi.org/10.1103/physreve.74.036104
34. Ronhovde, P., & Nussinov, Z. (2009). Multiresolution community detection for megascale networks by information-based replica correlations. Physical Review E, 80(1). https://doi.org/10.1103/physreve.80.016109
35. Adamcsek, B., Palla, G., Farkas, I. J., Derényi, I., & Vicsek, T. (2006). CFinder: locating cliques and overlapping modules in biological networks. Bioinformatics, 22(8), 1021–1023. https://doi.org/10.1093/bioinformatics/btl039
36. Ahn, Y. Y., Bagrow, J. P., & Lehmann, S. (2010). Link communities reveal multiscale complexity in networks. Nature, 466(7307), 761–764. https://doi.org/10.1038/nature09182
37. Gregory, S. (2007). An Algorithm to Find Overlapping Community Structure in Networks. Knowledge Discovery in Databases: PKDD 2007, 91–102. https://doi.org/10.1007/978-3-540-74976-9_12
38. Gregory, S. (2008). A Fast Algorithm to Find Overlapping Communities in Networks. Machine Learning and Knowledge Discovery in Databases, 408–423. https://doi.org/10.1007/978-3-540-87479-9_45

39. Hollocou, A., Lelarge, M., & Bonald, T. (2016). Improving PageRank for Local Community Detection. ArXiv, abs/1610.08722.
40. Anchuri, P., & Magdon-Ismail, M. (2012). Communities and Balance in Signed Networks: A Spectral Approach. 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. https://doi.org/10.1109/asonam.2012.48
41. Esmailian, P., & Jalili, M. (2015). Community Detection in Signed Networks: the Role of Negative ties in Different Scales. Scientific Reports, 5(1). https://doi.org/10.1038/srep14339
42. Van Dongen, S. M. (2000). Graph clustering by flow simulation (Doctoral dissertation)
43. Pons, P., & Latapy, M. (2006). Computing Communities in Large Networks Using Random Walks. Journal of Graph Algorithms and Applications, 10(2), 191–218. https://doi.org/10.7155/jgaa.00124
44. Raghavan, U. N., Albert, R., & Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. Physical Review E, 76(3). https://doi.org/10.1103/physreve.76.036106
45. Newman, M. E. J., & Leicht, E. A. (2007). Mixture models and exploratory analysis in networks. Proceedings of the National Academy of Sciences, 104(23), 9564–9569. https://doi.org/10.1073/pnas.0610537104
46. Rosvall, M., & Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. Proceedings of the National Academy of Sciences, 105(4), 1118–1123. https://doi.org/10.1073/pnas.0706851105
47. Traag, V. A., Aldecoa, R., & Delvenne, J. C. (2015). Detecting communities using asymptotical surprise. Physical Review E, 92(2). https://doi.org/10.1103/physreve.92.022816
48. Ni, C. C., Lin, Y. Y., Luo, F., & Gao, J. (2019). Community Detection on Networks with Ricci Flow. Scientific Reports, 9(1). https://doi.org/10.1038/s41598-019-46380-9
49. Platig, J., Castaldi, P. J., DeMeo, D., & Quackenbush, J. (2016). Bipartite Community Structure of eQTLs. PLOS Computational Biology, 12(9), e1005033. https://doi.org/10.1371/journal.pcbi.1005033
50. Moore, E. F. (1959). The shortest path through a maze. In Proc. Int. Symp. Switching Theory, 1959 (pp. 285-292).
51. Wu, F. Y. (1982). The Potts model. Reviews of Modern Physics, 54(1), 235–268. https://doi.org/10.1103/revmodphys.54.235
52. Palla, G., Derényi, I., Farkas, I., & Vicsek, T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. Nature, 435(7043), 814–818. https://doi.org/10.1038/nature03607
53. Rosvall, M., Axelsson, D., & Bergstrom, C. T. (2009). The map equation. The European Physical Journal Special Topics, 178(1), 13–23. https://doi.org/10.1140/epjst/e2010-01179-1
54. Yang, J., & Leskovec, J. (2013). Defining and evaluating network communities based on ground-truth. Knowledge and Information Systems, 42(1), 181–213. https://doi.org/10.1007/s10115-013-0693-z
55. Jianbo Shi, & Malik, J. (2000). Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8), 888–905. https://doi.org/10.1109/34.868688
56. Lancichinetti, A., Radicchi, F., & Ramasco, J. J. (2010). Statistical significance of communities in networks. Physical Review E, 81(4). https://doi.org/10.1103/physreve.81.046110

57. Karrer, B., Levina, E., & Newman, M. E. J. (2008). Robustness of community structure in networks. Physical Review E, 77(4). https://doi.org/10.1103/physreve.77.046119
58. Nicosia, V., Mangioni, G., Carchiolo, V., & Malgeri, M. (2009). Extending the definition of modularity to directed graphs with overlapping communities. Journal of Statistical Mechanics: Theory and Experiment, 2009(03), P03024. https://doi.org/10.1088/1742-5468/2009/03/p03024
59. Miyauchi, A., & Kawase, Y. (2016). Z-Score-Based Modularity for Community Detection in Networks. PLOS ONE, 11(1), e0147805. https://doi.org/10.1371/journal.pone.0147805
60. Lancichinetti, A., Fortunato, S., & Kertész, J. (2009). Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics, 11(3), 033015. https://doi.org/10.1088/1367-2630/11/3/033015
61. Meilă, M. (2007). Comparing clusterings—an information based distance. Journal of Multivariate Analysis, 98(5), 873–895. https://doi.org/10.1016/j.jmva.2006.11.013
62. Vinh, N. X., Epps, J., & Bailey, J. (2010). Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. The Journal of Machine Learning Research, 11, 2837-2854.
63. Hubert, L., & Arabie, P. (1985). Comparing partitions. Journal of classification, 2(1), 193-218.
64. Rossetti, G., Pappalardo, L., & Rinzivillo, S. (2016). A novel approach to evaluate community detection algorithms on ground truth. In Complex networks VII (pp. 133-144). Springer, Cham.
65. Murray, G., Carenini, G., & Ng, R. (2012, June). Using the omega index for evaluating abstractive community detection. In Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization (pp. 10-18).
66. Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association, 66(336), 846-850.
67. Newman, M. E. J. (2006b). Modularity and community structure in networks. Proceedings of the National Academy of Sciences, 103(23), 8577–8582. https://doi.org/10.1073/pnas.0601602103
68. Silva, F. N., Albeshri, A., Thayananthan, V., Alhalabi, W., & Fortunato, S. (2021). Robustness modularity in complex networks. arXiv preprint arXiv:2110.02297.
69. Lancichinetti, A., Fortunato, S., & Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. Physical review E, 78(4), 046110.
70. Holland, P. W., Laskey, K. B., & Leinhardt, S. (1983). Stochastic blockmodels: First steps. Social networks, 5(2), 109-137.
71. Srihari, S., Yong, C. H., & Wong, L. (2017). Computational prediction of protein complexes from protein interaction networks. Morgan & Claypool.
72. Zitnik, M., & Leskovec, J. (2017). Predicting multicellular function through multi-layer tissue networks. Bioinformatics, 33(14), i190–i198. https://doi.org/10.1093/bioinformatics/btx252
73. Zhao, B., Hu, S., Li, X., Zhang, F., Tian, Q., & Ni, W. (2016). An efficient method for protein function annotation based on multilayer protein networks. Human genomics, 10(1), 1-15.
74. Liang, L., Chen, V., Zhu, K., Fan, X., Lu, X., & Lu, S. (2019). Integrating data and knowledge to identify functional modules of genes: a multilayer approach. BMC bioinformatics, 20(1), 1-15.

75. Chen, J., Li, K., Bilal, K., Metwally, A. A., Li, K., & Yu, P. (2018). Parallel protein community detection in large-scale PPI networks based on multi-source learning. IEEE/ACM transactions on computational biology and bioinformatics.
76. Cantini, L., Medico, E., Fortunato, S., & Caselle, M. (2015). Detection of gene communities in multi-networks reveals cancer drivers. Scientific Reports, 5(1). https://doi.org/10.1038/srep17386
77. Stam, C., Tewarie, P., van Dellen, E., van Straaten, E., Hillebrand, A., & van Mieghem, P. (2014). The trees and the forest: Characterization of complex brain networks with minimum spanning trees. International Journal of Psychophysiology, 92(3), 129–138. https://doi.org/10.1016/j.ijpsycho.2014.04.001
78. Puxeddu, M. G., Petti, M., & Astolfi, L. (2021). A Comprehensive Analysis of Multilayer Community Detection Algorithms for Application to EEG-Based Brain Networks. Frontiers in Systems Neuroscience, 15. https://doi.org/10.3389/fnsys.2021.624183
79. Presigny, C., & Fallani, F. D. V. (2021). Multiscale modeling of brain network organization. arXiv preprint arXiv:2111.13473.
80. Huang, X., Chen, D., Ren, T., & Wang, D. (2020). A survey of community detection methods in multilayer networks. Data Mining and Knowledge Discovery, 35(1), 1–45. https://doi.org/10.1007/s10618-020-00716-6
81. Berlingerio, M., Coscia, M., & Giannotti, F. (2011). Finding and characterizing communities in multidimensional networks. In 2011 international conference on advances in social networks analysis and mining (pp. 490-494). IEEE.
82. Pan, Z., Hu, G., & Li, D. (2018). Detecting communities from multilayer networks: an aggregation approach. In proceedings of the international conference on intelligent science and technology (pp. 6-11).
83. Tang, L., Wang, X., & Liu, H. (2009, December). Uncovering groups via heterogeneous interaction analysis. In 2009 Ninth IEEE International Conference on Data Mining (pp. 503-512). IEEE.
84. Tagarelli, A., Amelio, A., & Gullo, F. (2017). Ensemble-based community detection in multilayer networks. Data Mining and Knowledge Discovery, 31(5), 1506-1543.
85. Huang, L., Wang, C. D., & Chao, H. Y. (2019). Higher-order multi-layer community detection. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, pp. 9945-9946).
86. Jutla, I. S., Jeub, L. G., & Mucha, P. J. (2011). A generalized Louvain method for community detection implemented in MATLAB. URL http://netwiki. amath. unc. edu/GenLouvain.
87. Kuncheva, Z., & Montana, G. (2015). Community Detection in Multiplex Networks using Locally Adaptive Random Walks. Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015. https://doi.org/10.1145/2808797.2808852
88. de Domenico, M., Lancichinetti, A., Arenas, A., & Rosvall, M. (2015). Identifying Modular Flows on Multilayer Networks Reveals Highly Overlapping Organization in Interconnected Systems. Physical Review X, 5(1). https://doi.org/10.1103/physrevx.5.011027
89. Pramanik, S., Tackx, R., Navelkar, A., Guillaume, J. L., & Mitra, B. (2017). Discovering community structure in multilayer networks. In 2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA) (pp. 611-620). IEEE.
90. Pizzuti, C., & Socievole, A. (2017). Many-objective optimization for community detection in multi-layer networks. 2017 IEEE Congress on Evolutionary Computation (CEC). https://doi.org/10.1109/cec.2017.7969341

91. Alimadadi, F., Khadangi, E., & Bagheri, A. (2019). Community detection in facebook activity networks and presenting a new multilayer label propagation algorithm for community detection. International Journal of Modern Physics B, 33(10), 1950089. https://doi.org/10.1142/s0217979219500899
92. De Bacco, C., Power, E. A., Larremore, D. B., & Moore, C. (2017). Community detection, link prediction, and layer interdependence in multilayer networks. Physical Review E, 95(4), 042317.
93. Albert, I., & Albert, R. (2004). Conserved network motifs allow protein–protein interaction prediction. Bioinformatics, 20(18), 3346-3352.
94. Shen-Orr, S. S., Milo, R., Mangan, S., & Alon, U. (2002). Network motifs in the transcriptional regulation network of Escherichia coli. Nature Genetics, 31(1), 64–68. https://doi.org/10.1038/ng881
95. Choobdar, S., Ribeiro, P., & Silva, F. (2015). Discovering weighted motifs in gene co-expression networks. Proceedings of the 30th Annual ACM Symposium on Applied Computing. https://doi.org/10.1145/2695664.2695773
96. Ribeiro, P., Silva, F., & Kaiser, M. (2009). Strategies for Network Motifs Discovery. 2009 Fifth IEEE International Conference on E-Science. https://doi.org/10.1109/e-science.2009.20
97. Kashtan, N., Itzkovitz, S., Milo, R., & Alon, U. (2004). Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. Bioinformatics, 20(11), 1746-1758.
98. Wernicke, S., & Rasche, F. (2006). FANMOD: a tool for fast network motif detection. Bioinformatics, 22(9), 1152-1153.
99. Grochow, J. A., & Kellis, M. (2007, April). Network motif discovery using subgraph enumeration and symmetry-breaking. In Annual International Conference on Research in Computational Molecular Biology (pp. 92-106). Springer, Berlin, Heidelberg.
100. Ribeiro, P., & Silva, F. (2010). G-tries: an efficient data structure for discovering network motifs. In Proceedings of the 2010 ACM symposium on applied computing (pp. 1559-1566).
101. Peel, L., Larremore, D. B., & Clauset, A. (2017). The ground truth about metadata and community detection in networks. Science advances, 3(5), e1602548.
102. Rossetti, G., Milli, L., & Cazabet, R. (2019). CDLIB: a python library to extract, compare and evaluate communities from complex networks. Applied Network Science, 4(1), 1-26.
103. Gene Ontology Consortium. (2004). The Gene Ontology (GO) database and informatics resource. Nucleic acids research, 32(suppl_1), D258-D261.