Network Science Cheatsheet

UNIVERSITÉ DE LYON

Made by
Remy Cazabet

Community Structure

Blocks and Communities: Definition

The general idea of blocks and communities is that nodes of a network can be grouped together in homogeneous sets, based on the network topology. The problem of automatically discovering those groups is one of the most studied problem of network science, but also one of the most difficult to properly define.

Partitions/Overlap

We must differentiate two types of node grouping:

1. A **Partition** of a graph is a division of its nodes such as each of them belongs to one and only one group.
2. Overlapping communities/blocks allow, on the contrary, nodes belonging to several groups. Unless specified differently, they also allow nodes to belong to no group.

Algorithms looking for partitions are much more common than those searching for overlapping groups, due to the increased complexity of the later task. Overlapping community detection is, nevertheless, an active field of research.

Community structure

The idea of having a network structured in **communities** is defined as an analogy with communities in social networks. Communities are therefore defined (informally) as groups of nodes that are strongly connected between themselves (**high internal density**) and more weakly connected to the rest of the network (**low external density**).
This definition however cannot be translated unambiguously into a mathematical formulation. The problem of **community detection**, or community discovery, is therefore complex to define.

Block structure

The general idea of the block structure is that the probability to observe an edge between two nodes is a function of the blocks they belong to. Contrary to communities, no assumption is made *a priori* about the respective values of those probabilities: **they can be high between nodes belonging to the same blocks or to different blocks**, and can **differ for each pair of block**.

Definition

C	a *community partition*, or, more generally, a set of set of nodes
c_i	community i, a set of nodes

Modularity

The most famous quality function to measure the *quality* of partitions is the **Modularity**. Introduced in[a], it is defined for a partition C and a graph G as the difference between the fraction of observed internal edges and the expected fraction of internal edges if G were rewired according to a configuration model, i.e., preserving the degrees of nodes.

$$Q = \frac{1}{2L}\sum_{uv}\left[A_{uv} - \frac{k_u k_v}{2L}\right]\delta(c_u, c_v)$$

with $\delta(c_u, c_v)$ the kronecker delta, i.e., $\delta(c_u, c_v) = 1$ if u, v belong to the same community, 0 otherwise. It can be rewritten for convenience as a sum over communities:

$$Q = \frac{1}{L}\sum_{i=1}^{|C|}(L_i - \frac{K_i^2}{4L})$$

with $L_i = L(H(c_i))$ the number of edges inside community i and $K_i = \sum_{u\in c_i} k_u$ the sum of degrees of nodes in community i.

[a]Girvan and Newman 2002.

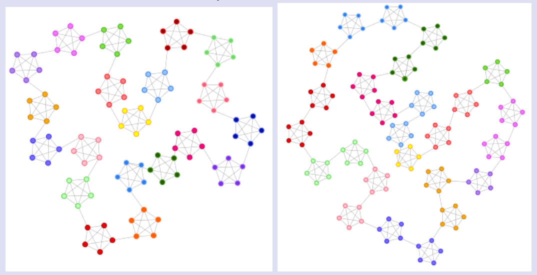
Modularity: null model

The modularity as expressed above compares the number of edges inside communities to the expected number of edges in a **null model**, i.e., a randomized version of the graph. In the original version, this null model is the **configuration model** (as easily recognized in the $\frac{k_u k_v}{2L}$ of the original formula).
Variants of the modularity have been proposed using different null models[a], for instance an ER null model, or a gravity model to take into account the effect of geographic distance[b]

[a]Jutla, Jeub, and Mucha 2011.
[b]Expert et al. 2011.

Modularity: resolution limit

It is important to remember that the Modularity is (only a) **quality function**, **not a definition** of the quality of communities. An important drawback of Modularity is known as the **resolution limit**[a]. It says that partitions of maximal modularity are biased toward a particular *scale*, i.e., for a graph of a give size (#nodes, #edges), communities smaller than a certain size cannot be found. The typical example of this limit is the clique-ring structure (set of cliques connected by a single edge), in which the expected partition is to have one community by clique, while the solution of highest modularity put several cliques in the same community, when we increase the number of cliques.



Examples of networks with 20(left) and 26(right) cliques of size 5. Colors represent communities found by a modularity maximization algorithm.

[a]Fortunato and Barthelemy 2007.

Modularity and random networks

Another well known limitation of a Modularity maximization approach is that it finds communities with high scores in random networks: since it is not *adjusted for chance*, random flucutations in a random network are mistaken for meaningful structure in the network.

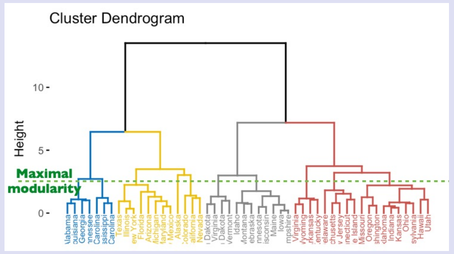
Multi-resolution Modularity

A simple solution has been proposed to the limit of resolution, consisting in adding a resolution parameter λ to *tune* the desired resolution[a], i.e., $(L_i - \frac{1}{2}K_i^2)$ becomes $(L_i - \lambda\frac{1}{2}K_i^2)$. It raises or shrinks the expected number of edges inside communities. It requires, however, to choose a proper value for λ, i.e., to choose arbitrarily a scale for communities.

[a]Reichardt and Bornholdt 2006.

Mod. maximization: Girvan Newman

Several of the most popular community detection algorithms have as objective to discover the partition of highest modularity. This is a difficult problem, and thus existing approaches are based on heuristics.
The original method by Girvan and Newman[a] first builds a dendrogram by iteratively removing the edge of highest betweenness. It is called a *divise* approach: At the top of the dendrogram, there is a single community, then 2, 3, 4 etc., until each node is in its own community. Modularity is used as a criterium to *cut* the dendrogram.



[a]Girvan and Newman 2002.

Mod. maximization: Louvain method

The Louvain method[a] is certainly the most used method for community detection. Its objective is to optimize the modularity using a greedy, agglomerative approach:
Step 1: Optimizing modularity at a hierarchical level

- Each node starts in its own community
- Repeat until convergence:
    1. **FOR** each node, compute the gain in modularity of adding it to the community of each of its neighbors
    2. choose the decision that increases the most the modularity (the best decision can be to keep the node in the same community)

Step 2: Global algorithm

- Repeat until convergence:
    1. Optimize modularity for the current hierarchical level according to Step 1
    2. Move to a higher hierarchical level by computing an **induced network**: each community becomes a node, the weight of the edge between nodes/communities i and j corresponds to the number (sum of weights) of edges between nodes of c_i and nodes of c_j.

The result of Louvain algorithm is a **hierarchy** of communities.

[a]Blondel et al. 2008.

Louvain method strengths and weaknesses

The main reason explaining the popularity of the Louvain method to this day is its **scalability**: The algorithm is very scalable in practice on real graphs, for several reasons: 1)It is a greedy approach, 2) By checking only the interest of moving to neighbor's communities, it benefits from the sparsity of networks, 3)Modularity gains of a partition change can be computed locally, using its definition as a sum of independent values for each community.
Another advantage of the Louvain method is that results at lower hierarchical levels can naturally mitigate the problem of the resolution limit, for instance on the ring clique example, Louvain find each clique in its community at the first level, and only in a second level yield the problematic partition.
However, it has also be shown in[a] that the greedy nature of the algorithm could lead to having counter-intuitive structures, such as disconnected communities. The authors of the paper proceed to introduce a variant of the algorithm called Leiden, solving this problem.

[a]Traag, Waltman, and Eck 2019.

Infomap

Infomap[a] is a method based on an objective function different from the Modularity. Its objective is to **Minimize the description length of an average random walk** in the network, i.e. maximize the **compression** of the description of such a walk. More formally, the code length to minimize for partition C is described as:

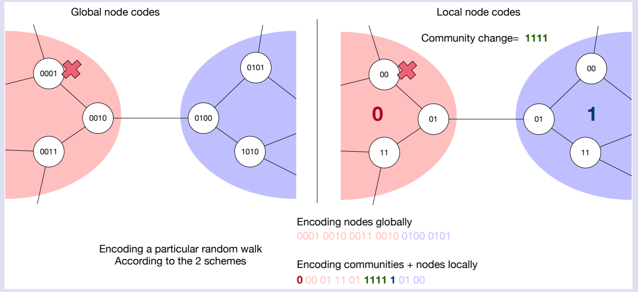
$$H(C) = qH(\curvearrowright) + \sum_{i}^{|C|} p^i H(\circlearrowright_i)$$

with q the probability for a move to be between modules, $H(\curvearrowright)$ the amount of information (bits) required to encode a move between modules, p^i the probability for a move to be inside community i and $H(\circlearrowright_i)$ the amount of information required to encode a move inside community i.
A greedy optimization algorithm, for instance one similar in nature to the one of Louvain, can then be used to minimize this description length.
Compared with Modularity, the main advantage of this approach is that it does not systematically find communities in random networks. It is known also to suffer from a form of resolution limit. Several improvements have been proposed, for instance to discover hierarchical partitions.

[a]Rosvall and Bergstrom 2008.

Infomap Algorithm intuition

Illustration of the intuition behind Infomap random walk compression. For a more accurate depiction, check the excellent illustration by the authors[a]. The real encoding length is not computed explicitly, but estimated for an infinite random walk based on information theory principles.



[a]Rosvall and Bergstrom 2008.

Stochastic Block Models (SBM)

A stochastic block model is a random graph model defined by:

b	$n \times 1$ vector such as b_i describes the index of the block of node i.
E	$k \times k$ **stochastic block matrix**, such as E_{ij} gives the number of edges between blocks i and j (or the probability to observe an edge between any pair of nodes chosen with one node in each of the two blocks).
θ	$n \times 1$ vector representing the node degrees (optional)

SBM inference

The objective of a community/block detection algorithm based on the SBM principle is thus to perform **SBM inference**, i.e., to find the parameters of the SBM that best explain the observed graph, usually in term of maximizing the likelihood. Said differently, we search –among a certain class of models– the model that has the highest probability to generate the observed graph. Note that for an observed graph, for each partition in blocks b, there is a single block matrix E that is relevant to consider, that can be found simply by counting the number of edges actually present between blocks in the graph.
More formally, the objective is:

$$b := \operatorname*{argmax}_{b} P(A|b)$$

SBM: Simple Graphs

Assuming a simple graph, the probability to observe a graph A given a partition b is computed as the product, for each pair of node, of the probability to obtain the observed situation: edge or no-edges. It uses a Bernouilli distribution.

$$p(A|b,E) = \prod_{i<j} \begin{cases} E_{b_i b_j} & \text{if } A_{ij} = 1 \\ 1 - E_{b_i b_j} & \text{if } A_{ij} = 0 \end{cases}$$

SBM: Poisson

Other assumptions on the distribution can be made. For scalability reasons, a common one is to assume a poisson distribution of edges

$$P(A|b,E) = \prod_{i<j} \frac{(E_{b_i b_j})^{A_{ij}}}{A_{ij}!} e^{-E_{b_i b_j}}$$

Assuming that A is sparse, Poisson or Bernouilli tends toward the same results for large graphs.

DC-SBM: Degree Correction

Much as the Modularity null model preserving nodes degrees, modern SBM usually integrate a degree correction

$$P(A|b,E,\theta) = \prod_{i<j} \frac{(\theta_i \theta_j E_{b_i b_j})^{A_{ij}}}{A_{ij}!} e^{-\theta_i \theta_j E_{b_i b_j}}$$

as E, the optima θ is deducted from b, a:

$$\hat{\theta}_i = \frac{k_i}{\kappa_{b_i}}$$

with κ_{b_i} the sum of degrees in i's cluster

SBM: quality function

The probability to observe a graph can be simplified[a], using log transformation and getting rid of constants, into objective functions as simple to compute as the modularity.
Poisson, with self-loops (m=nb edges, n=nb nodes)

$$\mathcal{L}(A|b) = \sum_{rs} m_{rs} \log \frac{m_{rs}}{n_r n_s}$$

Poisson, with self-loops, degree corrected (κ=sum of degrees)

$$\mathcal{L}(A|b) = \sum_{rs} m_{rs} \log \frac{m_{rs}}{\kappa_r \kappa_s}$$

[a]Karrer and Newman 2011.

SBM: number of blocks

As defined until now, the inference has a trivial solution: each node being in its own block, and $A = E$. The probability of such a model to generate the observed graph is maximal (1 for simple graphs). The solution to this problem is often to fix the number of blocks. This approach is not satisfying in the general case, when we do not know the expected number of blocks.

SBM: infer the number of blocks

More recently, approaches[a] have been proposed to infer also automatically the number of blocks. They adopt an approach from Information Theory called the Minimum Description Length (MDL), whose principle is to find the description which reduces the total cost of describing a graph, by minimizing both 1)The quantity of information needed to encode the graph, knowing that it is generated by a given model, and 2)The quantity of information needed to encode the model itself. Intuitively, a model with few blocks requires little information to be described, contrary to a model with many blocks. But a model with many blocks is more **constrained**, the graphs it generates are more *specific*, and therefore can be described at a lesser cost, knowing the model.

[a]Peixoto 2019.

SBM infering blocks: equation

More formally, we can decompose the probability of observing a graph and a model as:

$$P(A|b) = P(A|\theta,E,b)P(\theta|E,b)P(E|b)P(b)$$

with the last three probabilities being *priors*. Said differently, we can define the number of bits required to encode a model as $L = -log_2 P(\theta,E,b)$, the number of bits necessary to encode a graph knowing the model as $S = -log_2 P(A|\theta,E,b)$ and thus the total cost to minimize as $S + L$.
The objective thus becomes:

$$b := \operatorname*{argmin}_b -log_2 P(\theta,E,b) - log_2 P(A|\theta,E,b)$$

Variants of the SBM

Group inference using SBM is a very active field of research, and many variants have been proposed, including degree-corrected, nested, Overlapping, Mixed membership SBM, etc.
An introduction to the state of the art can be found for instance in[a].
A python library[b] exists to apply recent methods to observed graphs.

[a]Lee and Wilkinson 2019.
[b]https://graph-tool.skewed.de

Evaluation of Community structures

Since there isn't a unique accepted definition of what are good communities, the evaluation of the quality of a partition or set of communities is not a trivial task.
There are two main approaches:

- **Internal evaluation** consists in using *quality functions* (e.g., Modularity) to give a score for a pair partition/graph
- **External evaluation** consists in comparing a computed partition to a **ground truth** reference partition.

Internal evaluation can be used to evaluate the quality of communities found on a network of interest, while External evaluation is mostly used to asses the quality of algorithms on benchmarks.

Internal Evaluation - Global

Objective quality function themselves can be understood as different **definitions of community structures**. While some methods try directly to optimize one of those quality functions, some other methods are based on different principles (e.g., clique-based communities, consensus reaching based on game-theory, etc.). Quality functions can therefore be used a posteriori to assess the quality of a partition.
Typical examples are:

- **Modularity**
- **Description lenght**, as in Infomap or SBM
- **Surprise**[a] evaluates the departure of the observed partition from the expected distribution of nodes and links into communities given a null model.

[a]Aldecoa and Marin 2013.

Internal Evaluation - By community

Some quality functions are defined at the level of **individual communities**, instead of having one score for the whole partition. Those individual scores can however be combined to provide a global score, for instance using a weighted average. Some of the most popular are[a]:

- **Conductance**, the fraction of all stubs of nodes in the community that points outside of it
- **ODF**, Out Degree Fraction, the average for every node of its fraction of neighbors inside the community
- **Internal Transitivity**, the clustering coefficient inside the community
- **Scaled density**, the ratio of the node density to the total graph density

[a]leskovec2010empirical

External Evaluation

Partitions obtained by a given method can be compared with a ground truth. This approach is used on real networks, with a ground truth coming from metadata (e.g., classes in a network of social interactions between students), and on synthetic networks, with communities known by construction.
Although this is still discussed in the literature, it is mostly accepted that the evaluation on real networks using this approach is problematic[a], because there is no guarantee that the labels used as ground truth are indeed related to the **topological structure** of the network, which is what communities are about.
Most popular methods for partitions comparisons are:

- **NMI**, Normalized Mutual Information, and its adjusted for chance variant, **AMI**.
- **ARI**, Adjusted Rand Index

But more generally, any method for cluster comparison can be used[b]

[a]Peel, Larremore, and Clauset 2017.
[b]Dao, Bothorel, and Lenca 2020.

Overlapping communities

For many types of networks, the real organization of networks is thought to be **overlapping**, i.e., each node can belong to several communities. Think for instance of your personal social network: some of your family members might also be part of a group of friends, or some of your friends from high school might also be part of your friends from university, which are otherwise distinct groups.
Detecting overlapping clusters is considered harder than non-overlapping ones, for at least two reasons: the search space (number of possible solutions) is much larger (and even infinite), and defining what good communities are is even harder, since there isn't the natural limit for each edge to be either internal or external.
A large number of methods have nevertheless been proposed[a]. Extensions of non-overlapping quality functions have been proposed, such as the overlapping Modularity[b], or overlapping NMI[c], but they are not as widely used as their more constrained counterparts.

[a]Xie, Kelley, and Szymanski 2013.
[b]Nicosia et al. 2009.
[c]McDaid, Greene, and Hurley 2011.

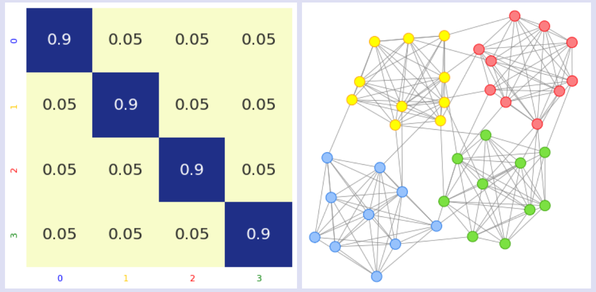
Other meso-scale structures

Beyond the usual community structure, other types of network structural organizations have been proposed and studied. Some of the most widely known are:

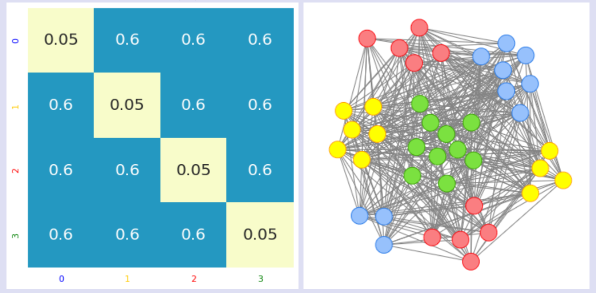
- **Link communities**, in which communities are defined as *sets of links*. Searching for (non-overlapping) partitions of edges yield a structure in which nodes naturally belong to several groups, i.e., a community can corresponds to *familial* edges, another to *professional* edges, etc. (Ahn, Bagrow, and Lehmann 2010)
- **Fuzzy communities**, in which nodes belong to (often several) communities with a certain probability or strength (Liu 2010)
- **Core-Periphery structure**, already defined when we introduced the notion of k-*cores*
- **Nestedness**, corresponding to a network with a hierarchical organization such as elements with few connections tends to be connected to a subset of the neighbors of a *parent* node. (Pawar 2014)
- **Spatial organization**, in which the probability of observing an edge between nodes depends on their distance. (Barthélemy 2011)

Meso-scale organization -1

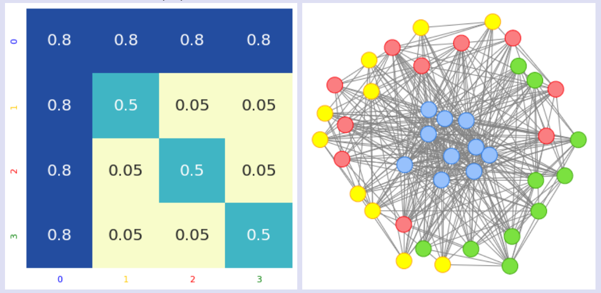
Examples of different types of organization that can be obtained using block structure



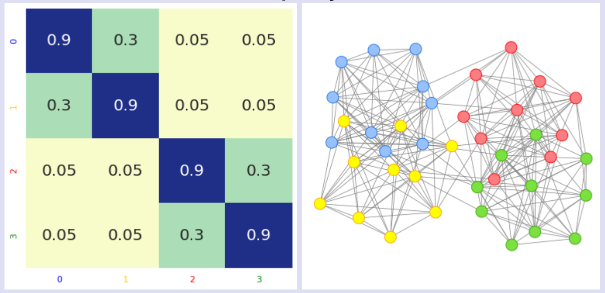
(1) Assortative Structure



(2) Disassortative Structure

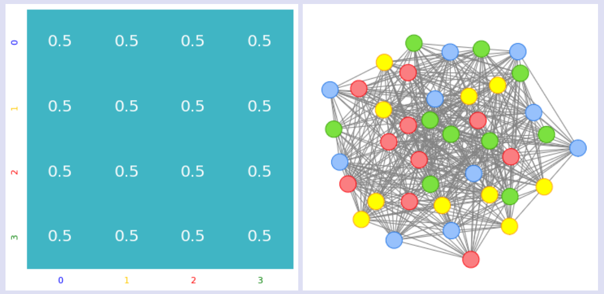


(3) Core Periphery Structure

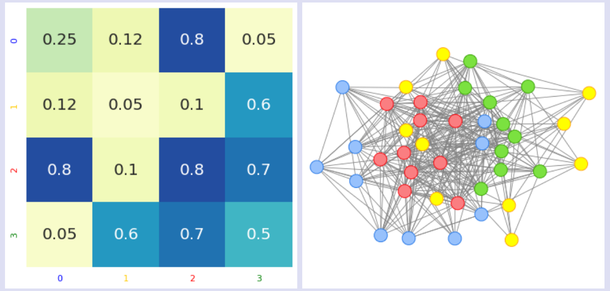


(4) Hierarchical Structure

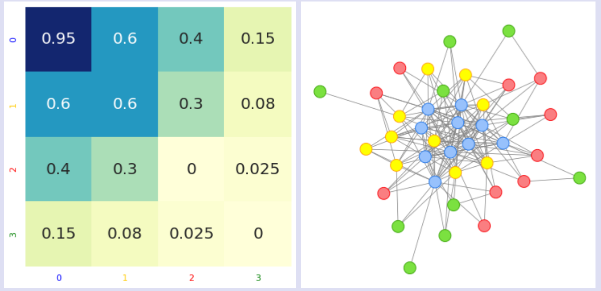
Meso-scale organization -2



(5) Uniform/ Random



(6) Mixed Structure



(7) Nested Structure

Going Further

Surveys: (Fortunato 2010) (Fortunato and Hric 2016)
On community detection approaches: (Rosvall, Delvenne, et al. 2019)
On stochastic Block Models:Funke and Becker 2019
Survey overlapping communities: (Xie, Kelley, and Szymanski 2013)
Community detection in dynamic networks (Rossetti and Cazabet 2018)
On ground Truth and community detection: (Peel, Larremore, and Clauset 2017)
Community detection in neuroscience (Betzel 2020)

References

[1] Yong-Yeol Ahn, James P Bagrow, and Sune Lehmann. "Link communities reveal multiscale complexity in networks". In: *nature* 466.7307 (2010), pp. 761–764.

[2] Rodrigo Aldecoa and Ignacio Marin. "Surprise maximization reveals the community structure of complex networks". In: *Scientific reports* 3.1 (2013), pp. 1–9.

[3] Marc Barthélemy. "Spatial networks". In: *Physics Reports* 499.1-3 (2011), pp. 1–101.

[4] Richard F Betzel. "Community detection in network neuroscience". In: *arXiv preprint arXiv:2011.06723* (2020).

[5] Vincent D Blondel et al. "Fast unfolding of communities in large networks". In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008.

[6] Vinh Loc Dao, Cécile Bothorel, and Philippe Lenca. "Community structure: A comparative evaluation of community detection methods". In: *Network Science* 8.1 (2020), pp. 1–41.

[7] Paul Expert et al. "Uncovering space-independent communities in spatial networks". In: *Proceedings of the National Academy of Sciences* 108.19 (2011), pp. 7663–7668.

[8] Santo Fortunato. "Community detection in graphs". In: *Physics reports* 486.3-5 (2010), pp. 75–174.

[9] Santo Fortunato and Marc Barthelemy. "Resolution limit in community detection". In: *Proceedings of the national academy of sciences* 104.1 (2007), pp. 36–41.

[10] Santo Fortunato and Darko Hric. "Community detection in networks: A user guide". In: *Physics reports* 659 (2016), pp. 1–44.

[11] Thorben Funke and Till Becker. "Stochastic block models: A comparison of variants and inference methods". In: *PloS one* 14.4 (2019), e0215296.

[12] Michelle Girvan and Mark EJ Newman. "Community structure in social and biological networks". In: *Proceedings of the national academy of sciences* 99.12 (2002), pp. 7821–7826.

[13] Inderjit S Jutla, Lucas GS Jeub, and Peter J Mucha. "A generalized Louvain method for community detection implemented in MATLAB". In: *URL http://netwiki. amath. unc. edu/GenLouvain* (2011).

[14] Brian Karrer and Mark EJ Newman. "Stochastic blockmodels and community structure in networks". In: *Physical review E* 83.1 (2011), p. 016107.

[15] Clement Lee and Darren J Wilkinson. "A review of stochastic block models and extensions for graph clustering". In: *Applied Network Science* 4.1 (2019), p. 122.

[16] Jian Liu. "Fuzzy modularity and fuzzy community structure in networks". In: *The European Physical Journal B* 77.4 (2010), pp. 547–557.

[17] Aaron F McDaid, Derek Greene, and Neil Hurley. "Normalized mutual information to evaluate overlapping community finding algorithms". In: *arXiv preprint arXiv:1110.2515* (2011).

[18] Vincenzo Nicosia et al. "Extending the definition of modularity to directed graphs with overlapping communities". In: *Journal of Statistical Mechanics: Theory and Experiment* 2009.03 (2009), P03024.

[19] Samraat Pawar. "Why are plant-pollinator networks nested?" In: *science* 345.6195 (2014), pp. 383–383.

[20] Leto Peel, Daniel B Larremore, and Aaron Clauset. "The ground truth about metadata and community detection in networks". In: *Science advances* 3.5 (2017), e1602548.

[21] Tiago P Peixoto. "Bayesian stochastic blockmodeling". In: *Advances in network clustering and blockmodeling* (2019), pp. 289–332.

[22] Jörg Reichardt and Stefan Bornholdt. "Statistical mechanics of community detection". In: *Physical review E* 74.1 (2006), p. 016110.

[23] Giulio Rossetti and Rémy Cazabet. "Community discovery in dynamic networks: a survey". In: *ACM Computing Surveys (CSUR)* 51.2 (2018), pp. 1–37.

[24] Martin Rosvall and Carl T Bergstrom. "Maps of random walks on complex networks reveal community structure". In: *Proceedings of the National Academy of Sciences* 105.4 (2008), pp. 1118–1123.

[25] Martin Rosvall, Jean-Charles Delvenne, et al. "Different approaches to community detection". In: *Advances in network clustering and blockmodeling* (2019), pp. 105–119.

[26] Vincent A Traag, Ludo Waltman, and Nees Jan van Eck. "From Louvain to Leiden: guaranteeing well-connected communities". In: *Scientific reports* 9.1 (2019), pp. 1–12.

[27] Jierui Xie, Stephen Kelley, and Boleslaw K Szymanski. "Overlapping community detection in networks: The state-of-the-art and comparative study". In: *Acm computing surveys (csur)* 45.4 (2013), pp. 1–35.