Network Science **8** (1): 1–41, 2020
doi:10.1017/nws.2019.59

CAMBRIDGE
UNIVERSITY PRESS

ORIGINAL ARTICLE

Community structure: A comparative evaluation of community detection methods

Vinh Loc Dao, Cécile Bothorel* and Philippe Lenca

IMT Atlantique, Lab-STICC CNRS UMR 6285, F-29238, Brest, France
(e-mails: loc.daovinh@gmail.com; philippe.lenca@imt-atlantique.fr)
*Corresponding author. Email: cecile.bothorel@imt-atlantique.fr

Action Editor: Filippo Menczer

Abstract

Discovering community structure in complex networks is a mature field since a tremendous number of community detection methods have been introduced in the literature. Nevertheless, it is still very challenging for practitioners to determine which method would be suitable to get insights into the structural information of the networks they study. Many recent efforts have been devoted to investigating various quality scores of the community structure, but the problem of distinguishing between different types of communities is still open. In this paper, we propose a comparative, extensive, and empirical study to investigate what types of communities many state-of-the-art and well-known community detection methods are producing. Specifically, we provide comprehensive analyses on computation time, community size distribution, a comparative evaluation of methods according to their optimization schemes as well as a comparison of their partitioning strategy through validation metrics. We process our analyses on a very large corpus of hundreds of networks from five different network categories and propose ways to classify community detection methods, helping a potential user to navigate the complex landscape of community detection.

Keywords: community detection; community structure; comparative analysis; empirical analysis; computation time; community size; structural quality function; validation metric; decision-making assistance for practitioners

1. Introduction

In network science, *community detection* (sometimes called *graph clustering*) is one of the fundamental challenges to discovering the structure of networks on the mesoscopic level. However, it is an ill-defined problem. According to Arifin *et al.* (2017), *"it does not have clear goals, solution paths or expected solution."* There is no universal definition or closed form formula of what kind of objects one should be looking for (Fortunato & Hric, 2016), and consequently, there is no golden standard to assess the quality of a community structure and the performance of a community detection algorithm.

The most frequently found definition of community in the network science literature is derived from the mechanism of connection preference. It implies that *a community is a group of nodes (a subgraph) in a graph where there must be more edges (denser) connecting them together than edges connecting the community with the rest of the graph* (Radicchi *et al.*, 2004; Fortunato, 2010). Newman defines a community as a *"group of vertices with a higher-than-average density of edges connecting them"* (Newman, 2006b). Depending on the context, a community may be called a *cluster*, a *module*, a *class*, or a *modular group*. This is the most basic definition that sets the fundamental requirement for most of its derivative definitions. Many different variations of community

© Cambridge University Press 2020

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

could be found in Wasserman (1994), for instance, *LS-set*, which is a set of nodes in a network such that each of its proper subsets has more ties to its complement within the set than outside; or *k-core*, which is a subgraph in which each node is adjacent to at least a minimum number k of the other nodes in the subgraph. However, in recent developments of community detection algorithms, there is no consensus of the quantity of edges in reality that could be considered as *"many."* Communities are just algorithmically defined, i.e., they are final products of the algorithm without any precise *a priori* definition (Fortunato, 2010).

In practice, there are even more constraints, which are sometimes not explicitly expressed, than which appeared in the announcement of the problem. If one only looks for a partition of a graph that maximizes the number of internal edges and minimizes the number of external edges, then the graph itself can be considered as a big community with no external connection. Another solution is to leave the node having the smallest degree of a graph in one community and all other nodes in another community. This solution could also maximize the ratio between the external and internal edges. However, these monotonous solutions do not seem to seduce most (if not all) analysts considering using a community detection algorithm to detect communities. In fact, it is preferable to cluster a network into groups of relatively similar size (Newman, 2010). This means that somehow the relative size of communities is significant but this notion of size has not been explicitly announced. Besides, there are many other criteria that could be mentioned such as community complete mutuality, reachability, vertex degree distribution, and the comparison of internal versus external cohesion (Wasserman, 1994; Fortunato, 2010). There exists a subtle compromise between adding into a community new vertices—along with their edges—and conserving the common property that defines the group. In fact, different community detection methods have different ways to define what defines the groups and how to consider those constraints or not. They produce different community structures. The following are the main reasons that could lead to these disagreements between detection methods:

- Different algorithms may implement different meanings of the notion of community.
- When two algorithms define the same concept of community, it may also mathematically and algorithmically be formalized in different ways (same objective but different objective functions) and hence leads us to different results.
- Even when two algorithms have exactly the same objective function, the algorithmic mechanism they employ to find communities also decides what they are going to find, especially in heuristic searching approaches.
- Initial configuration is also another important factor that affects the final result of an algorithm, many community detection methods are not deterministic.
- Each method may include a consideration between obtaining an optimized solution and providing reasonable performances (in terms of calculation time, memory consumption, etc.). This trade-off may be considered differently across the methods.
- Some algorithms are variable according to input data and will prove more or less efficient on some kinds of inputs than on others.
- Variations due to implementation factors could also impact the final result of an algorithm.
- Finally, in some algorithms, there are tie-break situations where they have to choose randomly without any factor related to their final objectives. It may also significantly affect the result that one would get if the tie-break problems have been resolved in a different way.

Due to the many reasons stated above, choosing the community detection method that corresponds well to a particular scenario or to an expectation of quality is not straightforward. This paper is not the first effort as a guide for choosing community detection methods. For the same purpose, readers can also refer to Fortunato & Hric (2016); Ghasemian *et al.* (2018), and Jebabli *et al.* (2015, 2018). The difference with existing work is that we provide an exhaustive empirical evaluation on many state-of-the-art methods, using a large network dataset (more

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

than 100 different real networks, of which only a few are synthetic) and using popular quality scoring functions of structural characteristics of communities. Hence, we are able to disclose a performance review of many different methods based solely on empirical experiments and measurements that help network practitioners to reveal the functionality of each method and behaviors that they could exhibit in real-world networks, independent of theoretical mechanisms. The objective of this work does not differ from existing work in the literature. It does not aim to bring readers an innovative factor in the community detection problem nor discuss every related challenging point of the community detection problem. However, it provides an evaluation as well as an approach to examine community detection algorithms. This evaluation is very useful for network practitioners to obtain a quick notion of community structure quality before actually applying a method to their concrete dataset. These notions of quality are expected to guide users to appropriate choices of methods according to their quality criteria.

This paper is organized as follows: Section 2 introduces some popular and state-of-the-art community detection methods that will be analyzed in this paper as well as the benchmarking dataset employed in our experiments. Then, Section 3 presents analyses of the most essential aspects of community detection performance including computation time, community size distribution as well as numbers of detected communities by each method. In Section 4, we address different structural quality aspects of community structures. This is followed by a comparative evaluation using many popular clustering validation metrics in Section 5, which are widely used in the context of community detection. The Sections 3, 4, and 5 are independent. They bring complementary material and criteria to describe the quality of community structures and compare the different methods from different perspectives. Each section is detailed, making this article a little long, but they end with a summary that allows readers to get an overview and provide them with the necessary key lessons to be learned. Finally, we present some close results of related work that can be found in the literature in Section 6 and conclude our study with some discussions and recommendations in Section 7.

2. Community detection methods and dataset

2.1 Community detection methods

We present, in this section, some popular community detection methods that have been widely used and discussed in the literature. Note that in recent years, there are a large number of innovative methods that have been proposed to solve either generic or specific cases. However, an empirical and exhaustive analysis of all methods would be impractical, even unrealizable. To the best of our knowledge, we are trying to include the most important and representative methods among the community detection approaches.

There are many possible theoretical taxonomies for community detection methods, depending on the final objective of each categorization. For instance, one could classify methods according to differences in searching mechanisms, objective functions, assumptions about the structure to be found, expected qualities, hypothesis models, or even the theoretical model employed, etc. Moreover, what makes the problem trickier is that many methods are not just some simple algorithms to resolve a specific problem, but instead are combinations of many different approaches in order to leverage as much algorithmic power as possible from each one. There is not a consensus on how different methods are similar and how they can be classified into different families whose functionality can be resumed in some simple words. Porter *et al.* (2009) use centrality-based, local techniques, *modularity* optimization, spectral clustering to describe communities in networks. In Fortunato (2010) and Fortunato & Hric (2016), the authors group community detection methods into traditional data clustering methods, divisive algorithms, modularity-based methods, spectral algorithms, dynamic algorithms, and methods based on statistical inference. Coscia *et al.* (2011) classify community discovering into feature distance based, internal density, bridge detection, diffusion process, closeness based, structural pattern based, link clustering, and meta clustering. In

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

a context of Social Media, Papadopoulos *et al.* (2011) compare methods in substructure detection, vertex clustering, community quality optimization, divisive, and model-based approaches. Bohlin *et al.* (2014) aggregate different approaches into three principle classes representing different network models: null models, block models, and flow models. Schaub *et al.* (2017) classify methods into four perspectives: cut based, clustering internal density based, stochastic equivalent based, and dynamic based showing four different facets of community structure. Finally, Ghasemian *et al.* (2018) adopt an experimental classification as well as group community detection methods in distinct families based on their outputs on many real-world networks using a validation metric (a topic we will address in Section 5.1).

In the following sections, we choose to classify community detection methods according to different theoretical approaches including edge removal, modularity optimization, spectral partitioning, dynamic process, and statistical inference. Although every theoretical taxonomy may be questionable, this categorization is expected to support the empirical analyses in the next sessions to verify whether theoretical and conceptual closeness could engender quality closeness in practice.

2.1.1 Edge removal-based methods

Edge betweenness (*GN*) Newman & Girvan (2004) detect communities by removing edges progressively according to their betweenness centrality scores. This method is based on the intuition that dense areas of a graph are loosely connected by a few edges that are located in the shortest paths between pairs of nodes. Removing these edges would reveal densely connected communities.

Edge clustering coefficient (*RCCLP*) Radicchi *et al.* (2004) suggest replacing the edge betweenness centrality of Girvan–Newman's method by an edge clustering coefficient, which requires less computation time and hence reduces the complexity. In this paper, we analyze two configurations of this method corresponding to triangular ($g = 3$ denoted by *RCCLP-3*) and quadrangular ($g = 4$ denoted by *RCCLP-4*) versions.

2.1.2 Modularity optimization methods

Greedy optimization (*CNM*) Clauset *et al.* (2004) greedily maximize the modularity function Q by aggregating iteratively connected communities, which induces a maximum increase or minimum decrease in modularity ΔQ.

Louvain method Blondel *et al.* (2008) adopt a two-step agglomerative process similar to the greedy optimization method. However, in each iteration of the first step, it allows nodes to move between communities until no additional gain in modularity can be obtained with these local switches. Then, a new graph whose vertices are the communities resulting from the first step is built and the process is repeated on the new graph to reduce computation time, leading to a hierarchical clustering.

Spectral method (*SN*) Newman (2006a) reformulates the idea of maximizing modularity as a spectral partitioning problem by constructing a *modularity matrix* and using its leading eigenvector to spectrally partition networks into subnetworks. The community structure is identified using the eigenvectors of this matrix. Eigenvectors are used to project each node into low-dimensional node vectors, and the community structure is identified through clustering the node vectors (e.g., with the k-means clustering method).

2.1.3 Dynamic process-based methods

Walktrap Pons & Latapy (2005) define a pairwise *dynamic distance* between nodes and then apply traditional hierarchical clustering to detect community structure. The distance between two nodes is defined in terms of a random walk process. The basic idea is that if two nodes are in the same

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

community, they tend to *"see"* other nodes in the same way, that is, the probability of getting to a third node through a random walk should not be very different for these two nodes, with higher distance for nodes belonging to other communities.

Infomod Rosvall & Bergstrom (2007) use an information theoretic model where a *signaler* tries to send the structure of a network over a limited capacity transmission channel to a *receiver*. The network must be encoded in community structure in a way that minimizes the transferred information and the information loss.

Infomap Rosvall *et al.* (2009) represent networks by a two-level structure description. Analogically, each node in a network is encrypted by a unique codeword composed of two parts: a prefix representing the community to which it belongs and a suffix representing the local code. Detecting community structure becomes equivalent to searching for the coding rule to minimize the average code length describing random walks on the network.

2.1.4 Statistical inference-based methods

Stochastic block model (*SBM*) (Riolo *et al.*, 2017). The SBM was introduced by Holland *et al.* (1983). Here, we use the implementation by Riolo *et al.* (2017) that uses a Monte Carlo sampling scheme to maximize a Bayesian posterior probability distribution over possible divisions of the network into communities. This probability implies an expected network model to be fitted from the observed network data. In this block model variant, the authors employ a new prior on the number of communities based on a queueing-type mechanism to calculate posterior probability. We analyze, in the following sections, both traditional *SBM* and the *degree-corrected* version *DCSBM*, which had been proven to perform better in practice.

Order statistics local optimization Lancichinetti *et al.* (2011) measure the statistical significance of a community by calculating the probability of finding a similar one in a null model. Following this concept, nodes are gradually aggregated into communities to find significant communities. Then, nodes are considered to be swapped between communities in order to increase significance level.

2.1.5 Other methods

Spin glass model (*RB*). Reichardt & Bornholdt (2006) is a method relying on an analogy between the statistical mechanics of complex networks and physical spin glass models. It finds communities by fitting the ground state of a spin glass model. Instead of favoring only intracommunity edges and penalizing intercommunity edges like the traditional modularity measure, this model also favors intercommunity nonedges and penalizes intracommunity nonedges.

Label propagation (*LPA*). Raghavan *et al.* (2007) exploit the topology of networks to infer community structure. It is closely related to the context of message passing paradigms or epidemic spreading. The principled idea of this method is based on the concept that nodes should belong to the community of most of their neighbors. Hence, they gradually update their memberships according to their incident nodes.

Speaker–listener label propagation (*SLPA*) by Xie & Szymanski (2012) modifies the propagation mechanism above by a new label update strategy. Also, instead of keeping only hard membership information, each node is equipped by a memory to contain the labels that it receives. Then, in the update phase, nodes transmit the membership to their neighbors according to the membership frequency in the memories.

Mixing global and local information (*Conclude*). Meo *et al.* (2014) combine a dynamic distance with a modularity optimization process to identify community structure. First, the authors define a new pairwise proximity function using random and nonbacktracking walks of finite length to determine distances between vertices. Then, the multilevel modularity optimization strategy of the

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Table 1. Community detection methods involved in the study.

Approach	Publication	Ref. label	Time order	Implementation
Edge removal	(Girvan & Newman, 2002)	GN	$\mathcal{O}(nm^2)$	igraph[a]
	(Radicchi *et al.*, 2004)	RCCLP	$\mathcal{O}(m^4/n^2)$	Authors[b]
Modularity optimization	(Clauset *et al.*, 2004)	CNM	$\mathcal{O}(m \log^2(n))$	igraph
	(Blondel *et al.*, 2008)	Louvain	$\mathcal{O}(n \log(n))$	Authors[c]
	(Newman, 2006a)	SN	$\mathcal{O}(nm \log(n))$	igraph
Dynamic process	(Pons & Latapy, 2005)	Walktrap	$\mathcal{O}(n)$	igraph
	(Rosvall & Bergstrom, 2007)	Infomod	Nonavailable	Authors[d]
	(Rosvall *et al.*, 2009)	Infomap	$\mathcal{O}(m)$	Authors[e]/igraph
Statistical inference	(Lancichinetti *et al.*, 2011)	OSLOM	$\mathcal{O}(n^2)$	Authors[f]
	(Riolo *et al.*, 2017)	(DC)SBM	Parametric	Authors[g]
Other methods	(Reichardt & Bornholdt, 2006)	RB	$\mathcal{O}(n^2 log(n))$	igraph
	(Raghavan *et al.*, 2007)	LPA	$\mathcal{O}(m)$	igraph
	(Xie & Szymanski, 2012)	SLPA	$\mathcal{O}(m)$	Authors[h]
	(Meo *et al.*, 2014)	Conclude	$\mathcal{O}(n+m)$	Authors[i]

[a] Published at http://igraph.org/
[b] Published at http://homes.sice.indiana.edu/filiradi/resources.html
[c] Published at https://sourceforge.net/projects/louvain/
[d] Published at http://www.tp.umu.se/~rosvall/code.html
[e] Published at http://www.mapequation.org/
[f] Published at http://www.oslom.org/
[g] Published at http://www-personal.umich.edu/~mejn/
[h] Published at https://sites.google.com/site/communitydetectionslpa/
[i] Published at http://www.emilio.ferrara.name/code/conclude/

Louvain method (Blondel *et al.*, 2008) is combined with the defined distance to find community structure.

Table 1 summarizes the methods presented previously, grouped by different approaches. Since community detection is receiving more and more attention in the network science community, there is a huge volume of work that has been published in recent years to evaluate different methods including both theoretical and empirical approaches. However, there is no formal and quantitative definition of community that is explicitly implemented inside algorithms. Therefore, it is challenging to distinguish the topological differences of community structures using different methods, even when the associated concepts are quite theoretically discernible. Additionally, it is still not clear whether proximity in the assumption of community concept will engender a structural similarity of communities that could be detected. Our comparative analysis in the next sections will try to address these questions in more detail.

2.2 Experimental dataset

In this section, we describe some statistical properties of networks that will be included in the following analysis. Available biological networks that have been published and analyzed widely are relatively small in comparison to the other networks of the other families. Besides, due to the complexity of the analysis process, we limit the domains of interest to five categories that are commonly researched and where numerous networks are available. We introduce a 6th category with various types of networks, each of which is underrepresented, such as ecological networks for example. In this study, we consider 108 different networks that are relatively large in comparison to many other studies. Many notable related works where some of these networks are also employed could be mentioned for quick reference: Orman *et al.* (2012) use six networks to evaluate the structure of communities discovered by several detection techniques; Lancichinetti *et al.* (2010) use 15 networks to characterize structural communities; Hric *et al.* (2014) use 16 networks to reveal differences between structural communities and ground truth; Leskovec *et al.*

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Table 2. A summary of network dataset used in this analysis, where "Size" is the number of networks analyzed in each category and "Nodes" and "Edges" indicate the average number of nodes and edges of networks in each category, respectively. *The last row shows the total number of networks in the whole dataset. This dataset is collected from several sources including: http://networkrepository.com (Rossi & Ahmed, 2015), http://konect.uni-koblenz.de (Jerome, 2013), http://snap.stanford.edu (Leskovec & Krevl, 2014).

Category	Size	Nodes	Edges	Notable networks
Biological	7	1,860	10,763	Yeast, brain, protein–protein interactions
Communication	9	39,595	195,032	Email, forums, message exchanges
Information	25	38,358	159,812	Amazon, DBLP, citation, and education webs
Social	37	6,888	49,666	Facebook, Youtube, Google Plus networks
Technological	19	18,431	48,494	Internet, CAIDA AS, Gnutella P2P networks
Miscellaneous	11	4,298	49,033	Ecology, power-grid, synthetic networks
Total*	108			

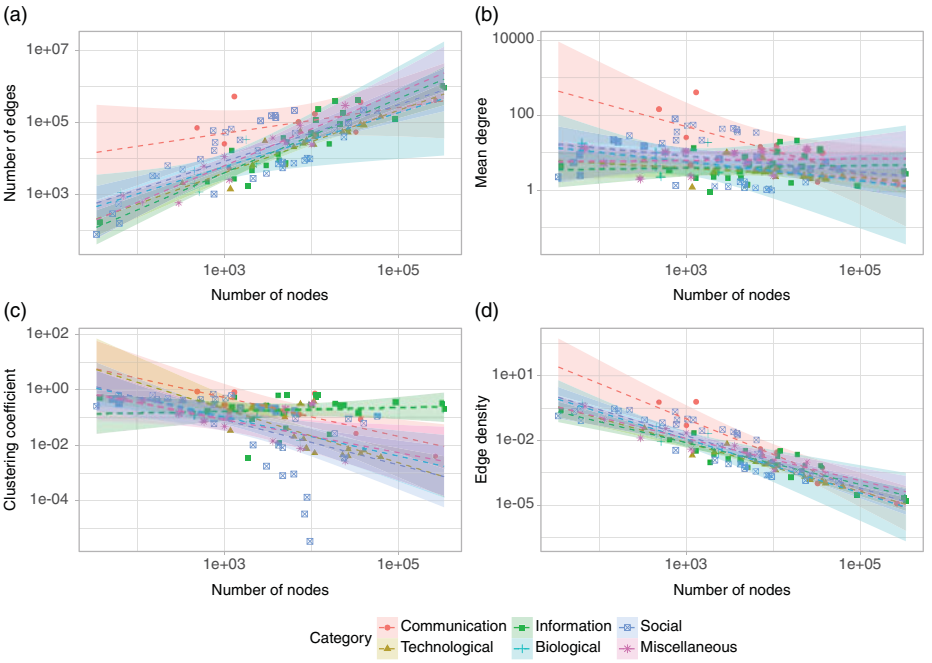


Figure 1. From left to right, top to bottom, we illustrate structural characteristics of the 108 networks: (a) the number of edges as a function of the number of nodes, (b) mean degree $\langle k \rangle$ as a function of the number of nodes, (c) clustering coefficient according to the number of nodes, and (d) edge density as a function of the number of nodes. The colored backgrounds represent the 95% confidence intervals of the relations estimated from the dataset using a linear regression model for the corresponding variables in each network category.

(2008) use over 100 networks to analyze network community profile and 230 networks to evaluate the goodness of ground-truth communities in social networks. Within this number, 225 samples of the Ning online social networking platform networks[1] are aggregated (Yang & Leskovec, 2013). Let us mention, finally, the related work by Ghasemian *et al.* (2018), which has introduced the large CommunityFitNet corpus[2] containing 572 real-world networks. Table 2 summarizes the composition of networks that have been analyzed in this section.

Some notable structural characteristics of networks in the dataset are illustrated in Figure 1. It is noticeable that apart from biological networks that are relatively small, the other classes cover quite a wide range of numbers of nodes, edges, mean degree, clustering coefficient, and

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

edge density. Since real-world networks are relatively sparse, the number of edges increases linearly according to the number of nodes and, consequently, the edge densities decrease linearly by the number of nodes (since the number of possible connections increases quadratically by the number of nodes in a community). This sparsity property can easily be seen in Figure 1(a) and (d). Specifically, the number of edges increases linearly according to the number of nodes with equivalent rates among different network categories as can be deduced from the gradients of the linear estimates. From Figure 1(b), it can be seen that the average degree of the networks in the dataset varies principally between 1 and 100 edges per node except for two communication networks. Also, the majority of networks have an average degree from 10 to 20 connections. From a global point of view, networks in the dataset have a quite strong modular quality since most of them have relatively high clustering coefficients as shown in Figure 1(c).

3. Preliminary analysis of community detection methods

3.1 Computation time performance

Since computation time is a crucial factor to be considered in the selection of an algorithm, it is worth analyzing experimental performances to see how different community detection methods accomplish their task in real-world networks. We tested the official implementations of community detection methods introduced in Table 1 on the dataset collection presented in Table 2. The implementations are provided officially either by their authors or by popular network analysis libraries that can be easily accessed by a large public.

We ran the implementations stated above to identify community structures on all the networks in the dataset. We measured the time needed for each implementation to compute each partition on each network. The default parameters configured by the implementations remained unchanged during the test. The calculations were executed on a server equipped by an Intel Xeon CPU E5-2650 with 32 cores of 2.60 GHz and a memory capacity of approximately 100 GB. However, due to the high complexity of some methods, only processes that finish within a practical time limit (less than 4 h) are taken into account. However, for a reference purpose, we let some longer computations continue. For example, *Conclude* method took approximately 9 days to identify community structures on a network of 300 thousand vertices and 1 million edges; *GN* method did not finish its calculation for networks of more than 4 thousand nodes and 40 thousand edges within 2 days. Consequently, the experiments that theoretically require too much time are neglected in the test. It is also worth noting that the calculations of communities on large-scale networks are also restrained by limited memory. Thus, calculations that should be finished within 4 h but required too much memory cannot be shown here either. We repeat the calculations 5 times for each pair graph/method to reduce the fluctuation impact. Eliminating all the cases that do not satisfy our requirements, the final successful rate (number of partitions identified over the number of possible tests) ended at around 44.72%, mainly because of time/memory overflow.

In the following figures (from Figures 2, 3, 4, 5, 6, and 7) that illustrate the analyses of experimental time consumption, some conventions are commonly used. Points in the figures correspond to separated executions that have been measured. The solid lines with the same colors are estimated relations between computation time and network size (number of vertices and number of edges), using a local regression model (Cleveland, 1979). The dark colored backgrounds around the regression curves represent 95% confidence intervals of the model parameters. Besides, for reference purposes, we also show *theoretical* execution time (i.e., in the worst case, in terms of number of calculations needed) for each algorithm. In our estimates, we plot this theoretical execution time by assigning $n = m$, as from the analysis of structural characteristics of the dataset shown above in Figure 1(a), most networks are sparse, that is, the number of edges (m) increases linearly with the number of nodes (n). For simplicity of illustration, we grouped our results according to our classification (Table 1).

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

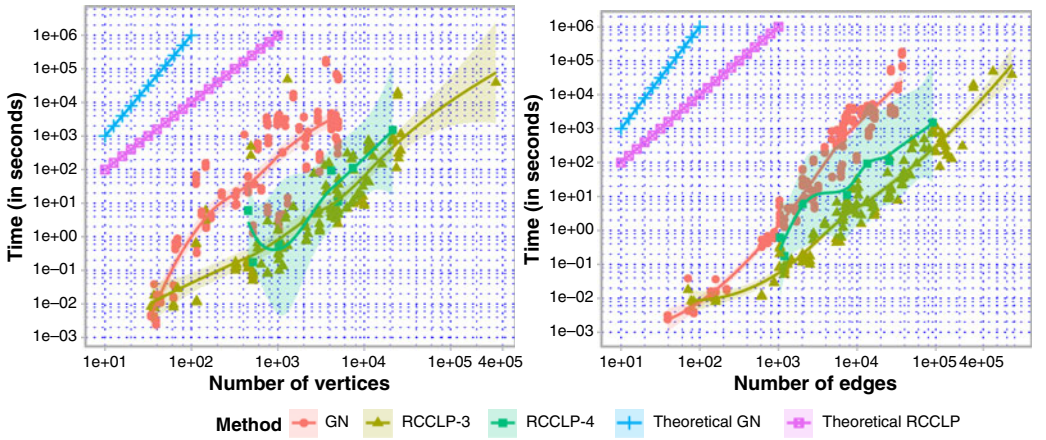


Figure 2. The execution time needed by GN, RCCLP-3, and RCCLP-4 methods to identify community structures on networks of the dataset.

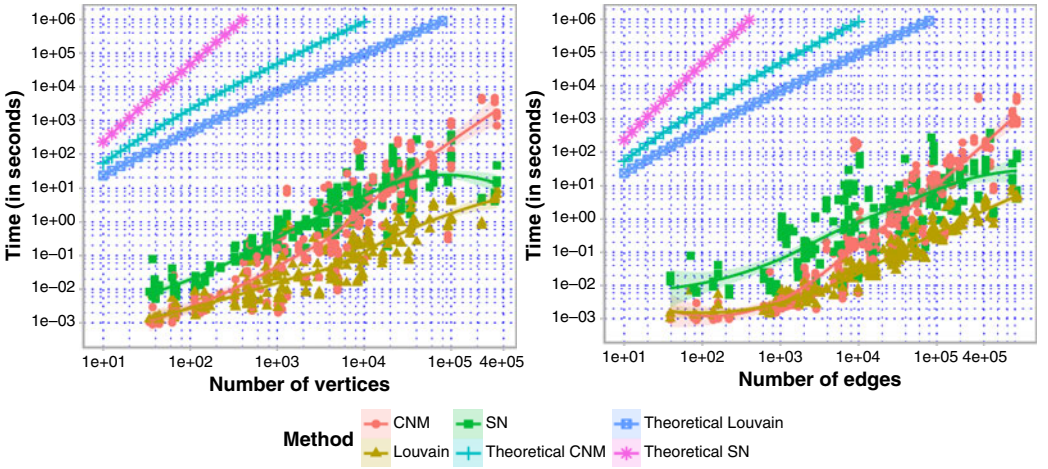


Figure 3. The execution time needed by CNM, Louvain, and SN methods to identify community structures on networks of the dataset.

The first group of methods consists of centrality detection techniques to identify community structure. As we can see in Figure 2, the *GN* method cannot be accomplished in our test for networks of more than 4 thousand nodes or 30 thousand edges. The outcome is quite reasonable since the theoretical estimation for this method is $\mathcal{O}(nm^2)$, which grows quickly with the network size. It should be remembered that the primary goal of the *RCCLP* method is to reduce the time complexity of the *GN* method. We can easily observe that this objective is achieved since the *RCCLP-3* reduces by an order of around 10^3 times for graphs of 300 nodes. *RCCLP-3* can function well with graphs up to millions of edges. However, when we used the same test with *RCCLP-4*, the method rarely reached its terminus for large graphs as well as small graphs. As we can see in the figure, there are few dots at either side. The reason is that there are not many (they may even be absent) four-step close paths on real-world networks. As it is not very probable that such structures exist in small graphs, finding them in large graphs also requires a huge amount of time, *RCCLP-4* shows a poor performance in our tests. Therefore, this configuration of the method is not recommended, as versions with $g > 4$ would logically perform poorly. It is also worth noticing that *RCCLP-3* and *RCCLP-4* are extremely memory consuming and are not suitable for limited

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

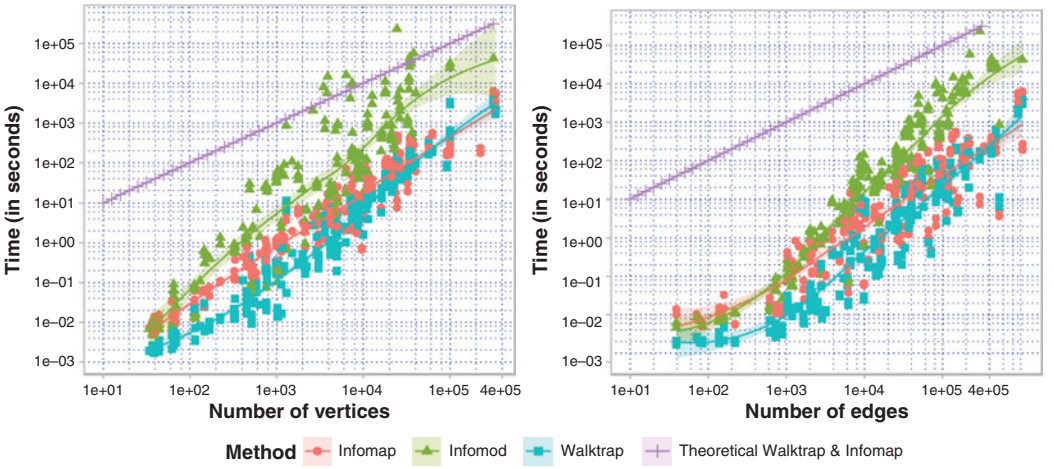


Figure 4. The execution time needed by Infomap, Infomod, and Walktrap methods to identify community structures on networks of the dataset.

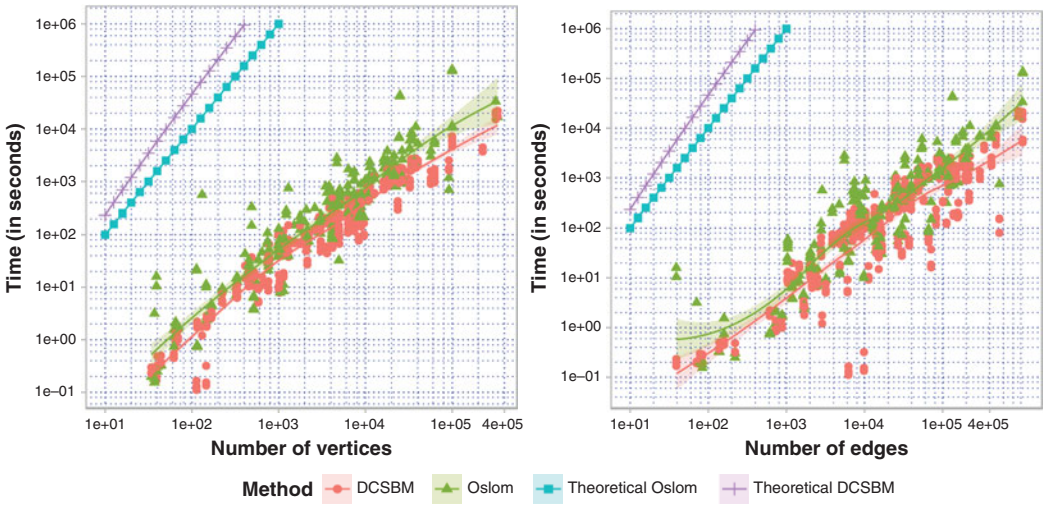


Figure 5. The execution time needed by DCSBM and OSLOM methods to identify community structures on networks of the dataset.

resource devices. Finally, theoretical and practical time seem to find a consensus as the increments of time according to network size are quite consistent in the three cases.

The next group includes methods using modularity optimization processes whose experimental measures are shown in Figure 3. Practically, the three methods in this family require a reasonable time for calculating community structures. The longest experiment took less than 2 h for a graph of 1 million edges. The *Louvain* method is the fastest in this group; its computation increases approximately in linear time. It took only 9 s for the largest graph. Among the three methods, the optimization using the spectral approach is the most expensive. However, all three methods perform better than the methods in the edge removal group previously stated. The experimental results also verify theoretical estimates concerning the complexity of these methods.

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

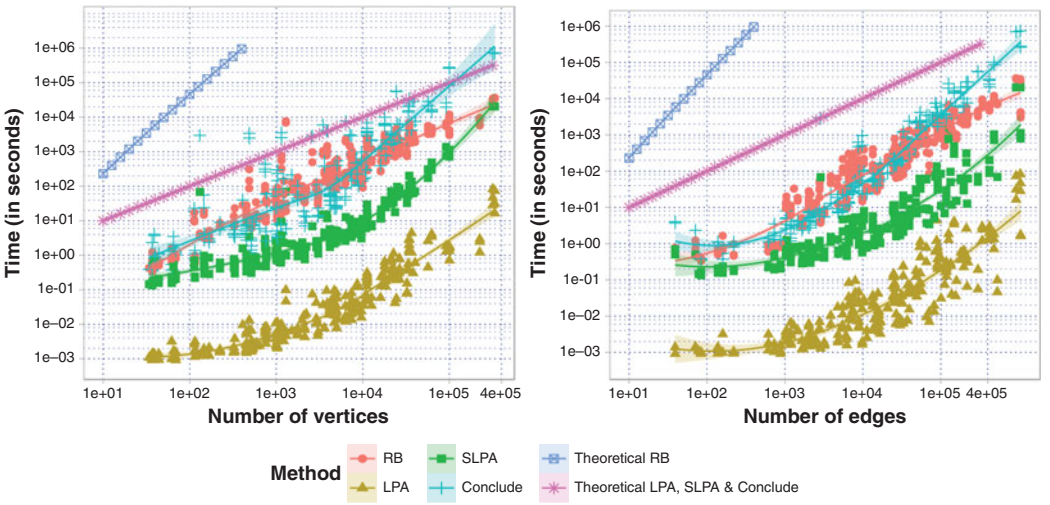


Figure 6. The execution time needed by RB, LPA, SLPA, and Conclude methods to identify community structures on networks of the dataset.

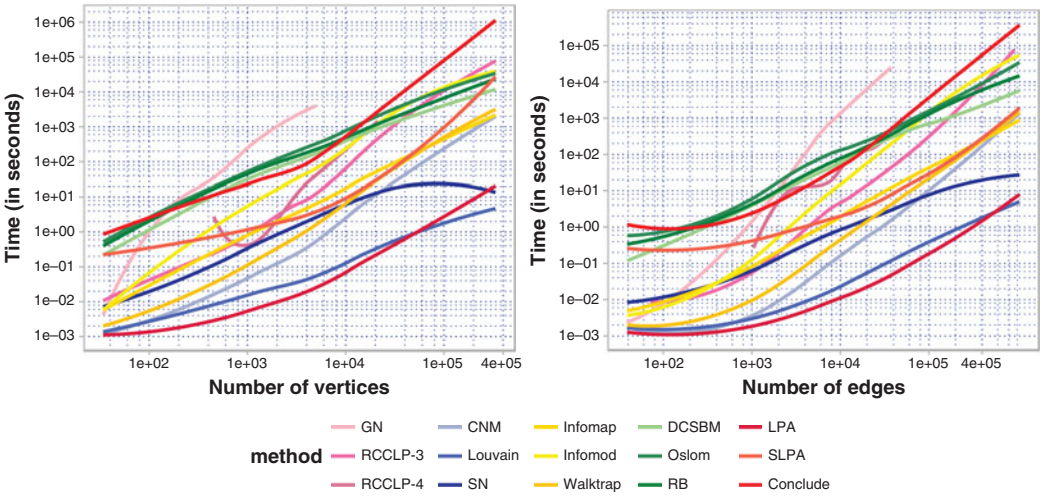


Figure 7. The estimated execution time needed for each method to identify community structures on networks of the dataset using a local regression model. Methods of the same theoretical family (in the same group) are represented by similar colors.

Similar to the two previous groups, the computation time needed by methods in the dynamic process group is illustrated in Figure 4. In terms of time consumption, this group performs better with respect to the first group, but generally worse than the modularity optimization group (except for the *Walktrap* method for small-size and average-size graphs). Among these three methods, *Infomod* generally requires more calculation time than the others. At the same time, *Walktrap* and *Infomap* work asymptotically equally well with a slightly better rendition for *Walktrap* in small-size and average-size graphs.

The same analyses for methods in the two final groups are shown in Figures 5 and 6. We can easily see that *DCSBM* and *OSLOM* have practically identical performances in terms of

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Table 3. Ranking of analyzed methods according to the amount of time consumed, to identify community structures on networks of the dataset.

Method label	Rank by average	Rank by median	Scalability
RCCLP-3	9	8	Low
CNM	5	3	Medium
Louvain	1	2	High
SN	3	5	High
Walktrap	4	4	High
Infomod	12	9	Low
Infomap	6	7	Medium
OSLOM	11	14	Low
DCSBM	8	12	Low
RB	10	13	Low
LPA	2	1	High
SLPA	7	6	Medium
Concude	13	11	Low

time consumption, *DCSBM* being slightly better. In the last group, the results are quite different between different methods. The *LPA* method shows a clear distinctive curve indicating its out performance compared with the other methods. Besides, *SLPA* works quite well, but not as fast as *LPA* although it employs some additional techniques to reduce the number of necessary calculations (Xie & Szymanski, 2012). This difference in performance is due to the more complicated mechanism that *SLPA* uses in comparison to *LPA*. It is due to the fact that *SLPA* manages dedicated memories for all the nodes to stock the membership information and update it regularly. Therefore, despite the improvement of 5 to 10 times in the label update strategy, the global performance cannot surpass that of *LPA* method. In terms of scalability, *LPA* and *SLPA* seem to exhibit the same behavior which is almost linear for small and medium graphs but increases in large graphs. The spin glass model *RB* manifests a better than expected presentation with an undeviating linear augmentation. The only unexpected behavior is spotted in *Conclude* method, as when the size of input graphs exceeds some thousands, the required time has been inflated by a factor of n, making it very demanding for large graphs.

Finally, we aggregate all the results into a common illustration as shown in Figure 7. At the same time, for more convenient observation, we remove all the points corresponding to each experiment and keep only the regression curves, which are the execution time estimates as a function of the number of vertices on the left side and similarly on the right side for the number of edges. At first sight, it is easy to see that except for *GN*, the necessary execution time for all other methods is limited in a range that increases polynomially with the network size, which accurately reflects theoretical estimates. This range is upper-bounded by *Conclude*/*OSLOM* and lower-bounded by *LPA*, corresponding to the worst and the best tested method(s), respectively. Another important fact that can be deduced from this figure is that, for most real-world networks in the range up to 1 million edges, choosing a fast detection method could economize an order of 10^3 times to 10^5 times the calculation effort. This is an important element to be considered where time consumption is a serious problem.

We demonstrate in Table 3 the ranking of these methods according to our tests for reference purposes. *GN* and *RCCLP-4* are not involved in this ranking since they failed to accomplish their tasks in large graphs, which also means they are the most time-consuming methods within the methods that we analyzed. We show both the rankings according to the average and the median of time. Since the average-time ranking is heavily affected by the measurements on large graphs, the

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

methods that were successful in discovering communities on very large graphs are ranked lower than methods that were not able to do so. In these cases, the median ranking is more accurate and reflects therefore the relative performance on small and medium graphs. For large graphs, we recommend using the average ranking.

3.2 Analysis on community size distribution

After these performance considerations, we focus on the nature of the results produced, that is, the communities themselves. The number of latent communities that should be induced from a given network is one of the major questions in community detection context (Fortunato & Hric, 2016; Riolo *et al.*, 2017). It is equivalent to the subject of the expected number of clusters in a classical clustering problem. Observing the number of communities reveals useful information about the mesoscopic structure of a network. The variation of the number of communities in a network involves different levels of resolutions. An analogous way to describe the concept of resolution is the distance from an object that we prefer in order to contemplate it. The closer you get to an object, the greater the detail of its microstructures that can be perceived, while, at the same time, information about the global organization tends to be less clear. Although several multiresolution approaches (Lambiotte, 2010; Pons & Latapy, 2011) incorporate resolution parameters into their solutions providing more flexible mechanisms and different modular scales of networks, it is not always obvious to regulate these parameters appropriately without *ad hoc* cases. The inclusion of multiresolution parameters, of course, widens the possibility of understanding networks, but at the expense of automation convenience, which is sometimes required in clustering problems.

In this section, we compare, once again, the previously mentioned methods but this time according to their resolution abilities. We use the same dataset collection, and again we keep all default configurations of the implementations unchanged to ensure the consistency of future results. From the previous analyses, some modifications will be applied to our testing process as follows:

1. From the observation of the network size distribution in Figure 1(a), as well as the previous computation time analyses, the linear relation between number of vertices and number of edges of networks in our corpus becomes clear. As a consequence, it will be redundant to address the relation of dependent variables in respect of these two latter predictors. Therefore, only analyses according to the number of vertices will be rendered.
2. In the case of the community detection problem, showing only the numbers of communities discovered would not always be enough. If we assume that community size in an arbitrary network exhibiting a negative power law distribution (as shown for example in Clauset *et al.* (2004)), it means that the number of communities depends heavily on the number of tiny communities. Therefore, we propose to observe the distribution of community size to discern the differences between methods which could not be recognized by seeing solely the number of blocks.
3. Due to the huge number of calculations required and a limited hardware resource, discovering processes in the last section was interrupted unless they could be finished in a few hours. Here, some more efforts have been made, when a method is supposed to be finished in a reasonable amount of time (a few days).

For a given network in the dataset, we applied all of the methods presented, to identify the set of communities predicted by each one and then measured their volumes. Similarly, to the previous section, for simplicity of observation, we group methods by different families depending on their approaches. We illustrate the results in Figures 8, 9, 10, 11, and 12 by using some conventions as follows:

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Conventions for Figures 8, 9, 10, 11, and 12

1. A figure (denoted *a*) at the top contains the following three subfigures:
   (a) The central Figure (a1) shows a scatter plot of the distribution of community size. The solid lines in the figure represent the estimated average community size using a local regression model (Cleveland, 1979). Dark colored backgrounds around the lines are 95% confidence intervals of the estimates.
   (b) The top Figure (a2) exhibits marginal density distributions of communities found in each range of network sizes. They are rendered by a Gaussian kernel estimator.
   (c) The right-hand Figure (a3) illustrates another type of marginal density distribution of communities, as a function of their sizes. They are also rendered by a Gaussian kernel estimator.
   (d) The axes of marginal figures are the same as the axes of the corresponding central figure. We simply omit them for ease of representation.
2. A figure on the bottom (denoted *b*) presents the number of communities as a function of network size (i.e., their number of vertices) as well as the estimated relation between these variables using the regression model stated above. Dark colored backgrounds around the lines show 95% confidence intervals of the estimate relations.

3.2.1 Edge removal approach: GN, RCCLP-3, and RCCLP-4

From Figure 8, we can notice again that the *GN* method is only able to function on small and medium networks, due to its high complexity, which is quite obvious from theoretical analysis. *RCCLP-3* and *RCCLP-4* can detect up to the largest networks in our corpus. By observing the right marginal density distribution, all of these methods identify a surprisingly high number of singleton communities. The average number of singleton communities is around 24%, but the number can reach 60% in some cases. The reason for this aberrant phenomenon is that in some dense, small networks, there exist too many high and equivalent central vertices and edges. The separating mechanism employed here keeps removing central nodes or edges until a large number of vertices are isolated, creating singletons or very small communities. Since *GN* only works on small graphs, it is highly impacted by this phenomenon in our experiment. Besides, from a global observation, we can see in the top figure that the majority of communities detected by these methods are very small for the same reason. From Figure 8(a), we can see that a large number of communities have less than 10 vertices, even in very large networks. This makes the number of communities increases rapidly, as illustrated in Figure 8(b). Bear in mind that the distributions of community size have right-skewed shapes, meaning that the majority of communities are small, and most are found under the lines of average community size. Therefore, the three methods of this family have very high resolutions. Notwithstanding, this result needs to be interpreted with caution, for the following reasons:

1. The density function in Figure 8(a1) reveals that the successful rates for discovering community structures of the three methods are fundamentally very different. In fact, due to the high complexity of time and memory, many networks are not successfully resolved, which significantly degrades the comparison quality.
2. As a consequence of the first reason, there is a high fluctuation in the dependent variables, which makes the confidence intervals quite large. A deeper investigation of the quality on small and medium networks could partially palliate this problem.

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

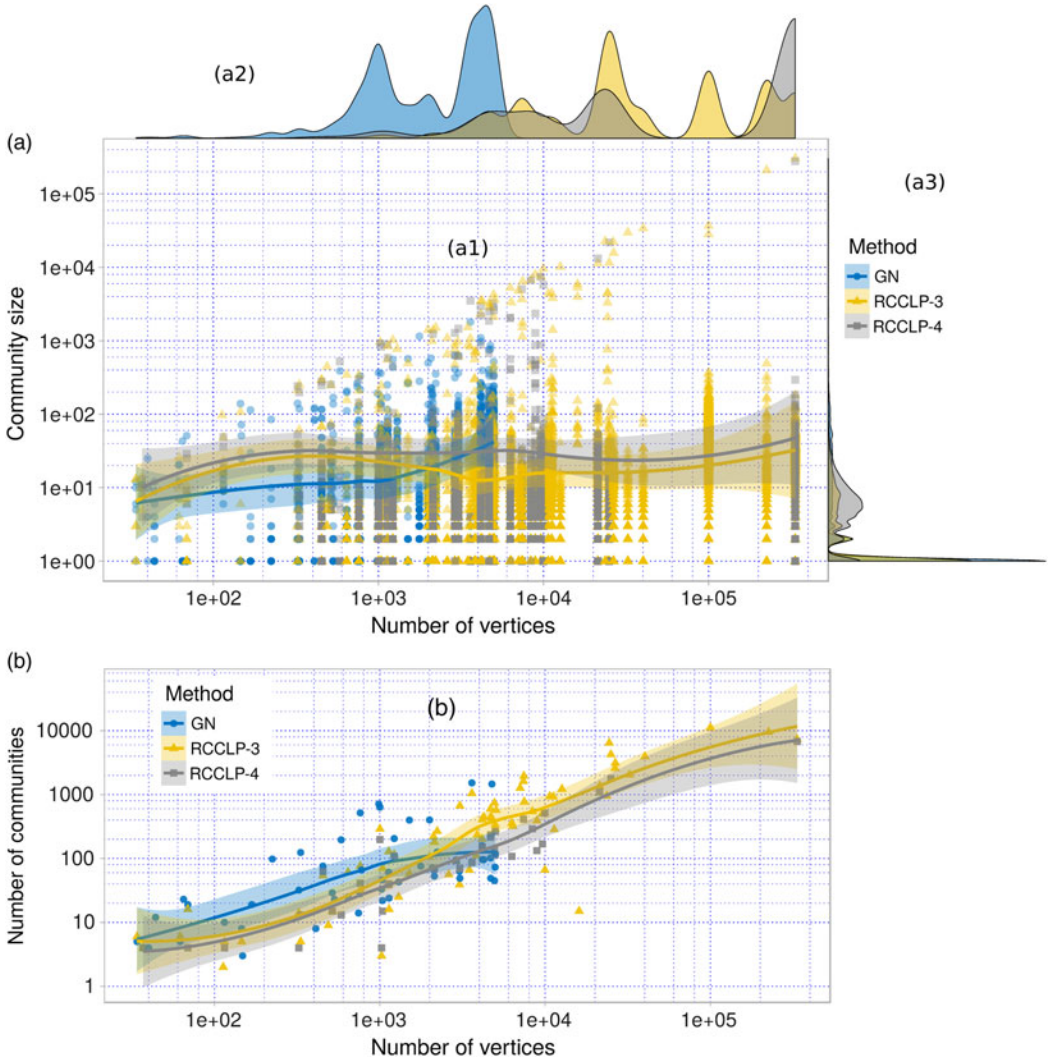


Figure 8. Fitting quality of GN, RCCLP-3, and RCCLP-4 methods: number of communities and community size.

Despite the previously mentioned issues, within this class of methods, there is a large consensus as regards the discovery of the highest number of communities.

3.2.2 Modularity optimization approach: CNM, Louvain, and SN

In this second group, our measurements are more complete since all three methods successfully resolved large networks. From Figure 9(a2), it can be seen that there is a regularity between the distributions of communities over the whole range of networks, except for the range of very large networks. Actually, in this range, the behavior is very different in the three methods. While *CNM* determines a very large number of medium and small communities, *Louvain* identifies fewer small communities and more medium and large communities. On the other hand, *SN* only proposes a partition of two giant communities. For instance, if we take the *Amazon* network (Leskovec & Krevl, 2014), while *CNM* detected 1,480 clusters, the number is 249 for *Louvain*, and only 2 for *SN*. The same phenomenon is also identified for another example, the *DBLP* network (Leskovec

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

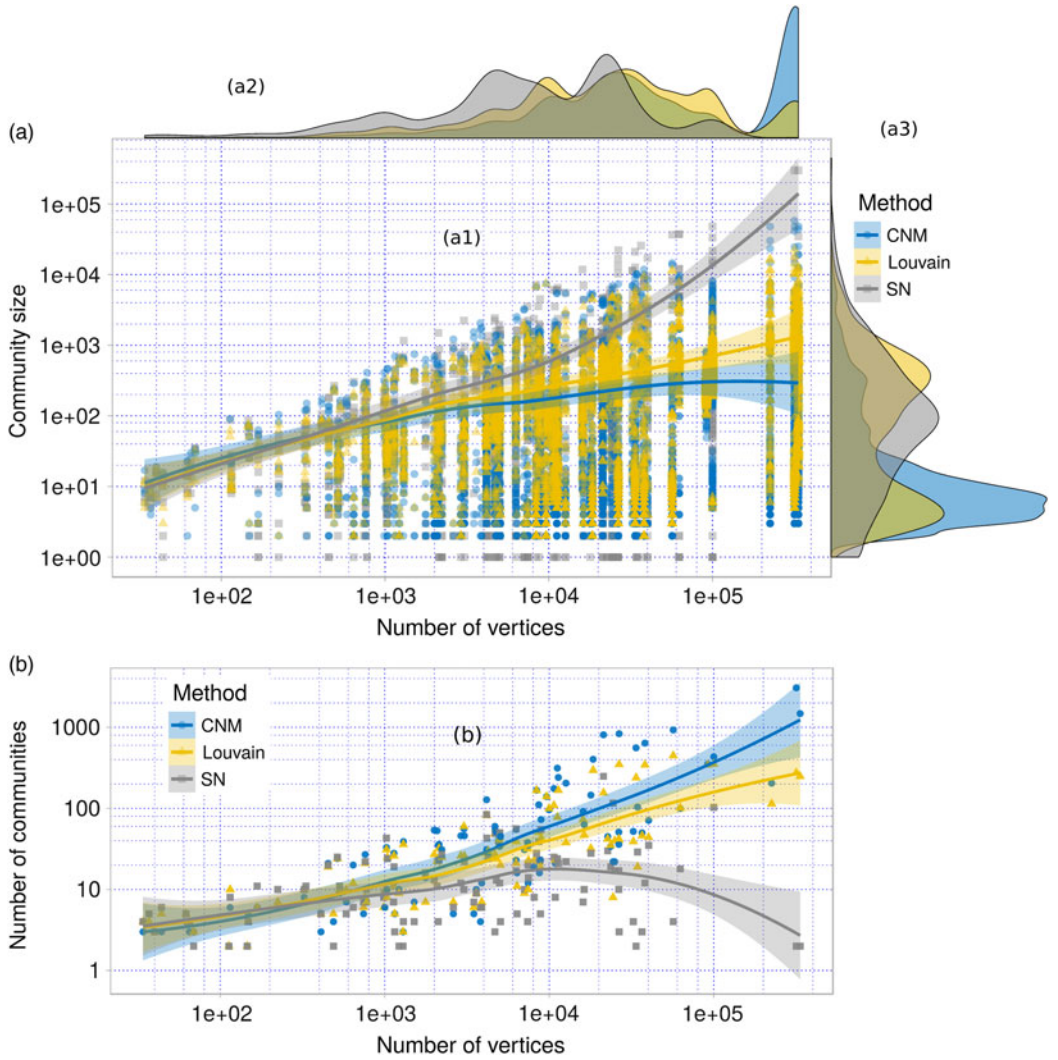


Figure 9. Fitting quality of CNM, Louvain, and SN methods: number of communities and community size.

& Krevl, 2014). The corresponding numbers are 3, 077, 275, and 2 in the same order for the three methods. This fact can also be remarked in smaller networks as can be seen in Figure 9(b). However, the gap between the number of communities reduces gradually from the right to the left of the figure. But, in general, the order remains unaltered as experienced in our observations, that is, the average number of communities detected by *CNM* is larger than that of *Louvain*, which is in turn larger than that of *SN*. Consequently, the order of community sizes is inversed, since the sizes of graphs are fixed, as can be seen in Figure 9(a1). Another fact can be extracted from Figure 9(a3) concerning the diversity of community size: while *CNM* and *SN* consistently move toward small and medium communities, respectively, *Louvain,* on the other hand, tends to propose both small- and medium-size communities.

3.2.3 Dynamic process approach: Infomap, Infomod, and Walktrap

At first glance, we can see a clear separation within the three methods. While *Infomap* and *Walktrap* display quite a comparable evolution of average community size, depicted by

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

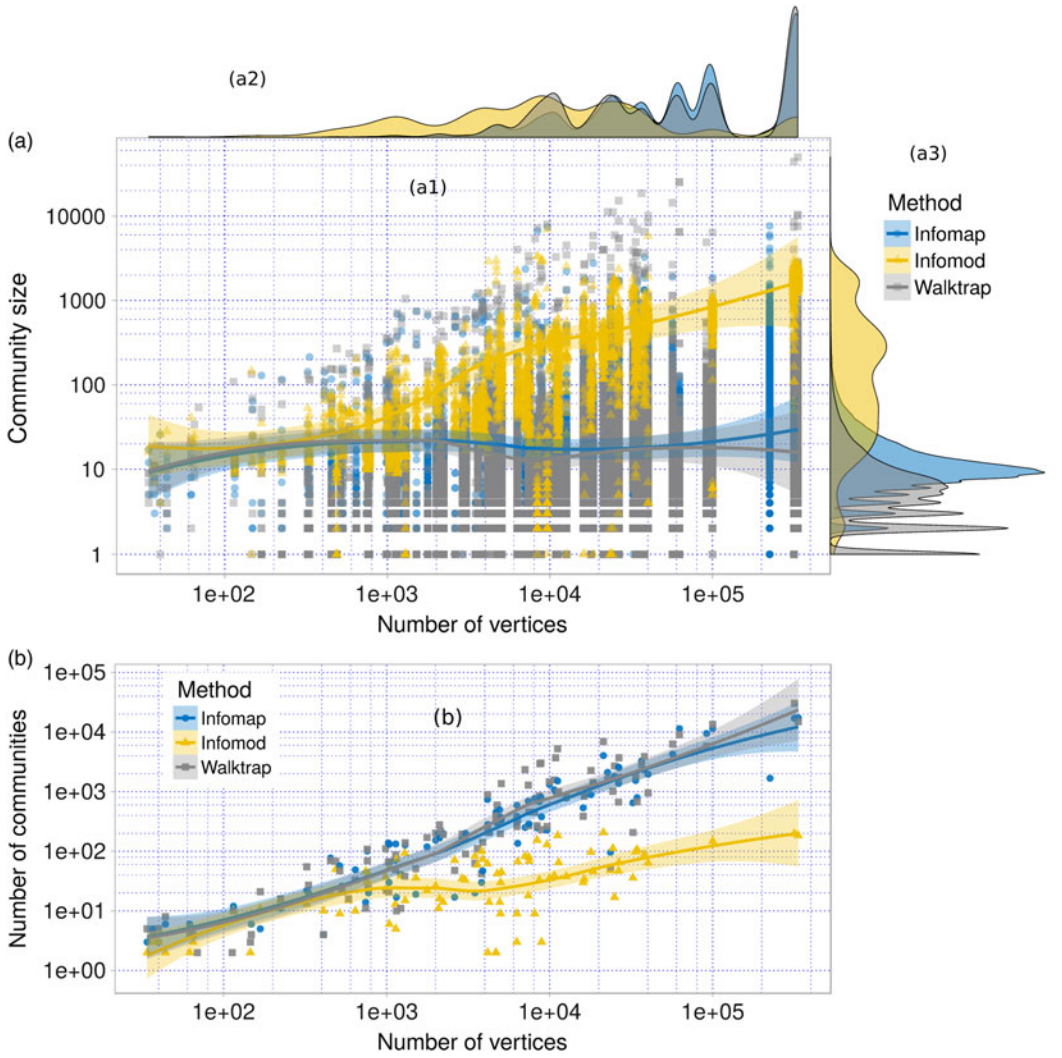


Figure 10. Fitting quality of Infomap, Infomod, and Walktrap methods: number of communities and community size.

Figure 10(a1), as well as marginal distribution, as depicted by Figure 10(a2) and (a3), *Infomod* is driven distinctly apart. After close examination, we notice that in *Infomod*, there is a relatively uniform partitioning of communities which is upper-bounded by the largest community containing 6, 948 vertices. Unlike many other methods including *Infomap* and *Walktrap*, the number of medium and large communities discovered by *Infomod* does not outnumber the number of small communities, as stipulated by heavy-tailed distributions. As a consequence, the total number of communities observed remains low and increases at a slow, constant pace.

Infomap and *Walktrap* tend to keep their average community size limited to around 10 to 30, over the whole range of networks. This phenomenon keeps them apart from the resolution limit issue. In both methods, the most popular community size can be found around 10 nodes or smaller. Our more specific experiment on the median community size shows almost similar results for *Infomap*, while the number decreases slightly for *Walktrap*. Above these values, the number of communities decreases significantly. The biggest difference between these two methods can be easily observed at the spurious region on the marginal distribution of Figure 10 (a3). In fact,

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

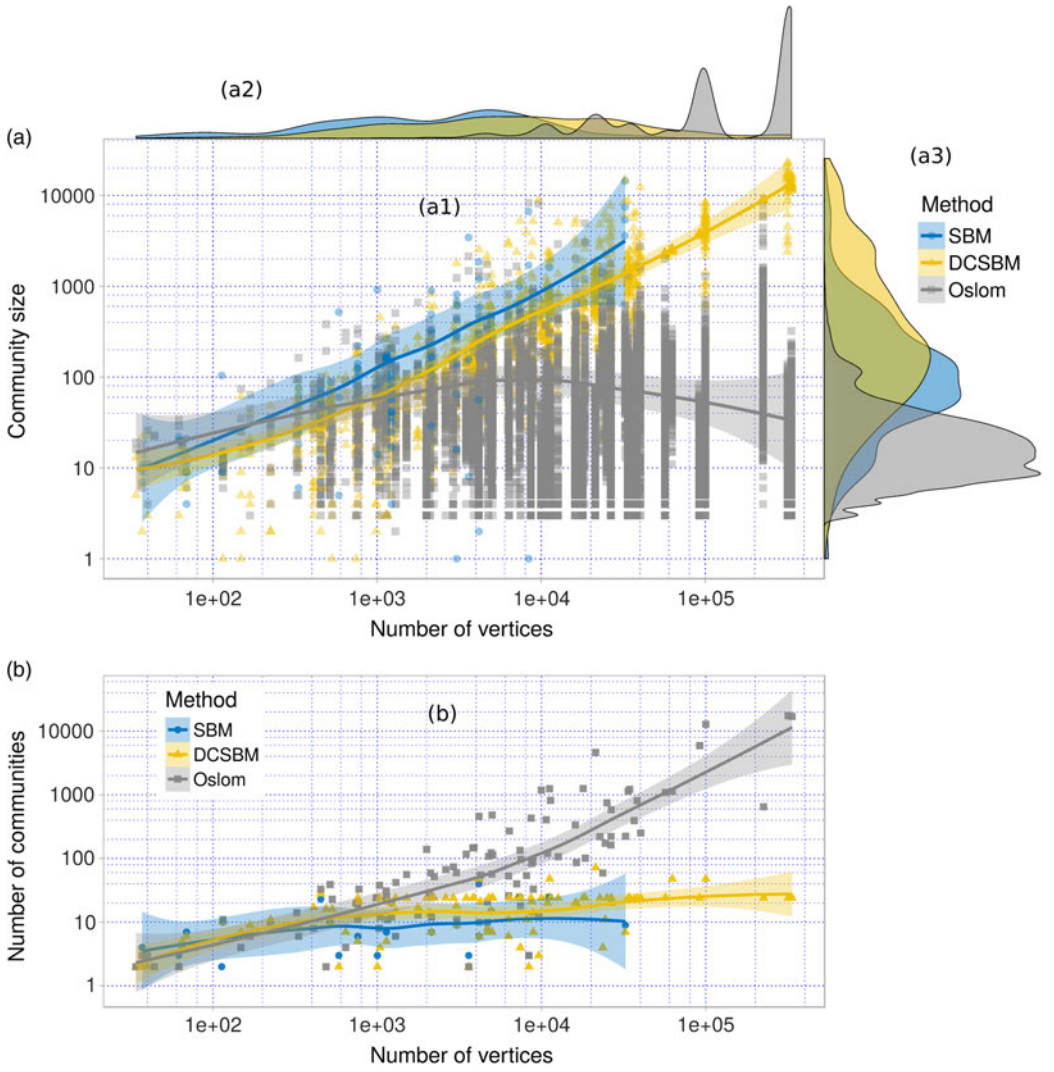


Figure 11. Fitting quality of SBM, DCSBM, and OSLOM methods: the number of communities and community size.

unlike *Infomap*, which produces moderately small communities, *Walktrap* identifies a huge number of isolated nodes (around 10% according to the statistics), similar to *RCCLP-3* and *RCCLP-4*, as indicated in the previous section. This problem may be due to the agglomerative hierarchical clustering employed by *Walktrap* to detect communities, which engenders orphaned peripheral vertices. This behavior has been identified in particular by Newman and Girvan (cf. Figure 3 in Newman & Girvan (2004)). This problem, however, is quite simple to palliate since these peripheral vertices could be assigned to their closest neighbor's community. By removing this issue, we obtained quite a similar result for *Infomap* and *Walktrap*.

In terms of average number of communities, *Infomap* and *Walktrap* show practically the same behavior. The evolutions coincide across almost all networks with small confidence intervals, particularly in the mid-range networks. For medium and large networks, as seen in Figure 10(b), it is very likely that *Infomod* identifies a much smaller number of communities. In fact, more than 75% of *Infomod*'s partitions have fewer communities than those of the other two methods.

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

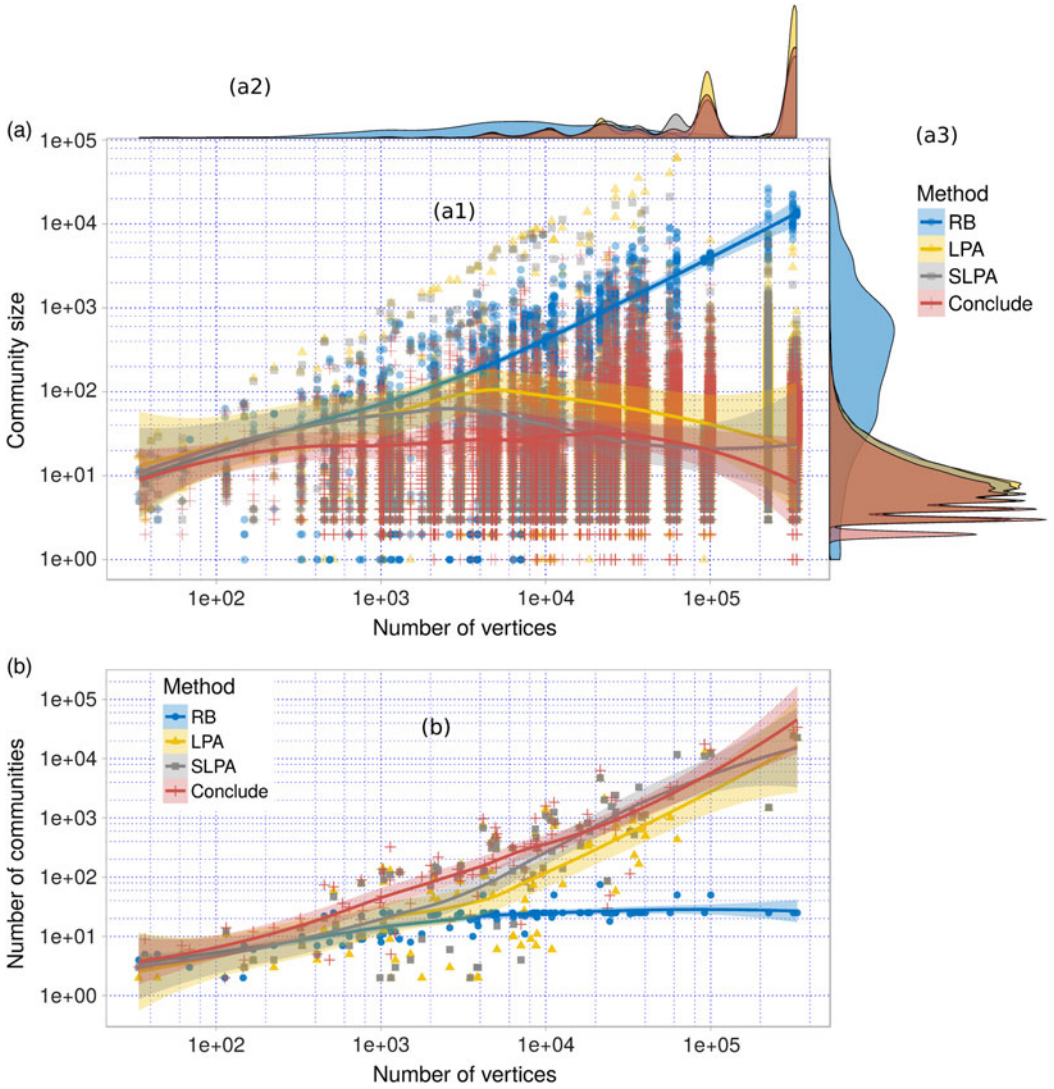


Figure 12. Fitting quality of RB, LPA, SLPA, and Conclude methods: number of communities and community size.

3.2.4 Statistical inference approach: SBM, DCSBM, and OSLOM

In the case of statistical inference, we see quite a similar phenomenon to that previously experienced in the dynamic approach. Specifically, the distributions of community size of the two implementations *SBM* and *DCSBM* nearly coincide with a slightly higher average community size for the former. In fact, in this Bayesian block model, it is necessary that the prior distribution of number of blocks be given. In the implementation that has been employed, the authors initialize the community discovering process by assigning nodes randomly to groups, according to a queuing-type mechanism, and then use a Monte Carlo sampling process to maximize the posteriori probability. However, the calculation becomes extremely time-consuming when the maximum number of communities is too large (Riolo *et al.*, 2017). Hence, by default, the maximum number of communities is configured at 25, as proposed by the authors, which leads to an underestimation of medium and large graphs, as shown in Figure 11(b), and also noticed by the authors. One can see the impact of this regulation as the number of communities approaches asymptotically

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

25, independently with the network size on the right-hand side of the figure. Discovering community structures using this method in large networks (more than 25^2 nodes, for example) would have to be done recursively to avoid resolution limits. In other words, one could apply community detection again on very large communities (larger than the square root of the number of nodes).

By observing the distribution of community size in Figure 11(a1), it is understandable that the average block size of *SBM* and *DCSBM* increases linearly according to the number of vertices. As the number of communities remains constant, the average community size must increase proportionately. Furthermore, Figure 11(a3) also reveals that community sizes are well distributed around their mean values, which makes the marginal distribution quite symmetric for both *SBM* and *DCSBM*. There is almost no particular inclination toward small communities, as acknowledged in some previous methods.

For the case of *OSLOM*, the separation is quite clear. It uncovers many more communities, making their sizes very small. Figure 11(a1) shows that the majority of *OSLOM* communities are found under the average values of the associated partitions of *SBM* and *DCSBM*. Our demonstrations show that there is indeed a significant difference in the partitioning strategies of these methods.

3.2.5 RB, LPA, SLPA, and Conclude methods

In the last group, we discover that there is a remarkable coincidence in all distributions of the three methods *LPA*, *SLPA*, and *Conclude*. In fact, the difference between them is almost indistinguishable on the marginal measures. There is only a small discrepancy in the number of detected communities in very large networks, as can be seen in Figure 12(a2), such that *LPA* detected slightly more communities than *SLPA* and *Conclude*. From Figure 12(a3), one can see that the majority of communities are quite small in these three methods. Similar to *CNM*, *Infomap*, or *Walktrap*, the majority of communities are small, that is, have less than 10 nodes.

In the three methods, one can see that the variation of the data is significantly large, which also produces a large variation in our estimates. Since the associated prediction intervals for the estimates are likely to be larger, predictions related to community size distribution are not expected to be accurate.

On the other hand, the *RB* method shows a solid consistency with far fewer variations in our examination. Average community size increases regularly and the number of communities becomes saturated from medium-size networks. The behavior of the *RB* method closely resembles that of *DCSBM* observed in Figure 10. Consequently, it is supposed to suffer the resolution limit for large networks. Nevertheless, since *RB* is provided with a resolution tune parameter, the method may avoid this effect if the parameter is correctly chosen.

3.2.6 Summary

For the final step in this section, in the same manner as the previously presented time computational analysis, we aggregate, for all methods, the estimates of average community size and the number of detected communities, as a function of number of vertices in the network in Figures 13(a) and (b), respectively. One can see that there exist several partitioning strategies hidden in these methods. If we use the preference of a theoretical number of recoverable communities in a k-planted partition model (Ames, 2013), being $O(\sqrt{n})$, the methods studied could be considered to overfit (create more than k clusters) or underfit (create less than k clusters) as presented in Table 4, in the third column.

With an overview of the second and third columns of Table 4, methods belonging to the same theoretical class, which shares a common assumption about the definition of community, have a tendency to show the same fitting quality. This has also been identified by Ghasemian *et al.* (2018). However, although being useful to help practitioners to presume the expected number of clusters

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

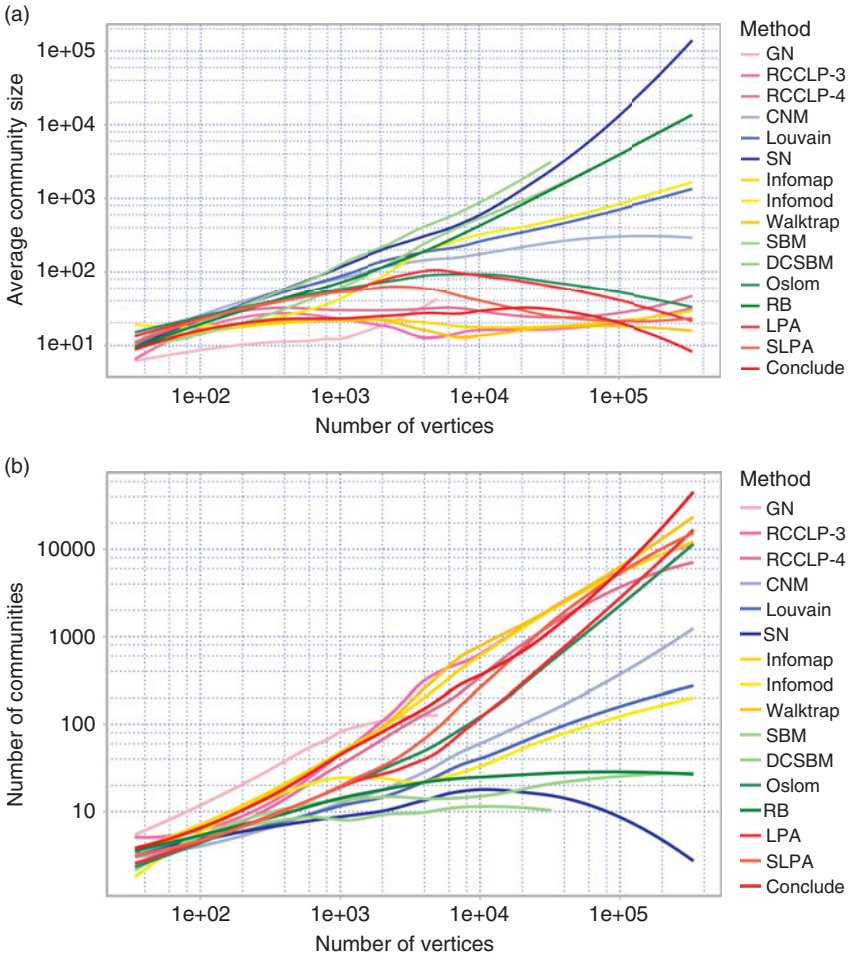


Figure 13. A summary of community size estimation.

which a method would detect with respect to the theoretical experience, it is still very challenging to decide which method to use, since the reference is based on a hypothesis about an underlying model. This also means that if the hypothesis about the partition model changes (another model than k-planted model), the expected number of communities will be diversified, and hence the indicated fitting quality preference becomes disproved. As a consequence, in the next section, we propose a novel technique to estimate the similarity of community detection methods based on community size distributions.

3.3 Similarity based on community size distribution

A very naive but efficient approach to evaluate the similarity of two methods is to inquire into the "*closeness*" of the two corresponding community size distributions (Dao *et al.*, 2018b). Two methods could be supposed to be similar, if their corresponding density distributions expose a large intersection area, as shown in Figure 14(a). From this notice, we can define our new similarity function as follows:

First, we denote two 2-tuples $(\mathscr{A}, n^a)$ and $(\mathscr{B}, n^b)$ being the multisets representing all communities detected on a set of networks $\mathscr{G} = \{G\}$ by method A and method B, respectively, where $\mathscr{A} = \{x_1^a, x_2^a, ..., x_r^a\}$ and $\mathscr{B} = \{x_1^b, x_2^b, ..., x_s^b\}$ being the ascending ordered sets of sizes of communities:

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Table 4. Ranking of analyzed methods according to their number of detected communities. A method is considered to overfit if it detects asymptotically more than $\sqrt{n}$ clusters. The group numbers exhibit the estimated similarity based on fitting quality.

Method label	Size wrt. k-planted model	Fitting
GN	Bigger	Overfit
RCCLP-3	Bigger	Overfit
RCCLP-4	Bigger	Overfit
CNM	Close	Overfit
Louvain	Close	Underfit
SN	Smaller	Underfit
Walktrap	Bigger	Overfit
Infomod	Close	Underfit
Infomap	Bigger	Overfit
OSLOM	Smaller	Underfit
SBM	Smaller	Underfit
DCSBM	Smaller	Underfit
RB	Smaller	Underfit
LPA	Bigger	Overfit
SLPA	Bigger	Overfit
Concude	Bigger	Overfit

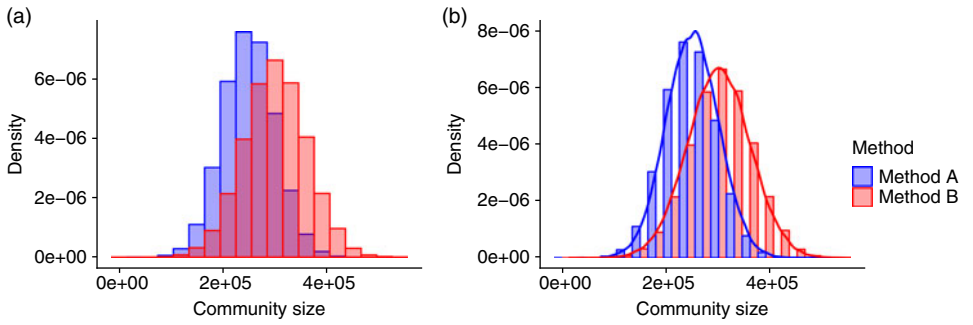


Figure 14. The distribution of sizes of communities detected by two different methods. On the left (a) overlap fraction using a histogram, on the right (b) when community sizes interlace, the similarity is better estimated using a kernel density estimator.

$1 \le x_1^a < x_2^a < \ldots < x_r^a$ and $1 \le x_1^b < x_2^b < \ldots < x_s^b$. The multiplicity functions $n^a : \mathscr{A} \to \mathbb{N}_{\ge 1}$ and $n^b : \mathscr{B} \to \mathbb{N}_{\ge 1}$ measure the number of communities of sizes x_i^a and x_i^b, respectively. Let $N^a = \sum_{i=1}^{r} n^a(x_i^a)$ and $N^b = \sum_{i=1}^{s} n^b(x_i^b)$ being the total number of communities of all sizes detected by each method, we define a similarity function describing the closeness of A and B on $\mathscr{G}$ as:

$$S_{\mathscr{G}}(A,B) = \frac{1}{2}\sum_{i=1}^{r}\sum_{j=1}^{s} \min\left\{\frac{n^a(x_i^a)}{N^a}, \frac{n^b(x_j^b)}{N^b}\right\} \delta(x_i^a, x_j^b), \tag{1}$$

where $\delta(x_i^a, x_j^b) = 1$ if $x_i^a = x_j^b$ and 0 otherwise. Equation (1) is simply the common fraction of same-size communities detected on $\mathscr{G}$ by both A and B: $0 \le S_{\mathscr{G}}(A,B) \le 1$. This definition seems to be intuitive but does not work well in practice. As illustrated in Figure 14(b), when the sizes interlace with each other, a low score will be produced, although the level of similarity in this case is that of Figure 14(a). Choosing an appropriate binning interval would mitigate the problem.

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

This solution is quite inflexible, however.A straightforward alternative can be envisioned by using a kernel density estimator to uncover the probability density function as shown by the solid lines in Figure 14(b). In this way, we approximate the common fraction of same-size communities of Equation (1) by the overlapping area of two corresponding continuous distributions. The premise behind this estimation is that two similar methods do not always produce a large portion of exact same-size communities but rather a large portion of comparable-size ones. Hence, we consider the following estimator to take into account local information of community size x_0:

$$\widehat{f}(x_0) = \frac{1}{hn} \sum_i K\left(\frac{x_i - x_0}{h}\right), \tag{2}$$

where h is the bandwidth controlling the neighborhood interval around x_0 and K is the kernel function controlling the weight given to the observations $\{x_i\}$, chosen as Gaussian in our analysis. One would wonder why we use a Gaussian, whereas Power law may be a better fit, as some papers (e.g. (Clauset *et al.*, 2004)) show that community size in many real-world networks exhibits a power law distribution. We would like to recall that expecting such ground-truth-like partitions does not mean that community detection methods impose this property into their mechanism and produce power law distribution. For example, on the Zachary network, although we try to discover two equivalent-size groups, community detection methods identify two to four similar-size communities. Some methods indeed fit a power law, but as a matter of fact, many others have a tendency to partition network nodes into quite balanced groups (spectral methods, modularity optimization methods, SBM, etc.) as seen in the previous section. This led us to use a Gaussian kernel, as it fits well in most of our cases on our large dataset and our panel of methods. In a less generic and exhaustive context, when the resulting community size varies significantly, following a power law distribution, we, of course, recommend using the appropriate law.

Using our estimator, we rewrite the similarity function defined in Equation (1) as follows:

$$S_{\mathscr{G}}(A, B) = \int \min\{\widehat{f}^{(a)}(x), \widehat{f}^{(b)}(x)\} dx, \tag{3}$$

where

$$\widehat{f}^{(u)}(x) = \frac{1}{hN^u} \sum_i^{N^u} \left[n^u(x_i^u) K\left(\frac{x_i^u - x}{h}\right)\right], \tag{4}$$

with $u \in \{a, b\}$. In the estimations of this paper, the bandwidth h is selected based on the normal reference rule (Silverman, 1986) to minimize the mean integrated squared error.

Using Equations (3) and (4) to estimate the similarity between pairs of detection methods on a large dataset will help us to discover different behaviors of community detection methods. Since the accuracy of the estimator depends on the networks of the dataset that we analyze, the result will obviously have to be relativized. However, our large and representative corpus helps to reduce the dependency impact.

3.3.1 Experimental results

From the communities identified in the previous section, we proceed to measure the volumes of communities detected by each method to determine the elements of the corresponding 2-tuples. Finally, we use the similarity function defined by Equation (3) to estimate the closeness between each pair of methods. Due to the huge number of experiments, only processes having a reasonable theoretical estimated time and memory consumption are maintained (less than a few days and at most 30–40 GB of memory). The outcome distributions are illustrated in Figure 15.

As we can see, there is a clear difference in the densities of community size, showing that these methods have various partitioning strategies. Knowing that methods belonging to the same theoretical group (as shown in Table 1) are placed next to each other, we can see some agreements between the theoretical families, with practical outcomes as follows:

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

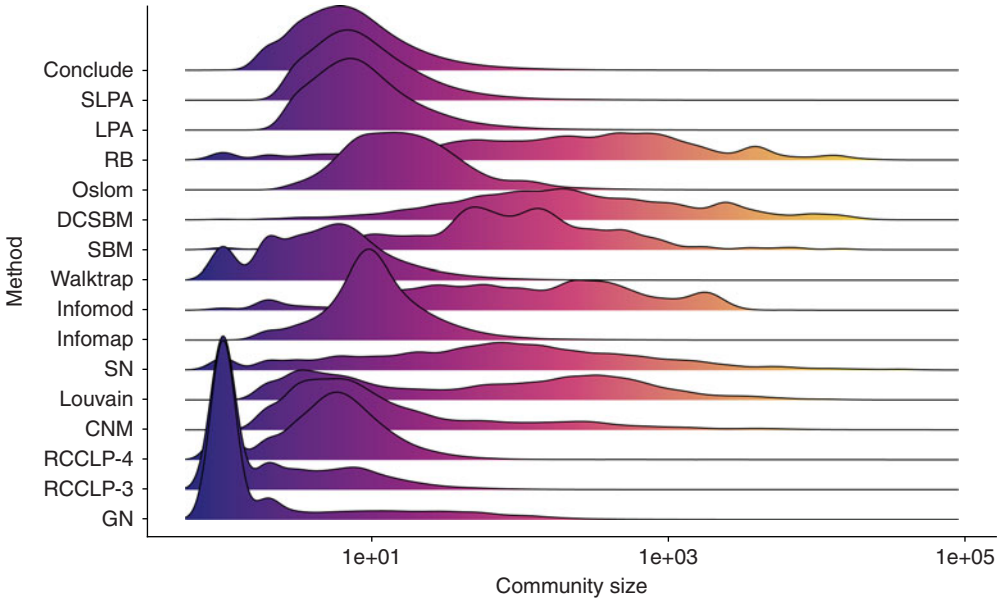


Figure 15. Community size distributions, for all the communities from the partitions detected on all the networks. The distributions are smooth using a Gaussian kernel estimator. The gradient color is used only for ease of observation.

Edge removal: *GN* and *RCCLP-3* have very similar distributions where a large number of communities are very small. This is due to the fact that in some highly local centralized networks, having star-like structures (Dao *et al.*, 2018a), they have a tendency to remove edges connecting hub and peripheral nodes and to create singletons (single-node community). This phenomenon is less distinguishable on *RCCLP-4*, since there are far fewer quadrangular than triangular connections in networks.

Modularity optimization: Modularity is known to suffer from resolution limit phenomenon (Fortunato & Barthelemy, 2006), which often aggregates small communities in large-scale networks. We can see from Figure 15 that *Louvain* and *SN* found very large communities, as predicted. At the same time, there are also a comparable number of small communities that are found on small graphs. However, the behavior is a little different on the *CNM* method, which is an agglomerative clustering algorithm based on modularity optimization.

Dynamic process: Methods in this family show very discernible distributions, although all are based on dynamic processes. In fact, they make different assumptions about community structure and searching mechanisms. Therefore, belonging to the same theoretical family does not lead to a similarity in practical results.

Statistical inference: The Bayesian *SBM* and *DCSBM* use the Monte Carlo sampling process, which is very time-consuming, in order to sweep the solution space. This makes the method unfeasible, if the maximum number of clusters is not limited. Indeed, in the default version, the maximum number of communities is limited to 25, meaning that the (*DC*)*SBM* methods find very large communities in large networks. On the other hand, the OSLOM method uses an agglomerative discovery mechanism and identifies globally smaller communities.

Other methods: In this group, *LPA*, *SPLA* (both based on label propagation), and *Conclude* display almost identical distributions. The *RB* method, being based on a very close concept with modularity (with a tuning parameter), exhibits a similarity with modularity optimization-based methods.

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

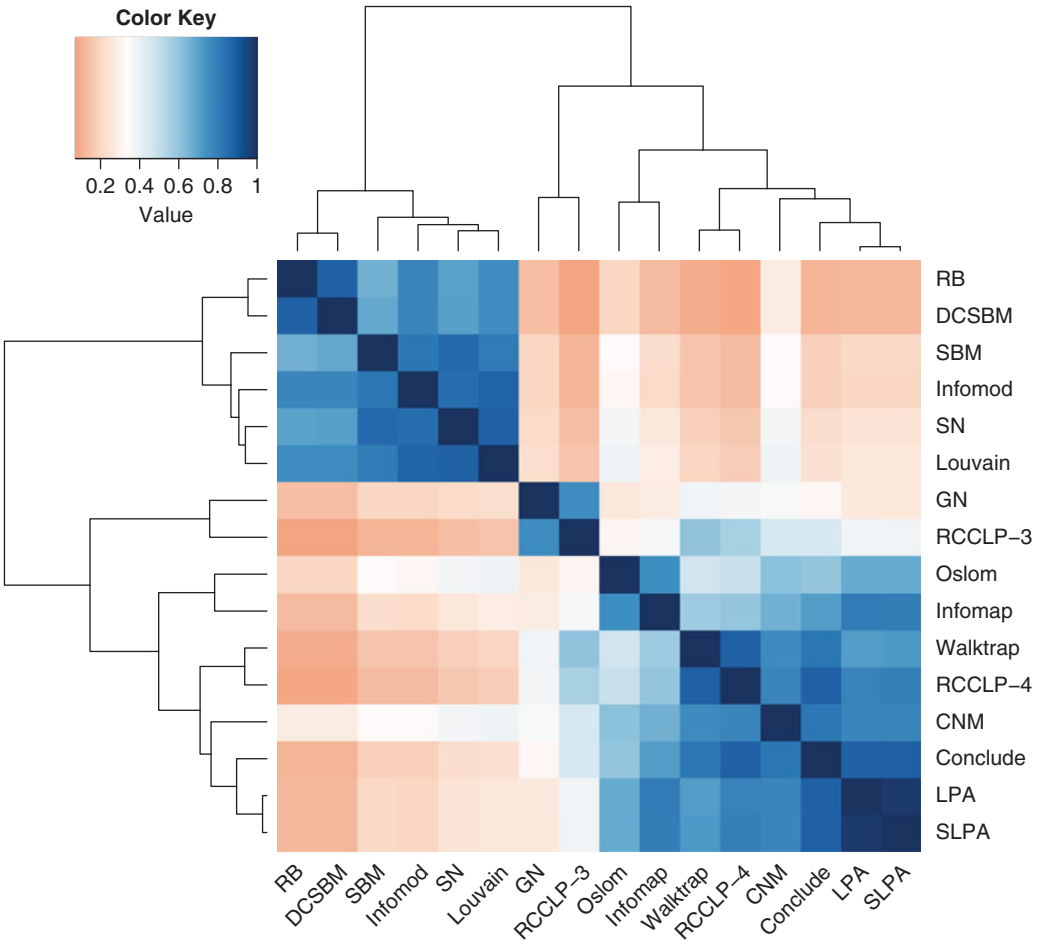


Figure 16. The similarity between community detection methods in terms of size fitting quality. Two methods are considered to be similar if they share a large fraction of same-size communities. Methods are ordered using hierarchical clustering (Joe H. Ward, 1963). The dendrogram proposes a hierarchical structure of the fitting closeness. Blue colors mean high similarity.

Quantitatively, applying the estimator presented in Equation (4), to compute pairwise similarities between the methods, leads us to the results demonstrated in Figure 16.

A well-known method for comparing two distributions is the Kolmogorov–Smirnov test (KS). Under the null hypothesis, the two distributions are identical. The test generates the cumulative distributions of the distributions, reports the maximum difference between them, and calculates a p-value. If the KS statistic (distance) is small or the p-value is high (above the significance level, e.g., 0.05), the assumption that the distributions of the two samples are identical cannot be rejected. Conversely, we can reject the null hypothesis, if the p-value is low. In our case, $p - value < 0.05$ for all the pairs, which means that the distributions do not fit the same model, except for a single pair (*Conclude*, *LPA*) with a value $p = 0.07$. A deeper exploration shows that these two methods do not fit a normal law, but they fit better a power law distribution than other heavy-tailed distributions (with a nonsignificant difference, however). In practice, the KS-distance is useful, if one is mostly interested in how similar two datasets are. As shown in Table 5, the smallest distances occur with couples (SLPA, LPA), (Conclude, LPA), (DCSBM, RB), (RCCLP-4, Walktrap), (GN, RCCLP-3), (Infomod, SN), (Louvain, SN), which is consistent with our results.

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Table 5. Extract of the smallest Kolmogorov-Smirnov distances between community size distributions.

Method	Method	KS-Distance	p-value
SLPA	LPA	0,007	0,017
Conclude	LPA	0,067	0,075
DCSBM	RB	0,070	0,000
RCCLP-4	Walktrap	0,080	0,000
GN	RCCLP-3	0,098	0,000
Conclude	SLPA	0,118	0,000
Infomod	SBM	0,118	0,000
Infomod	SN	0,126	0,000
DCSBM	Infomod	0,127	0,000
Louvain	SN	0,127	0,000

Thus, according to the community size criterion, the methods can be classified into different classes of partitioning strategy. Similar results are also discovered by (Ghasemian *et al.*, 2018), as they found that methodological similar methods usually lead to a similar outcome. The separations are clearly shaped, showing that the distinction is very clear between groups. Therefore, we choose to characterize these methods by three (possibly four) principle groups as follows:

1. **Group 1** - *RB*, *DCSBM*, *SBM*, *Infomod*, *SN*, *Louvain*: Methods in this group discover communities whose sizes vary in wide range of spectra, from very small to very large communities. The characterized community size distribution is quite flat, meaning that all sizes are nearly equally considered.
2. **Group 2** - *GN* and *RCCLP-3*: These two methods identify a huge number of very small communities including singletons regardless of network size. As a consequence, there are few variations in community volume.
3. **Group 3** - The others: These methods produce communities whose sizes approach bell-shaped distribution. The strategy can be translated as: neither left nor right, that is, communities that are neither too small and nor too big.

This characterization could help us to identify appropriate groups of community detection methods, according to different community size fitting strategies. Also, it helps to avoid brute-force attempts when a method does not succeed in proposing desired partitions by proposing substitute solutions. Moreover, by combining with the previous time-computation analysis in Section 3.1, one could also choose a group of methods corresponding to size distinction criteria and then select the fastest method that leads to a desired outcome.

Community distribution (or number of communities) is just one possible quality dimension, albeit possibly one of the most intuitive and important pieces of information when choosing a clustering method. In the next part, we demonstrate some techniques that can be used to define other similarity aspects. We show that these notions of similarity can be combined to accentuate the distinction between different community detection methods.

4. Quality profiling of community detection methods

4.1 Community quality metrics

A popular way to evaluate the structure of communities is to design quality metrics, in order to measure different expected characteristics from subgraphs that we want to obtain. In practice, metrics using network generative models are sometimes preferable, as they reflect different

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

assumptions about the underlying mechanisms that create community structure. One of the most widely used metrics that quantifies the quality of community structure is the *modularity* function. The idea here is to reveal how the quality of an identified community structure is different from that which would be expected. Although some unexpected phenomena, known as resolution limit (Fortunato & Barthelemy, 2006; Traag *et al.*, 2011) have been exposed, when the scale of community size is too small, modularity remains the standard measure of quality.

The advantage of this approach is that one can *"embed"* the assumption of community structure inside quality functions. Hence, they provide better performance in some cases. However, community structure is quite an open question, such that, according to different mechanisms that render the structure of networks, there will be models that are more suitable than others.

We present some quality metrics (also called "community scoring functions" or less often "goodness metrics") to evaluate community structure. Many of them are initially or gradually employed as objective functions in some community detection methods, since they expose good performance in searching processes. First, we recall some notations that will be used to describe structural characteristics of communities. A graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ consisting of $n = |\mathscr{V}|$ vertices and $m = |\mathscr{E}|$ edges can be represented by an associated adjacency matrix A. Given a community C of n_C vertices being a subgraph of $\mathscr{G}$ in an arbitrary partition P, a function $f(C)$ or $f(P)$ quantifies a structural goodness feature of community C or the whole partition P, according to a particular expectation of community structure. Let m_c be the number of edges inside community C, $m_C = |(i,j) \in \mathscr{E} : i \in C, j \in C|$, and l_C be the number of edges that connect C to other vertices outside of C, $l_C = |(i,j) \in \mathscr{E} : i \in C, j \notin C|$. Any vertex i belonging to community C has an *internal* degree k_{iC}^{int} and an external degree k_{iC}^{ext} satisfying $k_{iC}^{int} + k_{iC}^{ext} = k_i$, where k_i is the total degree of vertex i. The internal and external degrees can be expressed via the adjacency matrix A as: $k_{iC}^{int} = \sum_{j \in C} A_{ij}$ and $k_{iC}^{ext} = \sum_{j \notin C} A_{ij}$. If vertex i in community C has $k_{iC}^{ext} > 0$ and $k_{iC}^{int} \geq 0$, i is called *boundary vertex* since i has neighbor(s) outside of C. Otherwise, if $k_{iC}^{ext} = 0$ and $k_{iC}^{ext} > 0$, i is called *internal vertex* that only has connections with other vertices in the same community. In this paper, we employ the following functions to evaluate community structure:

4.1.1 Newman–Girvan modularity

The standard version of modularity (Newman & Girvan, 2004) reflects the difference for the fraction of intracommunity edges of a partition, with the expected number of such edges, if distributed according to a *null model*. In the standard version of modularity, the null model preserves the expected degree sequence of the graph under consideration. In other words, the modularity compares the real network structure with a corresponding one where nodes are connected without any preference about their neighbors. There are several ways to mathematically express modularity. In order to compare standard modularity with other variants, it is convenient to consider modularity as a sum of contributions from pairs of vertices of the same community:

$$Q_{NG}(P) = \frac{1}{m} \sum_{c \in P} \left[m_c - \frac{(2m_c + l_c)^2}{4m} \right] \tag{5}$$

Despite some existing issues such as the widely discussed resolution limits (Fortunato & Barthelemy, 2006) or the indistinguishability structures of two partitions having similar modularity scores, the Newman–Girvan modularity is still widely used in the community as a proof of quality.

4.1.2 Erdős–Rényi modularity

The Newman–Girvan modularity has attracted much attention in the research literature. Many alternative derivations have been proposed to adapt to different contexts. Some use different

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

null models to quantify the modular structure of partitions. For example, one could assume that vertices in a network are connected randomly with a constant probability, p, as formulated in the Erdős–Rényi (ER) model (Erdős & Rényi, 1959). The connection probability is calculated as $p = \frac{2m}{n(n-1)}$, being the number of presented edges over the total number of edges that could be established. The expected number of edges in a community of size n_c becomes $\langle m_c \rangle = p\binom{n_c}{2}$. This null model leads us to the ER modularity:

$$Q_{ER}(P) = \frac{1}{m} \sum_{c \in P} \left[m_c - \frac{m n_c (n_c - 1)}{n(n-1)} \right] \tag{6}$$

4.1.3 Modularity density (D-value or D-modularity)

Standard modularity is found to be impacted by resolution limits (Fortunato & Barthelemy, 2006), that is, it is claimed that the sizes of detected modules depend on the size of the whole network, such that optimizing standard modularity cannot identify communities having a small number of vertices. The expected number of intracommunity edges is highly sensitive to the total number of edges in the whole network (Rosvall & Bergstrom, 2007), as can be observed in the second term of Equation (5). Modularity density (Li *et al.*, 2008) is one of several propositions envisioned to palliate this issue. The idea of this metric is to include the information about community size in the expected density of a community to avoid the negligence of small and dense communities. For each community C in partition P, it uses the *average modularity degree* calculated by $d(C) = d^{int}(C) - d^{ext}(C)$, where $d^{int}(C)$ and $d^{ext}(C)$ are the average internal and external degrees of C, respectively, to evaluate the fitness of C in its network. Finally, the modularity density can be calculated as follows:

$$Q_D(P) = \sum_{c \in P} \frac{1}{n_c} \left(\sum_{i \in c} k_{ic}^{int} - \sum_{i \in c} k_{ic}^{ext} \right) \tag{7}$$

4.1.4 Z-modularity

Z-modularity is another variant of the standard modularity that is proposed to avoid the resolution limit (Miyauchi & Kawase, 2016). The founding idea of this version is based on the observation that the difference between the fraction of edges inside communities and the expected number of such edges in a null model should not be considered as the only contribution to the final quality of community structure. Specifically, the authors recommend that the statistical rareness of a community should also be taken into consideration. In other words, the rarer the structure of a community, the greater is its contribution to the final modularity. Therefore, the variance of the probability distribution of the fraction of the number of edges within each community is included in the quality function through a standardization using Z-score. Following the null model of the standard modularity, the probability that an edge is placed inside community C is $p = (\frac{D_C}{2m})^2$, where $D_C = 2m_C + l_C$ is the total degree of community C. The number of edges in each community follows a binomial distribution with the probability p, and its normalized value approaches a normal distribution when the number of edges is sufficiently large. The statistical rarity of partition P in terms of the fraction of the number of intracommunity edges using Z-score is thus translated into Z-modularity as follows:

$$Q_Z(P) = \left[\sum_{c \in P} \frac{m_c}{m} - \sum_{c \in P} \left(\frac{D_c}{2m} \right)^2 \right] \left[\sum_{c \in P} \left(\frac{D_c}{2m} \right)^2 \left(1 - \sum_{c \in P} \left(\frac{D_c}{2m} \right)^2 \right) \right]^{\frac{-1}{2}} \tag{8}$$

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

4.1.5 Surprise

This statistical approach proposes a quality metric assuming that edges between vertices emerge randomly according to a hyper-geometric distribution (Aldecoa & Marín, 2011). Specifically, for a graph of n vertices and m edges, there are $M = \binom{n}{2}$ possible ways of drawing m edges. For a particular partition, there are $M^{int} = \sum_{C \in P} \binom{n_c}{2}$ possible ways of drawing an intracommunity edge. Surprise metric computes the (minus logarithm of) probability of observing at least $m^{int} = \sum_{C \in P} \frac{k_C^{int}}{2}$ intracommunity edges within m draws without replacement from the population of M possible choices in which there are precisely M^{int} possible intracommunity edges. This probability is formalized as follows:

$$S(P) = -\log \sum_{k=m^{int}}^{\min(m,M^{int})} \frac{\binom{M^{int}}{k}\binom{M-M^{int}}{m-k}}{\binom{M}{m}}. \tag{9}$$

However, this formulation is not easy to work with in large-scale networks due to numerical computational problems. Hence, Traag *et al.* (2015) provide an asymptotic approximation for the metric, which is a good alternative. By assuming that the relative number of intracommunity edges $q = \frac{m^{int}}{m}$ and the relative number of expected intracommunity edges $\langle q \rangle = \frac{M^{int}}{M}$ remain fixed, Surprise metric is approximated at:

$$S(P) \approx mD(q||\langle q \rangle), \tag{10}$$

where $D(q||\langle q \rangle)$ is the Kullback–Leibler divergence (Kullback & Leibler, 1951):

$$D(q||\langle q \rangle) = p \log \frac{p}{\langle q \rangle} + (1-q) \log \frac{1-q}{1-\langle q \rangle}. \tag{11}$$

According to the Surprise metric, the higher the score of a partition, the less likely it is that it resulted from a random realization, and thus the better the quality of the community structure will be.

4.1.6 Significance

This metric uses a similar approach to the Surprise metric. It also estimates how likely a partition of dense communities is to appear in a random graph, but in a different way (Traag *et al.*, 2015). While Surprise uses global quantities q and $1 - \langle q \rangle$, Significance compares each community density $p_C = \frac{m_C}{\binom{n_C}{2}}$ to the average graph density $p = \frac{m}{M}$. The asymptotic form of Significance can be written as:

$$Z(P) = \sum_{C \in P} \binom{n_C}{2} D(p_C||p). \tag{12}$$

Similarly, $D(x||y)$ is the Kullback–Leibler divergence defined in Equation (11). Generally, if the number of communities is relatively large or the graph is relatively dense, Significance is more discriminative than Surprise. On the other hand, in the case where $\langle q \rangle > p$, Surprise can be better than Significance (Traag *et al.*, 2015).

4.2 Coperformance index

We propose a new comparative approach using a matrix called community detection *coperformance* matrix. The idea is that, given an expected quality function, one could investigate whether there exists a correlation in the efficiency of enhancing (or aggravating) its scores between different methods. The coperformance matrices reveal how understanding the performance of a

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

method in optimizing a quality would allow us to predict the performance of other methods on the same quality. Therefore, an exhaustive analysis of coperformance matrices on many qualities allows us to profile the characteristics of community detection methods in a comparative way. The index could be calculated as follows:

Let methods A and B divide a graph $G_i = (V_i, E_i)$ of dataset $\mathscr{G} = \{G_i | i = 1..N\}$ into α and β communities described by partitions $P^a_{G_i} = \{C^a_{1G_i}, C^a_{2G_i}, ..., C^a_{\alpha G_i}\} \in \mathscr{P}_{G_i}$ and $P^b_{G_i} = \{C^b_{1G_i}, C^b_{2G_i}, ..., C^b_{\beta G_i}\} \in \mathscr{P}_{G_i}$, respectively. We consider, solely, hard clustering methods, meaning $C^a_{uG_i} \cap C^a_{vG_i} = \varnothing : 1 \leq u < v \leq \alpha$ and $C^b_{uG_i} \cap C^b_{vG_i} = \varnothing : 1 \leq u < v \leq \beta$. A function $Q : \mathscr{P}_{G_i} \to \mathbb{R}$ quantifies a quality of a partition of graph G_i according to a particular goodness aspect (or model).

We define a *coperformance index* of two methods A and B on $\mathscr{G}$ by their mutual capacity in discovering community structures showing a particular quality Q. In other words, each couple of methods should be assigned a high index according to a quality Q, if knowing the performance of one method significantly reveals the information about the performance of the other. A straightforward solution for defining the index is using the Pearson correlation. In other words, it reflects the covariance of a quality score on two sets of partitions detected by two methods, indicating whether two methods *agree* or *disagree* about a quality criterion in the context of a network dataset.

Denoting $q^a_i = Q(P^a_{G_i})$ and $q^b_i = Q(P^b_{G_i})$, the coperformance index can be calculated as follows:

$$I_{\mathscr{G}}(A, B, Q) = \frac{N \sum q^a_i q^b_i - \sum q^a_i \sum q^b_i}{[N \sum (q^a_i)^2 - (\sum q^a_i)^2]^{1/2} [N \sum (q^b_i)^2 - (\sum q^b_i)^2]^{1/2}}, \tag{13}$$

where $0 \leq I_{\mathscr{G}}(A, B, Q) \leq 1$. With a Pearson correlation, a high positive (respectively, negative) score implies that two methods often find a strong consensus (respectively, disagreement) in discovering communities with regards to a particular quality. Given a coperformance index, knowing the quality scores of one method could provide predictive information about the outcomes of the other method on the same dataset. This information could, in fact, be very useful in a context where alternative solutions must be deployed while maintaining an assumed quality, as expected. We present in the following section the mutual performance of the presented detection methods by the previously presented quality functions.

Figure 17 illustrates the coperformance matrices, according to six different quality goodness criteria. Again, similar to the previous section, goodness functions with a *close* concept are placed together. For instance, *Newman-Girvan* modularity and *Erdős-Rényi* modularity are both based on null models whose concept uses an expected fraction of intracommunity edges. While the hypothesis of *NG* version is to keep the expected degree sequence of the graph in question, the *ER* version redistributes edges randomly with a constant average degree for every node. *D*-modularity and *Z*-modularity attempt to penalize large communities by including community sizes and significance level, respectively. One can notice a very slight similarity in the experimental results of the coperformance indexes between different quality functions. Also, it seems that the assumption about the quality of community structure has an impact on the coperformance outcome.

As shown in Figure 17, there is a class of methods (*Louvain, GN, CNM, RB, Infomod, Infomap, Walktrap, OSLOM, LPA, SLPA, Conclude*) in which all methods show very consistent results, except for the case of *D*-modularity[3]. Furthermore, there is also a strong relation between *SBM* and *DCSBM*. For the other methods (*RCCLP* and *SN*), no clear tendency could be observed from this experiment. The similarity of a large number of methods in many quality functions implies that, globally, if a method performs well on a given network, there is a signal that the others (from the same group) could also obtain good results. In other words, if the community structure in a network is clear, most methods will be able to detect it with more or less accuracy and conversely. However, as the coperformance indexes also vary significantly (0.2–0.3) inside each group, there will always be a remarkable difference if one goes from one method to another.

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

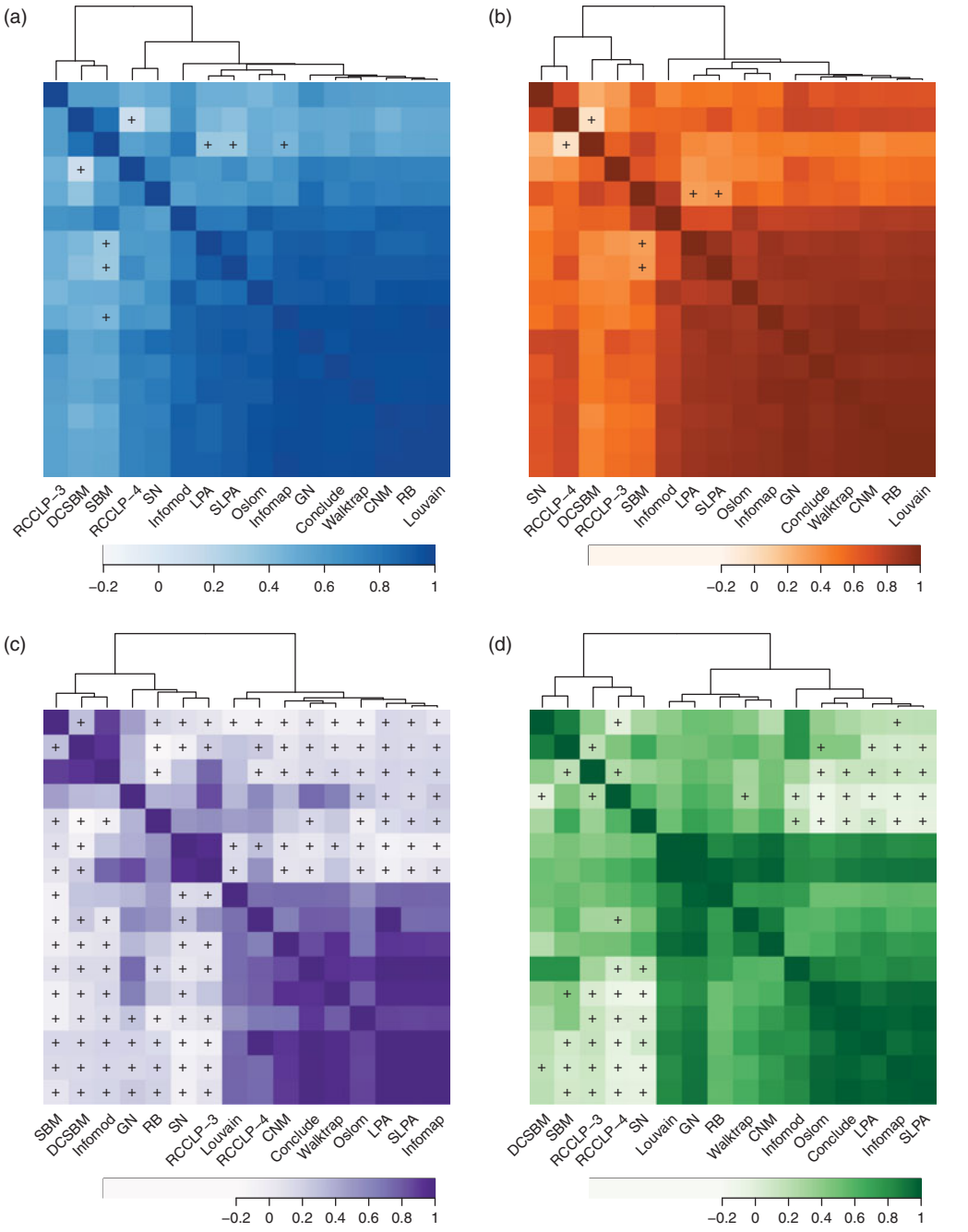


Figure 17. The *coperformance* matrices of different methods. The "+" marks indicate cases where p-values are larger than 0.05. (a) Newman-Girvan modularity, (b) Erdős-Rényi modularity, (c) Density modularity and (d) Z modularity.

Within the case of density modularity shown in Figure 17(c), we discover that the sizes of detected communities have a great impact on coperformance. Since density is a measure that penalizes large-size communities heavily, especially in sparse networks, *D*-modularity gives very small values for giant communities and very high values for small ones. Concretely, the methods *SBM*, *DCSBM*, *Infomod*, *RB*, *SN* discover very large communities (as shown in Section 3.3)

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

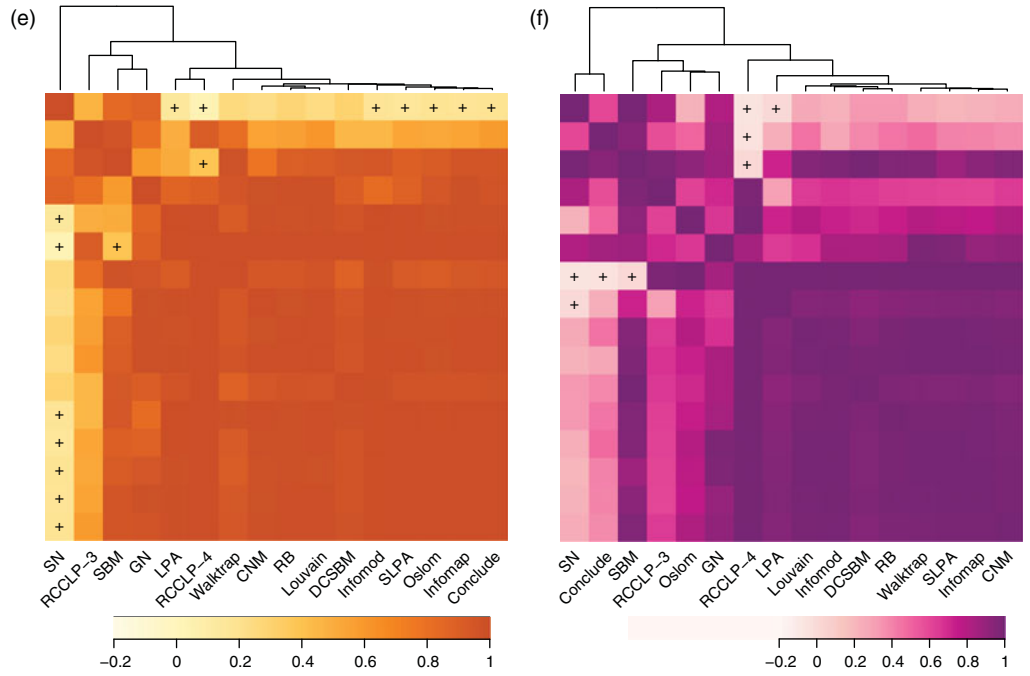


Figure 17. The *coperformance* matrices of different methods (cont.) (E) Surprise and (F) Significance.

and their coperformances in terms of *D*-modularity are very weak, showing that internal densities of communities detected by these methods are not linearly correlated. The reason is that the corresponding densities fluctuate unpredictably around zero. Similarly, *GN* and *RCCLP-3* found many tiny communities making the density either very high or zero (if the internal degree is equal to the external degree), consequently the coperformance index cannot show significant information. On the other hand, we notice a consistency between the similarity of community size and the coperformance when methods identify medium-sized communities. Specifically, we find high coperformance indexes between *CNM, Conclude, OSLOM, Walktrap, LPA, SLPA, Infomap* methods in most of the cases of the six quality fitness functions. This finding exposes a global agreement with our categorization, determined by community size distributions.

The coperformance matrices also disclose interesting information about quality functions. As we can see in Figure 17(a,b,d), the matrices imply a similarity between *NG* modularity, *ER* modularity, and *Z*-modularity in the assumptions of quality. In the same way, Surprise and Significance are quite close in practice, as illustrated in Figure 17(d) and (e). This experiment gives more proof about the closeness between the theoretical assumptions of community structure and the practical outcome. Moreover, although being based on different aspects of goodness, the performance of many methods tends to reach agreement on the modular structure of networks in general. This is to say, methods in the same group identify roughly and globally comparable results, although there are always significant differences. In order to strengthen and validate our conclusion, we are interested in using other popular approaches in the literature to compare these community detection algorithms, which will be presented in the next section.

5. Partitioning strategy comparison

This section is dedicated to using conventional clustering validation metrics from the literature to verify the previous similarity analyses. We employ some popular metrics in the traditional clustering context (and also widely used in community detection context), which measure directly the

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Table 6. Contingency table of P_1 and P_2 on the same graph provides information about the similarity between the two partitions.

Partition P_1 \ Partition P_2	$c_1^{(2)}$	$c_2^{(2)}$	$\cdots$	$c_S^{(2)}$	$\sum$
$c_1^{(1)}$	n_{11}	n_{12}	$\cdots$	n_{1S}	$n_{1\cdot}$
$c_2^{(1)}$	n_{21}	n_{22}	$\cdots$	n_{2S}	$n_{2\cdot}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$c_R^{(1)}$	n_{R1}	n_{R2}	$\cdots$	n_{RS}	$n_{R\cdot}$
$\sum$	$n_{\cdot 1}$	$n_{\cdot 2}$	$\cdots$	$n_{\cdot S}$	n

similarity of partitions using their corresponding contingency tables. These metrics do not take into consideration the structural information of community structures but only use the common numbers of nodes that are shared by pairs of communities in two partitions.

5.1 Validation metrics

The consensus of two partitions $P_1 = \{c_1^{(1)}, c_2^{(1)}, ..., c_R^{(1)}\}$ and $P_2 = \{c_1^{(2)}, c_2^{(2)}, ..., c_S^{(2)}\}$ can be more practically observed using a contingency table (sometimes called confusion matrix or association matrix), whose elements $n_{ij} = |c_i^{(1)} \cap c_j^{(2)}|$ correspond to the number of common vertices between the ith community of P_1 and the jth community of P_2 as shown in Table 6.

In the evaluation of community structure using a validation metric, some of the following validation metrics are often used in the context of community detection, to define the matching coefficient between two arbitrary partitions of a network.

5.1.1 Rand index

The Rand index (RI) is a pair-counting-based measure, defined as the ratio of the number of vertex pairs correctly classified (either in the same community or in different communities) by the total number of pairs (Rand, 1971). The RI penalizes both false positive and false negative decisions of the clusterings. When the false positive has to be neglected, we can refer to the *Jaccard index* (Kuncheva & Hadjitodorov, 2004). The RI value of two partitions can be calculated by the following:

$$RI(P_1, P_2) = \frac{\binom{n}{2} + 2\sum_i \sum_j \binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i\cdot}}{2} + \sum_j \binom{n_{\cdot j}}{2}\right]}{\binom{n}{2}} \tag{14}$$

The value varies between 1 (meaning the two partitions are identical) and 0 (indicating that the two partitions do not agree on any pair of vertices). However, this value is only observed in the scenario when one partition consists of one community and the other consists of n community of 1 vertex, which has little practical value. Another shortcoming of the RI is that its expected value for two randomly chosen partitions does not take a constant value, which is normally expected for a good matching index (Vinh *et al.*, 2010). Therefore, a modified version of RI has been suggested, taking into consideration the expected value of randomness (Hubert & Arabie, 1985), which is introduced in the following.

5.1.2 Adjusted Rand index

The corrected version of RI takes the form:

$$Adjust_index = \frac{Index - Expected_Index}{Max_Index - Expected_Index} \tag{15}$$

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

It quickly becomes a replacement recommended for measuring agreement between two partitions in the analysis of clusterings. Its values range from −1 to 1 indicating completely different and identical partitions, respectively. It is known to be less sensitive to the difference of the number of communities between two partitions. An adjusted Rand index (ARI) value of 0 indicates that the similarity is equal to the expected value from randomly chosen partitions. It can be calculated as:

$$ARI(P_1, P_2) = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left[\sum_i \binom{n_i}{2} \sum_j \binom{n_j}{2}\right] \Big/ \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_i}{2} + \sum_j \binom{n_j}{2}\right] - \left[\sum_i \binom{n_i}{2} \sum_j \binom{n_j}{2}\right] \Big/ \binom{n}{2}} \tag{16}$$

5.1.3 Normalized mutual information

Information theoretic-based metrics constitute another approach for validating community structure with a given reference partition. Using the same notations as previously presented, the mutual information (MI) between two partitions quantifying the mutual dependence is calculated as:

$$I(P_1, P_2) = \sum_{ij} p\left(c_i^{(1)}, c_j^{(2)}\right) \log \frac{p\left(c_i^{(1)}, c_j^{(2)}\right)}{p\left(c_i^{(1)}\right) p\left(c_j^{(2)}\right)} = \sum_{ij} \frac{n_{ij}}{n} \log \frac{n_{ij} n}{n_{i.} n_{.j}} \tag{17}$$

It measures how much knowing a partitioning of vertices in one way would reduce the uncertainty about the other way. In order words, it could be considered as an indicator of information *closeness* expressed by the joint distribution between two variables. Therefore, the MI can be used as a similarity metric between two partitions. However, it needs to be normalized to reflect a consistency between different measures. The normalization uses the entropy of each partition defined by:

$$H(P) = -\sum_k \frac{n_k}{n} \log \frac{n_k}{n} \tag{18}$$

Several variants of normalization can be considered, for instance, taking the average, the root, or the maximum of entropy of the two partitions as the denominator (Ana & Jain, 2003). In this document, we use the average version which is widely used in the context of community analysis (Danon *et al.*, 2005; Chakraborty *et al.*, 2017). The closed form of normalized mutual information (NMI) is hence defined from Equations (17) and (18), as follows:

$$NMI(P_1, P_2) = \frac{2I(P_1, P_2)}{H(P_1) + H(P_2)} = \frac{-2 \sum_{ij} n_{ij} \log\left(\frac{n_{ij} n}{n_i n_j}\right)}{\sum_i n_i \log\left(\frac{n_i}{n}\right) + \sum_j n_j \log\left(\frac{n_j}{n}\right)} \tag{19}$$

Likewise, the NMI similarity between two partitions varies between 0, corresponding to independent relation, and 1 when two partitions are identical. NMI does not follow triangle inequality.

5.1.4 Adjust mutual information

Similar to the RI, MI is also subject to the effect of randomness, that is, there is not a constant baseline value between random partitions of a graph. This issue raises many difficulties in the comparison mechanism, since it is expected that a comparative index should preserve the relativity between different clusterings and enhance intuitiveness about the mutual agreement. For that reason, traditional MI is normalized with a supplementary correction for chance. This normalization method has recently received attention for comparing graph partitions. It is calculated as follows (Vinh *et al.*, 2010):

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Table 7. Some popular validation metrics for comparing community partitions.

Label	Range	Measure
RI	[0, 1]	Fraction of commonly grouped and separated vertices in two partitions.
ARI	[0, 1]	Rand index with a chance correction, less sensitive to differences of community sizes.
NMI	[0, 1]	Information theoretic approach, indicates how much information in one partition will help to guess the other.
AMI	[0, 1]	Similar to NMI, with a chance correction to set a constant baseline for random partitions.
VI	[0, log (*n*)]	Shared information distance measures the amount of MI. The higher the value, the less resemblance between the two partitions.
NVI	[0, 1]	Normalized version of shared information distance with chance correction.

$$AMI(P_1, P_2) = \frac{I(P_1, P_2) - E\{I(M)|n_i.n_{.j}\}}{\frac{1}{2}(H(P_1) + H(P_2)) - E\{I(M)|n_i.n_{.j}\}}, \tag{20}$$

where $I(P_1, P_2)$ and $H(P)$ are introduced in equations (17) and (18), respectively. $E\{I(M)|n_i.n_{.j}\}$ is the expected MI value of all feasible contingency tables constructed from the actual table M with the same marginals $n_{i.}$, $n_{.j}$.

5.1.5 Normalized variation of information

Another popular metric that is often used in the context of comparing community partition similarity is the variation of information (VI) (Meilă, 2003), which is defined as:

$$VI(P_1, P_2) = H(P_1) + H(P_2) - 2I(P_1, P_2) \tag{21}$$

The VI metric can be interpreted as an index of shared information distance between two partitions. Its lower bound is 0, and it occurs when the two partitions are identical, whereas the upper bound $\log(n)$ happens when they are completely different. It is also preferable to use a normalized version, chance corrected, to avoid the effect of randomness. Similar to the construction of the adjusted mutual information (AMI), with the same notation, the normalized variation of information (NVI) is calculated as follows:

$$NVI(P_1, P_2) = \frac{H(P_1) + H(P_2) - 2I(P_1, P_2)}{H(P_1) + H(P_2) - 2E\{I(M)|n_i.n_{.j}\}}. \tag{22}$$

However, it turns out that *NVI* discloses the same information with *AMI*, since from Equations (20) and (22), one has $NVI(P_1, P_2) = 1 - AMI(P_1, P_2)$. Consequently, calculating *VI* and *NVI* is unnecessary. We have chosen the *RI*, *ARI*, *NMI*, and *AMI* metrics in our experiment. A summary of these validation metrics is shown in Table 7.

Validation metrics are often used in the context of community structure evaluation to measure the difference between the partition identified by a method with an expected partition of the network under consideration (*ground truth*). The more similar the discovered partition is to the ground truth, the better the performance of the method. However, in this section, validation metrics are exploited as a tool to compare community structures of different detection methods. They estimate the practical proximity of different algorithms through detected partitions, which constitutes a supplementary source of information for evaluating their performance in a comparative approach.

5.2 Empirical results

Once again, the experimental process is the same as those of the previous sections. From the partitions detected by the methods on the dataset, we calculate pairwise scores, quantified by each

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

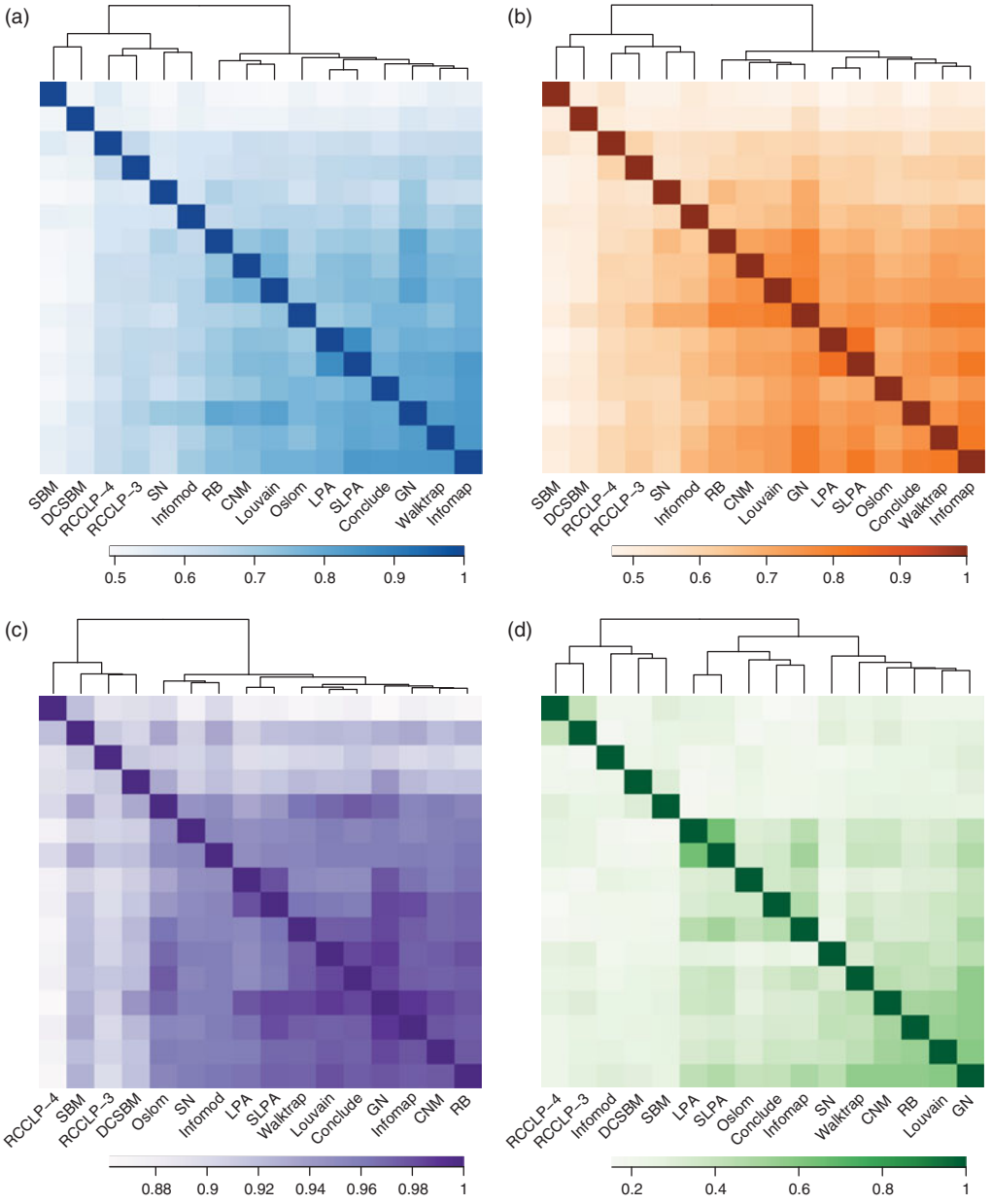


Figure 18. The similarity between community detection methods quantified by different validation metrics based on partitions discovered on networks of the dataset. Rows and columns are ordered according a hierarchical clustering method (Joe H. Ward, 1963). In order, the average score of (a). NMI, (b). AMI, (c). RI, (d). ARI.

validation metric on each network. Figure 18 illustrates pairwise average scores of the four metrics over the networks of the dataset[4].

Again, by observing the dendrograms in Figure 18, one can see that all four metrics classify methods into two principle groups in a similar way to the coperformance matrices exposed in the previous section. The group of methods *CNM, Conclude, OSLOM, Walktrap, LPA, SLPA, Infomap* mentioned in the last section also shows very strong similarities in this experiment. *LPA* and *SLPA*, especially, being based on label propagation mechanism show nearly identical results in many

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

cases. In addition, one could discern another group including *RB, CNM, GN*, and *Louvain* (modularity based), which shows high consistency in general. Additionally, even with weaker scores, *SBM* and *DCSBM* are often found in the same group, with *RCCLP-3* and *RCCLP-4* too. Globally, it seems that methods with a close theoretical approach tend to provide more similar results, which is also revealed in the previous sections.

Another fact that could be extracted from this experiment is that *RI* should not be used as a validation metric for evaluating detection performance. Since its average values vary generally over a small range (0.9–1.0), it is more difficult to see the difference between partitions. On the other hand, *NMI* and *AMI* show very close results in our experiment (between 0.5 and 1.0), meaning that structural communities detected by different methods are quite comparable, as concluded in the previous section. Finally, *ARI* seems to magnify the differences between methods. However, there is no major difference in the similarity evaluation in comparison with the other metrics.

6. Related work

Orman *et al.* (2012) published a comparative evaluation of eight community detection algorithms, most of which are also studied in this paper. Different validation metrics are used to evaluate the agreement between the discovered partitions and the reference community structures (ground truths). As in our work, they find that these metrics (*RI*, *ARI*, *NMI*) *"agree with each other with small differences when considering the way they rank algorithms,"* as illustrated in Section 5. Furthermore, the authors also focus on analyzing many topological aspects of community structure including transitivity, density, community size, etc. These topological qualities are then used to inspect community structures detected by different algorithms. The analyses allow the authors to conclude that these two approaches (topological metrics and validation metrics), used to evaluate community structures, are *"complementary and needed to perform a relevant and complete analysis of community detection results."* They also note that the *"traditional approach (RI, ARI, NMI) is much faster and easier to apply,"* and hence, they advise using these metrics first. However, in practice, ground truths are not usually available[5]. Therefore, from the above observations, our analyses in this paper could be an important support, dispensing additional information about the closeness between methods, both in terms of the topological aspect and the partition-based aspect.

Agreste *et al.* (2017) evaluate different community detection algorithms in an empirical and comparative approach, especially for the context of web data analytic. The authors find that *"time complexity is a crucial factor in the selection of a community detection algorithm"* and find that the label propagation method (LPA) *"has outstanding performance in scalability on both artificial and real graphs,"* which is also in global agreement with our analysis in Section 3.1 providing predictions about required time for each method, according to network size. They also conclude that *"Infomap algorithm showcased the best trade-off between accuracy and computational performance"* based on *NMI* score. Such a conclusion is valid in some specific cases where the expected ground-truth community structure is well understood. Otherwise, some additional analyses should be done to determine whether *ground-truth* information aligns the final objective of community detection algorithms (Peel *et al.*, 2017). Indeed, metadata information of nodes are often used in practice as a ground-truth community structure, whereas it has been found that metadata communities are sometimes very sparse (Dao *et al.*, 2017). One should be cautious as regards generalizing. Algorithms may have difficulty agreeing with some nonaligned metadata-based reference, but one should not assume this level of performance on other datasets.

Ghasemian *et al.* (2018) present, in a recent publication, an evaluation of overfitting and underfitting of several community detection models. The authors study the number of communities detected in practice by many methods, and the maximum number of detectable clusters, according to a theoretical model. Some conclusions are drawn about fitting qualities of methods in

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

comparison to theoretical estimates. This study helps to choose an appropriate method according to a fitting quality. Community detection methods are also grouped in distinct families, based on their outputs on many real-world networks (similar to our analysis in Section 5.1) using *AMI* metric. The authors also find that *"what an algorithm finds in a network depends strongly on the assumptions it makes about what to look for,"* which is aligned with our results through several analyses.

Jebabli *et al.* (2018) also propose a framework to assess the performance of community detection algorithms, based on the topological characteristics of the resulting community graphs. In their paper, the authors restrict their attention to networks with overlapping community structures, whereas our comparative evaluation concerns hard clustering. However, they discuss how their framework can be applied to networks with nonoverlapping community structure. They present and evaluate an efficient alternative methodology, as compared to the classical quality and clustering measures. Similar to our work, algorithms are compared in a decision-making scheme. They provide different rankings of the algorithms, according to different topological properties. They also introduce a multiple-criteria decision-making strategy in order to find the best compromise between these different rankings. A single ranking is produced. Such a decision-making strategy is therefore a very interesting approach. Converting recommendations into a decision process is one of our perspective studies.

7. Conclusion

It is quite challenging to recommend which method is best for which scenario. It is at least as demanding as defining all possible scenarios. We carried out several experiments, demonstrating different aspects of community structure quality, which can be combined together in a flexible way to assist network analysts to find appropriate methods, according to the context. The following questions, for example, could be asked sequentially during decision-making processes:

1. What is the size of the network under consideration and what is the acceptable computation time for the task of community detection?
2. What are the expectations about the number of communities as well as the community size distribution?
3. Is there any fitness function that should be optimized?
4. In circumstances where the targeted method cannot be deployed, is there an alternative solution?

The experiments and results in this paper could help to identify suitable method(s) quickly, if one is able to answer the above questions.

The consideration of computation time is indeed crucial in the process of choosing a community detection method for a problem at hand. Even if the theoretical estimate of time complexity is important and reveals the scalability of a community detection method, computation time is worth being studied in practice. Our estimates provide detailed information of practical time required by many popular community detection methods, according to network size. In particular, the most scalable methods that we tested (*Louvain*, *LPA*, and *SN*) approximately reduce 10^4 times the required computation time, compared with most of the other methods. This is not only significant but also crucial for a large network. Given a network size, our results help in filtering nonsuitable methods.

In addition, the expected number of communities to be obtained is another important criterion in choosing a community detection method. Depending on the context, one could prefer different granularity levels. Our study shows that there are globally three main strategies that community detection methods use to decompose a network. Specifically, some identify communities whose

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

sizes vary regularly over a wide range of values, from very small to very large communities. Some others divide networks into a huge number of very small communities and very few large communities. And finally, the latter distribute nodes into similar medium-size communities (around 10 members). Therefore, knowing how a network should be broken down is very useful in order to end up with an appropriate community discovery method.

In cases where (advanced) network analysts can determine a targeted objective function, designing new algorithms (or employing existing algorithms) that optimize the function would be the most effective. Since improving an objective function usually means expending more computation time, a compromise between getting higher fitness scores and using less time needs to be considered. However, finding a good method to optimize an objective function, satisfying a time constraint condition, is not straightforward and needs much investigation. Our approach, as presented in the coperformance analysis, provides network practitioners with a quick view of how different methods perform in improving some widely used quality functions. This predictive information about the effect of using alternative methods in achieving good fitness scores would suggest multiple solutions for network analysts to reach a certain objective function. This scenario is specifically useful when the desired method is too expensive, in terms of computation time. Therefore, a combination of our empirical analyses about scalability and/or community size distribution with the coperformance index can identify eligible alternatives for specific cases.

Finally, we find that using some validation metrics to estimate the similarity between community detection methods could also provide interesting information that could help the decision process of network analysts. In situations where one knows exactly (or has notions of) what should be found (ground-truth information), studying the way nodes are allocated to communities is important, as it provides useful information about how a method is able to reach the desired clusters. However, this scenario does not often occur, since community detection is in general used to discover the structures of networks when no *a priori* information is available. When performing several methods is possible, validation metrics are used to compare their results and then identify different types of partitions. From our empirical study, we noticed significant differences in the way that nodes are distributed in communities. Methods such as *SBM* or *RCCLP-4*, especially, seem to detect partitions that are very discernible from those of the others. Hence, the use of these methods needs to be examined, and we shall recommend their use, along with other methods, as they may bring totally different, and probably complementary insights to the data.

Conflict of interest. Vinh Loc DAO, Cécile BOTHOREL, and Philippe LENCA have nothing to disclose.

Notes

1 https://www.ning.com/

2 https://github.com/AGhasemian/CommunityFitNet

3 In fact, density modularity is somewhat apart from other traditional ways of defining modularity, as it is not defined based on a null model but solely on edge density. The term *D*-modularity is misused in this sense.

4 Where the corresponding methods are able to finish using a reasonable amount of time and memory as mentioned in the previous experiments.

5 In the context where a new algorithm is invented, one normally uses networks whose community structures are well known in order to validate the proposed method. In reality, since community detection is often employed to discover structures of new networks, it is therefore not likely that reference community structures always exist.

References

Agreste, S., Meo, P. D., Fiumara, G., Piccione, G., Piccolo, S., Rosaci, D., Sarne, G. M. L., & Vasilakos, A. V. (2017). An empirical comparison of algorithms to find communities in directed graphs and their application in web data analytics. *IEEE Transactions on Big Data*, *3*(3), 289–306.

Aldecoa, R., & Marín, I. (2011). Deciphering network community structure by surprise. *PLoS ONE*, *6*(9), e24195.

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Ames, B. P. W. (2013). Guaranteed clustering and biclustering via semidefinite programming. *Mathematical Programming*, *147*(1–2), 429–465.
Ana, L. N. F., & Jain, A. K. (2003). Robust data clustering. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Proceedings.*
Arifin, S., Zulkardi, Putri, R. I. I., Hartono, Y., & Susanti, E. (2017). Developing Ill-defined problem-solving for the context of South Sumatera. *Journal of Physics: Conference Series*, *943*(dec), 012038.
Blondel, V. D, Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*(10), P10008.
Bohlin, L., Edler, D., Lancichinetti, A., & Rosvall, M. (2014). Community detection and visualization of networks with the map equation framework. In *Measuring scholarly impact* (pp. 3–34). Springer International Publishing.
Chakraborty, T., Dalmia, A., Mukherjee, A., & Ganguly, N. (2017). Metrics for community analysis: A survey. *ACM Computing Surveys*, *50*(4), 1–37.
Clauset, A., Newman, M. E. J., & Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, *70*(6).
Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, *74*(368), 829–836.
Coscia, M., Giannotti, F., & Pedreschi, D. (2011). A classification for community discovery methods in complex networks. *Statistical Analysis and Data Mining*, *4*(5), 512–546.
Danon, L., Díaz-Guilera, A., Duch, J., & Arenas, A. (2005). Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, *2005*(09), P09008–P09008.
Dao, V. L., Bothorel, C., & Lenca, P. (2017). Community structures evaluation in complex networks: A descriptive approach. In *The 3rd international winter school and conference on network science, NetSciX*, 11–19.
Dao, V. L., Bothorel, C., & Lenca, P. (2018a). An empirical characterization of community structures in complex networks using a bivariate map of quality metrics. *Arxiv e-prints*, June.
Dao, V. L., Bothorel, C., & Lenca, P. (2018b). Estimating the similarity of community detection methods based on cluster size distribution. In *The 7th international conference on complex networks and their applications*. Springer International Publishing.
Erdős, P., & Rényi, A. (1959). On random graphs, I. *Publicationes mathematicae (debrecen)*, *6*, 290–297.
Fortunato, S., & Barthelemy, M. (2006). Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, *104*(1), 36–41.
Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, *486*(3–5), 75–174.
Fortunato, S., & Hric, D. (2016). Community detection in networks: A user guide. *Physics Reports*, *659*(Nov.), 1–44.
Ghasemian, A., Hosseinmardi, H., & Clauset, A. (2018). Evaluating overfit and underfit in models of network community structure. *Arxiv e-prints*, Feb.
Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, *99*(12), 7821–7826.
Holland, P. W, Laskey, K. B., & Leinhardt, S. (1983). Stochastic blockmodels: First steps. *Social Networks*, *5*(2), 109–137.
Hric, D., Darst, R. K., & Fortunato, S. (2014). Community detection in networks: Structural communities versus ground truth. *Physical Review E*, **90**(6).
Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, *2*(1), 193–218.
Jebabli, M., Cherifi, H., Cherifi, C., & Hamouda, A. (2015). Overlapping community detection versus ground-truth in amazon co-purchasing network. In *2015 11th international conference on signal-image technology & internet-based systems (SITIS)* (pp. 328–336). IEEE.
Jebabli, M., Cherifi, H., Cherifi, C., & Hamouda, A. (2018). Community detection algorithm evaluation with ground-truth data. *Physica A: Statistical Mechanics and Its Applications*, *492*, 651–706.
Jerome, K. (2013). The koblenz network collection. In *Proceedings conference on world wide web companion* (pp. 1343–1350).
Ward, J. H., Jr. (1963). Hierarchical grouping to optimize an objective function. *Journal of the american statistical association*, *58*(301), 236–244.
Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, *22*(1), 79–86.
Kuncheva, L. I., & Hadjitodorov, S. T. (2004). Using diversity in cluster ensembles. In *IEEE international conference on systems, man and cybernetics*.
Lambiotte, R. (2010). Multi-scale modularity in complex networks. In *8th international symposium on modeling and optimization in mobile, ad hoc, and wireless networks*, May (pp. 546–553).
Lancichinetti, A., Kivelä, M., Saramäki, J., & Fortunato, S. (2010). Characterizing the community structure of complex networks. *PLoS ONE*, *5*(8), e11976.
Lancichinetti, A., Radicchi, F., Ramasco, J. J., & Fortunato, S. (2011). Finding statistically significant communities in networks. *PLoS ONE*, *6*(4), e18961.
Leskovec, J., & Krevl, A. (2014). *SNAP Datasets: Stanford large network dataset collection*. June.
Leskovec, J., Lang, K. J., Dasgupta, A., & Mahoney, M. W. (2008). Statistical properties of community structure in large social and information networks. *Proceeding of the 17th international conference on world wide web – WWW 08*.

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press

Li, Z., Zhang, S., Wang, R.-S., Zhang, X.-S., & Chen, L. (2008). Quantitative function for community detection. *Physical Review E*, *77*(3).
Meilă, M. (2003). Comparing clusterings by the variation of information. *Learning Theory and Kernel Machines*, 173–187.
Meo, P. D., Ferrara, E., Fiumara, G., & Provetti, A. (2014). Mixing local and global information for community detection in large networks. *Journal of Computer and System Sciences*, *80*(1), 72–87.
Miyauchi, A., & Kawase, Y. (2016). Z-score-based modularity for community detection in networks. *PLOS ONE*, *11*(1), e0147805.
Newman, M. E. J. (2006a). Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, *74*(Sep), 036104.
Newman, M. E. J. (2006b). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, *103*(23), 8577–8582.
Newman, M. E. J. (2010). *Networks: An Introduction*. Oxford University Press.
Newman, M. E. J., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, *69*(2).
Orman, G. K., Labatut, V., & Cherifi, H. (2012). Comparative evaluation of community detection algorithms: A topological approach. *Journal of Statistical Mechanics: Theory and Experiment*, *2012*(08), P08001.
Papadopoulos, S., Kompatsiaris, Y., Vakali, A., & Spyridonos, P. (2011). Community detection in social media. *Data Mining and Knowledge Discovery*, *24*(3), 515–554.
Peel, L., Larremore, D. B., & Clauset, A. (2017). The ground truth about metadata and community detection in networks. *Science Advances*, *3*(5), e1602548.
Pons, P., & Latapy, M. (2005). Computing communities in large networks using random walks. In P. Yolum, T. Güngör, F. Gürgen, & C. Özturan (Eds.) *Computer and information sciences – ISCIS 2005* (pp. 284–293). Berlin, Heidelberg: Springer.
Pons, P., & Latapy, M. (2011). Post-processing hierarchical community structures: Quality improvements and multi-scale view. *Theoretical Computer Science*, *412*(8–10), 892–900.
Porter, M. A., Onnela, J.-P., & Mucha, P. J. (2009). Communities in Networks. *Notices of the American Mathematical Society*, Feb.
Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., & Parisi, D. (2004). Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences*, *101*(9), 2658–2663.
Raghavan, U. N., Albert, R., & Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, **76**(3).
Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, *66*(336), 846–850.
Reichardt, J., & Bornholdt, S. (2006). Statistical mechanics of community detection. *Physical Review E*, **74**(1).
Riolo, M. A., Cantwell, G. T., Reinert, G., & Newman, M. E. J. (2017). Efficient method for estimating the number of communities in a network. *Physical Review E*, **96**(3).
Rossi, R. A., & Ahmed, N. K. (2015). The network data repository with interactive graph analytics and visualization. In *Proceedings of the twenty-ninth aaai conference on artificial intelligence*.
Rosvall, M., & Bergstrom, C. T. (2007). An information-theoretic framework for resolving community structure in complex networks. *Proceedings of the National Academy of Sciences*, *104*(18), 7327–7331.
Rosvall, M., Axelsson, D., & Bergstrom, C. T. (2009). The map equation. *European Physical Journal Special Topics*, *178*(Nov.), 13–23.
Schaub, M. T., Delvenne, J.-C., Rosvall, M., & Lambiotte, R. (2017). The many facets of community detection in complex networks. *Applied Network Science*, **2**(1).
Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. London, New York: Chapman and Hall.
Traag, V. A., Dooren, P. Van, & Nesterov, Y. (2011). Narrow scope for resolution-limit-free community detection. *Physical Review E*, **84**(1).
Traag, V. A., Aldecoa, R., & Delvenne, J.-C. (2015). Detecting communities using asymptotical surprise. *Physical Review E*, **92**(2).
Vinh, N. X., Epps, J., & Bailey, J. (2010). Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *J. Mach. Learn. Res.*, *11*(Dec.), 2837–2854.
Wasserman, S. (1994). *Social network analysis: Methods and applications*. Cambridge, New York: Cambridge University Press.
Xie, J., & Szymanski, B. K. (2012). Towards linear time overlapping community detection in social networks. In *Advances in knowledge discovery and data mining* (pp. 25–36). Berlin, Heidelberg: Springer.
Yang, J., & Leskovec, J. (2013). Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, *42*(1), 181–213.

Cite this article: Dao V. L., Bothorel C., and Lenca P. (2020). Community structure: A comparative evaluation of community detection methods. *Network Science* **8**, 1–41. https://doi.org/10.1017/nws.2019.59

https://doi.org/10.1017/nws.2019.59 Published online by Cambridge University Press