Sensitivity of Community Structure to Network Uncertainty

Marc Mitri* Fragkiskos D. Malliaros† Michalis Vazirgiannis‡

Abstract

Community detection constitutes an important task for investigating the internal structure of networks, with a plethora of applications in a wide range of disciplines. A particularly important point, which is rarely taken into account while developing community detection algorithms, is their sensitivity (or stability) to network uncertainty. In many cases, the input graph data is incomplete or noisy (e.g., due to noise introduced during the collection of the data or for privacy preserving reasons). Then, the following question arises: how stable are the results produced by an algorithm with respect to the uncertainty (i.e., noise level) of the input data? In this paper, we propose a quantitative way to address this problem. We have considered several graph perturbation models to introduce uncertainty to the graph. Then, we examine the sensitivity of an algorithm, with respect to functional and structural characteristics of the detected communities under various perturbation levels. We have studied the performance of some of the most widely used community detection algorithms in practice, and our experimental results indicate that random walk based community detection algorithms tend to be robust under various conditions of network uncertainty.

Keywords: Community detection; Graph clustering; Robustness; Graph perturbation; Network uncertainty

1 Introduction

Over the last years, the study and analysis of real-world networks have received a considerable interest from the research community. A fundamental concept that is related to the structural properties of real networks is the existence of community structure; the nodes are organized into densely connected groups with sparse connections across different groups, that are commonly referred to as network communities, clusters or modules [9]. In the related literature, a plethora of graph clustering and community detection algorithms have been designed, where the main goal is to identify clusters based on the information encoded in the graph topology (e.g., see Ref. [9]).

*Computer Science Laboratory, École Polytechnique, Palaiseau, France. Email: marc.mitri@polytechnique.edu

†Department of Computer Science and Engineering, UC San Diego, La Jolla, USA. Email: fmalliaros@ucsd.edu

‡Computer Science Laboratory, École Polytechnique, Palaiseau, France. Email: mvazirg@lix.polytechnique.fr

However, one fundamental issue of real-world networks is that they are inherently noisy and incomplete [1]. For instance, protein-protein interaction networks are inferred experimentally, and therefore might contain measurement errors; Twitter's who-follows-whom networks are built using small samples available through public APIs. In addition to that, released graph data may contain a degree of uncertainty introduced by data owners, in order to protect data privacy. Those points constitute a challenge for practitioners who want to get reliable results while applying community detection algorithms for their applications, raising the following question: how robust are the results of a community detection algorithm under network uncertainty?

In this work, we tackle the problem of estimating the sensitivity of community detection algorithms to network uncertainty. To do so, we study the behavior of widely applied community detection algorithms under several perturbation strategies that model different settings of uncertainty (i.e., noise). The main contributions of this work address the following questions:

- *Functional analysis*: How do the community assignments produced by a particular clustering algorithm vary with increasing noise levels? Considering the communities detected on the original (i.e., unperturbed) graph as the ground truth information, we measure the similarity between this clustering and the community assignments delivered for perturbed graphs.
- *Structural analysis*: How do the structural properties of the communities detected by the algorithms evolve with respect to network uncertainty?

We consider that the proposed evaluation strategy adds an additional factor that rarely is taken into account while designing or applying community detection algorithms. This problem was recently highlighted by SIAM News, June 2016 (**https://goo.gl/Xc2Gz5**). Similar approaches have been considered in the case of web ranking algorithms [20] and more recently, in the problem of influence maximization in networks [1]. The rest of the paper is organized as follows: Section 2 gives the necessary background. Section 3 presents the proposed methodological approach used to model uncertainty and the tools used to assess the sensitivity of the algorithms. Sections 4 presents the experimental results and our observations, as well as possible explanations. Section 5

Table 1: Symbols and definitions.

Symbol	Definition
$G = (V, E)$	Graph representation of datasets
$n = \|V\|,\ m = \|E\|$	Number of nodes and edges
$\mathbf{A}, \mathbf{D}$	Adjacency and degree matrices of G
a_{ij}	Entry in matrix $\mathbf{A}$
$\kappa_i = \sum_j a_{ij}$	Degree of node i

surveys the related work and finally, Section 6 presents the conclusions. Code and datasets for the paper are available at: **http://fragkiskos.me/projects/communities_sensitivity/**.

2 Preliminaries and Background

2.1 Modularity. Considering a specific partition of the network into clusters, *modularity* measures the number of edges that lie within a cluster compared to the expected number of edges in a configuration model [19]. The modularity Q of a specific partition of an undirected network into communities is defined as $Q = \frac{1}{2m} \sum_{u,v} \left[A_{uv} - \frac{\kappa_u \kappa_v}{2m}\right] \delta(c_u, c_v)$, where c_u is the community membership of node u and $\delta(c_u, c_v) = 1$ if $c_u = c_v$ (i.e., nodes u and v belong to the same community) and $\delta(c_u, c_v) = 0$ otherwise. Modularity ranges between -1 and 1 and higher positive values indicate better community structure.

2.2 Community detection algorithms. In our evaluation, we have used a representative set of widely used algorithms.

Fast greedy modularity optimization. Modularity optimization is known to be an NP-hard problem and the `FastGreedyMM` algorithm proposed by Clauset, Newman and Moore [6] was one of the first efficient methods used to solve the problem. The algorithm can be considered as an agglomerative hierarchical clustering method.

Louvain modularity optimization. The `Louvain` algorithm proposed by Blondel et al. [3], is an alternative greedy, two-phase, modularity optimization method (modularity optimization and community aggregation), which gives a better modularity value compared to `FastGreedyMM`.

Walktrap. The intuition behind the `Walktrap` algorithm [22] is that random walks on a graph tend to get "trapped" into densely connected parts, that correspond to communities. It is a hierarchical agglomerative algorithm, where the similarity measure is computed via the information given by a discrete random walk process on the graph.

Spectral clustering. The algorithmic framework of `Spectral` clustering [21, 25] includes methods that partition the nodes of a graph into clusters, using information related to the spectrum of the (normalized) graph Laplacian matrix.

Label propagation. The `LabelPropag` algorithm [23] works similarly to the *k-nearest neighbors* classifier: the idea is to assign labels to each node, following the labels of the k closest nodes in a majority vote scheme. With labels propagating through the network, densely connected groups of nodes (i.e., communities) tend to have the same label.

Metis. `Metis` [13] is a multilevel graph partitioning method consisting of three consecutive phases. In the coarsening phase, the size of the initial graph is reduced by collapsing neighboring nodes. Then, a partitioning of the graph is performed, using for instance, the *Kernighan-Lin* algorithm [14]. Finally, the partitioning is projected to consecutively larger graphs where vertices that have been collapsed together in the first phase will be in the same partition (uncoarsening phase). It returns communities of equal sizes.

Leading eigenvector. It is a modularity maximization based approach [18], based on the spectral decomposition of the modularity matrix. Thus, the `LeadingEigen` algorithm can be considered as an adaptation of spectral methods to the community detection problem (similar to the `Spectral` clustering algorithm that optimizes graph cuts).

Infomap. In this method [24], communities are detected by compressing the probability flow of random walks (proxy of information flow) in the graph. Intuitively, a community is a group of nodes where information flows quickly. `Infomap` minimizes the expected description length of an infinite random walk (called *map equation*, i.e., the theoretical lower bound of the average number of bits needed to describe one step of the random walk).

3 Sensitivity of Community Detection Algorithms

In this section, we first introduce our approach to model network uncertainty. Then, we provide evaluation tools for the sensitivity of community detection algorithms, according to the proposed functional and structural analysis.

3.1 Perturbation strategies. Here we present the perturbation strategies used to model the uncertainty and noise that may occur in real-world networks. To avoid making restrictive assumptions about the distribution of randomness, we use random edge perturbation models (addition, deletion or rewiring), that are general enough to allow for a broad characterization of various types of noise [1, 2]. Let G be the original graph and $\mathbb{G}(n)$ be a random graph model. Then, the noise model $\theta(G, \mathbb{G}, \varepsilon_a, \varepsilon_d)$ using the random graph $\mathbb{G}(n)$ gives the probability of adding/deleting an edge between two nodes (u, v) by

$$\mathbb{P}_\theta((u,v)) = \begin{cases} \varepsilon_a \mathbb{P}_\mathbb{G}((u,v)), & \text{if } (u,v) \notin E_G \\ \varepsilon_d \mathbb{P}_\mathbb{G}((u,v)), & \text{if } (u,v) \in E_G \end{cases} \quad (3.1)$$

where $\mathbb{P}_\mathbb{G}((u,v))$ is the probability of selecting the edge (u,v), ε_a and ε_d are the probabilities of edge addition and deletion respectively. It can be interpreted as follows: if the edge (u,v) already exists, then it is deleted with probability $\varepsilon_d \mathbb{P}_\mathbb{G}((u,v))$, and if not, it is added with probability $\varepsilon_a \mathbb{P}_\mathbb{G}((u,v))$. By XOR-ing the original graph with one realization $R \in \theta(G, \mathbb{G}, \varepsilon_a, \varepsilon_d)$ of the random noise model, we obtain the perturbed graph $\tilde{G} = G \oplus R$. This general formulation allows us to define the following noise models.

ERP model. This is the *uniform perturbation* model ($\mathbb{G} = \mathcal{G}(n, 1/n)$ is the Erdős-Rényi random graph), where $\mathbb{P}_\mathbb{G}((u,v)) = 1/n$ (each edge is chosen independently with probability $1/n$). The associated noise model $ERP(G, \varepsilon_a, \varepsilon_d) = \theta(G, \mathcal{G}(n, 1/n), \varepsilon_a, \varepsilon_d)$ allows us to easily control the magnitude of addition and deletion. For instance, $ERP(G, 10, 20)$ gives a perturbed graph where edges are added and removed independently with probability $10/n$ and $20/n$ respectively.

CLP model. This noise model is called *degree assortative* model, and is based on the Chung-Lu random graph model [5]. In other words, the edge probability is proportional to the product of the degrees of its endpoints, i.e., $\mathbb{P}_\mathbb{G}((u,v)) \propto \kappa_u \cdot \kappa_v$. Thus, additions and deletions under this model are biased towards the hub nodes.

ConfMP model. Here, the random graph model $\mathbb{G} = \mathcal{G}(n, \vec{\kappa})$ corresponds to the well known configuration model, where $\vec{\kappa} = \{\kappa_u\}$ is the degree sequence of the unperturbed graph [12]. More precisely, the number of edges is the same as in the original network, while we only rewire edges by a certain amount α under the constraint that $\{\kappa_u\}$ remains unchanged after the perturbation. Actually, the configuration model is used to define the probability e_{uv} that any particular edge falls between vertices u and v, which is done by joining two stubs (i.e., half-edges) incident with the pair of edges, $e_{uv} = 2m p_u p_v = 2m \dfrac{\kappa_u \kappa_v}{4m^2} = \dfrac{\kappa_u \kappa_v}{2m}$. Denoting by κ_u and κ_v the degrees in the original network, the perturbation strategy is as follows: we consider each edge sequentially, and with probability α we delete it and replace it with a new edge between a pair of vertices (u,v) selected randomly with probability e_{uv}/m.

3.2 Sensitivity: functional point of view.

In a first perspective, we are interested to quantify the variation in community assignments which is related to the similarity between two clusterings: for a given community detection algorithm, the first clustering is the one given for the original network and the second clustering is the one given for the perturbed graph (we assume that the results of the algorithm on the initial graph act as our ground truth information). To this end, we compute commonly used community structure similarity metrics [12].

Normalized Mutual Information (NMI). Let $\{x_u\}$ and $\{y_u\}$ be the community labels of node u in partitions $\mathcal{X}$ and $\mathcal{Y}$, that correspond to the original and perturbed networks respectively. The normalized mutual information $I_{norm}(X,Y)$ is given by $I_{norm}(X,Y) = \dfrac{2I(X,Y)}{H(X)+H(Y)}$ [7], which ranges between 0 (independent clusterings) and 1 (identical clusterings). It is a commonly used measure in comparative analysis of community detection algorithms.

Variation of Information (VI). This is also an information theoretic metric [9] and can be expressed using Shannon conditional entropy as $VI(X,Y) = H(X|Y) + H(Y|X)$, where $VI(X,Y)$ is the sum of the information needed to describe X given Y and the information needed to describe Y given X. Since it is a distance measure (called shared information distance), it takes values between 0 (identical clusterings, i.e., $VI(X,X) = 0$) and a maximum of $\log(n)$ (obtained when X corresponds to one community containing all nodes and Y corresponds to every node being its own community).

Adjusted Rand Index (ARI). This measure of similarity between two assignments is different in nature from the previous ones and is based on pairs counting [9]. The Rand index (RI) is defined as $RI(X,Y) = \dfrac{a+b}{a+b+c+d}$, where a is the number of pairs of elements that are in the same set in X and in the same set in Y, b is the number of pairs of elements that are in different sets in X and in different sets in Y, c is the number of pairs of elements that are in the same set in X and in different sets in Y and d is the number of pairs of elements that are in different sets in X and in the same set in Y. RI values range between 0 (independent clusterings) and 1 (identical clusterings). However we use an enhanced version [10] called *Adjusted Rand Index* (ARI) which gives scores independently of the number of clusters and samples.

3.3 Sensitivity: structural point of view.

Instead of considering the differences in community assignments, here we are interested to analyze how the structural properties of clusters found by community detection algorithms, are affected by the presence of noise. In particular, we focus our analysis on the quality and the size of the extracted clusters. Hereafter, we give the tools used in our analysis.

Conductance. This scoring criterion (also known as the normalized cut metric) is a density-based function that combines internal and external connectivity and allows us to measure the quality of a given cluster [16]. Basically, the

Table 2: Summary of real-world networks used in this study.

Network Name	Nodes	Edges	Description
AS-CAIDA	16,301	65,910	Autonomous systems network
WIKI-VOTE	7,115	103,689	Wikipedia who-votes-on-whom network
CA-GR-QC	5,242	14,496	Collaboration network of Arxiv General Relativity
CA-HEP-TH	9,877	25,998	Collaboration network of Arxiv High Energy Physics
P2P-GNUTELLA	6,301	20,777	Gnutella peer to peer network

conductance ϕ of a set of nodes $S \subset V$ is defined as $\phi(S) = \frac{\sum_{u \in S, v \notin S} A_{uv}}{\kappa_S}$, where $\sum_{u \in S, v \notin S} A_{uv}$ is the number of edges on the boundary of S and $\kappa_S = \sum_{u \in S} \kappa_u$ is the sum of the degrees of nodes in S (the degrees are computed at the level of the graph G). The conductance of the graph is defined as $\phi(G) = \min_{S \subset V} \phi(S)$, and a lower value indicates "better" community-like set of clusters.

Network Community Profile Plot (NCP plot). In order to have a deeper understanding of the clustering given by an algorithm, we are interested in characterizing the quality of detected communities as a function of their size. The NCP plot can be defined as follows: for each possible community size k, we report the quality (measured by a scoring function $f(S)$ which is conductance in our case) of the best community among all communities of size k [16]. In other words, the NCP plot displays the following quantity: $\Phi(k) = \min_{S \subset V, |S|=k} f(S)$, where $f(S) = \phi(S)$ is the scoring function and $|S|$ is the number of nodes in S. The shape of the NCP plot provides information about the community structure of the entire graph at different scales.

Spectral lower bound. We are also interested in computing a theoretical lower bound for the NCP plots. Using elements from sprectral graph theory [4], we consider the following optimization problem: $\lambda_G = \min \left\{ \frac{x^T L x}{x^T D x} : x \perp \vec{d}, x \neq 0, \right\}$, where $\vec{d}$ is a column vector of nodes' degree and $\mathbf{L} = \mathbf{D} - \mathbf{A}$ is the non-normalized Laplacian matrix of G. Then $\frac{\lambda_G}{2}$ is the spectral lower bound on the conductance of any cut in the graph. Actually, the value of λ_G is approximated by the second smallest eigenvalue (or algebraic connectivity) of the normalized Laplacian matrix $\mathbf{L}_n$. We will analyze the evolution of this bound which is characteristic of each graph, for increasing levels of noise.

4 Experimental Results

In this section we present detailed experimental results, applying the methodology proposed in Sec. 3 to real-world graphs. All the experiments were designed to answer the following questions:

Q1 (*Functional sensitivity*) To what extent the cluster assignments of community detection algorithms are sensitive to the presence of noise in the data?

Q2 (*Structural sensitivity*) How do the structural properties of communities detected by the algorithms change with increasing uncertainty?

Table 2 presents the networks used in this work [17]. Due to space constraints, we use the CA-HEP-TH graph as the running dataset throughout the paper. The experiments for the rest datasets are presented in the accompanying website: **http://fragkiskos.me/projects/communities_sensitivity/**.

4.1 Experimental setup. In all cases, we consider the graphs as unweighted and undirected. Furthermore, we extract the largest connected component and use it as a good representative of the whole graph. We also ensure that the perturbed networks are still connected graphs after adding noise. Since we have to fix the number of clusters for `Metis` and `Spectral` algorithms, we made the following choice: we set this parameter to be equal to the number of communities found by `Louvain` algorithm in the original graph[1]. Moreover, since `LabelPropag` algorithm and `Infomap` are not deterministic algorithms, we use a simple averaging procedure over ten runs for each noise level (an aggregation method is proposed in the original paper [23]). Finally, due to the random nature of perturbation models, we run the entire experiments ten times for each noise level and average the results.

4.2 Functional sensitivity analysis. Here, we give some observations and possible explanations of our findings (see also the accompanying website for the rest datasets). Figure 1 presents the values of similarity metrics (NMI, VI, ARI) as functions of noise levels ranging from 0% to 30% for all the experiments. Each row corresponds to a different perturbation model. From a first look, it seems that `Infomap` is the most robust algorithm in almost all cases. This robustness can be further quantified as follows:

OBSERVATION 1. (*Low sensitivity to noise*) *The values of NMI and ARI are still very high even for high perturbation levels (since VI is a distance measure, its values are still very low) indicating that the algorithm's output is stable (community assignments are almost identical to the ones given for the original graph).*

OBSERVATION 2. (*Consistency with noise levels*) *The shape of the curves (NMI, VI, ARI) is smooth and monotonous indicating a stable behavior, contrary to the results observed for* `LeadingEigen` *or* `Metis`.

[1] We also considered the index of the largest eigengap as explained in [25]. However, this procedure gave us a solution of only three or four clusters which is too small to be used.

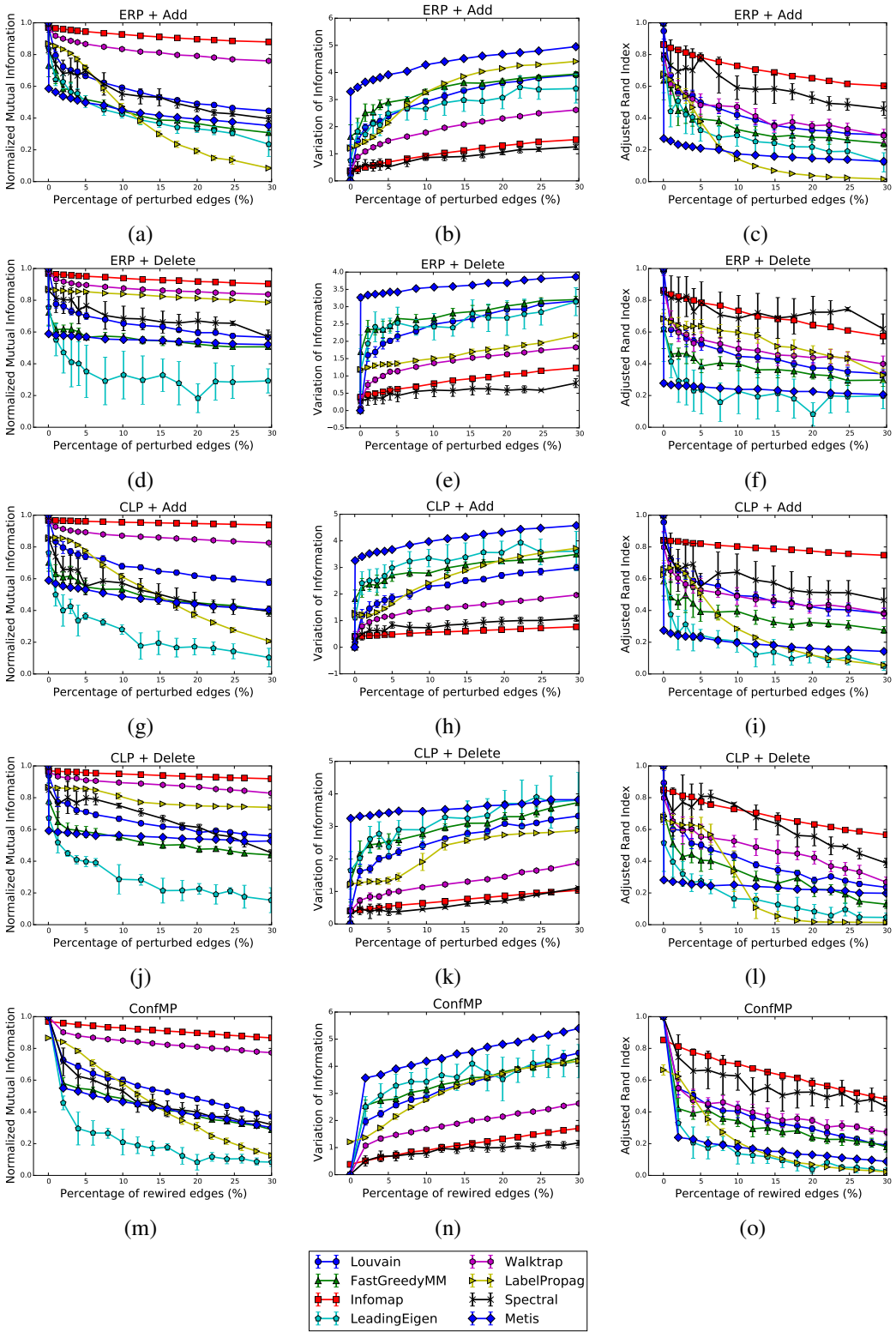

Figure 1: Functional sensitivity plots for several community detection algorithms on the CA-HEP-TH network. Rows correspond to graph perturbation models: ERP with edges addition, ERP with edges deletion, CLP with edges addition, CLP with edges deletion and ConfMP (respectively from top to button). Each column corresponds to one similarity measure: NMI, VI and ARI (respectively from left to right).

Table 3: Ranking of each algorithm according to its robustness. In each cell, we report the ranking of the considered algorithm for a given noise model in the following form: ranking according to NMI/ranking according to VI/ranking according to ARI. In our rankings, and for quite close curves, we take into account the stability of the curve as well as the variance of the metric's values. For instance, in Fig. 1 (i), the curves of `Louvain` and `Walktrap` are very close and we considered the former more robust than the later even if it ends with a lower ARI. Indeed, the errors bars for `Walktrap` are below for the majority of noise levels.

	ERP+Add	ERP+Delete	CLP+Add	CLP+Delete	ConfMP
`Louvain`	3/5/4	5/5/5	3/4/3	5/4/4	3/4/4
`FastGreedyMM`	6/6/5	6/7/6	5/5/5	7/6/5	5/5/5
`Infomap`	1/2/1	1/2/2	1/1/1	1/2/1	1/2/1
`LeadingEigen`	7/4/6	8/6/8	8/7/8	8/7/7	8/7/8
`Walktrap`	2/3/3	2/3/3	2/3/4	2/3/3	2/3/3
`LabelPropag`	8/7/8	3/4/4	7/6/7	3/5/8	7/6/7
`Spectral`	4/1/2	4/1/1	4/2/2	4/1/2	4/1/2
`Metis`	5/8/7	7/8/7	6/8/6	6/8/6	6/8/6

OBSERVATION 3. (*Low variance of the results*) *For each noise level, the results of* `Infomap` *have a very small variance (error bars in Fig. 1).*

Note that, this observation on real networks is consistent with previous ones on artificial data [15].

We have also summarized the behavior of all algorithms in Table 3. In each cell, we report the ranking of the considered algorithm for a given noise model in the following form: ranking according to NMI/ranking according to VI/ranking according to ARI. We can see that the most robust algorithms from a functional point of view are (in decreasing order): `Infomap`, `Walktrap`, `Spectral`, `Louvain`, `LabelPropag`, `FastGreedyMM`, `Metis` and `LeadingEigen`. Although an actual ranking of the algorithms is hard to be obtained, the one described above is quite persistent across all the datasets that we have examined.

We have observed that algorithms that utilize random walks (`Infomap` and `Walktrap`) perform quite well and have stable rankings across noise models and similarity metrics. `Spectral` clustering algorithm has also a good performance mainly according to VI and ARI metrics. This relative good performance could be explained by the relation between random walk and spectral clustering. In fact, by minimizing the cut size between clusters, the random walker is forced to spend more time within clusters (this is the rationale behind `Infomap` and `Walktrap` algorithms) [9]. Next, we also provide a theoretical justification of those observations concering the stability of random walk-based algorithms. Furthermore, the unnormalized Laplacian matrix is related to the transition matrix of the random walk (in our study we use the normalized Laplacian for the `Spectral` algorithm). `Metis` is a graph partitioning algorithm which is designed to produce clusters of equal sizes. This fact can explain its poor performance, since `Metis` is not able to adapt to the new perturbed community structure of the network.

Stability of the random walk-based algorithm: a spectral perspective. Here, we provide a theoretical justification of the low sensitivity observed in the `Infomap` and `Walktrap` algorithms. As we have already mentioned, both algorithms rely on random walks to extract the underlying communities. Next we consider the `Walktrap` algorithm, but similar arguments can be given for `Infomap`.

Let us consider a *random walk* over a graph G. Let $\mathbf{P}$ be the transition matrix of the random walk, where the transition from node i to j occurs with probability $P_{ij} = \frac{A_{ij}}{\kappa_i}$. In other words, the transition matrix can be expressed as $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$, where $\mathbf{D}^{-1}$ is the inverse of the diagonal degree matrix. Additionally, the probability that, starting from node i, we reach node j in t steps is given by the ij-th entry of matrix $P^t_{i,j}$. The idea behind the `Walktrap` algorithm is to define a random walk based distance function between the nodes of the graph. If two nodes i, j belong to the same community, then P^t_{ij} will be high. Based on this, the distance function r of `Walktrap` algorithm is defined as follows:

$$r_{ij} = \sqrt{\sum_{w=1}^{n} \frac{(P^t_{iw} - P^t_{jw})^2}{\kappa_w}} = \| \mathbf{D}^{-1/2} P^t_{i:} - \mathbf{D}^{-1/2} P^t_{j:} \|, \tag{4.2}$$

where $\| \cdot \|$ is the Euclidean norm and $P^t_{m:}$ is the m-th row of matrix $\mathbf{P}^t$. Then, the algorithm assigns nodes into communities following an agglomerative procedure that examines the distances between nodes.

Based on this formulation, it suffices to show that for small levels of noise that is added to the original graph G, the distance r_{ij} between nodes i, j will not be sufficiently affected, and thus, the clustering results will be slightly affected (as shown in Fig. 1). To do so, we will first express the distance function r of Eq. (4.2) using spectral information of the transition matrix $\mathbf{P}$ as

$$r_{ij} = \sqrt{\sum_{\xi=1}^{n} \lambda_\xi^{2t} \left(u_{\xi i} - u_{\xi j}\right)^2}, \tag{4.3}$$

where $\lambda_1, \ldots, \lambda_n$ and $\mathbf{u}_1, \ldots, \mathbf{u}_n$ are the eigenvalues and the right eigenvectors of matrix $\mathbf{P}$ respectively. The eigenvalues of $\mathbf{P}$ are real and satisfy the following property: $1 = \lambda_1 > \lambda_2 \geq \ldots \geq \lambda_n > -1$. Let $\tilde{\mathbf{P}} = \mathbf{P} + \mathbf{E}$ be the transition matrix of graph $\tilde{G}$, i.e., the graph after adding noise, where $\mathbf{E}$ captures the amount of perturbation added to the transition matrix; let also $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n$ and $\tilde{\mathbf{u}}_1, \ldots, \tilde{\mathbf{u}}_n$ be the eigenvalues and eigenvectors of $\tilde{\mathbf{P}}$. Our justification

relies on matrix perturbation theory and in particular on a direct application of the Davis and Kahan theorem [8, 11] for transition matrices. More precisely, the stability of the eigenvectors is determined by the eigengaps between the eigenvalues of $\mathbf{P}$, i.e., $\delta_k = |\lambda_k - \lambda_{k+1}|,\ k = 1, \ldots, n-1$. The following inequalities hold for the principal eigenpairs of $\mathbf{P}$ and $\tilde{\mathbf{P}}$:

(4.4)
$$\| \mathbf{u} - \tilde{\mathbf{u}} \| \leq \frac{4 \| \mathbf{E} \|_F}{\delta_1 - \sqrt{2} \| \mathbf{E} \|_F} \quad \text{and} \quad |\lambda - \tilde{\lambda}| \leq \sqrt{2} \| \mathbf{E} \|_F,$$

where $\| \cdot \|_F$ is the Frobenius norm of a matrix. When the amount of added or deleted edges is small, say ℓ edges, then $\| \mathbf{E} \|_F \leq \ell$. Additionally, assuming "good" community structure, then δ_1 should be sufficiently large. Thus, by the above bounds we can ensure that $\tilde{\lambda}_1 > \tilde{\lambda}_2$. Similar bounds can be given for the rest eigenpairs of $\mathbf{P}$ and $\tilde{\mathbf{P}}$ [11]. Hence, when the eigengaps of $\mathbf{P}$ are sufficiently large with respect to the amount of added noise ℓ, then the column space spanned by the corresponding eigenvectors will be far away from the one spanned by the eigenvectors of $\tilde{\mathbf{P}}$, and the spectral distance of nodes given by Eq. (4.3) will not be affected. Nevertheless, in practice, the spectral coordinates of some nodes are affected, which explains the small decrease in the clustering quality shown in Fig. 1.

4.3 Structural sensitivity analysis. Here, we present the experimental results regarding the evolution of the structural properties of communities detected by our set of algorithms. To do so, we use information from the Network Community Profile plot (NCP plot). Figure 2 depicts the results. In particular, it contains the following information: (i) *Left y-axis*: On this axis (in log scale), the green curve corresponds to the median of conductance scores over all detected communities, for each noise level. The blue curve corresponds to the global minimum of conductance scores over all detected communities, for each noise level. Finally, the magenta curve is the half of the algebraic connectivity, which is theoretically the spectral lower bound of conductance for each noise level (this value is not conditioned to a given clustering, but it is a property of each perturbed graph). (ii) *Right y-axis*: On this axis, the red curve corresponds to the number of nodes in the cluster for which the global minimum of conductance is achieved (i.e., size of the best cluster). We choose to plot results for `LabelPropag`, `Spectral` and `Infomap`, as a representative subset of the rest algorithms.

OBSERVATION 4. (*Consistency*) *In almost all cases, regardless the considered perturbation model, there is a perfect correlation between the real behavior of conductance curves (i.e., "real" in the sense that they are related to a given set of communities returned by an algorithm; see blue and green curves) and the theoretical behavior of the spectral lower bound (magenta curve) which is derived from the spectral properties of the graph Laplacian. In Table 4, we present Spearman's correlation values; we can observe that the correlation is weaker for ERP+Delete and CLP+Delete models.*

Table 4: Spearman's correlations. In each cell, for a given algorithm and noise model, we report the correlation between the global minimum of conductance scores (blue curve in Fig. 2) and the spectral lower bound of conductance (magenta curve in Fig. 2).

	ERP+Add	ERP+Delete	CLP+Add	CLP+Delete	ConfMP
`Louvain`	0.996	-0.064	0.976	0.694	0.952
`FastGreedyMM`	0.997	-0.421	0.976	0.518	0.974
`Infomap`	0.993	0.709	0.979	0.947	0.976
`LeadingEigen`	0.993	-0.515	0.932	0.406	0.915
`Walktrap`	0.993	0.709	0.974	0.947	0.971
`LabelPropag`	0.567	0.709	0.449	0.962	-0.250
`Spectral`	0.785	-0.288	0.735	0.667	0.644
`Metis`	0.988	0.229	0.964	0.121	0.974

OBSERVATION 5. (*Uptrend*) *The figures clearly show an uptrend in conductance curves for ERP+Add, CLP+Add and ConfMP. For ERP+Delete and CLP+Delete the curves are almost always stable. This behavior indicates that noise models with edges addition or rewiring (configuration model) tend to deteriorate the community structure of the network (see magenta curves). This theoretical point of view is supported by experimental results since the modules returned by the algorithms are less and less community-like with increasing noise levels (blue and green curves).*

We should also notice the particular behavior of `LabelPropag` algorithm in ERP+Add, CLP+Add and ConfMP models. In Fig. 2 (b), (h), (n), from 0% to 10% of perturbed/rewired edges, the cluster where the global minimum of conductance is achieved contains around 20 nodes. After this 10% level, we see a steep increase in the size of the best cluster which reaches $8K$ nodes (almost the entire graph). Hence, it seems that a phase transition is occurring in the algorithm's behavior. This explains the lack of robustness of `LabelPropag` observed in Fig. 1 for the three noise models (ERP+Add, CLP+Add and ConfMP). Indeed, the community assignments returned for noise levels greater than 10% are completely different from the clustering of the unperturbed graph.

5 Related Work

In the related literature, only a few works have addressed the problem of sensitivity analysis of community detection algorithms under network uncertainty. Karrer et al. [12] were based on the idea that, "good" community-like structure (relative to a division given by a community detection algorithm), is not necessarily reflected in a high modularity or a low conductance values but rather in the robustness

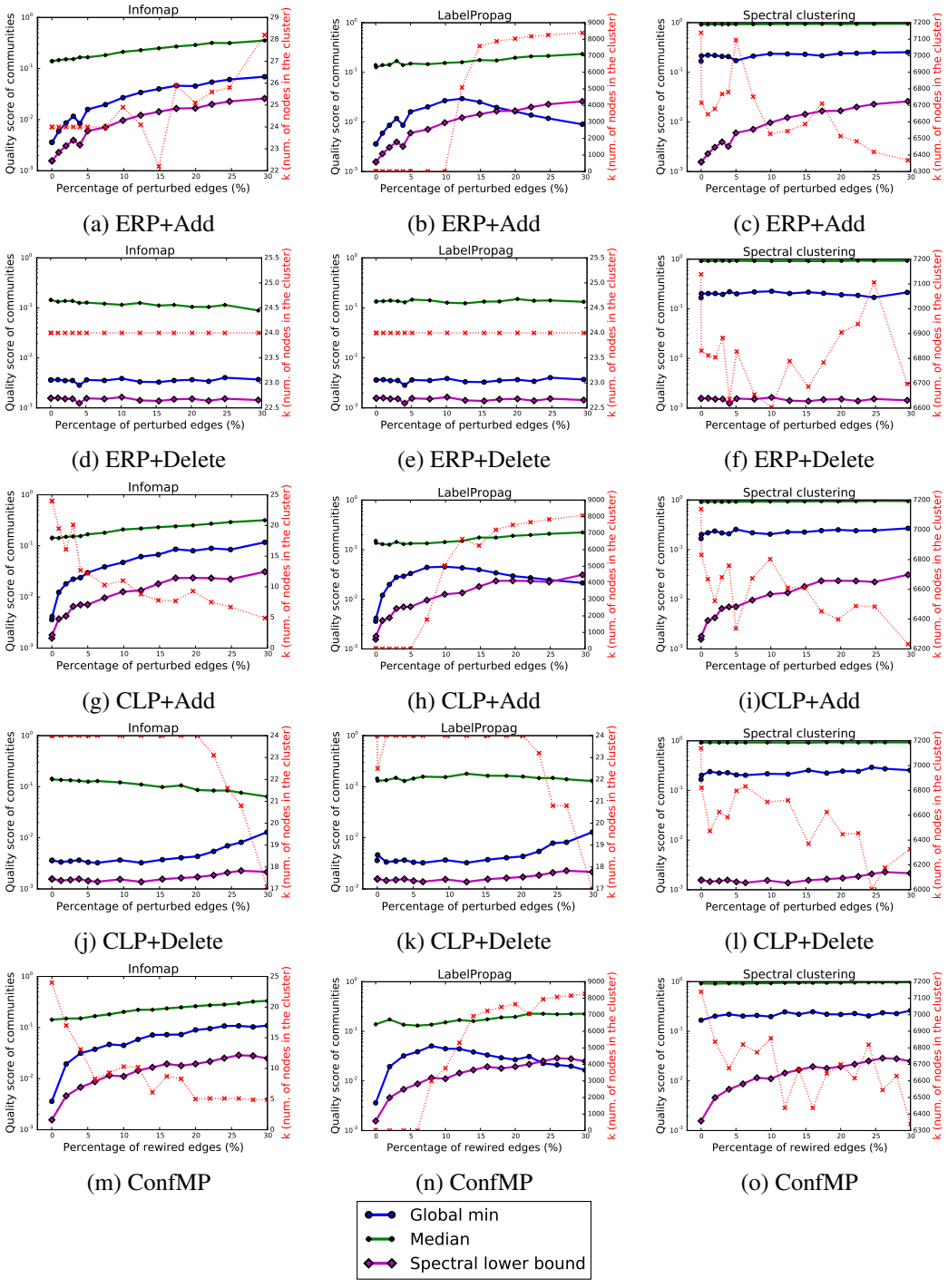

Figure 2: Structural sensitivity plots for the community detection algorithms on the CA-HEP-TH network. Rows represent graph perturbation models: ERP with edges addition, ERP with edges deletion, CLP with edges addition, CLP with edges deletion and ConfMP (respectively from top to bottom). Each column corresponds to a different algorithm: `Infomap`, `LabelPropag`, `Spectral` clustering (respectively from left to right).

of cluster assignments under graph perturbations. They applied the ConfMP model along with the spectral modularity maximization algorithm on real and artificial networks. Our approach significantly extends this study by considering a plethora of algorithms along with several perturbation models and evaluation measures. In a different spirit, Yang and Leskovec [26] developed a methodology to assess the extend of which community evaluation measures are robust against identifying ground-truth communities.

Additionally, several papers deal with the comparison of community detection algorithms (e.g., the study by Danon et al. [7] and Lancichinetti and Fourtunato [15] based on artificial networks). Contrary to those studies, our work considers a complementary evaluation tool – the one of resilience – that needs to be taken into account both by the designers of a new algorithm as well as by practitioners.

6 Concluding Remarks

In this paper, we studied the sensitivity of community detection algorithms under network uncertainty. We focused on algorithms that are widely applied in practice (e.g., the *igraph* library: http://**igraph.org**) and the ultimate goal was to introduce sensitivity under network uncertainty as a tool for the evaluation of community detection algorithms (in addition to accuracy and computational complexity). We observed that random walk based methods (i.e., `Infomap` and `Walktrap`) tend to have the lowest sensitivity to noise, in the sense that the community assignments that they provide remain highly unaffected under increasing uncertainty. We also observed a strong correlation between the quality of communities returned by the algorithms and the theoretical properties of the graph: the conductance curves have the same shape and trend. That way, we can say that most of the algorithms are stable from a structural point of view. Although a more generalized theoretical analysis – beyond the spectral and random walk based algorithms – would give further insights about the performance of the algorithms, it is quite difficult to be performed and this constitutes our current research effort. Also, we plan to examine the sensitivity of local community detection algorithms.

Acknowledgements. The authors would like to thank the anonymous reviewers for the constructive comments.

References

[1] A. Adiga, C.J. Kuhlman, H.S. Mortveit and A.K.S Vullikanti. Sensitivity of Diffusion Dynamics to Network Uncertainty. JAIR, 51(1), pp. 207-226, 2014.

[2] A. Adiga and A.K.S. Vullikanti. How Robust is the Core of a Network?. In PKDD, pages 541-556, 2013.

[3] V.D. Blondel, J.L. Guillaume, R. Lambiotte and E. Lefebvre. Fast Unfolding of Communities in Large Networks. Journal of Statistical Mechanics, 2008(10), 2008.

[4] F.R.K. Chung. Spectral Graph Theory. AMS, 1997.

[5] F. Chung and L. Lu. The average distances in random graphs with given expected degrees. PNAS, 99(25), 2002.

[6] A. Clauset, M.E. Newman and C. Moore. Finding community structure in very large networks. Phys. Rev. E, 70(6), 2004.

[7] L. Danon, A. Díaz-Guilera, J. Duch and A. Arenas. Comparing Community Structure Identification. J. Stat. Mech. Theor. Exp., 9(8), 2005.

[8] C. Davis and W. M. Kahan. The rotation of eigenvectors by a perturbation. SIAM J. Numer. Anal. 7, 1970.

[9] S. Fortunato. Community Detection in Graphs. Physics Reports, 486(3), pages 75-174, 2010.

[10] L. Hubert and P. Arabie. Comparing Partitions. J. Class, 2(1), pages 193-218, 1985.

[11] B. Hunter and T. Strohmer. Performance Analysis of Spectral Clustering on Compressed, Incomplete and Inaccurate Measurements. arXiv, 2010.

[12] B. Karrer, E. Levina and M.E.J. Newman. Robustness of Community Structure in Networks. Phys. Rev. E, 77, 2008.

[13] G. Karypis and V. Kumar. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. SIAM Journal of Scientific Computing, 20(1), pages 359-392, 1999.

[14] B.W. Kernighan and S. Lin. An Efficient Heuristic Procedure for Partitioning Graphs. Bell System Technical Journal, 49(2), pages 291-307, 1970.

[15] A. Lancichinetti and S. Fortunato. Community Detection Algorithms: A Comparative Analysis. Phys. Rev. E, 80(5), 2009.

[16] J. Leskovec, K.J. Lang, A. Dasgupta and M.W. Mahoney. Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-defined Clusters. Internet Mathematics, 6(1), pages 29-123, 2009.

[17] J. Leskovec and A. Krevl. Stanford Large Network Dataset Collection. **http://snap.stanford.edu/data**, 2014.

[18] M.E.J. Newman. Finding Community Structure in Networks Using the Eigenvectors of Matrices. Phys. Rev. E, 2006.

[19] M.E.J. Newman and M. Girvan. Finding and Evaluating Community Structure in Networks. Phys. Rev. E, 69(2), 2004.

[20] A.Y. Ng, A.X. Zheng and M.I. Jordan. Link Analysis, Eigenvectors and Stability. In IJCAI, pages 903–910, 2001.

[21] A.Y. Ng, M.I. Jordan and Y. Weiss. On Spectral Clustering: Analysis and an Algorithm. In NIPS, pages 849-856, 2002.

[22] P. Pons and M. Latapy. Computing Communities in Large Networks Using Random Walks. Computer and Information Sciences-ISCIS, Springer, pages 284-293, 2005.

[23] U.N. Raghavan, R. Albert and S. Kumara. Near Linear Time Algorithm to Detect Community Structures in Large-scale Networks. Physical Review E, 76(3), 2007.

[24] M. Rosvall and C.T. Bergstrom. Maps of Random Walks on Complex Networks Reveal Community Structure. PNAS, 105(4), pages 1118-1123, 2007.

[25] U. Von Luxburg. A Tutorial on Spectral Clustering. Statistics and Computing, 17(4), pages 395-416, 2007.

[26] J. Yang and J. Leskovec. Defining and Evaluating Network Communities Based on Ground-truth. Knowledge and Information Systems, 42(1), pages 181-213, 2015.