

Modularity revisited: A novel dynamics-based concept for decomposing complex networks

M. Sarich^{1,2} N. Djurdjevac¹ S. Bruckner¹
T.O.F. Conrad¹ Ch. Schuette¹

June 7, 2012

Abstract

Finding modules (or clusters) in large, complex networks is a challenging task, in particular if one is not interested in a full decomposition of the whole network into modules. We consider modular networks that also contain nodes that do not belong to one of modules but to several or to none at all. A new method for analyzing such networks is presented. It is based on spectral analysis of random walks on modular networks. In contrast to other spectral clustering approaches, we use different transition rules of the random walk. This leads to much more prominent gaps in the spectrum of the adapted random walk and allows for easy identification of the network's modular structure, and also identifying the nodes belonging to these modules. We also give a characterization of that set of nodes that do not belong to any module, which we call transition region. Finally, by analyzing the transition region, we describe an algorithm that identifies so called hub-nodes inside the transition region that are important connections between modules or between a module and the rest of the network. The resulting algorithms scale linearly with network size (if the network connectivity is sparse) and thus can also be applied to very large networks.

1 Introduction

Describing complex systems as abstract networks is a powerful tool. The description by means of a network reduces the system under consideration to the information about its constituents or elementary parts, represented by the nodes of the network, and the interaction between these parts, represented by edges between the nodes. On the one hand this description is as sparse as it can get, that is, it ignores all available additional information or properties of the system. On the other hand, it is so abstract that it can be used all over the sciences, the range spanning from protein interaction networks via traffic or computer networks to social networks (Albert and Barabasi, 2002; Newman, 2003; Newman et al., 2006).

In many applications these networks can be huge such that even as abstractions they remain highly complex. Approaches to reducing the complexity of

¹BioComputing Group, Institute of Mathematics, Freie Universität Berlin

²Corresponding author. Email: sarich@math.fu-berlin.de

networks and to understanding their structure include characterization of networks in terms of simple statistics such as degree distributions (Jeong et al., 2000), or decompositions of networks into loosely coupled sub-networks like in clustering and module finding (Newman and Girvan, 2004). In the latter case, networks are coarse-grained into modules where nodes belonging to one module are highly interconnected, but have relatively few connections to nodes in other modules. In the following, we will use the term “modules” instead of “clusters” because we want to avoid confusion of the approach to be presented with standard clustering approaches.

In recent years, an abundance of algorithms have been proposed to detect modules in networks, see for example Porter et al. (2009); Santo (2010); Nascimento and De Carvalho (2011) for reviews. These algorithms span a variety of categories. Many are designed around optimizing the maximal modularity measure, first proposed by Newman Newman and Girvan (2004), and its variants (see, e.g. Reichardt and Bornholdt (2006); Newman et al. (2006)). Others have been constructed based on various topological structures, such as edge betweenness Girvan and Newman (2002). Another family of algorithms are closer in spirit to what we propose, and are designed around the dynamical properties of the network. This family includes algorithms such as Markov Clustering (MCL Van Dongen (2000)), based on simulation of (stochastic) flow in graphs, various spectral clustering methods that utilize the eigenvectors of the graph Laplacian or transition matrix, (see for example Luxburg (2007) for an overview), and many others. Furthermore, there is the vigorously researched area of Machine Learning and statistical inference algorithms, see (Santo, 2010) for an extensive review.

Almost all of these approaches consider partitioning the network *completely* into modules, that is, *every* node is assigned to *exactly one* module. During the last five years there has been growing interest in developing alternative approaches for the case of module identification where *overlaps* between modules are allowed. Algorithms such as k-clique percolation Palla et al. (2005), or the extension of modularity score for overlaps Nicosia et al. (2008) are notable examples of this trend.

In contrast to such standard module-finding approaches, we will *not* consider complete or overlapping partitions of the network. Instead we suggest to consider modules as groups of densely connected nodes that do *not* partition the network completely and do *not* overlap so that there remain nodes that are not assigned to any module at all. We call the set of these nodes the *transition* or *interconnection region*.

The distinction between complete and non-complete partition is illustrated in Fig. 1 for an example network with modular structures connected by nodes that do not seem to belong to a module. While the MCL algorithm (as an example for the standard approaches) fully partitions the network into modules and assigns every node to exactly one of these modules (Fig. 1(a)), the new approach to be presented herein finds modular structures and an additional extended interconnection region which connects the modules (Fig. 1(b)). In order to understand its structure in more detail, we will introduce an algorithm which allows for identifying the nodes in the interconnection region that are most important for the connection between modules: the hubs of the network (see section 3.4).

The new approach to be presented exploits recent results on random walks

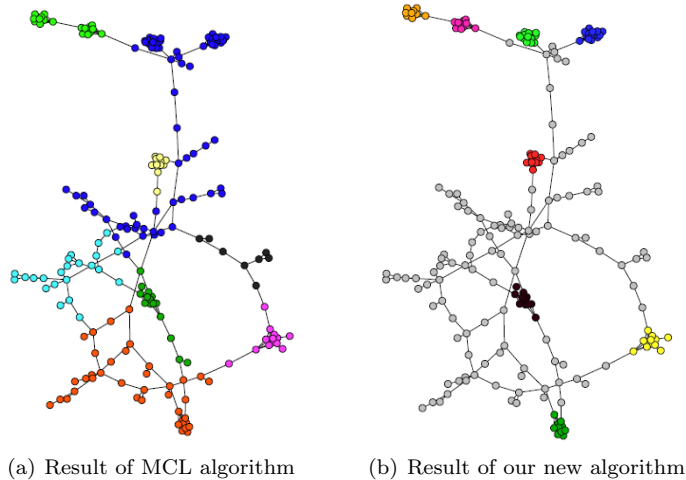


Figure 1: Simple example network. Colors indicate modules found by (a) the MCL algorithm (complete partition, left) and (b) our new approach (non-complete partition, right); nodes belonging to the interconnection regions are shown in gray.

on modular networks. It considers modules as metastable sets of the random walk and identifies these metastable sets via its spectral properties. These ideas are not new: In the last decades methods based of random walks have been well-established for structural analyses of networks, as it can fully account for local as well as global topological structure of the network (Garrido and Marro, 2002; Noh and Rieger, 2004) and is very useful for finding central nodes which can be used to identify hubs (Aldous and Fill, 2002; Lovasz, 1993; Noh and Rieger, 2004; Rosvall and Bergstrom, 2008). There is rich literature addressing different variants of the problems of identifying dominant metastable sets of Markov processes (Meila and Shi, 2001; Doyle and Snell, 2000; Meerbach et al., 2005; Mattingly, 1995; Cho and Meyer, 1999; Meyer, 1989; Marek and Mayer, 2001; Deuffhard and Weber, 2005) and similar related dynamics-based concepts on discrete structures Lafon and Lee (2006); Schulman and Gaveau (2005) and for set-oriented numerics for dynamical systems Dellnitz et al. (2000); Dellnitz and Preis (2003). However, almost all of these techniques aim at complete partitions, see the review in (Li et al., 2008).

However, our new approach differs from these approaches not only with respect to the non-completeness of the partition: We also introduce a new form of the random walk that removes problems with artificial metastable structures that hamper previous spectral approaches. Fig. 2(a) illustrates the advantage of the new random walk that is adapted to the module finding problem: The standard random walk exhibits no clear spectral gap since some loops and long arcs are metastable sets in their own right. In contrast the new random walk exhibits a clear spectral gap after the first two eigenvalues because its definition strongly relates metastability to dense connectivity.

In the new approach the assignment function that assigns nodes to modules takes the form of a probability that is essentially *dynamics-based*: The assignment probability of a given node x to a certain module M is the probability that

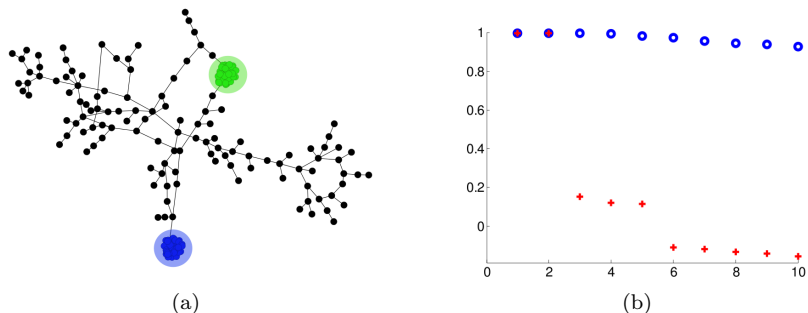


Figure 2: Example network (a) with two densely connected modules (colored green and blue) and the associated eigenvalue spectrum (b) (blue circles: standard random walker, red crosses: adapted random walker).

the random walker, if started in x , enters M first before it reaches any other module. Thus, it takes values 0 and 1 for nodes that are members of modules and values between 0 and 1 for all nodes that are not members of any module. Such an assignment function is called *soft* in contrast to *hard* assignment functions where every node is assigned to exactly one module with probability one. Soft assignment functions do also appear naturally in methods based on statistical interference but without relation to dynamical properties of the network and without being used for non-complete partitions, cf. (Santo, 2010). Very few other dynamics-based soft assignments for non-complete partitions have recently been discussed in the literature (Deuffhard and Weber, 2005; Sarich et al., 2010; Li et al., 2009). They are all based on spectral decomposition ideas but suffer from essential drawbacks resulting from artificial metastable sets and form inefficiency in application to very large networks. We will show that our new algorithm does not suffer from this problem.

The outline of the article is as follows: section 2 will introduce the necessary theory and explain the concepts needed in the remaining of the paper. In section 3 we describe our new methods in detail, before we demonstrate the method on a “real world” example network in section 4.

2 Theoretical background

2.1 Random walks on networks

Throughout the article, we will consider the graph $G = (V, E)$ associated with a network, where V is the set of nodes and E the set of edges of the graph. We denote the adjacency matrix of the graph by $(a(x, y))_{x, y \in V}$ and the degree of a node x by $d(x)$. We further assume that the graph is connected and undirected, that is, the adjacency matrix is symmetric.

The goal is to understand structural properties of the graph from the dynamics on the network. Therefore, we introduce a family of time-continuous random walks, that is, time-continuous Markov jump processes X_t on the finite

state space V , by defining their transition rules as a family of *rate matrices*

$$L(x, y) = \begin{cases} -\frac{1}{d(x)^p}, & x = y \\ \frac{k(x, y)}{k(x)d(x)^p}, & x \neq y, (x, y) \in E \\ 0, & \text{else} \end{cases} \quad (1)$$

with a scalar $p \geq 1$, non-negative weights $k(x, y)$ such that $k(x, y) = 0$ if $(x, y) \notin E$ and $k(x, y) = k(y, x)$, and $k(x) = \sum_y k(x, y)$. The interpretation of the rates is as follows: if being in node x , the expected waiting time till the next jump away from x is inversely proportional to $|L(x, x)|$, and $L(x, y)/|L(x, x)|$ is the probability that this jump leads to y . Therefore, the expected waiting time in a node is proportional to its degree $d(x)$, that is, the more neighbors a node has, the longer it takes the random walker to decide where to go next on average. One can directly compute that a Markov jump process with a rate matrix of this form has the unique invariant measure

$$\mu(x) = \frac{1}{Z} d(x)^p k(x) \quad (2)$$

and is always *reversible*, so it holds $\mu(x)L(x, y) = \mu(y)L(y, x)$. From the rate matrix (1) one can also directly compute the transition matrices of the random walk by $P_t = \exp(tL)$. Its entries $P(t, x, y)$ denote the transition probability from node x to node y in time t .

To illustrate the properties of this family of random walks introduced above, let us consider two special cases.

(C1) In the simplest case we have

$$k(x, y) = a(x, y), p = 1 \Rightarrow k(x) = d(x), \quad (3)$$

such that the process jumps to one of the neighbors without preference for one of them.

(C2) A more elaborated choice of the weights is ($p = 1$)

$$k(x, y) = a(x, y) \cdot \left(1 + \langle a_x, a_y \rangle\right), \quad (4)$$

where a_z is the z th row of the adjacency matrix and $\langle \cdot, \cdot \rangle$ is the usual Euclidean scalar product. Here, whether a jump away from x leads to y depends on the similarity of the neighborhood of the nodes. This can be seen better when using the often-used clustering coefficient for a node x , defined by

$$c(x) = \sum_{y, z \in V} a(x, y)a(x, z)a(y, z)/d(x)(d(x) - 1),$$

and measures the connectedness of the neighborhood of x via the ratio of the number of edges between neighbors of node x and the maximum possible number of such edges. By choice (4), it enters the invariant measure of our random walk,

$$\mu(x) = \frac{1}{Z} d(x)^2 \left(1 + (d(x) - 1) \cdot c(x)\right), \quad (5)$$

showing that the nodes with high degree and high clustering coefficient become very attractive.

For the sake of simplicity, we will assume in the following that the graph is unweighted. If we had weights $w(x, y)$ assigned to edges $(x, y) \in E$, we could simply choose $k(x, y) = w(x, y)$.

Remark 1 Usually, in network clustering one does not consider time-continuous random walks as above but a Markov chain with one-step transition matrix directly given by the adjacency structure,

$$P(x, y) = \frac{a(x, y)}{d(x)}. \quad (6)$$

Note that it is exactly the embedded Markov chain of the time-continuous random walk (3). As already outlined in the introduction we will not use this standard random walk since the process defined by L has crucial advantages in comparison to the Markov chain associated with P ; these advantages will be explained in detail in the next sections.

However, the standard discrete-time setting could still be kept by considering the one-step transition matrix $P_t = \exp(tL)$ associated with L for a certain preselected time t instead of P from (6). This also is no option because of the following reason: P_t in general is *not* sparse even if L is a sparse matrix (what it normally is for modular networks). As we will see later the sparsity of L is essential for getting algorithms that scale linearly with network size.

2.2 Modules and metastability

We are aiming at finding modules in our network. Assume we have m modules; these then are *disjoint* subsets $C_i \subset V$, $i = 1, \dots, m$, of the set of all nodes, V . The union of all modules form the set $\mathcal{M} = \cup_i C_i$. We want a fuzzy partition of the network, that is, we assume that \mathcal{M} does *not* contain all nodes of the network, meaning, there is a non-empty set $T = V \setminus \mathcal{M}$, which we call *transition region*.

When considering the relation of a specific module, say C_i , to the others, we will need the set $\mathcal{M}_i = \mathcal{M} \setminus C_i$, the union of all modules except C_i . Our key idea is that the modules are metastable sets of the random walk, that is, on the one hand they are attractive for the random walk while, on the other hand, they are well separated in the sense that *communication* between them is rare.

In order to make this more precise we will now introduce the notion of *metastable sets* of the random walk. In a rough sense, the random walk has metastable sets, $C_i \subset V$, $i = 1, \dots, m$, if it exhibits a specific relation between two timescales: the typical *return time* R the random walk needs to enter one of the C_i , if started outside of any module, is small compared to its typical *waiting time* W between transitions from one of the C_i to another one. In order to quantify these timescales we first denote by $\tau(A)$ the random time the process X_t needs to enter a set A . Then $\mathbb{E}_y(\tau(\mathcal{M}))$ denotes the expected entry time of the process into an arbitrary one of the C_i , if started in some node $y \in T$ in the transition region $T = V \setminus \mathcal{M}$. Likewise $\mathbb{E}_i(\tau(\mathcal{M}_i))$ denotes the expected entry

time into one of the C_j with $j \neq i$, if started from the invariant measure μ in C_i . The random walk is metastable with regards to the sets C_i , $i = 1, \dots, m$, if

$$R = \max_{y \notin \mathcal{M}} \mathbb{E}_y(\tau(\mathcal{M})) \ll \min_{i=1, \dots, m} \mathbb{E}_i(\tau(\mathcal{M}_i)) = W,$$

or, equivalently, if the relation of return and waiting time is small, $R/W \ll 1$. This still is a rough definition since there may be many different collections of disjoint sets C_i such that R/W is small. Below we will outline how to find *optimal* metastable sets that then define the modules of the network. That is, modules are optimal metastable sets of the random walk.

However, the reader should be aware that we cannot simply define the optimal modules via the property of minimizing R/W since every full decomposition, that is, every choice with $T = \emptyset$, leads to $R = 0$. Therefore we must look deeper.

2.3 Why time-continuous random walks?

Since our random walk is reversible, all eigenvalues Λ of the rate matrix L and λ of the associate transition matrix P_t are real with $\Lambda_0 = 0$ respectively $\lambda_0 = 1$ being the largest ones, and $\lambda = \exp(t\Lambda)$ in general. It is well-known that there is a direct relation between the existence of metastable sets and the largest eigenvalues of the transition matrix P_t of the random walk (Schuette and Huisinga, 2003; Huisinga and Schmidt, 2006; Djurdjevac et al., 2010a; Sarich and Schuette, 2011). For example, in the case of just two metastable modules, there is exactly one other eigenvalue $\lambda_1 = \exp(t\Lambda_1)$ of P_t close to $\lambda_0 = 1$ such that $\mathbb{E}_1(\tau(\mathcal{M}_1))$ is approximately given by $1/|\Lambda_1|$, while all other eigenvalues of P_t are significantly further away from 0. Whenever there is a gap after the leading m eigenvalues $\lambda_0, \dots, \lambda_{m-1}$ close to one we will find m metastable sets, and the longest relaxation timescales of the random walk, $t_i = 1/|\Lambda_i|$, are encoded by the leading eigenvalues $\Lambda_0, \dots, \Lambda_{m-1}$. Therefore, spectral clustering methods usually search for a spectral gap appearing after several eigenvalues which are close to one; their numbers then give the number of metastable sets. Our key idea is to exploit these groups of nodes which are strongly interconnected, but have only few connections to the rest of the network. These are candidates for forming modules and are metastable in the sense of the random walk as described above. The problem is that for the standard random walk (6) many other structures of nodes in a network also can cause metastability, for example large cyclic structures or long pathways connecting strongly connected groups. In the following, we use the network that was introduced in Figure 1 as an example. Figure 3 shows a different visualization and the spectra of the standard and the time-continuous random walk.

Obviously, the spectrum of the standard transition matrix (6) does not offer a clear gap to give an idea about the number of modules being present. In contrast, the time-continuous random walk shows a small gap after 7 and a strong gap after 8 eigenvalues. The benefit of the introduction of the waiting times proportional to the degree of the nodes (1) is that the process becomes faster in simple regions, which are loosely connected, and slows down in more complicated, interconnected structures. As it is observed in Figure 3, this leads to a better coherence of connectivity, the idea of modules, and metastability.

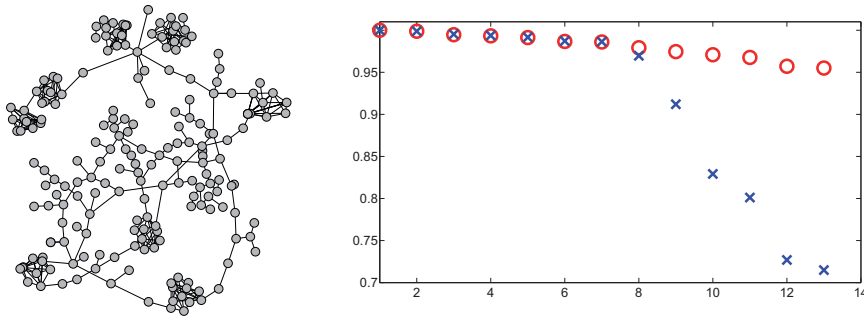


Figure 3: Example network and first 13 eigenvalues λ of standard random walk (red circles) and time-continuous random walk (blue crosses).

2.4 Fuzzy decomposition

When looking into the relevant literature, most articles are concerned with *complete* partitioning of networks, that is, hard clusterings in which the modules form a complete partition of the network. As outlined, we are not aiming at such complete partitions because in many relevant cases there are nodes which cannot be reasonably assigned to a particular module, but rather have an affiliation to several modules. For this purpose, it is necessary to consider a *fuzzy* decomposition of the network. That is, for every node x that does not belong to any module we specify its affiliation $f_i(x) \in [0, 1]$ to module i such that $\sum_i f_i(x) = 1$.

If we assume for a moment that we have already identified the modules C_i , there is a natural way to define this affiliation by learning from the random walk. To do this, we simply start the random walk in node x and see which module it will enter next. Then, we set the affiliation $f_i(x)$ to be the probability that the next module to be entered is C_i . There are two advantages to this approach. First, one can compute these affiliation functions, which are also known as *committor* functions, very efficiently by solving sparse, symmetric and positive definite linear systems. For sets C_1, \dots, C_n it is shown for example in (Metzner et al., 2009) that the committor f_i solves the linear system

$$\begin{aligned} (Lf_i)(x) &= 0 & \forall x \in T \\ f_i(x) &= 1 & \forall x \in C_i \\ f_i(x) &= 0 & \forall x \in C_j, j \neq i. \end{aligned} \quad (7)$$

By inserting the constraints into the linear equation and rewriting, one gets an equivalent linear equation of the form

$$\hat{L}\hat{f}_i = -g_i \quad (8)$$

where

$$\hat{L} = (\hat{L}(x, y))_{x, y \in T}, \hat{f}_i = (\hat{f}_i(x))_{x \in T} \quad \hat{L}(x, y) = L(x, y).$$

and

$$g_i = (g_i(x))_{x \in T} \quad g_i(x) = \sum_{z \in C_i} L(z, x).$$

Then, f_i is given by the constraints on the modules and by \hat{f}_i on the transition region. Moreover, by multiplying (8) with $D_{\hat{\mu}} = \text{diag}(\hat{\mu}), \hat{\mu}(x) = \mu(x), x \in T$ from the left, one obtains a sparse, symmetric and positive definite formulation because of the reversibility.

Second, this fuzzy decomposition can be well interpreted in the sense of a coarse graining of our random walk by Markov State Modeling (Djurdjevac et al., 2011; Schuette et al., 2011; Djurdjevac et al., 2010a;b; Deuffhard et al., 2000).

3 Methods

According to our considerations above the key idea of our approach is: Identify *modules as optimal metastable sets* and determine the *fuzzy decomposition* of the transition region T via *committor functions*.

3.1 Identification of modules

In order to approach the question of how to identify modules, let us first assume (in contrast to what we want to achieve) that the modules $C_i, i = 1, \dots, m$, form a full partition of the network. Furthermore, let us fix a timescale t and consider the transition matrix $P = P_t$ of our random walk X_t . Then the partition induces a coarse grained random walk on state space $\{1, \dots, m\}$ that jumps from module to module with transition probability $P(i, j) = \text{Prob}_{\mu}(X_t \in C_j | X_0 \in C_i)$ where the index μ refers to the fact that we start in C_i distributed due to the invariant measure. As it is well-known (Schuette et al., 2011; Djurdjevac et al., 2010a;b; Sarich, 2011) the matrix \hat{P} can be written in the form $\hat{P} = QPQ$, where Q is the orthogonal projection onto the finite-dimensional space D of all step-functions that are constant on the sets C_i .

Now assume that we do not have a full partition but just the candidate modules $C_i, i = 1, \dots, m$ and m associated non-negative affiliation functions f_i with $\sum_i f_i(x) = 1$ for all nodes x . Assuming that the f_i are the committor functions, we still find a coarse grained random walk that jumps between the modules but now takes the dynamics on the transition region T into account correctly (Sarich, 2011). It again has an $m \times m$ projected transition matrix $\hat{P} = QPQ$, where now Q is the orthogonal projection onto the space spanned by the affiliation/committor functions.

Now we want to find sets C_1, \dots, C_m such that the longest relaxation timescales of our random walk, being encoded by m dominant eigenvalues $\lambda_0, \dots, \lambda_{m-1}$ of P , are optimally reproduced by the timescales of the coarse grained random walk, encoded by the eigenvalues $\hat{\lambda}_0, \dots, \hat{\lambda}_{m-1}$ of \hat{P} . According to (Djurdjevac et al., 2010b;a) one can write

$$\max_{i=0, \dots, m-1} |\lambda_i - \hat{\lambda}_i| \leq \lambda_1(m-1) \max_{i=0, \dots, m-1} \delta_i^2 \quad (9)$$

where u_i is the normalized eigenvector of P with respect to λ_i and

$$\delta_i = \|u_i - Qu_i\| \quad (10)$$

is the associated projection error. That means that the dominant eigenvalues of P are well approximated by the eigenvalues of the projected matrix \hat{P} if the projection error of the corresponding eigenvectors is small enough.

Remark 2 Even if we have a full partition, that is, $f_i(x) \in \{0, 1\}$, the last statement still applies: The subspace D consists of step-functions that are constant on the sets C_i and δ_i measures how much the eigenvectors are varying within each set C_i . Many spectral clustering methods like PCCA (Deuffhard and Weber, 2005) exploit this connection to identify the optimal clustering, which minimizes this projection error. This will result in a complete metastable partitioning of the network (Deuffhard et al., 2000).

Let us return to the fuzzy affiliation functions f_i given by the committors. In (Sarich and Schuetter, 2011) it is shown that for any eigenvalue λ_i of T and the corresponding, normalized eigenvector u_i it holds

$$\delta_i \leq p(u_i) + 2\mu(T)p_{max}(u_i) + r(T)(1 - \lambda_i) \left(\sum_{x \in T} u_i(x)^2 \mu(x) \right)^{\frac{1}{2}} \quad (11)$$

with

$$\begin{aligned} r(T) &= \sup_{\substack{\|v\|=1, \\ v=0 \text{ on } \mathcal{M}}} \left(\frac{1}{\sum_{x \in T} (v(x) - (Pv)(x))^2 \mu(x)} \right)^{1/2} \\ p(u_i) &= \|e_i\| \quad p_{max}(u_i) = \|e_i\|_{\infty} \\ e_i(x) &= \begin{cases} 0, & \text{if } x \in T, \\ \frac{1}{\mu(C_j)} \sum_{y \in C_j} u_i(x) - u_i(y) \mu(y), & \text{if } x \in C_j. \end{cases} \end{aligned} \quad (12)$$

From this inequality we can deduce that modules should satisfy two things in order to ensure small projection errors $\|Q^{\perp}u_i\|$ for the dominant eigenvectors, and hence be a good approximation of the largest eigenvalues (3.1), which correspond to metastability of the random walk. First, from the transition region T the random walker should always enter some module quickly enough such that $r(T)(1 - \lambda_i)$ is small enough. More precisely, the more eigenvalues of P we want to approximate, the faster the transition region T has to be left. Second, the dominant eigenvectors should be almost constant on the modules to guarantee small values of $p(u_i)$ and $p_{max}(u_i)$. It will be particularly useful that the error bound decomposes into these two parts. The factor $r(T)(1 - \lambda_i)$ takes only the transition region T into account and the errors $p(u_i)$ and $p_{max}(u_i)$ depend only on the partitioning of $\mathcal{M} = V \setminus T$ into the modules.

3.2 Algorithm

The standard approach for the identification of modules as the most metastable sets (see Section 2.2) would be to compute a fuzzy decomposition of the network and to define the module C_i to contain all nodes x , for which $f_i(x) \geq \theta$ for some threshold $\theta \in [0, 1]$, that is,

$$C_i = \{x \in V : f_i(x) \geq \theta\}. \quad (13)$$

The problem is that the computation of fuzzy decompositions, e.g. (Deuffhard and Weber, 2005; Sarich et al., 2010), is usually very costly for large networks. We will now follow the ideas derived above and compute the modules and the fuzzy decomposition in three separate steps:

1. Identify the transition region T , which will not be assigned to any module.
2. Split the remaining nodes of the network into modules C_1, \dots, C_m .
3. Compute the fuzzy decomposition as the committors with respect to the modules.

There are several strong advantages of identifying the transition region T first. We will see that it allows to transform the fuzzy clustering problem for the whole network into a hard clustering problem, that is, a full partition with respect to the nodes belonging to \mathcal{M} only. This means, the number of nodes we have to cluster will decrease, often dramatically. Moreover, the resulting cluster problem will be easier and more robust to solve because it becomes hard (as opposed to fuzzy or soft) and we erased those nodes, which might be problematic to cluster because of affiliation to several modules. We will show now that every step can be computed very efficiently.

Step 1 We want to ensure a small factor $r(T)(1 - \lambda_i)$ in the error bound (11) for the dominant eigenvalues, which just depends on the transition region T . If we fix a specific choice of T , and let an ensemble of infinitely many random walkers start only in this region T , $r(T)$ measures how many of the random walkers will leave the transition region. That is, the higher the probability is that the random walker will leave T quickly, the smaller the factor $r(T)$ will be. Next, this factor is compared to $(1 - \lambda_i)$ for the dominant eigenvalues. This yields the following interdependency: For eigenvalues close to one, $(1 - \lambda_i)$ will be rather small, which gives us more flexibility with $r(T)$, that is, it can take the random walker more time to leave the transition region T . Remember that the closer to one the eigenvalues are, the stronger they indicate the presence of metastability in the system. So, if we also want to consider modules, which are less metastable, we will have to approximate eigenvalues, which are less close to one and therefore, the region T has to be left more quickly.

Now, algorithmically this leads to the following idea: We take the invariant measure μ^* of the random walker using a rate matrix as defined in (1) and parameter $p = 0$. That is, we turn off the effect of waiting times, which made the modules in the network more metastable. Then, we consider the random walk for $p = 1$ and choose a lag time $\alpha > 0$, at which we want the random walker to leave the transition region. As explained above, we will choose a rather large α if we are interested in finding only the most metastable set of modules, and decrease α if we also want to identify modules with less metastability. Then, we choose

$$\mathcal{M}^\alpha = \{x \in V | (P_\alpha^T \mu^*)(x) > \mu^*(x)\} \quad (14)$$

to be the set containing the modules with respect to the metastability parameter α . Connecting to the ensemble point of view from above, this set (14) is exactly the region, which rather attracted random walkers in the ensemble than let random walkers leave within the time step α .

Step 2 Having identified the set \mathcal{M}^α we now have to find a full partition into the final modules. For this purpose, we consider the random walk only on the

nodes belonging to \mathcal{M}^α with transition matrix

$$\hat{P}_\alpha(x, y) = \sum_{z \in V} P(x, z) q_y(z), \quad x, y \in \mathcal{M}^\alpha, \quad (15)$$

where $q_y(z)$ is the probability that y will be the next node from \mathcal{M}^α that is hit by the random walk starting in z , i.e., it can be computed from a linear equation like (7) with $C_i = \{y\}$ and $C_j = \emptyset$ for all other j . That is, $\hat{P}_\alpha(x, y)$ describes the transition probabilities between the nodes of \mathcal{M}^α , ignoring the waiting times and the transition region. Then, we use a hard spectral clustering method to split \mathcal{M} into the modules C_1, \dots, C_m . Note that \hat{P}_α describes the dynamics only between the nodes of modules. The absence of nodes with affiliation to several modules makes the spectrum of \hat{P}_α usually very amenable for interpretation.

Step 3 We compute the committors with respect to the modules C_1, \dots, C_m to get a fuzzy clustering of the remaining nodes. As mentioned above, committors can be computed by solving positive definite, symmetric linear systems which will be as sparse as the adjacency matrix. Such computations can be performed efficiently, even for large systems.

Algorithm summary.

1. Input: $\alpha > 0$, matrix A

Compute L according to (1), e.g. $p = 1$.

Solve

$$\frac{d}{dt} v_t = L^T v_t, \quad v_0 = \mu^*$$

until $t = \alpha$, so $v_\alpha = e^{L^T \alpha} \mu^*$. Set

$$\mathcal{M}^\alpha = \{x \in V | v_\alpha(x) > \mu^*(x)\}.$$

2. Compute committors $q_y(z), y \in \mathcal{M}^\alpha, z \in T$ with respect to the single nodes of \mathcal{M}^α and for $x, y \in \mathcal{M}^\alpha$

$$\hat{P}_\alpha(x, y) = \sum_{z \in V} P(x, z) q_y(z).$$

Choose number of modules n according to the spectrum of \hat{P}_α and use hard clustering method, e.g. (Deuffhard et al., 2000).

3. Compute committors $f_i(x)$ for every $x \in T$ with respect to the modules C_1, \dots, C_m .

Computational effort In the following, the number of nodes in the network is denoted by n , and the number of nodes belonging to the set \mathcal{M}^α is denoted by m .

Step 1: For example in (Al-Mohy and Higham, 2010) it is shown that the computational effort is dominated by matrix multiplications. For a large, sparse matrix L this effort is $\mathcal{O}(n)$.

Step 2: First, we have to solve a symmetric, positive definite linear system (8) for m right hand sides. Since the matrix \hat{L} is large and sparse, conjugate gradient methods allow to compute the solution in $\mathcal{O}(mn)$ point operations. Then, we have to compute a hard clustering with respect to the coarse grained random walk with $m \times m$ transition matrix \hat{P}_α . For this task, a lot of algorithms exist, for example, (Deuffhard et al., 2000; Li et al., 2008). The fastest combinatorial methods perform in $\mathcal{O}(m^2 \log m)$.

Step 3: Again, we have to solve linear systems of the form (8) as in step 2 with the same matrix \hat{L} , but for even less right hand sides.

Whole algorithm: This shows that the overall effort is dominated by step 2, where we have to compute a hard clustering for the m nodes belonging to \mathcal{M}^α . The total effort scales like $\mathcal{O}(m^2 \log m) + \mathcal{O}(mn)$. If $m \ll n$, that is, if the number of nodes in modules is much smaller than the number of nodes not assigned to modules, then the effort scales linearly with the total number of nodes. In general, our algorithmic strategy reduces the computational effort to calculate a fuzzy decomposition of a network with n nodes to the effort of computing a hard clustering for m nodes.

To substantiate these theoretical considerations we applied our method to several generated test networks. All these networks have the same structural concept in common. Between the n nodes of the network, edges are generated randomly with the intention that 4 modules in the sense above should form, each of them containing s nodes. Here, n and s are input parameters. Then, the network is constructed such that for a node in an intended module the average number of edges to other nodes within the same module is 9 and the average number of connections to other nodes is 2. The average degree of a node that should not belong to any module is approximately 3 and there is no preference with respect to neighbours. In Fig. 4, such a network is shown with $n = 500$ nodes and $s = 50$ nodes per module including the modules found by our algorithm. For each generated network the spectrum of the matrix \hat{P}_α clearly indicated the presence of the 4 modules.

According to this blueprint we generated networks with respect to all combinations between the total network sizes 500, 1000, 1500, 2000, 2500 and single module sizes 40, 50, 60, 70, 80. In Fig. 5, the computation time in seconds that was needed to perform the whole module identification on a desktop computer is plotted over the total network size for several choices of module sizes. As expected, each of the curves follows a linear trend while larger modules lead to a higher computational effort.

Choice of timescale α As discussed, different choices for the parameter α may lead to different clustering results. From the definition of the set \mathcal{M}^α it is clear that there exist two values $\alpha_\infty > \alpha_0$ such that the set \mathcal{M}^α does not change for $\alpha > \alpha_\infty$ and for $\alpha < \alpha_0$, respectively. So we have two limiting parameter values of α , which give only the most dominant modules or resolve also the less pronounced modular structures. Since α is connected to the timescale at which the random walk leaves the transition region, it is possible to get an idea about reasonable values for α from the spectrum of the generator L . That is, if the dominant eigenvalues of L are denoted by $0 = \Lambda_0 > \Lambda_1 \geq \Lambda_2 \geq \dots$ the implied timescales of the random walk, which are given by $1/|\Lambda_1| \geq 1/|\Lambda_2| \geq \dots$,

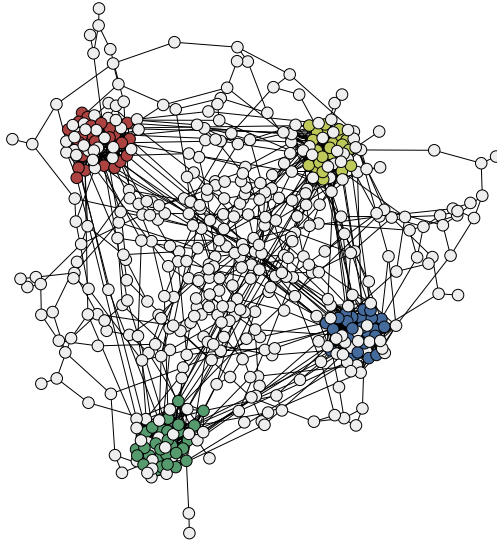


Figure 4: Test network with $n = 500$ nodes and $s = 50$ nodes per module and the modules found by our algorithm.

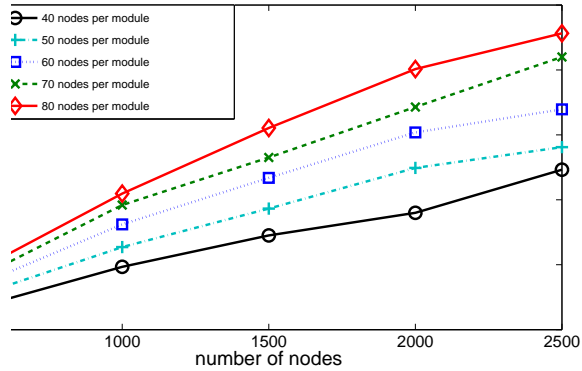


Figure 5: Computation time in seconds over network size for different module sizes.

provide estimates for possible choices of α . If there is a cluster of eigenvalues $0 = \Lambda_0 > \Lambda_1 \geq \dots \geq \Lambda_k$ around 0 separated by a spectral gap from all smaller eigenvalues, then a good choice of α would be $1/|\Lambda_k| > \alpha > 1/|\Lambda_{k+1}|$. Several spectral gaps therefore would give us a list of proposals for good values of α . As a general rule, the number m of nodes in the modules will increase with decreasing α . Since the last paragraph on the computational effort has shown that small m is computationally advantageous, we should start with the largest α from our proposal list.

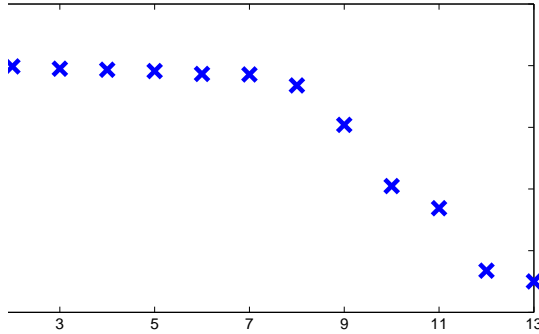


Figure 6: 13 largest eigenvalues of the generator L for the example network.

3.3 Multilevel structure

We will now illustrate the properties of our method using the example network shown in Figure 3. We start with the first step of our method, that is, the identification of the transition region T and \mathcal{M}^α . Therefore, we have to choose a metastability parameter α . To demonstrate the effect of this choice, we will consider two different metastability levels $\alpha = 1000$, and $\alpha = 150$. These values have been selected because the spectrum exhibits gaps after the eighth and after the ninth eigenvalue with $1/|\Lambda_8| = 1254$, $1/|\Lambda_9| = 254$, and $1/|\Lambda_{10}| = 104$.

In Figure 7, we see the spectrum of \hat{P}_α the corresponding random walk restricted to the nodes of modules as in (15).

The spectra offer clear gaps: for $\alpha = 1000$ after 8 eigenvalues, and for $\alpha = 150$ we find two additional eigenvalues indicating less pronounced metastability. Figure 8 shows why this is happening and is a good example for the multilevel structure that is detected by our method.

The upper two illustrations of the network show in black the nodes, which have been marked by our algorithm to belong to \mathcal{M}^α . For a rather large $\alpha \geq 1000$ this set only contains the most metastable parts of the network. When decreasing α to 150, we know that the region T has to be left even faster by the random walk. Therefore, sets of nodes referring to the next less pronounced metastability are added to \mathcal{M}^α . In the second step of our method we derive a partition of these nodes into the final modules C_1, \dots, C_n by hard clustering with respect to the associated \hat{P}_α . Note that having erased the transition region the spectrum of the random walk clearly indicate the number of modules corresponding to the chosen level of metastability. Nevertheless, we could also vary the number of modules we want to split the region \mathcal{M}^α into. As it is also shown in Fig. 8 this leads to another form of hierarchy which we can analyze with our method. This means, we can choose to find new less pronounced modules by decreasing α or splitting existing modules hierarchically into several modules by increasing the cluster number in the second hard clustering step of the algorithm.

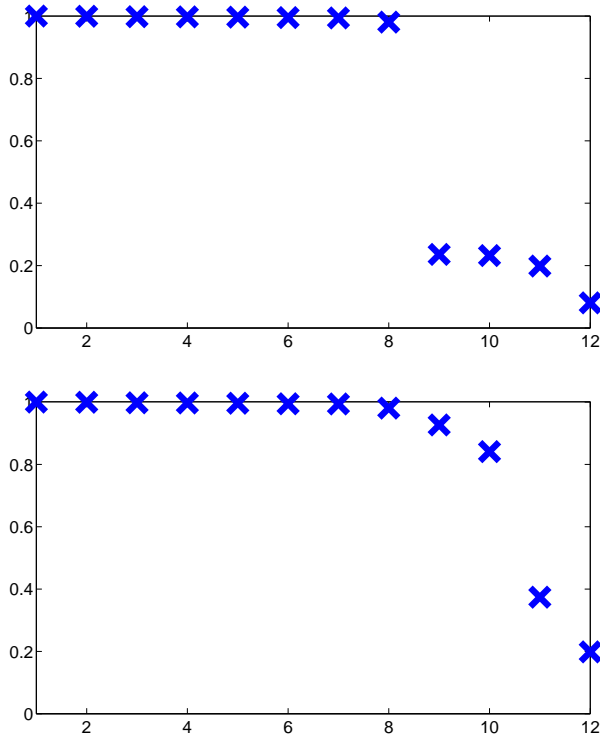


Figure 7: Top: Spectrum of \hat{P}_α for $\alpha = 1000$. Bottom: Spectrum for $\alpha = 150$.

3.4 Identification of hubs

After we have determined the modules we will now look at the transition behavior of the network, using the framework of Transition Path Theory (TPT) (E. and Vanden-Eijnden, 2010) and (Metzner et al., 2009). More specifically, we will show how to identify hubs, nodes that are essential for the communication between the modules. We will present two different concepts for declaring a node to be a hub, developed in (Djurdjevac et al., 2011).

We start by observing transitions from a module C_i to the union of all other modules $M_i = \mathcal{M} \setminus C_i$, taking into account only these parts of trajectories (realizations of the random walk), where the random walker transits directly from C_i to M_i . That is, for the n^{th} transition we consider the sequence of states

$$P_n = [x_n^{C_i}, x_n^1, \dots, x_n^k, x_n^{M_i}], \quad (16)$$

called the n^{th} reactive trajectory, where $x_n^{C_i} \in C_i$, $x_n^i \in T$, $x_n^{M_i} \in M_i$.

The union of all such trajectories is called *the set of reactive trajectories*. Statistical properties of these trajectories will provide us about the global, as well as local transition behavior of the system. We define the *discrete probability current* as

$$f_{xy}^{C_i M_i} = \begin{cases} \mu(x) f_i(x) L(x, y) (1 - f_i(y)), & \text{if } x \neq y \\ 0, & \text{otherwise,} \end{cases} \quad (17)$$

where $L(x, y)$ are the entries of the generator defined using (3). This is the average flow of reactive trajectories when going from state x to y , per time

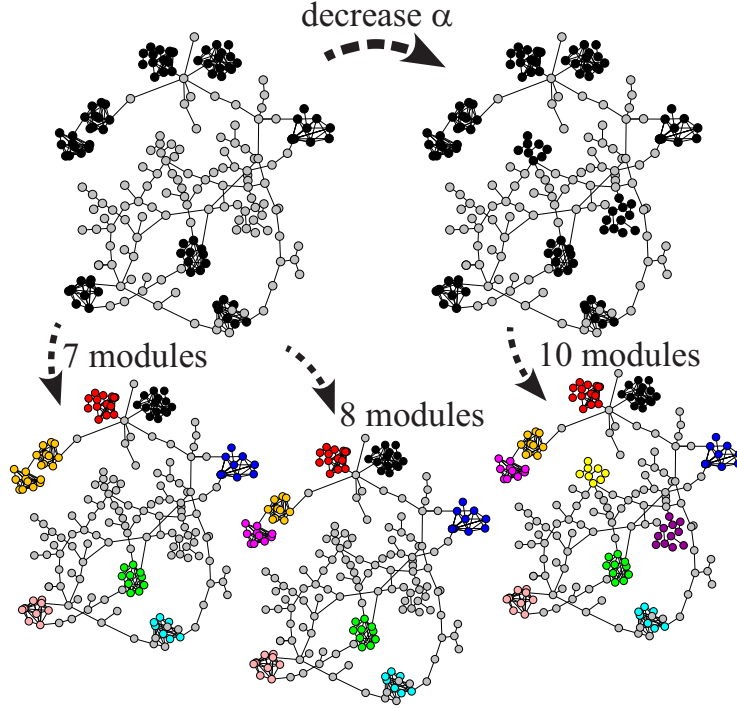


Figure 8: Multilevel structure of modules.

unit. To calculate the net amount of probability current between two states, we introduce the *effective current* f_{xy}^+ :

$$f_{xy}^+ = \max(f_{xy}^{C_i M_i} - f_{yx}^{C_i M_i}, 0). \quad (18)$$

We can now describe the global transition behavior from C_i to M_i , using the transition rate $k_{C_i M_i} = \sum_{x \in C_i, y \in V} f_{xy}^+$, that is the average number of transitions from C_i to M_i per time unit. For calculating the number of these transitions that are passing through a single node y

$$k_y = \sum_{x \in P_y} f_{xy}^+ = \sum_{x \in S_y} f_{yx}^+, \quad (19)$$

where $P_y = \{x \in V : f_{xy}^+ > 0\}$, $S_y = \{x \in V : f_{yx}^+ > 0\}$. Now, for every $y \in T$, we can calculate the *importance rate* of reactive trajectories that go through y by

$$p_y^{C_i M_i} = \frac{k_y}{k_{C_i M_i}}. \quad (20)$$

In this sense, a hub is a node which has a high importance rate, meaning that most of the communication goes through this node.

The second approach is to distinguish between the transition paths in the network and define a measure of how important they are for the global communication. Notice that every single transition from module C_i to $\mathcal{M} \setminus C_i$ can

be characterized by the path the random walker takes from C_i to $\mathcal{M} \setminus C_i$. To do this, let us assign to every edge of this path the effective current, meaning, the net average number of reactive trajectories per time unit that make transitions through this edge when going from one set to another. The edge with the minimal effective current is called the *dynamical bottleneck* of the path, since it limits the amount of flow that can be transported through this path. In practical applications, reaction paths that have the maximal minimal current are of particular interest, since they can transport the most flow. These will be the most important reaction paths. Therefore, those nodes that are taking part in the most important transitions (in the above sense) in the network would be hubs.

Example Let us now apply our method for identifying hubs on the example network from Figure 3. As before, we set $\alpha = 1000$ and $\theta = 0.85$, where θ is used as in (13). With these parameters the algorithm identifies eight modules that are shown in Figure 9. We calculated the importance rates given by equation (20) for all nodes that belong to the transition region. For the sake of simplicity, we highlight only some of these nodes, namely the top 14 with the highest importance rate. Out of these nodes we picked three nodes (A , B and C in Figure 9) to illustrate different type of hub nodes. Node B is a node that connects two modules with the rest of the network and therefore has a high importance rate. Compared to this, node C has a higher importance rate, since it is a node that connects four modules with the rest of the network. However, node C is not crucial for the communication between these four modules and therefore its importance rate is smaller than the one of node A . This is the reason why node A has the highest rate of all the nodes, since this node connects four of the modules with the rest of the network and moreover, A is also the only node that connects these four modules among each other. Therefore, node A is crucial for their communication.

4 Results

In this section we demonstrate our new method by analyzing a network of US political books, which was introduced in (Newman, 2006) (see Figure 4). The network contains 105 nodes, each representing a book about US politics sold by the online retailer Amazon. If customers frequently buy book A and B together¹ an edge will be inserted between nodes (books) A and B .

We apply our algorithm to this network to obtain modules and a transition region. After performing step 1 (Section 3.2) for $\alpha = 100$ we identify the region of modules \mathcal{M}^α , which is colored black in Figure 11. Setting aside the transition region we can then compute eigenvalues of $\hat{P}_\alpha(x, y)$ on the nodes of \mathcal{M}^α . The largest eigenvalues are: 1.00, 0.97, 0.31, 0.25, 0.18. As we noted previously (Section 3.2), the resulting spectrum is convenient to interpret: It is easy to see that there is a clear gap after the first two eigenvalues, indicating that clustering into two modules is the natural choice for this example. Next, we perform hard clustering (step 2) to assign nodes to these two modules. Figure 11 shows the final modules computed in this step.

¹According to Amazon’s “Customers who bought this book also bought...” feature

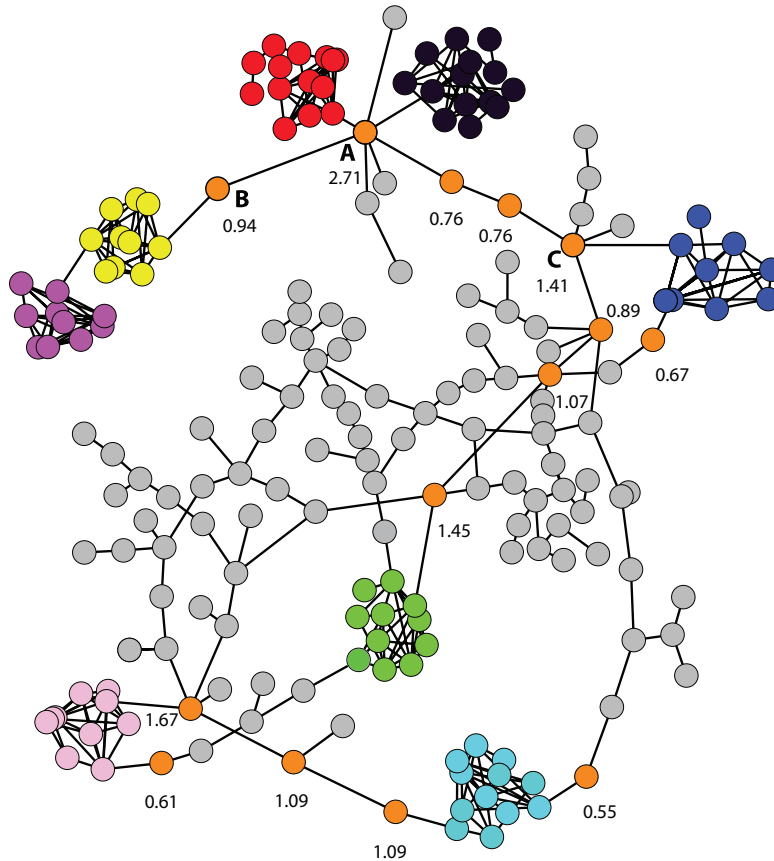


Figure 9: This shows the 14 most important hubs in our example network colored in orange and having their importance rates written next to them.

For this particular example, we can actually interpret the results. In (Newman, 2006), the books have been classified according the author's personal judgment into three categories of political alignment: conservative, liberal, and neutral. This manual, hard clustering is illustrated in Figure 12 together with the modules.

Moreover, we can compute the committors (see step 3) to find the affiliation of the remaining nodes to the modules. Figure 13 shows also two sets which consist of all nodes that have a higher affiliation to one of the modules for two

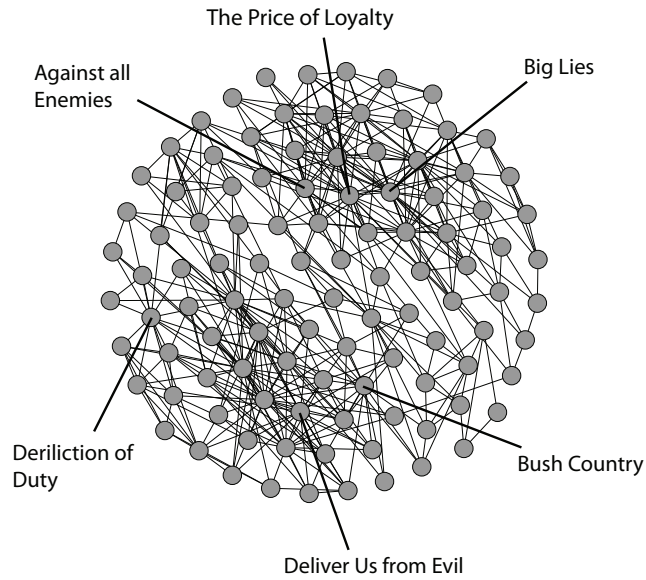


Figure 10: The political books network. (Taken from (Newman, 2006))

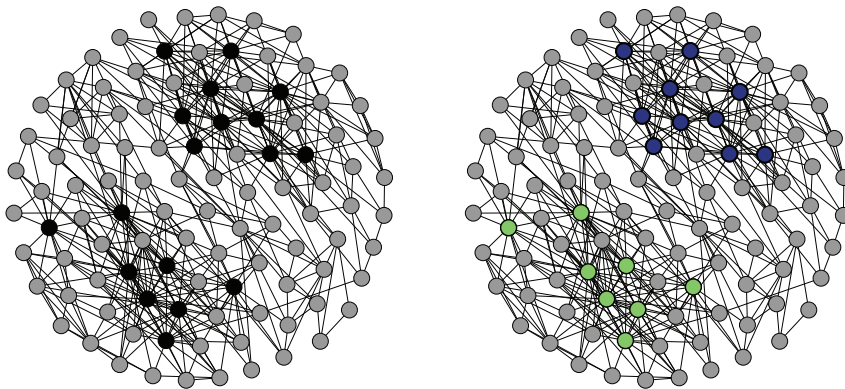


Figure 11: Left: Black nodes have been identified by the algorithm to belong to modules. Right: Clustering the nodes within modules using PCCA.

different thresholds.

Most books that have been classified by Newman as belonging to the conservative or liberal group also have a rather high affiliation to one of the modules

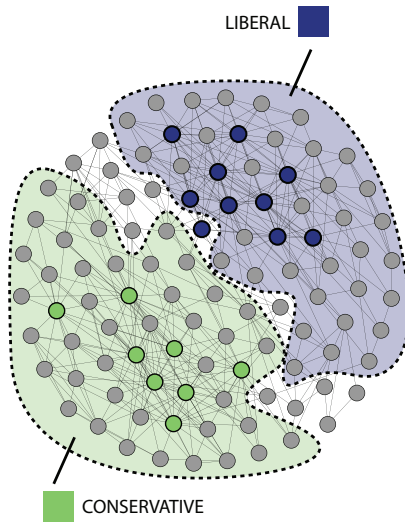


Figure 12: Identified modules and the assignment of the books to three categories (conservative, liberal, and neutral without coloring) according to Newman’s personal judgment as in (Newman, 2006).

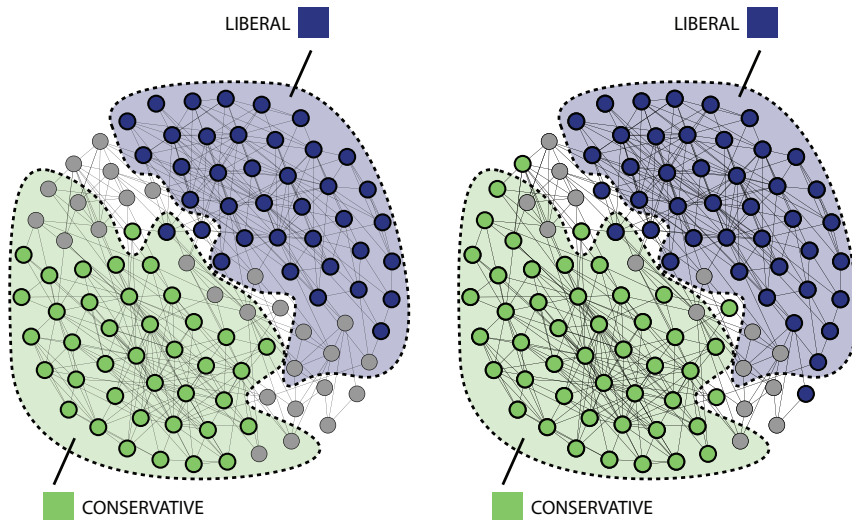


Figure 13: Left: Nodes with affiliation higher than $\theta = 0.9$. Right: Nodes with affiliation higher than $\theta = 0.8$

found by our algorithm. On the other hand, for most of the neutral books we find an affiliation which is less specific. Moreover, if we generate a hard clustering by assigning every book to the module, which it has the highest affiliation to, all liberal books end up in the same cluster, and only two books that have been classified as conservative are merged into the liberal cluster.

Of course, one has to be careful with the interpretation. For example, one cannot expect any clustering algorithm applied to this network to uncover the hard assignment that was based on human judgment. In (Newman, 2006), much

more background information was used to decide about the political alignment of the books. Moreover, it is a hard assignment to the three categories conservative, liberal and neutral, but as usual some books will have a stronger affiliation to one class than others.

In contrast, the information the network is based on is very different. It has only used the selling statistics of Amazon. That is, our algorithm found that there are two strongly interconnected groups of books, namely the modules above. By construction of the network this means that they have been purchased frequently by the same customers. Now, one could formulate the hypothesis that people having a particular political disposition would buy rather corresponding books. The results above would support this idea, but it also shows that such a simplification cannot hold for every single book. Therefore, one should not always expect a perfect full partitioning of a network like this, matching the background information. The approach of identifying only subgroups, that is, modules in the network, has the advantage that one only looks at the books for which an interpretation really seems to exist. For the remaining books one just computes tendencies, then. We have seen that using this algorithmic strategy one is able to uncover very accurate and interesting connections, which are also interpretable.

5 Concluding Remarks

Random walks on complex networks offer a way for a global analysis of the topology of the network. We have shown how to adapt recent research regarding coarse graining of random walks for the tasks of finding modules and hubs based on fuzzy decomposition of complex modular networks. To this end, we demonstrated how to overcome two essential difficulties of spectral methods in application to network decomposition: (1) We presented a re-design of the transition rules of the random walk such that the dominant spectrum of the adapted random walk exhibits spectral gaps related to the modular structure. (2) The resulting algorithms can be applied to large-scale networks since they scale linearly with network size as long as the connectivity is sparse.

Other important problems, like how to identify the appropriate number of modules in a network or the related question of how to uncover the multilevel structure of many modular networks, have also been discussed. One crucial aspect, however, has been kept open: We did not report on applications of our novel approach to very large real-world networks. This is mainly because in such cases "correct" results do not exist in general, and we would have to compare the output of different algorithms, which is a topic in its own right. Future research will demonstrate how our approach performs on, for example, biological networks in comparison to other algorithms and the insights that can be gathered through its application.

6 Acknowledgments

The authors would like to thank Tiejun Li for some insightful discussions about fuzzy decompositions of networks. ND acknowledges support by the Berlin Mathematical School (BMS), MS and CS support by the DFG Research Center

MATHEON in Berlin.

7 Bibliography

References

- A. H. Al-Mohy and N. J. Higham. Computing the action of the matrix exponential, with an application to exponential integrators. *MIMS EPrint 2010.30*, The University of Manchester, to appear in *SIAM J*, 2010. 12
- R. Albert and A.L. Barabasi. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74(1):47–97, Jan 2002. doi: 10.1103/RevModPhys.74.47. 1
- D. Aldous and J. Fill. *Reversible Markov Chains and Random Walks on Graphs*. University of California, Berkeley, 2002. 3
- G.E. Cho and C.D. Meyer. Aggregation/disaggregation methods for nearly uncoupled Markov chains. *Technical Report NCSU no. 041600-0400*, North Carolina State University, 1999. 3
- Michael Dellnitz and Robert Preis. Congestion and almost invariant sets in dynamical systems. In *Proceedings of the 2nd international conference on Symbolic and numerical scientific computation*, SNSC'01, pages 183–209, Berlin, Heidelberg, 2003. Springer-Verlag. ISBN 3-540-40554-2. URL <http://dl.acm.org/citation.cfm?id=1763852.1763862>. 3
- Michael Dellnitz, Gary Froyland, and Oliver Junge. The algorithms behind GAIO - set oriented numerical methods for dynamical systems. In *In Ergodic theory, analysis, and efficient simulation of dynamical systems*, pages 145–174. Springer, 2000. 3
- P. Deuffhard and M. Weber. Robust Perron cluster analysis in conformation dynamics. *Linear Algebra and its Applications*, 398 Special issue on matrices and mathematical biology:161–184, 2005. 3, 4, 10
- P. Deuffhard, W. Huisinga, A. Fischer, and Ch. Schuette. Identification of almost invariant aggregates in reversible nearly uncoupled Markov chains. *Linear Algebra and its Applications*, 315:39–59, 2000. 9, 10, 12, 13
- N. Djurdjevac, M. Sarich, and Ch. Schuette. Estimating the eigenvalue error of Markov state models. *Multiscale Modeling & Simulation (Accepted)*, 2010a. 7, 9
- N. Djurdjevac, M. Sarich, and Ch. Schuette. On Markov state models for metastable processes. *Proceeding of the ICM 2010 as invited lecture*, 2010b. 9
- N. Djurdjevac, S. Bruckner, T. O. F. Conrad, and Ch. Schuette. Random walks on complex modular networks. *Journal of Numerical Analysis, Industrial and Applied Mathematics*, 2011. 9, 16
- Peter G Doyle and J Laurie Snell. *Fortune*, 94(January):1–120, 2000. 3

- W. E. and E. Vanden-Eijnden. Transition-path theory and path-finding algorithms for the study of rare events. *Annual Review of Physical Chemistry*, 61:391–420, 2010. 16
- P. Garrido and J. Marro, editors. *Exploring complex graphs by random walks*, volume 661, 2002. American Institute of Physics. 3
- M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, June 2002. 2
- Wilhelm Huisinga and Bernd Schmidt. Metastability and dominant eigenvalues of transfer operators. In Benedict Leimkuhler, Christophe Chipot, Ron Elber, Aatto Laaksonen, Alan Mark, Tamar Schlick, Christof Schuette, and Robert Skeel, editors, *New Algorithms for Macromolecular Simulation*, volume 49 of *Lecture Notes in Computational Science and Engineering*, chapter 11, pages 167–182. Springer-Verlag, 2006. 7
- H. Jeong, B. Tombor, R. Albert, Z. Oltvai, and A.L. Barabasi. The large-scale organization of metabolic networks. *Nature*, 407:651–654, 2000. 2
- S. Lafon and A.B. Lee. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1393–1403, 2006. 3
- T. Li, W E, and E Vanden Eijnden. Optimal partition and effective dynamics of complex networks. *Proc. Nat. Acad. Sci.*, 105:7907-7912, 2008. 3, 13
- T. Li, J. Liu, and W. E. A probabilistic framework for network partition. *Phys. Rev. E*, 80:026106, 2009. 4
- L. Lovasz. Random walks on graphs: A survey. *Bolyai Society Mathematical Studies*, 2, 1993. 3
- Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, December 2007. 2
- I. Marek and P. Mayer. Aggregation/disaggregation iterative methods applied to Leontev and Markov chain models. *Appl. Math.*, 47:131-156, 2001. 3
- R.B. Mattingly. A revised stochastic complementation algorithm for nearly completely decomposable Markov chains. *ORSA Journal on Computing*, 7(2): 117-124, 1995. 3
- E. Meerbach, Ch. Schuette, and A. Fischer. Eigenvalue bounds on restrictions of reversible nearly uncoupled Markov chains. *Lin. Alg. Appl.*, 398, 2005. 3
- M. Meila and J. Shi. A random walks view of spectral segmentation. *AI and Statistics (AISTATS)*, 2001. 3
- P. Metzner, Ch. Schuette, and E. Vanden-Eijnden. Transition path theory for Markov jump processes. *Multiscale Modeling and Simulation*, 7(3):1192–1219, 2009. 8, 16

- C. D. Meyer. Stochastic complementation, uncoupling Markov chains, and the theory of nearly reducible systems. *SIAM Rev*, pages 240–272, 1989. 3
- Mari C. V. Nascimento and Andr C. P. L. F. De Carvalho. Spectral methods for graph clustering: A survey. *European Journal Of Operational Research*, 211(2):221–231, 2011. 2
- M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45,167–256, 2003. 1
- M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *PHYS.REV.E*, 74:036104, 2006. 18, 19, 20, 21
- M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, Feb 2004. 2
- M.E.J. Newman, A.L. Barabasi, and D.J. Watts. *The Structure and Dynamics of Networks*. Princeton Univ Press, Princeton, NJ, 2006. 1, 2
- V. Nicosia, G. Mangioni, V. Carchiolo, and M. Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment*, 2009:22, 2008. 2
- J.D. Noh and H. Rieger. Random walks on complex networks. *Phys. Rev. Lett.*, 92, 2004. 3
- Gergely Palla, Imre Derenyi, Illes Farkas, and Tamas Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):1–10, 2005. 2
- Mason A Porter, Jukka-Pekka Onnela, and Peter J Mucha. Communities in networks. *World Wide Web Internet And Web Information Systems*, 56(9):1082–1097, 2009. 2
- Joerg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *PHYS.REV.E*, 74:016110, 2006. 2
- M. Rosvall and C.T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proc Natl Acad Sci*, 105:1118–1123, 2008. 3
- Fortunato Santo. Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174, 2010. ISSN 0370-1573. 2, 4
- M. Sarich and Ch. Schuette. Approximating selected non-dominant timescales by Markov state models. *Comm Math Sci (to appear)*, 2011. 7, 10
- M. Sarich, Ch. Schuette, and E. Vanden-Eijnden. Optimal fuzzy aggregation of networks. *Multiscale Modeling and Simulation*, 8(4):1535–1561, 2010. 4, 10
- Marco Sarich. *Projected Transfer Operators*. PhD thesis, FU Berlin, 2011. 9
- Ch. Schuette and W. Huisinga. Biomolecular conformations can be identified as metastable sets of molecular dynamics. In *Handbook of Numerical Analysis*, pages 699–744. Elsevier, 2003. 7

- Ch. Schuette, F. Noé, Jianfeng Lu, M. Sarich, and E. Vanden-Eijnden. State models based on milestoning. *J. Chem. Phys*, 2011. 9
- L. S. Schulman and B. Gaveau. Dynamical distance: coarse grains, pattern recognition, and network analysis. *Bull. Sci. math.*, 129:631–642, 2005. 3
- Stijn Van Dongen. *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, 2000. URL <http://igitur-archive.library.uu.nl/dissertations/1895620/inhoud.htm>.