Graph clustering with local search optimization: the resolution bias of the objective function matters most

Twan van Laarhoven* and Elena Marchiori†
Institute for Computing and Information Sciences, Radboud University Nijmegen, The Netherlands

Results of a recent a comparative experimental assessment of methods for network community detection applied to benchmark graphs indicate that the two best methods use different objective functions but a similar local search-based optimization procedure (hereafter called LSO). This observation motivates the following research question: given the LSO optimization procedure, how much does the choice of the objective function influence the results, and in what way? In this paper we address this question empirically in a broad graph clustering context, that is, when graphs are either given as such or are k-nearest neighbor graphs generated from a given dataset. We consider normalized cut, modularity, and infomap; as well as two new objective functions. We show that all these objectives have a resolution bias, that is, they tend to prefer either small or large clusters. When removing this bias, by forcing the objective to generate a given number of clusters, LSO achieves similar performance across the considered objective functions on benchmark networks with built-in community structure. These results indicate that the resolution bias is the most important difference between objective functions in graph clustering with LSO. Spectral clustering is an alternative to LSO, which has been used to optimize the popular normalized cut and modularity objectives. We show experimentally that LSO often achieves superior performance than spectral clustering on various benchmark, real-life and k-nearest neighbor graphs. These results, the flexibility of LSO and its efficiency, provide arguments in favor of this optimization method.

I. INTRODUCTION

Clustering is the problem of grouping a set of objects in such a way that objects in each group are more 'related' to each other than to objects in the other groups. This problem is called graph clustering or network community detection when a pairwise relation of interest between objects is explicitly (yet possibly partially) observed (see, e.g., [1, 2]). When objects are described by the values they assume on a set of attributes, the problem can also be transformed into a graph clustering task, by constructing a similarity graph, for instance the k-nearest neighbor graph (see, e.g., [3, 4]). In particular Ruan [5] has shown that a slight modification of a popular network community detection algorithm is also effective when applied to k-nearest neighbor graphs. The algorithm automatically selects the value of k during the clustering process.

Due to the intrinsic difficulty of the problem, graph clustering has been tackled by many researchers, yielding a vast amount of heuristic and approximate methods as well as interesting experimental and theoretical results. We refer the reader to the surveys on this topic, e.g., [1, 2, 6]. Many methods for graph clustering are based on optimizing a global objective function. The 'optimal' clustering is then the one that minimizes the objective (throughout this paper we will use minimization; some objectives are traditionally maximized, in those cases we negate the objective function). This discrete optimization problem is computationally intractable (at least for the objectives for which hardness is known, see for instance [1, 2, 7]). Therefore all effective and scalable methods are based on heuristic and approximate techniques.

One can distinguish two main classes of heuristic for optimizing clustering objectives. The first one is based on relaxing the discrete cluster labels to continuous variables, and solving the resulting problem with spectral methods. To convert the continuous clustering to a discrete one, a separate step is used, usually k-means clustering (see, e.g., the review by Luxburg [6]). This principled spectral approach is only possible for some objectives, such as normalized cut [8] or modularity [9].

The other class of optimization methods is directly based on (local heuristic) discrete optimization. The objective is to find, among all partitions of the data set, the best one according to a given objective function (see, e.g., the review by Fortunato [1]). Although heuristic in nature, this latter approach has broader applicability since any objective function can be used.

A central issue in network community detection is the resolution limit of objective functions, which has been investigated from multiple perspectives, in particular for modularity [10–14]. In particular, Fortunato and Barthélemy [15] showed that modularity optimization is unable to detect small clusters; Good *et al.* [16] showed that the modularity function exhibits extreme degeneracies such that the globally maximum modularity partition is typically hidden among an exponential number of structurally dissimilar, high modularity solutions.

An experimental study by Lancichinetti and Fortunato [17] showed that the common spectral methods are far from optimal for the purpose of graph community detection on benchmark graphs. In their review, the two best methods are those of Blondel *et al.* [18] and Rosvall and Bergstrom [19]. Both of these methods use a similar local search optimization procedure, here called LSO, which is

* tvanlaarhoven@cs.ru.nl
† elenam@cs.ru.nl

based on moving nodes between clusters, and constructing a clustering bottom-up. In principle this procedure can be used with any graph clustering objective. Since good results are obtained with at least two different objective functions, this raises the following questions: In how far does the clustering result of LSO depend on the objective that is being optimized? And in what way does the choice of the objective function influence the results?

In order to address these questions we consider five objective functions, namely normalized cut, modularity, infomap, and two novel simple objective functions. These novel objectives are designed in such a way that (1) clusterings are better if they contain more within cluster edges, and (2) clusters should not be too small or too large. First, we analyze the resolution bias of these functions, by showing that their optimum is achieved for clusterings consisting of either relatively small or relatively large communities. Next, we apply LSO to these objective functions on benchmark graphs. Results indicate that diverse quality performance is achieved across different types of objectives. We introduce a procedure to automatically control the resolution bias of an objective function. Using this method, we force LSO to output a fixed number of clusters for each objective. Results of experiments show that the resolution bias plays a central role for the difference in performance of the objectives. When the resolution bias is 'removed' by fixing the number of clusters, the performance of LSO across these objective functions becomes much more similar.

Spectral clustering is a principled alternative to LSO, which has been used to optimize the popular normalized cut and modularity objectives [8, 9]. We show experimentally that LSO often achieves superior performance compared to spectral clustering on various benchmark, real-life and k-nearest neighbor graphs. These results confirm the findings reported by Lancichinetti and Fortunato [17] also for k-nearest neighbor graphs. In general these results, the flexibility of LSO and its efficiency, provide arguments in favor of this optimization method.

The paper is structured as follows. In Section II we present the five objectives, analyze their resolution bias, and introduce a procedure for controlling the size of clusters. In Section III we describe the LSO optimization method. In Section IV we apply LSO to the objective functions. We show that the resolution bias is the most important difference between objective functions in graph clustering with LSO, and that LSO has difficulties in optimizing specific objectives. Furthermore, we assess comparatively LSO and spectral clustering on benchmark and real-life networks and k-nearest neighbor graphs. Conclusions are reported in Section V.

II. OBJECTIVE FUNCTIONS

A. Notation

Before continuing we introduce some notation. We denote the set of nodes of the graph to be clustered by V. The weight of the edge between nodes i and j is denoted as a_{ij}. If there is no edge between two nodes, then $a_{ij} = 0$. The strength of a node is the sum of weights of all edges incident to it, $s_i = \sum_{j \in V} a_{ij}$. For unweighted graphs, the strength of a node is equal to its degree. The volume of a set of nodes X is the sum of strengths of all nodes in X, $v_X = \sum_{i \in X} s_i$. The total volume of the graph is $M = v_V$. For an unweighted undirected graph this is equal to twice the number of edges.

We say that an edge is 'within' cluster X if both end points are in X. Then $w_X = \sum_{i,j \in X} a_{ij}$ is the total weight of edges within X.

The normalized volume of a cluster is $\widehat{v}_X = v_X/M$, and the normalized within weight is $\widehat{w}_X = w_X/M$.

In this paper a clustering C of a graph is a partition of the nodes. That is, a set of disjoint sets of nodes which we call clusters, that together cover all nodes. So, every node is in exactly one cluster.

B. The considered objectives

Different objective functions have been proposed for graph clustering. Perhaps the most well known is normalized cut [8], which is defined as

$$Q_{\text{NCut}}(C) = \sum_{c \in C} \frac{v_c - w_c}{v_c}. \tag{1}$$

Another common objective is modularity, introduced by Girvan and Newman [20],

$$Q_{\text{modularity}}(C) = -\sum_{c \in C} \left(\widehat{w}_c - \widehat{v}_c^2\right). \tag{2}$$

Note that this is the negative of the usual definition, so that the optimum is a minimum as with normalized cut. Both of these objectives can be written as a sum over all clusters,

$$Q(C) = \sum_{c \in C} q(c), \tag{3}$$

for some function q. This means that it makes sense to look at the objective value of just a subset of the clusters, or of the clustering of just a subset of the nodes.

A notable objective that does not follow this pattern is infomap [19]. This objective is based on the length of a code for paths through the graph. In addition to a sum of per-cluster scores, infomap also include a global term

TABLE I. Objective functions considered in this paper.

Objective	Expression
normalized cut	$\sum_{c\in C}(v_c - w_c)/v_c$
modularity	$-\sum_{c\in C}(\widehat{w}_c - \widehat{v}_c^2\ )$
w-log-v	$\sum_{c\in C}\widehat{w}_c \log(\widehat{v}_c)$
parabola	$\sum_{c\in C}\widehat{w}_c(\widehat{v}_c - 1)$
infomap	$\sum_{c\in C} h(\widehat{v}_c + \widehat{v}_c - \widehat{w}_c)$ $- 2\sum_{c\in C} h(\widehat{v}_c - \widehat{w}_c) + h\big(\sum_{c\in C}(\widehat{v}_c - \widehat{w}_c)\big)$

based on the probability of an edge exiting a cluster[21],

$$Q_{\text{infomap}}(C) = \sum_{c\in C} h\big(\widehat{v}_c + \widehat{v}_c - \widehat{w}_c\big) - 2\sum_{c\in C} h\big(\widehat{v}_c - \widehat{w}_c\big) + h\Big(\sum_{c\in C}(\widehat{v}_c - \widehat{w}_c)\Big), \tag{4}$$

where $h(p) = p\log(p)$. The original infomap objective contains an additional term,

$$Q_{\text{infomap[19]}}(C) = Q_{\text{infomap}}(C) - \sum_{i\in V} h(\widehat{v}_{\{i\}}), \tag{5}$$

which is needed to make the objective correspond to a code length. However, since this last term is the same for all clusterings, we don't include it. In addition, without this extra term, the objective value of the trivial clustering with one cluster is exactly 0.

There are many more possible objective functions that could be used for graph clusterings. Some considerations for designing such functions are that:

1. All else being equal, clusters are better if they contain more within cluster edges.

2. Clusters should not be too small or too large.

For the first consideration, we can look at the ratio between the volume of a cluster and the number of within edges. For the second consideration, we use a weight function $f(\widehat{v}_c)$ where $f(0) = f(1) = 0$, while $f(x) < 0$ for $0 < x < 1$. Combining these ingredients leads to an objective of the form

$$Q(C) = \sum_{c\in C} \frac{\widehat{w}_c}{\widehat{v}_c} f(\widehat{v}_c). \tag{6}$$

Two simple functions that fit the criteria for f are the parabola $f(x) = x(x-1)$ and the function $h(x) = x\log(x)$. They give the objectives

$$Q_{\text{parabola}}(C) = \sum_{c\in C}\big(\widehat{w}_c\widehat{v}_c - \widehat{w}_c\big), \tag{7}$$

and

$$Q_{\text{w-log-v}}(C) = \sum_{c\in C} \widehat{w}_c \log(\widehat{v}_c). \tag{8}$$

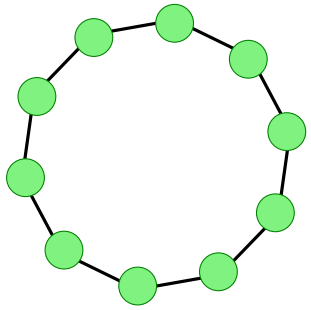

FIG. 1. (Color online) Model graph for showing the resolution limits. The circles represent strongly connected 'modules' with $q-r$ internal edges, while the lines represent r edges each.

Of course, there are infinite other possibilities. We focus on these two because the former is similar to modularity, while the latter resembles infomap while being much simpler.

Table I lists all the different graph clustering objectives that we will consider in this paper. Many other objectives have been discussed in the literature, see [1] for an overview. Many of them do not apply in our setting, because they assign a score to a single cluster, not to a clustering. Therefore it is not clear how the cluster scores should be combined into a score for a clustering. When the number of clusters is fixed one could use the sum of scores, but when the number of clusters is allowed to vary this will often not give a good objective.

C. Resolution biases

It was shown by Fortunato and Barthélemy [15] that the modularity objective has a resolution limit, in the sense that it tends to combine small communities into larger ones. Specifically, in a network which has L edges, there is a characteristic number of edges, such that communities with less than $\sqrt{L/2}$ edges are not visible. Kumpula *et al.* [22] have generalized this result by showing that the graph clustering framework introduced by Reichardt and Bornholdt [13] also has a resolution threshold. The model contains a parameter by which this threshold can be tuned, but no a priori principle is known to select the proper value. They conclude that single global optimization criteria do not seem capable for detecting all communities if their size distribution is broad [22].

In the sequel we show that the other clustering objectives here considered have resolution limits. In fact, these are not just limits, but a general bias towards certain cluster sizes. For example, the w-log-v objective has a resolution limit at smaller cluster sizes, and it always leads to smaller clusters than modularity.

Consider a graph that has n densely connected *modules*, which are loosely connected in a ring [15]. Fig. 1 illustrates such a graph. The modules themselves could

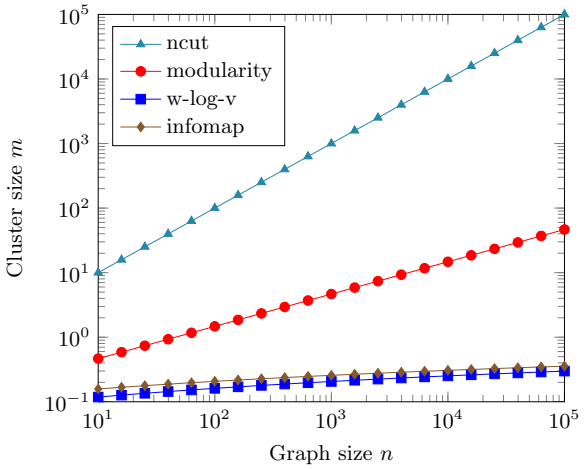

FIG. 2. (Color online) The resolution limits of graph clustering objectives as a function of the graph size. We used $r = 1$ and $q = 46$, which corresponds to modules that are cliques with 10 nodes. The resolution limit of the parabola objective is the same as that of modularity.

be single nodes, cliques or other subgraphs, we are only interested in their volume. In particular, imagine each module having $q - r$ internal edges, and connected to both of its neighbors with r edges each. The volume of a single module X is then $v_X = 2q$, while the volume of the entire graph is $M = 2nq$.

A cluster X_m consisting of m adjacent modules will have normalized volume $\widehat{v}_{X_m} = m/n$. And since all but $2r$ of the edges will be within the cluster, the normalized within weight will be $\widehat{w}_{X_m} = (m - r/q)/n$. By symmetry, we would expect all clusters in the optimal clustering C^* to have the same size (assuming m divides n). There will then be n/m such clusters. So, the total modularity of this clustering is

$$Q_{\text{modularity}}(C^*) = -\frac{n}{m}\left(\frac{m - r/q}{n} - \left(\frac{m}{n}\right)^2\right). \quad (9)$$

This expression has a minimum at $m = \sqrt{nr/q}$. So, the larger the graph, or the less dense the connections in each module, the larger the clusters that are found. The parabola objective reaches a minimum at the same point.

For the w-log-v objective

$$Q_{\text{w-log-v}}(C^*) = \frac{n}{m}\left(\frac{m}{n} - \frac{r}{qn}\right)\log\left(\frac{m}{n}\right). \quad (10)$$

The optimum is at $m = W(enr/q)r/q$ where W is the Lambert W function.

The normalized and ratio cut objectives behave differently,

$$Q_{\text{NCut}}(C^*) = \frac{n}{m}\frac{(r/q)(1/n)}{m/n} = \frac{nr}{m^2 q}. \quad (11)$$

This expression has no minimum value, it decreases as m gets larger. Since the size of a cluster can not be larger than n, the actual optimum is at $m = n$, i.e. when all modules are in a single cluster.

Finally, for infomap there is no analytical expression for the optimum cluster size, but it can be easily calculated numerically. Fig. 2 shows the cluster size of the optimal clustering as a function of the number of modules. This optimal size was found by numerical optimization of the objective in terms of the cluster size m. For this figure we have used modules with $r = 1$ and $q = 46$, which corresponds to 45 internal edges, i.e. cliques with 10 nodes. Other module sizes give qualitatively similar results. For each of the objectives, the optimal cluster size depends only on the ratio r/q and the number of modules n.

Note that in this figure, for the w-log-v and infomap objectives the theoretically optimal clustering always has less than 1 clique per cluster, which in practice means that the cliques are perfectly clusterable. To actually see the resolution limit in action for these objective, the number of cliques must be *very* large. For example for the w-log-v objective, the objective value for $m = 2$ overtakes $m = 1$ when $n > 2^{91} \approx 10^{27}$.

The resolution bias discussed in this section reflect preferences towards certain sizes of clusters, in a situation where all vertices are similar. There are other biases that come into play when the graph is less uniform and when the sizes of clusters will differ [12].

D. Controlling the size of clusters

Several generalizations of the modularity objective have been proposed that allow for control over the size of the clusters [11, 13, 23]. Each of these objectives has a parameter that controls the trade-off between the size of the clusters and the strength of edges within clusters. For example, the objective introduced by Reichardt and Bornholdt [13] is, in our notation,

$$Q_{\text{RB}}(C, \gamma) = -\sum_{c \in C}\left(\widehat{w}_c - \gamma \widehat{v}_c^2\right). \quad (12)$$

In this paper we also consider other objective functions besides modularity, and hence we would like to add similar size-control parameters to them. In the previous section we have shown that the size of the clusters depends on the size of the graph. Often this dependency is implicit, through the use of the normalized volume and normalized within weight. This dependence can be used to control the cluster sizes.

The idea is to embed the graph in a larger graph, with total volume αM, and thereby change the optimal clustering. Since objective functions such as modularity are a sum over clusters in the form of (3), we can look at the contribution to the modularity of a clustering C of the

original graph. Denote this contribution by

$$Q^{\text{embed}}_{\text{modularity}}(C,\alpha) = -\sum_{c\in C}\left(\widehat{w}_c/\alpha - (\widehat{v}_c/\alpha)^2\right)$$
$$= \frac{1}{\alpha^2}\Big(Q_{\text{modularity}}(C) - (\alpha-1)\sum_{c\in C}\widehat{w}_c\Big). \quad (13)$$

The optimal clustering does not change when the objective function is multiplied by a constant. Therefore, embedding within a larger graph is equivalent to adding a term to the objective,

$$Q^{+}(C,\beta) = Q(C) + \beta\sum_{c\in C}\widehat{w}_c. \quad (14)$$

This holds also for the parabola and the w-log-v objectives.

On the other hand, the normalized cut objective does not depend on the size of the graph at all. Despite this, we can still use (14) to adjust that objective.

In this way we get a *family* of objective functions parameterized by β for each original objective function. Note also that $Q^{+}_{\text{modularity}}$ is equivalent to Q_{RB} with $\gamma = 1+\beta$; and it is equivalent to Q_{NL} introduced in [11] with $t = 1/(1+\beta)$.

By adjusting the parameter β, the size of the clusters can be controlled. A negative value of β corresponds to embedding the graph in a larger one, so it will lead to fewer larger clusters. A positive β will lead to more and smaller clusters. However, we have to be careful with large positive values of β, since that punishes within cluster edges, instead of rewarding them.

Since a large part of the difference between objective functions lies in the different preferred cluster sizes, this added flexibility might be enough to get rid of much of these differences. Suppose, for example that the number of clusters is known. Then we can use binary search to look for a value of β that leads to the desired number of clusters. The resolution bias of the objective is then no longer important, since by fixing the number clusters we also fix their average size.

III. OPTIMIZATION METHOD

The optimization procedure that we use is the local search method developed by Blondel *et al.* [18], which we call LSO. They proposed it for optimizing modularity, but the same method can also be used for any other graph clustering objective. The method is very fast, and can deal with millions of nodes in seconds. We will briefly describe the algorithm here.

Initially, each node is assigned to a singleton cluster. Then, iteratively, nodes are moved between clusters as long as the objective improves. For each node, only moves to neighboring cluster are considered; where neighboring clusters are those clusters that contain neighboring nodes. The nodes are visited in a random order.

The most expensive part of the algorithm is recomputing the value of the objective function. For objectives that are written as a sum over the clusters, as in (3), this computation can be done efficiently, because only two terms of the sum change when a node is moved between clusters.

Because the objective improves with each move, eventually a local optimum will be reached. However, the clusters in this local optimum will often be too small. It is just that they can not be improved by moving *single nodes*. We will call the clusters found at this point *small clusters*. The next step is to repeat the optimization procedure, but this time moving entire small clusters instead of single nodes. Effectively, we are then clustering a condensed graph, where each node in the condensed graph is a small cluster.

The step of moving small clusters is again repeated until convergence. The clusters at that point become the new small clusters. At some point no small clusters will be moved, and then the algorithm stops.

Several variations to this basic recipe are possible. For instance, if the clusters become too large, one could apply the clustering algorithm from scratch to only the nodes in a single cluster. This might lead to a better optimum. However, we do not find this step to improve the results in our experiments. Another improvement is to simply run the algorithm several times with different random seeds, and to pick the best solution.

IV. EXPERIMENTS

A. Community detection benchmarks

We consider the LFR graph generator by Lancichinetti, Fortunato, and Radicchi [24] which constructs networks with built-in community structure. In this benchmark, the size of each cluster is drawn from a power-law distribution; as is the degree of each node. These benchmarks are specifically constructed to closely resemble real world graphs. Indeed, it has previously been observed that real world graphs also have such a power-law degree distribution [25].

The LFR model has several parameters. The most important is the mixing parameter μ, which controls the fraction of edges that are between clusters. Essentially this is the amount of noise in the graph. If $\mu = 0$ all edges are within cluster edges, if $\mu = 1$ all edges are between nodes in different clusters.

Other parameters control the number of nodes, the distributions of cluster sizes, the distribution of degrees, etc. If something is known about the target graph, then these parameters should be chosen to match that graph. However, in this paper we do not try to match any particular graph. We therefore follow the settings used by Lancichinetti and Fortunato [17]. They describe four benchmarks. Two with 'small clusters' of between 10 and 50 nodes, and two with 'large clusters' of between 20 and

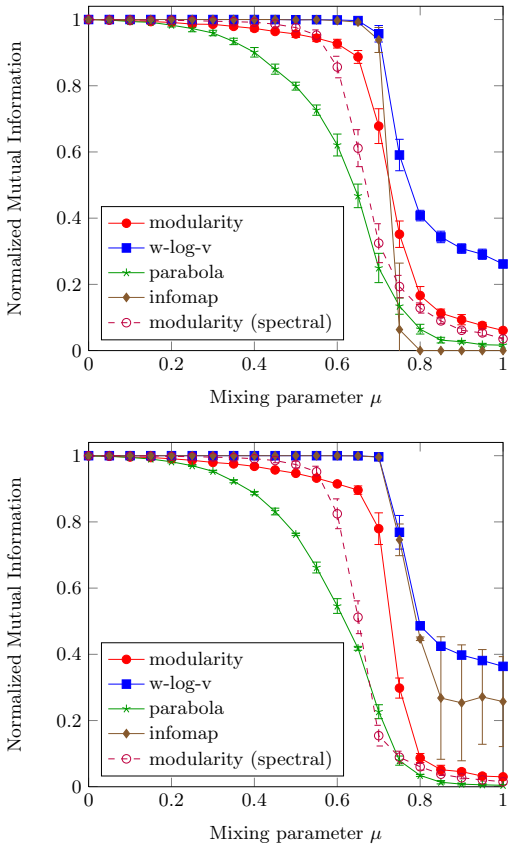

FIG. 3. (Color online) Normalized mutual information as a function of the mixing parameter, for various objective functions; on the Small 1000 dataset (top), and the Big 5000 dataset (bottom). The error bars indicate standard deviation.

100 nodes. Each graph has either 1000 or 5000 nodes in total.

To measure the quality of a clustering, we compare it to the ground truth with the normalized mutual information (NMI) metric [26],

$$\hat{I}(C_1, C_2) = \frac{2I(C_1, C_2)}{H(C_1) + H(C_2)}, \tag{15}$$

where I is mutual information and H is entropy. Fig. 3 shows the normalized mutual information as a function of the mixing parameter for the different clustering objectives. We used LSO to optimize all objectives. We did not include the normalized cut objective, since without adjustment this objective always leads to a single cluster. We only show the results for the benchmark with 1000 nodes and small clusters and the benchmark with 5000 nodes and large clusters. The results for the other two benchmarks are similar.

For comparison, besides LSO, we also include two spectral clustering methods in our experiments. First of all a simple method that approximately minimizes the normalized cut objective by solving a generalized eigenvalue problem for the graph Laplacian, and then finds discrete clusters using k-means [8]. Secondly the method of Newman [9], which uses eigenvectors of the modularity matrix. Based on these eigenvectors a few (two or three) clusters are found, which are then recursively subdivided until the optimal modularity is reached. The clusterings are further optimized by a Kernighan-Lin style algorithm [27], which moves single nodes around in a similar fashion to the first iteration of the LSO algorithm. These two methods represent the complete opposite approach to clustering. Whereas LSO uses a greedy search to grow clusters from the bottom up, these spectral methods use a smooth approximation to repeatedly subdivide the graph.

Optimizing the w-log-v and infomap objectives always leads to a perfect recovery of the clustering up to $\mu = 0.65$ on the small dataset and $\mu = 0.7$ on the big dataset. For higher μ, infomap suddenly gives a clustering with normalized mutual information 0. This is the clustering where all nodes are put into a single cluster. Notice the large standard deviation on the Big 5000 dataset. In some cases the optimizer finds the trivial clustering, and in other cases it finds a clustering comparable to that found with the w-log-v objective.

The w-log-v objective does not have the instability of the infomap objective, and the performance normalized mutual information decreases more gradually. The other two objectives, modularity and parabola, perform worse. As we show next, this mainly due to the failure to recover the right number of clusters.

When the true number of clusters is known, we can adjust the objective to get the desired number of clusters, as described in section II D. In this case it is also possible to use spectral clustering to optimize the normalized cut objective, which requires the number of clusters as an input parameter. Fig. 4 shows the results on the same benchmark graphs when forcing the number of clusters to be equal to the number of clusters in the ground truth.

The behavior of the different objective functions is now very similar. However, with the normalized cut objective we are still unable to find the right clustering. This is due only to the optimizer, because when normalized cut is optimized with spectral clustering, the correct clustering is found.

B. Objectives versus optimization

One might wonder in how far the results of these experiments depend on the objective function, and how much they depend on the objective that is being optimized. To distinguish between the two, we compare the objective value for the clustering found by the LSO algorithm to the objective value for the ground truth clustering. If the objective is lower at the ground truth clustering, then this indicates that optimizer has failed to find a good enough clustering. On the other hand, if the objective of the ground truth clustering is higher, then the optimizer has found a clustering that is 'better than the ground

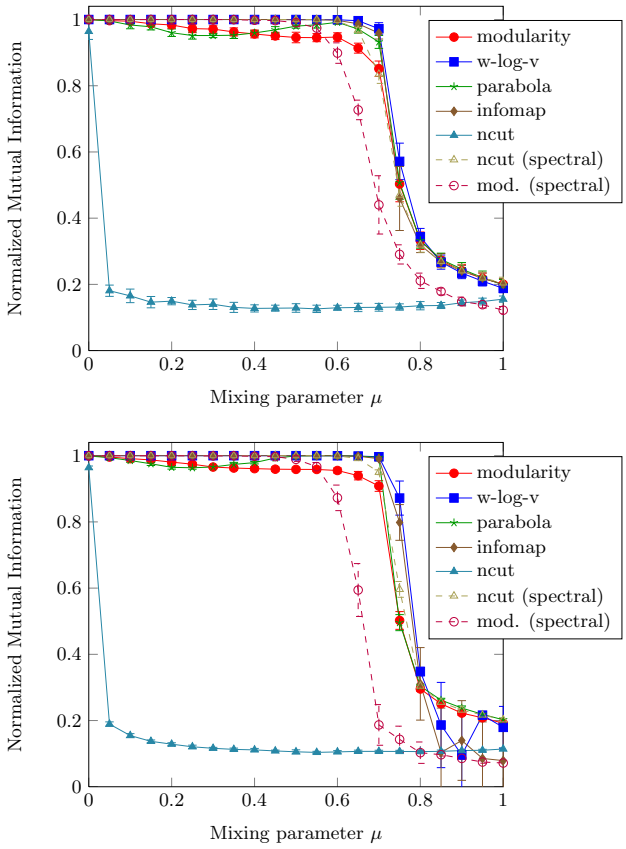

FIG. 4. (Color online) Normalized mutual information as a function of the mixing parameter, when the number of clusters has been fixed to the actual number of clusters. We show results for the Small 1000 dataset (top), and the Big 5000 dataset (bottom). The error bars indicate standard deviation.

truth' according to the objective. That means that the objective is unsuitable in this situation. As a baseline, we also compare with a clustering obtained on a randomly rewired graph with the same degree distribution.

Fig. 5 shows the value of the objective functions for different mixing parameters. We can see that the objective value of the ground truth crosses that of the randomly rewired graph at different points for different objectives. Beyond this point, there is little hope of recovering the true clustering, since the graph has then no more cluster structure than a random one. We can also see cases where the optimizer fails, such as with the w-log-v objective at $\mu = 0.75$. Here the ground truth has a better objective value than the clustering found by the optimizer. Repeating the optimization 20 times leads to a slightly better optimum, but not yet to the ground truth. Even more repetitions can further improve performance, but only slightly.

The parabola objective shows a different picture. The clustering found by the optimizer has a lower objective value than the ground truth in many cases. This means that the LSO method often fails to find the optimal clustering or one close to it. The clustering that is found instead has too few clusters. In section II C we showed that the parabola objective has the same resolution bias as modularity, while optimizing modularity with LSO does not give a clustering with a lower objective value than the ground truth. This means that the resolution bias does not tell the whole story. Another important aspect of the results seem to be how easy the objective is to optimize with LSO.

With the infomap objective, the clustering found for the randomly rewired graph always has objective value 0. This corresponds to a clustering where all nodes belong to the same cluster. This clustering is always a possible one, but it is not always found by the optimizer. For example, at $\mu = 0.7$, the optimizer sometimes finds an infomap objective value that is greater than 0. In these cases the optimizer is stuck in a local minimum that is not globally optimal.

C. Real world community graphs

We next applied the optimizer to several small real-world networks. We only looked at networks for which some kind of ground truth clustering is know. For other networks, often only a modularity score is reported in the literature. But since we use several different objective functions, this makes no sense in our context. The networks we considered are:

- Zachary's karate club [28].
- Football: A network of American college football games [20].
- Political books: A network of books about US politics [29]. The clusters are left-wing, right-wing and neutral books.
- Political blogs: Hyperlinks between weblogs on US politics [30].

In each case, we force the number of clusters found by the methods to be the same as the number of clusters in the dataset. The first part of table II gives the results of these experiments. In most experiments the spectral methods give the best results. We believe that this is due to the small number of clusters. The difference is especially large for the Political Blogs dataset, which is the largest dataset in this experiment. Since the spectral methods start with a single large cluster, the final solution with just two or three clusters is a relatively close to this starting point. In contrast, the LSO method starts from singleton clusters, which are gradually merged.

The second part of the table gives results when the number of clusters is not fixed. In these experiments the results are more varied. Observe that for the football dataset the modularity and parabola objective no longer find the same clustering as the other objectives, instead giving fewer clusters. This is due to the biases of these objectives.

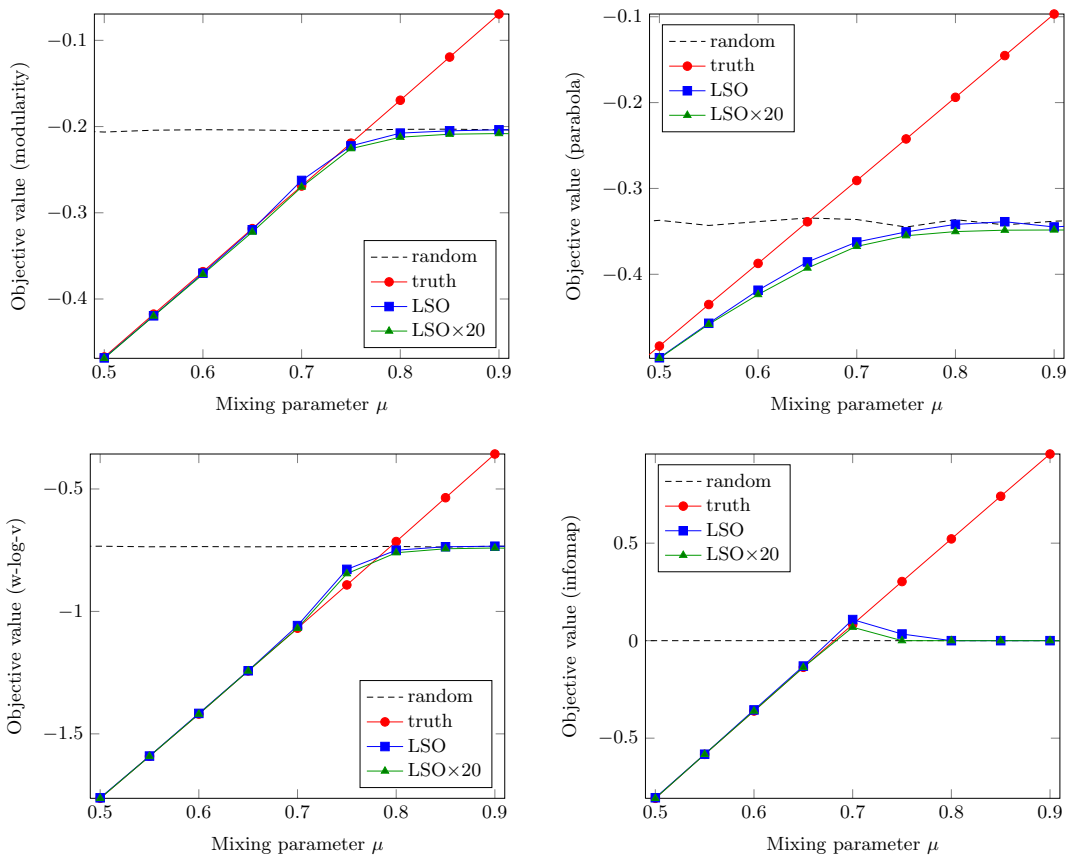

FIG. 5. (Color online) The value of the objective as a function of the mixing parameter on the Small 1000 dataset. 'random' is the objective value reached by the optimizer on a randomly rewired graph, 'truth' is the objective value of the ground truth clustering. LSO is the value reached by the optimizer, and LSO×20 is the best objective value out of 20 restarts. The objectives shown are: modularity (top left), parabola (top right), w-log-v (bottom left), and infomap (bottom-right).

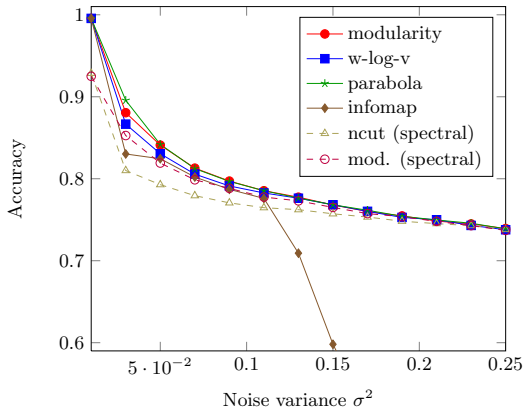

FIG. 6. (Color online) Accuracy of the clustering methods on the two moons dataset, as a function of the variance of the noise.

D. Artificial nearest neighbor data

We now consider the applicability of LSO to clustering nearest neighbor graphs. We follow the setup from Bühler and Hein [31].

First we ran experiments on the two moons dataset. This dataset consists of points on two half-circles, that are offset from each other, embedded in a d dimensional space, and have added Gaussian noise.

For each point x_i in the dataset we add edges to its k-nearest neighbors with the weights

$$a'_{ij} = e^{-4\|x_i - x_j\|^2 / \|x_i - x_i^k\|^2},$$

where x_i^k is the k-nearest neighbor of x_i. To make the graph symmetric, we take the maximum weight over the two edge directions, $a_{ij} = \max(a'_{ij}, a'_{ji})$.

In our experiments we used $n = 2000$ points of dimension $d = 100$, and $k = 10$ neighbors of each point. The optimizer is restricted to finding 2 clusters. We evaluate the performance by looking at the leave-one-out accuracy. That is, the fraction of points that have the same label as the majority of the other nodes in the same cluster. Fig. 6 shows the accuracy as a function of the variance of the noise.

The results are in some ways opposite to those on the LFR benchmark. For these K nearest neighbor graphs,

TABLE II. The normalized mutual information for optimizing the various objectives on real world networks. The best results are indicated in boldface. In the first part of the table the number of clusters is forced equal to the true number, which is shown in the 'clusters' column. In the second part the number of clusters is left free, in parenthesis is the number of clusters that are found for each method.

	Dataset		LSO				spectral	
Name	vertices	clusters	modularity	parabola	w-log-v	infomap	ncut	modularity
Zachary	34	2	67.7%	67.7%	67.7%	30.1%	**73.2%**	67.7%
Football	115	12	**92.4%**	**92.4%**	**92.4%**	**92.4%**	**92.4%**	89.5%
Pol. Books	105	3	55.4%	55.4%	**57.4%**	**57.4%**	54.2%	54.2%
Pol. Blogs	1490	2	11.5%	11.5%	11.5%	11.5%	0.9%	**52.2%**
Zachary	34	*free*	**58.8%** (4)	**58.8%** (4)	42.8% (6)	56.8% (3)	-	**58.8%** (4)
Football	115	*free*	89.0% (10)	82.0% (8)	**92.4%** (12)	**92.4%** (12)	-	85.2% (9)
Pol. Books	105	*free*	**56.0%** (5)	46.9% (6)	40.7% (11)	53.7% (5)	-	52.1% (4)
Pol. Blogs	1490	*free*	37.2% (278)	33.9% (280)	25.1% (314)	33.9% (303)	-	**52.2%** (2)

the modularity and parabola objectives outperform w-log-v and infomap. We conjecture that this has to do with the resolution bias of the methods. In the LFR benchmarks we search for more clusters, around 40, compared to only 2 in this experiment. Thus, the objectives that are biased towards larger clusters will perform better here. However, at the moment we have no proof or additional evidence to support this conjecture.

E. Real world nearest neighbor datasets

We used the same construction of a nearest neighbor graph outlined in the previous paragraph also on real-world and UCI datasets. In each case, we force the number of clusters to be the same as the number of classes in the dataset. Table III contains the results of these experiments. Since this is a classification task, we have also measured the performance with leave one out accuracy instead of normalized mutual information. The LOO accuracy is the fraction of points that would be correctly classified if the most common label among all other points in the same cluster is used as that cluster's label. Table IV contains the LOO accuracy results.

On the iris dataset all methods except spectral modularity optimization achieve the same high accuracy. This can be explained by the small size of the dataset and the relatively easy classification task. The iris dataset was previously used in a comparison of different multi-resolution methods [32], the accuracy reported in that paper is the same 96% that we found. On the MNIST and USPS datasets, LSO significantly outperforms spectral clustering. These datasets have many classes, many features and are not completely balanced. On the other hand, on the ringnorm dataset spectral normalized cut optimization perform much better than other methods.Overall, as on the two-moons dataset, the parabola objective gives the best results.

The second part of the table IV shows that, when the number of clusters is not fixed, the w-log-v objective has the highest or close to the highest accuracy in all cases. But this is merely because the w-log-v objective has an optimum with the most clusters, and the accuracy is nearly always higher with such a more fine grained clustering. On the other hand, the normalized mutual information is higher when the number of clusters is closer to the true number of clusters. In this regard, the modularity and parabola objectives give the best results.

V. CONCLUSIONS

The results of our investigation show that the choice of objective function matters for graph clustering with LSO. The objective function has two important main effects.

First of all we have shown that different graph clustering objectives have different resolution biases. These form the largest difference between the different objectives. Our experiments show that on benchmark network graphs with built-in community structure, when controlling the number of clusters, the clusterings found with different objective functions are very similar. However, when the number of clusters is not fixed, the resolution bias has a large influence on the performance of the method.

Secondly, some objectives are easier to optimize with LSO than others. For example, in the experiments on the LFR benchmarks, the clustering found with the parabola objective function is often not optimal for that objective. In addition, optimizing other objectives such as normalized cut turns out to be very hard. For that objective function spectral methods are more suitable.

For nearest neighbor graphs, LSO often significantly outperforms spectral clustering, while never performing significantly worse. When the number of clusters is fixed to a small number, the modularity and parabola objectives give the best results. When the number of clusters is not fixed, the w-log-v objective finds the most clus-

TABLE III. The normalized mutual information of various methods on real world datasets. The best results are indicated in boldface. In the first part of the table the number of clusters is forced equal to the true number, which is shown in the 'clusters' column. In the second part the number of clusters is left free, in parenthesis is the number of clusters that are found for each method.

	Dataset		LSO				spectral	
Name	vertices	clusters	modularity	parabola	w-log-v	infomap	ncut	modularity
MNIST	70000	10	91.1%	**91.9%**	87.7%	87.6%	76.3%	22.7%
USPS	9298	10	87.9%	87.9%	**89.8%**	89.6%	79.9%	35.7%
iris	150	3	**86.4%**	**86.4%**	**86.4%**	**86.4%**	**86.4%**	74.0%
coil20	1440	20	92.5%	92.4%	**92.8%**	92.4%	91.9%	49.4%
waveform	5000	2	41.7%	**42.1%**	41.6%	41.3%	36.5%	12.6%
ringnorm	7400	2	7.8%	9.0%	0.0%	0.0%	**14.4%**	2.0%
faces	624	20	86.2%	86.2%	85.7%	85.7%	**86.4%**	62.8%
MNIST	70000	*free*	**82.8%** (18)	82.6% (18)	45.0% (2058)	46.7% (1523)	-	51.8% (382)
USPS	9298	*free*	84.0% (17)	**84.4%** (16)	52.4% (473)	55.0% (332)	-	59.1% (122)
iris	150	*free*	60.4% (9)	61.1% (9)	51.6% (18)	54.5% (15)	-	**64.5%** (8)
coil20	1440	*free*	88.7% (27)	**88.8%** (27)	74.1% (143)	76.7% (107)	-	75.7% (110)
waveform	5000	*free*	28.4% (6)	28.1% (6)	12.8% (298)	13.8% (193)	-	**29.3%** (5)
ringnorm	7400	*free*	2.4% (19)	3.8% (5)	**4.8%** (559)	4.6% (469)	-	3.4% (8)
faces	624	*free*	**88.4%** (32)	**88.4%** (32)	80.4% (92)	81.9% (76)	-	80.3% (61)

TABLE IV. The classification accuracy of various methods on real world datasets. The best results are indicated in boldface. In the first part of the table the number of clusters is forced equal to the true number, which is shown in the 'clusters' column. In the second part the number of clusters is left free, in parenthesis is the number of clusters that are found for each method.

	Dataset		LSO				spectral	
Name	vertices	clusters	modularity	parabola	w-log-v	infomap	ncut	modularity
MNIST	70000	10	95.0%	**96.8%**	84.2%	84.2%	75.8%	31.0%
USPS	9298	10	89.7%	89.7%	91.2%	**92.6%**	80.5%	43.1%
iris	150	3	**96.0%**	**96.0%**	**96.0%**	**96.0%**	**96.0%**	83.3%
coil20	1440	20	85.6%	85.2%	82.9%	82.8%	**86.9%**	43.0%
waveform	5000	2	80.0%	**80.3%**	80.0%	79.3%	79.5%	69.5%
ringnorm	7400	2	66.1%	67.4%	50.5%	50.5%	**72.0%**	58.1%
faces	624	20	**77.2%**	**77.2%**	76.0%	76.0%	76.4%	48.6%
MNIST	70000	*free*	**96.8%** (18)	96.7% (18)	96.8% (2058)	96.8% (1523)	-	79.1% (382)
USPS	9298	*free*	96.7% (17)	96.6% (16)	**96.8%** (473)	96.8% (332)	-	84.9% (122)
iris	150	*free*	96.0% (9)	**96.7%** (9)	96.0% (18)	94.7% (15)	-	**96.7%** (8)
coil20	1440	*free*	84.9% (27)	85.2% (27)	**95.4%** (143)	94.7% (107)	-	87.8% (110)
waveform	5000	*free*	81.3% (6)	81.4% (6)	**86.3%** (298)	85.7% (193)	-	83.4% (5)
ringnorm	7400	*free*	60.8% (19)	61.9% (5)	**68.3%** (559)	67.9% (469)	-	58.6% (8)
faces	624	*free*	84.8% (32)	84.8% (32)	**99.4%** (92)	97.2% (76)	-	89.5% (61)

ters, and has the best accuracy. But the modularity and parabola objectives stay close to the true number of clusters, and they give the best NMI.

The way we adjust the number of clusters, by embedding the graph in a larger one, works best when we want to decrease the number of clusters. For some objectives, in particular normalized cut, we instead wish to increase the number of clusters. Other adjustments to the objective function might work better in that case, for example adding a term to directly reward the number of clusters. Such an adjustment will lead to another family of objective functions, perhaps with different resolution bias characteristics.

The parabola and modularity objectives have the same resolution bias, $\sqrt{nr/q}$. However, the behavior of the two objectives on the LFR benchmarks differ significantly.

This is in part due to the inability of LSO to find the optimal clustering for the parabola objective, but it seems that the objectives also differ in other ways. An avenue for future work is to improve the resolution bias model to show how these objectives differ.

In this paper we have only considered undirected graphs. Each of the objectives can be adapted to directed graphs by using a variation of the volume that is based on the indegree or outdegree of nodes, or on a combination of the two. In [19], the infomap objective is defined based on the outdegree and on edges leaving a cluster. It is not clear what the advantages are of directly using undirected graphs for clustering, or how the clustering of a graph should differ from the clustering of its transpose.

Our implementation of the methods described in this paper is available from `http://cs.ru.nl/~T.vanLaarhoven/clustering2012/`.

ACKNOWLEDGMENTS

This work has been partially funded by the Netherlands Organization for Scientific Research (NWO) within the NWO project 612.066.927.

[1] S. Fortunato, Physics Reports **486**, 75 (2010).
[2] S. Schaeffer, Computer Science Review **1**, 27 (2007).
[3] M. Brito, E. Chávez, A. Quiroz, and J. Yukich, Statist. Probab. Lett. **35**, 33 (1997).
[4] P. Franti, O. Virmajoki, and V. Hautamaki, IEEE Trans. Pattern Anal. Mach. Intell. **28**, 1875 (2006).
[5] J. Ruan, in *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, ICDM '09 (IEEE Computer Society, Washington, DC, USA, 2009) pp. 968–973.
[6] U. Luxburg, Statist. Comput. **17**, 395 (2007).
[7] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, *Knowledge and Data Engineering, IEEE Transactions on*, IEEE Transactions on Knowledge and Data Engineering **20**, 172 (2008).
[8] J. Shi and J. Malik, in *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, CVPR '97 (IEEE Computer Society, Washington, DC, USA, 1997) pp. 731–.
[9] M. E. J. Newman, Phys. Rev. E **74**, 036104 (2006), arXiv:physics/0605087.
[10] A. Arenas, A. Fernández, and S. Gómez, New J. Phys. **10**, 053039 (2008), arXiv:physics/0703218.
[11] R. Lambiotte, in *Proceedings of the 8th Int. Symp. on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks* (2010) pp. 546–553.
[12] A. Lancichinetti and S. Fortunato, Phys. Rev. E **84**, 066122 (2011).
[13] J. Reichardt and S. Bornholdt, Phys. Rev. Lett. **93**, 218701 (2004).
[14] V. A. Traag, P. Van Dooren, and Y. E. Nesterov, Phys. Rev. E **84**, 016114 (2011).
[15] S. Fortunato and M. Barthélemy, Proc. Natl. Acad. Sci. USA **104**, 36 (2007).
[16] B. H. Good, Y. A. de Montjoye, and A. Clauset, Phys. Rev. E **81**, 046106 (2010), arXiv:0910.0165.
[17] A. Lancichinetti and S. Fortunato, Phys. Rev. E **80**, 056117 (2009).
[18] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, J. Stat. Mech. Theory Exp. **2008**, P10008 (2008), arXiv:0803.0476.
[19] M. Rosvall and C. T. Bergstrom, Proc. Natl. Acad. Sci. USA **105**, 1118 (2008).
[20] M. Girvan and M. E. J. Newman, Proc. Natl. Acad. Sci. USA **99**, 7821 (2002).
[21] This expression for the infomap objective is based on the source code of the tools provided by Rosvall et.al., `http://www.tp.umu.se/~rosvall/code.html`.
[22] J. M. Kumpula, J. Saramaki, K. Kaski, and J. Kertesz, Eur. Phys. J. B **56**, 41 (2007).
[23] L. Angelini, D. Marinazzo, M. Pellicoro, and S. Stramaglia, J. Stat. Mech. Theory Exp. **2007**, L08001 (2007).
[24] A. Lancichinetti, S. Fortunato, and F. Radicchi, Phys. Rev. E **78**, 046110 (2008).
[25] A. Clauset, C. R. Shalizi, and M. E. J. Newman, SIAM Rev. **51**, 661 (2009).
[26] L. Danon, J. Duch, A. Arenas, and A. Díaz-Guilera, J. Stat. Mech. Theory Exp. **2005**, P09008 (2005).
[27] B. W. Kernighan and S. Lin, Bell Syst. Tech. J. **49**, 291 (1970).
[28] W. W. Zachary, Journal of Anthropological Research **33**, 452 (1977).
[29] V. Krebs, "New political patterns," Editorial (2004).
[30] L. Adamic and N. Glance, in *In LinkKDD 05: Proceedings of the 3rd international workshop on Link discovery* (2005) pp. 36–43.
[31] T. Bühler and M. Hein, in *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09 (ACM, New York, NY, USA, 2009) pp. 81–88.
[32] C. Granell, S. Gómez, and A. Arenas, Internat. J. Bifur. Chaos Appl. Sci. Engrg. **22** (2012).