Spectral methods for community detection and graph partitioning

M. E. J. Newman
Department of Physics and Center for the Study of Complex Systems, University of Michigan, Ann Arbor, Michigan 48109, USA

(Received 13 August 2013; published 30 October 2013)

We consider three distinct and well-studied problems concerning network structure: community detection by modularity maximization, community detection by statistical inference, and normalized-cut graph partitioning. Each of these problems can be tackled using spectral algorithms that make use of the eigenvectors of matrix representations of the network. We show that with certain choices of the free parameters appearing in these spectral algorithms the algorithms for all three problems are, in fact, identical, and hence that, at least within the spectral approximations used here, there is no difference between the modularity- and inference-based community detection methods, or between either and graph partitioning.

DOI: 10.1103/PhysRevE.88.042822 PACS number(s): 89.75.Hc, 02.50.Cw, 02.70.Hm

I. INTRODUCTION

Networked systems, such as social, biological, and technological networks, have been the subject of much recent research activity [1,2]. Along with many studies focusing on local properties of networks, such as clustering [3], degree distributions [4,5], and correlations [6,7], there have also been studies that examine large-scale properties such as path lengths [3], percolation [8,9], or hierarchy [10,11]. Among large-scale network properties, however, the one attracting by far the most attention has been community structure [12]. Many networks are found to possess communities or modules, groups of nodes within which connections are relatively dense and between which they are sparser. Communities are of fundamental interest in networked systems because of their functional implications—communities in a social network, for instance, may indicate factions, interest groups, or social divisions; communities in a metabolic network might correspond to functional units, cycles, or circuits that perform certain tasks.

The detection of communities in network data is also of interest from an algorithmic point of view. It is a remarkably challenging and subtle task for which a large number of approaches have been proposed. In this paper we examine two of the most widely used, the modularity maximization method [13] and the method of statistical inference by maximum likelihood [14,15]. In both of these approaches the community detection problem is mapped to one of optimizing a given objective function (either modularity or likelihood) over possible divisions of a network into groups, but the resulting optimization problem is, in general, a computationally hard one [16], so one typically employs one of a range of polynomial-time heuristics to find approximate optima, such as Markov chain Monte Carlo [14,17,18], extremal optimization [19], or greedy algorithms [20].

In this paper we study one of the most elegant classes of heuristics for network optimization problems, the spectral algorithms, inherently global methods based on the eigenvectors of matrix representations of network structure. We show that both the maximum modularity and maximum likelihood methods for community detection can be formulated as spectral algorithms that rely on the eigenvectors of the so-called normalized Laplacian matrix. We also describe a standard spectral algorithm for a third network problem, the well-known problem of normalized-cut graph partitioning. Our primary finding is that the spectral algorithms for all three of these problems are identical. At least within the spectral approach taken here, there is no difference between the detection of community structure using the methods of maximum modularity and maximum likelihood, or between either and normalized-cut graph partitioning. The latter equivalence is of particular interest because graph partitioning has been studied in depth for several decades and a broad range of results both applied and theoretical have been established, some of which can now be applied to the community detection problem as well.

The outline of this paper is as follows. In Secs. II, III, and IV we derive in turn our spectral algorithms for the maximum modularity, maximum likelihood, and normalized-cut partitioning problems, which, as we have said, turn out all to be the same. In Sec. V we give a selection of applications of the method to example networks, including both computer-generated benchmark networks and real-world networks, demonstrating its efficacy in community detection. In Sec. VI we give our conclusions.

II. MODULARITY MAXIMIZATION

In its most basic form, the problem of community detection in networks is one of dividing the vertices of a given network into nonoverlapping groups such that connections within groups are relatively dense while those between groups are sparse. As it stands, this definition is imprecise and leaves room for interpretation, and there have, as a result, been a large number of different methods proposed for solving the problem [12]. Of these probably the most widely used is the method of modularity maximization, in which the objective function known as modularity is optimized over possible divisions of the network [13]. The modularity for a given division of a network is defined to be the fraction of edges within groups minus the expected fraction of such edges in a randomized null model of the network. Various null models have been used, but the most common by far is the so-called configuration model [21,22], a random graph model in which the degrees of vertices are fixed to match those of the observed network but edges are in other respects placed at random. The expected number of edges falling between two vertices i and j in the configuration model is equal to $k_i k_j/2m$, where k_i is the degree of vertex i and m is the total number of edges in the observed network. The actual number of edges observed to fall between the same two vertices is equal to the element A_{ij} of the adjacency matrix **A**, so that the actual-minus-expected edge count for the vertex pair is $A_{ij} - k_i k_j/2m$. Giving integer labels to the groups in the proposed network division and denoting by g_i the label of the group to which vertex i belongs, the modularity Q is then equal to

$$Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\delta_{g_i g_j}, \qquad (1)$$

where δ_{ij} is the Kronecker delta. The leading constant $1/2m$ is purely conventional; it has no effect on the results presented here.

High values of the modularity correspond to good divisions of a network into groups and low values to bad, so the modularity maximization method finds good divisions by maximizing the modularity over divisions. Discussion of the background to the method, including past work and various pros and cons of the approach, can be found, for example, in Refs. [12] and [23]. The modularity can be calculated for divisions of a network into any number of groups, but for the purposes of this paper we will focus on the simplest case of division into just two groups, which is probably the most widely studied case.

Consider, then, a network of n vertices and m edges, which is to be divided into two groups of any size so as to maximize the modularity, Eq. (1). The modularity can be conveniently rewritten in terms of a set of n Ising spin variables s_i, one for each vertex, having values

$$s_i = \begin{cases} +1 & \text{if vertex } i \text{ belongs to group 1,} \\ -1 & \text{if vertex } i \text{ belongs to group 2.} \end{cases} \qquad (2)$$

Then $\delta_{g_i g_j} = \frac{1}{2}(s_i s_j + 1)$ and

$$Q = \frac{1}{4m}\sum_{ij}\left[A_{ij} - \frac{k_i k_j}{2m}\right](s_i s_j + 1). \qquad (3)$$

We define the quantity

$$B_{ij} = A_{ij} - \frac{k_i k_j}{2m} \qquad (4)$$

to be an element of a symmetric $n \times n$ matrix **B**, called the modularity matrix [24]. The modularity matrix has the crucial property that the sums of all its rows and columns are zero:

$$\sum_j B_{ij} = \sum_j A_{ij} - \frac{k_i}{2m}\sum_j k_j = k_i - \frac{k_i}{2m}2m = 0, \qquad (5)$$

where we have made use of $\sum_j A_{ij} = k_i$ and $\sum_j k_j = 2m$. Thus Eq. (3) can be written as

$$Q = \frac{1}{4m}\sum_{ij} B_{ij}(s_i s_j + 1) = \frac{1}{4m}\sum_{ij} B_{ij} s_i s_j, \qquad (6)$$

the second term in the brackets vanishing because of (5).

The matrix elements B_{ij} are fixed once the network is given, while the spins s_i represent the division of the network into groups. Our task is to maximize Q over the possible choices of the s_i—the values of s_i that achieve the maximum indicate the optimal division of the network into communities. This is still a difficult computational task, known to be NP complete in general [16], so the maximization is usually performed using approximate heuristics. In this paper we consider a spectral optimization strategy, similar in spirit to the spectral method proposed previously in Ref. [24], but differing from it in one crucial detail.

Maximization of (6) is difficult because the variables s_i are discrete valued. The problem can be made much easier by relaxing the discreteness and allowing the s_i to take any real values. This is an approximation—we will be solving a somewhat different problem from the one we really want to solve—but in practice it often gives good results. When we relax the s_i, however, we must still impose at least a minimal constraint on them to prevent them from becoming arbitrarily large, which would make Q large but only in a trivial way that yields no information about community structure. Most commonly one applies a constraint of the form $\sum_i s_i^2 = n$, which limits any individual s_i to the range $-\sqrt{n} \leqslant s_i \leqslant \sqrt{n}$ and fixes the mean-square value at 1.

In the language of spin models, this would be called a "spherical model" [25]. One can think of it in geometric terms, as shown in Fig. 1. If we consider the variables s_i to be the elements of an n-element vector **s**, then the allowed values $s_i = \pm 1$ in the original unrelaxed problem restrict the vector to the corners of an n-dimensional hypercube centered on the origin, while the relaxed values $\sum_i s_i^2 = n$ of the spherical model fall on the bounding hypersphere of radius $\sqrt{n}$ that touches the hypercube at each of its corners. Thus the relaxed values include all the allowed values in the original problem, but also include many other values as well.

While this spherical relaxation is the commonest approach to the spectral method, it is only one of an infinite number of possible relaxations, differing from one another in the details of the constraint used to prevent the values of the s_i from diverging. For instance, rather than relaxing onto the bounding hypersphere, we can relax onto any hyperellipsoid that touches the hypercube at all of its corners. In other words, we can choose a constraint of the form $\sum_i a_i s_i^2 = \sum_i a_i$ for any set of nonnegative constants a_i. It is trivially the case that this constraint is satisfied by the unrelaxed values $s_i = \pm 1$.

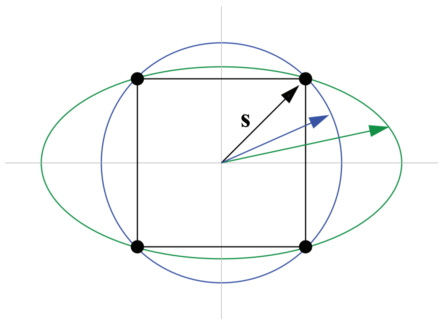

FIG. 1. (Color online) Geometric representation of the relaxation method employed here. The true optimization is over values of the vector **s** falling at the corners of a hypercube centered on the origin. The most common relaxation involves generalizing to values that lie anywhere on the bounding hypersphere that touches the cube at the corners (blue). In this paper, however, we generalize instead to a bounding hyperellipsoid, which also touches the cube at its corners (green).

(The possibility of alternative relaxations has been touched upon previously in the literature, for example in Ref. [26].) The standard hypersphere corresponds to $a_i = 1$ for all i, but in this paper we will find it convenient to make a different choice, leading to a spectral modularity optimization algorithm that is different in some important respects from previous algorithms.

Our choice is to set a_i equal to k_i, the observed degree of vertex i, so that our constraint takes the form

$$\sum_i k_i s_i^2 = 2m, \tag{7}$$

where m is again the number of edges in the network and we have made use of $\sum_i k_i = 2m$.

Although the original unrelaxed modularity maximization problem is a hard one to solve, this relaxed problem is much easier. It can be solved exactly by simple differentiation. Applying the constraint (7) with a Lagrange multiplier λ, the maximum is given by

$$\frac{\partial}{\partial s_l}\left[\sum_{ij} B_{ij} s_i s_j - \lambda \sum_i k_i s_i^2\right] = 0. \tag{8}$$

Performing the derivatives and rearranging, we find that

$$\sum_j B_{ij} s_j = \lambda k_i s_i, \tag{9}$$

or, in matrix notation,

$$\mathbf{Bs} = \lambda \mathbf{Ds}, \tag{10}$$

where $\mathbf{D}$ is the diagonal matrix with elements equal to the vertex degrees $D_{ii} = k_i$. In other words $\mathbf{s}$ is a solution of a generalized eigenvector equation, with λ being the eigenvalue.

To determine which eigenvector we should take, we multiply Eq. (9) by s_i and sum over i, making use of (6) and (7), to get an expression for the modularity:

$$Q = \frac{1}{4m}\sum_{ij} B_{ij} s_i s_j = \frac{\lambda}{4m}\sum_i k_i s_i^2 = \frac{\lambda}{2}. \tag{11}$$

To achieve the highest value of the modularity, therefore, we should choose λ to be the highest (most positive) eigenvalue of the generalized eigenvector equation (10).

Since all rows of the modularity matrix $\mathbf{B}$ sum to zero, it follows that Eq. (10) always has a solution $\mathbf{s} = (1,1,1,\ldots)$ with eigenvalue $\lambda = 0$. This solution, with all $s_i = +1$, corresponds to putting all vertices in group 1 and none in group 2, i.e., not dividing the network at all. This tells us that if $\lambda = 0$ is the highest eigenvalue then the best modularity is achieved by not dividing the network at all—the calculation is telling us that there is no good division of the network into groups, so we should leave it undivided. In our previous work we called such networks "indivisible."

If, however, there is even a single strictly positive eigenvalue, then there will exist some nontrivial solution vector $\mathbf{s}$ that achieves a higher modularity than the undivided network. Almost all networks do have such a strictly positive eigenvalue and we will assume this to be the case here—the only exceptions are unrealistically dense networks such as the complete graph.

The solution above can be simplified further. Using the definition (4) of the modularity matrix, we can rewrite Eq. (9) as

$$\sum_j A_{ij} s_j = k_i\left(\lambda s_i + \frac{1}{2m}\sum_j k_j s_j\right), \tag{12}$$

or in matrix notation as

$$\mathbf{As} = \mathbf{D}\left(\lambda \mathbf{s} + \frac{\mathbf{k}^T\mathbf{s}}{2m}\mathbf{1}\right), \tag{13}$$

where $\mathbf{k}$ is the vector with elements k_i and $\mathbf{1} = (1,1,1,\ldots)$. Noting that $\mathbf{A1} = \mathbf{D1} = \mathbf{k}$ and $\mathbf{k}^T\mathbf{1} = 2m$, we now multiply Eq. (13) throughout by $\mathbf{1}^T$ to get $\lambda \mathbf{k}^T\mathbf{s} = 0$, which implies either that the largest eigenvalue λ is zero or that

$$\mathbf{k}^T\mathbf{s} = 0. \tag{14}$$

Since we are assuming there exists a nontrivial eigenvalue $\lambda > 0$, we know that $\lambda \neq 0$ and hence (14) applies, which in turn means that Eq. (13) simplifies to

$$\mathbf{As} = \lambda \mathbf{Ds}. \tag{15}$$

Thus our solution vector $\mathbf{s}$ is also a solution of this generalized eigenvector equation, involving only the standard adjacency matrix. Again we should choose the largest allowed value of λ. Now, however, the most positive eigenvalue is disallowed—it is straightforward to see that the uniform vector $\mathbf{1}$ is an eigenvector and by the Perron-Frobenius theorem it must have the most positive eigenvalue, since it has all elements positive. But this choice of eigenvector fails to satisfy Eq. (14) and hence is forbidden, in which case the best we can do is choose the eigenvector corresponding to the second most positive eigenvalue [which can easily be shown to satisfy (14), as indeed do all the remaining eigenvectors]. This eigenvector is precisely equal to the leading eigenvector of Eq. (10), and hence either (10) or (15) will give us the solution we seek.

This is an exact solution of our relaxed modularity maximization problem. To get a solution to the original unrelaxed problem, in which s_i is constrained to take only the values ± 1, the normal approach is simply to round the s_i to the nearest allowed value ± 1. In practice, this just means that positive elements get rounded to $+1$ and negative elements to -1. Thus our final algorithm is a simple one: we calculate the eigenvector $\mathbf{s}$ of Eq. (15) corresponding to the second-highest eigenvalue, then divide the vertices of our network into two groups according to the signs of the elements of this vector. This is an approximation. It is not guaranteed to give an exact solution to the unrelaxed problem, but in many cases it does a good job, as we will later see.

As a practical matter, the solution of the generalized eigenvector equation (15) is most straightforwardly achieved by defining a rescaled vector $\mathbf{u} = \mathbf{D}^{1/2}\mathbf{s}$, where $\mathbf{D}^{1/2}$ is the diagonal matrix with diagonal elements equal to $\sqrt{k_i}$. Substituting into Eq. (15) and rearranging, we then find that

$$(\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2})\mathbf{u} = \lambda \mathbf{u}. \tag{16}$$

The matrix $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ is symmetric, and thus $\mathbf{u}$ is an ordinary eigenvector of a symmetric matrix, with elements having the same signs as those of $\mathbf{s}$, and with the same eigenvalue. The spectral algorithm is thus a simple matter of calculating the eigenvector for the second-highest eigenvalue of this symmetric matrix and then dividing the vertices according to the signs of its elements. For sparse networks this can be done efficiently using sparse matrix methods such as the Lanczos method.

The matrix

$$\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} \tag{17}$$

is sometimes called the normalized Laplacian of the network and we will use that terminology here. (The normalized Laplacian is sometimes defined as $\mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, where $\mathbf{I}$ is the identity, but the two matrices differ only in a trivial transformation of their eigenvalues and the eigenvectors are the same for both.) Connections between the normalized Laplacian and modularity maximization have been pointed out previously, for example by Delvenne *et al.* [27], who showed that maximum modularity divisions and Laplacian eigenvectors appear in different limits of a certain random walk process on networks.

III. STATISTICAL INFERENCE

We now turn to a second method for community detection in networks, the method of statistical inference using stochastic block models. This method has attracted attention in recent years for the excellent results it returns and because of the solid mathematical foundations on which it rests, which have allowed researchers to prove rigorously a range of results about its expected performance. Indeed the method is provably optimal for certain classes of networks, in the sense that no other method will classify more vertices into their correct groups on average [28,29].

The simplest form of the method is based on the standard stochastic block model [14], sometimes also called the planted partition model [30], a random graph model of a network containing community structure. The model does not itself constitute a method for community detection. Instead it provides a way of generating synthetic networks. To perform community detection, one fits the model to observed network data using a maximum likelihood method in much the same way as one might fit a straight line through a set of points to estimate their slope.

While formally elegant, however, this method has been found to work poorly in practice. The standard stochastic block model generates networks whose vertices have a Poisson degree distribution, quite unlike the degree distributions of most real-life networks, which means that the model is not, typically, a good fit to observed networks for any values of its parameters. The situation is akin to fitting a straight line through an inherently curved set of points—even the best fit will be a poor one because all fits are poor.

We can get around this problem by employing a slightly more sophisticated model, the degree-corrected block model [31], which incorporates additional parameters that allow the model to fit non-Poisson degree distributions, improving the fit to real-world data to the point where the model appears to give good community inference in practical situations.

The problem of fitting the degree-corrected block model to network data by likelihood maximization is, like the modularity maximization problem, a computationally difficult one in general, but it too can be tackled using approximate spectral methods, as we now describe. Indeed, as we will see, the spectral algorithm for the degree-corrected model is ultimately identical to the one we derived for the maximum modularity problem in the previous section.

In the degree-corrected block model n vertices are divided into groups and edges placed between them independently at random with probabilities that depend on the desired degrees of the vertices and on their group membership. Let g_i again denote the label of the group to which vertex i belongs. Then between each pair i,j of vertices we place a Poisson-distributed number of edges with mean equal to $k_i k_j \omega_{g_i g_j}$, where k_i is the desired degree of vertex i and ω_{rs} is a set of parameters whose values control the relative probabilities of connections within and between groups.

If A_{ij} is again an element of the adjacency matrix of the observed network, equal to the number of edges between vertices i and j, then the probability, or likelihood, that this network was generated by the degree-corrected stochastic block model is

$$L = \prod_{i<j} \frac{\left(k_i k_j \omega_{g_i g_j}\right)^{A_{ij}}}{A_{ij}!} \exp\left(-k_i k_j \omega_{g_i g_j}\right), \tag{18}$$

where the desired degrees k_i are equal to the actual degrees of the vertices in the observed network. Thus the likelihood of observing the network that we did in fact observe, assuming it was generated by this model, depends on the assignment of the vertices to the groups. For some assignments the network would be highly unlikely to have occurred; for others it is more likely. In the maximum likelihood approach, we assume that the best assignment of vertices to groups is the one that maximizes the likelihood. This again turns the community detection problem into an optimization problem which, although hard to solve exactly, often has good approximate solutions that can be found with relative ease.

Typically, in fact, we maximize not the likelihood itself but its logarithm $\mathscr{L}$, which has its maximum in the same place:

$$\mathscr{L} = \frac{1}{2}\sum_{ij}\left[A_{ij}\ln\omega_{g_i g_j} - k_i k_j \omega_{g_i g_j}\right], \tag{19}$$

where we have switched to a sum over all i,j and compensated with the leading factor of $\frac{1}{2}$, and we have assumed that the observed number of edges A_{ij} between any pair of vertices is either one or zero so that $A_{ij}! = 1$ for all i,j.

As in Sec. II, we will concentrate on the simplest case of a network with just two groups, and in addition we will assume (as most other authors also have) that there are just two different values for the model parameters: ω_{in} for pairs of vertices that fall in the same group and ω_{out} for pairs in different groups, with $\omega_{\text{in}} > \omega_{\text{out}}$ for traditional community structure (so-called assortative structure). Introducing indicator variables $s_i = \pm 1$ to denote group membership as we did in Sec. II, we note that

$$\omega_{g_i g_j} = \frac{1}{2}[(\omega_{\text{in}} + \omega_{\text{out}}) + s_i s_j(\omega_{\text{in}} - \omega_{\text{out}})], \tag{20}$$

$$\ln\omega_{g_i g_j} = \frac{1}{2}\left[\ln(\omega_{\text{in}}\omega_{\text{out}}) + s_i s_j \ln\frac{\omega_{\text{in}}}{\omega_{\text{out}}}\right]. \tag{21}$$

Substituting these expressions into Eq. (19), we then find that

$$\mathscr{L} = \sum_{ij}(A_{ij} - \nu k_i k_j)s_i s_j, \tag{22}$$

where ν is a positive constant given by

$$\nu = \frac{\omega_{\text{in}} - \omega_{\text{out}}}{\ln \omega_{\text{in}} - \ln \omega_{\text{out}}}, \tag{23}$$

and we have dropped unimportant additive and multiplicative constants, which have no effect on the position of the likelihood maximum.

Our goal is now to maximize Eq. (22) with respect to the variables s_i, but there is a problem: in most cases we don't know the values of the parameters ω_{in} and ω_{out} and hence we don't know ν either. Let us, however, suppose for the moment that we do know ν and see where it leads us. Equation (22) is closely similar in form to the modularity of Eq. (6), the only differences being a trivial leading constant, and the substitution of ν in the place of $1/2m$ when compared to the modularity matrix of Eq. (4). The similarities are sufficiently strong that we can use the same spectral approach to maximize (22) as we did for the modularity, and it turns out to give the same answer. We relax the variables s_i, allowing them to take any real values subject only to the elliptical constraint of Eq. (7), then introduce a Lagrange multiplier λ and differentiate to get

$$\sum_j (A_{ij} - \nu k_i k_j) s_j = \lambda k_i s_i, \tag{24}$$

or, in matrix notation,

$$(\mathbf{A} - \nu \mathbf{k}\mathbf{k}^{\mathbf{T}})\mathbf{s} = \lambda \mathbf{D}\mathbf{s}, \tag{25}$$

where $\mathbf{D}$ is the diagonal matrix of degrees as previously. Thus the solution to our relaxed maximization problem is an eigenvector of the matrix $\mathbf{A} - \nu \mathbf{k}\mathbf{k}^T$ and, by the same argument as before, we should choose the leading eigenvector.

Multiplying (25) on the left by $\mathbf{1}^T$ and making use of $\mathbf{A}\mathbf{1} = \mathbf{D}\mathbf{1} = \mathbf{k}$ and $\mathbf{k}^T\mathbf{1} = 2m$, we get

$$\mathbf{k}^T\mathbf{s} - 2m\nu\mathbf{k}^T\mathbf{s} = \lambda\mathbf{k}^T\mathbf{s}, \tag{26}$$

which implies either that $\mathbf{k}^T\mathbf{s} = 0$ or that $\lambda = 1 - 2m\nu$. We are not at liberty, however, to choose any of the quantities λ, ν, or m, and hence cannot in general satisfy the latter condition. Hence we must have $\mathbf{k}^T\mathbf{s} = 0$ and Eq. (25) simplifies to

$$\mathbf{A}\mathbf{s} = \lambda\mathbf{D}\mathbf{s}, \tag{27}$$

which is identical to Eq. (15) for the maximum modularity problem. Note that the constant ν has dropped out of the equation, so the fact that its value is unknown is, after all, not a problem.

From this point onward, the argument is the same as for the maximum modularity problem and leads to the same result, that the optimal division of the network is given by the signs of the elements of the eigenvector of the normalized Laplacian matrix of Eq. (17) corresponding to the second most positive eigenvalue.

Thus, within the spectral approximation used here, the maximum modularity and maximum likelihood methods for community detection are functionally identical and give identical results.

IV. GRAPH PARTITIONING

We now turn to the third of the three problems mentioned in the introduction, the problem of normalized-cut graph partitioning, which, when tackled using the spectral method, we will show to be identical to the community detection problems of the previous sections. In a previous paper [32] we noted a mapping between maximum-likelihood community detection and the slightly different problem of minimum-cut partitioning, although that mapping requires an extra computational step not required by the mapping presented here. Connections between graph partitioning and modularity maximization have also been noted previously [33,34], although only for modified forms of the modularity and not for the standard modularity studied in this paper.

Traditional graph partitioning is the problem of dividing a network into a given number of parts of given sizes such that the cut size R—the number of edges running between parts—is minimized. In the most commonly studied case the parts are taken to be of equal size. In many situations, however, one is willing to tolerate a little inequality of sizes if it allows for a better cut. Focusing once more on the case of division into two parts, a standard way to achieve this kind of tolerance is to minimize not the cut size but the *ratio cut* R/n_1n_2, where R is again the cut size and n_1 and n_2 are the sizes of the two groups. The minimization is now performed with no constraint on the group sizes, but since n_1n_2 is maximized when $n_1 = n_2 = \frac{1}{2}n$, the minimization still favors equally sized groups, but it balances this favoritism against a desire for small cut size, and the compromise seems to work well in many practical situations.

Another variant on the same idea, which is particularly effective for networks that have broad degree distributions, as do many real-world networks, is minimization of the normalized cut $R/\kappa_1\kappa_2$, where κ_1 and κ_2 are the sums of the degrees of the vertices in the two groups. This choice favors divisions of the network where the groups contain equal numbers of edges, rather than equal numbers of vertices, which is desirable in certain applications. It is on this normalized-cut partitioning problem that we focus in this section.

The normalized-cut problem, like the other problems we have studied, is hard to solve exactly, but good approximate solutions can be found using spectral methods. The spectral approach given here is a standard one and is not new to this paper—see, for example, Zhang and Jordan [35]. As before, we define index variables s_i to denote the group membership of each vertex, but rather than the ± 1 values we used previously, we define

$$s_i = \begin{cases} \sqrt{\kappa_2/\kappa_1} & \text{if} \quad i \text{ is in group } 1, \\ -\sqrt{\kappa_1/\kappa_2} & \text{if} \quad i \text{ is in group } 2, \end{cases} \tag{28}$$

where κ_1 and κ_2 are again the sums of the degrees of the vertices in each group. Note that this means that the values denoting the two groups change when the composition of the groups changes.

With this choice for the s_i, and using our previous notations $\mathbf{k}$ and $\mathbf{D}$ for the vector and diagonal matrix of degrees respectively, we have

$$\begin{aligned} \mathbf{k}^T\mathbf{s} &= \sum_i k_i s_i = \sqrt{\frac{\kappa_2}{\kappa_1}} \sum_{i\in 1} k_i - \sqrt{\frac{\kappa_1}{\kappa_2}} \sum_{i \in 2} k_i \\ &= \sqrt{\kappa_2\kappa_1} - \sqrt{\kappa_1\kappa_2} = 0, \end{aligned} \tag{29}$$

and

$$\mathbf{s}^T\mathbf{D}\mathbf{s} = \sum_i k_i s_i^2 = \frac{\kappa_2}{\kappa_1}\sum_{i\in 1} k_i + \frac{\kappa_1}{\kappa_2}\sum_{i\in 2} k_i$$
$$= \kappa_2 + \kappa_1 = 2m, \tag{30}$$

where m is the number of edges in the network as before and the notation $i \in 1$ indicates that vertex i is a member of group 1.

Note also that

$$s_i + \sqrt{\frac{\kappa_1}{\kappa_2}} = \frac{2m}{\sqrt{\kappa_1\kappa_2}}\,\delta_{g_i,1}, \tag{31}$$

meaning this quantity is nonzero only if i belongs to group 1. Similarly

$$s_i - \sqrt{\frac{\kappa_2}{\kappa_1}} = -\frac{2m}{\sqrt{\kappa_1\kappa_2}}\,\delta_{g_i,2}. \tag{32}$$

Using these results, we have

$$\sum_{ij} A_{ij}\left(s_i + \sqrt{\frac{\kappa_1}{\kappa_2}}\right)\left(s_j - \sqrt{\frac{\kappa_2}{\kappa_1}}\right)$$
$$= -\frac{(2m)^2}{\kappa_1\kappa_2}\sum_{ij} A_{ij}\,\delta_{g_i,1}\delta_{g_j,2} = -\frac{(2m)^2}{\kappa_1\kappa_2}R, \tag{33}$$

where, as before, R is the cut size between the two groups. But the quantity on the left can also be written in matrix form as

$$\left(\mathbf{s} + \sqrt{\frac{\kappa_1}{\kappa_2}}\mathbf{1}\right)^T \mathbf{A}\left(\mathbf{s} - \sqrt{\frac{\kappa_2}{\kappa_1}}\mathbf{1}\right) = \mathbf{s}^T\mathbf{A}\mathbf{s} - 2m, \tag{34}$$

where we have made use of $\mathbf{k} = \mathbf{A}\mathbf{1}$, $\mathbf{1}^T\mathbf{A}\mathbf{1} = 2m$, and Eq. (29).

Combining Eqs. (33) and (34), we now have a matrix expression for the normalized cut:

$$\frac{R}{\kappa_1\kappa_2} = \frac{2m - \mathbf{s}^T\mathbf{A}\mathbf{s}}{(2m)^2}. \tag{35}$$

Thus minimizing the normalized cut is equivalent to maximizing $\mathbf{s}^T\mathbf{A}\mathbf{s}$ over choices of s_i satisfying (28).

This hard optimization problem is once more made easier by relaxation. We relax the requirement that the s_i take the values in Eq. (28), allowing them to take any real values subject only to the constraints (29) and (30). The relaxed problem can then be solved straightforwardly by introducing Lagrange multipliers λ,μ for the two constraints and differentiating, which gives

$$\mathbf{A}\mathbf{s} = \lambda\mathbf{D}\mathbf{s} + \mu\mathbf{k}. \tag{36}$$

Multiplying on the left by $\mathbf{1}^T$ and making use of $\mathbf{1}^T\mathbf{A} = \mathbf{1}^T\mathbf{D} = \mathbf{k}^T$ gives

$$\mathbf{k}^T\mathbf{s} = \lambda\mathbf{k}^T\mathbf{s} + 2m\mu, \tag{37}$$

which implies that $\mu = 0$ because of Eq. (29), and hence we find once again that $\mathbf{s}$ is a solution of the generalized eigenvector equation

$$\mathbf{A}\mathbf{s} = \lambda\mathbf{D}\mathbf{s}. \tag{38}$$

Using Eqs. (30) and (35), the optimal value of the normalized cut is then

$$\frac{R}{\kappa_1\kappa_2} = \frac{2m - \lambda\mathbf{s}^T\mathbf{D}\mathbf{s}}{(2m)^2} = \frac{1-\lambda}{2m}, \tag{39}$$

which is minimized by choosing λ as large as possible. The leading eigenvalue, however, is ruled out, since its eigenvector $\mathbf{1}$ fails to satisfy Eq. (29), so once again our solution of the relaxed problem is given by the eigenvector corresponding to the second largest eigenvalue of Eq. (38) [which does satisfy (29), as do all the other eigenvectors].

Reversing the relaxation process is a little more complicated in this case than in the previous cases we have studied, because the discrete values of s_i that we are rounding to, given by Eq. (28), are not constant, but depend on the composition of the groups themselves. In principle, the most correct way to do it is to go through every possible division of the elements of the leading eigenvector, of which there are $n+1$, and find the one that gives the smallest value of the normalized cut. In practice, however, since we are looking for solutions with roughly equal group sizes, the values of κ_1 and κ_2 are also roughly equal, meaning that the discrete values of s_i are approximately ± 1, and we can usually get good solutions by rounding to these values, which is equivalent to dividing vertices according to the signs of the vector elements. As we show in the next section, the divisions returned by the method are typically insensitive to the precise threshold value at which we divide the vector elements, so the results do not depend strongly on the rounding strategy chosen.

With this choice, which is the most common one, the algorithm becomes the same as the algorithms we have given

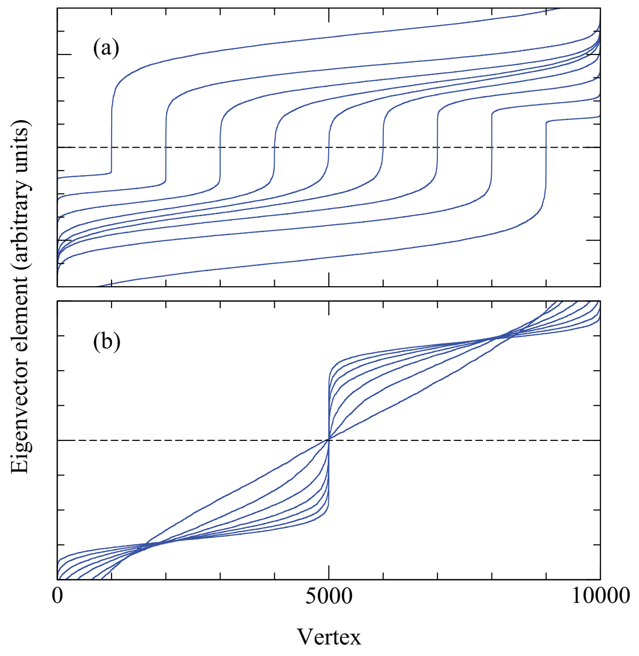

FIG. 2. (Color online) Results from the application of the algorithm described here to networks generated using the stochastic block model with two communities. (a) Each curve shows the values, plotted in increasing order, of the elements of the second eigenvector of Eq. (15) for single networks with $n = 10\,000$ vertices and within-group and between-group edge probabilities $\omega_{\text{in}} = 75/n$ and $\omega_{\text{out}} = 25/n$ respectively. The curves are, from left to right, for networks in which group 1 has size 1000, 2000, 3000, ... ,9000. (b) Similar curves for networks of 10 000 vertices and equally sized groups, but with varying edge probabilities. The edge probabilities are given by $n\omega_{\text{in}} = 60, 65, 70, 75, 80, 85,$ and 90 and $n\omega_{\text{out}} = 100 - n\omega_{\text{in}}$.

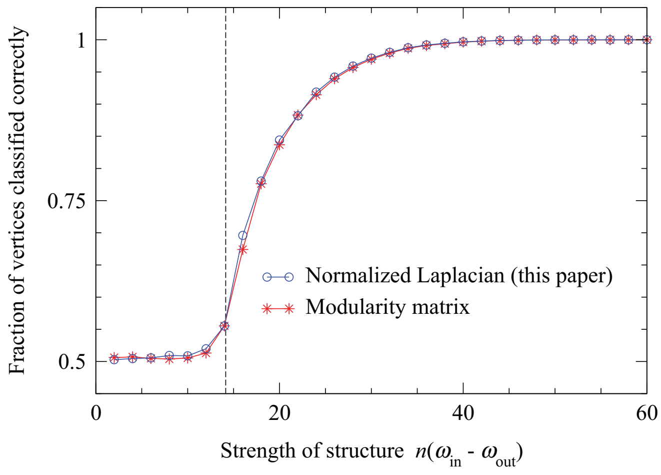

FIG. 3. (Color online) The fraction of vertices classified into the correct groups by the algorithm described in this paper (blue circles) and by a standard spectral algorithm based on the leading eigenvector of the modularity matrix (red stars) for networks of $n = 10\,000$ vertices generated from a stochastic block model with two equally sized groups and mean degree 50. The vertical dashed line represents the position of the detectability threshold below which all community detection algorithms must fail this test [28,37–40].

for community detection, either by modularity maximization or the method of maximum likelihood.

V. EXAMPLES

We have shown that three different problems—two-way community detection by maximum modularity and maximum likelihood, and normalized-cut bisection of a graph—can all be solved using the same spectral algorithm. We compute the eigenvector of the normalized Laplacian matrix corresponding to the second-largest eigenvalue and divide vertices according to the signs of the vector elements. In this section we give some example applications of the algorithm to both computer-generated and real-world networks.

Figure 2 shows results from the application of the method to networks generated using the stochastic block model of Sec. III, which in addition to its use in community inference is also widely used as a benchmark test for community detection methods [30,36]. Figure 2(a) shows a series of curves representing the elements of the second eigenvector of Eq. (15) in increasing order for single networks with two communities of varying sizes. The horizontal dashed line indicates the point at which the values of the elements pass zero—vertices on one side of this line are placed in the first group and vertices on the other side are placed in the second. Each curve passes briskly through zero at a point close to the sizes of the two groups planted in the network—the size of the first group in this test was 1000, 2000, 3000, and so on for each successive curve. This shows that the algorithm is capable of the accurate unsupervised detection of groups of a wide range of different sizes. Moreover detection is robust against fluctuations—because the line is close to vertical as it passes zero, the division of the network is insensitive to changes in the cut point. If the dashed line were moved up or down, even by quite a large amount, very few vertices would change group membership. This observation provides some justification for our contention at the end of Sec. IV that the exact choice of the cut point is unimportant.

Figure 2(b) shows similar curves for stochastic block model networks with two equally sized groups (which is the most

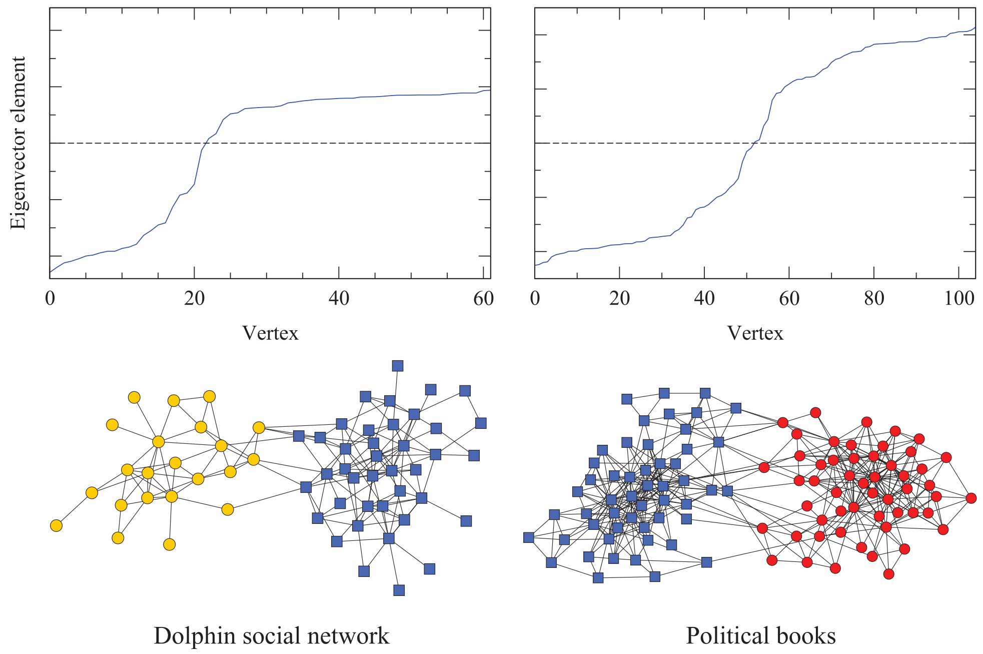

FIG. 4. (Color online) Left: results of applying the algorithm to a network of frequent association among a group of bottlenose dolphins studied by Lusseau *et al.* [43], which is believed to divide into two clear communities. The top panel shows the values of the eigenvector elements in increasing order. The bottom panel shows the resulting division of the network. Right: equivalent plots for a copurchasing network of books about US politics in which vertices represent books and edges connect books frequently purchased by the same purchaser. This network is thought to split strongly along lines of political ideology.

challenging case) but varying strength of community structure. When the structure is strongest the curves show a pronounced step at the halfway point, indicating robust detection of the equally sized communities, but the step becomes progressively smaller as the planted structure gets weaker, and eventually disappears completely, so that the curve becomes featureless. The point at which the step disappears coincides with the "detectability threshold" below which it is believed that all algorithms (including this one) must fail to detect community structure [28,37–40].

Figure 3 further quantifies the algorithm's success at detecting community structure in block model networks. The figure shows the fraction of vertices classified into the correct groups for the same situation as in Fig. 2(b)—block model networks with $n = 10\,000$, two equally sized groups, and varying strength of community structure. For most of the parameter range spanned by the figure the algorithm does a good job of putting vertices in the right groups. The vertical dashed line in the figure shows the position of the detectability threshold, below which we expect the algorithm to return results no better than a random guess (which means 50% of vertices classified correctly). As we can see, the algorithm does better than random all the way down to the transition point (if only by a small margin in the region close to the threshold), which agrees with previous theoretical results finding that other spectral algorithms do the same [39], at least on networks with sufficiently high average degree [41]. Also shown in Fig. 3 are results for tests on the same networks of the standard spectral community detection method of Ref. [24], in which one examines the leading eigenvector of the modularity matrix. As the figure shows, the performance of the two algorithms, at least in this test, is essentially identical, although some previous studies have found small differences in performance between algorithms [42].

Figure 4 shows example applications to two well-studied real-world networks, the dolphin social network of Lusseau *et al.* [43] and the political book network of Krebs [24], both of which are believed to break clearly into two communities. The top two panels in the figure show the equivalent of the curves in Fig. 2—values of the elements of the second eigenvector in increasing order. Each shows a clear step where it crosses the zero line (dashed lines in the plots) and the groups generated by dividing the vertices at this point are shown in the lower panels. In both cases the groups correspond closely to the accepted ground truth for these networks.

A further interesting example is given in Fig. 5, which shows an application of the method to a network of US political weblogs compiled by Adamic and Glance [44]. Again this network is believed to divide strongly into two communities (along lines of political outlook), and the algorithm finds the accepted division to a good approximation. In this case, however, the division was found by examining the *third* eigenvector of the normalized Laplacian, not the second, as the developments of this paper would suggest. An examination of the second eigenvector reveals that it is entirely uncorrelated with the community structure in the network, instead being strongly localized around a few of the highest-degree vertices in the network—very large vector elements for these few hub vertices and small and apparently random elements for all other vertices. It is known that very high-degree vertices in networks can give rise to high-lying, localized eigenvectors [45], by mechanisms quite different from those that produce the eigenvectors containing community structure, and the two types of high-lying eigenvectors may compete to be the highest in the overall spectrum. The network of political blogs has a particularly broad distribution of vertex degrees, with some degrees far above the network mean, which in this case is apparently enough to create an additional eigenvector with eigenvalue above that of the vector containing the community structure. Nonetheless, the community structure is still there, clearly present in the third eigenvector. In practice, this means that application of the algorithm may not be quite as simple as our derivations suggest: it may require some finesse to extract useful community structure, particularly in the case of networks with very high-degree hubs. Anecdotally, based on our experiments, we believe that the replacement of the second eigenvector with a localized vector related to network hubs may occur more frequently in the algorithm described in

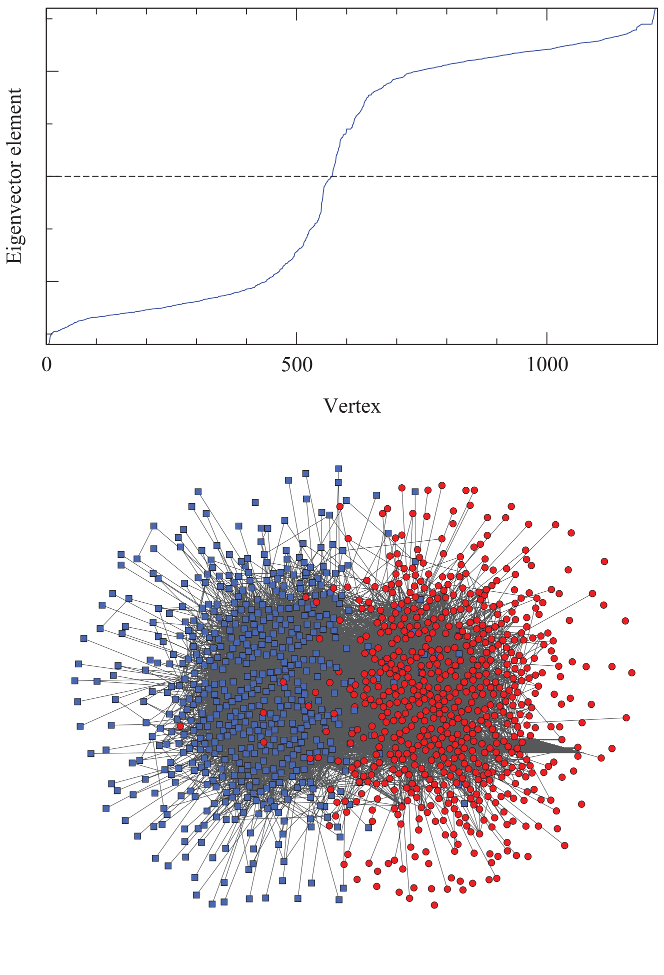

FIG. 5. (Color online) Eigenvector elements and community division found by application of the algorithm described in this paper to a network of weblogs about US politics, and the web hyperlinks between them, compiled by Adamic and Glance [44]. Again the algorithm finds clear two-way community structure that corresponds closely to the acknowledged division of the network, but in this case the community structure was obtained from the third, not the second, eigenvector of the normalized Laplacian.

this paper than it does in more conventional algorithms based on the eigenvectors of the modularity matrix [24], but this at present is merely conjecture.

VI. CONCLUSIONS

In this paper we have given spectral algorithms for the solution of three distinct network problems: community detection by modularity maximization, community detection by likelihood maximization using the degree-corrected block model, and normalized-cut graph partitioning. As we have shown, the algorithms for all three of these problems turn out to be the same, so that there is no difference, at least within the spectral formulation we use, between these three problems, although the algorithm described is different from standard spectral algorithms for modularity maximization that have appeared in the previous literature. We have given results from applications of the algorithm to a range of computer-generated and real-world networks, and it appears to perform well in practice.

One clear possibility for extension of the calculations outlined here is their generalization to the case of networks containing more than two groups or communities. The fundamental techniques needed for such a generalization are known [35,46]—one replaces the index variables s_i of Eq. (2) with vectors pointing to the corners of a (possibly irregular) simplex and the objective function (modularity, likelihood, or normalized cut) with the trace of a quadratic form involving the appropriate matrix. At present, however, a good all-purpose approach for community detection using such methods has yet to be found, and so the generalization to more than two communities must be considered an open problem.

ACKNOWLEDGMENTS

The author thanks Cris Moore, Partha Niyogi, and Tiago Peixoto for useful comments. This work was funded in part by the National Science Foundation under Grant No. DMS–1107796 and by the Air Force Office of Scientific Research (AFOSR) and the Defense Advanced Research Projects Agency (DARPA) under Grant No. FA9550–12–1–0432.

[1] M. E. J. Newman, SIAM Rev. **45**, 167 (2003).
[2] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang, Phys. Rep. **424**, 175 (2006).
[3] D. J. Watts and S. H. Strogatz, Nature (London) **393**, 440 (1998).
[4] A.-L. Barabási and R. Albert, Science **286**, 509 (1999).
[5] L. A. N. Amaral, A. Scala, M. Barthélemy, and H. E. Stanley, Proc. Natl. Acad. Sci. USA **97**, 11149 (2000).
[6] R. Pastor-Satorras, A. Vázquez, and A. Vespignani, Phys. Rev. Lett. **87**, 258701 (2001).
[7] M. E. J. Newman, Phys. Rev. Lett. **89**, 208701 (2002).
[8] R. Cohen, K. Erez, D. ben-Avraham, and S. Havlin, Phys. Rev. Lett. **85**, 4626 (2000).
[9] D. S. Callaway, M. E. J. Newman, S. H. Strogatz, and D. J. Watts, Phys. Rev. Lett. **85**, 5468 (2000).
[10] E. Ravasz and A.-L. Barabási, Phys. Rev. E **67**, 026112 (2003).
[11] A. Clauset, C. Moore, and M. E. J. Newman, Nature (London) **453**, 98 (2008).
[12] S. Fortunato, Phys. Rep. **486**, 75 (2010).
[13] M. E. J. Newman, Phys. Rev. E **69**, 066133 (2004).
[14] T. A. B. Snijders and K. Nowicki, J. Classification **14**, 75 (1997).
[15] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, J. Mach. Learn. Res. **9**, 1981 (2008).
[16] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner, in *Proceedings of the 33rd International Workshop on Graph-Theoretic Concepts in Computer Science*, No. 4769 in Lecture Notes in Computer Science (Springer, Berlin, 2007).
[17] R. Guimerà, M. Sales-Pardo, and L. A. N. Amaral, Phys. Rev. E **70**, 025101 (2004).
[18] A. Medus, G. Acuña, and C. O. Dorso, Physica A **358**, 593 (2005).
[19] J. Duch and A. Arenas, Phys. Rev. E **72**, 027104 (2005).
[20] A. Clauset, M. E. J. Newman, and C. Moore, Phys. Rev. E **70**, 066111 (2004).
[21] M. Molloy and B. Reed, Random Struct. Algorithm. **6**, 161 (1995).
[22] M. E. J. Newman, S. H. Strogatz, and D. J. Watts, Phys. Rev. E **64**, 026118 (2001).
[23] M. E. J. Newman, Nat. Phys. **8**, 25 (2012).
[24] M. E. J. Newman, Proc. Natl. Acad. Sci. USA **103**, 8577 (2006).
[25] T. H. Berlin and M. Kac, Phys. Rev. **86**, 821 (1952).
[26] R. Lambiotte, *Proceedings of the 8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks* (Institute of Electrical and Electronics Engineers, New York, 2010).
[27] J.-C. Delvenne, S. N. Yalirakia, and M. Barahona, Proc. Natl. Acad. Sci. USA **107**, 12755 (2010).
[28] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová, Phys. Rev. Lett. **107**, 065701 (2011).
[29] A. A. Amini, A. Chen, P. J. Bickel, and E. Levina, arXiv:1207.2340.
[30] A. Condon and R. M. Karp, Random Struct. Algorithm **18**, 116 (2001).
[31] B. Karrer and M. E. J. Newman, Phys. Rev. E **83**, 016107 (2011).
[32] M. E. J. Newman, Europhys. Lett. **103**, 28003 (2013).
[33] L. Yu and C. Ding, in *Proceedings of the Eighth Workshop on Mining and Learning with Graphs* (Association of Computing Machinery, New York, 2010), pp. 34–36.
[34] M. Bolla, Phys. Rev. E **84**, 016108 (2011).
[35] Z. Zhang and M. I. Jordan, Stat. Sci. **23**, 383 (2008).
[36] M. Girvan and M. E. J. Newman, Proc. Natl. Acad. Sci. USA **99**, 7821 (2002).
[37] J. Reichardt and M. Leone, Phys. Rev. Lett. **101**, 078701 (2008).
[38] D. Hu, P. Ronhovde, and Z. Nussinov, Phil. Mag. **92**, 406 (2012).
[39] R. R. Nadakuditi and M. E. J. Newman, Phys. Rev. Lett. **108**, 188701 (2012).
[40] E. Mossel, J. Neeman, and A. Sly, arXiv:1202.1499.

[41] F. Krzakala, C. Moore, E. Mossel, J. Neeman, A. Sly, L. Zdeborová, and P. Zhang, arXiv:1306.5550.

[42] T. P. Peixoto, Phys. Rev. Lett. **111**, 098701 (2013).

[43] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, Behav. Ecol. Sociobiol. **54**, 396 (2003).

[44] L. A. Adamic and N. Glance, in *Proceedings of the WWW-2005 Workshop on the Weblogging Ecosystem* (2005).

[45] R. R. Nadakuditi and M. E. J. Newman, Phys. Rev. E **87**, 012803 (2013).

[46] M. A. Riolo and M. E. J. Newman, arXiv:1209.5969.