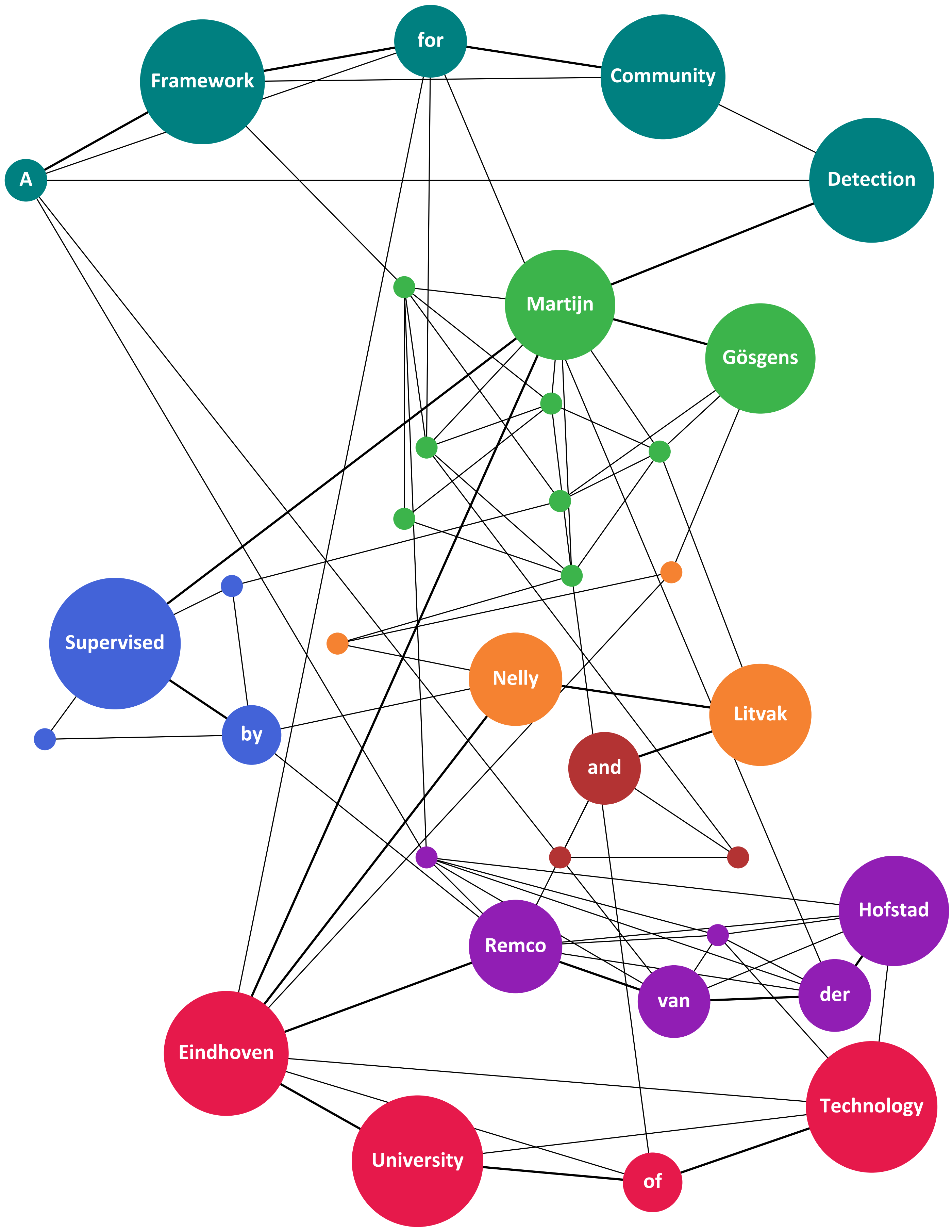
A framework for community detection

Gösgens, M.M.

Award date:
2020


for
Framework
Community
A
Detection
Martijn
Gösgens
Supervised
Nelly
Litvak
by
and
Hofstad
Remco
van
der
Eindhoven
Technology
University
of

Abstract

In real-world networks, one can often distinguish groups of nodes that are more connected to each other than to the rest of the network. In network science, these groups of nodes are called *communities.* These communities usually have meaning in the real world, e.g. fields of studies in citation networks or groups of friends in social networks.

Currently, there is no sound problem definition for *community detection.* Attempts in defining this tend to get stuck with the question "What are communities?", which does not have a single answer since it depends heavily on the context. Because of this lack of formalism, many community detection methods have been developed based on heuristics and the absence of sound ways to compare them makes it difficult for practitioners to find the most suitable method for a given application. As a result of that, the choice is often made based on the popularity of the method instead of its suitability for the application at hand.

In this thesis, we intend to make a first step towards a sound formalization of community detection. We do so by distinguishing 4 *ingredients* that are necessary for properly performing community detection: 1) a validation index to compare community structures; 2) a joint distribution of the networks and community structures; 3) a quality function that measures how well a community structure fits a network; and 4) an algorithm that is able to find a candidate by optimizing this quantity. For each of these ingredients, we discuss the available options and discuss criteria upon which a reasonable choice can be based for given a application. In doing so, we hope to pave the way for an approach to community detection that relies less on heuristic one-size-fits-all solutions and more on theoretical guarantees based on verifiable assumptions.

With regards to the validation index, we list a number of desirable properties and verify which indices satisfy them. Based on this research, we conclude that for many applications, the Correlation Coefficient is the most desirable validation index. Surprisingly, this index has rarely been used in practice.

We further introduce a class of detection methods for which we derive worst-case guarantees on the performance by lower-bounding the value of the validation index. Although this lower-bound is significantly lower than its actual performance, it does provide valuable insight into the kinds of networks for which these methods perform well. In future research, similar guarantees could be obtained for other detection methods. Given more such guarantees, the proposed framework would allow community detection methods to be tailored to the application at hand.

Acknowledgements

First of all, I would like to thank Nelly Litvak and Remco van der Hofstad for their valuable guidance and feedback during the internship and graduation. Furthermore, I would like to give special thanks to Liudmila Prokhorenkova for providing the possibility to do my internship in Moscow. I also would like to thank the Moscow Institute of Physics and Technology for their accommodation and hospitality during my internship there. Special thanks goes out to Yandex and their taxi drivers, for allowing me to work from their office and for helping me learn the bare minimum of Russian that is essential for survival. Finally, I thank my friends and family for their crucial support and distraction during the last years.

Contents

Chapter 1

What are communities?

In real-world networks, one can often distinguish groups of nodes that are more connected to each other than to the rest of the network[16]. In network science, these groups of nodes are called *communities*. These communities tend to have meaning in the real world. For example, communities in citation networks closely resemble fields of study while communities in social networks resemble groups of friends. The communities of a network are often referred to as its *community structure*. There are two kinds of community structures: overlapping community structure (a vertex can be part of multiple communities) and non-overlapping community structure (each vertex is part of exactly one community). In this thesis we will focus on the latter. We will illustrate community detection in the following simple example.

1.1 A real-world example

Let us analyze a network of airports where nodes are connected if there is a flight between them (according to flight data from 2014). The data is obtained from `openflights.org`. In general, airports within a country tend to be better connected (more available flights) than airports in different countries. There are multiple reasons for this: international flights need to comply with regulations from two countries, require extra customs checks and tend to be longer. Furthermore, for passengers, an international trip comes with extra difficulties such as different languages and currency.

We consider the network of flight routes between airports in Chile and Argentina from 2014. These countries are chosen because a) both countries have a relatively small number of airports (16 and 32) b) both countries speak the same language and do not require visa c) the long shape of Chile makes it such that flights between these countries are not significantly longer than domestic flights in these countries, so that we would expect many connections between airports from these countries.

We expect there to be two clear communities corresponding to Chile and Argentina. To find this split, we optimize the Newman-Girvan modularity, a quality function that will be discussed in Section 6.2. Informally, it finds communities by maximizing the difference between the number of edges within those communities and the expected number of edges within those communities w.r.t. a random graph model that does not have community structure. In practice, the expected number of edges is scaled by a parameter called the *resolution parameter*. Decreasing this *resolution* leads to a more coarse community structure. We set this resolution parameter to 1 (its default value) and optimize this modularity using the Louvain algorithm, a popular algorithm that will be discussed in Chapter 7. The results are shown in Figure 1.1.

We see that the detection algorithm manages to distinguish Chile (including Easter Island in the Pacific Ocean). However, Argentina is split into three communities. We see one community in the south (Patagonia) consisting of airports that are internally well connected and externally connected almost as well to Chile as to the rest of Argentina. The remainder of Argentina is split in two. The reason for this is that the capital, Buenos Aires, has two airports and these are the biggest airports of the country. The main airport is connected to all Argentinian airports outside the capital. One community is formed by the second airport and the Argentinian airports that it is connected to, while the remainder of the Argentinian airports form the last community.

As we can see, in hindsight, it is possible to explain why the outcome is different than desired, yet this would have been difficult to predict beforehand. When we decrease the resolution parameter to 0.7, we do get the desired outcome, as shown in Figure 1.2. However, we could not have known beforehand what parameter setting was needed.

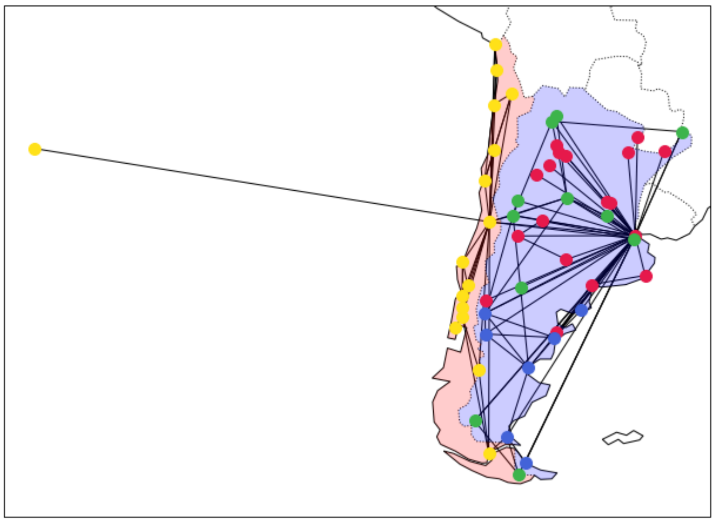


Figure 1.1: Flight routes between Chile and Argentina from 2014 are visualized on the map. Communities are found by the Louvain algorithm for maximizing Newman-Girvan modularity with resolution 1. The data was obtained from `openflights.org`. The node on the left is Eastern Island, which belongs to Chile.

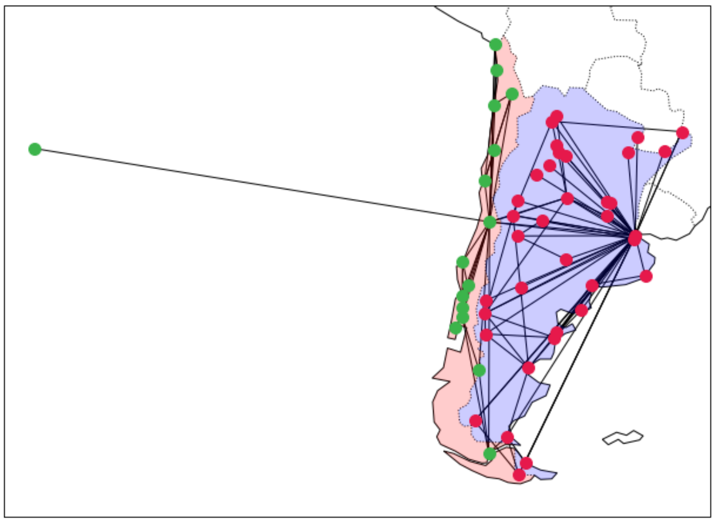


Figure 1.2: Flight routes between Chile and Argentina from 2014 are visualized on the map. Communities are found by the Louvain algorithm for maximizing Newman-Girvan modularity with resolution parameter 0.7. The data was obtained from `openflights.org`.

Currently there are no clear guarantees about how well certain algorithms will perform for a specific setting. As you can see, we are able to tweak the parameters to obtain the desired outcome, but we are only able to do so because we already know exactly what the desired outcome is. In practice, this knowledge is not available since otherwise community detection would not be needed. This makes it difficult to know which detection methods are appropriate for a given application and how parameters need to be chosen.

Furthermore, in the above example we were lucky that the network actually reveals the split between the countries. In general, this may not be the case. One could imagine that in a citation network, there might exist a paper from a certain field of study (e.g. statistics) that is cited mainly by papers from another study (e.g. sociology). In this case, any reasonable detection algorithm will wrongly classify it as a sociology paper. This shows that the relation between the network and the *true* community structure must also be sufficiently strong in order for a community detection algorithm to perform well.

This thesis attempts to make a humble start in laying down the formalisms that are required for making guarantees about which community detection methods will work well for which networks.

1.2 Community detection: an ill-defined problem

As illustrated by the example, the field of community detection is in need of some formalism. Community detection is known to be an ill-defined problem [15] and is mostly tackled by heuristic methods without much underlying theory. Because of the absence of a clear problem definition, scholars have significant freedom in choosing the way in which they compare their own community detection methods to existing ones. This lowers the bar for introducing new detection methods [13]. Because of the vast amount of available community detection methods and the lack of formalism to compare them, it is unclear to decide which method is best suited for a certain application. This decision is therefore often made based on the wrong reasons such as popularity of the algorithm or its inventor [13].

Attempts of formalizing community detection tend to get stuck with the question "What are communities?". The problem with this question is that its answer depends on the specific application [13]. This thesis aims to formalize the problem of community detection in a way that avoids this question. This will allow us to obtain theoretical guarantees for detection methods so that the choice for a detection method no longer has to be based on heuristics. In particular, we will introduce a framework that enables mathematical derivation of performance guarantees for community detection algorithms. To demonstrate this framework, we will introduce a new community detection method and prove that, under certain conditions, a lower bound for its worst-case performance is guaranteed. Even though this worst-case guarantee will be significantly worse than the actual performance of this method (or any other state-of-the-art community detection method), it will provide insight in the situations where this method performs well. We emphasize that this new detection method is not intended to replace the state-of-the-art community detection methods but that it is merely a demonstration of the feasibility and usefulness of obtaining theoretical guarantees that are helpful in understanding which detection method is suitable in a given context.

Outline. The remainder of this thesis is organized as follows: Chapter 2 will discuss preliminaries. The theoretical framework will be introduced in Chapter 3. This framework will distinguish various *ingredients* of community detection which will be discussed in depth in Chapters 4, 5, 6 and 7. After that, Chapter 8 will describe a class of community detection methods for which lower bounds on the worst-case performance can be obtained. Chapter 9 will provide numerical experiments that compare our obtained detection method to the current state-of-the-art detection method. Finally, Chapter 10 will give a conclusion and discuss directions for future research.

Chapter 2

Preliminaries

This chapter introduces the notation that will be used throughout the thesis as well as some preliminary results about power-law distributions. In Section 2.3, a few existing random graph models are introduced.

2.1 Notation

In this section, we will introduce the main notation that will be used throughout the thesis.

2.1.1 Graphs

We will consider simple undirected graphs denoted by G. Unless stated otherwise, our graphs consist of n vertices and m edges. We write $[n] = \{1, \dots, n\}$. We will represent vertices by integers. Hence, our vertex-set will be given by $[n]$. The number of vertex-pairs will be denoted by $N = \binom{n}{2}$. We say that its *density* is given by $p_G = m/N$. For $a, b \subseteq [n]$, we will denote the number of edges that go from a to b by $m^G_{a,b}$, e.g. for a community structure C and $c, c' \in C$ there are $m^G_{c,c'}$ edges going from community c to community c'. The degree of a vertex i is denoted by d^G_i. The superscripts are omitted whenever the graph is clear from the context.

2.1.2 Clusterings

Community structures will be represented by partitions of the vertices into their communities, e.g. $A = \{a_1, \dots, a_{|A|}\}$ where $a_i, a_j \subseteq [n]$ are disjoint for all $i \neq j$ and $\bigcup_{i \in [1,\dots,k]} a_i = [n]$. Often, we will use the word *clustering* instead of community structure for brevity. The ground truth clustering will be denoted by T while the candidate clustering will be denoted by C. The letters A and B will be used when the clustering has no clear role. The community that vertex i is part of in a clustering C is denoted by $C(i)$. Note that a community is represented by its vertex-set so that $i \in C(i)$.

Clusterings as graphs. By G_A we will denote the graph where vertices are connected if and only if they are in the same community in A. Hence, each connected component of G_A will be clique corresponding to a community in A. We will refer to G_A as the *caveman graph* of A (as termed by [49]).

2.1.3 Pair-counting

For graphs G_1, G_2, let us introduce binary vectors $\vec{b}_{G_1}, \vec{b}_{G_2}$ indexed by the vertex-pairs such that an entry is 1 if the corresponding vertex-pair is connected by an edge and 0 else. If we concatenate $\vec{b}_{G_1}, \vec{b}_{G_2}$ into an $N \times 2$ matrix, then each row is either 11, 10, 01 or 00. The number of times each of these rows occur in this matrix is given by the *pair-counts* $N^{G_1,G_2}_{11}, N^{G_1,G_2}_{10}, N^{G_1,G_2}_{01}, N^{G_1,G_2}_{00}$. Similarly, for clusterings T and C, we can write $\vec{b}_T = \vec{b}_{G_T}, \vec{b}_C = \vec{b}_{G_C}$. Note that this representation has some redundancy: whenever i, j and j, k form intra-cluster pairs, we know that i, k must also be an intra-cluster pair. Hence, not every binary vector of length N represents a clustering. The class of N-dimensional binary vectors is, however, isomorphic to the class of undirected graphs on n vertices. Representing both graphs and clusterings as binary vectors allows us to compare clusterings to each other by $N^{T,C}_{11} = N^{G_T,G_C}_{11}$ (etc.) while comparing graphs to clusterings by $N^{G,T}_{11} = N^{G,G_T}_{11}$ (etc.). Often, it is clear from the context which objects are being compared. In these cases the superscripts are omitted. Note that these pair-counts can also be defined as inner-products, for example: $N_{11} = \langle \vec{b_1}, \vec{b_2} \rangle, N_{01} = \langle \mathbf{1} - \vec{b_1}, \vec{b_2} \rangle$ (etc), where $\langle \cdot, \cdot \rangle$ is the standard inner product and $\mathbf{1}$ is the N-dimensional all-one vector.

For a graph G and a clustering T, we will denote by p_G, p_T the fraction of nonzero entries in $\vec{b}_G$ and $\vec{b}_T$ respectively. These will be referred to as the *densities* of the graph and clustering respectively. We will write p_{GT} as the fraction of entries that are nonzero in both $\vec{b}_G$ and $\vec{b}_T$. This will be referred to as the *intersection* between G and T. Note that

$$p_G = \frac{N_{11}^{G,T} + N_{10}^{G,T}}{N}, \quad p_T = \frac{N_{11}^{G,T} + N_{01}^{G,T}}{N}, \quad p_{GT} = \frac{N_{11}^{G,T}}{N}.$$

In a similar way, the intersection between two clusterings A and B is defined as p_{AB}. For a graph G and a ground truth clustering T, the *mixing rate* is defined as the fraction of edges that connect vertices of different communities and it is denoted by μ_{GT}, i.e. $\mu_{GT} = 1 - p_{GT}/p_G$. The subscript of this mixing rate is often omitted as the graph and clustering are usually clear from the context.

2.1.4 Community detection algorithms

Throughout the thesis, $\mathcal{A}$ will denote a community detection algorithm and its result when applied to a graph G will be denoted by $\mathcal{A}(G)$. If we want to stress that a community detection algorithm finds a candidate by optimizing a quality function Q, then we will denote the algorithm by $\mathcal{A}_Q$.

2.1.5 Convergence

For a random variable X and a sequence of random variables $(X_k)_{k \geq 1}$, we will say that X_k converges *in distribution* to X and write $X_k \overset{\mathcal{D}}{\to} X$ whenever

$$\mathbb{P}(X_k \leq x) \to \mathbb{P}(X \leq x),$$

for all x as $k \to \infty$. Furthermore, we write $X_k \overset{\mathbb{P}}{\to} X$ and say that X_k converges *in probability* to X whenever

$$\mathbb{P}(|X_k - X| > \varepsilon) \to 0,$$

for all $\varepsilon > 0$ as $k \to \infty$.

2.2 Power-law distributions

Many real-world networks have degree sequences which satisfy a power law [41]. A *power-law relation* with exponent α between $f(x)$ and x is often denoted by $f(x) \sim x^\alpha$. Here, the $\sim$ is meant to denote that the left and right hand side are "roughly proportional". In other studies [48], this is formalized in terms of *slowly varying functions*, where $f(x) \sim g(x)$ if there exists a function $L(x)$ such that $f(x) = L(x)g(x)$ and

$$\lim_{x \to \infty} \frac{L(rx)}{L(x)} = 1,$$

for all $r > 0$.

However, we will use a different notation and a different formalization:

Definition 1. *We write $f(x) = \Theta(g(x))$ and say that $f(x)$ is* roughly proportional *to $g(x)$, when there exist $c_L, c_U > 0$ such that for all $x \geq 1$,*

$$c_L g(x) \leq f(x) \leq c_U g(x).$$

We say that $f(x)$ satisfies a *power-law relation* with exponent α whenever $f(x) = \Theta(x^\alpha)$.

Note that these definitions are not equivalent. For example, $f(x) = [2+\cos(x)]g(x)$ satisfies the second definition but not the first while $f(x) = \log(x)g(x)$ satisfies the first but not the second definition. We chose to use Definition 1 instead of the first definition for its simplicity. We start with the following useful lemma.

Lemma 1. *Let $f(x) = \Theta(x^\alpha), g(x) = \Theta(x^\beta)$, then $f(x) + g(x) = \Theta(x^{\max\{\alpha,\beta\}})$.*

Proof. Let c_L^f, c_U^f be the lower-bound and upper-bound constants of f and let c_L^g, c_U^g be those of g. We assume w.l.o.g. that $\alpha > \beta$. For $x \geq 1$,

$$f(x) + g(x) \leq c_U^f x^\alpha + c_U^g x^\beta \leq (c_U^f + c_U^g) x^\alpha,$$

and

$$f(x) + g(x) \geq c_L^f x^\alpha + c_L^g x^\beta \geq c_L^f x^\alpha,$$

so that the conditions of Definition 1 are satisfied and $f(x) + g(x)$ indeed follows a power law with exponent given by $\max\{\alpha, \beta\}$. □

Many real-world networks have degree sequences and community sizes that follow a power law. The distribution of a random variable X is said to satisfy a *power law with exponent* $\alpha > 1$ if

$$\mathbb{P}(X > x) = \Theta(x^{-(\alpha-1)}).$$

Here we have $\alpha - 1$ so that for a continuous random variable with density $f(x)$, there will be a power law with exponent $-\alpha$ between x and $f(x)$. The following lemma will provide an easy way to generate random variables from a continuous Power-law distribution.

Lemma 2. *Let $U \sim$ Unif$[0,1]$, then $X = x_{\min} \cdot U^{-\frac{1}{\alpha-1}}$ follows a power-law distribution with exponent α.*

Proof. For $x > x_{\min}$, the CDF satisfies

$$\mathbb{P}(X > x) = \mathbb{P}(x_{\min} \cdot U^{-\frac{1}{\alpha-1}} > x) = \mathbb{P}\left(U < \left(\frac{x}{x_{\min}}\right)^{-(\alpha-1)}\right) = \left(\frac{x}{x_{\min}}\right)^{-(\alpha-1)}.$$

By setting $c_L = c_U = x_{\min}^{\alpha-1}$, it follows that Definition 1 applies so that this random variable indeed has a power-law distribution. □

We will refer to this distribution as the Pareto distribution with minimum $x_{\min}$ and exponent α. This distribution will be denoted by $\text{Par}(\alpha, x_{\min})$.

The following two Lemmas will be useful when dealing with power-law distributed random variables:

Lemma 3. *Let X be a non-negative random variable,then*

$$\mathbb{E}[X] = \int_0^\infty \mathbb{P}(X > x)dx.$$

Proof. A proof for this can be found in [31]. □

Lemma 4. *Let the distribution of X follow a power law with exponent $\alpha > 1$ and let $f(x) = \Theta(x^\beta)$ for $\beta > 0$, then*

$$\mathbb{P}(f(X) > x) = \Theta(x^{-\frac{\alpha-1}{\beta}}),$$

i.e. $f(X)$ has a power-law distribution with exponent $\frac{\alpha+\beta-1}{\beta}$.

Proof. We write c_U^f, c_L^f for the constants of the power law of f while we use c_U^g, c_L^g for those of g. Then

$$\mathbb{P}(f(X) > x) \leq \mathbb{P}(c_U^f X^\beta > x) = \mathbb{P}\left(X > \left(\frac{x}{c_U^f}\right)^{\frac{1}{\beta}}\right) \leq c_U^g \left(\frac{x}{c_U^f}\right)^{-\frac{\alpha-1}{\beta}} = c_U^g (c_U^f)^{\frac{\alpha-1}{\beta}} x^{-\frac{\alpha-1}{\beta}}.$$

For the other inequality, we write

$$\begin{aligned}\mathbb{P}(f(X) > x) &\geq \mathbb{P}(c_L^f X^\beta > x) \\ &= \mathbb{P}\left(X > \left(\frac{x}{c_L^f}\right)^{\frac{1}{\beta}}\right) \\ &\geq c_L^g (c_L^f)^{\frac{\alpha-1}{\beta}} x^{-\frac{\alpha-1}{\beta}}.\end{aligned}$$

□

Lemma 5. *Let the distribution of X follow a power law with exponent $\alpha > 1$, then $\mathbb{E}[X^\beta]$ is finite if $\beta < \alpha - 1$ while it is infinite for $\beta > \alpha - 1$.*

Proof. For $\beta < \alpha - 1$, we can use Lemma 3 and Lemma 4 to rewrite

$$\mathbb{E}[X^\beta] = \int_0^\infty \mathbb{P}(X^\beta > x)dx \leq \int_0^\infty c_U x^{-\frac{\alpha-1}{\beta}} dx < \infty.$$

For $\beta > \alpha - 1$, we note that for any x, the following lower-bound holds

$$\mathbb{E}[X^\beta] \geq \mathbb{E}[x^\beta \mathbf{1}_{\{X^\beta > x^\beta\}}] = x^\beta \mathbb{P}(X > x) \geq c_L x^{\beta-(\alpha-1)}.$$

This lower-bound diverges to infinity for $x \to \infty$. □

Many of the distributions that we will encounter will converge in distribution to a *mixed Poisson* random variable, i.e. Poisson distributed with a random parameter. We will denote such mixed Poisson random variables by $Poi(\Lambda)$, where the random variable Λ denotes the parameter. For the proof of the following lemma, we refer to Corollary 2.7 of [20].

Lemma 6. *Let $Poi(\Lambda)$ be a mixed Poisson random variable and let Λ have a power-law distribution with exponent $\alpha > 2$. Then $Poi(\Lambda)$ also has a power-law degree distribution with exponent α.*

We are mostly interested in power-law distributions to analyze the distributions of vertex-degrees and community sizes. Since we consider graphs of size n, the degrees and community sizes will always be bounded by n. Therefore, we need an asymptotic definition for power-laws:

Definition 2. *Consider $(\mathcal{S}^{(s)})_{s\geq 1}$ such that $\mathcal{S}^{(s)} = (X_1^{(s)}, \dots, X_s^{(s)})$. Furthermore, let $I^{(s)} \in [s]$ be chosen uniformly at random. We say that this sequence has an* asymptotic power-law distribution *with exponent α when the limiting distribution of $X_{I^{(s)}}^{(s)}$ for $s \to \infty$ exists and is a power-law distribution with exponent α.*

In the context of power-law distributed community sizes, one may be interested in the size of the community of a vertex chosen uniformly at random. This size is distributed according to the *size-biased* distribution of the community-size distribution.

Definition 3. *Let X be a non-negative random variable, we define the* size-biased *version X^* of X to be the random variable with CDF given by*

$$\mathbb{P}(X^* \leq x) = \frac{\mathbb{E}[X\mathbf{1}_{\{X\leq x\}}]}{\mathbb{E}[X]}.$$

The size-biased distribution of the community sizes corresponds to the size of the community that a randomly chosen vertex belongs to. For power-law community sizes, the relation between the power-law exponents is given by the following lemma:

Lemma 7. *Let X have a power-law distribution with exponent $\alpha > 2$, then X^* has a power-law distribution with exponent $\alpha - 1$.*

Proof. We use Lemma 3 to rewrite the expected value in the numerator

$$\begin{aligned}\mathbb{P}(X^* > x) = \frac{\mathbb{E}[X\mathbf{1}_{\{X>x\}}]}{\mathbb{E}[X]} &= \frac{1}{\mathbb{E}[X]}\int_0^\infty \mathbb{P}(X\mathbf{1}_{\{X>x\}} > y)dy \\ &\leq \frac{1}{\mathbb{E}[X]}\int_x^\infty c_U y^{-(\alpha-1)}dy \\ &= \frac{c_U}{(\alpha-2)\mathbb{E}[X]}x^{-(\alpha-2)}.\end{aligned}$$

The lower bound can be obtained by replacing c_U with c_L and flipping the inequality. □

When we want to draw community-sizes from a power-law distribution, we need the community sizes to be discrete and sum to n. First, we will introduce a method to generate s random variables that will sum to m_s such that the sample distribution converges to a Pareto distribution. After that, we will show that we can round these random variables to integers without affecting the power law.

Lemma 8. *Let $\gamma > 2$ and assume that $\frac{m_s}{s} \to m$. Let the random variable $I^{(s)} \in \{1, \dots, s\}$ be chosen uniformly at random. The random variable $X^{(s)} = g(I^{(s)}; s)$, where*

$$g(I^{(s)}; s) = m_s \left[\left(\frac{I^{(s)}}{s} \right)^{\frac{\gamma-2}{\gamma-1}} - \left(\frac{I^{(s)} - 1}{s} \right)^{\frac{\gamma-2}{\gamma-1}} \right],$$

converges in distribution to a Pareto distribution with exponent γ and mean m as $s \to \infty$.

Proof. For large s, we can linearize the second term in the brackets around $I^{(s)}/s$ to obtain

$$\left(\frac{I^{(s)}}{s} \right)^{\frac{\alpha-2}{\alpha-1}} - \left(\frac{I^{(s)} - 1}{s} \right)^{\frac{\alpha-2}{\alpha-1}} = \frac{1}{s} \frac{\alpha - 2}{\alpha - 1} \left(\frac{I^{(s)}}{s} \right)^{-\frac{1}{\alpha-1}} + o(s^{-1}).$$

Furthermore, note that $I^{(s)}/s \xrightarrow{\mathcal{D}} U$ where $U \sim \text{Unif}[0, 1]$. Hence

$$\begin{aligned} X^{(s)} &= m_s \left[\frac{1}{s} \frac{\alpha - 2}{\alpha - 1} \left(\frac{I^{(s)}}{s} \right)^{-\frac{1}{\alpha-1}} + o(s^{-1}) \right] \\ &= \frac{m_s}{s} \left[\frac{\alpha - 2}{\alpha - 1} \left(\frac{I^{(s)}}{s} \right)^{-\frac{1}{\alpha-1}} + o(1) \right] \\ &\xrightarrow{\mathcal{D}} m \frac{\alpha - 2}{\alpha - 1} U^{-\frac{1}{\alpha-1}}, \end{aligned}$$

which, by Lemma 2, follows a Pareto distribution with exponent α. □

Note that, for the sample $X_1, \dots, X_s$ where $X_i = g(i; s)$,

$$\sum_{i=1}^{s} X_i = m_s \sum_{i=1}^{s} \left[\left(\frac{i}{s} \right)^{\frac{\alpha-2}{\alpha-1}} - \left(\frac{i - 1}{s} \right)^{\frac{\alpha-2}{\alpha-1}} \right] = m_s \left[\left(\frac{s}{s} \right)^{\frac{\alpha-2}{\alpha-1}} - \left(\frac{0}{s} \right)^{\frac{\alpha-2}{\alpha-1}} \right] = m_s,$$

so that the sum of the sample is indeed given by m_s. However, these would be continuous variables while community sizes and degrees need to be discrete. We introduce the following way to generate a sample of discrete random variables that are asymptotically power-law distributed:

Definition 4. *We say that a sample $X_1, \dots, X_s$ follows a* discrete fixed-sum power law *with exponent α and sum m_s, denoted by DFSPL(α, s, m_s), if it is obtained by rounding the variables $g(1; s), \dots, g(s; s)$ to integers such that they sum to m_s.*

Next we prove that this rounding does not disturb the power law:

Theorem 1. *A sample $X_1, \dots, X_s \sim$ DFSPL(α, s, m_s) as defined in Definition 4 has an asymptotic power law for $s \to \infty$.*

Proof. Lemma 8 proves that $g(1; s), \dots, g(s; s)$ has an asymptotic power law. For the rounding, define $R_1, \dots, R_s$ by $R_i = X_i - g(i; s)$. Note that $|R_i| \leq 1$. Let $I^{(s)} \in [s]$ be chosen uniformly at random, then $X_{I^{(s)}} = \Theta(g(I^{(s)}; s))$ so that by Lemma 4, $X_{I^{(s)}}$ has an asymptotic power law with the same exponent. □

The lemmas and theorems presented in this section will prove to be helpful in Chapter 5, where we will use these to define a random graph model.

2.3 Random Graph Models

In this section, we will give a brief summary of the random graph models that are most relevant to this thesis. Many random graphs have an adaptation to incorporate community structure. We will adhere to this structure by first introducing the general model before introducing its community-adaptation.

2.3.1 Erdős-Rényi and Stochastic Block Model

The simplest and most well-known random graph model is the Erdős-Rényi (ER) model. In the ER-model, each pair of vertices is connected independently with probability p. Hence, for $G \sim ER(n,p)$, the expected number of edges equals $\binom{n}{2}p$, while the degree of each vertex is binomially distributed with probability p and $n-1$ trials. For $p_n = \lambda/n$, the degree of a vertex in an $ER(n,p_n)$ converges to a Poisson distribution with parameter λ.

A simple way to extend the ER-model to incorporate community structure is to make the probability that the vertices i and j are connected depend on the communities they reside in. This results in the Stochastic Block Model (SBM, [23]). Given a community structure C and a $|C| \times |C|$-matrix B indexed by the communities of C and with entries in $[0,1]$, the probability that i and j are connected is given by

$$p_{ij} = B_{C(i),C(j)}.$$

The Planted Partition Model (PPM) is the simplest case of the SBM. Here B equals p_{in} on the diagonal and p_{out} outside of the diagonal.

2.3.2 Configuration model and Hierarchical Configuration Model

The Configuration Model (CM) starts with a degree sequence and then randomly connects vertices in order to obtain a graph that matches that degree sequence. This is done by giving each vertex a number of half-edges (stubs) corresponding to its desired degree. Then we iteratively pick two stubs at random and connect them to each other. Sometimes this may lead to a vertex being connected to itself (self-loops) or multiple edges between two vertices (multi-edges). It can be proven that, if the degree distribution satisfies certain conditions, the probability that a graph generated by CM is simple (i.e. no self-loops or multi-edges) converges to a strictly positive constant for $n \to \infty$ (see Theorem 7.12 of [22]). Furthermore, for any degree sequence, conditioning on the event that CM generates in a simple graph results in a distribution that is uniform over the set of graphs with that degree distribution (see Proposition 7.15 of [22]).

An extension of CM that can model community structure is the Hierarchical Configuration Model (HCM, [21]). For each community, it takes two inputs: A graph (community interior) that represents the connections within the community and a degree sequence that represents the number of neighbors each community member has outside the community (inter-community degrees). A graph is then generated as follows: The community interiors are kept intact while additional edges are drawn according to a CM with the prescribed inter-community degrees. This way, the local structure of the communities is separated from the mesoscopic inter-community structure, so that both structures can be analyzed separately. Among others, under the assumption that the distribution of community-sizes follows a power-law, each community is connected, sufficiently dense and has inter-community degree that can be bounded by a constant multiple of the community size, it can be proven that the power-law exponent of the degree distribution is related to the power-law exponent of the degree-distribution.

2.3.3 Chung-Lu model and Degree-Corrected SBM

A major drawback of the ER-model is that it cannot model any heterogeneity in the degree distribution. On the other hand, a drawback of the CM is that the event of a vertex i being connected to a vertex j is dependent on the event that i is connected to another vertex k. Both of these issues are resolved in the Chung-Lu model by assigning a weight θ_i to each vertex $i \in [n]$. The probability that the vertices i and j are connected by an edge is p_{ij} (independent of the other edges), where

$$p_{ij} = \min\left\{ \frac{\theta_i \theta_j}{\sum_{k\in[n]} \theta_k}, 1 \right\}.$$

Note that, when allowing for self-loops and under the assumption that $\max_{i\in[n]} \theta_i^2 \leq \sum_{k\in[n]} \theta_k$, the expected degree of each vertex is equal to its weight.

Similar to how SBM adapts the ER-model to model community structure, a block matrix can be used to model community structure in the Chung-Lu model. This results in the Degree-Corrected Stochastic Block Model (DCSBM, [26]). Given a clustering C, a sequence of relative weights $(\theta_i)_{i\in[n]}$ and a symmetric connectivity matrix $(B_{c_1,c_2})_{c_1,c_2\in C}$, DCSBM connects the vertices i and j with probability

$$p_{ij} = \min\left\{ \frac{\theta_i \theta_j}{\sum_{k\in[n]} \theta_k} B_{C(i),C(j)}, 1 \right\},$$

independently of the other edges. However, for general choices of this block matrix, the expected degrees do not equal the given weights, making it difficult to match a degree distribution. This issue is resolved by the Independent LFR model [35], a model related to the LFR model that will be discussed in Section 5.3.3. This model takes a sequence of vertex-weights, a community structure and a mixing parameter μ as input and then sets the parameters of DCSBM such that the expected degree sequence coincides with the vertex-weights while each vertex has a fraction of μ neighbors outside its community in expectation.

Chapter 3

A framework for community detection

In this chapter, we will attempt to formalize the problem of community detection in a probabilistic framework. We distinguish four *ingredients*: 1) the joint distribution of the graph and its communities; 2) the validation index; 3) the quality function; and 4) the algorithm used to optimize the quality function. To guide this formalization, we will look for analogies between community detection and the field of Statistical Learning Theory (SLT).

3.1 The joint distribution of the graph and its communities

The following two premises will form the starting point for our formalization of community detection:

Premise 1 (P1)**.** *There exists a* true *community structure.*

Premise 2 (P2)**.** *The true community structure manifests itself through the connectivity of the graph under consideration.*

The details of this manifestation will be dictated by domain knowledge of the real-world setting. We will denote the graph by G while its true community structure is denoted by T. T will be represented as a partition of the vertices into their communities. Inspired by SLT, we model Premises P1 and P2 by the assumption that our graph and true community structure are drawn from a joint distribution, i.e. $(G, T) \sim \mathcal{D}$. The choice of this distribution ought to be based on domain knowledge and the real-world network under consideration. This choice will be discussed in more detail in Chapter 5. Hereafter, we will use *ground truth* to refer to the true community structure T. The community structure C that results from the community detection method will be referred to as the *candidate*.

In summary, we choose to model the relation between graph and communities by a joint distribution instead of answering the question "What are communities?" in terms of properties of the graph. With this approach, we avoid making assumptions about the intricate relation between the graph and the ground truth. In the flight-data example given in Section 1.1, the relation is relatively simple: the distinction between the countries is older than the airports and practicalities make airports within a country better connected than between countries. Therefore, in this example, the graph comes after the communities. In many other examples, such as social networks, both graph and communities co-evolve, making the relationship between graph and communities even more complicated.

3.2 Validation index

Regardless of what the goal is that one may want to achieve by applying community detection, it is always desirable that the method results in a candidate that resembles the true community structure. This leads to the following premise:

Premise 3 (P3)**.** *The objective of community detection is to find a candidate that has high similarity to the true community structure.*

The exact meaning of "similarity to the true community structure" may differ per application and may depend on the cost of various types of errors. We will assume that the similarity between ground truth T and candidate C is measured by a *validation index* $V(T, C)$. The choice of this index depends on which properties are desirable in the current application. In Chapter 4, we will elaborate on properties that are desirable across many settings and will show which validation indices satisfy them. We note that although typically the validation index does not depend on the graph, it is possible to define graph-dependent validation indices [28]. However, these are outside the scope of this thesis.

Since we assume that our graph and ground truth are generated by a random graph model and C is obtained via an algorithm $\mathcal{A}$ that takes the graph as input, $V(T, \mathcal{A}(G))$ will be a random variable. This makes the task of choosing $\mathcal{A}$ to optimize $V(T, \mathcal{A}(G))$ more subtle. For example, we may aim to optimize its expected value, or a probabilistic lower bound. In Chapter 8, we will obtain a lower-bound $B(G, T)$ for which we can approximate $\mathbb{E}_{G,T\sim\mathcal{D}}[B(G, T)]$ and can prove $V(T, \mathcal{A}(G)) \geq B(G, T)$.

3.3 Quality function and optimization algorithm

Let us compare the objective of finding a candidate that maximizes the validation index to the objective of finding the classifier that minimizes the loss in SLT: in both cases, the quantity to be optimized cannot be directly computed. For SLT, this is resolved by optimizing an intermediate quantity that can be computed from the available data. Many community detection algorithms opt for a similar approach and optimize a *quality function* $Q(G, C)$ that quantifies how well C fits to G. Hence, it may be useful to draw analogies to quality functions in community detection and empirical loss in SLT. Although there are many community detection methods that do not rely on optimizing an intermediate quantity [13], these are outside the scope of this thesis. Restricting to quality-optimizing detection algorithms allows us to split the detection algorithm into two components: the quality function and the optimization algorithm. We stress that the choice of the quality function Q should be instrumental; for $(G, T) \sim \mathcal{D}$, high values of $Q(G, C)$ should correspond to high values of $V(T, C)$. In particular, the choice of Q should depend on $\mathcal{D}$ and V and should therefore be made *after* choosing the others. This is in contrast to what happens in many other works on community detection, where new methods are introduced based on heuristics and afterwards compared to popular alternative methods by a rather arbitrary choice of validation index and random graph. At this point, it is not at all obvious what choices of Q would be reasonable for given choices of $\mathcal{D}$ and V. In Chapter 8, we will attempt to shrink this gap while Chapter 6 will discuss various well-known quality functions.

When the quality function is chosen, we can choose an algorithm that is capable of optimizing this quantity. In this choice, one may also take into account which algorithms perform well when applied to graphs generated by the chosen graph model. Chapter 7 discusses optimization algorithms that can be used to optimize general quality functions.

3.4 Order of ingredients

Above we have argued that in order to do community detection in a theoretically sound manner, multiple ingredients need to be chosen and combined. These dependencies dictate the following order of choosing the ingredients:

1. A joint distribution of the graph and ground truth is chosen based on the real-world setting and the given network.

2. A validation index is chosen based on the application.

3. A quality function is chosen that provides the most desirable guarantees with respect to the graph model and validation index.

4. An optimization algorithm is chosen that is suitable for optimizing the chosen quality function for graphs from the chosen model.

In other works of community detection, the quality function and optimization algorithm are considered one entity (the community detection method). Furthermore, not much attention is paid to the dependencies between this detection method and the other ingredients. Figure 3.1 illustrates the described dependencies between the choices of the ingredients.

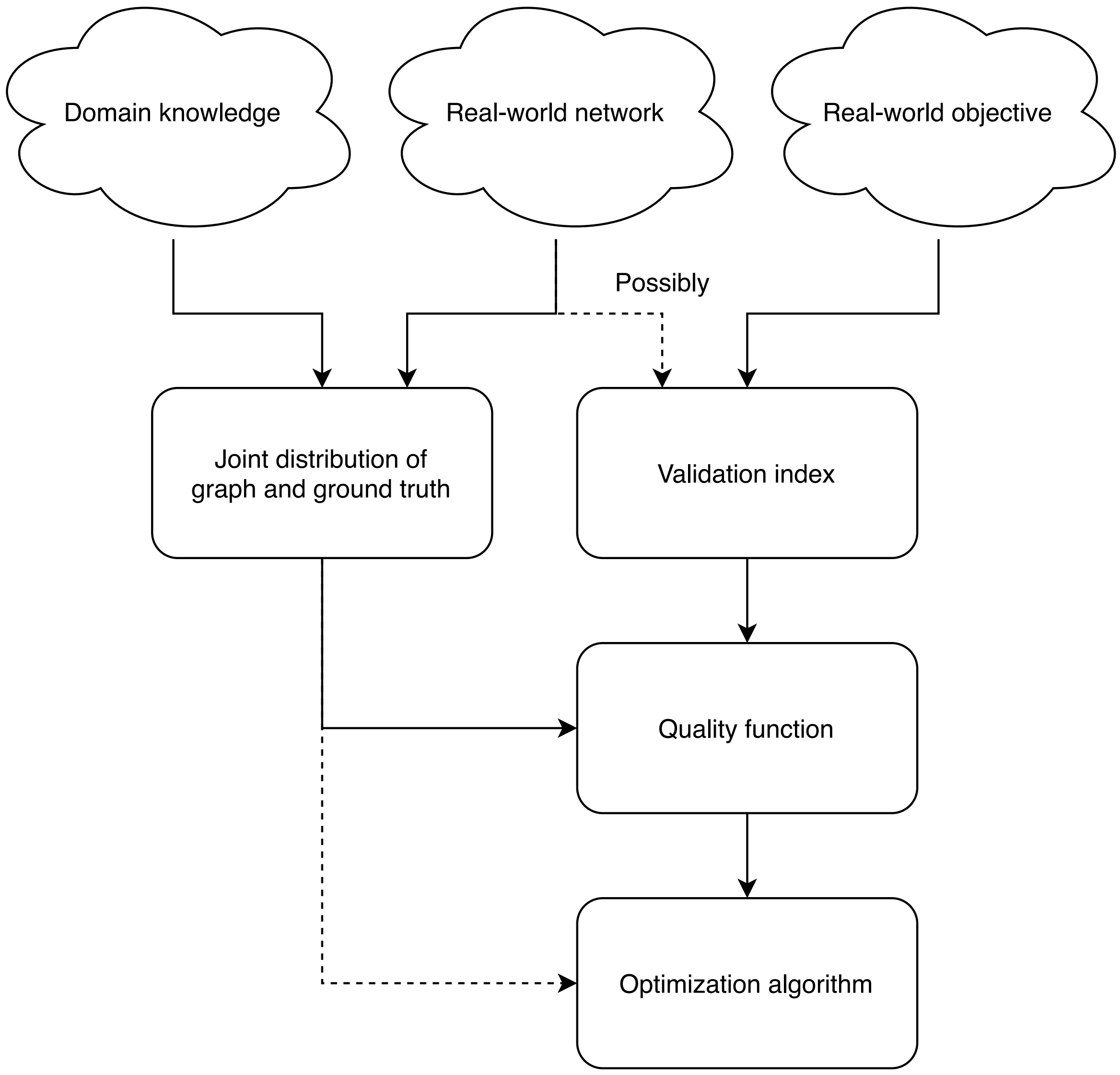


Figure 3.1: The proposed workflow of community detection. The boxes denote choices that need to be made while the clouds denote external inputs. Dashed lines are optional dependencies.

Chapter 4

Validation indices

In this chapter, we discuss the choice of a suitable validation index to measure the similarity between the candidate clustering and the ground truth. This chapter is joint work with Liudmila Prokhorenkova and the original version is available on arXiv [18]. It addresses the more general subject of *cluster similarity indices*, where the items being clustered are not necessarily nodes of a network. Because of this, there will be some differences in notation and terminology. These are re-introduced where necessary. In particular, we will denote indices by I instead of V.

4.1 Introduction

Clustering is an unsupervised machine learning problem, where the task is to group objects which are similar to each other. When the objects are vertices in a network, this problem corresponds to community detection. Clustering is used across various applications, including text mining, online advertisement, anomaly detection, and many others [2, 50].

To measure the quality of a clustering algorithm, one has to compare two partitions: candidate and ground truth (the latter one can be obtained, e.g., by human assessors). Nowadays, there are many cluster similarity indices proposed for that, but which one is the best is still a subject of debate [30], and the current chapter addresses this issue. Depending on the application, different properties of a similarity index could be desirable. In this chapter, we formally define requirements that are desirable across various applications, discuss their importance, and formally analyze which similarity indices satisfy them.

While many of the ideas discussed in this chapter can be applied to all similarity indices, we particularly focus on pair-counting ones (Rand and Jaccard are the most well-known examples). We formally prove that among dozens of known indices, only two satisfy all the properties except for being a distance: Correlation Coefficient and Sokal & Sneath's first index [30]. Surprisingly, both indices are rarely used for cluster evaluation. The correlation coefficient has an additional advantage of being easily convertible to a distance measure via the arccosine function. The obtained index, which can be thought of as an *angle between partitions*, satisfies all the requirements except the *constant baseline*, which is still satisfied asymptotically.

The *constant baseline* requirement is a particular focus of this chapter. Informally, a sensible index should not prefer one candidate partition over another just because it has too large or too small clusters. To the best of our knowledge, we are the first to formalize this requirement. We tested this property for all indices under consideration. For pair-counting indices, we carried some further analysis and defined several particular types of biases. In this respect, our work improves the results from the recent work by Lei et al. [30]. We also propose a simple statistical test allowing to reject the constant baseline for a general index.

We discuss the practical importance of this analysis in Section 4.7. In particular, we describe an online experiment within a major news aggregator system, which illustrates that an improper choice of a similarity index can lead to degraded user experience.

4.2 Related work

Several attempts to the comparative analysis of cluster similarity indices have been made in the literature, both in machine learning and complex networks communities. In particular, the problem of indices favoring clusterings with smaller or larger clusters has been identified in numerous works [1, 30, 42, 46, 47]. Some remedies to these biases have been proposed: in one attempt [3], the similarity index is multiplied by a penalty factor that decreases with the difference in the number of clusters. In another attempt [39], the authors propose standardized mutual information.

A paper closely related to this chapter [4] formulates several constraints (axioms) for cluster similarity indices. Some of the properties are particular cases of those discussed in this chapter, while others seem to be strongly application dependent. In this chapter, we give a more comprehensive list of constraints and focus on those that are desirable in a wide range of applications.

While we focus on cluster similarity indices (partition-partition comparisons), some work has been done for graph-partition similarity indices, often referred to as goodness or quality measures. Such indices quantify how well a community structure (given by a partition) fits a graph; the most well-known example is *modularity* [34]. Axioms that these measures ought to satisfy are given in [7, 45]. Note that all pair-counting indices discussed in this chapter can also be used for graph-partition similarity (see Section 4.3). Furthermore, these indices can also be used for graph-graph comparison. For example, [12] discusses measuring the Hamming distance and Jaccard distance between the edge-sets of graphs. Here, the Hamming distance corresponds to the Mirkin metric (defined in the next section).

4.3 Background and notation

Similarity indices used throughout the literature generally fall into four categories [4]: 1) indices based on set matching such as Purity, Inverse Purity and their harmonic mean (F-measure); 2) indices based on entropy such as Variation of Information [32], Normalized Mutual Information [42] and Standardized Mutual Information [39]; 3) indices based on edit distance; and 4) pair-counting indices such as the well-known Rand [37] and Jaccard [25] indices. Some indices are mixtures of the above categories, such as BCubed [4]. In the supplementary materials (Sections A.1 and A.2), we define the indices that will be discussed throughout this chapter.

We pay special attention to pair-counting indices. For this, it is convenient to use an alternative representation for clusterings.

Definition 5. *A similarity index is a* pair-counting index *if it can be expressed as a function of the four pair-counts* $N_{11}, N_{10}, N_{01}, N_{00}$.

Note that even though we focus on comparing clusterings to clusterings, pair-counting functions can also be used to compare graphs to graphs and clusterings to graphs. So, one may see a connection between graph and cluster similarity indices. For example, the Mirkin metric is a pair-counting index that coincides with the Hamming distance between the edge-sets of two graphs [12]. Another example is the Jaccard graph distance, which turns out to be more appropriate for comparing sparse graphs [12]. Thus, all pair-counting indices and their properties discussed in this chapter can also be applied to graph-graph and graph-partition similarities.[1] For this reason, we will use I for a general similarity index instead of V so that $I(A, B)$ will denote the similarity between the clustering A and B. Throughout this chapter, we will often use the notation $k_A = |A|$ to denote the number of clusters in clustering A.

A list of 26 known pair-counting indices can be found in [30]. We have extended this list to Table 1 in the supplementary materials and will only mention indices of particular interest throughout the main text. Some of the pair-counting indices have slight variants that are essentially the same. For example, the Hubert Index [24] can be expressed as a linear transformation of the Rand index as $H(A, B) = 2R(A, B) - 1$. Similarly, the two Wallace indices are related as $W_1(A, B) = W_2(B, A)$. As all the requirements defined in this chapter are invariant under linear transformations and interchanging A and B, they do not have to be checked for such variants. Hence, we define the following linear equivalence relation on similarity indices and check the requirements for at most one representative of each equivalence class:

Definition 6. *Similarity indices* I_1 *and* I_2 *are* linearly equivalent *if there exists a nonconstant linear function* f *such that either* $I_1(A, B) = f(I_2(A, B))$ *or* $I_1(A, B) = f(I_2(B, A))$.

[1]In the supplementary materials (Section A.3) we also prove that pair-counting similarity indices can be uniquely characterized by the property of being pair-symmetric.

This definition is symmetric, reflexive, and transitive; thus, it indeed defines an equivalence relation. Furthermore, it allows us to conveniently restrict to indices for which higher numerical values indicate higher similarity of partitions. Table 2 in the supplementary materials lists the equivalent indices. Note that our linear equivalence differs from the less restrictive monotonous equivalence given in [6]. In this chapter, we have to restrict to linear equivalence as the constant baseline requirement is not invariant to non-linear transformations.

4.4 Requirements for cluster similarity indices

In this section, we motivate and formally define requirements that are desirable for cluster similarity indices. In Table 4.1, several indices of particular interest are listed along with the requirements satisfied. In Table 4.2, a similar list is given for pair-counting indices, along with some additional pair-counting-specific requirements. In Section B of the supplementary materials, we give the proofs for all entries of these tables.

Requirement 1. Maximal agreement. The numerical value that an index assigns to a similarity must be easily interpretable. In particular, it should be easy to see whether the candidate clustering is maximally similar to (i.e., coincides with) the ground truth clustering. Formally, we require that $I(A, A)$ is constant and either a strict upper or a strict lower bound for $I(A, B)$ for all $A \neq B$. The equivalence from Definition 6 allows us to assume that $I(A, A)$ is a maximum w.l.o.g. This requirement is easy to check and it is satisfied by almost all indices, except for SMI and Wallace.

Requirement 2. Symmetry. Similarity is intuitively understood as a symmetric concept. Therefore, a good similarity index is expected to be symmetric, i.e., $I(A, B) = I(B, A)$ for all partitions A, B.[2] Tables 4.1 and 4.2 show that most indices are indeed symmetric. The asymmetric ones are precision and recall (Wallace) and FNMI [3], which is a product of symmetric NMI with the asymmetric penalty factor $e^{-|k_A - k_B|/k_A}$.

Requirement 3. Monotonicity. Clearly, when one clustering is changed such that it resembles the other clustering more, the similarity score ought to improve. Hence, we require an index to be monotone w.r.t. changes that increase the similarity.

Definition 7. *For clusterings A and B, we say that B' is an A-*consistent improvement *of B iff $B \neq B'$ and all pairs of elements agreeing in A and B also agree in A and B'.*

This leads to the following monotonicity requirement.

Definition 8. *An index I satisfies the monotonicity requirement if for every two clusterings A, B and any B' that is an A-consistent improvement upon B, it holds that $I(A, B') > I(A, B)$.*[3]

We now give an alternative, equivalent, definition of this requirement. For this, we define the following operations:

- **Perfect split**: We say that B' is a perfect split of B (w.r.t. A) if B' is obtained from B by splitting a single cluster b_1 into two clusters b'_1, b'_2 such that no two elements of the same cluster of A are in different parts of this split, i.e., for all i, $a_i \cap b_1$ is a subset of either b'_1 or b'_2.
- **Perfect merge**: We say that B' is a perfect merge of B (w.r.t. A) if there exists some a_i and $b_1, b_2 \subset a_i$ such that B' is obtained by merging b_1, b_2 into b'_1.

It is easily verified that if B' is obtained from B by a sequence of perfect splits and merges, then B' is an A-consistent improvement of B, since at each step we only create new agreeing pairs. The following theorem states that the opposite also holds (see Section D.1 of the supplementary materials for the proof).

Theorem 2. *B' is an A-consistent improvement of B iff B' can be obtained from B by a sequence of perfect splits and perfect merges.*

[2] In some specific applications A and B may have different roles (e.g., ground truth and candidate partitions) and an asymmetric index may be appropriate if there are different consequences of making false positives or false negatives. We will refrain from discussing this further.

[3] Here we again use Definition 6 which allows us to assume that I gives higher numerical values for higher similarity.

	Max. agreement	Symmetry	Monotonicity	Distance	Const. baseline
NMI	✓	✓	✓	✗	✗
NMI$_{\max}$	✓	✓	✗	✓	✗
FNMI	✓	✗	✗	✗	✗
VI	✓	✓	✓	✓	✗
SMI	✗	✓	✗	✗	✓
FMeasure	✓	✓	✓	✗	✗
BCubed	✓	✓	✓	✗	✗

Figure 4.1: Requirements for non-pair-counting similarity indices

	Max. agreement	Min. agreement	Symmetry	Monotonicity	Strong monotonicity	Distance	Const. baseline	ACB
R	✓	✓	✓	✓	✓	✓	✗	✗
AR	✓	✗	✓	✓	✗	✗	✓	✓
J	✓	✗	✓	✓	✗	✓	✗	✗
W	✗	✗	✗	✗	✗	✗	✗	✗
D	✓	✗	✓	✓	✗	✗	✗	✗
CC	✓	✓	✓	✓	✓	✗	✓	✓
S&S1	✓	✓	✓	✓	✓	✗	✓	✓
CD	✓	✓	✓	✓	✓	✓	✗	✓

Figure 4.2: Requirements for pair-counting indices. ACB stands for Asymptotic Constant Baseline.[3]

Note that this monotonicity is a stronger form of the first two constraints defined in [4]: *Cluster Homogeneity* is a weaker form of our monotonicity w.r.t. perfect splits, while *Cluster Equivalence* is equivalent to our monotonicity w.r.t. perfect merges. The authors prove that BCubed satisfies their constraints.

Monotonicity is a critical requirement for cluster similarity indices. However, not all indices satisfy this: we have found counterexamples that prove that SMI, FNMI, and Wallace do not satisfy our monotonicity requirement. Furthermore, for NMI, whether monotonicity is satisfied depends on the normalization: the common normalization by the average of the entropies satisfies monotonicity, while the normalization by the maximum of the entropies does not.

Requirement 4. Distance. For some applications, a distance-interpretation of dissimilarity may be desirable: whenever A is similar to B and B is similar to C, then A should also be somewhat similar to C. A function d is a distance metric if it satisfies three distance axioms: 1) symmetry ($d(A,B) = d(B,A)$); 2) positive-definiteness ($d(A,B) \geq 0$ with equality iff $A = B$); 3) the triangle inequality ($d(A,C) \leq d(A,B) + d(B,C)$). We say that I is linearly transformable to a distance metric if there exists a linearly equivalent index that satisfies these three distance axioms. Note that all three axioms are invariant under re-scaling of d. We have already imposed the symmetry as a separate requirement, while the positive-definiteness is equivalent to the maximal agreement requirement. Therefore, whenever I has these two properties, it satisfies the distance requirement iff $d(A,B) = c_{\max} - I(A,B)$ satisfies the triangle inequality.

Examples of popular indices satisfying this requirement are Variation of Information and the Mirkin metric. In [47] it is proved that when Mutual Information is normalized by the maximum of entropies, the resulting NMI is equivalent to a distance metric. A proof that the Jaccard index is equivalent to a distance is given in [27].

Requirement 5. Constant baseline. Obviously, a good similarity index should not give a preference to a candidate clustering B over another clustering C just because B has many or few clusters. This intuition can be formalized using random partitions: assume that we have some ground truth clustering A and two independent random partitions B and C. While intuitively both random guesses are equally bad approximations of A, it has been known throughout the literature [1, 47, 39] that some indices tend to give higher scores for random guesses with a larger number of clusters. Ideally, we want the similarity value of a random candidate w.r.t. the ground truth partition to have a fixed expected value c_{base} (independent of A or the sizes of B). We formalize this in the following way. Let $S(B)$ denote the specification of the cluster sizes of the clustering B, i.e., $S(B) := [|b_1|, \ldots, |b_{k_B}|]$. Here we use the notation $[\ldots]$ to denote that it is a multiset since the sizes and their multiplicities matter but not their

[3]All indices of Table 1 in the supplementary materials that are excluded from this table do not satisfy either constant baseline, symmetry, or maximal agreement.

order. For a cluster sizes specification s, let $\mathcal{C}(s)$ be the uniform distribution over clusterings B with $S(B) = s$.

Definition 9. *An index I satisfies the* constant baseline requirement *whenever there exists a constant c_{base} so that for any cluster-sizes specification s and clustering A with $1 < k_A < n$, it holds that $\mathbb{E}_{B\sim\mathcal{C}(s)}[I(A,B)] = c_{base}$.*

In this definition, we have excluded the cases where A is a trivial clustering consisting of either 1 or n clusters. If we would include these cases in the definition, then we would run into problems for $s = S(A)$, as $\mathcal{C}(s)$ would be a constant distribution, surely returning A and any sensible index should have $I(A, A) \neq c_{\text{base}}$. Furthermore, note that this requirement is symmetric since it does not matter whether we permute the labels of A while keeping B constant or permute the labels of B while keeping A constant.

Let us show that this definition of the constant baseline applies not only to uniform (within a given sizes specification) but also to all symmetric distributions over clusterings.

Definition 10. *We say that a distribution over clusterings $\mathcal{B}$ is* element-symmetric *if for every two clusterings B and B' that have the same cluster-sizes, $\mathcal{B}$ returns B and B' with equal probabilities.*

Lemma 9. *Let I be an index with a constant baseline as defined in Definition 9, let A be a clustering with $1 < k_A < n$ and let $\mathcal{B}$ be an element-symmetric distribution. Then $\mathbb{E}_{B\sim\mathcal{B}}[I(A,B)] = c_{base}$.*

Proof. We write

$$\begin{aligned}
\mathbb{E}_{B\sim\mathcal{B}}[I(A,B)] =& \sum_s \mathbb{P}_{B\sim\mathcal{B}}(S(B)=s)\, \mathbb{E}_{B\sim\mathcal{B}}[I(A,B)|S(B)=s] \\
=& \sum_s \mathbb{P}_{B\sim\mathcal{B}}(S(B)=s)\, \mathbb{E}_{B\sim\mathcal{C}(s)}[I(A,B)] \\
=& \sum_s \mathbb{P}_{B\sim\mathcal{B}}(S(B)=s)\, c_{\text{base}} \\
=& c_{\text{base}},
\end{aligned}$$

where the sum ranges over cluster-sizes of n elements. □

Note that Definition 9 may be challenging to verify, as the contingency-distribution for fixed cluster-sizes is combinatorially complex. For pair-counting indices, we have developed some additional tools that help proving the constant baseline requirement (see Section 4.5).

The constant baseline requirement is extremely important in many practical applications: if an index violates this requirement, then its optimization may lead to undesirably biased results (see Section 4.7 for additional discussions). Examples of indices that satisfy the constant baseline requirement with $c_{\text{base}} = 0$ are Adjusted Rand index, correlation coefficient and SMI. Sokal&Sneath-1 has a constant baseline at $c_{\text{base}} = \frac{1}{2}$. However, all other considered indices including popular ones such as NMI, Rand, and Jaccard do not satisfy this requirement. Although it can be challenging to prove or disprove the constant baseline requirement for a general index not included to this chapter, it is relatively easy to verify it by numerical experiments. Section C of the supplementary materials describes statistical tests that can be used to test whether an index satisfies the constant baseline requirement. The Python implementation is available at `github.com/MartijnGosgens/validation_indices`.

4.5 Requirements for pair-counting indices

In this section, we discuss additional requirements for pair-counting indices. Throughout this section, we interchangeably use the notation $I(A, B)$ and $I(N_{11}, N_{10}, N_{01}, N_{00})$.

Requirement 1′. Minimal agreement. The maximal agreement requirement gives a numerical value to high agreement, which helps make the upper range of the index more interpretable. Similarly, a numerical value for low agreement would make the lower range of the index interpretable. For general partitions, minimal agreement is not well defined: it is not clear which partition would be most dissimilar to a given partition. However, referring to Lemma 1 of the supplementary materials, pair-counting indices form a subclass of graph similarity indices. For a given graph G, it is clear that the graph most dissimilar to G is its complement G^C (where two edges are connected precisely if they are not connected in G). Comparing a graph to its complement would result in pair-counts $N_{11} = N_{00} = 0$ and $N_{10} + N_{01} = N$.[4] This motivates the following definition:

Definition 11. *We define a pair-counting index I to satisfy the* minimal agreement requirement *if there exists a constant $c_{\min}$ so that $I(N_{11}, N_{10}, N_{01}, N_{00}) \geq c_{\min}$ with equality if and only if $N_{11} = N_{00} = 0$.*

Clearly, this requirement is satisfied by Rand, Correlation Coefficient and Sokal&Sneath-1, while it is not satisfied by Jaccard, Wallace and Dice. Less obvious is the fact that Adjusted Rand does not satisfy this requirement. Substituting $N_{11} = N_{00} = 0$ gives the non-constant $\mathrm{AR}(0, N_{10}, N_{01}, 0) = -\frac{N_{10}N_{01}}{\frac{1}{2}N^2 - N_{10}N_{01}}$.

Requirement 3′. Strong monotonicity. For pair-counting indices, the monotonicity requirement can be strengthened:

Definition 12. *An index is* pair-counting monotone *if it increases when incrementing either N_{11} or N_{00} while decrementing N_{10} or N_{01}, while $N_{11} + N_{00} + N_{10} + N_{01} = N$.*

Note that a perfect split increases N_{00} and decreases N_{01}, while a perfect merge increases N_{11} and decreases N_{10}. Therefore, monotonicity follows from pair-counting monotonicity. However, the opposite is not always true: in the supplementary materials (Section D.2) we show a simple example of such an index. Nevertheless, among the indices considered in this chapter we did not find such examples. So, below we give an even stronger definition of pair-counting monotonicity whose domain is not constrained to the four pair-counting variables summing to N.

Definition 13. *A pair-counting index I satisfies* strong monotonicity *if it increases with N_{11}, N_{00} and decreases with N_{10}, N_{01}.*

The strong monotonicity requirement allows for comparing similarities across different settings. For example, we could compare the similarity between two clusterings A_1, B_1 on n_1 elements with the similarity between A_2, B_2 on n_2 elements, even when $n_1 \neq n_2$. This ability to compare similarity scores across different numbers of elements is similar to the *Few data points* property of SMI [39] that allows its scale to have a similar interpretation across different settings.

We found several examples of indices that satisfy Requirement 3 while not satisfying Requirement 3′. Jaccard and Dice indices are constant w.r.t. N_{00}, so they are not strongly monotone. A more interesting example is the Adjusted Rand index which may become strictly larger if we only increase N_{10}. In Table 4.2 we list both monotonicity and strong monotonicity requirements.

Requirement 5′. Asymptotic Constant Baseline (ACB). For pair-counting indices, some further analysis of the expected value of the index is possible. Recall constant baseline from Definition 9. Let $m_A = N_{11} + N_{10}$, $m_B = N_{11} + N_{01}$ be the number of intra-cluster pairs of A and B, respectively. Note that m_A and m_B are constant as A is constant and $B \sim \mathcal{C}(s)$, so that its cluster-sizes are constant. Furthermore, the pair-counts N_{10}, N_{01}, N_{00} are functions of N, m_A, m_B, N_{11}. Hence, to find the expected value of the index, we only need to inspect it as a function of a single random variable N_{11}. For a given pair, the probability that it is an intra-cluster pair of both clusterings is given by $m_A m_B / N^2$, so the expected values of the pair-counts are

$$\begin{aligned} \overline{N_{11}} &:= \frac{m_A m_B}{N}, & \overline{N_{10}} &:= m_A - \overline{N_{11}}, \\ \overline{N_{01}} &:= m_B - \overline{N_{11}}, & \overline{N_{00}} &:= N - m_A - m_B + \overline{N_{11}}. \end{aligned} \tag{4.1}$$

To the best of our knowledge, all pair-counting indices that satisfy the constant baseline requirement are linear functions of N_{11} when expressed in terms of N_{11}, m_A, m_B, N. This can be explained by the fact that only the first moment of N_{11} can be expressed in terms of the pair-counting variables, while all higher moments depend on the specific cluster-sizes of both clusterings. For these N_{11}-linear indices, substituting the expected pair-counts gives the expected value of the index. This motivates the following relaxation of the constant baseline requirement.

[4]Note, however, that for a general partition A, there exists no partition B that disagrees on every pair; this is only possible if A consists of either 1 or n clusters.

Definition 14. *A pair-counting index I satisfies the* Asymptotic Constant Baseline requirement *if there exists a constant c_{base} so that for all A with $1 < k_A < n$, it holds that $I\left(\overline{N_{11}}, \overline{N_{10}}, \overline{N_{01}}, \overline{N_{00}}\right) = c_{base}$, for all cluster-size specifications s, where the arguments are the expected values of the pair-counts for $B \sim \mathcal{C}(s)$.*

Note that the above definition coincides with the earlier constant baseline from Definition 9 if I is a linear function in N_{11} for fixed m_A, m_B, N. The name of the above requirement is justified by the next result, for which we need to make a mild assumption:

Definition 15. *An index I is said to be* scale-invariant*, if it can be expressed as a continuous function of the three variables $p_A := m_A/N, p_B := m_B/N$ and $p_{AB} := N_{11}/N$.*

All indices in Table 4.2 are scale-invariant. For such indices, we will write $I^{(p)}(p_{AB}, p_A, p_B)$. Note that when $B \sim \mathcal{C}(s)$ for some s, the values p_A, p_B are constants while p_{AB} is a random variable. Therefore, we further write P_{AB} to stress that this is a random variable. The following theorem holds (see Section D.3 of the supplementary material for the proof).

Theorem 3. *Let I be a scale-invariant pair-counting index, and consider a sequence of clusterings $A^{(n)}$ and cluster-size specifications $s^{(n)}$. Let $N_{11}^{(n)}, N_{10}^{(n)}, N_{01}^{(n)}, N_{00}^{(n)}$ be the corresponding pair-counts. Then for any $\varepsilon > 0$, as $n \to \infty$,*

$$\mathbb{P}\left(\left|I\left(N_{11}^{(n)}, N_{10}^{(n)}, N_{01}^{(n)}, N_{00}^{(n)}\right) - I\left(\overline{N_{11}^{(n)}}, \overline{N_{10}^{(n)}}, \overline{N_{01}^{(n)}}, \overline{N_{00}^{(n)}}\right)\right| > \varepsilon\right) \to 0.$$

This result justifies the usage of the name *Asymptotic Constant Baseline requirement* as indices satisfying it, will converge in probability to c_{base} for $n \to \infty$. For smaller n, by computing the second order Taylor expansion of the index around $P_{AB} = p_A p_B$, one can approximate the difference from the expected value to its limiting value in terms of the variance of P_{AB}.

Biases of cluster similarity indices. In [30], three types of biases for cluster similarity indices are described: *NCinc* — the average value for a random guess increases monotonically with the Number of Clusters (NC) of the candidate; *NCdec* — the average value for a random guess decreases monotonically with the number of clusters, and *GTbias* — the direction of the monotonicity depends on the specific Ground Truth (GT). In particular, the authors conclude from numerical experiments that Jaccard suffers from NCdec and analytically prove that Rand suffers from GTbias, where the direction of the bias depends on the quadratic entropy of the ground truth clustering. First, we argue that these biases are not well defined, then we replace these by similar (but well-defined) biases and show how our analysis allows to easily test indices on these biases.

We argue that the quantity of interest should not be the number of clusters, but the number of intra-cluster pairs of the candidate. Theorem 3 shows that the asymptotic value of the index depends on the number of intra-cluster pairs of both clusterings. Although general clusterings with more clusters tend to have less intra-cluster pairs, one can easily construct clusterings that both have many clusters and intra-cluster pairs. For instance, let B be a random clustering consisting of three clusters, each of size $n/3$. Consider constructing B' from B by merging the first two clusters and splitting the third cluster into $n/3$ clusters of size 1. The number of clusters increases by $n/3 - 2$ while the number of intra-cluster pairs increases by $(n/3)^2 - \binom{n/3}{2}$. This will lead to an increase in the expected value of the Jaccard index with respect to any ground truth A (as will be shown below Definition 16). In contrast, [30] classifies Jaccard as an NCdec index, so that the expected value should increase, contradicting the definition of NCdec.

Therefore, we rename *NCinc* to *NPdec* and *NCdec* to *NPinc*, where NP stands for Number of Pairs.

Definition 16. *Let I be a scale-invariant pair-counting index whose corresponding $I^{(p)}$ is differentiable w.r.t. p_B and p_{AB}. We define the following biases:*

1. *I suffers from* NPinc bias *if there are $p_A, p_B \in (0, 1)$ such that $\frac{d}{dp_B}[I^{(p)}(p_A p_B, p_A, p_B)] > 0$;*

2. *I suffers from* NPdec bias *if there are $p_A, p_B \in (0, 1)$ such that $\frac{d}{dp_B}[I^{(p)}(p_A p_B, p_A, p_B)] < 0$;*

3. *I suffers from the* ground truth bias *if it suffers from both NPinc and NPdec biases.*

Applying this definition to Jaccard $J^{(p)}(p_A p_B, p_A, p_B) = \frac{p_A p_B}{p_A + p_B - p_A p_B}$ and Rand $R^{(p)}(p_A p_B, p_A, p_B) = 1 - p_A - p_B + 2p_A p_B$ immediately shows that Jaccard suffers from NPinc bias and Rand suffers from the ground truth bias, confirming the findings of [30]. Furthermore, the direction of the monotonicity for the ground truth bias of Rand is now determined by the condition $p_A > \frac{1}{2}$ instead of the more complicated but equivalent condition on the quadratic entropy of A that is given in [30]. Performing the same for Wallace and Dice shows that both suffer from NPinc bias. Note that an index satisfying the Asymptotic Constant Baseline requirement will not have any of these biases as $I^{(p)}(p_A p_B, p_A, p_B) = c_{\text{base}}$.

4.6 Correlation Distance

Although there are some works [11, 30] listing Pearson correlation as a cluster similarity index, it has not received the attention that our results suggest it deserves. The correlation coefficient satisfies all requirements except being a distance. In this section, we show how a monotone transformation of the correlation coefficient results in an index that may be even more suitable in applications where the distance requirement is important. When taking the arccosine of the coefficient, the resulting index is a distance metric, at the cost of not satisfying the exact constant baseline. It does, however, still satisfy an asymptotic constant baseline and we will prove that its expectation is very close to being constant. To the best of our knowledge, this Correlation Distance has never before been used as a similarity index for comparing clusterings throughout the literature.

We define Correlation Distance (CD) as

$$\mathrm{CD}(A, B) := \frac{1}{\pi} \arccos \mathrm{CC}(A, B), \tag{4.2}$$

where CC is the Pearson correlation coefficient, which has the scale-invariant representation

$$\mathrm{CC}^{(p)}(P_{AB}, p_A, p_B) = \frac{P_{AB} - p_A p_B}{\sqrt{p_A(1-p_A)p_B(1-p_B)}}.$$

The factor $\frac{1}{\pi}$ scales the index to $[0, 1]$.

The monotone arccosine transformation in 4.2 only affects the (exact) constant baseline and distance requirements, the rest of the requirements are inherited from the correlation coefficient. We first verify that this index is indeed a distance metric:

Theorem 4. *The Correlation Distance is indeed a distance.*

Proof. First we map each partition A to an N-dimensional vector on the unit sphere by

$$\vec{u}(A) := \begin{cases} \frac{1}{\sqrt{N}}\mathbf{1} & \text{if } k_A = 1, \\ \frac{\vec{b}_A - p_A \mathbf{1}}{\|\vec{b}_A - p_A \mathbf{1}\|} & \text{if } 1 < k_A < n, \\ -\frac{1}{\sqrt{N}}\mathbf{1} & \text{if } k_A = n, \end{cases}$$

where $\mathbf{1}$ is the N-dimensional all-one vector and $\vec{b}_A$ is the binary vector representation of a partition introduced in Section 2.1. Straightforward computation gives $\|\vec{b}_A - p_A \mathbf{1}\| = \sqrt{N p_A(1-p_A)}$, and standard inner product

$$\langle \vec{b}_A - p_A \mathbf{1}, \vec{b}_B - p_B \mathbf{1} \rangle = N(p_{AB} - p_A p_B),$$

so that indeed

$$\langle \vec{u}(A), \vec{u}(B) \rangle = \frac{\langle \vec{b}_A - p_A \mathbf{1}, \vec{b}_B - p_B \mathbf{1} \rangle}{\|\vec{b}_A - p_A \mathbf{1}\| \|\vec{b}_B - p_B \mathbf{1}\|} = CC^{(p)}(p_{AB}, p_A, p_B).$$

It is a well-known fact that the inner product of two vectors of unit length corresponds to the cosine of their angle. Hence, taking the arccosine gives us the angle. The angle between unit vectors corresponds to the distance along the unit sphere. As $\vec{u}$ is an injection from the set of partitions to points on the unit sphere, we may conclude that this index is indeed a distance on the set of partitions. $\square$

As the correlation has a constant baseline at 0, CD has an asymptotic constant baseline at $\frac{1}{2}$. The expected deviation from this baseline is given by the following theorem:

Theorem 5. *Given ground truth A with a number of clusters $1 < k_A < n$, a cluster-size specification s and a random partition $B \sim \mathcal{C}(s)$, the expected difference between Correlation Distance and its baseline is given by*

$$\mathbb{E}_{B\sim\mathcal{C}(s)}[\mathrm{CD}(A,B)] - \frac{1}{2} = -\frac{1}{\pi}\sum_{k=1}^{\infty}\frac{(2k)!}{2^{2k}(k!)^2}\frac{\mathbb{E}_{B\sim\mathcal{C}(s)}[\mathrm{CC}(A,B)^{2k+1}]}{2k+1}.$$

Proof. We take the Taylor expansion of the arccosine around $\mathrm{CC}(A,B)=0$ and get

$$\mathrm{CD}(A,B) = \frac{1}{2} - \frac{1}{\pi}\sum_{k=0}^{\infty}\frac{(2k)!}{2^{2k}(k!)^2}\frac{\mathrm{CC}(A,B)^{2k+1}}{2k+1}.$$

We take the expectation of both sides and note that the first moment of CC equals zero, so the starting index is $k=1$. $\square$

For $B \sim \mathcal{C}(s)$ and large n, the value $CC(A,B)$ will be concentrated around 0. This explains that in practice, the mean tends to be very close to the asymptotic baseline. To the best of our knowledge, there does not exist a cluster similarity index that is a distance while having the exact constant baseline.

4.7 Practical importance

Our analysis clearly suggests that not all similarity indices are equally good. To see whether the differences between indices are important in practical applications, we conducted a series of experiments within a major news aggregator system. This system aggregates all news articles to *events* and shows the list of most important events to users. For grouping, a clusterization algorithm is used and the quality of this algorithm affects the user experience: merging different clusters may lead to not showing an important event, while too much splitting may cause the presence of duplicate events.

There is an algorithm $\mathcal{A}_{prod}$ currently used in production and two alternative algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$. To decide which one is better for the system, we need to compare them. For that, we manually grouped 1K news articles about volleyball, collected during a period of three days into events. Then, we compared the obtained ground truth partition with partitions A_{prod}, A_1, and A_2 obtained by $\mathcal{A}_{prod}$, $\mathcal{A}_1$, and $\mathcal{A}_2$, respectively (see Table 3 in the Appendix). According to most of the indices, A_2 is closer to the ground truth partition than A_1, and A_1 is closer than A_{prod}. However, according to some indices, including the well-known $\mathrm{NMI}_{\max}$, NMI and Rand, A_1 better corresponds to the ground truth partition than A_2. As a result, we see that *different similarity indices may differently rank the algorithms* in practical applications.

To further see which algorithm better agrees with user preferences, we launched the following online experiment. During one week we compared $\mathcal{A}_{prod}$ and $\mathcal{A}_1$ and during another — $\mathcal{A}_{prod}$ and $\mathcal{A}_2$ (it is not technically possible to compare $\mathcal{A}_1$ and $\mathcal{A}_2$ simultaneously). In the first experiment, $\mathcal{A}_1$ gave +0.75% clicks on events shown to users; in the second, $\mathcal{A}_2$ gave +2.7%, which clearly confirms that $\mathcal{A}_2$ is a better alternative than $\mathcal{A}_1$. Note that all similarity indices having nice properties are in agreement with user preferences. In particular, all unbiased indices rank $\mathcal{A}_2$ higher than $\mathcal{A}_1$. Our results support the importance of using a similarity index with good properties.

Finally, let us mention a recent paper by [36], where a problem of community detection based on cascade data is analyzed. Figure 5 in their paper compares the results obtained using different popular similarity indices. The figure clearly demonstrates that using a biased metric can lead to unpredictable results: the worse algorithm can become the best according to a biased metric.

Conclusion. In this chapter, we have formally defined several requirements for cluster validation indices. The most important and non-trivial ones are monotonicity and constant baseline. While monotonicity reflects the core idea of similarity indices (it requires better values for more similar partitions), constant baseline is less trivial, but equally important, since violating it in practice can lead to unexpectedly biased algorithms. We proved that among the considered non-pair-counting indices, SMI is the only index that has a constant baseline, yet it does not satisfy monotonicity. Among pair-counting indices, these two requirements are satisfied by Adjusted Rand (AR), Sokal&Sneath-1 (S&S1) and Correlation Coefficient (CC). With respect to the proposed requirements, AR is dominated by S&S-1 and CC. The only requirement that is not satisfied by the latter two is distance. CC has the advantage of having an easier interpretation than S&S1. We have shown that CC can be monotonously transformed into a

Correlation Distance (CD) that satisfies all requirements except constant baseline, which is still satisfied asymptotically. Furthermore, the difference between the expected value of CD and its asymptotic value is shown to be negligible for large numbers of items. We advise using CD in applications where a distance is desirable, otherwise we advise using CC.

Chapter 5

Random graphs and random communities

This chapter discusses how to choose a distribution $\mathcal{D}$ so that a detection method working well for $(G, T) \sim \mathcal{D}$ gives confidence that it will work well for the real-world network at hand. We will start by describing some characteristics that are often found in real-world networks.

5.1 Characteristics of real-world networks

Many of the characteristics that are discussed in this section are defined asymptotically as the number of vertices n in the graph tends to infinity. Therefore, we consider a sequence of random graphs G_n with n vertices and a sequence of random clusterings T_n of these same n vertices. Furthermore, let the random variable I_n be chosen uniformly from $[n]$, while the random variable J_n is chosen uniformly from $\{1, \dots, |T_n|\}$. We next describe characteristics that are often observed in real-world networks.

Sparsity. Even though many real-world networks tend to be huge, the average degree in many of them is not so large. Such networks are often referred to as *sparse* networks. This *'not so large'* is an intuitive notion but is usually formalized in the following way:

Definition 17. *A sequence of random graphs G_n is sparse if the average degrees are a bounded sequence, i.e.* $\sup_n \mathbb{E}_{G_n,I_n}[d_{I_n}^{G_n}] < \infty$.

Power-law degrees and community sizes. In many real-world networks, the degrees and community sizes follow a power law [5, 13, 41]. We say that a sequence of random graphs G_n has a power-law degree distribution if the distribution of $d_{I_n}^{G_n}$ converges to a power-law distribution. Similarly, a sequence of random clusterings $T_n = \{T_n^{(1)}, \dots, T_n^{(|T_n|)}\}$ has a power-law size distribution if the distribution of $|T_n^{(J_n)}|$ converges to a power-law distribution.

In [41], it is found that many real-world networks also satisfy two additional power laws: The first one is a power-law relation between the size of a community and the number of edges within this community, where larger communities are less dense than smaller communities. The second is a power-law relation between the size of a community and the number of edges leaving this community. These power laws were originally described in the following way: for $t \in T$, it holds that $m_{t,t} \sim |t|^{\alpha+1}$ and $m_{t,[n]-t} \sim |t|^{\beta'}$, where $\alpha \in (0, 1), \beta' \in (1, 2)$ and $\sim$ as described in Section 2.2. However, since for a random graph model the variables $m_{t,t}, m_{t,[n]-t}$ are random, we must take special care in formalizing what these power-law relations mean. We choose to do this by satisfying the power laws in expectation:

Definition 18. *We say that a sequence of random graphs G_n satisfies the* interior power law *with exponent α if*

$$h_{in}(s) := \lim_{n\to\infty} \mathbb{E}\left[m_{T_n^{(J_n)},T_n^{(J_n)}} \middle| |T_n^{(J_n)}| = s\right],$$

exists and $h_{in}(s) = \Theta(s^{\alpha+1})$.

Definition 19. *We say that a sequence of random graphs G_n satisfies the* exterior power law *with exponent β if*

$$h_{out}(s) := \lim_{n\to\infty} \mathbb{E}\left[m_{T_n^{(J_n)},[n]-T_n^{(J_n)}} \middle| |T_n^{(J_n)}| = s\right],$$

exists and $h_{out}(s) = \Theta(s^{\beta+1})$.

Note that we deviate from the original definition by using $\beta = \beta' - 1$ to make sure that the two characteristics have similar forms.

5.2 Distribution of ground truth clustering

In the framework we described in Chapter 3, we need to assume that the ground truth is drawn T from a certain distribution. We stress that the choice of this distribution will heavily depend on the context. If we were to try to describe a distribution that would be applicable for any context, this would be similar to trying to answer the impossible question 'What are communities?'. However, in many applications, there will be knowledge available about this distribution. For example, when trying to cluster a population based on opinions, the number of communities (i.e. number of opinions) is known beforehand. In other situations, there may be knowledge of the typical size of the communities. As mentioned in the previous sections, real-world networks often have communities with sizes whose distribution seems to have a power law.

Many of the random graph models that are described in the remainder of the chapter take a community structure T as input. Hence, these random graph models can be considered as drawn from the conditional distributions $\mathcal{D}_T$ of the graph given the community structure T. Furthermore, the results described in Chapter 8 allow us to analyze for which combinations of graph and ground truth a detection method performs well. Therefore, assuming an explicit ground truth distribution can often be avoided.

5.3 Popular graph benchmark models

In this section, we will introduce a few random graph models that are commonly used to test the performance of community detection methods.

5.3.1 Girvan-Newman benchmark

One of the first random graph models that was used to compare the performance of community detection methods is the Girvan-Newman benchmark (GN-benchmark, [16]). It is a Planted Partition Model (PPM as defined in Section 2.3.1) with four communities, each consisting of 32 vertices. The PPM-parameters $p_{\text{in}}, p_{\text{out}}$ are chosen such that the average degree is 16 while, in expectation, a fraction μ of the edges are inter-community edges. This is satisfied when choosing

$$p_{\text{in}} = \frac{16}{31}(1-\mu), \quad p_{\text{out}} = \frac{\mu}{6}.$$

This provides a simple benchmark with a single parameter. The disadvantage of this graph model is that it does not have many of the characteristics that real-world networks have. Furthermore, it is small compared to many real-world networks.

5.3.2 Relaxed caveman graphs

One of the simplest benchmark graph models is the *relaxed caveman graph* [49]. Given a community structure T and a parameter $\mu \in [0, 1]$, a random graph is obtained by taking the caveman graph G_T and distorting it in the following way: With probability $1-\mu$ an edge in G_T is left intact while with probability μ it is rewired so that it becomes an inter-community edge.

We change this rewiring slightly and define the relaxed caveman graph as a Hierarchical Configuration Model (HCM, see Section 2.3.2). We generate the community interiors as ER-graphs with probability $1-\mu$ and set the inter-community degree sequences such that each vertex in a community $t \in T$ has total degree $|t| - 1$. In this random graph model, the number of edges will be equal to the number of intra-community pairs, i.e. $p_G = p_T$. Lemma 7 tells us that whenever the community sizes follow a power-law distribution with exponent γ, the degrees will follow a power-law distribution with exponent $\gamma - 1$. Furthermore, the fraction of edges that are inter-community edges will be approximately μ.

5.3.3 Lancichinetti-Fortunato-Radicchi Model

Real-world complex networks often display power-laws in both the degree-distribution as well as the distribution of community sizes [41]. The Lancichinetti-Fortunato-Radicchi (LFR) Model [29] is a random graph model that incorporates these empirical phenomena. The model has the following parameters:

- the number of vertices n,
- a power-law exponent for the vertex degrees τ,
- a power-law exponent for the community sizes γ,
- an average degree λ,
- a mixing rate μ.

It assures that the power laws are satisfied asymptotically, that the average degree is approximately λ while each vertex has approximately a fraction μ of its neighbors outside of its community.

It is generated in the following way: First, a degree sequence is chosen from a distribution that satisfies a power law asymptotically and has mean λ. Then, community sizes are drawn from a distribution that satisfies a power law asymptotically and that asserts that the community sizes sum to n, the minimum community size exceeds the minimum degree and the maximum community size exceeds the maximum degree. After that follows a complicated process that places vertices into communities and places edges between the vertices in order to satisfy all the posed constraints.

We will briefly describe this process: At first, every vertex is 'homeless' (i.e. not assigned to a community). Then vertices are iteratively assigned to communities. A vertex with degree d is assigned to a random community with size at least $(1 - \mu)d$. If the chosen community is already full, then a random vertex in that community will be made homeless again. This will be repeated until each vertex is assigned to a community. After that, edges are placed between vertices in a way to match the degree distribution. Finally, edges are iteratively 'rewired' so that the mixing rate μ is approximated as closely as possible.

Note that there are two parts of this process that could go wrong: Firstly, it may not be possible to assign vertices to communities such that $(1-\mu)d$ exceeds the degree size. Secondly, given the assignment of vertices to communities, it may not be possible to match the mixing rate. The original paper states that for degree-exponent $\tau \in [2, 3]$ and community-size-exponent $\gamma \in [1, 2]$, it is "very unlikely" that the process is unable to satisfy these constraints [29]. We observe that for many other choices of the parameters, this process is often unable to satisfy the constraints.

Another disadvantage of this complicated generation process is that it makes it difficult to analyze it theoretically. For example, it is not clear what dependencies are introduced by this rewiring process.

Difficulty parameter. We note that all described community detection benchmarks have a *mixing rate* parameter μ that determines the expected fraction of edges that will be inter-community edges. When this fraction is large, the margin between the internal and external connectedness will be smaller, making it more difficult to distinguish communities. Therefore, it is useful to think of this mixing parameter as a *difficulty parameter* of the community detection problem. In many studies it is also used as such: the performance of the detection method is plotted against the mixing parameter while keeping other parameters constant.

5.4 How to choose a random graph model

In the previous sections, we have described random graph models and characteristics that real-world networks have. Now we will advise a strategy for choosing a random graph model $\mathcal{D}_T$ for analyzing and benchmarking community detection methods.

Select by characteristics. When performing community detection on a network that is known to have some of the characteristics described in Section 5.1, it would be relevant to know about the performance of the detection method on random graphs with these characteristics. Therefore, we would advise to eliminate the random graph models that do not have these characteristics.

Parameter choices. Such random graph models often require additional parameters. Certain parameters can be extracted from the network at hand. For example, the density or the power-law exponent of the degree sequence can be estimated.

Other parameters, such as the power-law exponent of the community sizes, cannot be extracted from the network. In the absence of other ways to choose these parameters, it may be best to test the robustness of the method w.r.t. these parameters by trying it for various values. In particular, the performance of any community detection method will heavily depend on the value of the mixing parameter μ. Therefore, it will be advisable to test the method on a range of values for this parameter.

Choose for simplicity. Occam's razor would suggest that given a set of characteristics that our network is believed to have and a set of random graph models that are known to have these characteristics, we should choose the *simplest* of these random graph models. The simplicity of a random graph model could be interpreted in multiple ways. For example: on the one hand, when choosing an Erdős-Rényi random graph, we only need to estimate a single parameter. On the other hand, a configuration model will require n parameters (the degree sequence) but these can be set to equal the degrees of the network, avoiding us to make any assumptions about the degree sequence. We would advise to choose for the first interpretation and choose for the graph model for which we have to estimate the least number of parameters.

5.5 Power-Law SBM

In the previous section, we have argued that a graph model should be chosen that has all relevant characteristics while having a minimal number of parameters. With this strategy in mind, we are now going to construct a random graph model that satisfies many of the characteristics described in Section 5.1 while having a small number of parameters. We will name our random graph model *Power-Law SBM (PLSBM)* and it is a specific case of a Stochastic Block Model as described in Section 2.3.1. In particular, if the sizes of the blocks (the communities) are drawn from a power-law distribution, then the degree distribution will also have a power law. For this random graph model, we will introduce two parametrizations. The first will have more freedom while the second (SPLSBM, *Sparse PLSBM*) will satisfy all characteristics described in Section 5.1.

Definition 20. *Let T be a clustering with $|T| > 1$, $\alpha \in (0, 1], \beta \in (0, 1)$ and $x_{in}, x_{out} > 0$. We define* Power-Law Stochastic Block Model *to be the graph model PLSBM$(T, \alpha, x_{in}, \beta, x_{out})$ which is a Stochastic block model with the block matrix $B = (B_{t_1,t_2})_{t_1,t_2 \in T}$ given by*

$$B_{t_1,t_2} = \begin{cases} x_{in}\frac{|t_1|^\alpha}{|t_1|-1} & \text{if } t_1 = t_2, \\ x_{out}\xi_\beta(T)\frac{(|t_1|\cdot|t_2|)^\beta}{\sum_{t\in T}|t|^{\beta+1}} & \text{else.} \end{cases}$$

Here, $\xi_\beta(T)$ is a correction factor given by

$$\xi_\beta(T) = \left[1 - \sum_{t\in T_n}\left(\frac{|t|^{\beta+1}}{\sum_{t'\in T_n}|t'|^{\beta+1}}\right)^2\right]^{-1}.$$

The motivation for this correction factor will become clear in Theorem 7. Often, the Degree-Corrected SBM (as described in Section 2.3.3) is used to model a random graph because it is felt that an uncorrected SBM is unable to incorporate a power-law degree distribution. The following theorem proves the opposite:

Theorem 6. *A sequence of random graphs drawn from PLSBM$(T_n, \alpha, x_{in}, \beta, x_{out})$, where the sizes of T_n have an asymptotic power-law distribution with exponent $\gamma > 2 + \beta$, has the following characteristics:*

1. *The* interior power law *as given by Definition 18 is satisfied with exponent α.*
2. *The* exterior power law *as given by Definition 19 is satisfied with exponent β.*
3. *The degree distribution has an asymptotic power-law distribution with exponent*

$$\tau = 1 + \frac{\gamma - 2}{\max\{\alpha, \beta\}}.$$

Proof. To prove that the model satisfies the first two requirements, we simply compute the expectation and take the limit. Let $t \in T_n$, then

$$\mathbb{E}[m_{t,t}] = \binom{|t|}{2} B_{t,t} = \frac{x_{\text{in}}}{2}|t|^{\alpha+1} = \Theta(|t|^{\alpha+1}),$$

so that the interior power law is indeed satisfied.

Similarly,

$$\begin{aligned}\mathbb{E}[m_{t,[n]-t}] &= |t| \cdot \sum_{t' \in T_n \setminus \{t\}} |t'| B_{t,t'} \\ &= x_{\text{out}}|t|^{\beta+1}\xi_\beta(T_n)\frac{\sum_{t'\in T_n\setminus\{t\}}|t'|^{\beta+1}}{\sum_{t''\in T_n}|t''|^{\beta+1}} \\ &= x_{\text{out}}|t|^{\beta+1}\xi_\beta(T_n)\left[1 - \frac{|t|^{\beta+1}}{\sum_{t''\in T_n}|t''|^{\beta+1}}\right] \\ &= \Theta(|t|^{\beta+1}),\end{aligned}$$

since we have assumed that $\gamma > 2 + \beta$. Hence, the exterior power law is also satisfied.

To prove the remaining characteristic, we note for a vertex $i \in t \in T_n$, the exterior degree (number of neighbors outside c) is asymptotically Poisson distributed with parameter

$$\sum_{t' \in T_n \setminus \{t\}} |t'| B_{t,t'} \overset{n\to\infty}{\longrightarrow} x_{\text{out}}|t|^\beta = \Theta(|t|^\beta),$$

while, for large $|t|$, the interior degree is asymptotically Poisson distributed with parameter

$$(|t| - 1)B_{t,t} = \Theta(|t|^\alpha).$$

The sum of these parameters will give the parameter of the Poisson distribution for the total degree. By Lemma 1, this sum will obey a power law with exponent given by $\max\{\alpha, \beta\}$. Therefore, there is a power-law relation between $|t|$ and the parameter of the total degree distribution with exponent $\max\{\alpha, \beta\}$. Lemma 7 tells us that the asymptotic distribution of $|T_n^{(I_n)}|$ has a power-law with exponent $\gamma - 1$ so that by Lemma 4, the mixing parameter of the mixed Poisson limit of $d_{I_n}^{G_n}$ has an asymptotic power-law distribution with exponent

$$\tau = 1 + (\gamma - 2)/\max\{\alpha, \beta\},$$

so that by Lemma 6, the degree distribution satisfies a power-law with this same exponent asymptotically. □

In [41], the upper bound $\tau \leq \gamma/\alpha - 1$ is derived. It can be seen that our graph model satisfies this upper bound since

$$1 + \frac{\gamma - 2}{\max\{\alpha, \beta\}} \leq 1 + \frac{\gamma - 2}{\alpha} = \frac{\gamma}{\alpha} + 1 - \frac{2}{\alpha} \leq \frac{\gamma}{\alpha} - 1,$$

with equality for $\alpha = 1$, corresponding to *extremely dense communities* [41]. In [41], the exponents of the power laws of some large real-world networks are estimated. When one substitutes the estimated values of γ, α, β for the Gowalla Network (A location-based social network [10]) in the formula for τ, we get

$$\tau = 1 + \frac{\gamma - 2}{\max\{\alpha, \beta - 1\}} = 1 + \frac{0.44}{0.31} \approx 2.42,$$

which is quite close to the value $\tau = 2.48$ that is estimated in [41]. This suggests that this model may actually resemble some real-world networks in certain aspects.

Unfortunately, the parameters $x_{\text{in}}, x_{\text{out}}$ are hard to interpret while they do control the mixing rate and the sparsity of the graph model. This motivates the following alternative parametrization:

Definition 21. *Let T be a clustering with $|T| > 1$, $\tau > 2, \mu \in (0, 1)$ and $\lambda < n$. We define the* Sparse Power-Law Stochastic Block Model *to be the graph model SPLSBM(T, τ, μ, λ) which is a PLSBM with parameters*

$$\alpha = \beta = \frac{\gamma - 2}{\tau - 1}, \quad x_{in} = \lambda(1 - \mu)\frac{n}{\sum_{t\in T_n}|t|^{\alpha+1}}, \quad x_{out} = \lambda\mu\frac{n}{\sum_{t\in T_n}|t|^{\alpha+1}}.$$

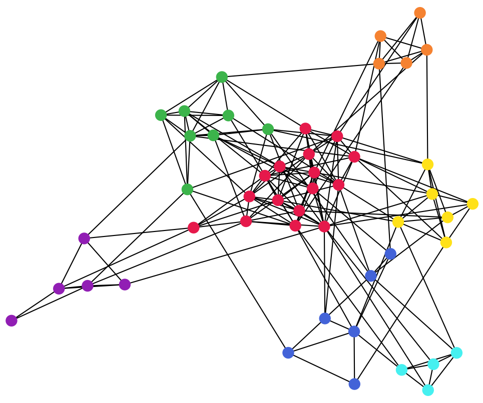


Figure 5.1: A graph drawn from SPLSBM. The cluster-sizes of the ground truth are generated by DFSPL (as given by Definition 4) with $n = 50$ nodes, $K = 7$ communities and exponent $\gamma = 3.5$. The other parameters are: expected degree $\lambda = 7$, mixing rate $\mu = 0.3$ and power-law degree exponent $\tau = 3$.

Figure 5.1 shows an example of a graph generated by this model. We prove the following characteristics:

Theorem 7. *A sequence of random graphs drawn from SPLSBM(T_n, τ, μ, λ), where the sizes of T_n have an asymptotic power-law distribution with exponent $\gamma < \tau + 1$, has the following characteristics:*

1. *All characteristics from Theorem 6 are satisfied.*
2. *The graph is sparse with expected degree λ.*
3. *The mixing rate is μ.*

Proof. First, we note that $\beta = \frac{\gamma-2}{\tau-1} < \gamma - 2$ so that $\gamma > 2 + \beta$ and the first condition of Theorem 6 is satisfied. We rewrite

$$x_{\text{in}}(n) = \lambda(1-\mu)\frac{n}{\sum_{t\in T_n}|t|^{\alpha+1}} = \lambda(1-\mu)\frac{|T_n|\mathbb{E}[|t_{J_n}|]}{|T_n|\mathbb{E}[|t_{J_n}|^{\alpha+1}]} = \lambda(1-\mu)\frac{\mathbb{E}[|t_{J_n}|]}{\mathbb{E}[|t_{J_n}|^{\alpha+1}]},$$

and similarly,

$$x_{\text{out}}(n) = \lambda\mu\frac{\mathbb{E}[|t_{J_n}|]}{\mathbb{E}[|t_{J_n}|^{\alpha+1}]}.$$

Since the sizes of T_n converge to a power-law distribution with exponent $\gamma > 2 + \alpha$, the parameters $x_{\text{in}} = x_{\text{in}}(n), x_{\text{out}} = x_{\text{out}}(n)$ converge to some $x^*_{\text{in}}, x^*_{\text{out}}$. Therefore, this SPLSBM will correspond to a PLSBM with these limiting parameters in the limit so that the asymptotic results from Theorem 6 apply.

To prove the last two characteristics, we compute the expected number of intra-community edges and inter-community edges. The expected number of intra-community edges is given by

$$\sum_{t\in T_n}\mathbb{E}[m_{t,t}] = \sum_{t\in T_n}\frac{x_{\text{in}}}{2}|t|^{\alpha+1} = \sum_{t\in T_n}\frac{\lambda(1-\mu)n|t|^{\alpha+1}}{2\sum_{t'\in T_n}|t'|^{\alpha+1}} = \frac{\lambda(1-\mu)n}{2}.$$

The expected number of inter-community edges can be computed as

$$\begin{aligned}\frac{1}{2}\sum_{t\in T_n}\mathbb{E}[m_{t,[n]-t}] &= \frac{1}{2}\sum_{t\in T_n}x_{\text{out}}\xi_\alpha(T_n)|t|^{\alpha+1}\frac{\sum_{t'\in T_n\setminus\{t\}}|t'|^{\alpha+1}}{\sum_{t''\in T_n}|t''|^{\alpha+1}}\\ &= \frac{\lambda\mu n}{2}\xi_\alpha(T_n)\sum_{t\in T_n}\frac{|t|^{\alpha+1}}{\sum_{t'\in T}|t'|^{\alpha+1}}\left[1-\frac{|t|^{\alpha+1}}{\sum_{t''\in T_n}|t''|^{\alpha+1}}\right]\\ &= \frac{\lambda\mu n}{2}\xi_\alpha(T_n)\left[1-\sum_{t\in T_n}\left(\frac{|t|^{\alpha+1}}{\sum_{t'\in T_n}|t'|^{\alpha+1}}\right)^2\right]\\ &= \frac{\lambda\mu n}{2}.\end{aligned}$$

Hence, the expected degree is

$$\frac{2}{n}\left[\frac{\lambda(1-\mu)n}{2}+\frac{\lambda\mu n}{2}\right]=\lambda,$$

proving the sparsity. The fraction of edges that are inter-community edges is then

$$\frac{\lambda\mu n}{2}\Big/\frac{\lambda n}{2}=\mu,$$

as required. □

To summarize, we have constructed two variants of the same model. The difference lies in the fact that in the first model, the density-exponent can be freely chosen from the out-degree exponent while in the second model we can easily specify a mixing rate, average density and degree exponent. Both models incorporate all the characteristics of LFR with the benefit of being more easily generated and analyzed.

However, since both models are a subclass of the SBM, they are unable to model any inhomogeneity within communities. This is in contrast to LFR and many real-world networks. Within a community in the SPLSBM, the degrees will have a variance equal to its expected value so that there will also be a power-law relation between the community size and the variance of the degrees within this community. The exponent of this relation will be given by $\max\{\alpha,\beta\}$.

Although we did not investigate this empirically, we do not expect this power-law relation to hold for real-world networks. However, investigating how the community-sizes relate to the variance of the degrees within these communities would provide useful insight in how these models could be extended to better resemble real-world networks. Given a description of this relation, the desired inhomogeneity could be incorporated by extending the model to a Degree-Corrected SBM (DCSBM, recall Section 2.3.3).

Chapter 6

Quality functions

In the literature, there are many functions to measure how well a clustering 'fits' a graph. These functions are called *quality functions*. In this chapter we will discuss the most popular quality functions and introduce the concept of using pair-counting functions (like the ones discussed in Chapter 4) as quality functions.

6.1 Quality maximization

Many community detection methods define some quality function which they maximize to obtain a candidate clustering [8, 35]. The quality function that is chosen is often a heuristic that measures how well the clustering 'fits' the graph. For these functions, it is usually reasonable to assume that the ground truth clustering will have a high value w.r.t. this quality function. However, we may also expect that there exist clusterings whose quality *exceeds* that of the ground truth. It is observed that this often occurs in practice [35]. Therefore, it is not entirely clear why such quality functions should be optimized. At any rate, it would be desirable to find a candidate C that satisfies

$$Q(G, C) \geq Q(G, T).$$

That is, our candidate has at least as much 'quality' w.r.t. the graph as the ground truth. In practice T is unavailable so that the value $Q(G, T)$ is also unavailable. This, together with the idea that a quality maximizer will likely be similar to the ground truth, can provide justification for maximizing Q, though it will likely result in finding a clustering with higher quality than the ground truth.

Overfitting. We note that this is similar to the problem of *overfitting* that is often encountered in Statistical Learning Theory (SLT) and Machine Learning. The objective of SLT is to obtain a function that is able to classify data points into classes. Such *classifier* is obtained by 'training' a function on a number of data points (examples): given a set of data points and the corresponding correct classes, a classifier is selected that *fits* the data best according to some *loss* function (e.g. the fraction of data points that it classified wrongly). However, the classifier that scores best on these examples is not necessarily the best classifier overall. The classical example is a classifier that will return the correct classes for all examples it has been trained on while classifying all data points that did not occur in the examples into a single class. This classifier perfectly performs on the examples while it performs poorly in general. We then say that we have been *overfitting* the classifier to the data. The root of this problem is usually identified as an incorrect balance between the complexity of the classifier and its fit to the examples. In SLT, this problem is overcome by adding a term to the loss function that penalizes complexity. In Section 6.2.1, we will show that a popular extention of modularity can be seen as an example of such penalized quality function.

6.2 Modularity

The Newman-Girvan modularity measures the quality of a clustering with respect to a graph [33]. Given a clustering C, it compares the density inside the clusters of C to the expected density given a random graph model without community structure. The original Newman-Girvan modularity uses the

Configuration Model (CM, see Section 2.3.2) as its null model. Therefore, we will denote it by Q_{CM} and refer to it as CM-modularity. It is given by

$$Q_{CM}(G,C) = \sum_{c \in C} \left[\frac{m_{c,c}}{m} - \left(\frac{2m_{c,c} + m_{c,[n]-c}}{2m} \right)^2 \right].$$

This corresponds to the fraction of edges that are intra-cluster edges w.r.t. the given clustering C minus the expected number of intra-cluster edges in C w.r.t. a Configuration Model (CM, see Section 2.3.2) with the same degree sequence as the original graph. Hence, $Q_{CM}(G,C) = 0$ means that there are no more edges within the communities than we would expect when connecting vertices at random, by which we can conclude that the clustering C is not the community structure of G. High values correspond to clusterings that represent the graph well. In real-world networks, CM-modularity maxima are typically found in the interval $(0.3, 0.7)$ [33]. Currently, CM-modularity is the most popular quality measure and its application has had many successes[17].

In [14], some limitations of CM-modularity are discussed. For example, let a community c have an average mixing rate $\mu_c = m_{c,[n]-c}/(2m_{c,c} + m_{c,[n]-c})$, i.e. a fraction μ_c of the half-edges in this community are of intra-community edges. For a community $c \in C$ to have a positive contribution to the modularity, the number of intra-edges must satisfy the upper bound

$$m_{c,c} < (1 - \mu_c)^2 m.$$

This implies that if the ground truth contains a community that does not satisfy this inequality, a community detection algorithm that maximizes CM-modularity will split this community up into smaller communities.

6.2.1 Resolution limit

A more severe limitation of CM-modularity is the so-called *resolution limit.* Let there be at least three communities and consider two communities both having ℓ internal edges, single edges between them and let each community also be connected to the rest of the graph with a single edge. The modularity gain of considering these two communities as a single community is given by

$$\frac{1}{m} - 2\left(\frac{2\ell + 2}{2m}\right)^2.$$

This is positive when

$$\ell + 1 < \sqrt{\frac{m}{2}}.$$

Hence, if we have two communities that are connected to each other but clearly distinct, then, for a sufficiently large number of edges in the remainder of the graph, CM-modularity will be increased by merging the communities. This result is often summarized by the statement that modularity is unable to distinguish communities with a number of edges smaller than $\sqrt{m/2}$. This has very counter-intuitive implications: suppose we have a ‘ring’ of equal-sized cliques, each having only one connection to its neighboring cliques in the ring. Then, for a sufficiently large ring, modularity optimization would merge neighboring communities.

Often, a *resolution parameter* is added to control the resolution limit [38]. CM-modularity becomes

$$Q_\gamma(G,C) = \sum_{c \in C} \left[\frac{m_{c,c}}{m} - \gamma \left(\frac{2m_{c,c} + m_{c,[n]-c}}{2m} \right)^2 \right].$$

This modularity is then able to distinguish communities with up to $\sqrt{m/(2\gamma)}$ internal edges.

Recall the example in Chapter 1, where modularity optimization with resolution parameter $\gamma = 0.7$ was able to perfectly distinguish Chilean airports form Argentinian airports. If we would take a third country into account that has a very large number of airports (e.g. the United States as in Figure 6.1), then modularity optimization would put Chile and Argentina into one big community because of this resolution limit.

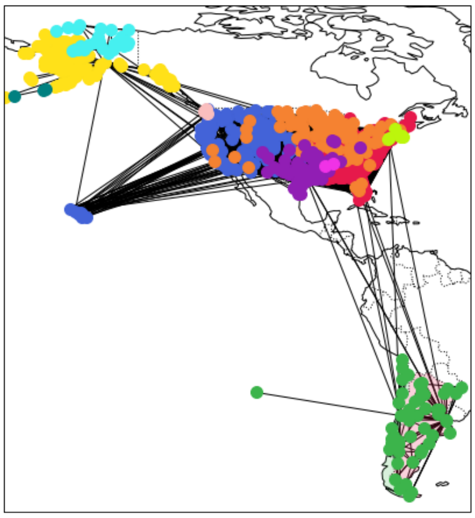


Figure 6.1: Flight routes between Chile, Argentina and the USA from 2014 are visualized on the map. Communities are found by the same method as used in Figure 1.2: the Louvain algorithm for maximizing CM-modularity with resolution 0.7. The data was obtained from `openflights.org`. We see that Chile and Argentina are put into a single community as a result of the resolution limit of CM-modularity.

CM-modularity as penalized quality function. In Section 6.1, we discussed the concept of adding a penalty term to the quality function to control the complexity of quality maximizers. We will now show that modularity can be viewed as an example of such penalized quality function where the resolution parameter γ controls the importance of complexity. First consider the following lousy quality function:

$$Q_0(G, C) = \frac{1}{m} \sum_{c \in C} m_{c,c},$$

i.e. the fraction of edges of G that are within clusters of C. Regardless of G, this is maximized by the clustering $\{[n]\}$. To avoid this undesirable and trivial result, we penalize its complexity. The Gini-Simpson index [40] of a distribution $p_1, \ldots, p_\ell$ is a measure of complexity and is given by

$$GS(p_1, \ldots, p_\ell) = 1 - \sum_{i=1}^{\ell} p_i^2.$$

It corresponds to the probability of two independent random variables from this distribution being unequal. Let C be a clustering and define for $c \in C$ the distribution

$$p_c = \frac{2m_{c,c} + m_{c,[n]-c}}{2m}.$$

This p_c corresponds to the probability that the vertex of a randomly chosen half-edge is in cluster c. Let us denote the complexity of C w.r.t. G by $GS(G, C)$. We multiply this complexity term by $\gamma > 0$ and add it to Q_0 to obtain

$$Q_{GS}(G, C) = Q_0(G, C) + \gamma GS(G, C) = \sum_{c \in C} \frac{m_{c,c}}{m} + \gamma \left[1 - \sum_{c \in C} \left(\frac{2m_{c,c} + m_{c,[n]-c}}{2m} \right)^2 \right] = \gamma + Q_\gamma(G, C).$$

This shows that maximizing Q_γ is equivalent to maximizing Q_0 with penalty term given by the Gini-Simpson complexity. This justifies viewing the resolution parameter γ as a parameter that controls the complexity of the clustering that will be obtained from quality-optimization.

6.2.2 Modularity of a random graph

Even though, for a given clustering C, a random CM graph G will have zero CM-modularity in expectation, the modularity maximum of this random graph may still be high because of random fluctuations in the graph density [19]. From this we can conclude that finding a high modularity maximum in a network is not even a guarantee that this network actually *has* community structure. Hence, after having found a modularity maximum of a network, it may be useful to repeat the optimization process on a graph generated by CM (with the same degrees) and compare the values of the modularity maxima to validate whether the given network actually has community structure.

6.2.3 Modularity maxima

In [17], it is shown that in practice, there exists an exponential number of clusterings with modularity values close to the maximum. These clusterings may be structurally dissimilar. For example, they may disagree on the distribution of the cluster sizes. The intuition of this phenomena is as follows: In many large networks, there exists many sets of nodes between which the number of edges is very close to the expected number of edges. In these cases, merging these sets into a single cluster or splitting these in two will not affect the modularity much while it will affect the distribution of the cluster-sizes.

Furthermore, [17] also proves that when repeatedly adding new communities that are only sparsely connected to other communities, the CM-modularity value of the 'true' clustering will converge to some constant while the CM-modularity maximum will converge to 1. The cause of this is that the resolution limit allows us to increase CM-modularity by merging sparsely connected communities into bigger clusters. From this, we can conclude that even though we may expect the true clusterings to have high CM-modularity, it might not *maximize* CM-modularity. This puts into question whether CM-modularity maximization is the best way to find the community structure in a network.

6.3 Pair-counting quality functions

Recall the definitions of $\vec{b}_G, \vec{b}_C$ given in Section 2.1. Since we are mapping both graphs and clusterings to a binary vector-space, we are also able to compare these entities to each other in this space. A nice subclass of functions that are able to compare such binary vectors is given by the class of pair-counting functions as given in Definition 5. In this section we will discuss using such pair-counting functions as quality functions. We will start by showing that a popular quality function is actually a pair-counting function and then we will discuss whether the Correlation Coefficient (which is, according to Chapter 4, a desirable validation index) would be an appropriate quality function.

6.3.1 Erdős-Rényi modularity

As defined in Section 6.2, modularity is given by the difference between the number of edges within the clusters and its expectation given a null model without community structure. Newman-Girvan modularity uses a Configuration Model as null model, but other random graph models are also possible. A popular variation of Newman-Girvan modularity is the Constant Potts Model [43]. This modularity uses an Erdős-Rényi random graph as null model. Therefore, we will refer to it as ER-modularity to avoid confusion. For an $ER(n,p)$ null model, ER-modularity is given by

$$Q_{ER}^{(p)}(G,C) = \frac{1}{m}\sum_{c\in C} m_{c,c} - p\binom{|c|}{2} = \frac{p_{GT} - p_C p}{p_G}.$$

Often, the parameter p of the ER-model is chosen to be the density of the graph, i.e. $p = p_G = m/N$. In that case, we have $Q_{ER}^{(p_G)}(G,C) = p_{GC}/p_G - p_C$. The following result relates this optimization function to the Rand index:

Lemma 10. *Optimizing ER-modularity with $p = \frac{1}{2}$ is equivalent to optimizing the Rand index between the graph and the clustering.*

Proof. Since the graph G remains constant throughout the optimization, we might as well multiply the modularity by $2p_G$ and add $1 - p_G$, resulting in

$$2p_G Q_{ER}^{(1/2)}(G,C) + 1 - 2(1-p)p_G = 1 + 2p_{GT} - p_C - p_G,$$

which equals the Rand index between G and C. □

We note, however, that a density of $p = \frac{1}{2}$ is very unrealistic in real-world graphs, even inside the communities.

6.3.2 Correlation between graph and clustering

In the previous section we have shown that, for the right choice of parameters, a well-known quality function coincides with a well-known validation index. This motivates us to also look in the other direction: In Chapter 4, we have concluded that the Correlation Coefficient (CC) is a validation index that has properties that are desirable in many situations. This sparks the question whether it would also be suitable as a quality function. We will start by an example which will explore the interpretation of a correlation between a graph and a clustering.

Friendship and opinions. Consider a social network where people are connected if they are friends. We define friendship as having had a one-on-one conversation recently that lasted at least one hour. Consider surveying these people on a number of divisive topics. For example, what the best beer is, what the best soccer club is and what political party to vote for.

On the one hand, it may occur that during this hour-long conversation one of these topics was discussed leading to one person changing his opinion to agree with the other (i.e. the network affects the divisions). On the other hand, people may prefer to converse with people with whom they agree on many subjects and may cut conversations short after they have come across too many disagreements (i.e. the divisions affects the network). Without diving further into the question of causation, we would, at any rate, expect there to be some positive correlation between whether two people are friends and whether they agree on some topic. Hence, the correlation between the network and the division says something about how well we can predict the agreement on a topic from the graph connectivity.

Correlation maximization. Although this does explain that the clusterings that we want to find will have a positive correlation with the graph, it does not directly justify looking for the clustering that maximizes this quantity. As described in Section 6.2.3, modularity maximization has a similar problem: there is no reason to assume that the clustering of interest will actually be the global modularity maximum. To justify looking for the correlation-maximum, we refer to Theorem 4, which shows that CC can be monotonously transformed to a distance metric. Hence, maximizing CC will result in finding a clustering that is, in some sense, *closest* to the graph. The performance of this method will then depend on the distance between the graph and the ground truth, which may be quite large. This will be further analyzed in Section 8.3.

In this chapter, we have discussed CM-modularity and its variants. Currently, these are the most popular quality functions. We then introduced the Correlation Coefficient as a possible quality function and compared the optimization of this quantity to optimizing modularity. In Section 8.3, we will further explore correlation as a quality function and use it to obtain a worst-case guarantee on the validation score of the resulting candidate.

Chapter 7

Optimization algorithms

In Chapter 6, we introduced various quality functions that measure how well a clustering fits a graph. In this chapter we will discuss algorithms that can be used to find clusterings that optimize these quality functions. We will mainly focus on the Louvain algorithm [8], because it is the most widely used modularity-optimization algorithm and it is also able to optimize general quality functions [35]. In this chapter, we will denote an algorithm that optimizes Q by $\mathcal{A}_Q$ while we will denote the result of this algorithm when optimizing w.r.t. a graph G by $\mathcal{A}_Q(G)$. We will assume that high values of $Q(G, C)$ denote high quality of C.

7.1 Criteria for optimization algorithms

After having chosen a quality function Q, we need to find an algorithm that is capable of optimizing this function. Ideally, we would want an algorithm that would be able to find the clustering $C^*_Q(G)$ that maximizes $Q(G, C)$ for all possible G. However, since it is notoriously difficult to find the global maximum of a general function in a discrete domain (for modularity, NP-hardness has been proven [9]), it would be foolish to strive for such an algorithm. A first relaxation would be to restrict to optimizing $Q(G, \cdot)$ for $G \sim \mathcal{D}$ (for $\mathcal{D}$ as defined in Chapter 5). This would lead us to desire an algorithm $\mathcal{A}$ that has a low value of

$$\mathbb{E}_{G\sim\mathcal{D}}\left[\max_C Q(G, C) - Q(G, \mathcal{A}(G))\right].$$

Unfortunately, we are unable to compare algorithms on this criterion because the value $\max_C Q(G, C)$ is unavailable and $\mathcal{D}$ can be very complex.

Therefore, we introduce the following criterion, which we can validate numerically for a given $\mathcal{D}$. As described in Section 6.1, it is at least desirable that $Q(G, \mathcal{A}(G)) \geq Q(G, T)$. We say that an optimization algorithm performs *sufficiently* if this goal is achieved and define the *probability of sufficiency* of an algorithm $\mathcal{A}$ w.r.t. a quality function Q and a distribution $\mathcal{D}$ by

$$\text{Sufficiency}_{\mathcal{D},Q}(\mathcal{A}) = \mathbb{P}_{(G,T)\sim\mathcal{D}}(Q(G, \mathcal{A}(G)) \geq Q(G, T)).$$

This value can easily be estimated by repeatedly generating $(G, T) \sim \mathcal{D}$ and testing how often $\mathcal{A}$ performs sufficiently. In Chapter 9, we will estimate this for the popular Louvain algorithm and some choices of $\mathcal{D}$ and Q.

7.2 Louvain algorithm

The Louvain algorithm was originally intended as a modularity optimization algorithm [8] but it was found to be suitable for optimizing various quality functions [35]. The Louvain algorithm is a greedy algorithm that takes a multi-graph as input and outputs a hierarchical clustering, where each deeper level is a refinement of the level above it. Each level of this hierarchy will correspond to a clustering that has a higher quality value than the level below it, so that the top-level clustering will have the highest quality. In the original paper, it is suggested that the hierarchical approach of Louvain would be a suitable way to deal with the resolution limit, since the desired clustering might be found at some level of the hierarchy [8]. We represent a hierarchical clustering as a clustering of clusterings (of disjoint vertex-sets), e.g. the hierarchical clustering $\{\{\{1\}, \{2, 3\}\}, \{\{4\}\}\}$ has the top-level clustering $\{\{1, 2, 3\}, \{4\}\}$ and bottom-level clustering $\{\{1\}, \{2, 3\}, \{4\}\}$.

Louvain is initialized by setting $G_0 = G$ and choosing an initial clustering C_0 (usually a clustering that places each vertex in its own community). Given a clustering C_ℓ and a multi-graph G_ℓ it finds $C_{\ell+1}, G_{\ell+1}$ by going through the following two phases:

- In the first phase, a new clustering C' is obtained. We go through all items $i \in \bigcup_{c\in C_\ell} c$ and see whether we can increase the quality by moving this item to another cluster. If this is possible, we move v to the cluster that gives the highest increase. We repeat this until no more improvement can be achieved by moving single items.
- Then, in the second phase, we obtain the hierarchical clustering $C_{\ell+1}$ from C' as

  $$C_{\ell+1} = \{\{c\}|c \in C'\}.$$

  The multi-graph $G_{\ell+1}$ is then constructed with vertex-set C' and an edge $(C'(i), C'(j))$ for each edge (i, j) of G_ℓ. Hence, a vertex $c \in C'$ will have a number of self-loops equal to the number of internal edges $m_{c,c}^{G_\ell}$ of cluster c in G_ℓ.

This is repeated as long as the first phase is able to improve the quality. When this is no longer possible, the resulting hierarchical clustering is returned.

7.2.1 Optimality guarantees

In this section, we discuss the optimality of a clustering that is obtained by optimizing a quality function Q using Louvain. We consider optimality w.r.t. three operations:

- Merging two clusters.
- Moving a single vertex to a new cluster.
- Splitting a cluster in two.

It is proven that a clustering obtained by Louvain only satisfies optimality w.r.t. merging [44]. That is, merging two clusters will never increase the quality function. However, Louvain can also be applied iteratively: Let C_{top} be the top-level clustering that is obtained by running Louvain on a graph G. We can then run Louvain again with input C_{top}, G to obtain a top-level clustering C'_{top}. Repeating this until convergence (i.e. $C'_{\text{top}} = C_{\text{top}}$), gives the additional guarantee of optimality w.r.t. moving a single vertex [44].

Disconnected communities. Even when applying Louvain iteratively, the optimality w.r.t. splitting cannot be guaranteed. This has consequences for the clusterings that are obtained by Louvain: In [44], it is shown that it may occur that optimizing modularity by Louvain may result in disconnected communities. That is, the subgraph corresponding to the community is disconnected. This can occur in the following way: during the first phase, a vertex i is moved to some community c and in subsequent moves, all neighbors of i in c are moved to different clusters so that i will be disconnected from its community.

7.2.2 Related algorithms

The Leiden algorithm is an improvement upon the Louvain algorithm that incorporates a third phase where the clustering C' is refined in a greedy but random way. For modularity optimization, the randomness in this *refinement* phase provides an additional guarantee that when performing Leiden iteratively, eventually a clustering will be obtained that satisfies optimality w.r.t. splitting [44]. Furthermore, it is also shown that this new algorithm actually runs faster than Louvain. Unfortunately, it remains unclear whether this refinement phase is also able to guarantee optimality w.r.t. splitting for general quality functions.

Numerical experiments in Chapter 9 indicate that the Louvain algorithm has a probability of sufficiency that is very close to 1. In this respect, it does not seem urgent to look for better optimization algorithms such as the Leiden algorithm.

Chapter 8

Triangle inequality bounds for pair-counting distances

In this chapter, we provide a class of community detection methods for which we can obtain a lower bound on the resulting value of the validation index. In practice, the guarantees presented in this chapter will not be strong enough to be helpful. We emphasize that this is a theoretical result and merely intended to demonstrate that within the theoretical framework presented in Chapter 3, it is possible to give guarantees on the performance of detection methods. Section 8.1 will discuss the general theory that applies to any validation index that is a pair-counting distance. Section 8.2 will give an example for a specific validation index while in Section 8.3, we will extend these results to the Correlation Coefficient, which was concluded in Chapter 4 to be a desirable validation index.

8.1 Worst case analysis using triangle inequalities

We will assume that our validation index is a pair-counting function and a distance metric. Let us denote this *validation distance* by d. Recall from Section 6.3 that we can compute pair-counting similarity scores between a graph G and the caveman graph G_C corresponding to the clustering C. Therefore, we represent the validation score by $d(G_T, G_C) := d(T, C)$. By the triangle inequality, we have

$$d(G_T, G_C) \leq d(G_T, G) + d(G, G_C). \tag{8.1}$$

In Section 6.1, we introduced the problem of *overfitting*, where optimizing a quality function results in a candidate whose quality exceeds that of the ground truth. The following assumption uses this problem to the advantage:

Assumption 1. *We assume that our optimization algorithm is good enough to guarantee that the resulting candidate C satisfies $d(G, G_C) \leq d(G, G_T)$.*

There always exists such a candidate since the inequality always holds for $C = T$. This assumption is a relaxation of the assumption that the algorithm always returns the global optimum. Note that in the terminology of Section 7.1, this assumption corresponds to assuming that our algorithm $\mathcal{A}$ has a perfect *probability of sufficiency* w.r.t. $\mathcal{D}$ and $Q = -d$, i.e. we assume $\text{Sufficiency}_{\mathcal{D},-d}(\mathcal{A}) = 1$. In Chapter 9 we will numerically verify this assumption. Applying this assumption to (8.1) results in the following theorem:

Theorem 8. *Under Assumption 1, the validation distance of the candidate C is bounded by*

$$d(T, C) \leq 2d(G, G_T).$$

We note that this bound is a worst-case result: d defines a geometry on the space of graphs and clusterings and we assume that our optimization algorithm finds a candidate clustering inside a ball with radius $d(G, G_T)$ centered around G. The bound that is given in Theorem 8 corresponds to the situation where T and C are on opposite sides of this ball. Furthermore, the r.h.s. of Theorem 8 compares a graph to a clustering while most real-world networks are not close to a clustering at all. Therefore, $d(G, G_T)$ will likely be large, so that this bound will not be a strong result.

8.2 Rand distance

Recall the definitions of p_G, p_T as the densities of G and T given in Section 2.1. The Rand distance (one minus the Rand index, otherwise known as the Mirkin metric) is given by

$$d_R(G, G_T) = (p_G - p_{GT}) + (p_T - p_{GT}) = p_G + p_T - 2p_{GT}.$$

Hence, applying Theorem 8 gives

$$d_R(T, C) \leq 2p_G + 2p_T - 4p_{GT}.$$

However, when taking the clustering $C' = \{\{i\}|i \in [n]\}$ that gives each vertex its own community, we get $d_R(T, C') = p_T$. So, for the bound of Theorem 8 to actually guarantee better performances than this trivial candidate C', we need

$$p_{GT} > \frac{1}{2}p_G + \frac{1}{4}p_T.$$

This can be a very stringent constraint. Take for example the case where all communities have size $k + 1$ and all vertices have degree k. Then the average degree of G_T is k so that $p_T = p_G$ and we get $p_{GT} = \frac{3}{4}p_G$ so that vertices must at least be connected to $\frac{3}{4}k$ members of its community on average.

Or, more generally, for a Planted Partition Model (PPM, recall from Section 2.3.1) with k ground truth communities each of size s, we need

$$p_{\text{in}} > \frac{1}{2} + \frac{(k-1)s}{s-1}p_{\text{out}}.$$

In particular, this implies the stringent conditions

$$p_{\text{in}} > \frac{1}{2} \quad \text{and} \quad p_{\text{out}} < \frac{s-1}{2(k-1)s}.$$

An example of a graph that satisfies these bounds would be a PPM with 10 communities of size 10 with $p_{\text{in}} = 0.8$ and $p_{\text{out}} = 0.02$. We conclude that there exist graph models for which Theorem 8 applied on the Rand distance gives non-trivial bounds, but that these will not correspond to real-world networks.

We note that this triangle inequality result is especially weak because it takes into account the worst-case situation where the set of vertex-pairs on which G disagrees with T is disjoint from the set of vertex-pairs on which C disagrees with G. In practice, two vertices i and j that are not connected but have many common neighbors are likely to be in the same ground truth community and many community detection methods will be able to cluster them together.

8.3 Correlation coefficient

The triangle inequality for the Correlation Distance (CD) allows us to perform a similar analysis. We will formulate the results in terms of the Correlation Coefficient (CC) since its constant baseline at 0 makes them easier to interpret. Theorem 8 gives us

$$\arccos(CC(T, C)) \leq 2\arccos(CC(G, G_T)).$$

We assume $CC(G, G_T) \geq 0$ so that $\arccos(CC(G, G_T)) \leq \frac{\pi}{2}$ and we take the cosine on both sides to get

$$\begin{aligned} CC(T, C) &\geq \cos(2\arccos(CC(G, G_T))) \\ &= 1 - 2\sin^2(\arccos(CC(G, G_T))) \\ &= 1 - 2(1 - CC(G, G_T)^2) \\ &= 2CC(G, G_T)^2 - 1. \end{aligned} \tag{8.2}$$

We summarize the result in the following theorem:

Theorem 9. *Let G, T be a graph and ground truth with densities p_G, p_T respectively. Furthermore, let the graph have an average mixing rate $\mu < 1 - p_T$, i.e. $p_{GT} = (1 - \mu)p_G$. Then, under Assumption 1, it holds that*

$$CC(T, C) \geq 2\frac{(1 - p_T - \mu)^2}{\frac{p_T}{p_G}(1 - p_T)(1 - p_G)} - 1.$$

Proof. Rewriting (8.2) and substituting $p_{GT} = (1-\mu)p_G$ gives the desired result. □

The lower bound given in Theorem 9 is still hard to interpret. It can be seen that better-than-random performance (i.e. $CC(G, G_T) > 0$) is guaranteed whenever

$$\mu < (1-p_T)\left(1-\sqrt{\frac{p_T}{2p_G}\frac{1-p_G}{1-p_T}}\right),$$

But it remains unclear what this lower bound would be for large real-world networks.

Large sparse networks. As mentioned in Section 5.1, many real-world networks are sparse, meaning that $p_G = O(n^{-1})$. Moreover, it is reasonable to assume that G_T is also sparse, for example when the community sizes are drawn from a power-law distribution with exponent $\gamma > 3$. If both p_G and p_T are $O(n^{-1})$, then their *ratio* $r := p_T/p_G$ may be more relevant to describe the behavior for large n. Rewriting the bound of Theorem 9 in these variables results in the following corollary:

Corollary 1. *Let G_n be a sequence of sparse graphs of size n, i.e. $p_{G_n} \to 0$. Let T_n be a sequence of ground truths whose densities are asymptotically proportional to those of G_n, i.e. $p_{T_n}/p_{G_n} \to r$, for some $r > 0$. Furthermore, let the average mixing rate μ_n of G_n converge to μ. Then, the lower bound in Theorem 9 converges, i.e.,*

$$CC(T_n, C_n) \geq 2\frac{(1-p_{T_n}-\mu_n)^2}{\frac{p_{T_n}}{p_{G_n}}(1-p_{T_n})(1-p_{G_n})} - 1 \to \frac{2}{r}(1-\mu)^2 - 1.$$

It can be seen that even for small r this lower bound never exceeds one, since

$$1-\mu = \frac{p_{GT}}{p_G} \leq \frac{p_T}{p_G} = r,$$

so that this lower bound is upper-bounded by $2r - 1$.

Remark 1. *Note that for* Relaxed caveman graphs *as described in Section 5.3.2, it holds that $r = 1$ so that the asymptotic lower bound is given by $2(1-\mu)^2 - 1$. Hence, positive correlation can be guaranteed asymptotically for $\mu < 1 - \sqrt{1/2}$.*

In Chapter 9, we will demonstrate the lower bound of Theorem 9 and show that maximizing correlation actually results in significantly better validation scores than this triangle inequality bound can guarantee. This shows that better guarantees are needed.

Chapter 9

Numerical experiments

In this chapter, we will numerically verify the results that have been obtained throughout this thesis. In particular, we will compare our new community detection method of maximizing CC to the popular community detection method of maximizing CM-modularity. For different random graphs with various characteristics we will assess how these characteristics influence the performance of the detection methods. For both methods, the Louvain algorithm as described in Chapter 7 is used to optimize the quality function.

Chapter 4 suggests that the Correlation Coefficient (CC) is a suitable measure for comparing candidate clusterings to the ground truth, so we will use this validation index to measure the performance of the detection methods. Theorem 9 gives a lower bound for the performance of our detection method. This lower bound will be verified and compared to the actual performance of the detection method. This lower bound relies on Assumption 1, which will also be verified in our experiments.

We start with a simple graph model that does not have many real-world characteristics and progressively move on to more complex graph models that have more real-world characteristics.

9.1 Optimizing CC vs CM-modularity on relaxed caveman graphs

Our results from Corollary 1 suggest that optimizing Correlation Coefficient as a quality function will work well whenever p_T is relatively small compared to p_G. To test this, we will generate networks with $p_G = p_T$ so that $r = p_T/p_G = 1$. *Relaxed caveman graphs*, as described in Section 5.3.2, satisfy this condition. This random graph model requires two parameters: a ground truth and a mixing rate. We will consider two choices for the ground truth and vary the mixing rate for each of them.

Relaxed caveman with equal-sized communities. For the first comparison we will choose a ground truth consisting of 14 communities, each containing 7 vertices. The performances of the two competing community detection methods are shown in Figure 9.1.

We see that, even though CC optimization slightly outperforms modularity optimization, the performances of both detection methods are comparable. Furthermore, it is also visible that CC optimization performs significantly better than the lower bound guaranteed by Theorem 9.

Relaxed caveman with power-law-sized communities. Suppose the only knowledge of our network is that it has n vertices, K communities and the degree sequence has a power-law distribution with exponent τ. Then, the strategy as described in Section 5.4 would prescribe to choose a relaxed caveman graph with k communities of sizes drawn from a power-law distribution with exponent $\tau + 1$ (e.g. by generating it from DFSPL as defined in 4) so that the degree sequence has the desired power-law exponent. In Figure 9.2, the performance of the two community detection methods is shown for varying μ.

We see that modularity optimization works worse than CC optimization for power-law community sizes. This can be explained by the fact that for power-law sizes there will be a large amount of small communities that are combined into bigger clusters as a result of the resolution limit (described in Section 6.2.1).

These two experiments confirm that CC optimization indeed performs well when the ground truth and the graph have similar densities.

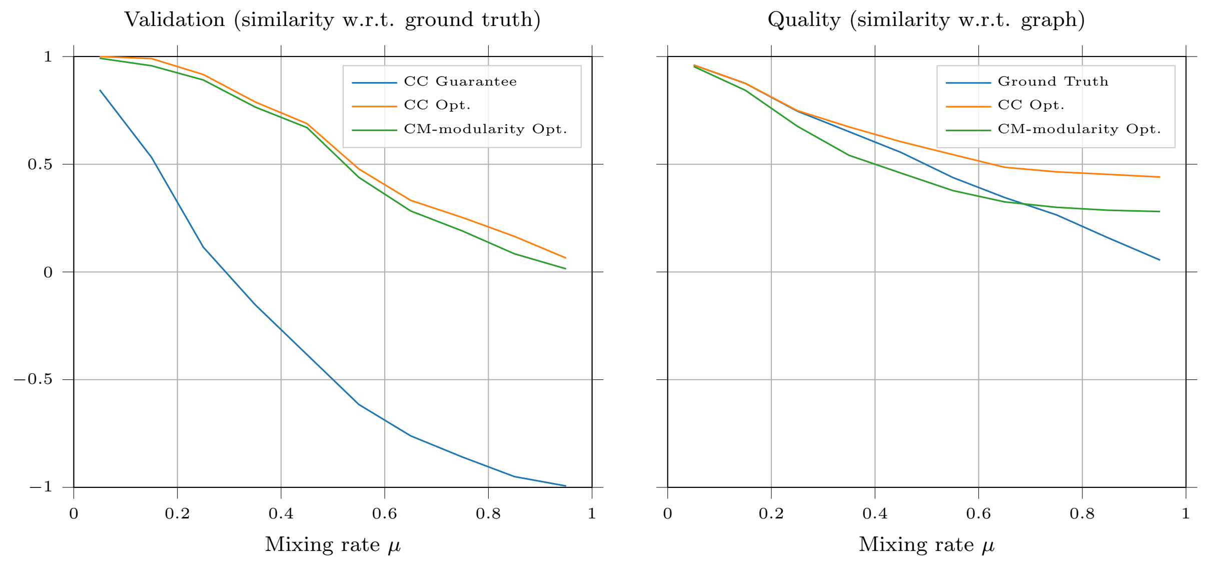


Figure 9.1: For relaxed caveman graphs with 14 communities each of size 7, the performance of CC optimization is compared to the performance of CM-modularity optimization. In both cases, the Louvain algorithm was used for optimization and CC was used as validation index. The left plot shows the correlation to the ground truth while the right plot shows the correlation to the graph.

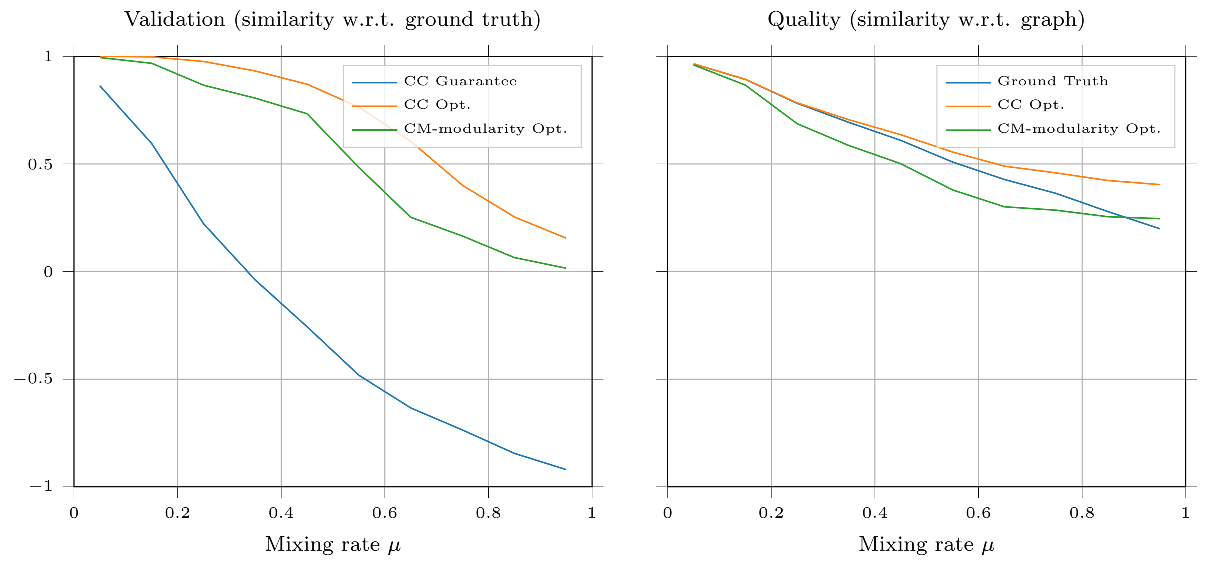


Figure 9.2: For relaxed caveman graphs with ground truth generated by DFSPL with 14 communities and power-law exponent $\gamma = 3.5$, the performance of CC optimization is compared to the performance of CM-modularity optimization. In both cases, the Louvain algorithm was used for optimization and CC was used as validation index. The left plot shows the correlation to the ground truth while the right plot shows the correlation to the graph.

9.2 Optimizing CC vs CM-modularity on SPLSBM graphs

Next, we will create a graph that satisfies all characteristics mentioned in Section 5.1. We will do so by using the SPLSBM model as defined in Definition 21. To inspect the influence of the interior and exterior power laws in isolation (Definitions 18 and 19) we will make sure that on all other characteristics, the graph model will coincide with the previous experiment. Hence, we will have $r \approx 1$ just as in relaxed caveman graphs. The performances of the two methods on this random graph model are shown in Figure 9.3.

We see that for both methods, the performance is worse than for the relaxed caveman graphs with power-law-sized communities. For CC optimization, the degradation in performance is bigger so that on this random graph model the performances of both methods are comparable again (though CC

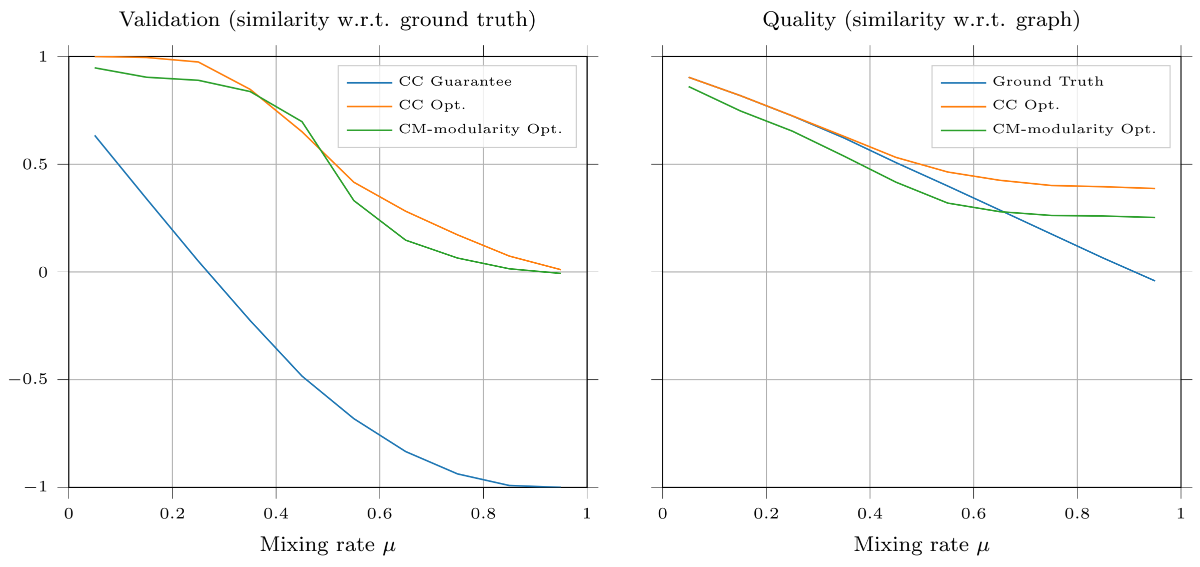


Figure 9.3: For SPLSBM graphs with power-law degrees (asymptotically) with exponent $\tau = 3$, ground truth generated by DFSPL with 14 communities and power-law exponent $\gamma = 3.5$ and density such that $r \approx 1$, the performance of CC optimization is compared to the performance of CM-modularity optimization. In both cases, the Louvain algorithm was used for optimization and CC was used as validation index. The left plot shows the correlation to the ground truth while the right plot shows the correlation to the graph.

optimization performs slightly better for most μ). This performance degradation may be explained by the plot on the right-hand side: this shows that the correlation between the ground truth and the generated graph is significantly lower for this type of graph compared to the previous experiment.

Higher values of r. Finally, we decrease the number of ground truth communities so that the density of the ground truth doubles (approximately) and we will have $r \approx 2$. This will allow us to assess the vulnerability of both methods to this quantity. We keep all other parameters unchanged. The results are shown in Figure 9.4.

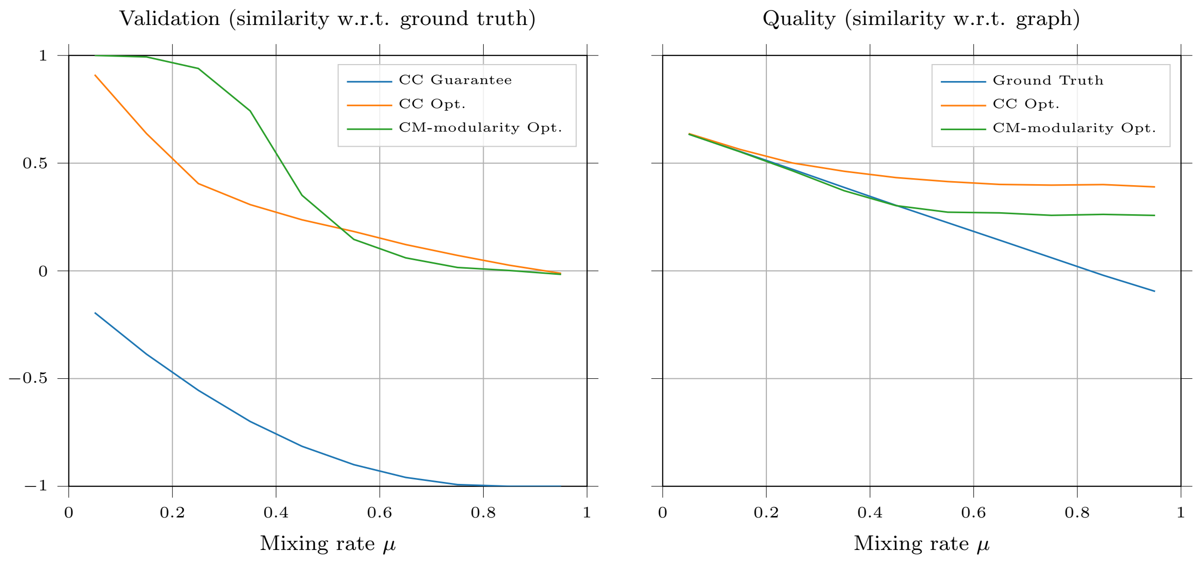


Figure 9.4: For SPLSBM graphs with power-law degrees (asymptotically) with exponent $\tau = 3$, ground truth generated by DFSPL with 7 communities and power-law exponent $\gamma = 3.5$ and density such that $r \approx 2$, the performance of CC optimization is compared to the performance of CM-modularity optimization. In both cases, the Louvain algorithm was used for optimization and CC was used as validation index. The left plot shows the correlation to the ground truth while the right plot shows the correlation to the graph.

From Corollary 1 we know that the lower bound of CC optimization decreases roughly like r^{-1}. We see that the guaranteed lower bound is significantly lower than in the previous experiments and see that the lower bound never exceeds 0. This is also reflected by the performance of CC optimization, which is significantly worse than for the previous experiments. However, the performance still significantly exceeds the guaranteed lower bound and remains to be better than random guesses (i.e. positive correlation). Interestingly, the performance of modularity optimization improves such that it clearly outperforms CC optimization in this experiment. However, for $\mu > 0.5$, CC optimization performs slightly better than modularity optimization again.

In all experiments with power-law-sized communities, we observe a sharp decrease in the performance of modularity optimization around $\mu = 0.5$ while this decrease is more subtle for CC optimization. This suggests that modularity is more vulnerable w.r.t. this parameter than our method.

Resolution parameter and r. These four experiments show that CM-modularity optimization is more suitable for situations where $r \approx 2$ than $r \approx 1$. It is likely that the performance of CM-modularity in the $r \approx 1$ experiments could have been improved by increasing modularity's resolution parameter. It would be interesting to investigate the relation between the value $r = p_T/p_G$ and the best-performing value of the resolution parameter. This might give useful insight into the way we should interpret and choose the resolution parameter.

Adapting CC to $\mathcal{D}$. This sparks the question whether we could parametrize CC-quality in analogy to the parametrization of modularity by the resolution parameter in order for this adapted quality function to perform better for higher values of r. At this point, the most promising approach to achieve this, is to replace the binary vector-representation $\vec{b}_G$ by a different vector $\vec{v}(G) \in [0,1]^N$ in a way that it has a high (linear) correlation to $\vec{b}_T$. To achieve this, we will need $\vec{v}(G)$ to have density (i.e. average value of the entries $\langle \vec{v}(G), \mathbf{1} \rangle / N$) close to p_T for $(G,T) \sim \mathcal{D}$. This would allow for the possibility to tune the quality function (given by the linear correlation between $\vec{v}(G)$ and $\vec{b}_C$) to the distribution, in a way that we can optimize the lower bound given by Theorem 8. This approach is outside the scope of this thesis but may be attempted in future research.

Worst case guarantee. Note that CC optimization always performs significantly better than the guaranteed lower bound. Moreover, CC optimization continues to give better-than-random results even for higher values of μ than our results can guarantee. This shows that the triangle-inequality bound is merely a worst-case guarantee that does not reflect its performance in practice. However, this worst-case guarantee was able to give us insight in the vulnerability of the method w.r.t. r. More sharp guarantees may be even more helpful in understanding for which kinds of networks optimizing CC is a suitable approach. Furthermore, similar analysis for modularity optimization would be equally helpful in assessing when this method is and is not appropriate.

Sufficiency of Louvain optimization. We will now address the verification of Assumption 1. For each of the four plots in this chapter and each of the 10 values of μ, we have generated 20 random graphs and used Louvain to find a clustering that optimizes CC for each of these. In all these $4 \times 10 \times 20 = 800$ runs of the algorithm, not once was Assumption 1 violated. That is, for all pairs G, T that were generated and the candidate clustering C obtained by Louvain optimization, it holds that

$$CC(G,C) \geq CC(G,T).$$

This is also visible in the right-hand plots of Figure 9.1, 9.2, 9.3 and 9.4. This suggests that the *sufficiency* (as defined in Section 7.1) of the Louvain algorithm for optimizing CC w.r.t. the distributions that were considered is close to 1. This gives confidence that this assumption will also hold for real-world networks.

Conclusion We have compared our community detection method to the state-of-the-art community detection method of CM-modularity maximization. We see that it outperforms modularity maximization whenever $p_G \approx p_T$ while it performs significantly worse than this method when $p_T > p_G$. We see that the theoretical lower bound guaranteed by Theorem 9 holds and that its dependency on p_T/p_G is reflected in its actual performance, even though the actual performance is way better than this worst-case guarantee. This shows that even though this guarantee is not strong, it is still helpful in obtaining insight in the cases where the method works well. Obtaining similar guarantees for other community detection methods would be very helpful in choosing a suitable method for a given application.

Chapter 10

Conclusions and extensions

The main message of this thesis is that the field of community detection is in dire need of theoretical formalism. Chapter 3 makes a humble start at this formalization and distinguishes the four main *ingredients* that are required for performing community detection: 1) *a joint distribution* that describes the networks that can occur in the given setting and its relation to its community structure, 2) a *validation index* that measures how similar a candidate is to the ground truth, 3) a *quality function* that measures how well a candidate fits the network and 4) an *optimization algorithm* that finds a good candidate according to this quality function.

In Chapter 4, we concluded that for general applications, Correlation Coefficient is the most appropriate validation index for comparing candidate community structures to the actual ground truth community structure.

In Chapter 5, we argue that for assessing the performance in a given setting, we should model the network by a random graph that has similar characteristics as the given network. A number of characteristics that are often found in real-world networks are described in Section 5.1 while a random graph model that satisfies all of them is introduced in Section 5.5.

To demonstrate the framework that was proposed in Chapter 3, we introduced the idea of finding a candidate community structure by optimizing the correlation between the graph and candidate community structure. This allowed us in Chapter 8 to guarantee a lower bound for the worst-case performance of this method under an assumption that was empirically validated. The value of this lower bound could asymptotically be expressed in two parameters: the fraction μ of edges that connect vertices within the same community and the proportion r between the number of intra-community pairs and the number of connections in the network.

The experiments that were performed in Chapter 9 show that the method performs significantly better than this worst case guarantee. However, the lower bound does predict the vulnerability of the method to the value of r. The method is also compared to modularity optimization, the community detection method that is currently most used throughout the literature. It turns out that for $r \approx 1$, our method outperforms modularity optimization while for $r \approx 2$ modularity optimization works best.

This thesis opens a variety of directions for future research: First of all, the theoretical framework of Chapter 3 could be further explored and extended. Secondly, the method of modularity maximization may be analyzed in a similar way as our newly proposed method to obtain theoretical guarantees, hopefully providing insight in the cases where modularity optimization is appropriate. Thirdly, the random graph model that was introduced in Section 5.5 could be compared to real-world networks and could be extended to match more real-world characteristics. Finally, new quality functions may be designed such that the worst-case guarantee of Theorem 8 provides stronger guarantees. In particular, changing the way we represent a graph as a vector could decrease the 'distance' between the graph and its ground truth clustering so that the worst-case guarantee of Theorem 8 would be more desirable.

We hope that this thesis paves the way for an approach to community detection that relies less on heuristic one-size-fits-all methods, but more on theoretical guarantees that are made based on verifiable assumptions.

Bibliography

[1] Ahmed N. Albatineh, Magdalena Niewiadomska-Bugaj, and Daniel Mihalko. "On Similarity Indices and Correction for Chance Agreement". In: *Journal of Classification* 23.2 (2006), pp. 301–313. ISSN: 1432-1343. DOI: 10.1007/s00357-006-0017-z. URL: https://doi.org/10.1007/s00357-006-0017-z.

[2] Mehdi Allahyari et al. "A brief survey of text mining: Classification, clustering and extraction techniques". In: *arXiv preprint arXiv:1707.02919* (2017).

[3] Alessia Amelio and Clara Pizzuti. "Is normalized mutual information a fair measure for comparing community detection methods?" In: *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*. ACM. 2015, pp. 1584–1585.

[4] Enrique Amigó et al. "A comparison of extrinsic clustering evaluation metrics based on formal constraints". In: *Information retrieval* 12.4 (2009), pp. 461–486.

[5] Albert-László Barabási and Márton Pósfai. *Network science*. Cambridge: Cambridge University Press, 2016. ISBN: 9781107076266 1107076269. URL: http://barabasi.com/networksciencebook/.

[6] Vladimir Batagelj and Matevz Bren. "Comparing resemblance measures". In: *Journal of Classification* 12.1 (1995), pp. 73–90. ISSN: 1432-1343. DOI: 10.1007/BF01202268. URL: https://doi.org/10.1007/BF01202268.

[7] Shai Ben-David and Margareta Ackerman. "Measures of clustering quality: A working set of axioms for clustering". In: *Advances in neural information processing systems*. 2009, pp. 121–128.

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

[9] U. Brandes et al. *Maximizing Modularity is hard*. Sept. 2006.

[10] Eunjoon Cho, Seth Myers, and Jure Leskovec. "Friendship and Mobility: User Movement In Location-Based Social Networks". In: Aug. 2011, pp. 1082–1090. DOI: 10.1145/2020408.2020579.

[11] SHC Choi, Sung-Hyuk Cha, and Charles Tappert. "A Survey of Binary Similarity and Distance Measures". In: *J. Syst. Cybern. Inf.* 8 (Nov. 2009).

[12] Claire Donnat and Susan Holmes. *Tracking network dynamics: a survey of distances and similarity metrics*. 2018. arXiv: 1801.07351 [stat.AP].

[13] Santo Fortunato. "Community detection in graphs". In: *Physics Reports 486, 75-174 (2010)* (June 3, 2009). DOI: 10.1016/j.physrep.2009.11.002. arXiv: http://arxiv.org/abs/0906.0612v2 [physics.soc-ph].

[14] Santo Fortunato and Marc Barthélemy. "Resolution limit in community detection". In: *Proceedings of the National Academy of Sciences* 104.1 (2007), pp. 36–41. ISSN: 0027-8424. DOI: 10.1073/pnas.0605965104. eprint: https://www.pnas.org/content/104/1/36.full.pdf. URL: https://www.pnas.org/content/104/1/36.

[15] Santo Fortunato and Darko Hric. "Community detection in networks: A user guide". In: *Physics Reports* 659 (2016), 1–44. ISSN: 0370-1573. DOI: 10.1016/j.physrep.2016.09.002. URL: http://dx.doi.org/10.1016/j.physrep.2016.09.002.

[16] M. Girvan and M. E. J. Newman. "Community structure in social and biological networks". In: *Proceedings of the National Academy of Sciences* 99.12 (2002), pp. 7821–7826. ISSN: 0027-8424. DOI: 10.1073/pnas.122653799. eprint: https://www.pnas.org/content/99/12/7821.full.pdf. URL: https://www.pnas.org/content/99/12/7821.

[17] Benjamin H. Good, Yves-Alexandre de Montjoye, and Aaron Clauset. "Performance of modularity maximization in practical contexts". In: *Phys. Rev. E* 81 (4 2010), p. 046106. DOI: `10.1103/PhysRevE.81.046106`. URL: `https://link.aps.org/doi/10.1103/PhysRevE.81.046106`.

[18] Martijn Gösgens, Liudmila Prokhorenkova, and Alexey Tikhonov. *Systematic Analysis of Cluster Similarity Indices: Towards Bias-free Cluster Validation.* 2019. arXiv: `1911.04773 [cs.DM]`.

[19] Roger Guimerà, Marta Sales-Pardo, and Luís A. Nunes Amaral. "Modularity from fluctuations in random graphs and complex networks". In: *Phys. Rev. E* 70 (2 2004), p. 025101. DOI: `10.1103/PhysRevE.70.025101`. URL: `https://link.aps.org/doi/10.1103/PhysRevE.70.025101`.

[20] Remco van der Hofstad. "Random graphs and complex networks, Volume 2." In: *Book in preparation* (2018+).

[21] Remco van der Hofstad, Johan SH van Leeuwaarden, and Clara Stegehuis. "Hierarchical configuration model". In: *arXiv preprint arXiv:1512.08397* (2015).

[22] Remco van der Hofstad. *Random Graphs and Complex Networks.* Vol. 1. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2016. DOI: `10.1017/9781316779422`.

[23] Paul W. Holland et al. "Stochastic blockmodels: first steps". In: *Social Networks. An International Journal of Structural Analysis* 5.2 (1983), pp. 109–137. ISSN: 0378-8733. DOI: `10.1016/0378-8733(83)90021-7`.

[24] Lawrence Hubert. "Nominal scale response agreement as a generalized correlation". In: *British Journal of Mathematical and Statistical Psychology* 30.1 (1977), pp. 98–103. DOI: `10.1111/j.2044-8317.1977.tb00728.x`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.2044-8317.1977.tb00728.x`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1111/j.2044-8317.1977.tb00728.x`.

[25] Paul Jaccard. "THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE.1". In: *New Phytologist* 11.2 (1912), pp. 37–50. DOI: `10.1111/j.1469-8137.1912.tb05611.x`. eprint: `https://nph.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1469-8137.1912.tb05611.x`. URL: `https://nph.onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-8137.1912.tb05611.x`.

[26] Brian Karrer and M. E. J. Newman. "Stochastic blockmodels and community structure in networks". In: *Phys. Rev. E 83, 016107 (2011)* (Aug. 23, 2010). DOI: `10.1103/PhysRevE.83.016107`. arXiv: `http://arxiv.org/abs/1008.3926v1 [physics.soc-ph]`.

[27] Sven Kosub. "A note on the triangle inequality for the Jaccard distance". In: *Pattern Recognition Letters* 120 (2019), pp. 36 –38. ISSN: 0167-8655. DOI: `https://doi.org/10.1016/j.patrec.2018.12.007`. URL: `http://www.sciencedirect.com/science/article/pii/S0167865518309188`.

[28] Vincent Labatut. "Generalized measures for the evaluation of community detection methods". In: *arXiv preprint arXiv:1303.5441* (2013).

[29] Andrea Lancichinetti and Santo Fortunato. "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities". In: *Physical Review E80, 016118 (2009)* (Apr. 24, 2009). DOI: `10.1103/PhysRevE.80.016118`. arXiv: `http://arxiv.org/abs/0904.3940v2 [physics.soc-ph]`.

[30] Yang Lei et al. "Ground truth bias in external cluster validity indices". In: *Pattern Recognition* 65 (2017), pp. 58 –70. ISSN: 0031-3203. DOI: `https://doi.org/10.1016/j.patcog.2016.12.003`. URL: `http://www.sciencedirect.com/science/article/pii/S0031320316303910`.

[31] Ambrose Lo. "Demystifying the Integrated Tail Probability Expectation Formula". In: *The American Statistician* 73.4 (2019), pp. 367–374. DOI: `10.1080/00031305.2018.1497541`. eprint: `https://doi.org/10.1080/00031305.2018.1497541`. URL: `https://doi.org/10.1080/00031305.2018.1497541`.

[32] Marina Meilă. "Comparing clusterings—an information based distance". In: *Journal of multivariate analysis* 98.5 (2007), pp. 873–895. URL: `https://www.sciencedirect.com/science/article/pii/S0047259X06002016`.

[33] M. E. J. Newman and M. Girvan. "Finding and evaluating community structure in networks". In: *Phys. Rev. E* 69 (2 2004), p. 026113. DOI: `10.1103/PhysRevE.69.026113`. URL: `https://link.aps.org/doi/10.1103/PhysRevE.69.026113`.

[34] Mark EJ Newman and Michelle Girvan. "Finding and evaluating community structure in networks". In: *Physical review E* 69.2 (2004), p. 026113.

[35] Liudmila Prokhorenkova and Alexey Tikhonov. "Community detection through likelihood optimization: in search of a sound model". In: *The World Wide Web Conference*. ACM. 2019, pp. 1498–1508.

[36] Liudmila Prokhorenkova, Alexey Tikhonov, and Nelly Litvak. "When Less is More: Systematic Analysis of Cascade-based Community Detection". In: *arXiv preprint arXiv:2002.00840* (2020).

[37] William M. Rand. "Objective Criteria for the Evaluation of Clustering Methods". In: *Journal of the American Statistical Association* 66.336 (1971), pp. 846–850. DOI: `10.1080/01621459.1971.10482356`. eprint: `https://amstat.tandfonline.com/doi/pdf/10.1080/01621459.1971.10482356`. URL: `https://amstat.tandfonline.com/doi/abs/10.1080/01621459.1971.10482356`.

[38] Jörg Reichardt and Stefan Bornholdt. "Statistical mechanics of community detection". In: *Phys. Rev. E* 74 (1 2006), p. 016110. DOI: `10.1103/PhysRevE.74.016110`. URL: `https://link.aps.org/doi/10.1103/PhysRevE.74.016110`.

[39] S Romano et al. "Standardized mutual information for clustering comparisons: One step further in adjustment for chance". In: *31st International Conference on Machine Learning, ICML 2014* 4 (Jan. 2014), pp. 2873–2882.

[40] E. H. SIMPSON. "Measurement of Diversity". In: *Nature* 163.4148 (Apr. 1949), pp. 688–688. DOI: `10.1038/163688a0`. URL: `https://doi.org/10.1038/163688a0`.

[41] Clara Stegehuis, Remco van der Hofstad, and Johan van Leeuwaarden. "Power-law relations in random networks with communities". In: *Physical Review E* 94 (July 2016), p. 012302. DOI: `10.1103/PhysRevE.94.012302`.

[42] Alexander Strehl. "Relationship-based clustering and cluster ensembles for high-dimensional data mining". PhD thesis. 2002.

[43] V. A. Traag, P. Van Dooren, and Y. Nesterov. "Narrow scope for resolution-limit-free community detection". In: *Phys. Rev. E* 84 (1 2011), p. 016114. DOI: `10.1103/PhysRevE.84.016114`. URL: `https://link.aps.org/doi/10.1103/PhysRevE.84.016114`.

[44] V. A. Traag, L. Waltman, and N. J. van Eck. "From Louvain to Leiden: guaranteeing well-connected communities". In: *Scientific Reports* 9.1 (Mar. 2019). DOI: `10.1038/s41598-019-41695-z`. URL: `https://doi.org/10.1038/s41598-019-41695-z`.

[45] Twan Van Laarhoven and Elena Marchiori. "Axioms for graph clustering quality functions". In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 193–215.

[46] Nguyen Xuan Vinh, Julien Epps, and James Bailey. "Information theoretic measures for clusterings comparison: is a correction for chance necessary?" In: *Proceedings of the 26th annual international conference on machine learning*. ACM. 2009, pp. 1073–1080.

[47] Nguyen Xuan Vinh, Julien Epps, and James Bailey. "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance". In: *Journal of Machine Learning Research* 11.Oct (2010), pp. 2837–2854.

[48] Ivan Voitalov et al. "Scale-free networks well done". In: *Phys. Rev. Research* 1 (3 2019), p. 033034. DOI: `10.1103/PhysRevResearch.1.033034`. URL: `https://link.aps.org/doi/10.1103/PhysRevResearch.1.033034`.

[49] Duncan J. Watts. "Networks, Dynamics, and the Small-World Phenomenon". In: *American Journal of Sociology* 105.2 (1999), pp. 493–527. ISSN: 00029602, 15375390. URL: `http://www.jstor.org/stable/10.1086/210318`.

[50] Dongkuan Xu and Yingjie Tian. "A comprehensive survey of clustering algorithms". In: *Annals of Data Science* 2.2 (2015), pp. 165–193.

Appendix

A Cluster similarity indices

A.1 General indices

Here we give the definitions of the indices listed in Table 1 of the main text. We define the contingency variables as $n_{ij} = |a_i \cap b_j|$. We note that all indices discussed in this paper can be expressed as functions of these contingency variables.

The *F-Measure* is defined as the harmonic mean of recall and precision. Recall is defined as

$$r(A, B) = \frac{1}{n} \sum_{i=1}^{k_A} \max_{j \in [k_B]} \{n_{ij}\},$$

and precision is its symmetric counterpart $r(B, A)$.

In [4], recall is redefined as

$$r'(A, B) = \frac{1}{n} \sum_{i=1}^{k_A} \frac{1}{|a_i|} \sum_{j=1}^{k_B} n_{ij}^2,$$

and *BCubed* is defined as the harmonic mean of $r'(A, B)$ and $r'(B, A)$.

The remainder of the indices are information-theoretic and require some additional definitions. Let $p_1, \ldots, p_\ell$ be a discrete distribution (i.e., all values are nonnegative and sum to 1). The Shannon entropy is then defined as

$$H(p_1, \ldots, p_\ell) := -\sum_{i=1}^{\ell} p_i \log(p_i).$$

The entropy of a clustering is defined as the entropy of the cluster-label distribution of a random item, i.e.,

$$H(A) := H(|a_1|/n, \ldots, |a_{k_A}|/n),$$

and similarly for $H(B)$. The joint entropy $H(A, B)$ is then defined as the entropy of the distribution with probabilities $(p_{ij})_{i \in [k_A], j \in [k_B]}$, where $p_{ij} = n_{ij}/n$.

Variation of Information [32] is defined as

$$\mathrm{VI}(A, B) = 2H(A, B) - H(A) - H(B).$$

Mutual information is defined as

$$M(A, B) = H(A) + H(B) - H(A, B).$$

The mutual information between A and B is upper-bounded by $H(A)$ and $H(B)$, which gives multiple possibilities to normalize the mutual information. In Chapter 4, we discuss two normalizations: normalization by the average of the entropies $\frac{1}{2}(H(A)+H(B))$, and normalization by the maximum of entropies $\max\{H(A), H(B)\}$. We will refer to the corresponding indices as NMI and $\mathrm{NMI}_{\max}$, respectively:

$$\mathrm{NMI}(A, B) = \frac{M(A, B)}{(H(A) + H(B))/2},$$

$$\mathrm{NMI}_{\max}(A, B) = \frac{M(A, B)}{\max\{H(A), H(B)\}}.$$

Fair NMI is a variant of NMI that includes a factor that penalizes large differences in the number of clusters [3]. It is given by

$$\mathrm{FNMI}(A,B) = e^{-|k_A-k_B|/k_A}\mathrm{NMI}(A,B).$$

In this definition, NMI may be normalized in various ways. We note that a different normalization would not result in more requirements being satisfied.

Standardized Mutual Information standardizes the mutual information w.r.t. random permutations of the items [39], i.e.,

$$\mathrm{SMI}(A,B) = \frac{M(A,B) - \mathbb{E}_{B'\sim\mathcal{C}(S(B))}(M(A,B'))}{\sigma_{B'\sim\mathcal{C}(S(B))}(M(A,B'))},$$

where σ denotes the standard deviation. Calculating the expected value and standard deviation of the mutual information is nontrivial and requires significantly more computation power than other indices. For this, we refer to the original paper [39]. Note that this index is symmetric since it does not matter whether we keep A constant while randomly permuting B or keep B constant while randomly permuting A.

A.2 Pair-counting indices and their equivalences

Pair-counting similarity indices are defined in Table 1. Table 2 lists linearly equivalent indices (see Definition 6).

A.3 Defining a subclass of pair-counting indices

In this section, we show that the subclass of pair-counting similarity indices can be uniquely defined by the property of being pair-symmetric.

For two graphs G_1 and G_2 let $M_{G_1G_2}$ denote the $N \times 2$ matrix that is obtained by concatenating their adjacency vectors. Let us write $I_M^{(G)}(M_{G_1G_2})$ for the similarity between two graphs G_1, G_2 according to some graph similarity index $I^{(G)}$. We will now characterize all pair-counting similarity indices as a subclass of the class of similarity indices between undirected graphs.

Definition 22. *We define a graph similarity index $I_M^{(G)}(M_{G_1G_2})$ to be* pair-symmetric *if interchanging two rows of M_{G_1,G_2} leaves the index unchanged.*

We give the following result.

Lemma 11. *The class of pair-symmetric graph similarity indices coincides with the class of pair-counting cluster similarity indices.*

Proof. A matrix is an ordered list of its rows. An unordered list is a multiset. Hence, when we disregard the ordering of the matrix M_{AB}, we get a multiset of the rows. This multiset contains at most four distinct elements, each corresponding to one of the pair-counts. Therefore, each $I_M^{(G)}(M_{AB})$ that is symmetric w.r.t. interchanging rows is equivalently a function of the pair-counts of A and B. □

B Checking requirements for indices

In this section, we check all non-trivial requirements for all indices. The requirements of symmetry, maximal/minimal agreement and asymptotic constant baseline can trivially be tested by simply checking $I(B,A) = I(A,B)$, $I(A,A) = c_{\max}$, $I(0,N_{10},N_{01},0) = c_{\min}$ and $I^{(p)}(p_Ap_B, p_A, p_B) = c_{\text{base}}$ respectively.

B.1 Strong monotonicity

Positive cases

Correlation Coefficient. This index has the property that inverting one of the binary vectors results in the index flipping sign. Furthermore, the index is symmetric. Therefore, we only need to prove that

[1]Throughout the literature, the Mirkin metric is defined as $2(N_{10} + N_{01})$, but we use this variant as it satisfies the scale-invariance.

Table 1: A selection of pair-counting indices. Most of these indices are taken from [30].

Index (Abbreviation)	**Expression**
Rand (R)	$\frac{N_{11}+N_{00}}{N_{11}+N_{10}+N_{01}+N_{00}}$
Adjusted Rand (AR)	$\frac{N_{11}-\frac{(N_{11}+N_{10})(N_{11}+N_{01})}{N_{11}+N_{10}+N_{01}+N_{00}}}{\frac{(N_{11}+N_{10})+(N_{11}+N_{01})}{2}-\frac{(N_{11}+N_{10})(N_{11}+N_{01})}{N_{11}+N_{10}+N_{01}+N_{00}}}$
Jaccard (J)	$\frac{N_{11}}{N_{11}+N_{10}+N_{01}}$
Jaccard Distance (JD)	$\frac{N_{10}+N_{01}}{N_{11}+N_{10}+N_{01}}$
Wallace1 (W)	$\frac{N_{11}}{N_{11}+N_{10}}$
Wallace2	$\frac{N_{11}}{N_{11}+N_{01}}$
Dice	$\frac{2N_{11}}{2N_{11}+N_{10}+N_{01}}$
Correlation Coefficient (CC)	$\frac{N_{11}N_{00}-N_{10}N_{01}}{\sqrt{(N_{11}+N_{10})(N_{11}+N_{01})(N_{00}+N_{10})(N_{00}+N_{01})}}$
Correlation Distance (CD)	$\frac{1}{\pi}\arccos\left(\frac{N_{11}N_{00}-N_{10}N_{01}}{\sqrt{(N_{11}+N_{10})(N_{11}+N_{01})(N_{00}+N_{10})(N_{00}+N_{01})}}\right)$
Sokal&Sneath-I ($S\&S_1$)	$\frac{1}{4}\left(\frac{N_{11}}{N_{11}+N_{10}}+\frac{N_{11}}{N_{11}+N_{01}}+\frac{N_{00}}{N_{00}+N_{10}}+\frac{N_{00}}{N_{00}+N_{01}}\right)$
Minkowski	$\sqrt{\frac{N_{10}+N_{01}}{N_{11}+N_{10}}}$
Hubert (H)	$\frac{N_{11}+N_{00}-N_{10}-N_{01}}{N_{11}+N_{10}+N_{01}+N_{00}}$
Folkes&Mallow	$\frac{N_{11}}{\sqrt{(N_{11}+N_{10})(N_{11}+N_{01})}}$
Sokal&Sneath-II	$\frac{\frac{1}{2}N_{11}}{\frac{1}{2}N_{11}+N_{10}+N_{01}}$
Rand Distance (a.k.a. Mirkin metric[1])	$\frac{N_{10}+N_{01}}{N_{11}+N_{10}+N_{01}+N_{00}}$
Kulczynski	$\frac{1}{2}\left(\frac{N_{11}}{N_{11}+N_{10}}+\frac{N_{11}}{N_{11}+N_{01}}\right)$
McConnaughey	$\frac{N_{11}^2-N_{10}N_{01}}{(N_{11}+N_{10})(N_{11}+N_{01})}$
Yule	$\frac{N_{11}N_{00}-N_{10}N_{01}}{N_{11}N_{10}+N_{01}N_{00}}$
Baulieu-I	$\frac{(N_{11}+N_{10}+N_{01}+N_{00})(N_{11}+N_{00})+(N_{10}-N_{01})^2}{(N_{11}+N_{10}+N_{01}+N_{00})^2}$
Russell&Rao	$\frac{N_{11}}{N_{11}+N_{10}+N_{01}+N_{00}}$
Fager&McGowan	$\frac{N_{11}}{\sqrt{(N_{11}+N_{10})(N_{11}+N_{01})}}-\frac{1}{2\sqrt{N_{11}+N_{10}}}$
Peirce	$\frac{N_{11}N_{00}-N_{10}N_{01}}{(N_{11}+N_{01})(N_{00}+N_{10})}$
Baulieu-II	$\frac{N_{11}N_{00}-N_{10}N_{01}}{(N_{11}+N_{10}+N_{01}+N_{00})^2}$
Sokal&Sneath-III	$\frac{N_{11}N_{00}}{\sqrt{(N_{11}+N_{10})(N_{11}+N_{01})(N_{00}+N_{10})(N_{00}+N_{01})}}$
Gower&Legendre	$\frac{N_{11}+N_{00}}{N_{11}+\frac{1}{2}(N_{10}+N_{01})+N_{00}}$
Rogers&Tanimoto	$\frac{N_{11}+N_{00}}{N_{11}+2(N_{10}+N_{01})+N_{00}}$
Goodman&Kruskal	$\frac{N_{11}N_{00}-N_{10}N_{01}}{N_{11}N_{00}+N_{10}N_{01}}$

this index is increasing in N_{11}. We take the derivative and omit the constant factor $((N_{00}+N_{10})(N_{00}+N_{01}))^{-\frac{1}{2}}$:

$$
\begin{aligned}
&= \frac{N_{00}}{\sqrt{(N_{11}+N_{10})(N_{11}+N_{01})}} - \frac{(N_{11}N_{00}-N_{10}N_{01})\cdot\frac{1}{2}(2N_{11}+N_{10}+N_{01})}{[(N_{11}+N_{10})(N_{11}+N_{01})]^{1.5}} \\
&= \frac{\frac{1}{2}N_{11}N_{00}(N_{10}+N_{01})+N_{00}N_{10}N_{01}}{[(N_{11}+N_{10})(N_{11}+N_{01})]^{1.5}} + \frac{\frac{1}{2}N_{10}N_{01}(2N_{11}+N_{10}+N_{01})}{[(N_{11}+N_{10})(N_{11}+N_{01})]^{1.5}} > 0.
\end{aligned}
$$

Correlation Distance. The correlation distance satisfies strong monotonicity as it is a monotone transformation of the correlation coefficient, which meets the requirement.

Table 2: Equivalent pair-counting indices

Representative Index	**Equivalent indices**
Rand	Rand Distance, Hubert
Jaccard	Jaccard Distance
Wallace1	Wallace2
Kulczynski	McConnaughey

Sokal&Sneath-I. All four fractions are nondecreasing in N_{11}, N_{00} and nonincreasing in N_{10}, N_{01} while for each of the variables there is one fraction that satisfies the monotonicity strictly so that the index is strongly monotonous.

Rand Index. For the Rand index, it can be easily seen from the form of the index that it is increasing in N_{11}, N_{00} and decreasing in N_{10}, N_{01} so that it meets the requirement.

Negative cases

Jaccard, Wallace, Dice. All these three indices are constant w.r.t. N_{00}. Therefore, these indices do not satisfy strong monotonicity.

Adjusted Rand. It holds that $AR(1,2,1,0) < AR(1,3,1,0)$, so that the index does not meet the strong monotonicity requirement.

B.2 Monotonicity

Positive cases

Rand, Correlation Coefficient, Sokal&Sneath-1, Correlation Distance. Strong monotonicity implies monotonicity. Therefore, these pair-counting indices satisfy the monotonicity requirement.

Jaccard and Dice. It can be easily seen that these indices are increasing in N_{11} while decreasing in N_{10}, N_{01}. For N_{00}, we note that whenever N_{00} gets increased, either N_{10} or N_{01} must decrease, resulting in an increase of the index. Therefore, these indices satisfy monotonicity.

Adjusted Rand. Note that for $b, b+d > 0$, it holds that

$$\frac{a+c}{b+d} > \frac{a}{b} \Leftrightarrow c > \frac{ad}{b}. \tag{1}$$

For Adjusted Rand, we have

$$a = N_{11} - \frac{1}{N}(N_{11}+N_{10})(N_{11}+N_{01}),$$
$$b = a + \frac{1}{2}(N_{10}+N_{01}).$$

Because of this, when we increment either N_{11} or N_{00} while decrementing either N_{10} or N_{01}, we get $d = c - \frac{1}{2}$. Hence, we need to prove $c > a(c - \frac{1}{2})/b$, or, equivalently

$$c > -\frac{a}{2(b-a)} = \frac{\frac{1}{N}(N_{11}+N_{10})(N_{11}+N_{01}) - N_{11}}{N_{10}+N_{01}}.$$

For simplicity we rewrite this to

$$c + \frac{p_{AB} - p_A p_B}{p_A + p_B - 2p_{AB}} > 0.$$

Where $p_A = \frac{1}{N}(N_{11}+N_{10}) \in (0,1)$ and $p_B = \frac{1}{N}(N_{11}+N_{01}) \in (0,1)$. If we increment N_{00} while decrementing either N_{10} or N_{01}, then

$$c \in \left\{\frac{1}{N}(N_{11}+N_{10}), \frac{1}{N}(N_{11}+N_{01})\right\} = \{p_A, p_B\}.$$

The symmetry of AR allows us to w.l.o.g. assume that $c = p_A$. We write

$$p_A + \frac{p_{AB} - p_A p_B}{p_A + p_B - 2p_{AB}} = \frac{p_A^2 + (1 - 2p_A)p_{AB}}{p_A + p_B - 2p_{AB}}.$$

When $p_A \leq \frac{1}{2}$, then this is clearly positive. For the case $p_A > \frac{1}{2}$, we bound $p_{AB} \leq p_A$ and bound the numerator by

$$p_A^2 + (1 - 2p_A)p_A = (1 - p_A)p_A > 0.$$

This proves the monotonicity for increasing N_{00}. When incrementing N_{11} while decrementing either N_{10} or N_{01}, we get $c \in \{1 - p_A, 1 - p_B\}$. Again, we assume w.l.o.g. that $c = 1 - p_A$ and write

$$1 - p_A + \frac{p_{AB} - p_A p_B}{p_A + p_B - 2p_{AB}} = \frac{p_A(1 - p_A) + (1 - 2p_A)(p_B - p_{AB})}{p_A + p_B - 2p_{AB}}.$$

This is clearly positive whenever $p_A \leq \frac{1}{2}$. When $p_A > \frac{1}{2}$, we bound $p_{AB} \geq p_A + p_B - 1$ and rewrite the numerator as

$$p_A(1 - p_A) + (1 - 2p_A)(p_A - 1) = (1 - p_A)(3p_A - 1) > 0.$$

This proves monotonicity for increasing N_{11}. Hence, the monotonicity requirement is met.

NMI and VI. Let B' be obtained by a perfect split of a cluster b_1 into b'_1, b'_2. Note that this increases the entropy of the candidate while keeping the joint entropy constant. Let us denote this increase in the candidate entropy by the conditional entropy $H(B'|B) = H(B') - H(B) > 0$. Now, for NMI, the numerator increases by $H(B'|B)$ while the denominator increases by at most $H(B'|B)$ (dependent on $H(A)$ and the specific normalization that is used). Therefore, NMI increases. Similarly, VI decreases by $H(B'|B)$. Concluding, both NMI and VI are monotonous w.r.t. perfect splits. Now let B'' be obtained by a perfect merge of b_1, b_2 into b''_1. This results in a difference of the entropy of the candidate $H(B'') - H(B) = -H(B|B'') < 0$. The joint entropy decreases by the same amount, so that the mutual information remains unchanged. Therefore, the numerator of NMI remains unchanged while the denominator may or may not change, depending on the normalization. For min- or max-normalization, it may remain unchanged while for any other average it increases. Hence, NMI does not satisfy monotonicity w.r.t. perfect merges for min- and max-normalization but does satisfy this for average-normalization. For VI, the distance will decrease by $H(B|B'')$ so that it indeed satisfies monotonicity w.r.t. perfect merges.

FMeasure BCubed. Note that a perfect merge increases recall while leaving precision unchanged and that a perfect split increases precision while leaving recall unchanged. The same holds for BCubed recall and BCubed precision. Hence, the harmonic mean increases.

Negative cases

FNMI We will give the following numerical counter-example: Consider $A = \{\{0, 1\}, \{2\}, \{3\}\}, B = \{\{0\}, \{1\}, \{2, 3\}\}$ and merge the first two clusters to obtain $B' = \{\{0, 1\}, \{2, 3\}\}$. This results in

$$\text{FNMI}(A, B) \approx 0.67 > 0.57 \approx \text{FNMI}(A, B').$$

This non-monotonicity is caused by the penalty factor that equals 1 for the pair A, B and equals $\exp(-1/3) \approx 0.72$ for A, B'.

SMI. For this numerical counter-example we rely on the Matlab-implementation of the index by its original authors [39]. Let $A = \{\{0, \ldots, 4\}, \{5\}\}, B = \{\{0, 1\}, \{2, 3\}, \{4\}, \{5\}\}$ and consider merging the two clusters resulting in $B' = \{\{0, 1, 2, 3\}, \{4\}, \{5\}\}$. The index remains unchanged and equals 2 before and after the merge.

Wallace. Let $k_A = 1$ and let $k_B > 1$. Then any merge of B is a perfect merge, but no increase occurs since $W_1(A, B) = 1$.

B.3 Distance

Positive cases

NMI and VI. In [47] it is proven that for max-normalization $1 - \text{NMI}$ is a distance, while in [32] it is proven that VI is a distance.

Rand. The Rand Distance $1-R$ corresponds to a rescaled version of the size of the symmetric difference between the sets of intra-cluster pairs. The symmetric difference is known to be a distance metric.

Jaccard. In [27] it is proven that the Jaccard distance $1 - J$ is indeed a distance.

Correlation Distance. In Theorem 4, it is proven that Correlation Distance is indeed a distance.

Negative cases

To prove that an index that satisfies symmetry and maximal agreement is not linearly transformable to a distance metric, we only need to disprove the triangle inequality for one instance of its equivalence class that is nonnegative and equals zero for maximal agreement.

FNMI and Wallace. These indices cannot be transformed to distances as they are not symmetric.

SMI. SMI does not satisfy the maximal agreement requirement [39], so it cannot be transformed to a metric.

FMeasure and BCubed. We will use a simple counter-example, where $|V| = 3, k_A = 1, k_B = 2, k_C = 3$. Let us denote the FMeasure and BCubed by FM, BC respectively. We get

$$1 - \text{FM}(A, C) = 1 - 0.5 > (1 - 0.8) + (1 - 0.8) = (1 - \text{FM}(A, B)) + (1 - \text{FM}(B, C))$$

and

$$1 - \text{BC}(A, C) = 1 - 0.5 > (1 - 0.71) + (1 - 0.8) \approx (1 - \text{BC}(A, B)) + (1 - \text{BC}(B, C)),$$

so that both indices violate the triangle inequality in this case.

Adjusted Rand, Dice, Correlation Coefficient and Sokal&Sneath-1. For these indices, we use the following counter-example: Let $A = \{\{0, 1\}, \{2\}, \{3\}\}, B = \{\{0, 1\}, \{2, 3\}\}, C = \{\{0\}, \{1\}, \{2, 3\}\}$. Then $p_{AB} = p_{BC} = 1/6$ and $p_{AC} = 0$ while $p_A = p_C = 1/6$ and $p_B = 1/3$. By substituting these variables, one can see that

$$1 - I^{(p)}(p_{AC}, p_A, p_C) > (1 - I^{(p)}(p_{AB}, p_A, p_B)) + (1 - I^{(p)}(p_{BC}, p_B, p_C)),$$

holds for each of these indices, contradicting the triangle inequality.

B.4 Constant baseline

Positive cases

SMI. As SMI is standardized, it satisfies the constant baseline requirement by construction.

Adjusted Rand, Correlation Coefficient and Sokal&Sneath-1. These indices all satisfy ACB while being P_{AB}-linear for fixed p_A, p_B. Therefore, the expected value equals the asymptotic constant.

Negative cases

For all the following indices, we will analyse the following counter-example. Let $|V| = n, k_A = k_B = n-1$. For each index, we will compute the expected value and show that it is not constant. All of these indices satisfy the maximal agreement requirement and maximal agreement is achieved with probability $1/N$ (the probability that the single intra-pair of A coincides with the single intra-pair of B). Furthermore, each case where the intra-pairs do not coincide will result in the same contingency variables and hence the same value of the index. We will refer to this value as $c_n(I)$. Therefore, the expected value will only have to be taken over two values and will be given by

$$\mathbb{E}[I(A,B)] = \frac{1}{N}c_{\max} + \frac{N-1}{N}c_n(I).$$

For each of these indices we will conclude that this is a non-constant function of n so that the index does not satisfy the constant baseline requirement.

Jaccard and Dice. For both these indices we have $c_{\max} = 1$ and $c_n(I) = 0$ (as $N_{11} = 0$ whenever the intra-pairs do not coincide). Hence, $\mathbb{E}[I(A,B)] = \frac{1}{N}$, which is not constant.

Rand and Wallace. As both functions are linear in P_{AB}, we can compute the expected value by simply substituting $P_{AB} = p_A p_B$. This will result in expected values $1 - 2/N + 2/N^2$ and $1/N$ for Rand and Wallace respectively, which are both non-constant.

Correlation distance. Here $c_{\max} = 0$ and

$$c_n(CD) = \frac{1}{\pi}\arccos\left(\frac{0 - 1/N^2}{(N-1)/N^2}\right),$$

so that the expected value will be given by

$$\mathbb{E}[CD(A,B)] = \frac{N-1}{N\pi}\arccos\left(-\frac{1}{N-1}\right).$$

This is non-constant (it evaluates to $0.44, 0.47$ for $n = 3, 4$ respectively). Note that this expected value converges to $\frac{1}{2}$ for $n \to \infty$, which is indeed the asymptotic baseline of the index.

FNMI and NMI. Note that in this case $k_A = k_B$ so that the penalty term of FNMI will equal 1 and FNMI will coincide with NMI. Again $c_{\max} = 1$. For the case where the intra-pairs do not coincide, the joint entropy will equal $H(A,B) = \ln(n)$ while each of the marginal entropies will equal

$$H(A) = H(B) = \frac{n-2}{n}\ln(n) + \frac{2}{n}\ln(n/2) = \ln(n) - \frac{2}{n}\ln(2).$$

This results in

$$c_n(NMI) = \frac{2H(A) - H(A,B)}{H(A)} = 1 - \frac{2\ln(n)}{n\ln(n) - 2\ln(2)},$$

and the expected value will be given by the non-constant

$$\mathbb{E}[NMI(A,B)] = 1 - \frac{N-1}{N}\frac{2\ln(n)}{n\ln(n) - 2\ln(2)}.$$

Note that as $H(A) = H(B)$, all normalizations of MI will be equal so that this counter-example proves that none of the variants of (F)NMI satisfy the constant baseline requirement.

Variation of Information. In this case $c_{\max} = 0$. We will use the entropies from the NMI-computations to conclude that

$$\mathbb{E}[VI(A,B)] = \frac{N-1}{N}(2H(A,B) - H(A) - H(B)) = \frac{N-1}{N}\frac{4}{n}\ln(2),$$

which is again non-constant.

F-measure. Here $c_{\max} = 1$. In the case where the intra-pairs do not coincide, all contingency variables will be either one or zero so that both recall and precision will equal $1 - 1/n$ so that $c_n(FM) = 1 - 1/n$. This results in the following non-constant expected value

$$\mathbb{E}[FM(A,B)] = 1 - \frac{N-1}{N}\frac{1}{n}.$$

Note that because recall equals precision in both cases, this counter-example also works for other averages than the harmonic average.

BCubed. Again $c_{\max} = 1$. In the other case, the recall and precision will again be equal. Because for BCubed, the contribution of cluster i is given by $\frac{1}{n}\max\{n_{ij}^2\}/|a_i|$, the contributions of the one- and two-clusters will be given by $\frac{1}{n}, \frac{1}{2n}$ respectively. Hence, $c_n(BC) = \frac{n-2}{n} + \frac{1}{2n} = 1 - \frac{3}{2n}$ and we get the non-constant

$$\mathbb{E}[BC(A,B)] = 1 - \frac{N-1}{N} \cdot \frac{3}{2n}.$$

We note that again, this counter-example can be extended to non-harmonic averages of the BCubed recall and precision.

C Statistical tests for constant baseline

In this section, we provide two statistical tests: one test to check whether an index I satisfies the constant baseline requirement and another to check whether I has a selection bias towards certain cluster sizes.

Checking constant baseline. Given a reference clustering A and a number of cluster sizes specifications $s_1, \dots, s_k$, we test the null hypothesis that

$$\mathbb{E}_{B \sim \mathcal{C}(s_i)}[I(A,B)]$$

is constant in $i = 1, \dots, k$. We do so by using one-way Analysis Of Variance (ANOVA). For each cluster sizes specification, we generate r clusterings. Although ANOVA assumes the data to be normally distributed, it is known to be robust for sufficiently large groups (i.e., large r).

Checking selection bias. In [39] it is observed that some indices with a constant baseline do have a *selection bias*; when we have a pool of random clusterings of various sizes and select the one that has the highest score w.r.t. a reference clustering, there is a bias of selecting certain cluster sizes. We test this bias in the following way: given a reference clustering A and cluster sizes specifications $s_1, \dots, s_k$, we repeatedly generate $B_1 \sim \mathcal{C}(s_1), \dots, B_k \sim \mathcal{C}(s_k)$. The null-hypothesis will be that each of these clusterings B_i has an equal chance of maximizing $I(A, B_i)$. We test this hypothesis by generating r pools and using the Chi-squared test.

We emphasize that these statistical tests cannot prove whether an index satisfies the requirement or has a bias. Both will return a confidence level p with which the null hypothesis can be rejected. Furthermore, for an index to not have these biases, the null hypothesis should be true for all choices of $A, s_1, \dots, s_k$, which is impossible to verify statistically.

The statistical tests have been implemented in Python and the code is available at `github.com/MartijnGosgens/validation_indices`. We applied the tests to all our indices and set $s_1 = [n^{1/4}, \dots, n^{1/4}]$, $s_2 = [n^{1/2}, \dots, n^{1/2}]$, $s_3 = [n^{3/4}, \dots, n^{3/4}]$ and considered $n \in \{50, 100, 150, \dots, 1000\}$. We use $r = 100$ and reject the null hypothesis whenever $p < 0.05$. The obtained results agree with Tables 4.1 and 4.2 except for Correlation Distance, which is so close to having a constant baseline that the tests are unable to detect it.

D Additional results

D.1 Proof of Theorem 2

Let B' be an A-consistent improvement of B. We define

$$B \otimes B' = \{b_j \cap b'_{j'} | b_j \in B, b'_{j'} \in B', b_j \cap b'_{j'} \neq \emptyset\}$$

and show that $B \otimes B'$ can be obtained from B by a sequence of perfect splits, while B' can be obtained from $B \otimes B'$ by a sequence of perfect merges. Indeed, the assumption that B' does not introduce new disagreeing pairs guarantees that any $b_j \in B$ can be split into $b_j \cap b'_1, \dots, b_j \cap b'_{k_{B'}}$ without splitting over any intra-cluster pairs of A. Let us prove that B' can be obtained from $B \otimes B'$ by perfect merges. Suppose there are two $b''_1, b''_2 \in B \otimes B'$ such that both are subsets of some $b'_{j'}$. Assume that this merge is not perfect, then there must be $v \in b''_1, w \in b''_2$ such that v, w are in different clusters of A. As v, w are in the same cluster of B', it follows from the definition of $B \otimes B'$ that v, w must be in different clusters of B. Hence, v, w is an inter-cluster pair in both A and B, while it is an intra-cluster pair of B', contradicting the assumption that B' is an A-consistent improvement of B. This concludes the proof.

D.2 Monotonicity counter-example

Let us show that there may exist similarity indices satisfying monotonicity while not satisfying the pair-counting monotonicity. The idea is that not any clustering can be obtained from a given one via perfect splits and perfect merges.

Let R' be a modification of the Rand index defined by $R'(3,3,0,0) = 1/4$ while it coincides with the standard Rand index for all other values (i.e. $R'(N_{11}, N_{10}, N_{01}, N_{00}) = R(N_{11}, N_{10}, N_{01}, N_{00})$). The modified value corresponds to a situation with four vertices ($N = 6$) where A consists of a single cluster while B consists of two clusters of sizes 3 and 1 respectively. This index is clearly not pair-counting monotone since $R'(2,4,0,0) = 1/3 > 1/4$. However, these pair-counts correspond to a situation where B' consists of two clusters of size 2. This does not form a counter-example for monotonicity since B is not an A-consistent improvement upon B'. Moreover, the only B' upon which B is an A-consistent improvement consist of either 4 or 3 clusters, for which R' gives scores 0 and $1/6$ respectively. In both cases $R'(A, B') < R'(A, B) = 1/4$, so that monotonicity is not violated. Note that for all A_2, B_2, B'_2 such that B'_2 is an A_2-consistent improvement upon B_2 and $B'_2 \neq B'$, we have that $R'(A_2, B_2) \leq R(A_2, B_2)$ so that monotonicity follows from the fact that the (unmodified) Rand index is monotone. This proves that R' satisfies monotonicity while it does not satisfy pair-counting monotonicity.

D.3 Proof of Theorem 3

We prove the equivalent statement

$$I^{(p)}\left(P_{AB}^{(n)}, p_A^{(n)}, p_B^{(n)}\right) - I^{(p)}\left(p_A^{(n)} p_B^{(n)}, p_A^{(n)}, p_B^{(n)}\right) \xrightarrow{P} 0\,.$$

We first prove that $P_{AB}^{(n)} - p_A^{(n)} p_B^{(n)} \xrightarrow{P} 0$ so that the above follows from the continuous mapping theorem. Chebychev's inequality gives

$$\mathbb{P}\left(|P_{AB}^{(n)} - p_A^{(n)} p_B^{(n)}| > \varepsilon\right) \leq \frac{1}{\binom{n}{2}^2 \varepsilon^2} \mathrm{Var}\left(N_{11}^{(n)}\right) \to 0.$$

The last step follows from the fact that $\mathrm{Var}(N_{11}) = o(n^4)$, as we will prove in the remainder of this section. Even though in the definition, A is fixed while B is randomly permuted, it is convenient to assume both clusterings are randomly permuted for this proof.

We will show that $\mathrm{Var}(N_{11}) = o(n^4)$. To compute the variance, we first inspect the second moment. Let $A(S)$ denote the indicator function of the event that all elements of $S \subset [n]$ are in the same cluster in A. Define $B(S)$ similarly and let $AB(S) = A(S)B(S)$. Let e, e_1, e_2 range over subsets of $[n]$ of size 2. We write

$$\begin{aligned}
N_{11}^2 = \left(\sum_e AB(e)\right)^2 &= \sum_{e_1, e_2} AB(e_1)AB(e_2) \\
&= \sum_{|e_1 \cap e_2| = 2} AB(e_1)AB(e_2) + \sum_{|e_1 \cap e_2| = 1} AB(e_1)AB(e_2) + \sum_{|e_1 \cap e_2| = 0} AB(e_1)AB(e_2) \\
&= N_{11} + \sum_{|e_1 \cap e_2| = 1} AB(e_1 \cup e_2) + \sum_{e_1 \cap e_2 = \emptyset} AB(e_1)AB(e_2).
\end{aligned}$$

We take the expectation

$$\begin{aligned}\mathbb{E}[N_{11}^2] =&\mathbb{E}[N_{11}] + 6\binom{n}{3}\mathbb{E}[AB(\{v_1, v_2, v_3\})] \\ &+ \binom{n}{2}\binom{n-2}{2}\mathbb{E}[AB(e_1)AB(e_2)],\end{aligned}$$

where $v_1, v_2, v_3 \in V$ distinct and $e_1 \cap e_2 = \emptyset$. The first two terms are obviously $o(n^4)$. We inspect the last term

$$\binom{n}{2}\binom{n-2}{2}\mathbb{E}[AB(e_1)AB(e_2)] = \binom{n}{2}\sum_{i,j}\mathbb{P}(e_1 \subset a_i \cap b_j) \quad \times \binom{n-2}{2}\mathbb{E}[AB(e_2)|e_1 \subset a_i \cap b_j]\,. \tag{2}$$

Now we rewrite $\mathbb{E}[N_{11}]^2$ to

$$\mathbb{E}[N_{11}]^2 = \binom{n}{2}\sum_{i,j}\mathbb{P}(e_1 \subset a_i \cap b_j)\binom{n}{2}\mathbb{E}[AB(e_2)].$$

Note that $\binom{n}{2}\mathbb{E}[AB(e_2)] > \binom{n-2}{2}\mathbb{E}[AB(e_2)]$ so that the difference between (2) and $\mathbb{E}[N_{11}]^2$ can be bounded by

$$\binom{n}{2}\binom{n-2}{2}\sum_{i,j}\mathbb{P}(e_1 \subset a_i \cap b_j)\cdot(\mathbb{E}[AB(e_2)|e_1 \subset a_i \cap b_j] - \mathbb{E}[AB(e_2)]).$$

As $\binom{n}{2}\binom{n-2}{2} = O(n^4)$, what remains to be proven is

$$\sum_{i,j}\mathbb{P}(e_1 \subset a_i \cap b_j)\cdot(\mathbb{E}[AB(e_2)|e_1 \subset a_i \cap b_j] - \mathbb{E}[AB(e_2)]) = o(1).$$

Note that it is sufficient to prove that

$$\mathbb{E}[AB(e_2)|e_1 \subset a_i \cap b_j] - \mathbb{E}[AB(e_2)] = o(1),$$

for all i, j. Note that $\mathbb{E}[AB(e_2)] = m_A m_B/N^2$, while

$$\mathbb{E}[AB(e_2)|e_1 \subset a_i \cap b_j] = \frac{(m_A - (2|a_i| - 3))(m_B - (2|b_j| - 3))}{(N - (2n-3))^2}.$$

Hence, the difference will be given by

$$\begin{aligned}&\frac{(m_A - (2|a_i| - 3))(m_B - (2|b_j| - 3))}{(N - (2n-3))^2} - \frac{m_A m_B}{N^2} \\ =&\frac{N^2(m_A - (2|a_i| - 3))(m_B - (2|b_j| - 3))}{N^2(N - (2n-3))^2} - \frac{(N - (2n-3))^2 m_A m_B}{N^2(N - (2n-3))^2} \\ =&\frac{N^2((2|a_i| - 3)(2|b_j| - 3) - m_A(2|b_j| - 3) - m_B(2|a_i| - 3))}{N^2(N - (2n-3))^2} + \frac{m_A m_B(2N(2n-3) - (2n-3)^2)}{N^2(N - (2n-3))^2} \\ =&\frac{((2|a_i| - 3)(2|b_j| - 3) - m_A(2|b_j| - 3) - m_B(2|a_i| - 3))}{(N - (2n-3))^2} + \frac{m_A m_B}{N^2}\frac{(2N(2n-3) - (2n-3)^2)}{(N - (2n-3))^2} \\ =&\frac{O(n^3)}{(N - (2n-3))^2} + \frac{m_A m_B}{N^2}\frac{O(n^3)}{N^2(N - (2n-3))^2} \\ =&o(1),\end{aligned}$$

as required.

Table 3: Similarity of candidate partitions to the ground truth one. In bold are the inconsistently ranked pairs of partitions.

	A_{prod}	A_1	A_2
NMI	0.9326	0.9479	0.9482
NMI$_{\max}$	0.8928	**0.9457**	**0.9298**
FNMI	0.7551	**0.9304**	**0.8722**
VI	0.6996	0.5662	0.5503
FMeasure	0.8675	0.8782	0.8852
BCubed	0.8302	0.8431	0.8543
R	0.9827	**0.9915**	**0.9901**
AR	0.4911	0.5999	0.6213
J	0.3320	0.4329	0.4556
W1	**0.8323**	**0.6287**	0.8010
W2	0.3558	**0.5816**	**0.5138**
D	0.4985	0.6042	0.6260
S&S1	0.7926	0.8004	0.8262
CC	0.5376	0.6004	0.6371
CD	0.3193	0.2950	0.2802